7.3

Systems Management Application Programming

SC24-6327-73

Note:

Before you use this information and the product it supports, read the information in “Notices” on page 891.

This edition applies to version 7, release 3 of IBM® z/VM® (product number 5741-A09) and to all subsequent releases and modifications until otherwise indicated in new editions.

Last updated: 2025-06-16

© Copyright International Business Machines Corporation 2003, 2025.
US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

Figures xi

Tables xiii

About This Document xv
Intended Audience xv
Where to Find More Information xv
Links to Other Documents and Websites xv

How to provide feedback to IBM xvii

Summary of Changes for z/VM: Systems Management Application Programming xix
SC24-6268-73, z/VM 7.3 (June 2025) xix
SC24-6327-73, z/VM 7.3 (December 2023) xix
SC24-6327-73, z/VM 7.3 (June 2023) xix
SC24-6327-73, z/VM 7.3 (May 2023) xx
SC24-6327-73, z/VM 7.3 (September 2022) xx

Part 1. Overview 1

Chapter 1. Introduction 3
The Directory Manager 4
The Socket-based Server Environment 4
SMAPI Quick Start Guide 4

Chapter 2. API Functions Overview 7
ABEND Dump Management APIs 7
Authorization APIs 7
Directory Manager Control APIs 7
Directory Manager Local Tag and Scan APIs 7
Directory Parsing APIs 8
Directory Updates Subscription APIs 8
Disk Management APIs 8
Event Management APIs 9
Image Characteristics APIs 9
Image Connectivity APIs 10
Image CPUs APIs 12
Image Devices APIs 12
Image IPL Management APIs 13
Image Operations APIs 13
Image Volume Management APIs 14
List-Directed IPL APIs 14
Name List APIs 14
Network Interface Configuration APIs 15
Profile Management APIs 15
Prototype Management APIs 15
Response Recovery APIs 16
Server Management APIs 16
Shared Memory Management APIs 16

Single System Image (SSI) Cluster Management APIs....16
System Management APIs....17
Virtual Machine Reader Management APIs....17
VMRM Configuration Update APIs....17

Part 2. Installation and Configuration....19

Chapter 3. Defining the Servers....21
Request Servers....21
Worker Servers....22
LOHCOST....23
DTCSMAPI....24
PERSMAPI....24
OPERATNS....25

Chapter 4. Setting up and Configuring the Server Environment....27
Shared File System Directories....27
The Server Names File....27
Configuring SMAPI....30
SMAPI Configuration Properties....30
TCP/IP Requirements....36
Client Authentication....36
Configuring SMAPI to use an ESM to Authorize Requests....36
Authorizing API Requests....36
How Authorizing Requests Are Processed....38
Name Lists....39
Starting and Restarting the Server Environment....43
Stopping the Server Environment....43
Defining Additional Servers....44
Activating or Deactivating Servers....44

Part 3. User's Guide and Reference....47

Chapter 5. Programming Considerations....49
Sockets Overview....49
Data Types....49
Call Format....51

Chapter 6. Socket Application Programming Interfaces....55
Asynchronous_Notification_Disable_DM....56
Asynchronous_Notification_Enable_DM....60
Asynchronous_Notification_Query_DM....65
Authorization_List_Add....70
Authorization_List_Query....74
Authorization_List_Remove....79
Check_Authentication....82
Delete_ABEND_Dump....85
Directory_Manager_Local_Tag_Define_DM....88
Directory_Manager_Local_Tag_Delete_DM....91
Directory_Manager_Local_Tag_Query_DM....94
Directory_Manager_Local_Tag_Set_DM....97
Directory_Manager_Search_DM....101
Directory_Manager_Task_Cancel_DM....105
Event_Stream_Add....108
Event_Subscribe....111
Event_Unsubscribe....115
Image_Activate....117

Image_Active_Configuration_Query....121
Image_Console_Get....126
Image_CPU_Define....128
Image_CPU_Define_DM....131
Image_CPU_Delete....135
Image_CPU_Delete_DM....138
Image_CPU_Query....141
Image_CPU_Query_DM....145
Image_CPU_Set_Maximum_DM....149
Image_Create_DM....152
Image_Deactivate....157
Image_Definition_Async_Updates....161
Image_Definition_Create_DM....164
Image_Definition_Delete_DM....175
Image_Definition_Query_DM....182
Image_Definition_Update_DM....190
Image_Delete_DM....202
Image_Device_Dedicate....205
Image_Device_Dedicate_DM....208
Image_Device_Reset....211
Image_Device_Undedicate....214
Image_Device_Undedicate_DM....217
Image_Disk_Copy....220
Image_Disk_Copy_DM....223
Image_Disk_Create....229
Image_Disk_Create_DM....233
Image_Disk_Delete....240
Image_Disk_Delete_DM....243
Image_Disk_Query....246
Image_Disk_Share....250
Image_Disk_Share_DM....254
Image_Disk_Unshare....258
Image_Disk_Unshare_DM....261
Image_IPL_Characteristics_Define_DM....264
Image_IPL_Characteristics_Query_DM....273
Image_IPL_Delete_DM....276
Image_IPL_Query_DM....279
Image_IPL_Set_DM....282
Image_Lock_DM....285
Image_Lock_Query_DM....288
Image_MDISK_Link_Query....292
Image_Name_Query_DM....296
Image_Password_Set_DM....299
Image_Pause....302
Image_Query_Activate_Time....305
Image_Query_DM....308
Image_Recycle....311
Image_Replace_DM....315
Image_SCSI_Characteristics_Define_DM....318
Image_SCSI_Characteristics_Query_DM....322
Image_Status_Query....326
Image_Unlock_DM....329
Image_Volume_Add....332
Image_Volume_Delete....338
Image_Volume_Share....343
Image_Volume_Space_Define_DM....346
Image_Volume_Space_Define_Extended_DM....350
Image_Volume_Space_Query_DM....356

Image_Volume_Space_Query_Extended_DM ... 361
Image_Volume_Space_Remove_DM ... 366
Metadata_Delete ... 370
Metadata_Get ... 373
Metadata_Set ... 375
Metadata_Space_Query ... 379
Name_List_Add ... 382
Name_List_Destroy ... 385
Name_List_Query ... 388
Name_List_Remove ... 391
Network_IP_Interface_Create ... 394
Network_IP_Interface_Modify ... 401
Network_IP_Interface_Query ... 405
Network_IP_Interface_Remove ... 413
Page_or_Spool_Volume_Add ... 417
Process_ABEND_Dump ... 422
Profile_Create_DM ... 425
Profile_Delete_DM ... 428
Profile_Lock_DM ... 431
Profile_Lock_Query_DM ... 434
Profile_Query_DM ... 438
Profile_Replace_DM ... 441
Profile_Unlock_DM ... 444
Prototype_Create_DM ... 447
Prototype_Delete_DM ... 450
Prototype_Name_Query_DM ... 453
Prototype_Query_DM ... 456
Prototype_Replace_DM ... 459
Query_ABEND_Dump ... 462
Query_All_DM ... 466
Query_API_Functional_Level ... 471
Query_Asynchronous_Operation_DM ... 474
Query_Directory_Manager_Level_DM ... 477
Response_Recovery ... 480
Shared_Memory_Access_Add_DM ... 483
Shared_Memory_Access_Query_DM ... 487
Shared_Memory_Access_Remove_DM ... 491
Shared_Memory_Create ... 494
Shared_Memory_Delete ... 499
Shared_Memory_Query ... 502
Shared_Memory_Replace ... 507
SMAPI_Status_Capture ... 511
SSI_Query ... 515
Static_Image_Changes_Activate_DM ... 520
Static_Image_Changes_Deactivate_DM ... 523
Static_Image_Changes_Immediate_DM ... 526
System_Config_Syntax_Check ... 529
System_Compliance_Information_Query ... 533
System_Disk_Accessibility ... 535
System_Disk_Add ... 538
System_Disk_IO_Query ... 541
System_Disk_Query ... 546
System_EQID_Query ... 550
System_FCP_EQID_Set ... 554
System_FCP_Free_Query ... 558
System_Image_Performance_Query ... 562
System_Information_Query ... 565
System_Page_Utilization_Query ... 569

System_Performance_Information_Query ... 573
System_Performance_Threshold_Disable ... 580
System_Performance_Threshold_Enable ... 583
System_Processor_Query ... 586
System_RDR_File_Manage ... 589
System_RDR_File_Query ... 592
System_SCSI_Disk_Add ... 595
System_SCSI_Disk_Delete ... 599
System_SCSI_Disk_Query ... 602
System_Service_Query ... 606
System_Shutdown ... 610
System_Spool_Utilization_Query ... 614
System_WWPN_Query ... 618
Virtual_Channel_Connection_Create ... 622
Virtual_Channel_Connection_Create_DM ... 625
Virtual_Channel_Connection_Delete ... 628
Virtual_Channel_Connection_Delete_DM ... 631
Virtual_Network_Adapter_Connect_LAN ... 634
Virtual_Network_Adapter_Connect_LAN_DM ... 638
Virtual_Network_Adapter_Connect_Vswitch ... 642
Virtual_Network_Adapter_Connect_Vswitch_DM ... 645
Virtual_Network_Adapter_Connect_Vswitch_Extended ... 648
Virtual_Network_Adapter_Create ... 651
Virtual_Network_Adapter_Create_DM ... 654
Virtual_Network_Adapter_Create_Extended ... 658
Virtual_Network_Adapter_Create_Extended_DM ... 662
Virtual_Network_Adapter_Delete ... 666
Virtual_Network_Adapter_Delete_DM ... 669
Virtual_Network_Adapter_Disconnect ... 672
Virtual_Network_Adapter_Disconnect_DM ... 675
Virtual_Network_Adapter_Query ... 678
Virtual_Network_Adapter_Query_Extended ... 682
Virtual_Network_LAN_Access ... 690
Virtual_Network_LAN_Access_Query ... 693
Virtual_Network_LAN_Create ... 696
Virtual_Network_LAN_Delete ... 700
Virtual_Network_LAN_Query ... 703
Virtual_Network_OSA_Query ... 707
Virtual_Network_VLAN_Query_Stats ... 711
Virtual_Network_Vswitch_Create ... 716
Virtual_Network_Vswitch_Create_Extended ... 724
Virtual_Network_Vswitch_Delete ... 731
Virtual_Network_Vswitch_Delete_Extended ... 737
Virtual_Network_Vswitch_Query ... 740
Virtual_Network_Vswitch_Query_Byte_Stats ... 749
Virtual_Network_Vswitch_Query_Extended ... 755
Virtual_Network_Vswitch_Query_Stats ... 769
Virtual_Network_Vswitch_Set ... 774
Virtual_Network_Vswitch_Set_Extended ... 782
VMRELOCATE ... 791
VMRELOCATE_Image_Attributes ... 796
VMRELOCATE_Modify ... 800
VMRELOCATE_Status ... 803
VMRM_Configuration_Query ... 807
VMRM_Configuration_Update ... 810
VMRM_Measurement_Query ... 814

Chapter 7. Return and Reason Code Summary ... 819

All Return Codes (Including Internal)....819
Syntax Error Reason Codes (RC = 24)....833
Internal Return Codes (RC = 396, 592, or 596)....834
Return Code 396....834
Return Code 592, 596....842

Appendix A. The Directory Manager Exit....843
Directory Manager Exit Input Interface....843
Directory Manager Exit Output Interface....850

Appendix B. Creating Custom APIs....851
Designing the Custom API....851
Writing the Custom EXEC....852
Installing the Custom EXEC....852
Return and Reason Codes....853
Step-by-step Example....853
Example: Designing the API....853
Example: Writing the Custom Exec....854
Example: Installing the Custom Exec....855

Appendix C. ENROLL and GRANT Commands Performed Automatically During z/VM Installation....857

Appendix D. Sample Code....859
Sample C Program....859
Sample Java Program....869

Appendix E. Diagnosing Configuration Errors During Server Startup....873

Appendix F. Using SMAPI with an External Security Manager....877
Using SMAPI with RACF....877
Enabling RACROUTE....877
Making the SMAPI Service Machines Exempt From Certain Command Checking....878
Enabling SMAPI to Access DIAGNOSE X'88'....879
Enabling SMAPI to Access Needed Resources....879
Migrating to Using the ESM Policies for Authorizing APIs....880

Appendix G. Capturing SMAPI Data for Problem Resolution....883

Appendix H. Utilities and Common Procedures....885
DMSAPISD....886
DMSAPISL....887
DMSAPISP....888
SMCFGDM EXEC....889

Notices....891
Programming Interface Information....892
Trademarks....892
Terms and Conditions for Product Documentation....892
IBM Online Privacy Statement....893

Bibliography....895
Where to Get z/VM Information....895
z/VM Base Library....895
z/VM Facilities and Features....896

Prerequisite Products..... 898
Related Products..... 898

Index..... 899

Figures

1. VSMWORK1 Server Authorization File.....37

Tables

1. Fields in the DMSSISVR NAMES File....28

2. Input Keywords and Values for Image_Definition_Create_DM....165

3. MDISK= Keywords by Directory Manager Operation....172

4. Input Keywords and Values for Image_Definition_Delete_DM....176

5. Output Keywords and Values for Image_Definition_Query_DM....184

6. Input Keywords and Values for Image_Definition_Update_DM....191

7. MDISK= Keywords by Directory Manager Operation....198

8. Maximum Starting Location Allowed, by Allocation Unit....227

9. Maximum Starting Location and image_disk_size Value Allowed, by Allocation Unit....238

10. boot_data parameter examples....267

11. boot_record parameter examples....268

12. Input Keywords and Values for Query_All_DM....468

13. Output Keywords and Values for System_Disk_IO_Query....543

14. Output Keywords and Values for System_Information_Query....567

15. Input Keywords and Values for System_Performance_Information_Query....574

16. Output Keywords and Values for System_Performance_Information_Query....576

17. SEGTPRC DSECT field names corresponding to System_Performance_Information_Query DETAILED_CPU=output_subkeyword=value pairs....578

18. Input Keywords and Values for System_Service_Query....607

19. Output Keywords and Values for System_Service_Query....608

20. Output Keywords and Values for Virtual_Network_Adapter_Query_Extended....683

21. All Return Codes (Including Internal)....819

22. Internal Return Codes (RC = 396, 592, or 596)....834

23. CP Commands Used by Systems Management APIs.....834

24. Directory Manager Function-Specific Arguments..... 844

25. Configuration Errors, With Explanation and Affected Areas.....873

About This Document

This document contains socket-based application programming interfaces (APIs) to perform system management functions for virtual images (guests) in an IBM® z/VM® environment.

Intended Audience

This document is intended for systems programmers and applications programmers who will be writing programs to perform system management of virtual systems in a z/VM environment.

You should have experience with z/VM and z/VM guests. You should also have a working knowledge of programming with sockets.

Where to Find More Information

See "Bibliography" on page 895 at the back of this document.

Links to Other Documents and Websites

The PDF version of this document contains links to other documents and websites. A link from this document to another document works only when both documents are in the same directory or database, and a link to a website works only if you have access to the Internet. A document link is to a specific edition. If a new edition of a linked document has been published since the publication of this document, the linked document might not be the latest edition.

© Copyright IBM Corp. 2003, 2025

How to provide feedback to IBM

We welcome any feedback that you have, including comments on the clarity, accuracy, or completeness of the information. See How to send feedback to IBM for additional information.

© Copyright IBM Corp. 2003, 2025

Summary of Changes for z/VM: Systems Management Application Programming

This information includes terminology, maintenance, and editorial changes. Technical changes or additions to the text and illustrations for the current edition are indicated by a vertical line (**|**) to the left of the change.

SC24-6268-73, z/VM 7.3 (June 2025)

This edition includes changes to support product changes that are provided or announced after the general availability of z/VM 7.3.

[7.3 VM66822] SMAPI support for EQDIO

With the PTF for APAR VM66822, z/VM 7.3 provides support that allows users to develop or enhance system management tools that are related to EQDIO devices.

The following topics are updated:

- “Network_IP_Interface_Create” on page 394
- “Network_IP_Interface_Query” on page 405
- “Virtual_Network_Adapter_Query_Extended” on page 682
- “Virtual_Network_OSA_Query” on page 707
- “Virtual_Network_Vswitch_Create_Extended” on page 724
- “Virtual_Network_Vswitch_Query_Extended” on page 755
- “Virtual_Network_Vswitch_Set_Extended” on page 782

SC24-6327-73, z/VM 7.3 (December 2023)

This edition includes terminology, maintenance, and editorial changes.

SC24-6327-73, z/VM 7.3 (June 2023)

This edition includes changes to support product changes provided or announced after the general availability of z/VM 7.3.

Revision flags are left on for VM66646 (Security Settings and Compliance Interfaces), which was published in the previous edition.

[VM66424, VM66434, VM66650] Guest Secure IPL Support

With the PTF for APARs VM66424 (DirMaint), VM66434 (CP), and VM66650 (SMAPI), z/VM V7.3 supports guest secure IPL (load and dump) for both ECKD and SCSI devices. A z/VM user can request that the machine loader validate the signed IPL code by using the security keys that were previously loaded by the customer into the HMC certificate store. The validation ensures that the IPL code is intact, unaltered, and originates from a trusted build-time source.

Support is provided for the following guest operating systems:

- This support provides the ability for a Linux® guest to exploit hardware to validate the code being booted, helping to ensure it is signed by the client or its supplier.

  Linux on IBM zSystems instances that previously were able to perform secure boot first level on an IBM z15® or IBM LinuxONE III prior to Driver D41C Bundle S73a, or an IBM z16 or IBM LinuxONE 4 prior to

© Copyright IBM Corp. 2003, 2025

Driver D51C Bundle S18, will no longer be able to use secure boot until appropriate additional support is applied to the Linux image. Details are available about the required service level of Linux to properly IPL securely first or second level after driver D41C Bundle S73a or Driver D51C Bundle S18 has been applied. See 230428 Machine Alert for 8561, 8562, 3931, 3932 (https://www-40.ibm.com/servers/resourcelink/lib03020.nsf/pagesByDocid/272B3DD994A65B538525899F005FA0E6?OpenDocument).

- z/OS® is supported in audit mode only. Full exploitation requires Virtual Flash Memory support, which is not available to a guest. In audit mode, the IPL code is checked but the IPL continues even if the code is not valid.

z/VM and the z/VM stand-alone dump utility do not support performing host IPL via List-Directed IPL (LD-IPL) from ECKD. In addition, Secure IPL of the z/VM host and z/VM stand-alone dump are not supported.

The following new Systems Management API calls are added to define and query LOADDEV user directory statements:

- “Image_IPL_Characteristics_Define_DM” on page 264
- “Image_IPL_Characteristics_Query_DM” on page 273
-

The following topics are updated:

- “Call Format” on page 51
- “Image_IPL_Query_DM” on page 279
- “Image_IPL_Set_DM” on page 282
- “Image_SCSI_Characteristics_Define_DM” on page 318
- “Image_SCSI_Characteristics_Query_DM” on page 322
- “List-Directed IPL APIs” on page 14

Miscellaneous updates for June 2023

The following topic is updated:

- “Making the SMAPI Service Machines Exempt From Certain Command Checking” on page 878

SC24-6327-73, z/VM 7.3 (May 2023)

This edition includes changes to support product changes that are provided or announced after the general availability of z/VM 7.3.

[VM66646] Security Settings and Compliance Interfaces

With the PTF for APAR VM66646, z/VM 7.3 provides security settings and compliance API and command interfaces for compliance status extractors. The output from these new extractor interfaces contains security-relevant configuration data that can be analyzed for Payment Card Industry Data Security Standard (PCI DSS) compliance or for adherence to security configuration baselines. The API is provided through a new Systems Management interface that passes data to the extractor program.

This is the new Systems Management API:

- “System_Compliance_Information_Query” on page 533

SC24-6327-73, z/VM 7.3 (September 2022)

This edition supports the general availability of z/VM 7.3. Note that the publication number suffix (-73) indicates the z/VM release to which this edition applies.

Release-specific changes

z/VM 7.3 requires an architecture level set (ALS) from z/VM 7.2. “Query_API_Functional_Level” on page 471 has been updated.

SMAPI FCP EQID API Support

The following new Systems Management API has been added to enable modification of device equivalency IDs (EQID) for real FCP Adapter devices:

“System_FCP_EQID_Set” on page 554

This new API is particularly beneficial for customers who use SCSI disks connected via FCP (not edevices) and who desire to utilize Live Guest Relocation (LGR) in an SSI environment.

Other changes

The following changes are described in “SMAPI Configuration Properties” on page 30:

- Updated configuration parameter: LOHCOST Server Defaults
- New SMAPI configuration parameter: SFS Garbage Collection Periodic Check

Part 1. Overview

© Copyright IBM Corp. 2003, 2025

Chapter 1. Introduction

One of the major advantages of z/VM has always been its ability to provide each user with an individual working environment, a *virtual machine (virtual image)*. The virtual machine simulates a dedicated, real machine, including processor functions, memory, and input/output resources. A variety of operating systems and applications can run in a virtual machine. Managing a large number of guest operating systems (virtual images), however, requires a thorough understanding of VM concepts and the knowledge and skill to execute a complex set of commands.

The Systems Management APIs simplify the task of managing many virtual images running under a single z/VM image. A standard, platform-independent client interface reduces the amount of z/VM-specific programming skills required. The Systems Management APIs include a basic set of interfaces that can be used to:

- Create new virtual images in a variety of operating environments:
  - Linux on IBM Z®
  - z/OS
  - z/VM
  - z/VSE®
  - z/TPF
  - CMS
- Allocate and manage resources for virtual images
- Change a virtual image configuration
- Manage connectivity between virtual images
- Activate and deactivate:
  - Individual virtual images
  - Multiple virtual images
- Manage DASD volumes and groups
- Update Virtual Machine Resource Manager Service Virtual Machine (VMRM SVM) configuration files and query data without logging onto the VMRM server.
- Support the directory manager's ability to manage subscriptions to directory updates, with the ability to add new subscriptions, delete old subscriptions, and query existing subscriptions
- Support the directory manager's tag and scan functions
- Allow creation and updating of the LOADDEV directory statement for a virtual image, as well as queries of a virtual image's LOADDEV settings
- Query the time when a virtual image was activated.

Note the following when using SMAPI:

- VSMGUARD must always be used to start SMAPI, regardless of whether the system in running in a Unified Resource Manager environment.
- A Directory Manager license is not required. If a Directory Manager is not purchased and installed, a "SMAPI USE ONLY" instance of DirMaint will be installed and configured.
- A Performance Toolkit for z/VM license is not required. SMAPI will install and configure a "SMAPI USE ONLY" instance of the Performance Toolkit for z/VM to obtain performance data for use in provided SMAPI APIs.

In addition:

- LOHCOST, the SMAPI database server, is supported in all SMAPI installations.

© Copyright IBM Corp. 2003, 2025

The Directory Manager

Note that the Systems Management APIs require a directory manager. If the IBM z/VM Directory Maintenance Facility (DirMaint) is your directory manager, then Function Level 730 or later is required with the new socket-based environment. For more information on DirMaint, please consult the following publications:

- *z/VM: Directory Maintenance Facility Commands Reference*
- *z/VM: Directory Maintenance Facility Tailoring and Administration Guide*
- *z/VM: Directory Maintenance Facility Messages*

Note that if you have installed the full DirMaint product, you should review the configuration steps documented in "Appendix B. DirMaint Support for Systems Management APIs", in the *z/VM: Directory Maintenance Facility Tailoring and Administration Guide*.

If you are using a different directory manager, you'll need to replace the directory manager exit as defined in Chapter 3, “Defining the Servers,” on page 21. For more information on this exit, see Appendix A, “The Directory Manager Exit,” on page 843.

The Socket-based Server Environment

The socket-based server environment consists of one or more request servers and two or more worker servers. The request server listens for socket connections initiated by a client program. The server accepts the connection, receives the data, and then calls the appropriate worker server to process the request, while the client program waits for the response.

Three types of API functions are supported:

- IBM-supplied directory manager functions
- IBM-supplied non-directory manager functions
- Customer-defined functions.

See Chapter 3, “Defining the Servers,” on page 21 for more information.

SMAPI Quick Start Guide

Read this section if you want to get started with running SMAPI using the IBM default configuration. (For more information, see Chapter 4, “Setting up and Configuring the Server Environment,” on page 27.)

Starting SMAPI

During z/VM installation, you responded to the following prompt:

```
Would you like to have your system automatically configured to be managed by
a SMAPI client for system management? Enter Y or N _________
```

- If you entered a "Y", SMAPI is already running with the IBM default configuration.
- If you entered a "N", but you want to run SMAPI with the IBM default configuration, follow these steps to start SMAPI:
  - If you *do not* have the Directory Maintenance Facility (DirMaint) or another directory manager installed, issue the following command from the MAINT virtual machine to start SMAPI:

    ```
    XAUTOLOG VSMGUARD
    ```

  - If you *do* have the Directory Maintenance Facility (DirMaint) or another directory manager installed:

    1. Make sure MAINT is accessing its 193 disk and has the following DirMaint authorizations defined:

```
CMDSET ADGHMOPS CMDL 140A
CMDSET ADGHMOPS CMDL 150A
ALLOW_ASUSER_NOPASS
```

2. Start SMAPI by issuing the following commands from MAINT:

```
SMCFGDM
XAUTOLOG VSMGUARD
```

The SMCFGDM EXEC configures DirMaint as required by SMAPI. The EXEC tells you how many DATAMOVE servers that DirMaint can use to maximize SMAPI parallel execution. If you want to configure additional DATAMOVE machines for DirMaint, see *z/VM: Directory Maintenance Facility Tailoring and Administration Guide*. Note that even if you reconfigure DirMaint, SMAPI does not have to be restarted.

Configuring SMAPI to Communicate with Clients via SSL

After SSL is installed on your z/VM System, you can configuring SMAPI to communicate with clients via SSL. By default, SMAPI allows IPV4 communication via port 44444 and IPV6 communication via port 44445. You can use the PORT statement in your TCP/IP configuration to define these ports as secure ports. See *z/VM: TCP/IP Planning and Customization* for information about the PORT statement and about SSL in general. Once SSL is up and running with SMAPI's ports defined as secure, start SMAPI from MAINT by issuing the following commands:

```
SMCFGDM
XAUTOLOG VSMGUARD
```

Authorizing API Requests (Optional)

By default, SMAPI requests are authorized by the external security manager (ESM), if one is running. If an ESM is not installed, or if the ESM defers the request, SMAPI's native authorization mechanism is used. For more information, see “Configuring SMAPI to use an ESM to Authorize Requests” on page 36.

If you do not want to run SMAPI with the IBM default configuration, you can configure (almost) every aspect of SMAPI. Continue reading this document for information on configuring SMAPI.

Chapter 2. API Functions Overview

The various functions of the Systems Management APIs may be categorized as follows:

ABEND Dump Management APIs

These APIs are called for maintenance of ABEND dumps.

Delete_ABEND_Dump
Instruct the dump processing userid to remove a specified ABEND dump from the reader or from the dump processing location specified in the DMSSICNF COPY file.

Process_ABEND_Dump
Instruct the dump processing userid to process one or more ABEND dumps from its reader and place them in the dump processing location specified in the DMSSICNF COPY file.

Query_ABEND_Dump
Display the current ABEND dumps that appear in the OPERATNS userid's reader or have already been processed to the dump processing location specified in the DMSSICNF COPY file.

Authorization APIs

These APIs are called for maintenance of the systems management server authorization file.

Authorization_List_Add
Add an entry to the authorization file.

Authorization_List_Remove
Remove an entry from the authorization file.

Authorization_List_Query
Query the entries in the authorization file.

Directory Manager Control APIs

These APIs control the behavior of the directory manager.

Directory_Manager_Task_Cancel_DM
Cancel a specific asynchronous task being performed by the directory manager.

Query_Asynchronous_Operation_DM
Query the status of an asynchronous directory manager operation.

Query_Directory_Manager_Level_DM
Query the directory manager that is being used and its functional level.

Static_Image_Changes_Activate_DM
Enable changes to the source directory to be made available to virtual images.

Static_Image_Changes_Deactivate_DM
Prevent changes to the source directory from being made available to virtual images.

Static_Image_Changes_Immediate_DM
Make changes to the source directory immediately available to virtual images regardless of the current status of static image changes (active or inactive).

Directory Manager Local Tag and Scan APIs

These APIs manage tags in the directory and perform searches of the directory.

© Copyright IBM Corp. 2003, 2025

Directory_Manager_Local_Tag_Define_DM
Define a local tag or named comment record to contain installation-specific information about a virtual image.

Directory_Manager_Local_Tag_Delete_DM
Remove a local tag or named comment record from the directory manager's internal tables, so that users will no longer be able to set or query the tag.

Directory_Manager_Local_Tag_Query_DM
Obtain the value of a virtual image's local tag or named comment record.

Directory_Manager_Local_Tag_Set_DM
Set the value of a virtual image's local tag or named comment record.

Directory_Manager_Search_DM
Search the directory for records that match the specified pattern.

Directory Parsing APIs

These APIs parse directory statements.

Image_Definition_Async_Updates
Change the completion notification for Image_Definition_Update_DM, Image_Definition_Delete_DM, or Image_Definition_Create_DM.

Image_Definition_Create_DM
Create a new virtual machine directory entry for a particular system.

Image_Definition_Delete_DM
Remove a directory statement for a user or profile.

Image_Definition_Query_DM
Extract directory records and parse them into certain keywords.

Image_Definition_Update_DM
Update (replace) a directory statement for a user or profile – or create one if not found.

Metadata_Delete
Delete metadata values associated with a textual identifier (typically a directory entry name).

Metadata_Get
Obtain metadata values associated with a textual identifier (typically a directory entry name).

Metadata_Set
Set metadata values associated with a textual identifier (typically a directory entry name).

Metadata_Space_Query
Obtain information about metadata space used and available.

Query_All_DM
Obtain the contents of the entire system directory.

Directory Updates Subscription APIs

These APIs manage subscriptions to directory updates.

Asynchronous_Notification_Disable_DM
End notification of updates to specified entities as they occur.

Asynchronous_Notification_Enable_DM
Begin notification of updates to a specified entity as the updates occur.

Asynchronous_Notification_Query_DM
Query which users are subscribed to receive notification of updates to specified entities.

Disk Management APIs

These APIs manage disks.

Page_or_Spool_Volume_Add
Add a full volume page or spool disk to the system.

System_Disk_Accessibility
Verify that a device is available to be attached.

System_Disk_Add
Dynamically add an ECKD disk to a running z/VM system.

System_Disk_IO_Query
Obtain DASD read and write byte counts for SCSI EDEV and ECKD volumes owned by z/VM, and for which the control units have information.

System_Disk_Query
Query a real ECKD disk or all real ECKD disks.

System_EQID_Query
Obtain a list of the system devices assigned a device equivalency ID.

System_FCP_EQID_Set
Set the EQID (device equivalency ID) for a specific FCP device.

System_FCP_Free_Query
Query free FCP disk information.

System_SCSI_Disk_Add
Dynamically add a SCSI disk to a running z/VM system.

System_SCSI_Disk_Delete
Delete a real SCSI disk.

System_SCSI_Disk_Query
Query a real SCSI disk or all real SCSI disks.

System_WWPN_Query
Query all FCPs on a z/VM system and return a list of WWPNs.

Event Management APIs

These APIs manage system events.

Event_Stream_Add
Add an event to the event stream.

Event_Subscribe
Arrange to be asynchronously notified of events of interest.

Event_Unsubscribe
End asynchronous notification of events of interest.

System_Performance_Threshold_Disable
Disable thresholds for asynchronous event production.

System_Performance_Threshold_Enable
Enable thresholds for asynchronous event production.

Image Characteristics APIs

These APIs invoke the directory manager to define and modify virtual images and their characteristics.

Image_Create_DM
Define a new virtual image in the directory.

Image_Delete_DM
Delete a virtual image's definition from the directory.

Image_Lock_DM
Lock a virtual image’s directory entry or a specific device in a virtual image’s directory entry so that it cannot be changed.

Image_Lock_Query_DM
Query the status of directory manager locks in effect for a specific virtual image.

Image_Name_Query_DM
Obtain a list of defined virtual images.

Image_Password_Set_DM
Set or change a virtual image's password.

Image_Query_DM
Obtain a virtual image's directory entry.

Image_Replace_DM
Replace a virtual image's directory entry.

Image_Unlock_DM
Unlock a virtual image's directory entry or a specific device in a virtual image's directory entry so it can be changed.

Image Connectivity APIs

These APIs are called to establish and manage connectivity between virtual images. They may be used to:

- Change or query the configuration of an active virtual image, or
- Change the static configuration of a virtual image in the directory (these APIs end in "_DM").

New APIs are provided to extend support for connectivity between virtual images:

Virtual_Channel_Connection_Create
Establish a virtual network connection between two active virtual images. A virtual network connector (CTCA) is added to each virtual image's configuration if one is not already defined.

Virtual_Channel_Connection_Create_DM
Add a virtual network connection between two virtual images to their directory entries. A virtual network connector (CTCA) is added to each virtual image's directory entry if one is not already defined.

Virtual_Channel_Connection_Delete
Terminate a virtual network connection between two active virtual images and remove the virtual network connector (CTCA) from the virtual image's configuration.

Virtual_Channel_Connection_Delete_DM
Remove a virtual network connection from a virtual image's directory entry and remove the virtual network connector (CTCA) from the virtual image's directory entry.

Virtual_Network_Adapter_Connect_LAN
Connect an existing virtual network adapter on an active virtual image to an existing virtual network LAN.

Virtual_Network_Adapter_Connect_LAN_DM
Define a virtual network LAN connection for an existing virtual network adapter in a virtual image's directory entry.

Virtual_Network_Adapter_Connect_Vswitch
Connect an existing virtual network adapter on an active virtual image to an existing virtual switch.

Virtual_Network_Adapter_Connect_Vswitch_DM
Define a virtual switch connection for an existing virtual network adapter in a virtual image's directory entry.

Virtual_Network_Adapter_Connect_Vswitch_Extended
Connect an existing virtual network adapter on an active virtual image to an existing virtual switch (extended version of Virtual_Network_Adapter_Connect_Vswitch).

Virtual_Network_Adapter_Create
Add a virtual network interface card (NIC) to an active virtual image.

Virtual_Network_Adapter_Create_DM
Add a virtual network interface card (NIC) to a virtual image's directory entry.

Virtual_Network_Adapter_Create_Extended
Add a virtual network interface card (NIC) to an active virtual image (extended version of Virtual_Network_Adapter_Create).

Virtual_Network_Adapter_Create_Extended_DM
Add a virtual network interface card (NIC) to a virtual image's directory entry (extended version of Virtual_Network_Adapter_Create_DM).

Virtual_Network_Adapter_Delete
Remove a virtual network interface card (NIC) from an active virtual image.

Virtual_Network_Adapter_Delete_DM
Remove a virtual network interface card (NIC) from a virtual image's directory entry.

Virtual_Network_Adapter_Disconnect
Disconnect a virtual network adapter on an active virtual image from a virtual network LAN or virtual switch.

Virtual_Network_Adapter_Disconnect_DM
Remove a virtual network LAN or virtual switch connection from a virtual network adapter definition in a virtual image's directory entry.

Virtual_Network_Adapter_Query
Obtain information about the specified adapter for an active virtual image.

Virtual_Network_Adapter_Query_Extended
Obtain information about the specified adapter for an active virtual image (extended version of Virtual_Network_Adapter_Query).

Virtual_Network_LAN_Access
Grant users access to a restricted virtual network LAN.

Virtual_Network_LAN_Access_Query
Query which users are authorized to access a specified restricted virtual network LAN.

Virtual_Network_LAN_Create
Create a virtual network LAN.

Virtual_Network_LAN_Delete
Delete a virtual network LAN.

Virtual_Network_LAN_Query
Obtain information about a virtual network LAN.

Virtual_Network_OSA_Query
Query data about real OSA devices.

Virtual_Network_VLAN_Query_Stats
Query a virtual LAN's statistics.

Virtual_Network_Vswitch_Create
Create a virtual switch.

Virtual_Network_Vswitch_Create_Extended
Create a virtual switch (extended version of Virtual_Network_Vswitch_Create).

Virtual_Network_Vswitch_Delete
Delete a virtual switch.

Virtual_Network_Vswitch_Delete_Extended
Delete a virtual switch (extended version of Virtual_Network_Vswitch_Delete).

Virtual_Network_VSwitch_Query
Obtain information about the specified virtual switch or switches.

Virtual_Network_Vswitch_Query_Extended
Obtain information about the specified virtual switch or switches (extended version of Virtual_Network_Vswitch_Query).

Virtual_Network_Vswitch_Query_Stats
Query a virtual switch's statistics.

Virtual_Network_Vswitch_Set
Change the configuration of an existing virtual switch.

Virtual_Network_Vswitch_Set_Extended
Change the configuration of an existing virtual switch (extended version of Virtual_Network_Vswitch_Set).

Image CPUs APIs

These APIs manage virtual processors used by virtual images.

Image_CPU_Define
Add a virtual processor to an active virtual image's configuration.

Image_CPU_Define_DM
Add a virtual processor to a virtual image's directory entry.

Image_CPU_Delete
Delete a virtual processor from an active virtual image's configuration.

Image_CPU_Delete_DM
Delete a virtual processor from a virtual image's directory entry.

Image_CPU_Query
Query the virtual processors in an active virtual image's configuration.

Image_CPU_Query_DM
Query a virtual processor in a virtual image's directory entry.

Image_CPU_Set_Maximum_DM
Set the maximum number of virtual processors that can be defined in a virtual image's directory entry.

Image Devices APIs

These APIs manage devices used by virtual images. They may be used to:

- Change the configuration of an active virtual image, or
- Change the static configuration of a virtual image in the directory (these APIs end in "_DM").

Image_Device_Dedicate
Add a dedicated device to an active virtual image's configuration.

Image_Device_Dedicate_DM
Add a dedicated device to a virtual image's directory entry.

Image_Device_Reset
Clear all pending interrupts from the specified virtual device.

Image_Device_Undedicate
Delete a dedicated device from an active virtual image's configuration.

Image_Device_Undedicate_DM
Delete a dedicated device from a virtual image's directory entry.

Image_Disk_Copy
Clone a disk in an active virtual image's configuration.

Image_Disk_Copy_DM
Clone a disk in a virtual image's directory entry.

Image_Disk_Create
Add a disk that is defined in a virtual image's directory entry to that virtual image's active configuration.

Image_Disk_Create_DM
Add a disk to a virtual image's directory entry.

Image_Disk_Delete
Delete a disk from an active virtual image's configuration.

Image_Disk_Delete_DM
Delete a disk from a virtual image's directory entry.

Image_Disk_Query
Display the status of all DASDs accessible to a virtual image, including temporary disks and virtual disks in storage.

Image_Disk_Share
Add a disk that is defined in a virtual image's directory entry to a different active virtual image's configuration.

Image_Disk_Share_DM
Add a disk that is defined in a virtual image's directory entry to a different virtual image's directory entry.

Image_Disk_Unshare
Delete a shared disk from an active virtual image's configuration.

Image_Disk_Unshare_DM
Delete a shared disk from a virtual image's directory entry.

Image_MDISK_Link_Query
Query the links to an image's MDISK.

Image IPL Management APIs

These APIs manage the named saved system or device number that CP automatically loads (IPLs) when a virtual image is activated.

Image_IPL_Delete_DM
Delete the IPL statement from a virtual image's directory entry or a profile directory entry.

Image_IPL_Query_DM
Query the information about the operating system, or device containing the operating system, that is specified on the IPL statement in a virtual image's directory entry or a profile directory entry.

Image_IPL_Set_DM
Add an IPL statement to a virtual image's directory entry or a profile directory entry.

Image Operations APIs

These are operational APIs that can be requested for virtual images.

Image_Activate
Activate a virtual image or list of virtual images.

Image_Active_Configuration_Query
Obtain current configuration information for an active virtual image.

Image_Deactivate
Stop a virtual image or list of virtual images.

Image_Pause
Pause a running virtual image or restart a paused virtual image.

Image_Query_Activate_Time
Obtain the date and time when a virtual image was activated.

Image_Recycle
Deactivate and then reactivate a virtual image or list of virtual images.

Image_Status_Query
Determine whether virtual images are active (logged on or logged on disconnected) or inactive.

System_Image_Performance_Query
Obtain performance data for a virtual image.

Image Volume Management APIs

These APIs manage DASD volumes.

Image_Volume_Add
Add a DASD volume to be used by virtual images to the z/VM system configuration file.

Image_Volume_Delete
Delete a DASD volume definition from the z/VM system configuration file.

Image_Volume_Share
Indicate a full-pack minidisk is to be shared by the users of many real and virtual systems.

Image_Volume_Space_Define_DM
Define space on a DASD volume to be allocated by the directory manager for use by virtual images.

Image_Volume_Space_Define_Extended_DM
Define space on a DASD volume to be allocated by the directory manager for use by virtual images (extended version of Image_Volume_Space_Define_DM).

Image_Volume_Space_Query_DM
Query how space on a DASD volume is allocated by the directory manager.

Image_Volume_Space_Query_Extended_DM
Query how space on a DASD volume is allocated by the directory manager (extended version of Image_Volume_Space_Query_DM).

Image_Volume_Space_Remove_DM
Remove the directory manager's space allocations from a DASD volume.

List-Directed IPL APIs

These APIs create, update, and query the LOADDEV directory statement for a virtual image.

Image_IPL_Characteristics_Define_DM
Define or change the location of a program to be loaded as a result of a list-directed IPL, and the data to be passed to the loaded program, in a virtual image's directory entry.

Parameters for a secure IPL (*fcp_vdev* and SECURE) are supported.

Image_IPL_Characteristics_Query_DM
Query the location of a program to be loaded as a result of a list-directed IPL, and the data to be passed to the loaded program, in a virtual image's directory entry.

Parameters for a secure IPL (*fcp_vdev* and SECURE) are supported.

Image_SCSI_Characteristics_Define_DM
Define or change the location of a program to be loaded as a result of an FCP list-directed IPL, and the data to be passed to the loaded program, in a virtual image's directory entry.

Parameters for a secure IPL (*fcp_vdev* and SECURE) are not supported.

Image_SCSI_Characteristics_Query_DM
Query the location of a program to be loaded as a result of an FCP list-directed IPL, and the data to be passed to the loaded program, from a virtual image's directory entry.

Parameters for a secure IPL (*fcp_vdev* and SECURE) are not supported.

Name List APIs

These APIs help manage lists of names in the systems management server name list file. Names in lists may include virtual images or functions.

Name_List_Add
Add a name to a list in the name list file. If the list that is specified in target_identifier does not exist, a new list will be created.

Name_List_Destroy
Delete a list from the name list file.

Name_List_Query
Query the names that are in a list in the name list file.

Name_List_Remove
Delete a name from a list in the name list file. If there are no names remaining in the list, the list is also deleted.

Network Interface Configuration APIs

These APIs manage the network interface configuration for the z/VM TCP/IP stack.

Network_IP_Interface_Create
Create the initial network interface configuration for the z/VM TCP/IP stack.

Network_IP_Interface_Modify
Change the configuration of the existing network interface.

Network_IP_Interface_Query
Obtain interface configurations for a specified TCP/IP stack virtual machine.

Network_IP_Interface_Remove
Remove the existing network interface.

Profile Management APIs

These APIs manage profile directory entries.

Profile_Create_DM
Create a profile directory entry to be included in the definition of a virtual image in the directory.

Profile_Delete_DM
Delete a profile directory entry.

Profile_Lock_DM
Lock a profile directory entry so that it cannot be changed.

Profile_Lock_Query_DM
Query the status of directory manager locks in effect for a specific profile.

Profile_Query_DM
Query a profile directory entry.

Profile_Replace_DM
Replace the definition of a profile to be included in a virtual image in the directory.

Profile_Unlock_DM
Unlock a profile directory entry so it can be changed.

Prototype Management APIs

These APIs manage virtual image prototype definitions, which the directory manager uses to create new images.

Prototype_Create_DM
Create a new virtual image prototype.

Prototype_Delete_DM
Delete an image prototype.

Prototype_Name_Query_DM
Obtain a list of names of defined prototypes.

Prototype_Query_DM
Query the characteristics of an image prototype.

Prototype_Replace_DM
Replace an existing prototype.

Response Recovery APIs

This API performs response recovery.

Response_Recovery
Obtain response data from previous calls that may have failed.

Server Management APIs

These APIs validate a userid/password pair, query the support level of the API server and functions, and capture data to assist with identification and resolution of a problem with the SMAPI servers.

Check_Authentication
Validate a userid/password pair.

Query_API_Functional_Level
Obtain the support level of the server and functions.

SMAPI_Status_Capture
Capture data to assist with identification and resolution of a problem with the SMAPI servers. (Note that you can use the stand-alone SMSTATUS EXEC to perform this same function when SMAPI_Status_Capture cannot be executed because SMAPI is not responsive.)

Shared Memory Management APIs

These APIs manage shared memory on your virtual images through z/VM's shared physical segment functions.

Shared_Memory_Access_Add_DM
Add restricted (RSTD) access to a shared memory segment.

Shared_Memory_Access_Query_DM
Query the restricted (RSTD) access to a shared memory segment.

Shared_Memory_Access_Remove_DM
Remove restricted (RSTD) access from a shared memory segment.

Shared_Memory_Create
Create a memory segment that can be shared among virtual images.

Shared_Memory_Delete
Delete a shared memory segment.

Shared_Memory_Query
Query information about system data files that are contained in the saved memory segment.

Shared_Memory_Replace
Replace a shared memory segment previously defined by Shared_Memory_Create.

Single System Image (SSI) Cluster Management APIs

These APIs manage relocations of virtual machines within a z/VM SSI cluster.

SSI_Query
Obtain SSI and system status.

VMRELOCATE
Relocate, test relocation eligibility, or cancel the relocation of a virtual machine within a z/VM SSI cluster.

VMRELOCATE_Image_Attributes
Modify the relocation setting for a specified image.

VMRELOCATE_Modify
Modify the time limits associated with a relocation already in progress for the specified image.

VMRELOCATE_Status
Obtain information about virtual machine relocations currently in progress.

System Management APIs

These APIs query and check various aspects of overall system information, and the System_Shutdown API stops all system function.

System_Config_Syntax_Check
Check the syntax of a system configuration file located on a system parm disk.

System_Information_Query
Obtain information about a CP instance, including time, storage, system levels, IPL time, system generation time, language, CPU ID, and CPU capability information, and more.

System_Page_Utilization_Query
Obtain information about the z/VM paging space defined on the system.

System_Performance_Information_Query
Gather hypervisor performance data, including available/used, processor number, total processor percentages, and optional detailed CPU information for all visible LPARs on the CEC, and query, set and stop the monitor rate and interval values.

System_Processor_Query
List all processors accessible to the system and indicate the way in which each processor is being used. This API also returns the mode of the logical partition.

System_Service_Query
Query the status of an APAR, PTF, or RSU for a zVM component.

System_Shutdown
Systematically end all system function.

System_Spool_Utilization_Query
Obtain information about the z/VM spool space defined on the system.

Virtual Machine Reader Management APIs

These are APIs that can be requested for virtual machine reader operation.

System_RDR_File_Manage
Manage the reader files of the target virtual machine.

System_RDR_File_Query
Query the reader files of the target virtual machine.

VMRM Configuration Update APIs

These APIs update VMRM configuration files and query VMRM data without logging onto the VMRM Server Virtual Machine.

VMRM_Configuration_Query
Query the contents of the VMRM configuration file.

VMRM_Configuration_Update
Add, delete, and change VMRM configuration file statements.

VMRM_Measurement_Query
Obtain current VMRM measurement values.

Part 2. Installation and Configuration

© Copyright IBM Corp. 2003, 2025

Chapter 3. Defining the Servers

There are two types of SMAPI servers:

- Request servers
- Worker servers

Request Servers

A listening request server completes a connection with a client, and then accepts requests from that client. Specifically, there are:

- One or more INET/INET6 servers, which use either AF_INET (IPv4) or AF_INET6 (IPv6) family sockets to connect with clients
- One or more IUCV servers, which use AF_IUCV family sockets to connect with clients
- One AF_EVNT server, used to listen for and then propagate *VMEVENT and directory updates.

Note that there can be more than one AF_INET/AF_INET6 request server and more than one AF_IUCV request server.

These servers are defined as separate virtual machines in the default z/VM installation. The following is the recommended directory entry for each request server.

```
IDENTITY name password 128M 512M G
   BUILD ON MEMBn USING SUBCONFIG subname-n
         ⋮
   IPL CMS PARM AUTOCR
   OPTION  DIAG88
   MACHINE ESA
   IUCV auth MSGLIMIT 255
   IUCV *VMEVENT   (See note 2)
   IUCV *LOGREC    (See note 2)
   NAMESAVE VSMDCSS
   CONSOLE 0009 3215 T
   SPOOL 000C 2540 READER *
   SPOOL 000D 2540 PUNCH  A
   SPOOL 000E 1403 A

SUBCONFIG subname-n
   LINK  MAINT 190 190 RR
   LINK  MAINT 19E 19E RR
   LINK  MAINT 193 193 RR
   LINK  TCPMAINT  591 591 RR
   LINK  TCPMAINT  592 592 RR
   MDISK 191 3390 strt 025 label MR READ WRITE MULTIPLE
         ⋮
```

where *name*, *subname*, and *auth* are:

- VSMREQIN, VSMRQN, and ANY for the AF_INET request server
- VSMREQI6, VSMRQ6, and ANY for the AF_INET6 request server
- VSMREQIU, VSMRQU, and ALLOW for the AF_IUCV request server(s)
- VSMEVSRV, VSMEVS, and ANY for the AF_EVNT request server

and where *n* is the member number in a SSI cluster. (If there is only one member, or if the system is not a member of an SSI, use *n*=1 and replace 'MEMB*n*' with '*'.)

Note:

1. Change the MDISK statement to reflect the information as appropriate to your specific 191 disk.
2. The lines IUCV *VMEVENT and IUCV *LOGREC are required only for the AF_EVNT request server.
3. Keep in mind that neither request servers nor worker servers can run with multiple CPUs defined.

© Copyright IBM Corp. 2003, 2025

4. A sample profile exec for the request servers is provided in file VSMREQIN SAMPPROF on MAINT's 193 disk. At installation, the sample profile is copied to each request server's 191 disk as PROFILE EXEC.
5. If you are applying service updates to an existing system, you may currently have less than 128M defined in your USER *name* *name* statement. IBM recommends that you increase this amount to at least 128M. (Note that 512M is the maximum allowed.)
6. You must specify a NAMESAVE VSMDCSS entry. The server will not create one automatically.

Worker Servers

The worker servers process API function requests. Three worker servers are defined in the default installation – VSMWORK1, VSMWORK2, and VSMWORK3. A fourth worker server, VSMGUARD, is also defined. VSMGUARD is a "guard" server which helps provide better resiliency and error recovery.

There are two types of API calls: “short call” and “long call.” The first worker server, VSMWORK1, is always the “short call” worker. All other worker servers are designated as “long call” workers. These workers handle API requests that require more time than the “short call” requests. When more than one “long call” server is active, a worker server that is not busy will receive the request. If all worker servers are busy, the request will be queued so that it will be picked up by the first free long call server.

Note:

1. There must always be at least one short call worker server and at least one long call worker server, but a total of four (one short call and two long call, plus the VSMGUARD worker server) is the recommended minimum.
2. The VSMGUARD worker server does *not* process any requests.
3. The VSMGUARD worker server will grant authority to all the other SMAPI servers that are configured to access the SMAPI file space. Therefore, VSMGUARD must be made an administrator of the VMSYS: file pool. This is done by adding VSMGUARD to the list of users authorized for ADMIN authority. In the default environment, this is done by updating the VMSERVS DMSPARMS file on the VMSERVS 191 disk.

The following is the recommended directory entry for the worker servers (including VSMGUARD). Because the worker servers process requests that require various privileges, the worker servers must have all of the IBM-defined privilege classes (A through G).

```
IDENTITY name AUTOONLY 128M 512M ABCDEFG
   BUILD ON MEMBn USING SUBCONFIG subname-n
         ⋮
   IPL CMS PARM AUTOCR
   OPTION MAINTCCW LNKS LNKE DIAG88 LNKNOPAS
   MACHINE ESA
   IUCV ANY MSGLIMIT 255
   NAMESAVE VSMDCSS
   NAMESAVE SMAPIOUT
   CONSOLE 0009 3215 T
   SPOOL 000C 2540 READER *
   SPOOL 000D 2540 PUNCH  A
   SPOOL 000E 1403 A

   SUBCONFIG subname-n
   LINK  MAINT 190 190 RR
   LINK  MAINT 19E 19E RR
   LINK  MAINT 193 193 RR
   LINK  PMAINT 551 551 RR
   LINK  PMAINT CF0 CF0 MD
   LINK  TCPMAINT  591 591 RR
   LINK  TCPMAINT  592 592 RR
   MDISK 191 3390 strt 025 label MR READ WRITE MULTIPLE
   MDISK A91 3390 strt 005 label MR ALL ALL ALL
         ⋮
```

where *name* and *subname* are VSMWORK1 and VSMWK1, VSMWORK2 and VSMWK2, or VSMWORK3 and VSMWK3 (assuming you're using three worker servers, as per the default installation), plus VSMGUARD,

and where *n* is the member number in a SSI cluster. (If there is only one member, or if the system is not a member of an SSI, use *n*=1 and replace 'MEMB*n*' with '*'.)

Note:

1. Keep in mind that neither request servers nor worker servers can run with multiple CPUs defined.
2. Just as for the request servers, a sample profile exec for the worker servers is provided in file VSMWORK1 SAMPPROF on MAINT's 193 disk. At installation, the sample profile is copied to each worker server's 191 disk as PROFILE EXEC.
3. If you are applying service updates to an existing system, you may currently have less than 128M defined in your USER *name* *name* statement. IBM recommends that you increase this amount to at least 128M. (Note that 512M is the maximum allowed.)
4. The SMAPI servers can be defined to have up to 512M of virtual storage. If SMAPI is expected to handle a particularly heavy load, defining the SMAPI servers to have their maximum virtual storage is recommended.
5. The worker servers need write access to the service directories in the VMPSFS and VMSYS filepools. Therefore, the worker servers must be made administrators of the VMSYS: and VMPSFS: file pools, by adding VSMGUARD, VSMWORK1, VSMWORK2 and VMSWORK3 to the list of users authorized for ADMIN authority. In the default environment, this is done by updating the VMSERVS DMSPARMS file on the VMSERVS 191 disk and the VMSERVP DMSPARMS file on the VMSERVP 191 disk.
6. You must specify both a `NAMESAVE VSMDCSS` and a `NAMESAVE SMAPIOUT` entry. The server will not create these automatically.

LOHCOST

The LOHCOST server is used for caching the system directory contents required to satisfy the various query APIs (see note 3 below). It is also used to store and retrieve data used by the metadata APIs. The following is the required directory entry for the LOHCOST server:

```
IDENTITY LOHCOST AUTOONLY 768M 2G G
   BUILD ON MEMBn USING SUBCONFIG LOHCOS-n
         ⋮
   COMMAND DEFINE NIC F000 TYPE QDIO
   COMMAND COUPLE F000 TO SYSTEM DTCSMAPI
   MACH ESA 2
   OPTION LXAPP LANG AMENG DEVINFO DEVMAINT LNKNOPAS DIAG88
   NAMESAVE VSMDCSS
   CONSOLE 009 3215 T
   SPOOL 000C 2540 READER *
   SPOOL 000D 2540 PUNCH A
   SPOOL 000E 1403 A

SUBCONFIG LOHCOS-n
   IPL 190 PARM AUTOCR
   LINK MAINT 0190 0190 RR
   LINK MAINT 0193 0193 RR
   LINK MAINT 019D 019D RR
   LINK MAINT 019E 019E RR
   LINK MAINT 0400 0400 RR
   MDISK 191 3390 strt 010 label MR READ WRITE MULTIPLE
   MDISK 197 3390 strt 150 label MR READ WRITE MULTIPLE
         ⋮
```

where *n* is the member number in a SSI cluster. (If there is only one member, or if the system is not a member of an SSI, use *n*=1 and replace 'MEMB*n*' with '*'.)

Note:

1. Change the MDISK statement to reflect the information as appropriate to your specific 191 and 197 disks.
2. The Directory Manager must be enabled to receive asynchronous update notifications. If DirMaint is being used as the Directory Manager, follow the instructions to enable TCP notification as documented in the "Enabling the Asynchronous Update Notification Exit" section of "Appendix B. DirMaint Support for Systems Management APIs" in the *z/VM: Directory Maintenance Facility Tailoring and Administration Guide*.
3. If enabled, LOHCOST will be used to satisfy the following APIs:

- Image_Definition_Query_DM
- Image_Query_DM
- Image_Volume_Space_Query_DM
- Image_Volume_Space_Query_Extended_DM
- Metadata_Delete
- Metadata_Get
- Metadata_Set
- Metadata_Space_Query
- Profile_Query_DM

DTCSMAPI

The following is the required directory entry for the DTCSMAPI server:

```
IDENTITY DTCSMAPI AUTOONLY 32M 128M BG
   INCLUDE TCPCMSU
   BUILD ON MEMBn USING SUBCONFIG DTCSMA-n
         ⋮
   IPL CMS PARM AUTOCR
   OPTION QUICKDSP SVMSTAT MAXCONN 1024 DIAG98 APPLMON DIAG88
   SHARE RELATIVE 3000
   IUCV ANY PRIORITY
   IUCV ALLOW
   NAMESAVE VSMDCSS

SUBCONFIG DTCSMA-n
   LINK TCPMAINT 491 491 RR
   LINK TCPMAINT 492 492 RR
   LINK TCPMAINT 591 591 RR
   LINK TCPMAINT 592 592 RR
   LINK TCPMAINT 198 198 RR
   LINK MAINT    193 193 RR
   MDISK 191 3390 strt 005 label MR READ WRITE MULTIPLE
         ⋮
```

where *n* is the member number in a SSI cluster. (If there is only one member, or if the system is not a member of an SSI, use *n*=1 and replace 'MEMB*n*' with '*'.)

Note: Change the MDISK statement to reflect the information as appropriate to your specific 191 disk.

PERSMAPI

The PERSMAPI server is used for performance monitoring. The following is the required directory entry for the PERSMAPI server:

```
IDENTITY PERSMAPI AUTOONLY 128M 512M ABDEG
   BUILD ON MEMBn USING SUBCONFIG PERSMA-n
         ⋮
   MACHINE ESA
   XAUTOLOG AUTOLOG1
   ACCOUNT xxxx
   NAMESAVE MONDCSS
   NAMESAVE VSMDCSS
   IUCV *MONITOR MSGLIMIT 255
   IUCV ALLOW
   SHARE ABS 3%
   IPL ZCMS PARM AUTOCR
   OPTION QUICKDSP DIAG88
   CONSOLE 0009 3215
   SPOOL 000C 2540 READER *
   SPOOL 000D 2540 PUNCH A
   SPOOL 000E 1403 A

SUBCONFIG PERSMA-n
   LINK MAINT 190 190 RR
   LINK MAINT 19D 19D RR
   LINK MAINT 19E 19E RR
   LINK MAINT 193 193 RR
```

```
   LINK PERFSVM 200 200 RR
   LINK PERFSVM 29D 29D RR
   LINK PERFSVM 201 201 RR
   LINK PERFSVM 1CC 1CC RR
   MDISK 191 3390 strt 120 label MR READ WRITE MULTIPLE
   MDISK 195 3390 strt 060 label MR READ WRITE MULTIPLE
         ⋮
```

where *n* is the member number in a SSI cluster. (If there is only one member, or if the system is not a member of an SSI, use *n*=1 and replace 'MEMB*n*' with '*'.)

Note: Change the MDISK statement to reflect the information as appropriate to your specific 191 and 195 disks.

OPERATNS

The OPERATNS server is used collect, format, and distribute ABEND dumps. The following is the required directory entry for the OPERATNS server:

```
IDENTITY OPERATNS password 128M 128M BCEG
   INCLUDE IBMDFLT
   BUILD ON MEMBn USING SUBCONFIG OPRATN-n
         ⋮
   AUTOLOG AUTOLOG1 OP1 MAINT
   ACCOUNT 1 OPERATNS
   MACH ESA
   IPL 190
   OPTION DIAG88
   NAMESAVE VSMDCSS

SUBCONFIG OPRATN-n
   LINK MAINT 193 193 RR
   MDISK 191 3390 strt 100 label MR RDVF WDVF MDVF
         ⋮
```

where *n* is the member number in a SSI cluster. (If there is only one member, or if the system is not a member of an SSI, use *n*=1 and replace 'MEMB*n*' with '*'.)

Note: Change the MDISK statement to reflect the information as appropriate to your specific 191 disk.

Important: To activate automated dump handling, you must first uncomment the entry for the OPERATNS server in the DMSSISVR NAMES file (see “The Server Names File” on page 27) and also allocate enough space to contain the dump files. You specify the location of this space, as well as the interval at which the OPERATNS server will check its reader for new dump files to process automatically, using the `Dump_Processing_Location =` and `Dump_Processing_Interval =` entries in the DMSSICNF COPY file (see “Configuring SMAPI” on page 30).

A sample profile exec for the OPERATNS server is provided in file OPERATNS SAMPPROF on MAINT's 193 disk. This sample profile must be copied to each OPERATNS's 191 disk as PROFILE EXEC in order to complete activation of dump handling.

Chapter 4. Setting up and Configuring the Server Environment

The following topics are covered in this chapter:

- Shared File System Directories
- The Server Names File
- The Server Configuration File
- TCP/IP Requirements
- Client Authentication
- Authorizing API Requests
- Name Lists
- Starting the Server Environment
- Stopping the Server Environment
- Defining Additional Servers
- Activating or Deactivating Servers

Shared File System Directories

The request servers and worker servers use Shared File System (SFS) directories to access configuration files and other data. In the default installation, these directories are in the z/VM default filepool (VMSYS). The default directories are owned by the “short call” worker server (VSMWORK1).

Files that are required by the request and worker servers are copied to the SFS directories as part of the default z/VM installation. Access to the directories by the request and worker servers is also set up during server startup.

For details on the names of the SFS directories and how they may be configured, see “Configuring SMAPI” on page 30.

For more information on increasing the size of the VMSYS: filepool, if necessary, see *z/VM: CMS File Pool Planning, Administration, and Operation*.

For the list of GRANT and ENROLL commands that are automatically issued during the normal z/VM installation process, see Appendix C, “ENROLL and GRANT Commands Performed Automatically During z/VM Installation,” on page 857.

The Server Names File

The DMSSISVR NAMES file is a CMS NAMES file that determines how each specific request and worker server will function in the overall server environment. It is located on MAINT's 193 disk by default. The file consists of comments and entries. Comment lines are preceded by an asterisk and entry lines are preceded by a colon. The file contains the following fields, one per server:

© Copyright IBM Corp. 2003, 2025

Table 1. Fields in the DMSSISVR NAMES File

Field	Description	Tag	Value	Example
Comment	Explanatory remark, preceded by an asterisk	*	*text*	* Default AF_INET Server
Name	Name of the server	:server.	*name*	:server.VSMREQIN
Type	Type of server	:type.	REQUEST, WORKER, DMGR, or [1]	:type.REQUEST
Protocol	Request server network protocol	:protocol.	AF_INET, AF_INET6, AF_IUCV, or AF_EVNT[2]	:protocol.AF_INET
Address	Address for server to bind to (AF_INET/ AF_INET6/ request servers only)	:address.	Valid IP address	:address.INADDR_ANY
Port	Port for server to listen on (AF_INET/ AF_INET6/ request servers only)	:port.	Valid port	:port:44444
Short (Worker Type)	Type of worker server (worker server only, short, long, or guard for VSMGUARD)	:short.	YES, NO, GUARD, PMM, PSTK, DMPH, or DBS[3]	:short.YES

[1] **Type Values:**

REQUEST	Request server
WORKER	Worker server
DMGR	Directory manager

[2] **Protocol Values:**

AF_INET	Use AF_INET (IPv4) family sockets
AF_INET6	Use AF_INET6 (IPv6) family sockets
AF_IUCV	Use AF_IUCV family sockets
AF_EVNT	Listen for and propagate *VMEVENT and directory updates

[3] **Short (Worker Type) Values:**

YES	Short call server
NO	Long call server
GUARD	Guard server
PMM	Performance monitoring server
PSTK	SMAPI TCP/IP stack
DMPH	Dump handler
DBS	Database server

The DMSSISVR NAMES file can be used to define any number of worker servers and request servers. Note, however, that the configuration must contain at least one worker server where the worker type is defined as YES (:short.YES), and at least one worker server where the worker type is defined as NO (:short.NO). Multiple AF_INET/AF_INET6 servers may be defined, as long as their ports are unique. Multiple AF_IUCV servers may also be defined.

The initial entries in the provided DMSSISVR NAMES file are as follows:

Note: The comment lines shown here reflect recent updates and may differ slightly from those in the DMSSISVR NAMES file delivered with your system.

```
* Tag to determine if SMAPI autostarted at install time. NO OTHER ENTRY
* can contain the 'auto' tag. This tag is used by VM INSTALLATION
* code and should NOT be modified by the customer. Modification of this
* tag will have unpredictable results.
```

```
:server.AUTOLOG1
:auto.NO

* Default AF_INET Server
:server.VSMREQIN
:type.REQUEST
:protocol.AF_INET
:address.INADDR_ANY
:port.44444

* Default AF_INET6 Server
:server.VSMREQI6
:type.REQUEST
:protocol.AF_INET6
:address.INADDR_ANY
:port.44445

* Default AF_IUCV Server
:server.VSMREQIU
:type.REQUEST
:protocol.AF_IUCV

* Default AF_EVNT Server
:server.VSMEVSRV
:type.REQUEST
:protocol.AF_EVNT

* Default Guard Server
:server.VSMGUARD
:type.WORKER
:short.GUARD

* Default Short Call Server
:server.VSMWORK1
:type.WORKER
:short.YES

* Default Long Call Server
:server.VSMWORK2
:type.WORKER
:short.NO

* Default Long Call Server
:server.VSMWORK3
:type.WORKER
:short.NO

* Optional Directory Manager. This tag is set by VM INSTALLATION
* code so that SMAPI can determine if the "limited access" copy of
* DIRMAINT is desired. If the limited access copy of DIRMAINT is
* desired, this tag should never be changed by the customer. If the
* limited access copy of DIRMAINT is not desired, this tag can be
* used to indicate that SMAPI, upon its instantiation, should autolog
* the designated server.

:server.DIRMAINT
:type.DMGR

* Default Performance Monitoring Server
:server.PERSMAPI
:type.WORKER
:short.PMM

* Default Database Server
:server.LOHCOST
:type.WORKER
:short.DBS

* Default SMAPI TCP/IP stack
:server.DTCSMAPI
:type.WORKER
:short.PSTK
:address.10.60.100.100

* Dump Handler
*:server.OPERATNS
*:type.WORKER
*:short.DMPH
```

Configuring SMAPI

SMAPI obtains its configuration from two files on the MAINT 193 disk: the IBMCNF COPY file and the DMSSICNF COPY file. Note the following about these two files:

- You must include changes to the configuration properties for the SMAPI servers in the DMSSICNF COPY file. Any configuration property value in the DMSSICNF COPY file supersedes an assignment of the same property in the IBMCNF COPY file. IBM will never ship service for the DMSSICNF COPY file. Configuration property assignments that are identical in DMSSICNF COPY and IBMCNF COPY can safely be removed from DMSSICNF COPY.
- IBM supplies default configuration values in the IBMCNF COPY file. Do not make changes to the IBMCNF COPY file. Configuration property values in IBMCNF COPY can be overwritten by IBM Support Center personnel or superseded by configuration property values in DMSSICNF COPY.
- It is important for all SMAPI servers to be accessing the same SMAPI configuration files at the same time; therefore it is recommended that you restart SMAPI after either SMAPI configuration file is changed.

SMAPI Configuration Properties

To allow services to function properly on z/VM, use XEDIT to edit the DMSSICNF COPY file on the MAINT 193 disk. IBM recommends that you keep at least two previous versions of the file as backups. The following SMAPI configuration properties are specified in the DMSSICNF COPY file. (In most cases, an attribute is shown with its initial value):

- **Authorization Policy**

```
Authorization_Policy = policy
```

The Authorization Policy determines how each API request is authorized. *policy* can be:

Authorization_Policy_EsmAuthlist
Specifies that if an External Security Manager (ESM) is installed, SMAPI calls the ESM first to authorize the request. This is the default setting. If the ESM defers (due to the way the ESM is configured) or is not installed, SMAPI uses the SMAPI authorization process, described in “Authorizing API Requests” on page 36, to decide if the request is authorized, and SMAPI calls the ESM to audit the decision (that is, to allow the ESM to record SMAPI's decision). The ESM's logging options control which, if any, of the audit requests result in audit records.

Note:

- For requests against a list of *n* targets, you can get up to *n* audit requests. SMAPI stops checking the list when the first target is rejected, so you get *n* audit requests whenever the SMAPI authorization process authorizes the list request, and 1-*n* audit requests when a list request is ultimately rejected.
- This setting logs the result of SMAPI authorization processing when an External Security Manager (ESM) defers the authorization request. If REQUEST=AUDIT results in SAF RC=8, the activity is traced but no message is sent to the operator. For non-zero return codes (other than "ESM not installed"), the activity is traced and a message is sent to the system operator.

Authorization_Policy_EsmOnly
Specifies that SMAPI calls the ESM, and never uses the SMAPI authorization process, described in “Authorizing API Requests” on page 36. If the ESM defers, is not installed, or produces a return code other than "authorized" (RACROUTE SAF RC=0), SMAPI rejects the request.

Note: For requests against a list of *n* targets, you can get up to *n* authorization requests. The ESM stops checking the list when the first target is rejected, so you get *n* audit requests whenever the ESM authorizes the list request, and 1-*n* audit requests when a list request is ultimately rejected.

Authorization_Policy_AuthlistOnly
Specifies that SMAPI uses the SMAPI authorization process, described in "Authorizing API Requests" on page 36, and never calls the ESM.

- **SMAPI Instance Name**

```
SMAPI_Instance_Name = "SMAPI"
```

The SMAPI instance name is used to construct ESM profile names when authorizing requests through an External Security Manager (ESM). For more information, see "Configuring SMAPI to use an ESM to Authorize Requests" on page 36.

- **Directory Manager Exit**

```
DM_exit = "DMSSIXDM"
```

The directory manager exit is the code that is called to perform directory manager functions. The `DM_exit` configurable variable should be set to the name of the REXX exec supplied by your directory manager. The default is set to DMSSIXDM, which is the directory manager exit for the IBM Directory Maintenance Facility. Please contact the supplier of your preferred directory manager for more information on configuring your directory manager exit.

For more information on the implementation of the directory manager exit, see Appendix A, "The Directory Manager Exit," on page 843. For more information on the IBM Directory Maintenance Facility and its specific use with the Systems Management APIs, see the *z/VM: Directory Maintenance Facility Tailoring and Administration Guide*.

- **Authorization Exit**

```
XIA_exit = ""
```

The IBM-supplied authorization routine will check the authorization file to determine whether the requested function is authorized to be performed by the requesting userid (authenticated userid) on behalf of the target userid. An external security manager may implement its own authorization functions for the Systems Management APIs by setting the `XIA_exit` configurable variable to the name of an authorization REXX exec. The input parameters to this exit shall be the *authenticated_userid*, *target_identifier* and *function_name* specified on the API call. The input parameters will be in EBCDIC (codepage 924). The function call is as follows:

```
Reason = XIA_exit(authId, targetId, funcName)
```

On input the parameters should be parsed as follows:

```
Parse Upper Arg authId, targetId, funcName
```

The authorization exit must return a 4-byte binary return code directly followed by a 4-byte binary reason code.

- **RPIVAL Program Name**

```
RPIVAL_prog = ""
```

The `RPIVAL_prog` configurable variable may be used to set the name of a program to be used by an external security manager (ESM) to authenticate userids and passwords supplied by client programs (an RPIVAL program is only required if the ESM does not support DIAGNOSE X'88'). When no value is specified for this setting, the default is RPIVAL. If a different program is used, it must follow the

programming conventions (parameter format and return codes) used by RPIVAL. More information on the RPIVAL command may be found in *z/VM: RACF Security Server Macros and Interfaces*.

- **Server_DCSS**

```
Server_DCSS = DCSS_name
```

The `Server_DCSS` configurable variable is used to specify the name of the DCSS which will be automatically created and used by the SMAPI server machines for communication with each other.

- **Asynch Update Port**

```
Asynch_Update = "55555"
```

This is an internal port used by SMAPI to receive asynchronous notifications and pass them on via the event stream.

- **LOHCOST Server Defaults**

```
LOHCOST Port =  "49998"              /* LOHCOST port               */
LOHCOST Addr =  "10.70.100.100"      /* LOHCOST IP address         */
LOHCOST_STACK   = "DTCSMAPI"         /*  private tcp/ip stack       */
LOHCOST_DIRECTORY = 1                /* directory cache enablement mask */
LOHCOST_METADATA = 4                 /* metadata cache enablement mask */
LOHCOST_Enabled = LOHCOST_DIRECTORY
```

The LOHCOST server is used for caching the system directory data required to satisfy the various query APIs. Making changes to the first three lines requires changes to configuration settings and directory entry changes to other SMAPI servers, and the two enablement mask settings must *not* be changed. Therefore, the first five lines of this section should *not* be modified. The last line may be modified as follows:

- To enable LOHCOST caching of directory user data only, set `LOHCOST_Enabled = LOHCOST_DIRECTORY`
- To enable LOHCOST for support of the METADATA APIs only (no caching of directory manager directory or storage group data), set `LOHCOST_Enabled = LOHCOST_METADATA`
- To disable LOHCOST caching of directory data and directory metadata, set `LOHCOST_Enabled = 0`

Note:

- LOHCOST support for METADATA APIs is always enabled unless `LOHCOST_Enabled = 0`.

- **Server Log Level**

```
log_level = 3
```

By default the log level is set to 3, meaning that all request, entry, exit, and parameter information is logged. The log level identifies which debug information is provided and when to provide it. The valid log levels for the systems management server are as follows:

0
No logging.

1
Request logging only – the receipt of a request and confirmation of its completion are logged.

2
Request, entry, and exit – request trace data and entry and exit point trace data is included.

3
Request, entry, exit and parameter logging – all information from log level 2 in addition to parameters and associated log information is provided.

Log entries are written to VSMAPI LOG1 and VSMAPI LOG2 files in the data SFS directory. By default, the files can be found in the VMSYS:VSMWORK1.DATA directory. The server will write time-stamped log entries to VSMAPI LOG1. When the file reaches the maximum size, the file will be copied to VSMAPI LOG2 (replacing previous log entries) and a new VSMAPI LOG1 file will be started. By default, the VSMAPI LOG1 and VSMAPI LOG2 each have a default size of 10000 lines. This default may be altered by changing the `LogLimit =` value, as described in the "**Server Log File Size**" section.

In the event of a worker or request server reboot, SMAPI will save a snapshot of the most recent copies of the SMAPI log files. Up to two levels of the SMAPI log files are saved, with VSMAPI SV1LOG1 and VSMAPI SV1LOG2 being the most recent copies of the log files, and VSMAPI SV2LOG1 and VSMAPI SV2LOG2 being the older set of the log files. By default, these log files are saved in the VMSYS:VSMWORK1.DATA SFS directory.

To view the log file while the server is running, a user can either copy a snapshot of the log file or XEDIT the file using the NOLOCK option.

Note: Do not lock the log file. If you do, this will prevent any further messages from being logged.

- **Authorization List and Name List Configuration**

```
NameListFileIdAny = "VSMWORK1 NAMELIST *"
AuthListFileIdAny = "VSMWORK1 AUTHLIST *"
```

The names of the authorization file and the name list file must be configured in DMSSICNF COPY. By default, these files are named VSMWORK1 NAMELIST and VSMWORK1 AUTHLIST during the installation process. If the names of these files are changed, DMSSICNF must reflect this change. For more information on configuring the authorization list or name list files, see “Authorizing API Requests” on page 36 and “Name Lists” on page 39.

- **SFS Configuration**

```
Server_SFSpool= "VMSYS:"                 /* Default Server filepool   */
Server_SFSdir = "VMSYS:VSMWORK1."        /* Default Server directory  */
Server_DATA   = "VMSYS:VSMWORK1.DATA"    /* Default DATA directory */
Server_SOURCE = "VMSYS:VSMWORK1."        /* Default SOURCE directory */
Server_STATUS = "VMSYS:VSMWORK1.STATUS"  /* Default STATUS directory */
Server_StatusLog_Max = 2                 /* Default STATUS file num  */

DataDisk = "A"
SourceDisk = "B"
```

The default SFS configuration is defined in DMSSICNF COPY. If the configuration is changed, this must be reflected in the DMSSICNF COPY file. For more information about SFS, see “Shared File System Directories” on page 27.

If you change the SFS configuration, make sure that all of the directories are created, that the servers are enrolled in the file pools, and that the VSMWORK1 AUTHLIST and VSMWORK1 NAMELIST files are in the directory specified in Server_SOURCE. Note that all of these directories should be in the same parent directory.

The `Server_STATUS =` and `Server_StatusLog_Max =` attributes are used in conjunction with either the SMAPI_Status_Capture API or the SMSTATUS EXEC. When that API or EXEC completes, there will be an output file created in the VMSYS:VSMWORK1.STATUS directory. The EXEC itself will indicate the name and location of this file. It will be a text file, and can be provided to IBM Service to assist with diagnosing suspected problems. SMAPI will retain the *n* most recent output files from invocations of the API or EXEC. Note that *n* is determined by the `Server_StatusLog_Max =` attribute. See “SMAPI_Status_Capture” on page 511 and Appendix G, “Capturing SMAPI Data for Problem Resolution,” on page 883 for more information.

The DataDisk and SourceDisk variables tell the server profiles where to access the VSMWORK1. and VSMWORK1.DATA SFS directories. By default, they are accessed as file modes B and A, so that executables on those directories supersede executables on other disks (such as the servers' 191 disks and the MAINT 193 disk). An administrator can change this ordering for testing purposes.

Note:

1. The VSMGUARD worker server will grant authority to all the other SMAPI servers that are configured to access the SMAPI file space. Therefore, VSMGUARD must be made an administrator of the VMSYS: file pool. This is done by adding VSMGUARD to the list of users authorized for ADMIN authority. In the default environment, this is done by updating the VMSERVS DMSPARMS file on the VMSERVS 191 disk.
2. For more information on increasing the size of the VMSYS: filepool, if necessary, see *z/VM: CMS File Pool Planning, Administration, and Operation*.

- **SFS Garbage Collection Periodic Check**

```
SFSgcchk = 900
```

SMAPI makes heavy use of SFS. Garbage collection is performed on a periodic basis to allow the file pool server to better manage its virtual storage use. This value is the number of seconds that make up that period.

- **VMRM Configuration**

```
VMRM_SFSdir   = "VMSYS:VMRMSVM."    /* Default VMRM filepool and dir  */
```

The default VMRM configuration is defined in DMSSICNF COPY. If the configuration is changed this must be reflected in the DMSSICNF COPY file. For more information about VMRM, see *z/VM: Performance*.

- **Custom APIs**

```
UserParserFileIdAny = "DMSSIUSR NAMES *"
ulong = ''
```

The name of the file used to specify the user-defined custom APIs must be configured in DMSSICNF COPY by setting the UserParserFileIdAny variable. By default, this file is named DMSSIUSR NAMES. This file must be a CMS NAMES file. A sample of this file is included in DMSSIUSR SAMPNAME on MAINT's 193 disk, as shown below.

```
* Custom API named "Custom_API_1" with custom exec "CUSTOM1 EXEC"
:nick.Custom_API_1
:program.CUSTOM1

* Custom API named "Custom_API_2" with custom exec "CUSTOM2 EXEC"
:nick.Custom_API_2
:program.CUSTOM2
```

The ulong variable should be set to the list of "long" custom APIs. These are APIs that you would like dispatched to the additional worker servers for improved multitasking capability. API names should be blank-separated. Note that the ulong variable has a character restriction of 771 characters. An example:

```
ulong = "Custom_API_1 Custom_API_2"
```

Use the REXX continuation character (a comma) to continue a clause across the following line.

For more information on user-defined custom APIs and configuring the DMSSICNF COPY file, see Appendix B, "Creating Custom APIs," on page 851.

- **Default SYSTEM CONFIG Link Values**

```
System_Config_File_Name = 'SYSTEM'
System_Config_File_Type = 'CONFIG'
Parm_Disk_Owner         = 'PMAINT'
```

These values will be used as the default values in APIs that update SYSTEM CONFIG, when any of the link parameters are left to the default value.

Note: The `Parm_Disk_Number` and `Parm_Disk_Password` values are no longer included in the DMSSICNF COPY file. These values are now hardcoded to CF0 for the disk number, and to a comma for the password (indicating a password is not provided).

- **Dump Processing Values**

```
Dump_Processing_Location = "VMSYSU:OPERATNS."
Dump Processing Interval = "1"
```

The location entry specifies an SFS directory or minidisk where a processed dump should be placed by the dump handler (if activated). If specifying a minidisk, both the owner and the virtual device should be given. Example:

```
 Dump_Processing_Location = "MAINT 999"
```

The interval entry specifies the interval (expressed in minutes) at which the OPERATNS server will check its reader for new dump files to process automatically.

Important: To activate automated dump handling, you must first uncomment the entry for the OPERATNS server in the DMSSISVR NAMES file (see "The Server Names File" on page 27) and also allocate enough space to contain the dump files, at the location specified by the `Dump_Processing_Location =` entry above.

A sample profile exec for the OPERATNS server is provided in file OPERATNS SAMPPROF on MAINT's 193 disk. This sample profile must be copied to each OPERATNS's 191 disk as PROFILE EXEC in order to complete activation of dump handling.

- **IMAGE RECYCLE Maximum Wait Time**

```
Max_Image_Wait_Time = 120
```

The `Max_Image_Wait_Time =` attribute is used to specify the maximum wait time in seconds that the Image_Recycle API will wait for an image to deactivate before attempting to reactivate the image. For more information, see "Image_Recycle" on page 311.

- **Server Log File Size**

```
LogLimit = 10000
```

Log entries are written to VSMAPI LOG1 and VSMAPI LOG2 files in the data SFS directory. While the `log_level =` value determines which debug information is written to those files, the `LogLimit =` value determines the size of those files. The default size is 10000 lines.

- **Temporary Virtual Device Number and Access Mode**

```
Temp_Disk_Vdev = 'A91'
Temp_Acc_Mode  = 'C'
```

These constants specify that the TCPIP IFCONFIG command will have a VDEV default of A91, and that it will be accessed dynamically by SMAPI worker servers as file mode C.

TCP/IP Requirements

All of the SMAPI servers access the TCPMAINT 592 disk by default, and many of those SMAPI servers require that a common TCP/IP stack be operational. If a custom or separate TCP/IP stack is configured for SMAPI, the modified TCPIP DATA file must be placed on each server's 191 disk. (Do *not* place a customized TCP/IP DATA file on the VMSYS:VSMWORK1. or VMSYS:VSMWORK1.DATA directories). All of the SMAPI servers should use this common TCP/IP stack, with the exception of DTCSMAPI.

Client Authentication

A requesting userid and password must be supplied for authentication with each AF_INET/AF_INET6 request. The userid and password pair must be valid on the z/VM system receiving the request in order for authentication to be successful.

For AF_IUCV requests, the requesting userid and password are only required when the requesting userid is different than the userid of the virtual machine that the request is sent from. If the requesting userid is specified and is the same as the userid of the sending virtual machine, it is not authenticated.

Once authentication is complete, the authenticated userid is used to determine if the request is authorized to be performed by the authenticated userid on behalf of the target userid, using the system management authorization file. When authentication is not required for an AF_IUCV request, the userid of the sending client is used in place of the authenticated userid to determine if the request is authorized.

Configuring SMAPI to use an ESM to Authorize Requests

If you are using an External Security Manager (ESM), you can configure SMAPI to use the ESM to authorize requests, while at the same time incorporating SMAPI's existing authorization method. When an ESM makes an authorization decision, the ESM will log the authorization decision based on its active policy, without SMAPI's knowledge or intervention. The ESM can also defer an authorization decision to the SMAPI authorization method. When the ESM makes the authorization decision, the ESM is responsible for all audit logging. When the ESM defers the authorization decision to SMAPI, SMAPI is responsible for all audit logging.

Use SMAPI configuration properties to control the ESM's role in authorizing SMAPI requests. For more information, see the description of the authorization policy properties in “Configuring SMAPI” on page 30. For an explanation of the SMAPI authorization method, see “Authorizing API Requests” on page 36.

You might need to migrate from using the SMAPI authorization method to using the ESM authorization policy attributes. For more information see, “Migrating to Using the ESM Policies for Authorizing APIs” on page 880.

Authorizing API Requests

Authenticated users must be authorized to issue API requests. A server authorization file, described in this section, can be used for this purpose, depending on how SMAPI is configured. For more information on configuring SMAPI to decide if a request is authorized, see the description of the authorization policy properties in “Configuring SMAPI” on page 30.

The authorization file contains entries that authorize authenticated users to perform specific functions for specific virtual images (target users) or lists of virtual images. Each entry is a single record in the file consisting of three fields. Field 1 contains the *requesting user* (authenticated user), field 2 contains the *target virtual image or list* field, and field 3 contains the *requested function*. Note that each entry may be represented as a name list. When a name list is used in the authorization file, all of the items in the list (image names and/or function names) are considered part of that authorization entry. Nested lists, however, are not expanded. If a list name is specified in the authorization file, the items in that list are treated as image or function names.

The authorization file is located on the source SFS directory (VMSYS:VSMWORK1.). It is placed there as part of the default z/VM installation. The default file name is VSMWORK1 AUTHLIST. If you choose to use a different name for the authorization file, you must specify the new name in variable

AuthListFileIdAny in file DMSSICNF COPY. See the "Authorization List and Name List Configuration" entry in "Configuring SMAPI" on page 30.

There are Authorization_List APIs which can be used to update and query the authorization file without stopping the server. These APIs are listed under "Authorization APIs" on page 7. Note that in order for a user to call any of the Authorization_List APIs, there must be an entry in the authorization file that specifically authorizes that user to do so.

The authorization file may also be updated manually. If the file is updated manually, the attributes of fixed record format (RECFM) with a record length (LRECL) of 195 must be maintained, and all entries must be in upper case.

The three fields are described in more detail below:

requesting user
: This is the name of a user, or a list name for a list of users, who will be allowed to perform the *requested function* against the *target virtual image* or list. (Note that the *requesting user* is the same as the *authenticated_userid* on an API call.) The *requesting user* field must start in column 1 of the authorization file entry and be no more than 64 characters in length (8 characters for a single user and 64 characters for a list name).

target virtual image (or list)
: This is the name of the virtual image, or a list name for a list of virtual images to be updated. A keyword **ALL** may also be specified to indicate that the requesting user is authorized to modify all virtual images (users). This field must start in column 66 of the authorization file entry and be no more than 64 characters in length (8 characters for a single user and 64 characters for a list name).

  Note: Although an equal sign (**=**) is accepted by the Authorization_List APIs, it is not a valid token and should not be manually inserted in place of the *target virtual image* or list.

requested function
: This is the function name, or the list name for a list of functions, that the *requesting user* is authorized to perform. A keyword **ALL** may also be specified to indicate an authorization file entry that allows the *requesting user* authorization to all functions for the specified *target virtual image* or list. This field must start in column 131 of the authorization file entry and is a maximum of 64 characters.

Names in the authorization file may be specified as one of the following:

name
: Specific userid or virtual image or function

name list
: The name of a list containing a group of userids or virtual images or functions

ALL
: A keyword encompassing all userids or virtual images or functions

Figure 1 on page 37 shows the default entries that are provided in the VSMWORK1 AUTHLIST file (headings are not included in the actual file).

```
Column 1                    Column 66                   Column 131
|                           |                           |
|                           |                           |
V                           V                           V
DO.NOT.REMOVE               DO.NOT.REMOVE               DO.NOT.REMOVE
MAINT                       ALL                         ALL
IBMVM1                      ALL                         ALL
```

Figure 1. VSMWORK1 Server Authorization File

Note:

1. The DO.NOT.REMOVE line must *not* be removed, and must remain as the first line in the file.
2. SMAPI requests submitted via INET/INET6 servers – which use either AF_INET (IPv4) or AF_INET6 (IPv6) family sockets to connect with clients -- that use MAINT as the authorized user will fail if MAINT

is defined with a password of LBYONLY (which is the default). If MAINT does have a password of LBYONLY, it is recommend you change the authorized user for these API calls to IBMVM1 or another user you have added to the VSMWORK1 AUTHLIST. Requests submitted with MAINT as the authorized user are not affected if the SMAPI call is submitted via IUCV. For more information, see *z/VM: Migration Guide*.

How Authorizing Requests Are Processed

An API call is authorized when all components of the API call are matched with the corresponding element within a single authorization file line. If the target ID component of the API call is a list, all target user IDs within that list must be matched within a single authorization file line in order for the API call to be authorized. If the API call is authorized by a single authorization file line, that is sufficient to authorize the API call. That is, it does not matter if any or all other lines in the authorization file result in a failed authorization.

Each element of an authorization file line is interpreted as follows:

- A fully articulated name or ALL is self-evident
- A list of names contains fully articulated names or ALL

The following examples use these list entries and authorization file:

- MY_TARGETS consists of:

  JOHN
  EMILY
- TRY_TARGETS consists of:

  SCOTT
  JOHN
  STEVG
  STEVES
- MATCH_TARGETS consists of:

  SCOTT
  JOHN
  STEVEW
  EMILY
- Authorization file contents:

```
MAINT      MATCH_TARGETS   IMAGE_ACTIVATE
MAINT      STEVEG          IMAGE_ACTIVATE
MAINT      STEVES          IMAGE_DEACTIVATE
```

Example 1

Assume the following for this example:

```
API call: Image_Activate
Authorized User ID: Maint
Target User ID: MY_TARGETS
```

This call is authorized by the first line in the authorization file, because all the user IDs in MY_TARGETS are listed in MATCH_TARGETS.

Example 2

Assume the following for this example:

```
API call: Image_Activate
Authorized User ID: Maint
Target User ID: TRY_TARGETS
```

This call fails because the target list entry STEVEG is not included in authorization line 1, the other target user IDs are not included in authorization line 2, and authorization line 3 does not authorize for Image_Activate.

Example 3

Assume the following for this example:

```
API call: Image_Deactivate
Authorized User ID: Maint
Target User ID: STEVES
```

This call is authorized because authorization line 3 authorizes the target STEVES for Image_Deactivate. It does not matter that the other authorization lines do not authorize this call.

Name Lists

A list of names may be defined to represent a group of users, virtual images, or functions. Name lists may be used in authorization file entries and as parameters for certain function calls (for example, to activate a group of virtual images).

When a function is called, the target image may be either a list name or a single image name, depending on the specific function. Some functions accept either a list name or single image name as the *target_identifier*. These functions check the name to determine whether it is a list, and if not, process the name as a single image name. Therefore, lists should be given names that cannot be confused with image names.

During authorization checking and function processing, name lists are only expanded once – if a name within a list is also the name of a list, the second (nested) list will not be expanded.

A name list file is used to specify name lists. The name list file is located on the source SFS directory (VMSYS:VSMWORK1). It is placed there as part of the default z/VM installation. The default file name is VSMWORK1 NAMELIST. If you choose to use a different name for the authorization file, you must specify the new name in variable `NameListFileIdAny` in file DMSSICNF COPY (see “Authorization List and Name List Configuration”).

There are Name_List APIs which can be used to update and query the name list file without stopping the server. These APIs are listed under“Name List APIs” on page 14.

The name list file may also be updated manually. Follow these rules when making manual updates to the name list file:

- All records in the file must begin at column 1 of the file.
- The file format of the name list file must be record format fixed with records of length 80.
- There is no limit to the number of names that are in a list.
- Names in a list must be on separate records of the file.
- Names specified in a list must be upper case and up to 8 characters for a userid or virtual image name and up to 64 characters for a list name.
- The last record of the file must be **:nick.LNAME.DO.NOT.REMOVE**

Note: While a list name specified for *target_identifier* is generally limited to 64 characters (in the char43 character set) for most APIs, the IBM DirMaint directory manager limits a list name to 8 characters in the char42 character set (meaning that no underscores are allowed) for three specific APIs: Shared_Memory_Access_Add_DM, Shared_Memory_Access_Query_DM, and Shared_Memory_Access_Remove_DM.

A name list file entry has this format:

```
:nick.List_Name
:list.
FIRST_NAME_IN_LIST
...
...
...
LAST_NAME_IN_LIST
```

Entries in the name list file consist of the following:

:nick.*List_Name*
: This record contains the name of the list. The keyword **:nick.** must be specified in lower case and identifies that this is the start of a new name list file entry. The *List_Name* must immediately follow the **:nick.** keyword with no blanks in between. The list name must be in upper case, can be up to 64 characters in length, can contain underscores, and must not begin with a colon.

:list.
: This record indicates the start of the list of names. The keyword **:list.** must be the next record that follows the **:nick.** list name identifier. The **:list.** keyword must be in lower case.

FIRST_NAME_IN_LIST
: The first name in the list must be on the next record following the **:list.** record.

...
: Indicates more names in the list.

LAST_NAME_IN_LIST
: The last name in the list must be the last record preceding the next name list entry, if any.

Note: The name list file is *not* a CMS NAMES file and should not be used with CMS NAMES utility functions.

Here is the provided initial VSMWORK1 NAMELIST file:

```
:nick.ABEND_DUMP_MANAGEMENT
:list.
DELETE_ABEND_DUMP
PROCESS_ABEND_DUMP
QUERY_ABEND_DUMP
:nick.AUTHORIZATION
:list.
AUTHORIZATION_LIST_ADD
AUTHORIZATION_LIST_REMOVE
AUTHORIZATION_LIST_QUERY
:nick.DIRECTORY_MANAGER_CONTROL
:list.
DIRECTORY_MANAGER_TASK_CANCEL_DM
QUERY_ASYNCHRONOUS_OPERATION_DM
QUERY_DIRECTORY_MANAGER_LEVEL_DM
STATIC_IMAGE_CHANGES_ACTIVATE_DM
STATIC_IMAGE_CHANGES_DEACTIVATE_DM
STATIC_IMAGE_CHANGES_IMMEDIATE_DM
:nick.DIRECTORY_MANAGER_LOCAL_TAG_AND_SCAN
:list.
DIRECTORY_MANAGER_LOCAL_TAG_DEFINE_DM
DIRECTORY_MANAGER_LOCAL_TAG_DELETE_DM
DIRECTORY_MANAGER_LOCAL_TAG_QUERY_DM
DIRECTORY_MANAGER_LOCAL_TAG_SET_DM
DIRECTORY_MANAGER_SEARCH_DM
:nick.DIRECTORY_PARSING
:list.
IMAGE_DEFINITION_ASYNC_UPDATES
IMAGE_DEFINITION_CREATE_DM
IMAGE_DEFINITION_DELETE_DM
IMAGE_DEFINITION_QUERY_DM
IMAGE_DEFINITION_UPDATE_DM
METADATA_DELETE
METADATA_GET
METADATA_SET
QUERY_ALL_DM
:nick.DIRECTORY_UPDATES_SUBSCRIPTION
:list.
ASYNCHRONOUS_NOTIFICATION_DISABLE_DM
ASYNCHRONOUS_NOTIFICATION_ENABLE_DM
ASYNCHRONOUS_NOTIFICATION_QUERY_DM
:nick.DISK_MANAGEMENT
```

```
:list.
PAGE_OR_SPOOL_VOLUME_ADD
SYSTEM_DISK_ACCESSIBILITY
SYSTEM_DISK_ADD
SYSTEM_DISK_IO_QEURY
SYSTEM_DISK_QUERY
SYSTEM_EQID_QUERY
SYSTEM_FCP_EQID_SET
SYSTEM_FCP_FREE_QUERY
SYSTEM_SCSI_DISK_ADD
SYSTEM_SCSI_DISK_DELETE
SYSTEM_SCSI_DISK_QUERY
SYSTEM_WWPN_QUERY
:nick.EVENT_MANAGEMENT
:list.
EVENT_STREAM_ADD
EVENT_SUBSCRIBE
EVENT_UNSUBSCRIBE
SYSTEM_PERFORMANCE_THRESHOLD_DISABLE
SYSTEM_PERFORMANCE_THRESHOLD_ENABLE
:nick.IMAGE_CHARACTERISTICS
:list.
IMAGE_CREATE_DM
IMAGE_DELETE_DM
IMAGE_LOCK_DM
IMAGE_LOCK_QUERY_DM
IMAGE_NAME_QUERY_DM
IMAGE_PASSWORD_SET_DM
IMAGE_QUERY_DM
IMAGE_REPLACE_DM
IMAGE_UNLOCK_DM
:nick.IMAGE_CONNECTIVITY
:list.
VIRTUAL_CHANNEL_CONNECTION_CREATE
VIRTUAL_CHANNEL_CONNECTION_CREATE_DM
VIRTUAL_CHANNEL_CONNECTION_DELETE
VIRTUAL_CHANNEL_CONNECTION_DELETE_DM
VIRTUAL_NETWORK_ADAPTER_CONNECT_LAN
VIRTUAL_NETWORK_ADAPTER_CONNECT_LAN_DM
VIRTUAL_NETWORK_ADAPTER_CONNECT_VSWITCH
VIRTUAL_NETWORK_ADAPTER_CONNECT_VSWITCH_DM
VIRTUAL_NETWORK_ADAPTER_CONNECT_VSWITCH_EXTENDED
VIRTUAL_NETWORK_ADAPTER_CREATE
VIRTUAL_NETWORK_ADAPTER_CREATE_DM
VIRTUAL_NETWORK_ADAPTER_CREATE_EXTENDED
VIRTUAL_NETWORK_ADAPTER_CREATE_EXTENDED_DM
VIRTUAL_NETWORK_ADAPTER_DELETE
VIRTUAL_NETWORK_ADAPTER_DELETE_DM
VIRTUAL_NETWORK_ADAPTER_DISCONNECT
VIRTUAL_NETWORK_ADAPTER_DISCONNECT_DM
VIRTUAL_NETWORK_ADAPTER_QUERY
VIRTUAL_NETWORK_ADAPTER_QUERY_EXTENDED
VIRTUAL_NETWORK_LAN_ACCESS
VIRTUAL_NETWORK_LAN_ACCESS_QUERY
VIRTUAL_NETWORK_LAN_CREATE
VIRTUAL_NETWORK_LAN_DELETE
VIRTUAL_NETWORK_LAN_QUERY
VIRTUAL_NETWORK_OSA_QUERY
VIRTUAL_NETWORK_VLAN_QUERY_STATS
VIRTUAL_NETWORK_VSWITCH_CREATE
VIRTUAL_NETWORK_VSWITCH_CREATE_EXTENDED
VIRTUAL_NETWORK_VSWITCH_DELETE
VIRTUAL_NETWORK_VSWITCH_DELETE_EXTENDED
VIRTUAL_NETWORK_VSWITCH_QUERY
VIRTUAL_NETWORK_VSWITCH_QUERY_EXTENDED
VIRTUAL_NETWORK_VSWITCH_QUERY_STATS
VIRTUAL_NETWORK_VSWITCH_SET
VIRTUAL_NETWORK_VSWITCH_SET_EXTENDED
:nick.IMAGE_CPUS
:list.
IMAGE_CPU_DEFINE
IMAGE_CPU_DEFINE_DM
IMAGE_CPU_DELETE
IMAGE_CPU_DELETE_DM
IMAGE_CPU_QUERY
IMAGE_CPU_QUERY_DM
IMAGE_CPU_SET_MAXIMUM_DM
:nick.IMAGE_DEVICES
:list.
IMAGE_DEVICE_DEDICATE
IMAGE_DEVICE_DEDICATE_DM
IMAGE_DEVICE_RESET
```

```
IMAGE_DEVICE_UNDEDICATE
IMAGE_DEVICE_UNDEDICATE_DM
IMAGE_DISK_COPY
IMAGE_DISK_COPY_DM
IMAGE_DISK_CREATE
IMAGE_DISK_CREATE_DM
IMAGE_DISK_DELETE
IMAGE_DISK_DELETE_DM
IMAGE_DISK_QUERY
IMAGE_DISK_SHARE
IMAGE_DISK_SHARE_DM
IMAGE_DISK_UNSHARE
IMAGE_DISK_UNSHARE_DM
IMAGE_MDISK_LINK_QUERY
:nick.IMAGE_IPL_MANAGEMENT
:list.
IMAGE_IPL_DELETE_DM
IMAGE_IPL_QUERY_DM
IMAGE_IPL_SET_DM
:nick.IMAGE_OPERATIONS
:list.
IMAGE_ACTIVATE
IMAGE_ACTIVE_CONFIGURATION_QUERY
IMAGE_DEACTIVATE
IMAGE_QUERY_ACTIVATE_TIME
IMAGE_RECYCLE
IMAGE_STATUS_QUERY
:nick.IMAGE_VOLUME_MANAGEMENT
:list.
IMAGE_VOLUME_ADD
IMAGE_VOLUME_DELETE
IMAGE_VOLUME_SHARE
IMAGE_VOLUME_SPACE_DEFINE_DM
IMAGE_VOLUME_SPACE_DEFINE_EXTENDED_DM
IMAGE_VOLUME_SPACE_QUERY_DM
IMAGE_VOLUME_SPACE_QUERY_EXTENDED_DM
IMAGE_VOLUME_SPACE_REMOVE_DM
:nick.LIST_DIRECTED_IPL
:list.
IMAGE_SCSI_CHARACTERISTICS_DEFINE_DM
IMAGE_SCSI_CHARACTERISTICS_QUERY_DM
:nick.NAME_LIST
:list.
NAME_LIST_ADD
NAME_LIST_DESTROY
NAME_LIST_QUERY
NAME_LIST_REMOVE
:nick.NETWORK_INTERFACE_CONFIGURATION
:list.
NETWORK_IP_INTERFACE_CREATE
NETWORK_IP_INTERFACE_MODIFY
NETWORK_IP_INTERFACE_QUERY
NETWORK_IP_INTERFACE_REMOVE
:nick.PROFILE_MANAGEMENT
:list.
PROFILE_CREATE_DM
PROFILE_DELETE_DM
PROFILE_LOCK_DM
PROFILE_LOCK_QUERY_DM
PROFILE_QUERY_DM
PROFILE_REPLACE_DM
PROFILE_UNLOCK_DM
:nick.PROTOTYPE_MANAGEMENT
:list.
PROTOTYPE_CREATE_DM
PROTOTYPE_DELETE_DM
PROTOTYPE_NAME_QUERY_DM
PROTOTYPE_QUERY_DM
PROTOTYPE_REPLACE_DM
:nick.RESPONSE_RECOVERY
:list.
RESPONSE_RECOVERY
:nick.SERVER_MANAGEMENT
:list.
CHECK_AUTHENTICATION
QUERY_API_FUNCTIONAL_LEVEL
SMAPI_STATUS_CAPTURE
:nick.SHARED_MEMORY_MANAGEMENT
:list.
SHARED_MEMORY_ACCESS_ADD_DM
SHARED_MEMORY_ACCESS_QUERY_DM
SHARED_MEMORY_ACCESS_REMOVE_DM
```

```
SHARED_MEMORY_CREATE
SHARED_MEMORY_DELETE
SHARED_MEMORY_QUERY
SHARED_MEMORY_REPLACE
:nick.SINGLE_SYSTEM_IMAGE_CLUSTER_MANAGEMENT
:list.
SSI_QUERY
VMRELOCATE
VMRELOCATE_IMAGE_ATTRIBUTES
VMRELOCATE_MODIFY
VMRELOCATE_STATUS
:nick.SYSTEM_MANAGEMENT
:list.
SYSTEM_CONFIG_SYNTAX_CHECK
SYSTEM_INFORMATION_QUERY
SYSTEM_PAGE_UTILIZATION_QUERY
SYSTEM_PERFORMANCE_INFORMATION_QUERY
SYSTEM_PROCESSOR_QUERY
SYSTEM_SERVICE_QUERY
SYSTEM_SHUTDOWN
SYSTEM_SPOOL_UTILIZATION_QUERY
:nick.VMRM_CONFIGURATION_UPDATE
:list.
VMRM_CONFIGURATION_QUERY
VMRM_CONFIGURATION_UPDATE
VMRM_MEASUREMENT_QUERY
:nick.LNAME.DO.NOT.REMOVE
```

Starting and Restarting the Server Environment

The SMAPI VSMGUARD virtual machine has the knowledge and responsibility for starting the SMAPI servers in the correct order. On startup of the VSMGUARD virtual machine, it will start the other SMAPI servers in the proper order. If a SMAPI server is already running, it will use the appropriate commands to first log them off and then log them back on. Thus, the recommended procedure for either starting or restarting the SMAPI servers is to use the VSMGUARD server to perform the start.

If you wish to restart the SMAPI servers, first log off the VSMGUARD server:

```
FORCE VSMGUARD
```

For either a start or restart, autolog the VSMGUARD server:

```
XAUTOLOG VSMGUARD
```

This will shut down all other SMAPI worker and request servers in an orderly fashion, and then restart them in the proper sequence. You can verify that the server has been started by issuing a simple request, such as Query_API_Functional_Level. If configuration errors are found during startup, messages will be sent to the VSMGUARD console. See Appendix E, “Diagnosing Configuration Errors During Server Startup,” on page 873 for more information.

Stopping the Server Environment

SMAPI servers should only be shut down for specific special cases. Because the various SMAPI servers work together to process requests, shutting down one and not the others can affect SMAPI’s ability to process requests. When SMAPI servers are going to be shut down, care should be taken to shut them down correctly. Some servers can be damaged if the FORCE command is used. Unless specifically instructed to do so, follow the instructions in “Starting and Restarting the Server Environment” on page 43 for recycling servers.

The following condition requires that a server be stopped:

- When instructed by IBM support in order to correct a specific error condition.

The SMAPI LOHCOST server must be signaled to shut down rather than using the force command. Thus, it is a good practice to signal the LOHCOST server prior to attempting to force it. The signal response will indicate if the server is set up to receive signals or whether a FORCE command may be used.

Issue the SIGNAL command and specify the WITHIN parameter to instruct the server to shut down within 10 minutes (600 seconds):

```
SIGNAL SHUTDOWN USER LOHCOST  WITHIN 600
```

You will receive the following response if the server has responded to the signal and shut down:

```
HCPSIG2113I User LOHCOST has reported successful termination
```

The following response indicates that the server is not enabled for signals and will need to be shut down with the FORCE command:

```
HCPSIG2110E User LOHCOST is not enabled for signals
```

Servers that are not enabled for signals may be shutdown with the FORCE command:

```
FORCE VSMGUARD
```

Defining Additional Servers

In the default installation, three worker servers, one AF_INET/AF_INET6 request server and one AF_IUCV request server are defined. Additional worker servers, AF_INET/AF_INET6 request servers, or AF_IUCV request servers can be defined in preparation for activating them.

To define an additional worker server, follow these steps:

1. Define the server in the directory, using the directory entry in “Worker Servers ” on page 22 as a model and changing the userid and the definition of the 191 disk, as appropriate.
2. Copy the file VSMWORK1 SAMPPROF on MAINT's 193 disk to PROFILE EXEC on the server's 191 disk.
3. Enroll the server in the file pool specified by the Server_SFSpool variable in the server configuration file.

   Note: See Appendix C, “ENROLL and GRANT Commands Performed Automatically During z/VM Installation,” on page 857 for the list of ENROLL commands that are performed automatically during normal z/VM installation. These can be a useful guide if you are adding a new worker or request server, and wish to enroll your new server in the correct file pool.
4. If using the IBM Directory Maintenance Facility, perform the appropriate configuration as described in *z/VM: Directory Maintenance Facility Tailoring and Administration Guide*.

To define an additional request server, follow these steps:

1. Define the server in the directory, using the directory entry in “Request Servers ” on page 21 as a model and changing the userid and the definition of the 191 disk, as appropriate.
2. Copy the file VSMREQIN SAMPPROF on MAINT's 193 disk to PROFILE EXEC on the server's 191 disk.
3. Enroll the server in the file pool specified by the Server_SFSpool variable in the server configuration file.

Note that an additional server will not be used until it is activated, as described in “Activating or Deactivating Servers” on page 44.

Activating or Deactivating Servers

Worker servers and request servers can be activated (added to the server environment's configuration) and deactivated (removed from the server environment's configuration). When the server environment is not running, worker servers and request servers can be activated and deactivated. Worker servers can also be activated and deactivated while the server environment is running.

Note that servers must be defined, as described in “Defining Additional Servers” on page 44, before they can be activated.

- **Activating or Deactivating Servers When the Server Environment is Not Running**

To activate or deactivate a worker server or request server when the server environment is not running, simply add or remove the corresponding entry in the server names file as described in "The Server Names File" on page 27.

- **Activating or Deactivating Servers When the Server Environment is Running**

  To activate a worker server when the server environment is running, follow these steps:

  1. Grant certain SFS authorizations to the server by issuing the following commands (using the values of the variables defined in the server configuration file):

     ```
     GRANT AUTHORITY Server_SFSdir TO worker_server
     GRANT AUTHORITY Server_DATA TO worker_server (WRITE NEWWRITE
     GRANT AUTHORITY Server_SOURCE TO worker_server (READ NEWREAD
     GRANT AUTHORITY * * Server_DATA TO worker_server (WRITE
     GRANT AUTHORITY * * Server_SOURCE TO worker_server (READ
     ```

     Note:

     a. Before issuing any of these GRANT commands, ensure that you have already issued the appropriate ENROLL command for the file pool. See "Defining Additional Servers" on page 44.

     b. See Appendix C, "ENROLL and GRANT Commands Performed Automatically During z/VM Installation," on page 857 for the list of GRANT commands that are performed automatically during normal z/VM installation. These can be a useful guide if you have added a new worker or request server, and wish to grant the appropriate SFS authorizations.

  2. Autolog the server if it is not already logged on.
  3. Add the corresponding entry in the server names file as described in "The Server Names File" on page 27.

  The request servers will now begin assigning requests to this worker server.

  To deactivate a worker server when the server environment is running, remove (or comment out) the corresponding entry in the server names file. The request servers will then no longer assign requests to this worker server, but the worker server will complete any requests already assigned to it.

Part 3. User's Guide and Reference

© Copyright IBM Corp. 2003, 2025

Chapter 5. Programming Considerations

The following topics are covered in this chapter:

- Sockets Overview
- Data Types
- Call Format

Sockets Overview

The request servers create and bind a listening socket, and then listen for connections. When a connection request is received from a client, the request server accepts the connection, receives the data on the socket, replies to the client with a request ID, and then calls the appropriate worker server. When the worker server completes its work, it sends a request response. The request servers send the response to the client and then close the socket to signal that the request is complete. The request server then optionally logs the request. Information that can be logged includes the requester, the requested function name, the request ID, and the IP address of the client.

Request servers require IPv4 stream (TCP) sockets for the AF_INET request server (VSMREQIN), IPv6 stream (TCP) sockets for the AF_INET6 request server (VSMREQI6), or AF_IUCV stream sockets for the AF_IUCV request server (VSMREQIU).

Data Types

The data in API input and output parameters occur in four distinct types, as follows:

Integer
shown as "(int*n*)" where *n* will be 1, 4, or 8. This denotes a binary integer, 1, 4, or 8 bytes in length, respectively. In some cases, the int*n* definition will be followed by a range. So for example, (int4; range 0-65535) denotes a 4-byte integer with a value between 0 and 65535.

If this is an int4 parameter used to define the length of an ensuing string parameter, note that this integer will be in network byte order (i.e. big-endian).

For int4 and int8 parameters, a value of -1 is used to denote that the parameter is unspecified.

Int1 parameters represent an enumerated data type, with a value of 0 used to denote that the parameter is unspecified.

Note: All negative integer values are stored as unsigned binary numbers. Therefore, an invalid negative integer in an API parameter may result in a "Numeric value greater than maximum" error.

String
shown in one of the following formats:

- (string,*min_length-max_length,character_set*) for a variable length string, where:

  min_length
  is the minimum length required for this string parameter.

  max_length
  is the maximum length allowed for this parameter.

  If this value is shown as "maxlength", then the length of the string has no theoretical maximum – although the length of any output parameter is subject to system limitations, and the length of any input parameter is limited to 16MB-1 minus the length of the other input parameters (and again subject to system limitations).

© Copyright IBM Corp. 2003, 2025

character_set
is the range of characters allowed for the parameter. This will appear in the form "char*nn* [plus *extra_character1 extra_character2*...]," where char*nn* will be as follows:

char
indicates any non-null (x'00') characters

char10
indicates numeric digits 0-9

char16
indicates hexadecimal digits 0-9 and A-F

char17
indicates hexadecimal digits 0-9 and A-F, plus the hyphen (-) or minus sign.

char26
indicates alphabetics A-Z

char36
indicates alphanumerics A-Z plus 0-9

char37
indicates alphanumerics A-Z, 0-9, and the hyphen (-) or minus sign

char42
indicates A-Z plus 0-9 plus @#$+-:

char43
indicates A-Z plus 0-9 plus @#$+-: plus underscore (_)

char44
indicates A-Z plus 0-9 plus @#$+-: plus underscore (_) and the equal sign (=)

charNA
no known character set restrictions

charNB
indicates non-blank (x'20'), non-null (x'00'), non-delimiter (x'FF'), non-carriage return (x'0D'), and non-line-feed (x'0A').

For example, (string,0-8,char26) denotes a string parameter 0 to 8 bytes in length (the 0 meaning that it's optional), with each character in the string being one of the alphabetics A-Z.

In addition, the optional "plus *extra_character1 extra_character2*..." will be used to denote one or more extra characters that will be accepted along with the main "char*nn*" character set. For example:

– (string,1-153,char43 plus.) denotes the char43 character set plus the period (.)
– (string,0-19,char10 plus blank -) denotes the char10 character set plus both the blank ( ) and the minus sign (-).

- (string,*length,character_set*) for a fixed length string, where:

length
is the length of this string parameter. Note that unspecified string parameters have a length of 0.

character_set
is the range of characters allowed for the parameter, similar to the variable length string above.

For example, (string,7,char36) denotes a string that is always seven characters long, with each character in the string belonging to the char36 character set (A-Z plus 0-9).

- (string,*length,constant*) for a constant, where:

length
is the length of this constant.

constant
is the value of the constant. For example, two common constants are the asterisk (*) and "ALL" – which will be shown as follows:

- (string,1,*)
- (string,3,ALL)

Note:

1. While special characters such as "@#$+-!_=" are allowed in some of the above character set definitions, some of these characters may have special uses on certain platforms. They should therefore be used with care. See *z/VM: CP Commands and Utilities Reference* for more information.
2. The character definitions above indicate that alphabetic characters may only be upper case. Although z/VM generally translates character inputs to upper case from its command line, the socket server will accept lower case characters from the client.
3. Strings should be specified using code page 924, the ISO 8859-15 Latin Character (Western Europe) ASCII. These will be translated to characters from code page 924, ISO 8859-15 EBCDIC. Output strings will be translated to code page 924 ASCII before being sent to the client program.

Array
shown as "(array)", this a set of data consisting of *zero or more* instances of one type of component. In this context, a "component" is defined as an integer, a string, a structure, or a nested array.

Structure
shown as "(structure)", this is a set of data consisting of *exactly one* set of components. As in an array, these components can be integer, string, array, or a nested structure, but in a structure there can be a mix of more than one type of component.

Call Format

Each successful API interaction includes a call and two responses. The call and response formats are described.

API call
The API call includes the following parts:

API call name
The call name is required and must precede all parameters. The format and explanation of each call is documented in the Chapter 6, “Socket Application Programming Interfaces,” on page 55 section in a topic whose topic name is the API call name.

Common input parameters
Most calls require nine common input-parameters.

The common parameters are identified by their position and must be specified in the order that they are listed in the documentation. The common parameters must follow the call name and must precede any additional positional parameters and any key-value pair (KVP) parameters.

Syntax errors (RC = 24 and RS = *pprr*) apply only to the nine common input-parameters.

All appropriate parameters for a call and their required position are described in the documentation for that call.

Additional positional input-parameters
Some calls include positional parameters in addition to the nine common parameters.

Additional positional parameters must be specified in the order that they are listed in the documentation. They must follow the common parameters and must precede any key-value pair (KVP) parameters.

The number and types of additional positional parameters varies among API calls. All appropriate parameters for a call and their required position are described in the documentation for that call.

Key-value pair (KVP) input parameters
Some calls include key-value pair (KVP) parameters.

KVP parameters have the following format: `parameter_name=value`. The value must be terminated by an ASCIIZ null character. Each null character is counted as one byte for the total length of the input and must be included in the value of the *input_length* parameter.

KVP parameters must follow positional parameters and can be specified in any order.

If a valid KVP parameter is specified multiple times, the last value specified for that parameter is used.

KVP parameters are not case sensitive.

The value of some KVP parameters contain two or more parts. The parts are separated by a delimiting character or blank spaces or both, and depend on the parameter. The documentation for an API call indicates how the parts can be separated. For example, see the boot_data and boot_record parameters in "Image_IPL_Characteristics_Define_DM" on page 264.

The documentation of KVP parameters indicates allowed data types for parameter values. The data types are described in "Data Types" on page 49. However, KVP parameters are not checked for syntax errors.

KVP parameters are optional unless the documentation states otherwise. If a parameter is optional, the parameter description indicates a minimum length of zero. For example: (string,0-8,char26).

All appropriate KVP parameters for a call are described in the documentation for that call.

Response 1 – Immediate request verification
Every API call yields an immediate, one-parameter response that identifies the request. The response verifies that the request was received.

Response 2 – Output parameters
When the API receives a response from the target system, additional output parameters are returned. Output parameters for a call are described in the documentation for that call.

Usage Notes

- Some parameters identify the length of one or more following parameters. The following requirements apply to length parameters:
  - An ASCIIZ null terminator counts as one byte when calculating the parameter length.
  - If a length field precedes an optional parameter and the optional parameter is not used, then the length field must be defined and the value must be 0.
  - Input-parameter length specifications must exactly match the actual length of the data provided for each parameter. Length errors for a specific parameter could result from an incorrect length specification of a previous parameter.
  - If *input_length* is specified as either zero or as a value over the maximum allowable length of 16MB-1, then the server resets the socket's connection. In this instance, the client does not receive error notification.
- The maximum allowable length of all input parameters is 16MB-1.
- Even parameters that are ignored (or parameters that can be left unspecified) must still be syntactically correct. If an entry does not conform to the character set specified for that parameter, an error is generated.
- In a new release, some existing APIs might have new or changed parameters. To maintain backward compatibility, the parameter syntax will always be checked against the newest format first. If it matches, then that format of the API is used. If a syntax error is encountered, then the server evaluates the previous format of that API, if available. Testing continues against each former version of the API, stopping when a successful syntax match is made, or else returning a syntax error based on the last (oldest) format for that API. Note that this might result in unexpected syntax errors, as the same API call might return different errors for different versions of the API.

- The *target_identifier* parameter identifies the user for which the function is performed. The identified user must match an entry in the authorization file that also contains the *authenticated_userid* and the *function_name*. Even though some APIs do not apply to a specific virtual image or list of virtual images, the value of *target_identifier* is used to verify authorization for all calls. Therefore, a valid *target_identifier* value must be specified for all API calls. The one exception is the Check_Authentication call.
- Return codes for each API call are described in the documentation for that call. All commonly occurring code numbers, values, and descriptions are documented in a separate section. See Chapter 7, “Return and Reason Code Summary,” on page 819.

Chapter 6. Socket Application Programming Interfaces

The socket-based application programming interfaces are described in this chapter.

Refer to Appendix D, “Sample Code,” on page 859 to see a sample C program and a sample Java program, both using several of these APIs.

© Copyright IBM Corp. 2003, 2025

Asynchronous_Notification_Disable_DM

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
entity_type
communication_type
port_number
ip_address_length
ip_address
encoding
subscriber_data_length
subscriber_data

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Asynchronous_Notification_Disable_DM to end notification of updates to specified entities as they occur. The entity type and communication type are specified on input.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 36.

function_name
(string,36,char43) The API function name – in this case, 'Asynchronous_Notification_Disable_DM'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
One of the following:

- (string,1-8,char42) The userid for which notifications will be disabled.
- (string,3,ALL) Disables *all* userids.
- (string,1,*) Disables all those userids which have matching notification subscriptions.

entity_type
(int1) The entity type for which notifications will be sent, as follows:

1
DIRECTORY

Currently, only directory change notifications are supported.

When the *entity_type* is "DIRECTORY", the following additional input arguments must be specified:

communication_type
(int1) The communication used for notifications, as follows:

1
TCP

2
UDP

Currently only "TCP" and "UDP" are supported.

When the *communication_type* is "TCP" or "UDP", the following additional input arguments must be specified:

port_number
(int4; range 0-65535) The port number of the socket that will no longer be receiving the notifications.

ip_address_length
(int4) Length of *ip_address*.

ip_address
(string,7-15,char10 plus .) The IPV4 dotted-decimal IP address of the socket that will no longer receive the notifications.

Note: This interface is intended for IPV4 only.

encoding
(int1) The encoding of the notification data string, as follows:

0
Unspecified

1
ASCII

2
EBCDIC

If unspecified, the default value of ASCII will be used.

subscriber_data_length
(int4) Length of *subscriber_data*.

subscriber_data
One of the following:

- (string,0-64,charNA) The matching subscriber data.
- (string,1,*) Disables *all* matching notifications.

If unspecified, only those subscriptions without subscriber data will be disabled.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Usage Note

The asterisk (*) is not supported in the *target_identifier* field, and will result in a 100/16 reason code/ return code if the SMAPI authorization policy is set to either of the following:

Authorization_Policy_ESMAuthlist
Authorization_Policy_ESMOnly

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter pprr

RC	RC Name	RS	RS Name	Description
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		12	RS_AUTHERR_DM	Request not authorized by directory manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
400	RCERR_IMAGEDEF	4	RS_NOT_FOUND	Image definition not defined
428	RCERR_NOTIFY	8	RS_NOTIFY_NOT_FOUND	No matching entries
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available
592	RCERR_ASYNC_DM	nnnn	opid	Asynchronous operation started - product-specific asynchronous operation ID (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
596	RCERR_INTERNAL_DM	nnnn	psrc	Internal directory manager error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Asynchronous_Notification_Enable_DM

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
entity_type
subscription_type
communication_type
port_number
ip_address_length
ip_address
encoding
subscriber_data_length
subscriber_data

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Asynchronous_Notification_Enable_DM to begin notification of updates to a specified entity as the updates occur. The entity type and communication type are specified on input.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 35.

function_name
(string,35,char43) The API function name – in this case, 'Asynchronous_Notification_Enable_DM'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
One of the following:

- (string,1-8,char42) The image to be notified.
- (string,3,ALL) Notifies *all* images.

entity_type
(int1) The entity type for which notifications will be sent, as follows:

1
DIRECTORY

Currently, only directory change notifications are supported.

When the *entity_type* is "DIRECTORY", the following additional input arguments must be specified:

subscription_type
(int1) The subscription type, as follows:

1
INCLUDE – The *target_identifier* will receive notifications for associated directory changes.

2
EXCLUDE – The *target_identifier* will not receive notifications for associated directory changes. Note that EXCLUDE may be used to omit images from being notified when an INCLUDE subscription exists for all images.

communication_type
(int1) The communication used for notifications, as follows:

1
TCP

2
UDP

Currently only "TCP" and "UDP" are supported.

When the *communication_type* is "TCP" or "UDP", the following additional input arguments must also be specified:

port_number
(int4; range 0-65535) The port number of the socket that will receive the notifications.

ip_address_length
(int4) Length of *ip_address*.

ip_address
(string,7-15,char10 plus .) The IPV4 dotted-decimal IP address of the socket that will receive the notifications.

Note: This interface is intended for IPV4 only.

encoding
(int1) The encoding of the notification data string, as follows:

0
Unspecified

1
ASCII

2
EBCDIC

If unspecified, the default value of ASCII will be used.

subscriber_data_length
(int4) Length of *subscriber_data*.

subscriber_data
One of the following:

- (string,0-64,charNA) Anything the subscriber wishes to receive along with the notifications. The format of this data will be as specified in the *encoding* parameter.
- (string,1,*) A single asterisk may be specified here as subscriber data – however such a subscription cannot be separately queried or deleted.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Usage Notes

1. The program to receive asynchronous subscription notifications is a socket application which, for example, could do the following:
   a. Obtain a socket for the desired protocol, UDP or TCP, which will be specified on the *communication_type* parameter.
   b. Bind the socket to either a desired port or allow the system to provide the port by specifying port 0. Also, specify on the bind() that all interfaces should be bound by specifying the value 'INADDR_ANY' for the address. The port value used should specified on the *port_number* parameter. The IP address specified on the *ip_address* parameter should be 0, which indicates to the VSMWORK1 server that it should determine the IP address.

c. If the socket protocol is TCP, post a listen on the socket.

d. Issue this API with the *communication_type*, *port_number* and *ip_address* parameters as determined by the previous steps. Specify the *encoding* parameter based on the characteristic of the operating system to run the socket application. The directory manager will send the asynchronous subscription notification in either ASCII or EBCDIC form, based on the value of the *encoding* parameter. Specify ASCII as the *encoding* parameter, for example, when your operating system is Linux on IBM Z. Specify EBCDIC, for example, when your socket application will run on CMS. Specify the *subscriber_data* parameter, if desired. The *subscriber_data* is any data that is useful for your application, including binary data. For example, it could be the address of a control block.

e. Wait to be informed of asynchronous subscription notifications on the socket.

- When *communication_type* is specified as TCP, then your application will need to accept an incoming connection, receive the asynchronous subscription notification message, close the connection and wait for the next connection. The accept, receive, close sequence will need to be done for each asynchronous subscription notification message.
- When *communication_type* is specified as UDP, then your application will need to either receive or wait on incoming asynchronous subscription notification requests, for example, using either the recvfrom() or select() socket functions.

f. When data is available, the format of each asynchronous subscription notification message is the same for both the TCP and UDP protocol and is encoded based on the specified *encoding* parameter, as follows:

userid_length
(int4) Length of the following userid.

userid
(string,1-8,char42) The new, deleted, or changed userid.

user_word_length
(int4) Length of the following *user_word* field.

user_word
(string,1-16,char42) Any additional data, provided for display or information purposes only, that a directory manager would like to convey to the client about the notification. This could be the name of a new, deleted, or changed directory statement, a command name that caused the notification, or any other information that the directory manager deems useful.

sub_data_length
(int4) Length of the following subscriber data.

sub_data
(stirng,0-64,charNA) The value of the subscriber data causing this notification. The first 5 bytes of the subscriber data is the word 'DATA', followed by the subscriber data you specified.

2. Since this API requires information that is used by the program that is to receive asynchronous subscription notifications, it might be useful for that program to set up the socket as described in Usage Note “1” on page 62 above and then call this API with the appropriate information (as illustrated in Step “1.d” on page 63).
3. If *communication_type* is specified as UDP, each asynchronous subscription notification message received contains the complete message. If the length specified for the receive of the data on the socket is too small, the data will be truncated. For TCP, parts of the asynchronous subscription notification message can be received. For example, a technique might be useful where the length field is received and then the field itself is received for the specified length.
4. The length fields of the asynchronous subscription notification message as described above in Step “1.f” on page 63 (the length of the userid, length of the *user_word*, and length, if any, of the subscriber data) are in network byte order. Network byte order uses the big-endian byte ordering, which is the byte order used by the directory manager sending the asynchronous subscription notification messages. Your socket application receiving asynchronous subscription notification messages may need to convert from network byte order to host byte order if, for example, it runs on Linux on IBM

Z. Functions such as ntohl(), which convert between network byte order and host byte order, can be used.

5. For more information about the *user_word* field described above in Step "1.f" on page 63, refer to the product-specific information for your directory manager.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter pprr
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		12	RS_AUTHERR_DM	Request not authorized by directory manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
400	RCERR_IMAGEDEF	4	RS_NOT_FOUND	Image definition not defined
428	RCERR_NOTIFY	4	RS_NOTIFY_DUPLICATE	Duplicate subscription
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available
592	RCERR_ASYNC_DM	nnnn	opid	Asynchronous operation started - product-specific asynchronous operation ID (See "Internal Return Codes (RC = 396, 592, or 596)" on page 834)
596	RCERR_INTERNAL_DM	nnnn	psrc	Internal directory manager error - product-specific return code (See "Internal Return Codes (RC = 396, 592, or 596)" on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Asynchronous_Notification_Query_DM

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
entity_type
communication_type
port_number
ip_address_length
ip_address
encoding
subscriber_data_length
subscriber_data

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
notification_array_length
notification_array (1)
    notification_structure (2)
        notification_structure_length
        userid_length
        userid
        subscription_type
        communication_type
        port_number
        ip_address_length
        ip_address
        encoding
        subscriber_data_length
        subscriber_data

Note:

1. An array consists of zero or more of its components.
2. A structure consists of one set of its components.

Purpose

Use Asynchronous_Notification_Query_DM to query which users are subscribed to receive notification of updates to specified entities.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 34.

function_name
(string,34,char43) The API function name – in this case, 'Asynchronous_Notification_Query_DM'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See “Client Authentication” on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See “Client Authentication” on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
One of the following:

- (string,1-8,char42) The images to be notified.
- (string,3,ALL) Queries *all* userids.
- (string,1,*) Queries all those userids which have matching notification subscriptions.

entity_type
(int1) The entity type for which notifications will be sent, as follows:

1
DIRECTORY

Currently, only directory change notifications are supported.

When the *entity_type* is "DIRECTORY", the following additional input arguments must be specified:

communication_type
(int1) The communication type of the notification strings being queried, as follows:

0
Unspecified

1
TCP

2
UDP

If unspecified, all types of notification strings for all communication protocols will be returned. Note that currently, only "TCP" and "UDP" are supported.

When the *communication_type* is "TCP" or "UDP", the following additional input arguments must also be specified:

port_number
(int4; range 0-65535) The port number of the socket that will receive the notifications. A null selects all that qualify.

ip_address_length
(int4) Length of *ip_address*.

ip_address
(string,0-15,char10 plus .) The IPV4 IP address of the socket that will receive the notifications. A null selects all that qualify.

Note: This interface is intended for IPV4 only.

encoding
(int1) The encoding of the notification strings being queried, as follows:

0
Unspecified

1
ASCII

2
EBCDIC

If unspecified, all types of encoded notification strings will be returned.

subscriber_data_length
(int4) Length of *subscriber_data*.

subscriber_data
One of the following:

- (string,0-64,charNA) Anything the subscriber wishes to receive along with the notifications. The format of this data will be as specified in the ENCODING parameter. A null selects only those entries with *no* subscriber data.
- (string,1,*) Selects *all* that qualify.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

notification_array_length
(int4) Length of *notification_array*.

notification_array
(array) An array consisting of zero or more instances of *notification_structure*, as follows:

notification_structure
(structure) A structure consisting of one set of the following parameters:

notification_structure_length
(int4) The combined length of the remaining parameters in *notification_structure* (not including this parameter).

userid_length
(int4) Length of *userid*.

userid
(string,1-8,char42) A userid or "ALL".

subscription_type
(int1) The subscription type, as follows:

1
INCLUDE

2
EXCLUDE

communication_type
(int1) One of the following:

1
TCP

2
UDP

port_number
(int4) Port number.

ip_address_length
(int4) Length of *ip_address*.

ip_address
(string,7-15,char10 plus .) The IPV4 IP address

encoding
(int1) The encoding of the notification string, as follows:

1
ASCII

2
EBCDIC

subscriber_data_length
(int4) Length of *subscriber_data*.

subscriber_data
(string,0-64,charNA) Subscriber data fields.

Usage Note

The asterisk (*) is not supported in the *target_identifier* field, and will result in a 100/16 reason code/return code if the SMAPI authorization policy is set to either of the following:

Authorization_Policy_ESMAuthlist
Authorization_Policy_ESMOnly

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
		28	RS_NOTIFY_NOT_FOUND	No matching entries found
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter pprr
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		12	RS_AUTHERR_DM	Request not authorized by directory manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
400	RCERR_IMAGEDEF	4	RS_NOT_FOUND	Image definition not defined
428	RCERR_NOTIFY	8	RS_NOTIFY_NOT_FOUND	No matching entries
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available
592	RCERR_ASYNC_DM	nnnn	opid	Asynchronous operation started - product-specific asynchronous operation ID (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
596	RCERR_INTERNAL_DM	nnnn	psrc	Internal directory manager error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Authorization_List_Add

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
for_id_length
for_id
function_id_length
function_id

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Authorization_List_Add to add an entry to the authorization file.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 22.

function_name
(string,22,char43) The API function name – in this case, 'Authorization_List_Add'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See “Client Authentication” on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
One of the following:

- (string,1-8,char42) The userid or image name.
- (string,1-64,char43) The name of the list of userids or image names.

This is the userid or list of userids being authorized. The *target_identifier* is placed in the "Requesting User(s)" field of the authorization file record.

for_id_length
(int4) Length of *for_id*.

for_id
One of the following:

- (string,1-8,char42) The userid.
- (string,1-64,char43) The name of the list of userids.
- (string,1,=) The value in *target_identifier* is also used as the value for *for_id*. Although = is accepted by this function as input to *for_id*, it is not a valid authorization file entry.
- (string,3,ALL) *target_identifier* is authorized to perform the designated function(s) for all images.

This is the virtual image or list of virtual images for which *target_identifier* will be authorized to perform the designated function(s).

for_id is placed in the "Target Image(s)" field of the authorization file record.

function_id_length
(int4) Length of *function_id*.

function_id
One of the following:

- (string,1-64,char43) The function or list of functions that *target_identifier* is authorized to perform for *for_id*.
- (string,3,ALL) Authorizes *target_identifier* to perform all functions for the designated virtual image(s).

This specifies the name of the function(s) in the "Function(s)" field of the authorization file record(s) being queried.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Usage Notes

1. The following authorization file entry is created by Authorization_List_Add:

Requesting User(s)	Target Image(s) or User(s)	Function(s)
target_identifier	*for_id*	*function_id*

2. This function checks the name to determine whether it is a list, and if not, processes the name as a single image name. Therefore, lists should be given names that cannot be confused with image names.
3. During authorization checking and function processing, name lists are only expanded once; although a name within a list may also be the name of a list, the second (nested) list will not be expanded.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
28	RCERR_FILE_NOT_FOUND	0	RS_NONE	Namelist file not found
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
104	RCERR_NO_AUTHFILE	0	RS_NONE	Authorization file not found
106	RCERR_AUTHFILE_RO	0	RS_NONE	Authorization file cannot be updated
108	RCERR_EXISTS	0	RS_NONE	Authorization file entry already exists
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
396	RCERR_INTERNAL	nnnn	psrc	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist

RC	RC Name	RS	RS Name	Description
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Authorization_List_Query

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
for_id_length
for_id
function_id_length
function_id

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
auth_record_array_length
auth_record_array (1)
    auth_record_structure (2)
        auth_record_structure_length
        requesting_userid_length
        requesting_userid
        requesting_list_indicator
        for_userid_length
        for_userid
        for_list_indicator
        function_name_length
        function_name
        function_list_indicator

Note:

1. An array consists of zero or more of its components.
2. A structure consists of one set of its components.

Purpose

Use Authorization_List_Query to query the entries in the authorization file.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 24.

function_name
(string,24,char43) The API function name – in this case, 'Authorization_List_Query'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
One of the following:

- (string,0-8,char42) The userid or image name.
- (string,0-64,char43) The name of the list of userids or image names.

This is the userid or list of userids in the "Requesting User(s)" field of the authorization file record(s) being queried.

If unspecified, all authorizations are queried.

for_id_length
(int4) Length of *for_id*.

for_id
One of the following:

- (string,0-8,char42) The userid.
- (string,0-64,char43) The name of the list of userids.
- (string,1,=) The value in *target_identifier* is also used as the value for *for_id*. Although = is accepted by this function as input to *for_id*, it is not a valid authorization file entry.

- (string,1,*) Authorization is queried for all virtual images currently listed for *target_identifier* in the authorization file.

This is the virtual image or list of virtual images for which *target_identifier* will be authorized to perform the designated function(s).

for_id is placed in the "Target Image(s)" field of the authorization file record.

If unspecified, an asterisk (*) is assumed (authorization is queried for all virtual images).

function_id_length
(int4) Length of *function_id*.

function_id
One of the following:

- (string,0-64,char43) The function or list of functions that *target_identifier* is authorized to perform for *for_id*.
- (string,1,*) Authorization is queried for all functions currently listed for *target_identifier* in the authorization file.

This specifies the name of the function(s) in the "Function(s)" field of the authorization file record(s) being queried.

If unspecified, an asterisk (*) is assumed (authorization is queried for all functions).

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

auth_record_array_length
(int4) Length of *auth_record_array*.

auth_record_array
(array) An array consisting of zero or more instances of *auth_record_structure*, as follows:

auth_record_structure
(structure) A structure consisting of one set of the following parameters:

auth_record_structure_length
(int4) The combined length of the remaining parameters in *auth_record_structure* (not including this parameter).

requesting_userid_length
(int4) Length of *requesting_userid*.

requesting_userid
One of the following:

- (string,1-8,char42) A userid.
- (string,1-64,char43) A list of userids.

requesting_list_indicator
(int1) This will be 0 if *requesting_userid* is a single userid, 1 if it is a list of userids.

for_userid_length
(int4) Length of *for_userid*.

for_userid
One of the following:

- (string,1-8,char42) A userid.
- (string,1-64,char43) A list of userids.

for_list_indicator
(int1) This will be 0 if *for_userid* is a single userid, 1 if it is a list of userids.

function_name_length
(int4) Length of *function_name*.

function_name
(string,1-64,char43) A function or list of functions.

function_list_indicator
(int1) This will be 0 if *function_name* is a single userid, 1 if it is a list of userids.

Usage Notes

1. If a list name is specified for *target_identifier*, *for_id*, or *function_id*, the list name is not expanded. The authorization file record with the list name in the appropriate field will be returned.
2. This function checks the name to determine whether it is a list, and if not, processes the name as a single image name. Therefore, lists should be given names that cannot be confused with image names.
3. During authorization checking and function processing, name lists are only expanded once; although a name within a list may also be the name of a list, the second (nested) list will not be expanded.
4. Authorization file records which contain a list name including the specified *target_identifier*, *for_id*, or *function_id* will be returned as a match for the query request.
5. The keyword 'ALL' in the *for_id* or *function_id* field of an authorization file record will match any input value that is specified for *for_id* or *function_id*.
6. The asterisk (*) is not supported in the *target_identifier* field, and will result in a 100/16 reason code/ return code if the SMAPI authorization policy is set to either of the following:

   Authorization_Policy_ESMAuthlist
   Authorization_Policy_ESMOnly

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
28	RCERR_FILE_NOT_FOUND	0	RS_NONE	Namelist file not found
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
104	RCERR_NO_AUTHFILE	0	RS_NONE	Authorization file not found
112	RCERR_NO_ENTRY	0	RS_NONE	Authorization file entry does not exist

RC	RC Name	RS	RS Name	Description
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
396	RCERR_INTERNAL	nnnn	psrc	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Authorization_List_Remove

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
for_id_length
for_id
function_id_length
function_id

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Authorization_List_Remove to remove an entry from the authorization file.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 25.

function_name
(string,25,char43) The API function name – in this case, 'Authorization_List_Remove'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See “Client Authentication” on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
One of the following:

- (string,1-8,char42) The userid or image name.
- (string,1-64,char43) The name of the list of userids or image names.

This is the userid or list of userids whose authorization to perform the designated function(s) is to be removed. *target_identifier* is located in the "Requesting User(s)" field of the authorization file record.

for_id_length
(int4) Length of *for_id*.

for_id
One of the following:

- (string,1-8,char42) The userid.
- (string,1-64,char43) The name of the list of userids.
- (string,1,=) The value in *target_identifier* is also used as the value for *for_id*. Although = is accepted by this function as input to *for_id*, it is not a valid authorization file entry.
- (string,1,*) Authorization is removed for all virtual images currently listed for *target_identifier* in the authorization file.

for_id is located in the "Target Image(s)" field of the authorization file record.

function_id_length
(int4) Length of *function_id*.

function_id
One of the following:

- (string,1-64,char43) The function or list of functions for which *target_identifier*'s authorization to perform for *for_id* will be removed.
- (string,1,*) Authorization is removed for all functions currently listed for *target_identifier* in the authorization file.

This specifies the name of the function(s) in the "Function(s)" field of the authorization file record(s) being queried. *function_id* is located in the "Function(s)" field of the authorization file record.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Usage Notes

1. This function checks the name to determine whether it is a list, and if not, processes the name as a single image name. Therefore, lists should be given names that cannot be confused with image names.
2. During authorization checking and function processing, name lists are only expanded once; although a name within a list may also be the name of a list, the second (nested) list will not be expanded.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
28	RCERR_FILE_NOT_FOUND	0	RS_NONE	Namelist file not found
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
104	RCERR_NO_AUTHFILE	0	RS_NONE	Authorization file not found
106	RCERR_AUTHFILE_RO	0	RS_NONE	Authorization file cannot be updated
112	RCERR_NO_ENTRY	0	RS_NONE	Authorization file entry does not exist
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
396	RCERR_INTERNAL	nnnn	psrc	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Check_Authentication

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Check_Authentication to validate a userid/password pair.

Note: Because it does not include a *target_identifier* parameter, Check_Authentication is the only API that does not conform to the set of common input parameters (as described in “Call Format” on page 51).

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 20.

function_name
(string,20,char43) The API function name – in this case, 'Check_Authentication'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See “Client Authentication” on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Usage Notes

- Only a user that is authorized for the CHECK AUTHENTICATION call for *all* targets can issue a successful call.
- A call to this function will result in a 100/16 reason code/return code if the SMAPI authorization policy is set to either of the following:

  Authorization_Policy_ESMAuthlist
  Authorization_Policy_ESMOnly

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
396	RCERR_INTERNAL	nnnn	psrc	Internal system error - product-specific return code (See "Internal Return Codes (RC = 396, 592, or 596)" on page 834)

RC	RC Name	RS	RS Name	Description
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Delete_ABEND_Dump

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
`id`=*value*

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Delete_ABEND_Dump to instruct the dump processing userid to remove a specified ABEND dump from the reader or from the dump processing location specified in the DMSSICNF COPY file. (See the `Dump_Processing_Location =` entry in "Configuring SMAPI" on page 30 for more information.)

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 17.

function_name
(string,17,char43) The API function name – in this case, 'Delete_ABEND_Dump'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) This must match an entry in the authorization file that also contains the *authenticated_userid* and the *function_name* (Delete_ABEND_Dump).

Note: The format for specifying the following additional input parameter is *parameter_name=value*, followed by a null (ASCIIZ) terminator. For more information, see "Key-value pair (KVP) input parameters" on page 51.

`id=`*value*
(string,1-8,char42) The filename (SFS directory) or spool ID (reader) of a dump file. This input parameter is required.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Usage Notes

1. The actual deletion of the dump file occurs asynchronously. When it has completed, a type 2009 event will be transmitted indicating success or failure.
2. Syntax errors (RC = 24 and RS = *pprr*) are only applicable to the common input parameters. Syntax checking is *not* performed on the additional input parameters for this API. If a valid parameter is specified multiple times, the last value specified for that parameter will be used. For more information, see "Key-value pair (KVP) input parameters" on page 51.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
8	RC_ERR	3002	RS_INVALID_PARAMETER	Invalid parameter name
		3003	RS_INVALID_OPERAND	Invalid parameter operand

RC	RC Name	RS	RS Name	Description
		3004	RS_MISSING_PARAMETER	Required parameter missing
24	RCERR_SYNTAX	*pprr*	*pprr*	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
200	RCERR_IMAGEOP	12	RS_NOT_ACTIVE	The dump processing userid (OPERATNS) is either not logged on or is busy processing a dump
396	RCERR_INTERNAL	*nnnn*	*psrc*	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Directory_Manager_Local_Tag_Define_DM

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
tag_name_length
tag_name
tag_ordinal
define_action

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Directory_Manager_Local_Tag_Define_DM to define a local tag or named comment record to contain installation-specific information about a virtual image.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 37.

function_name
(string,37,char43) The API function name – in this case, 'Directory_Manager_Local_Tag_Define_DM'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See “Client Authentication” on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) This must match an entry in the authorization file that also contains the *authenticated_userid* and the *function_name* (Directory_Manager_Local_Tag_Define_DM).

tag_name_length
(int4) Length of *tag_name*.

tag_name
(string,1-8,char36) The name of the local tag or named comment to be defined.

tag_ordinal
(int4; range 0-999) The value of the tag sort ordinal, relative to other defined local tags.

define_action
(int1) Specifies creation of a new tag or change of a tag ordinal value, as follows:

1
Create a new tag.

2
Change an existing tag's ordinal value. See Usage Note "2" on page 89.

If unspecified, the default is 1 (create a new tag).

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Usage Notes

1. See the "Creating and Updating a User Directory" chapter in *z/VM: CP Planning and Administration* for more information on the directory format and on specific directory statements.
2. When ordinal values are changed, any new tags will be ordered according to these values. Existing tags, however, will not be reordered.

3. For more information on how tag data is stored in the directory, see the Directory_Manager_Local_Tag_Set_DM Usage Note “2” on page 98.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
		8	RS_OFFLINE	Request successful; object directory offline
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter pp
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		12	RS_AUTHERR_DM	Request not authorized by directory manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
432	RCERR_TAG	4	RS_DUP_NAME	Tag name is already defined.
		8	RS_NOT_DEFINED	Tag name is not defined.
		12	RS_DUP_ORDINAL	Tag ordinal is already defined.
		16	RS_CANNOT_REVOKE	Tag is in use in one or more directory entries, can not be revoked.
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available
592	RCERR_ASYNC_DM	nnnn	opid	Asynchronous operation started - product-specific asynchronous operation ID (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
596	RCERR_INTERNAL_DM	nnnn	psrc	Internal directory manager error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Directory_Manager_Local_Tag_Delete_DM

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
tag_name_length
tag_name

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Directory_Manager_Local_Tag_Delete_DM to remove a local tag or named comment record from the directory manager's internal tables. Users will no longer be able to set or query the tag.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 37.

function_name
(string,37,char43) The API function name – in this case, 'Directory_Manager_Local_Tag_Delete_DM'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See “Client Authentication” on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) This must match an entry in the authorization file that also contains the *authenticated_userid* and the *function_name* (Directory_Manager_Local_Tag_Delete_DM).

tag_name_length
(int4) Length of *tag_name*.

tag_name
(string,1-8,char36) Specifies the name of the tag to be deleted.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Usage Notes

1. For more information on how tag data is stored in the directory, see the Directory_Manager_Local_Tag_Set_DM Usage Note "2" on page 98.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
		8	RS_OFFLINE	Request successful; object directory offline
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		12	RS_AUTHERR_DM	Request not authorized by directory manager

RC	RC Name	RS	RS Name	Description
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
432	RCERR_TAG	8	RS_NOT_DEFINED	Tag Name Is Not Defined
		16	RS_CANNOT_REVOKE	Tag is in use by one or more directory entries, cannot be revoked
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available
592	RCERR_ASYNC_DM	nnnn	opid	Asynchronous operation started - product-specific asynchronous operation ID (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
596	RCERR_INTERNAL_DM	nnnn	psrc	Internal directory manager error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Directory_Manager_Local_Tag_Query_DM

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
tag_name_length
tag_name

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
tag_value_length
tag_value

Purpose

Use Directory_Manager_Local_Tag_Query_DM to obtain the value of a virtual image's local tag or named comment record.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 36.

function_name
(string,36,char43) The API function name – in this case, 'Directory_Manager_Local_Tag_Query_DM'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See “Client Authentication” on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The target userid whose tag is being queried.

tag_name_length
(int4) Length of *tag_name*.

tag_name
(string,1-8,char36) The name of the local tag or named comment to be queried.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

tag_value_length
(int4) Length of *tag_value*.

tag_value
(string,1-1024,charNA) The value of the associated tag.

Usage Notes

1. See the "Creating and Updating a User Directory" chapter in *z/VM: CP Planning and Administration* for more information on the directory format and on specific directory statements.
2. For more information on how tag data is stored in the directory, see the Directory_Manager_Local_Tag_Set_DM Usage Note "2" on page 98.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful

RC	RC Name	RS	RS Name	Description
		28	RS_NONE_FOUND	No matching entries found. Return buffer is empty.
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter pp
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		12	RS_AUTHERR_DM	Request not authorized by directory manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
400	RCERR_IMAGEDEF	4	RS_NOT_FOUND	Image definition not found
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available
592	RCERR_ASYNC_DM	nnnn	opid	Asynchronous operation started - product-specific asynchronous operation ID (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
596	RCERR_INTERNAL_DM	nnnn	psrc	Internal directory manager error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Directory_Manager_Local_Tag_Set_DM

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
tag_name_length
tag_name
tag_value_length
tag_value

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Directory_Manager_Local_Tag_Set_DM to set the value of a virtual image's local tag or named comment record.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 34.

function_name
(string,34,char43) The API function name – in this case, 'Directory_Manager_Local_Tag_Set_DM'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The target userid whose tag is being set.

tag_name_length
(int4) Length of *tag_name*.

tag_name
(string,1-8,char36) The name of the local tag or named comment to be set.

tag_value_length
(int4) Length of *tag_value*.

tag_value
(string,1-1024,charNA) The value of a virtual image's local tag or named comment to be set (or the key word "DELETE"). This value consists of tokens of data separated by blanks. The total length of all tokens plus the blanks separating them may not exceed 1024. In addition, the total length of any one token, plus the length of the tag_name, cannot exceed 57.

See Usage Note "2" on page 98 for more information on how this tag data is stored in the directory.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Usage Notes

1. See the "Creating and Updating a User Directory" chapter in *z/VM: CP Planning and Administration* for more information on the directory format and on specific directory statements.
2. Tag data is stored in associated *target_identifier* directory entries as comment records, according to the following rules:

- When stored in the directory, each comment record consists of a prefix token followed by a blank and then a token of data. The prefix token is the *tag_name* preceded by an asterisk (*) and appended with a colon (:). So for example, a tag name 'Class' comment record might look like this:

```
*Class: This is an example of a single-line comment record
```

- The comment record may be indented by the directory manager. It does not have to begin in column 1.
- Any sequence of multiple blanks in tag data will be reduced to one blank. For example: 'A B C D' will be stored as 'A B C D'.
- Each line in the directory is limited to 60 characters in length. (Hence the 57-character limit on any one token in *tag_value* plus the length of *tag_name*. Including the added asterisk, colon, and blank, that equals 60.) If necessary, the contents of *tag_value* will be split at the appropriate blank separators, and the prefix token will be repeated on each additional line. Here's how a multiple-line entry might look:

```
*Class: This is an example of a multiple-line comment
*Class: record. Because of the maximum length requirement,
*Class: it must be split into as many lines as needed.
```

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
		8	RS_OFFLINE	Request successful; object directory offline
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter pp
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		12	RS_AUTHERR_DM	Request not authorized by directory manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
400	RCERR_IMAGEDEF	4	RS_NOT_FOUND	Image definition not found
432	RCERR_TAG	8	RS_NOT_DEFINED	Tag name is not defined.
		20	RS_NOT_AUTHORIZED	Use not allowed by exit routine.
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available
592	RCERR_ASYNC_DM	nnnn	opid	Asynchronous operation started - product-specific asynchronous operation ID (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)

RC	RC Name	RS	RS Name	Description
596	RCERR_INTERNAL_DM	nnnn	psrc	Internal directory manager error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Directory_Manager_Search_DM

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
search_pattern_length
search_pattern

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
statement_array_length
statement_array (1)
statement_structure (2)
target_id_length
target_id
statement_length
statement

Note:

1. An array consists of zero or more of its components.
2. A structure consists of one set of its components.

Purpose

Use Directory_Manager_Search_DM to search the directory for records that match the specified pattern.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 27.

function_name
(string,27,char43) The API function name – in this case, 'Directory_Manager_Search_DM'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) This must match an entry in the authorization file that also contains the *authenticated_userid* and the *function_name* (Directory_Manager_Search_DM).

search_pattern_length
(int4) Length of *search_pattern*.

search_pattern
(string,1-72,charNA) The records to be searched for. Tokens must be separated by blanks.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

statement_array_length
(int4) Length of *statement_array*.

statement_array
(array) An array consisting of zero or more instances of *statement_structure*, as follows:

statement_structure
(structure) A structure consisting of one set of the following parameters:

target_id_length
(int4) Length of *target_id*.

target_id
(string,1-8,char42) The userid or profile containing the matching statement. If the statement is not associated with a userid or profile entry (for example, a DIRECTORY or GLOBALDEFS statement), then this field will be '*NONE*'.

statement_length
(int4) Length of *statement*.

statement
(string,1-72,charNA) The matching statement (1-72 bytes, with trailing blanks removed).

Usage Notes

1. Each record from the CP source directory is matched against a pattern string. The pattern string consists of up to 13 blank-delimited tokens. The pattern string tokens are matched up against blank-delimited tokens from each record of the CP directory, and matching records are returned to the caller.
2. If all tokens in the pattern match the corresponding tokens in the directory record, then the directory record is considered a match.
3. An asterisk (*) may be used as a wildcard character in the pattern. Any number of asterisks may appear in a token of the pattern. An asterisk is considered to match any number of characters (including zero characters) in the corresponding token of the directory record. An asterisk can be used alone as a wild card indicating that all values found in that position in the record are matches.
4. Any pattern token consisting only of wild card characters (for instance, '*****') is treated the same as a token consisting of a single asterisk.
5. A pattern consisting of a single asterisk returns all comment records (not all records).
6. A scan pattern consisting of nothing but wild card designators in the form '* * * * * * * * * * * * *' is equivalent to '*'.
7. Special processing occurs when evaluating the first token of a directory record with the first token of the pattern. The first token is the directory statement type. The first token of a directory record may be an abbreviation of the directory statement type. If so, it is expanded out to the full, non-abbreviated statement type. For example, "I" is expanded to "IPL", or "IN" is expanded to "INCLUDE". Similarly, if the first token of the pattern does not contain asterisks, and is an abbreviation of a valid directory statement type, it also is expanded out to the full unabbreviated statement type.
8. Scan results may cause sensitive information (logon and minidisk passwords) to be sent to the requestor.
9. The search parameter list provided is uppercased and compared to uppercased directory records. This must be considered when attempting to scan for directory statements that allow mixed case arguments such as POSIXGLIST, POSIXGROUP and POSIXINFO statements.
10. See the "Creating and Updating a User Directory" chapter in *z/VM: CP Planning and Administration* for more information on the directory format and on specific directory statements.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
		28	RS_NONE_FOUND	No matching entries found. Return buffer is empty.
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter pp

RC	RC Name	RS	RS Name	Description
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		12	RS_AUTHERR_DM	Request not authorized by directory manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available
592	RCERR_ASYNC_DM	nnnn	opid	Asynchronous operation started - product-specific asynchronous operation ID (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
596	RCERR_INTERNAL_DM	nnnn	psrc	Internal directory manager error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Directory_Manager_Task_Cancel_DM

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
operation_id

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Directory_Manager_Task_Cancel_DM to cancel a specific asynchronous task being performed by the directory manager.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 32.

function_name
(string,32,char43) The API function name – in this case, 'Directory_Manager_Task_Cancel_DM'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See “Client Authentication” on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) This must match an entry in the authorization file that also contains the *authenticated_userid* and the *function_name* (Directory_Manager_Task_Cancel_DM).

operation_id
(int4; range 0-2147483647) The identifier of the task.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter pprr
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		12	RS_AUTHERR_DM	Request not authorized by directory manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
452	RCERR_TASK	4	RS_NOT_FOUND	Task not found
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available

RC	RC Name	RS	RS Name	Description
592	RCERR_ASYNC_DM	nnnn	opid	Asynchronous operation started - product-specific asynchronous operation ID (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
596	RCERR_INTERNAL_DM	nnnn	psrc	Internal directory manager error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Event_Stream_Add

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
event_info

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Event_Stream_Add to add an event to the event stream.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 16.

function_name
(string,16,char43) The API function name – in this case, 'Event_Stream_Add'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See “Client Authentication” on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) This must match an entry in the authorization file that also contains the *authenticated_userid* and the *function_name* (Event_Stream_Add).

event_info
(string,1-maxlength,charNA) Data to be added to the event stream. Note that the first 4 bytes are an int4 *event_type*, and that values 0-16777215 are reserved for IBM use.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
8	RC_ERR	999	RS_NOT_AVAILABLE	Function not available
24	RCERR_SYNTAX	*pprr*	*pprr*	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
396	RCERR_INTERNAL	nnnn	psrc	Internal system error - product-specific return code (See "Internal Return Codes (RC = 396, 592, or 596)" on page 834)

RC	RC Name	RS	RS Name	Description
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Event_Subscribe

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
match_key_length
match_key

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length (error only)
request_id (error only)
return_code (error only)
reason_code (error only)

See Usage Note "1" on page 112.

Purpose

Use Event_Subscribe to arrange to be notified of events of interest. The events will be sent on this connection – see Usage Note "1" on page 112 for more information.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 15.

function_name
(string,15,char43) The API function name – in this case, 'Event_Subscribe'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) This must match an entry in the authorization file that also contains the *authenticated_userid* and the *function_name* (Event_Subscribe).

match_key_length
(int4) Length of *match_key* (which is optional, so this value may be 0).

match_key
(string,0-16M,charNA) Binary match key, either exact or fuzzy, to be used for determining which events are to be seen. See Usage Note "2" on page 113.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

Note: See Usage Note "1" on page 112.

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Usage Notes

1. When using this API, you will always receive the immediate request verification (**Response 1**, above). If errors are encountered, you will receive the normal set of output parameters (**Response 2**, above).

   If there are *no* errors encountered, then you will *not* receive the normal set of output parameters. Instead, the output data will be returned in multiple socket reads, each set of data consisting of the following:

   a. Length of data (4 bytes)

   b. Type of data (4 bytes), as follows:

   0
   Means that format 0 data will follow, as described in the "VM Event System Service" chapter in CP Programming Services. Note that there may be multiple events in this data. Use the data length to navigate to the next event.

1
Means that format 1 data will follow in a single event. The data will have a 4-byte ID followed by whatever data is left over in the buffer. Note that IDs 0-16777215 are reserved for IBM use.

c. The actual data, of the type as described above

You'll continue to receive another format 0 or 1 event in the subsequent socket reads, repeating until an error is encountered or until you unsubscribe. (See "Event_Unsubscribe" on page 115). Note the output data may not all be returned immediately, and may keep coming in asynchronously as long as you are subscribed. It will continue to be returned in the same socket, so make sure to use a different socket for any other API calls made while the data is still being received.

The list of events produced by the VM Event System Service (*VMEVENT) can be found in the "VM Event System Service (*VMEVENT)" chapter in *z/VM: CP Programming Services*. There are other events *not* produced by *VMEVENT, as follows (note that all are format 1):

- Type 2, Performance threshold hit (see "System_Performance_Threshold_Enable" on page 583)
- Type 500, Async directory update complete (see "Image_Definition_Async_Updates" on page 161)
- Type 2008, Processing of a dump completed (see "Process_ABEND_Dump" on page 422)
- Type 2009, Deletion of a dump completed (see "Delete_ABEND_Dump" on page 85)
- Type 2010, Automated processing of a dump completed (see "Process_ABEND_Dump" on page 422).

2. A match key can be either exact or fuzzy, as follows:

Exact match key
The match key is exact if it contains no wildcard characters. Message keys against which the match key is compared must match the match key exactly (same length, same data) for the requested operation to have effect.

Fuzzy match key
The match key is fuzzy if it contains wildcard characters. Message keys against which the match key is compared must match the pattern specified by the match key, allowing for wildcards, for the requested operation to have effect.

The allowable wildcard characters are * (asterisk), % (percent), and ' (apostrophe). They are interpreted in a similar way as the wildcard characters in CMS file names and file types. To be more precise, these wildcard characters have the following meanings:

% (X'6C')
Matches any single character in a message key. For example, match key a%c matches message keys `abc`, `acc`, and `axc`.

*** (X'5C')**
Matches a variable-length (zero or more characters) substring within the message key. This usually means that the match key is actually a series of fragments, all of which must be present in the message key for a match to occur, but that the spacing between the fragments is irrelevant. For example, message key `abcde` is matched by match keys `a*`, `*de`, `a*e`, and `*a*b*c*d*e*`.

' (X'7D')
Indicates that the next character in the match key should be interpreted literally (that is, without regard to whether it is a wildcard character or not). A character performing this function is commonly called an escape character.

Note that a match key may contain more than one kind of wildcard character. For example, message key `abcdefg` is matched by match key `*b%d*`.

If neither *match_key_length* nor *match_key* are specified, Event_Subscribe will deliver all possible events.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
8	RC_ERR	13	RS_INVALID_KEY	Match key length does not match the match key specified
24	RCERR_SYNTAX	*pprr*	*pprr*	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
396	RCERR_INTERNAL	nnnn	psrc	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Event_Unsubscribe

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

None. See Usage Note "1" on page 116.

Purpose

Use Event_Unsubscribe to end asynchronous notification of events of interest.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 17.

function_name
(string,17,char43) The API function name – in this case, 'Event_Unsubscribe'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) This must match an entry in the authorization file that also contains the *authenticated_userid* and the *function_name* (Event_Unsubscribe).

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

None. See Usage Note "1" on page 116.

Usage Notes

1. When using this API, you will always receive the immediate request verification (**Response 1**, above), but unlike other APIs you will receive no further output parameters.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RC_NONE	Request successful
24	RCERR_SYNTAX	*pprr*	*pprr*	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
396	RCERR_INTERNAL	nnnn	psrc	Internal system error - product-specific return code (See "Internal Return Codes (RC = 396, 592, or 596)" on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Image_Activate

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
activated
not_activated
failing_array_length
failing_array (1)
failing_structure (2)
failing_structure_length
image_name_length
image_name
return_code
reason_code

Note:

1. An array consists of zero or more of its components.
2. A structure consists of one set of its components.

Purpose

Use Image_Activate to activate a virtual image or list of virtual images.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 14.

function_name
(string,14,char43) The API function name – in this case, 'Image_Activate'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
One of the following:

- (string,1-8,char42) The name of the image being activated.
- (string,1-64,char43) The name of a list containing names of images to be activated.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

activated
(int4) The number of images activated successfully.

not_activated
(int4) The number of images *not* activated successfully.

failing_array_length
(int4) Length of *failing_array*.

failing_array
(array) An array consisting of zero or more instances of *failing_structure* for every image that failed, as follows:

failing_structure
(structure) A structure consisting of one set of the following parameters:

failing_structure_length
(int4) The combined length of the remaining parameters in *failing_structure* (not including this parameter).

image_name_length
(int4) Length of *image_name*.

image_name
(string,1-8,char42) The name of the image.

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Usage Notes

1. This function checks the name to determine whether it is a list, and if not, processes the name as a single image name. Therefore, lists should be given names that cannot be confused with image names.
2. During authorization checking and function processing, name lists are only expanded once; although a name within a list may also be the name of a list, the second (nested) list will not be expanded.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
200	RCERR_IMAGEOP	4	RS_NOT_FOUND	Image not found
		8	RS_ALREADY_ACTIVE	Image already active
		16	RS_BEING_DEACT	Image being deactivated
		28	RS_NOT_ALL	Some images in list not activated
396	RCERR_INTERNAL	nnnn	psrc	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist

RC	RC Name	RS	RS Name	Description
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Image_Active_Configuration_Query

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
memory_size
memory_unit
share_type
share_value_length
share_value
number_CPUs
CPU_info_array_length
CPU_info_array (1)
CPU_info_structure (2)
CPU_info_structure_length
CPU_number
CPU_id_length
CPU_id
CPU_status
device_info_array_length
device_info_array (1)
device_info_structure (2)
device_info_structure_length
device_type
device_address_length
device_address

Note:

1. An array consists of zero or more of its components.
2. A structure consists of one set of its components.

Purpose

Use Image_Active_Configuration_Query to obtain current configuration information for an active virtual image.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 32.

function_name
(string,32,char43) The API function name – in this case, 'Image_Active_Configuration_Query'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The userid being queried.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

memory_size
(int4) Size of memory, in units as per *memory_unit*.

memory_unit
(int1) One of the following:

1
KB

2
MB

3
GB

share_type
(int1) Allowed values are:

1
Relative

2
Absolute

share_value_length
(int4) Length of *share_value*.

share_value
(string,1-5,char10 plus .) For a relative share, this value is a number from 1 to 10000, indicating the amount of scheduled system resources available minus the amount allocated to absolute share users.

For an absolute share, this value is a decimal real number from 0.1 to 100, indicating (by percentage) your share of system resources which includes CPU, storage, and paging capacity.

number_CPUs
(int4) Number of CPUs active.

CPU_info_array_length
(int4) Length of *CPU_info_array*.

CPU_info_array
(array) An array consisting of zero or more instances of *CPU_info_structure*, as follows:

CPU_info_structure
(structure) A structure consisting of one set of the following parameters:

CPU_info_structure_length
(int4) The combined length of the remaining parameters in *CPU_info_structure* (not including this parameter).

CPU_number
(int4) CPU number.

CPU_id_length
(int4) Length of *CPU_id*.

CPU_id
(string,1-16,char16) CPU ID (example: FF319B9E20948000)

CPU_status
(int1) Allowed values are:

1
Base

2
Stopped

3
Check-stopped

4
Non-base, active

device_info_array_length
(int4) Length of *device_info_array*.

device_info_array
(array) An array consisting of zero or more instances of *device_info_structure*, as follows:

device_info_structure
(structure) A structure consisting of one set of the following parameters:

device_info_structure_length
(int4) The combined length of the remaining parameters in *device_info_structure* (not including this parameter).

device_type
(int1) Allowed values are:

1
CONS

2
RDR

3
PUN

4
PRT

5
DASD

device_address_length
(int4) Length of *device_address*.

device_address
(string,4,char16) The 4-digit device address.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter pprr
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
200	RCERR_IMAGEOP	12	RS_NOT_ACTIVE	Image not active
396	RCERR_INTERNAL	nnnn	psrc	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory

RC	RC Name	RS	RS Name	Description
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Image_Console_Get

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Image_Console_Get to put the most recent console spool file of the target into the reader of the issuer.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 18.

function_name
(string,26,char43) The API function name – in this case, 'Image_Console_Get'

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See “Client Authentication” on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) This must match an entry in the authorization file that also contains the *authenticated_userid* and the *function_name* (Image_Console_Get).

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Usage Notes

1. The console spool file has a class of T, and an origin ID of the target. Obtaining this console spool file is destructive; that is, once you obtain it, no one else can ever obtain this particular console spool file.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
8	RC_ERR	8	RS_NOT_EXIST	No spool file available

Image_CPU_Define

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
cpu_address_length
cpu_address
cpu_type

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Image_CPU_Define to add a virtual processor to an active virtual image's configuration.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 16.

function_name
(string,21,char43) The API function name – in this case, 'Image_CPU_Define'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The name of the virtual image for which to define a virtual CPU.

cpu_address_length
(int4) Length of *cpu_address*.

cpu_address
(string,1-2,char16) The virtual CPU address to add to the virtual image (in the hexadecimal range of 0-3F).

cpu_type
(int1) The type of processor to add, as follows:

0
Unspecified

1
CP

2
IFL

4
ZIIP

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
		4	RS_AFFINITY_SUPPRESSED	CPU defined, but CPU affinity suppressed

RC	RC Name	RS	RS Name	Description
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
216	RCERR_IMAGECPU	2	RS_INVALID_DEVICE	Input virtual CPU value out of range
		12	RS_NOT_ACTIVE	Image not active
		24	RS_VCPU_ALREADY_EXISTS	Virtual CPU already exists
		28	RS_VCPU_OUT_OF_RANGE	Virtual CPU address beyond allowable range defined in directory
		40	RS_TYPE_NOT_SUPPORTED	Processor type not supported on your system
396	RCERR_INTERNAL	nnnn	psrc	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Image_CPU_Define_DM

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
cpu_address_length
cpu_address
base_cpu
cpuid_length
cpuid
dedicate_cpu
crypto

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Image_CPU_Define_DM to add a virtual processor to a virtual image's directory entry.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 19.

function_name
(string,21,char43) The API function name – in this case, 'Image_CPU_Define_DM'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).

- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The name of the virtual image for which to statically define a virtual CPU.

cpu_address_length
(int4) Length of *cpu_address*.

cpu_address
(string,1-2,char16) The virtual CPU address to add to the static definition of the virtual image (in the hexadecimal range of 0-3F).

base_cpu
(int1) Whether this CPU defines the base virtual processor, as follows:

0
Unspecified

1
BASE

Note: If BASE is not specified for any static virtual CPU, the base virtual processor will be the lowest virtual processor address.

cpuid_length
(int4) Length of *cpuid*.

cpuid
(string,0-6,char16) The processor identification number to be stored in bits 8 through 31 of the CPU ID, returned in response to the store processor ID (STIDP) instruction.

dedicate_cpu
(int1) Specifies whether the virtual processor is to be dedicated at LOGON time to a real processor, as follows:

0
Unspecified

1
NODEDICATE

2
DEDICATE

Note: This parameter is allowed (but ignored) for compatibility reasons.

crypto
(int1) Specifies whether the virtual Cryptographic Coprocessor Facility (CCF) should be defined automatically for the virtual CPU at LOGON time, as follows:

0
Unspecified (no CRYPTO)

1
CRYPTO

Note: Although the CCF is no longer supported, this parameter is allowed (but ignored) for compatibility reasons and must be accounted for in the overall input parameter length specifications. If specified as 1, an RC=520/RS=45 error code (RS_CRYPTO_NOT_INSTALLED) will be received.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter pprr
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		12	RS_AUTHERR_DM	Request not authorized by directory manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available
520	RCERR_CPU_DM	24	RS_ONLY1_BASE_ALLOWED	Only one base CPU may be defined
		28	RS_CPU_OUT_OF_RANGE	Input virtual CPU value out of range
		32	RS_MAX_EXCEEDED	Maximum allowable number of virtual CPUs is exceeded
		45	RS_CRYPTO_NOT_INSTALLED	The Cryptographic Coprocessor Facility (CCF) is not installed on this system

RC	RC Name	RS	RS Name	Description
592	RCERR_ASYNC_DM	nnnn	opid	Asynchronous operation started - product-specific asynchronous operation ID (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
596	RCERR_INTERNAL_DM	nnnn	psrc	Internal directory manager error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Image_CPU_Delete

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
cpu_address_length
cpu_address

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Image_CPU_Delete to delete a virtual processor from an active virtual image's configuration.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 16.

function_name
(string,21,char43) The API function name – in this case, 'Image_CPU_Delete'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The name of a virtual image for which a virtual CPU will be deleted.

cpu_address_length
(int4) Length of *cpu_address*.

cpu_address
(string,1-2,char16) The virtual CPU address to delete from the virtual image (in the hexadecimal range of 0-3F).

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Usage Notes

1. The successful completion of this function will result in a system restart, and the virtual image will require a re-IPL (image activation).

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid

RC	RC Name	RS	RS Name	Description
216	RCERR_IMAGECPU	2	RS_INVALID_DEVICE	Input virtual CPU value out of range
		4	RS_NOT_FOUND	Virtual CPU not found
		12	RS_NOT_ACTIVE	Image not active
396	RCERR_INTERNAL	nnnn	psrc	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Image_CPU_Delete_DM

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
cpu_address_length
cpu_address

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Image_CPU_Delete_DM to delete a virtual processor from a virtual image's directory entry.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 19.

function_name
(string,21,char43) The API function name – in this case, 'Image_CPU_Delete_DM'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The name of the virtual image from which to statically delete a virtual CPU.

cpu_address_length
(int4) Length of *cpu_address*.

cpu_address
(string,1-2,char16) The virtual CPU address to delete from the static definition of the virtual image (in the hexadecimal range of 0-3F).

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Usage Notes

1. The successful completion of this function will result in a system restart, and the virtual image will require a re-IPL (image activation).

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter pprr
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		12	RS_AUTHERR_DM	Request not authorized by directory manager
		16	RS_AUTHERR_SERVER	Request not authorized by server

RC	RC Name	RS	RS Name	Description
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available
520	RCERR_CPU_DM	28	RS_CPU_OUT_OF_RANGE	Input virtual CPU value out of range
		30	RS_CPU_NOT_FOUND	CPU not found
592	RCERR_ASYNC_DM	nnnn	opid	Asynchronous operation started - product-specific asynchronous operation ID (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
596	RCERR_INTERNAL_DM	nnnn	psrc	Internal directory manager error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Image_CPU_Query

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
number_CPUs
CPU_info_array_length
CPU_info_array (1)
CPU_info_structure (2)
CPU_info_structure_length
CPU_address
CPU_id_length
CPU_id
CPU_base
CPU_status
CPU_type

Note:

1. An array consists of zero or more of its components.
2. A structure consists of one set of its components.

Purpose

Use Image_CPU_Query to query the virtual processors in an active virtual image's configuration.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 15.

function_name
(string,21,char43) The API function name – in this case, 'Image_CPU_Query'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The name of the virtual image whose virtual CPUs are being queried.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

number_CPUs
(int4) Number of CPUs active.

CPU_info_array_length
(int4) Length of *CPU_info_array*.

CPU_info_array
(array) An array consisting of zero or more instances of *CPU_info_structure*, as follows:

CPU_info_structure
(structure) A structure consisting of one set of the following parameters:

CPU_info_structure_length
(int4) The combined length of the remaining parameters in *CPU_info_structure* (not including this parameter).

CPU_address
(int4) CPU address.

CPU_id_length
(int4) Length of *CPU_id*.

CPU_id
(string,16,char16) CPU ID (for example: FF319B9E20948000).

CPU_base
(int1) Whether this CPU defines the base virtual processor, as follows:

1
BASE

2
Not BASE

CPU_status
(int1) The CPU status, as follows:

1
Stopped

2
Check-stopped

3
Soft-stopped or active

CPU_type
(int1) The CPU type, as follows:

1
CP

2
IFL

4
ZIIP

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
216	RCERR_IMAGECPU	12	RS_NOT_ACTIVE	Image not active

RC	RC Name	RS	RS Name	Description
396	RCERR_INTERNAL	nnnn	psrc	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Image_CPU_Query_DM

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
cpu_address_length
cpu_address

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
cpu_address_length
cpu_address
base_cpu
cpuid_length
cpuid
dedicate_cpu
crypto

Purpose

Use Image_CPU_Query_DM to query a virtual processor in a virtual image's directory entry.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 18.

function_name
(string,21,char43) The API function name – in this case, 'Image_CPU_Query_DM'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The name of the virtual image from which to query a virtual CPU.

cpu_address_length
(int4) Length of *cpu_address*.

cpu_address
(string,1-2,char16) The virtual CPU address to query from the static definition of the virtual image (in the hexadecimal range of 0-3F).

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

cpu_address_length
(int4) Length of *cpu_address*.

cpu_address
(string,2,char16) The existing virtual CPU address (in the hexadecimal range of 0-3F).

base_cpu
(int1) Whether this CPU defines the base virtual processor, as follows:

0
Unspecified

1
BASE

Note: If BASE is not specified for any static virtual CPU, the base virtual processor will be the lowest virtual processor address.

cpuid_length
(int4) Length of *cpuid*.

cpuid
(string,6,char16) The processor identification number that was stored in bits 8 through 31 of the CPU ID, returned in response to the store processor ID (STIDP) instruction.

dedicate_cpu
(int1) Whether the virtual processor was dedicated at LOGON time to a real processor, as follows:

0
Unspecified

1
NODEDICATE

2
DEDICATE

Note: This parameter is allowed for compatibility reasons. A value of 0 (Unspecified) is always returned.

crypto
(int1) Whether the virtual Cryptographic Coprocessor Facility (CCF) should be defined automatically for the virtual CPU at LOGON time, as follows:

0
Unspecified (no CRYPTO)

1
CRYPTO

Note: The CCF is no longer supported, but this parameter is allowed for compatibility reasons. A value of 0 (Unspecified) is always returned.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter pprr
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		12	RS_AUTHERR_DM	Request not authorized by directory manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available
520	RCERR_CPU_DM	28	RS_CPU_OUT_OF_RANGE	Input virtual CPU value out of range

RC	RC Name	RS	RS Name	Description
		30	RS_CPU_NOT_FOUND	CPU not found
592	RCERR_ASYNC_DM	nnnn	opid	Asynchronous operation started - product-specific asynchronous operation ID (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
596	RCERR_INTERNAL_DM	nnnn	psrc	Internal directory manager error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Image_CPU_Set_Maximum_DM

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
max_cpu

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Image_CPU_Set_Maximum_DM to set the maximum number of virtual processors that can be defined in a virtual image's directory entry.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 24.

function_name
(string,21,char43) The API function name – in this case, 'Image_CPU_Set_Maximum_DM'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The name of the virtual image for which to set the maximum number of virtual processors.

max_cpu
(int4) The maximum number of virtual processors the user can define. The number must be between 1 and 64 (decimal).

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Usage Notes

1. If the maximum number of virtual processors that can be defined in a virtual image's directory entry is *not* defined using this API, the default value will be either 1 or the number of CPU statements for the image, whichever is greater.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter pprr
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		12	RS_AUTHERR_DM	Request not authorized by directory manager
		16	RS_AUTHERR_SERVER	Request not authorized by server

RC	RC Name	RS	RS Name	Description
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
400	RCERR_IMAGEDEF	4	RS_NOT_FOUND	Image or profile definition not found
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available
592	RCERR_ASYNC_DM	nnnn	opid	Asynchronous operation started - product-specific asynchronous operation ID (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
596	RCERR_INTERNAL_DM	nnnn	psrc	Internal directory manager error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Image_Create_DM

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
prototype_name_length
prototype_name
initial_password_length
initial_password
initial_account_number_length
initial_account_number
image_record_array_length
image_record_array (1)
image_record_structure (2)
image_record_length
image_record

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
operation_id

Note:

1. An array consists of zero or more of its components.
2. A structure consists of one set of its components.

Purpose

Use Image_Create_DM to define a new virtual image in the directory.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 15.

function_name
(string,15,char43) The API function name – in this case, 'Image_Create_DM'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The name of the image to be created.

prototype_name_length
(int4) Length of *prototype_name*.

prototype_name
(string,0-8,char42) The prototype to use for creating the image.

Note: If both the *prototype_name* and *image_record_array* parameters are specified, then the *prototype_name* will be used and the *image_record_array* parameter will be ignored.

initial_password_length
(int4) Length of *initial_password*. If the length is 0, the password will default to "NOLOG".

initial_password
(string,0-200,charNA) The logon password to be assigned initially to the virtual image being created.

Note: This parameter (along with *initial_account_number*) may not be specified if *image_record_array* is specified.

initial_account_number_length
(int4) Length of *initial_account_number*. If the length is 0, the account number will default to the value specified in the prototype (if any), or to the value specified in the included profile (if any), or to the image name.

initial_account_number
(string,0-8,charNB) The account number to be assigned initially to the virtual image being created.

Note: This parameter (along with *initial_password*) may not be specified if *image_record_array* is specified. See also Usage Note "3" on page 154.

image_record_array_length
(int4) Length of *image_record_array*.

image_record_array
(array) An array consisting of zero or more instances of *image_record_structure*, as follows:

image_record_structure
(structure) A structure consisting of one set of *image_record_length* and *image_record*, as follows:

image_record_length
(int4) Length of *image_record*.

image_record
(string,1-72,charNA) The user or profile entry.

Note:

1. If both the *prototype_name* and *image_record_array* parameters are specified, then the *prototype_name* will be used and the *image_record_array* parameter will be ignored.
2. Neither the *initial_password* nor the *initial_account_number* input parameters may be specified if *image_record_array* is specified.
3. If you are using IBM DirMaint as your directory manager and you specify a *initial_password* longer than 8 characters, you will receive an internal directory manager error (RC=596, RS=1203).

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

operation_id
(int4; range -1-2147483647) The identifier of the task. If the operation is asynchronous and has not completed, *return_code* will be 592, *reason_code* will be 0, and *operation_id* will be in the range 0-2147483647. If the operation is complete, *operation_id* will be -1.

Usage Notes

1. See the "Creating and Updating a User Directory" chapter in *z/VM: CP Planning and Administration* for more information on the directory format and on specific directory statements.
2. If both the *prototype_name* and *image_record_array* parameters are specified, then the *prototype_name* will be used and the *image_record_array* parameter will be ignored.
3. Neither the *initial_password* nor the *initial_account_number* input parameters may be specified if *image_record_array* is specified.
4. Use Image_Create_DM to create a USER directory ENTRY, or an IDENTITY directory entry with *no* BUILD statements. An error will result from an attempt to define a PROFILE or SUBCONFIG entry, or an IDENTITY entry with BUILD statements.

   Use Image_Definition_Create_DM to create an IDENTITY with a SUBCONFIG by specifying the SYSTEM_UNIQUE= keyword, or use Profile_Create_DM to create a PROFILE entry.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
		8	RS_OFFLINE	Request successful; object directory offline
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		12	RS_AUTHERR_DM	Request not authorized by directory manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
400	RCERR_IMAGEDEF	8	RS_NAME_EXISTS	Image name already defined
		20	RS_NOT_DEFINED	Image prototype is not defined
		40	RS_MULTIPLE	Multiple user statements
408	RCERR_IMAGEDISKD	24	RS_NO_SPACE	Requested image disk space not available
420	RC_DASD_DM	8	RS_IVS_NAME_NOT_USED	Group, region, or volume name is not defined
436	RCERR_PROFILED	4	RS_NOT_FOUND	Profile included not found
		40	RS_MULTIPLE	Multiple profiles included
444	RCERR_POLICY_PW	0	RS_NONE	Password policy error
448	RCERR_POLICY_ACCT	0	RS_NONE	Account policy error
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available
		20	RS_PW_FORMAT_NOT _SUPPORTED	Password format not supported
592	RCERR_ASYNC_DM	0	RS_NONE	Asynchronous operation started
596	RCERR_INTERNAL_DM	nnnn	psrc	Internal directory manager error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory

RC	RC Name	RS	RS Name	Description
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Image_Deactivate

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
force_time_length
force_time

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
deactivated
not_deactivated
failing_array_length
failing_array (1)
failing_structure (2)
failing_structure_length
image_name_length
image_name
return_code
reason_code

Note:

1. An array consists of zero or more of its components.
2. A structure consists of one set of its components.

Purpose

Use Image_Deactivate to stop a virtual image or list of virtual images. The virtual image(s) will no longer be active on the system.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 16.

function_name
(string,16,char43) The API function name – in this case, 'Image_Deactivate'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See “Client Authentication” on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See “Client Authentication” on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
One of the following:

- (string,1-8,char42) The name of the image being deactivated.
- (string,1-64,char43) The name of a list containing names of images to be deactivated.

force_time_length
(int4) Length of *force_time*.

force_time
(string,0-12,char42) Specifies when the Image_Deactivate function is to take place. If unspecified, deactivation takes place according to the default signal timeout value set for the system. Valid inputs are:

IMMED
Immediate image deactivation

WITHIN *interval*
Where *interval* is a number of seconds in the the range 1–65535 (see Usage Note “4” on page 159)

BY *time*
Where *time* is specified as hh:mm or hh:mm:ss

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

deactivated
(int4) The number of images deactivated successfully.

not_deactivated
(int4) The number of images *not* deactivated successfully.

failing_array_length
(int4) Length of *failing_array*.

failing_array
(array) An array consisting of zero or more instances of *failing_structure* for every image that failed, as follows:

failing_structure
(structure) A structure consisting of one set of the following parameters:

failing_structure_length
(int4) The combined length of the remaining parameters in *failing_structure* (not including this parameter).

image_name_length
(int4) Length of *image_name*.

image_name
(string,1-8,char42) The name of the image.

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Usage Notes

1. This function checks the name to determine whether it is a list, and if not, processes the name as a single image name. Therefore, lists should be given names that cannot be confused with image names.
2. During authorization checking and function processing, name lists are only expanded once; although a name within a list may also be the name of a list, the second (nested) list will not be expanded.
3. Use of IMAGE_DEACTIVATE is intended for z/VM guests that enable SIGNAL SHUTDOWN. The z/VM system configuration setting for SHUTDOWNTIME and SIGNAL SHUTDOWNTIME should be set to allow sufficient time for all guests to complete their graceful, "pre-power-off" processing. If the SIGNAL SHUTDOWN is not successful, a CP FORCE is issued against the image. See *z/VM: CP Planning and Administration* and *z/VM: CP Commands and Utilities Reference* for more on shutdown timeout values.
4. If the image is enabled for SIGNAL, the valid range for the *force_time* WITHIN interval is 1–32767.
5. If no default value has otherwise been set, force_time defaults to "within 600".

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
		secs	secs	Request successful; Image Deactivated Within *secs* Seconds
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
200	RCERR_IMAGEOP	12	RS_NOT_ACTIVE	Image not active
		16	RS_BEING_DEACT	Image being deactivated
		32	RS_SOME_NOT_DEACT	Some images in list not deactivated
		36	RS_TIME_NOT_VALID	Specified time results in interval greater than max allowed
396	RCERR_INTERNAL	nnnn	psrc	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Image_Definition_Async_Updates

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
`enabled=`*value*

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Image_Definition_Async_Updates to change the completion notification for the following APIs:

- “Image_Definition_Update_DM” on page 190
- “Image_Definition_Delete_DM” on page 175
- “Image_Definition_Create_DM” on page 164

Note:

1. By default, these APIs are synchronous, meaning the caller's thread of execution will block until the issued API completes. Use Image_Definition_Async_Updates with the ENABLE=YES option to change these APIs to be asynchronous.
2. Issued synchronously, the output parameters from the above APIs contain the actual response data (RC/RS code and applicable responses). Issued asynchronously, the actual response data is provided in a completion event.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 30.

function_name
(string,30,char43) The API function name – in this case, 'Image_Definition_Async_Updates'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The userid for which ASYNC notifications are being changed.

Note: The format for specifying the following additional input parameter is *parameter_name=value*, followed by a null (ASCIIZ) terminator. For more information, see "Key-value pair (KVP) input parameters" on page 51.

enabled=*value*
(string,0-3,char26) One of the following:

YES
Asynchronous

NO
Synchronous

If unspecified, NO is the default.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Usage Notes

1. The expected return code from the asynchronous APIs is RC=592 (RCERR_ASYNC_DM), RS=4 (RS_WORK_OUTSTANDING).
2. The `enabled=`*value* setting will remain in effect until a subsequent call to this API.
3. This API only affects the userid identified by the *authenticated_userid* parameter.
4. A file named *authenticated_userid* ASYNCH will be saved on the SMAPI server A disk. (Note that the A disk for all SMAPI servers is a shared file system directory – by default, VMSYS:VSMWORK1.DATA). The presence of this file indicates `enabled=`YES for this userid.
5. After the operation(s) are complete, the actual return code(s) and data are put on the *VMEVENT queue using the Event_Stream_Add API. (See “Event_Stream_Add” on page 108. This will be format 1 data.) The first 4 bytes will represent an event type value of 500. The remaining event data will be the same as that shown in "Response 2 – Output Parameters" for each API (“Image_Definition_Update_DM” on page 190, “Image_Definition_Delete_DM” on page 175, and “Image_Definition_Create_DM” on page 164).
6. Syntax errors (RC = 24 and RS = *pprr*) are only applicable to the common input parameters. Syntax checking is *not* performed on the additional input parameters for this API. If a valid parameter is specified multiple times, the last value specified for that parameter will be used. For more information, see “Key-value pair (KVP) input parameters” on page 51.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
8	RC_ERR	24	RS_UPDATE_WRITE_ERROR	Unable to write ASYNCH file
		3003	RS_INVALID_OPERAND	Invalid parameter operand
		3004	RS_MISSING_PARAMETER	Required parameter missing
24	RCERR_SYNTAX	*pprr*	*pprr*	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
396	RCERR_INTERNAL	*nnnn*	*psrc*	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Image_Definition_Create_DM

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
definition_create_directory_keyword_parameter_list_length
definition_create_directory_keyword_parameter_list

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
asynch_length (if asynchronous operation started)
asynch_data (if asynchronous operation started)
error_length (error only)
error_data (error only)

Purpose

Use Image_Definition_Create_DM to create a new virtual machine directory entry for a particular system.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 26.

function_name
(string,26,char43) The API function name – in this case, 'Image_Definition_Create_DM'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The name of the image/subconfig being created.

definition_create_directory_keyword_parameter_list_length
(int4) Length of *definition_create_directory_keyword_parameter_list*.

definition_create_directory_keyword_parameter_list
(string,1-maxlength,charNA) The remaining set of *directory_keyword_parameter*= input parameters.

The format for specifying the following additional input parameters is either *directory_keyword_parameter*= followed by a blank-delimited series of *directory_keyword_operand=directory_keyword_operand_value* pairs, or *directory_keyword_parameter=directory_keyword_parameter_value*, in both cases followed by a null (ASCIIZ) terminator. They may be specified in any order. For more information, see "Key-value pair (KVP) input parameters" on page 51.

For example:

```
DEDICATE=VDEV=234 RDEV=234 USERACCESSID=FRED'00'x IPL=VDEV=CMS'00'x
   ('00'x = null terminator)
```

Table 2 on page 165 shows the keywords and values that can be specified for each type of directory entry, and note how keywords can be always optional, always required, or required only if certain other conditions are true. See *z/VM: CP Planning and Administration* for more information on how these directory entries work.

Table 2. Input Keywords and Values for Image_Definition_Create_DM

directory_keyword_parameter=	**Blank-delimited *directory_keyword_operand=directory_keyword_operand_value* pairs, OR *directory_keyword_parameter_value***
COMMAND_DEFINE_CPU=	• CPUADDR='*cpuaddr*' (can be any CPU address range, as described in *z/VM: CP Planning and Administration*) (**Required**) • TYPE=CP | IFL | ZIIP | ICF (Optional – if no type is specified when defining a new CPU, it defaults to the type of the primary virtual CPU.)
COMMAND_SET_CPUAFFINITY=	• CPUAFFINITY=ON | OFF (**Required**) • USERID=*userid* or * (Optional)

Table 2. Input Keywords and Values for Image_Definition_Create_DM (continued)	
directory_keyword_parameter=	**Blank-delimited *directory_keyword_operand=directory_keyword_operand_value* pairs, OR *directory_keyword_parameter_value***
COMMAND_SET_SHARE=	• USERID=*userid* (**Required**) • TYPE=ALL | CP | ZIIP | IFL | ICF (Optional, default is ALL) • OPERAND= INITIAL | ABSOLUTE | RELATIVE | NOLIMIT | LIMITSOFT | LIMITHARD (**Required**) • ABSOLUTE=*y*% (**Required** if OPERAND=ABSOLUTE) • RELATIVE=*z* (**Required** if OPERAND=RELATIVE) • ABSOLUTE_MAX=*a*% (Optional) • RELATIVE_MAX=*b* (Optional) • LIMIT=NOLIMIT | LIMITSOFT | LIMITHARD (Optional, default is NOLIMIT)
COMMAND_SET_VCONFIG=	• MODE=ESA390 | LINUX | VM (**Required**)
CONSOLE=	• VDEV=*vdev* (**Required**) • DEVTYPE=*devtype* (**Required**) • CLASS=T | *class* (Optional, default is T) • USERID=*userid* (**Required** if OBSERVER=YES is also specified, otherwise optional) • OBSERVER=YES | NO (Optional, default is NO)
CPU=	• CPUADDR=*cpuaddr* (**Required**) • BASE=YES | NO (Optional, default is NO) • CPUID=*cpuid* (Optional) • DEDICATE=YES | NO (Optional, default is NO) • CRYPTO=YES | NO (Optional, default is NO)
CPU_MAXIMUM=	• COUNT=*mcpu* (Optional, default is 1) • TYPE=ESA | XA | XC (**Required**) **Note:** A MACHINE statement will be created/updated with the information specified.
DEDICATE=	• VDEV=*vdev* (**Required**) • RDEV=*rdev* (**Required** for tape and "other". DASD must have either RDEV=, VOLID=, or both.) • VOLID=*volid* (**Required** for DASD, if no RDEV= was specified.) • R/O=YES | NO (Optional, default is NO) • USERACCESSID=*userid* (Optional) • USERTYPE=SINGLEUSER | MULTIUSER (Optional, default is SINGLEUSER for tape) • ASSIGN=NO (Optional) • QIOASSIST=NO (Optional)

Table 2. Input Keywords and Values for Image_Definition_Create_DM (continued)	
directory_keyword_parameter=	**Blank-delimited *directory_keyword_operand=directory_keyword_operand_value* pairs, OR *directory_keyword_parameter_value***
INCLUDE=	• *profilename* (**Required**)
IPL=	• VDEV=*vdev* (can also contain *nssname*) (**Required**) • LOADPARM=*loadparm* (Optional) • PARM= *parmstring* (Optional) See Usage Note "1" on page 171 and Usage Note "2" on page 171.
LINK=	• USERID=*userid* or * (**Required**) • VDEV1=*vdev1* (**Required**) • VDEV2=*vdev2* (Optional) • MODE=*mode* | *modesuffix* (Optional, default is R) • PASSWORD=*password* (Optional)
MDISK=	• VDEV=*vdev* • DEVTYPE=*devtype* • DISKTYPE=PERM | DEVNO | V-DISK | TDISK | AUTOG | AUTOR | AUTOV • START=*cyl* | *blk* • COUNT=*cyls* | *blks* • VOLID=*volid* • RDEV=*rdev* • MODE=W | *mode* | *modesuffix* • READPASSWORD=*pr* • WRITEPASSWORD=*pw* • MULTIPASSWORD=*pm* • NAME=*groupname* | *regionname* See Usage Note "9" on page 172 for a table of required, optional, and default *directory_keyword_operand* and *directory_keyword_operand_value* pairs for the MDISK= *directory_keyword_parameter*.
NICDEF=	• VDEV=*vdev* (**Required**) • TYPE=QDIO | HIPERSOCKETS (**Required**) • DEVICES=*devs* (Optional) • LAN=* | SYSTEM | *ownerid* (Optional) • LANNAME=*lanname* (**Required** if LAN=* or LAN=*ownerid*, otherwise ignored) • SWITCHNAME=*switchname* (**Required** if LAN=SYSTEM, otherwise ignored) • CHPID=*chpid* (Optional) • MACID=*macid* (Optional)

Table 2. Input Keywords and Values for Image_Definition_Create_DM (continued)	
directory_keyword_parameter=	**Blank-delimited *directory_keyword_operand=directory_keyword_operand_value* pairs, OR *directory_keyword_parameter_value***

OPTION=	• A string containing all of the option statements to be appended to the user or profile. Note that no uppercasing, defaulting, or abbreviating will be performed on the data. (**Required**)
	For example:
	`OPTION=ACCT APPLMON MAXCONN 70'00'x`
	`('00'x = null terminator)`

PASSWORD=	• *password* (Optional, default is NOLOG.)
PRIVILEGE_CLASSES=	• *classes* (**Required**)
SHARE=	• ABSOLUTE=*y*% (Either ABSOLUTE= or RELATIVE= is **required**) • RELATIVE=*z* (See above) • ABSOLUTE_MAX=*a*% (Optional) • RELATIVE_MAX=*b* (Optional) • LIMIT=NOLIMIT | LIMITSOFT | LIMITHARD (Optional, default is NOLIMIT if ABSOLUTE_MAX or RELATIVE_MAX is *not* specified, otherwise default is LIMITSOFT)
SPOOL=	• VDEV=*vdev* (**Required**) • DEVTYPE=PCH | PUNCH | PRINTER | PRT | RDR | READER | VAFP | 1403 | 2501 | 2540_READER | 2540_PUNCH | 3203 |3211 | 3262 | 3505 | 3525 | 3800 | 3800-1 | 3800-3 | 4245 | 4248 (**Required**) • CLASS=0-9 | A-Z | * (**Required** if WIDTH= and LENGTH= are specified) Additional options for the 3800 printer only: • WIDTH=*hexadecimal_value* (**Required** if any of the following keywords are specified) • LENGTH=*decimal_value_of_half_inches* (**Required** if any of the following keywords are specified) • CHARACTER_GENERATION_MODULES=4 | 2 (Optional, default is 4) • STACKER=CONTINUOUS | BURSTER (Optional, default is CONTINUOUS) • CP_PROCESS_DATA_CHECK=YES | NO (Optional, default is NO)
STORAGE_INITIAL=	• *stor* (storage and unit) (**Required**)
STORAGE_MAXIMUM=	• *mstor* (storage and unit) (**Required**)
SYSTEM_UNIQUE=	• YES | NO (Optional, default is NO.) **Note:** If you specify SYSTEM_UNIQUE=YES, an IDENT entry will be created. If you specify SYSTEM_UNIQUE=NO (or if you specify nothing), a USER entry will be created.

Table 2. Input Keywords and Values for Image_Definition_Create_DM (continued)	
directory_keyword_parameter=	**Blank-delimited *directory_keyword_operand=directory_keyword_operand_value* pairs, OR *directory_keyword_parameter_value***
VMRELOCATE=	• ENABLED=ON | OFF (Optional, default is ON) • DOMAIN=SSI | *domain_name* (Optional, default is SSI) **Note:** 1. This parameter is valid only if SYSTEM_UNIQUE=NO (i.e. for USER entries). It will be ignored if SYSTEM_UNIQUE=YES (i.e. for IDENT entries). 2. If you specify VMRELOCATE= with no values, the two defaults (ENABLED=ON and DOMAIN=SSI) will be assumed.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

If RC=592 and RS=4, the following parameters will be returned:

asynch_length
(int4) Length of *asynch_data*.

asynch_data
(string) "ASYNCH_IDS=", followed by a string enclosed in double quotes. The string contains a blank-delimited series of operation IDs from the directory manager, with the specific command concatenated inside parentheses. The string is followed by a null (ASCIIZ) terminator.

For example:

```
ASYNCH_IDS="1503(AMDISK 201 XXXX AUTOV 10 VOLXYZ)"'00'x
   ('00'x = null terminator)
```

If RC=8 and RS=3002, the following parameters will be returned:

error_length
(int4) Length of *error_data*.

error_data
One of the following:

- (string) "UNKNOWN_DIRECTORY_KEYWORD_PARAMETER=", followed by a blank-delimited series of directory keyword parameters (as specified in the *definition_create_directory_keyword_parameter_list* input parameter) that are not recognized by this API (i.e. are not in Table 2 on page 165), followed by a null (ASCIIZ) terminator.

If RC=8 and RS=3032, the following parameters will be returned:

error_length
(int4) Length of *error_data*.

error_data
One or more of the following:

- (string) "INVALID_DIRECTORY_KEYWORD_OPERAND_VALUE=", followed by a blank-delimited series of directory keyword operands (as specified in the *definition_create_directory_keyword_parameter_list* input parameter) that have invalid values specified, followed by a null (ASCIIZ) terminator.
- (string) "INVALID_DIRECTORY_KEYWORD_PARAMETER_VALUE=", followed by a blank-delimited series of directory keyword parameters (as specified in the *definition_create_directory_keyword_parameter_list* input parameter) that have invalid values specified, followed by a null (ASCIIZ) terminator.
- (string) "UNKNOWN_DIRECTORY_KEYWORD_OPERAND=", followed by a blank-delimited series of directory keyword operands (as specified in the *definition_create_directory_keyword_parameter_list* input parameter) that are not recognized by this API (i.e. are not in Table 2 on page 165), followed by a null (ASCIIZ) terminator.
- (string) "MISSING_DIRECTORY_KEYWORD_PARAMETER=", followed by a blank-delimited series of directory keyword parameters (as specified in the *definition_create_directory_keyword_parameter_list* input parameter) that are not specified, followed by a null (ASCIIZ) terminator.
- (string) "MISSING_DIRECTORY_KEYWORD_OPERAND=", followed by a blank-delimited series of directory keyword operands (as specified in the *definition_create_directory_keyword_parameter_list* input parameter) that are not specified, followed by a null (ASCIIZ) terminator.
- (string) "MISSING_DIRECTORY_KEYWORD_OPERAND_VALUE=", followed by a blank-delimited series of directory keyword operands (as specified in the *definition_create_directory_keyword_parameter_list* input parameter) that do not have values specified, followed by a null (ASCIIZ) terminator.
- (string) "MISSING_DIRECTORY_KEYWORD_PARAMETER_VALUE=", followed by a blank-delimited series of directory keyword parameters (as specified in the *definition_create_directory_keyword_parameter_list* input parameter) that do not have values specified, followed by a null (ASCIIZ) terminator.
- (string) "CONFLICTING_DIRECTORY_KEYWORD_OPERAND=", followed by a blank-delimited series of directory keyword operands (as specified in the *definition_create_directory_keyword_parameter_list* input parameter) that are mutually exclusive, followed by a null (ASCIIZ) terminator.

To make it easier to find the parameter name in error (for example, there may be multiple MDISK statements specified), "(*nnn*)" will be concatenated at the end of each parameter name. This number will correspond to the order of the parameter name as it's found in all of the parameters specified with this API.

For example (in this case a missing SWITCHNAME=*switchname* on the NICDEF= directory keyword parameter):

```
MISSING_DIRECTORY_KEYWORD_OPERAND=NICDEF(1)=SWITCHNAME'00'x
   ('00'x = null terminator)
```

The following illustrates some of the errors that may be returned in the output buffer when RC=8 and RS=3032:

- Input:

```
=XXX=1234(x'00)
STORAGE_INITIAL=64M(x'00)
STORAGE_MAXIMUM=M(x'00)
PRIVILEGE_CLASSES=(x'00)
CONSOLE=XXXX=009 CLASS=T(x'00)
CPU=CPUADDR(x'00)
IPL==XXX(x'00)
INCLUDE==YYY(x'00)
SHARE=ABSOLUTE=10% RELATIVE=10(x'00)
LINK=USERID=MAINT VDE1=XXXX VDEV2=0190 MODE=XX(x'00)
```

- Output:

```
INVALID_DIRECTORY_KEYWORD_OPERAND_VALUE=LINK(10)=MODE=XX(x'00)
INVALID_DIRECTORY_KEYWORD_PARAMETER_VALUE=STORAGE_MAXIMUM(3)=M INCLUDE(8)==YYY(x'00)
UNKNOWN_DIRECTORY_KEYWORD_OPERAND=CONSOLE(5)=XXXX IPL(7)==XXX LINK(10)=VDE1(x'00)
MISSING_DIRECTORY_KEYWORD_PARAMETER=(1)=XXX=1234(x'00)
MISSING_DIRECTORY_KEYWORD_OPERAND=CONSOLE(5)=VDEV CONSOLE(5)=DEVTYPE IPL(7)=VDEV
LINK(10)=VDEV1(x'00)
MISSING_DIRECTORY_KEYWORD_OPERAND_VALUE=CPU(6)=CPUADDR(x'00)
MISSING_DIRECTORY_KEYWORD_PARAMETER_VALUE=PRIVILEGE_CLASSES(4)=(x'00)
CONFLICTING_DIRECTORY_KEYWORD_OPERAND=SHARE(9)=(ABSOLUTE RELATIVE)(x'00)
```

For all other errors, the following parameters will be returned (if available):

error_length
(int4) Length of *error_data*.

error_data
(string) "COMMAND_IN_ERROR=", followed by the specific directory manager command that failed and any accompanying error message text, followed by a null (ASCIIZ) terminator.

Usage Notes

1. If LOADPARM=*loadparm* is specified with IPL=, note that *loadparm* can be a quoted string (as described in *z/VM: CP Planning and Administration*), but in this case, embedded blanks are *not* supported. If you need embedded blanks in *loadparm*, you'll have to update the directory with a GET and REPLACE instead of using this API.
2. If PARM= *parmstring* is specified with IPL=, it must be specified after VDEV=*vdev* and LOADPARM=*loadparm*, so that any characters can be used in *parmstring* (except binary zeroes).
3. A snapshot of the directory will be taken before any of the updates are processed. If a directory manager error occurs, an attempt to restore the original directory will be made. A log record will also be written to the SMAPI LOG with the directory manager command that failed (providing that the log level is set at least to level 3). If you are attempting to do multiple updates (for example, three separate MDISK= specifications) and a failure happens to occur, the reset directory may not correctly reflect the multiple updates. To avoid this potential problem, do only one update per API call, so that the reset directory will be valid.
4. A log record will also be written to the SMAPI LOG with the directory manager command that failed (providing that the log level is set at least to level 3).
5. Syntax errors (RC=24 and RS=*pprr*) are only applicable to the common input parameters. Syntax checking is *not* performed on the additional input parameter for this API. If a valid parameter is specified multiple times, the last value specified for that parameter will be used. For more information, see “Key-value pair (KVP) input parameters” on page 51.
6. If SYSTEM_UNIQUE=YES, most of the virtual machine information will be stored in a generated subconfig name specifically for this system.
7. If you wish to completely delete a user/identity/subconfig, use “Image_Delete_DM” on page 202.

8. Using Image_Definition_Create_DM to create a minidisk **always** results in a minidisk that is **not** formatted. If you want a to create a minidisk that is pre-formatted for use with CMS, use the Image_Disk_Create_DM API.
9. Table 3 on page 172 shows in further detail the required and optional keywords for disk-related directory manager operations associated with the MDISK= parameter name.

Table 3. MDISK= Keywords by Directory Manager Operation

Keyword Parameter	MDISK Operation			
	ADD	**MODEPW**	**REDEFINE**	**REPLACE**
VDEV	**Required**	**Required**	**Required**	**Required**
OPERATION	**Required**	**Required**	**Required**	**Required**
DEVTYPE	**Required**	Ignored	**Required**	**Required**
DISKTYPE	**Required**	Ignored	**Required**	**Required**
If DISKTYPE=PERM:				
COUNT	**Required**	Ignored	**Required**	**Required**
NAME	Ignored	Ignored	Ignored	Ignored
RDEV	Ignored	Ignored	Ignored	Ignored
START	**Required**	Ignored	**Required**	**Required**
VOLID	**Required**	Ignored	**Required**	**Required**
If DISKTYPE=DEVNO:				
COUNT	Ignored	Ignored	Ignored	Ignored
NAME	Ignored	Ignored	Ignored	Ignored
RDEV	**Required**	Ignored	**Required**	**Required**
START	Ignored	Ignored	Ignored	Ignored
VOLID	Ignored	Ignored	Ignored	Ignored
If DISKTYPE=AUTOG | AUTOR:				
COUNT	**Required**	Ignored	**Required**	**Required**
NAME	**Required**	Ignored	**Required**	**Required**
RDEV	Ignored	Ignored	Ignored	Ignored
START	Ignored	Ignored	Ignored	Ignored
VOLID	Ignored	Ignored	Ignored	Ignored
If DISKTYPE=AUTOV:				
COUNT	**Required**	Ignored	**Required**	**Required**
NAME	Ignored	Ignored	Ignored	Ignored
RDEV	Ignored	Ignored	Ignored	Ignored
START	Ignored	Ignored	Ignored	Ignored
VOLID	**Required**	Ignored	**Required**	**Required**
If DISKTYPE=T-DISK | V-DISK:				
COUNT	**Required**	Ignored	**Required**	**Required**

Table 3. MDISK= Keywords by Directory Manager Operation (continued)

Keyword Parameter	MDISK Operation			
	ADD	**MODEPW**	**REDEFINE**	**REPLACE**
NAME	Ignored	Ignored	Ignored	Ignored
RDEV	Ignored	Ignored	Ignored	Ignored
START	Ignored	Ignored	Ignored	Ignored
VOLID	Ignored	Ignored	Ignored	Ignored
Password Options:				
MODE	Optional (default is W)	Optional (default is unchanged)	Ignored	Optional (default is W)
READPASSWORD	Optional	Optional (default is unchanged)	Ignored	Optional
WRITEPASSWORD	Optional (requires read password)	Optional (default is unchanged)	Ignored	Optional (requires read password)
MULTIPASSWORD	Optional (requires write password)	Optional (default is unchanged)	Ignored	Optional (requires write password)

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
8	RC_ERR	3002	RS_INVALID_PARAMETER	Invalid parameter name
		3032	RS_INVALID_INPUT	Invalid input
24	RCERR_SYNTAX	*pprr*	*pprr*	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		12	RS_AUTHERR_DM	Request not authorized by directory manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
396	RCERR_INTERNAL	*nnnn*	*psrc*	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)

RC	RC Name	RS	RS Name	Description
400	RCERR_INTERNAL	8	RS_NAME_EXISTS	Image or profile name already defined
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available
592	RCERR_ASYNC_DM	4	RS_WORK_OUTSTANDING	Asynchronous operation started - product-specific asynchronous operation ID (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
596	RCERR_INTERNAL_DM	*nnnn*	*psrc*	Internal directory manager error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Image_Definition_Delete_DM

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
definition_delete_directory_keyword_parameter_list_length
definition_delete_directory_keyword_parameter_list

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
error_length/*asynch_length* (error only)
error_data/*asynch_data* (error only)

Purpose

Use Image_Definition_Delete_DM to remove a directory statement for a user or profile. The image must be unlocked before issuing this API.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 26.

function_name
(string,26,char43) The API function name – in this case, 'Image_Definition_Delete_DM'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See “Client Authentication” on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The name of the image or profile from which you wish to remove a directory statement.

definition_delete_directory_keyword_parameter_list_length
(int4) Length of *definition_delete_directory_keyword_parameter_list*.

definition_delete_directory_keyword_parameter_list
(string,1-maxlength,charNA) The remaining set of *directory_keyword_parameter*= input parameters.

The format for specifying the following additional input parameters is either *directory_keyword_parameter*= followed by a blank-delimited series of *directory_keyword_operand=directory_keyword_operand_value* pairs, or *directory_keyword_parameter=directory_keyword_parameter_value*, in both cases followed by a null (ASCIIZ) terminator. They may be specified in any order. For more information, see "Key-value pair (KVP) input parameters" on page 51.

For example:

```
CONSOLE='00'x DEDICATE=VDEV=2001'00'x
   ('00'x = null terminator)
```

Table 4 on page 176 shows the keywords and values that are, in some cases, required to determine which specific directory statement to delete. See *z/VM: CP Planning and Administration* for more information on how these directory entries work.

Table 4. Input Keywords and Values for Image_Definition_Delete_DM

directory_keyword_parameter=	**Blank-delimited *directory_keyword_operand=directory_keyword_operand_value* pairs, OR *directory_keyword_parameter_value***
COMMAND_DEFINE_CPU=	• CPUADDR='*cpuaddr*' (can be any CPU address range, as described in *z/VM: CP Planning and Administration*) (**Required**)
COMMAND_SET_CPUAFFINITY=	• USERID=*userid* or * (**Required**)
COMMAND_SET_SHARE=	• USERID=*userid* (**Required**)
COMMAND_SET_VCONFIG=	**Note:** The COMMAND_SET_VCONFIG statement will be removed from the directory.
CONSOLE=	

Table 4. Input Keywords and Values for Image_Definition_Delete_DM (continued)	
directory_keyword_parameter=	**Blank-delimited *directory_keyword_operand=directory_keyword_operand_value* pairs, OR *directory_keyword_parameter_value***
CPU=	• CPUADDR=*cpuaddr* (**Required**)
CPU_MAXIMUM=	**Note:** The MACHINE statement will be removed from the directory.
DEDICATE=	• VDEV=*vdev* (**Required**)
INCLUDE=	
IPL=	
LINK=	• USERID=*userid* or * (**Required**) • VDEV1=*vdev1* (**Required**) • VDEV2=*vdev2* (**Required**)
MDISK=	• VDEV=*vdev* (**Required**)
NICDEF=	• VDEV=*vdev* (**Required**)
OPTION=	• A string containing the blank-delimited option values to be deleted. (**Required**)
PASSWORD=	**Note:** The password will be changed to NOLOG.
PRIVILEGE_CLASSES=	• *classes* (**Required**)
SHARE=	**Note:** The SHARE statement will be removed from the directory.
SPOOL=	• VDEV=*vdev* (**Required**)
STORAGE_INITIAL=	**Note:** The STORAGE statement will be removed from the directory.
STORAGE_MAXIMUM=	**Note:** The MAXSTORAGE statement will be removed from the directory.
VMRELOCATE=	**Note:** The VMRELOCATE statement will be removed from the directory.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

If RC=8 and RS=4, the following parameters will also be returned:

error_length
(int4) Length of *error_data*.

error_data
(string) "COMMAND_IN_ERROR=", followed by the keyword and values specified on the call, followed by a null (ASCIIZ) terminator. For example:

```
COMMAND IN ERROR="COMMAND DEFINE CPU=CPUADDR=3"'00'x
   ('00'x = null terminator)
```

If RC=592 and RS=4, the following parameters will be returned:

asynch_length
(int4) Length of *asynch_data*.

asynch_data
(string) "ASYNCH_IDS=", followed by a string enclosed in double quotes. The string contains a blank-delimited series of operation IDs from the directory manager, with the specific command concatenated inside parentheses. The string is followed by a null (ASCIIZ) terminator.

For example:

```
ASYNCH_IDS="1503(AMDISK 201 XXXX AUTOV 10 VOLXYZ)"'00'x
   ('00'x = null terminator)
```

If RC=8 and RS=3002, the following parameters will be returned:

error_length
(int4) Length of *error_data*.

error_data
One of the following:

- (string) "UNKNOWN_DIRECTORY_KEYWORD_PARAMETER=", followed by a blank-delimited series of directory keyword parameters (as specified in the *definition_delete_directory_keyword_parameter_list* input parameter) that are not recognized by this API (that is are not in Table 4 on page 176), followed by a null (ASCIIZ) terminator.

If RC=8 and RS=3032, the following parameters will be returned:

error_length
(int4) Length of *error_data*.

error_data
One or more of the following:

- (string) "INVALID_DIRECTORY_KEYWORD_OPERAND_VALUE=", followed by a blank-delimited series of directory keyword operands (as specified in the *definition_delete_directory_keyword_parameter_list* input parameter) that have invalid values specified, followed by a null (ASCIIZ) terminator.
- (string) "INVALID_DIRECTORY_KEYWORD_PARAMETER_VALUE=", followed by a blank-delimited series of directory keyword parameters (as specified in the

definition_delete_directory_keyword_parameter_list input parameter) that have invalid values specified, followed by a null (ASCIIZ) terminator.

- (string) "UNKNOWN_DIRECTORY_KEYWORD_OPERAND=", followed by a blank-delimited series of directory keyword operands (as specified in the *definition_delete_directory_keyword_parameter_list* input parameter) that are not recognized by this API (i.e. are not in Table 4 on page 176), followed by a null (ASCIIZ) terminator.
- (string) "MISSING_DIRECTORY_KEYWORD_PARAMETER=", followed by a blank-delimited series of directory keyword parameters (as specified in the *definition_delete_directory_keyword_parameter_list* input parameter) that are not specified, followed by a null (ASCIIZ) terminator.
- (string) "MISSING_DIRECTORY_KEYWORD_OPERAND=", followed by a blank-delimited series of directory keyword operands (as specified in the *definition_delete_directory_keyword_parameter_list* input parameter) that are not specified, followed by a null (ASCIIZ) terminator.
- (string) "MISSING_DIRECTORY_KEYWORD_OPERAND_VALUE=", followed by a blank-delimited series of directory keyword operands (as specified in the *definition_delete_directory_keyword_parameter_list* input parameter) that do not have values specified, followed by a null (ASCIIZ) terminator.
- (string) "MISSING_DIRECTORY_KEYWORD_PARAMETER_VALUE=", followed by a blank-delimited series of directory keyword parameters (as specified in the *definition_delete_directory_keyword_parameter_list* input parameter) that do not have values specified, followed by a null (ASCIIZ) terminator.
- (string) "CONFLICTING_DIRECTORY_KEYWORD_OPERAND=", followed by a blank-delimited series of directory keyword operands (as specified in the *definition_delete_directory_keyword_parameter_list* input parameter) that are mutually exclusive, followed by a null (ASCIIZ) terminator.

To make it easier to find the parameter name in error (for example, there may be multiple MDISK statements specified), "(*nnn*)" will be concatenated at the end of each parameter name. This number will correspond to the order of the parameter name as it's found in all of the parameters specified with this API.

In the following example there is a missing SWITCHNAME=*switchname* on the NICDEF= directory keyword parameterh:

```
MISSING_DIRECTORY_KEYWORD_OPERAND=NICDEF(1)=SWITCHNAME'00'x
   ('00'x = null terminator)
```

The following illustrates some of the errors that may be returned in the output buffer when RC=8 and RS=3032:

Input:

```
=XXX=1234(x'00)
STORAGE_INITIAL=64M(x'00)
MDISK=VDEV=YYY(x'00)
NICDEF=123456=(x'00)
SPOOL=ABCDEFGHI(x'00)
CPU=CPUADDR(x'00)
IPL==XXX(x'00)
COMMAND_DEFINE_CPU=CPUADDR=123H(x'00)
```

Output:

```
INVALID_DIRECTORY_KEYWORD_OPERAND_VALUE=MDISK(3)=VDEV=YYY ...
INVALID_DIRECTORY_KEYWORD_PARAMETER_VALUE=STORAGE_INITIAL(2)=64M ...
UNKNOWN_DIRECTORY_KEYWORD_OPERAND=NICDEF(4)=123456 SPOOL(5)=...
MISSING_DIRECTORY_KEYWORD_PARAMETER=(1)=XXX=1234(x'00)
MISSING_DIRECTORY_KEYWORD_OPERAND=NICDEF(4)=VDEV SPOOL(5)=VDEV(x'00)
MISSING_DIRECTORY_KEYWORD_OPERAND_VALUE=CPU(6)=CPUADDR(x'00)
```

Usage Notes

1. A snapshot of the directory will be taken before any of the updates are processed. If a directory manager error occurs, an attempt to restore the original directory will be made. A log record will also be written to the SMAPI LOG with the directory manager command that failed (providing that the log level is set at least to level 3). If you are attempting to do multiple updates (for example, three separate MDISK= specifications) and a failure happens to occur, the reset directory may not correctly reflect the multiple updates. To avoid this potential problem, do only one update per API call, so that the reset directory will be valid.
2. Syntax errors (RC=24 and RS=*pprr*) are only applicable to the common input parameters. Syntax checking is *not* performed on the additional input parameter for this API. If a valid parameter is specified multiple times, the last value specified for that parameter will be used. For more information, see “Key-value pair (KVP) input parameters” on page 51.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
8	RC_ERR	4	RS_NOT_FOUND	Directory entry to be deleted not found
		3002	RS_INVALID_PARAMETER	Invalid parameter name
		3032	RS_INVALID_INPUT	Invalid input
24	RCERR_SYNTAX	*pprr*	*pprr*	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		12	RS_AUTHERR_DM	Request not authorized by directory manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
396	RCERR_INTERNAL	*nnnn*	*psrc*	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
400	RCERR_IMAGEDEF	4	RS_NOT_FOUND	Image or profile definition not found
		12	RS_LOCKED	Image or profile definition is locked
404	RCERR_IMAGEDEVD	8	RS_NOT_DEFINED	Image device not defined
408	RCERR_IMAGEDISKD	8	RS_NOT_DEFINED	Image disk not defined
		12	RS_LOCKED	Image device is locked
460	RC_IPL_DM	4	RS_IPL_NOT_FOUND	Image does not have an IPL statement

RC	RC Name	RS	RS Name	Description
500	RCERR_DM	4	RS_NO_UPDATES	Directory manager is not accepting updates
		8	RS_NOT_AVAILABLE	Directory manager is not available
520	RCERR_CPU_DM	30	RS_CPU_NOT_FOUND	CPU not found
592	RCERR_ASYNC_DM	4	RS_WORK_OUTSTANDING	Asynchronous operation started - product-specific asynchronous operation ID (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
596	RCERR_INTERNAL_DM	*nnnn*	*psrc*	Internal directory manager error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Image_Definition_Query_DM

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
definition_query_directory_keyword_parameter_list_length
definition_query_directory_keyword_parameter_list

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
directory_information_length (if no error)
directory_information_data (if no error)
error_length (error only)
error_data (error only)

Purpose

Use Image_Definition_Query_DM to extract directory records and parse them into certain keywords.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 25.

function_name
(string,25,char43) The API function name – in this case, 'Image_Definition_Query_DM'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The name of the image or profile from which you wish to extract directory records.

definition_query_directory_keyword_parameter_list_length
(int4) Length of *definition_query_directory_keyword_parameter_list*.

definition_query_directory_keyword_parameter_list
(string,1-maxlength,charNA) A set of blank-delimited keywords from the following list, followed by a null (ASCIIZ) terminator:

- COMMAND_DEFINE_CPU
- COMMAND_SET_CPUAFFINITY
- COMMAND_SET_SHARE
- COMMAND_SET_VCONFIG
- CONSOLE
- CPU
- CPU_MAXIMUM
- DEDICATE
- INCLUDE
- IPL
- LINK
- MDISK
- NICDEF
- OPTION
- PASSWORD
- PRIVILEGE_CLASSES
- SHARE
- SPOOL
- STORAGE_INITIAL
- STORAGE_MAXIMUM
- VMRELOCATE
- * (asterisk, meaning all of the above)

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

If no errors are encountered, the following parameters will be returned:

directory_information_length
(int4) Length of *directory_information_data*.

directory_information_data
(string) As per the list identified in the *definition_query_directory_keyword_parameter_list* input parameter, a series of null-terminated strings, each containing "*directory_keyword_parameter*= " followed by either a series of blank-delimited "*directory_keyword_parameter=directory_keyword_operand_value*" pairs or a *directory_keyword_parameter_value*.

If an appropriate entry is not found in the directory, then "*directory_keyword_parameter*= " (with a blank) will be returned. If the entry was found in the profile, then "*_PROFILE*" will be appended to *directory_keyword_parameter* (for example, "LINK_PROFILE=*value*" if the LINK entry was found in the profile).

For each specific *directory_keyword_parameter*, the output will be returned with the appropriate series of blank-delimited *directory_keyword=directory_keyword_operand_value* pairs or *directory_keyword_parameter_value*, as shown in Table 5 on page 184. Note that any operand containing blanks will be surrounded by single quotes.

Table 5. Output Keywords and Values for Image_Definition_Query_DM

directory_keyword_parameter=	**Blank-delimited *directory_keyword_operand=directory_keyword_operand_value* pairs, OR *directory_keyword_parameter_value***
COMMAND_DEFINE_CPU= COMMAND_DEFINE_CPU _PROFILE=	• CPUADDR='*cpuaddr*' (can be any CPU address range, as described in *z/VM: CP Planning and Administration*) • TYPE=CP | IFL | ZIIP | ICF (if specified in directory)
COMMAND_SET_CPUAFFINITY= COMMAND_SET_CPUAFFINITY _PROFILE=	• CPUAFFINITY=ON | OFF • USERID=*userid* or *

Table 5. Output Keywords and Values for Image_Definition_Query_DM (continued)

directory_keyword_parameter=	**Blank-delimited *directory_keyword_operand=directory_keyword_operand_value* pairs, OR *directory_keyword_parameter_value***
COMMAND_SET_SHARE= COMMAND_SET_SHARE _PROFILE=	• USERID=*userid* • TYPE=ALL | CP | ZIIP | IFL | ICF • OPERAND= INITIAL | ABSOLUTE | RELATIVE | NOLIMIT | LIMITSOFT | LIMITHARD • ABSOLUTE=*y*% • RELATIVE=*z* • ABSOLUTE_MAX=*a*% • RELATIVE_MAX=*b* • LIMIT=NOLIMIT | LIMITSOFT | LIMITHARD
COMMAND_SET_VCONFIG=	• MODE=ESA390 | LINUX | VM
CONSOLE= CONSOLE_PROFILE=	• VDEV=*vdev* • DEVTYPE=*devtype* • CLASS=T | *class* • USERID=*userid* (if specified in directory) • OBSERVER=YES | NO (if specified in directory)
CPU= CPU_PROFILE=	• CPUADDR=*cpuaddr* • BASE=YES | NO (if specified in directory) • CPUID=*cpuid* (if specified in directory) • DEDICATE=YES | NO (will return default NO if not specified in directory) • CRYPTO=YES | NO (if specified in directory) **Note:** If there is no CPU statement in the directory, "CPUADDR=00 BASE=YES" will be returned.
CPU_MAXIMUM= CPU_MAXIMUM_PROFILE=	• COUNT=*mcpu* • TYPE=ESA | XA | XC
DEDICATE= DEDICATE_PROFILE=	• VDEV=*vdev* • RDEV=*rdev* (if specified in directory) • VOLID=*volid* (if specified in directory) • R/O=YES | NO (if specified in directory) • USERACCESSID=*userid* (if specified in directory) • USERTYPE=SINGLEUSER | MULTIUSER • ASSIGN=YES | NO (always returned when USERTYPE=MULTIUSER, only if specified in directory when USERTYPE=SINGLEUSER) • QIOASSIST=YES | NO (always returned when USERTYPE=MULTIUSER, only if specified in directory when USERTYPE=SINGLEUSER)

Table 5. Output Keywords and Values for Image_Definition_Query_DM (continued)

directory_keyword_parameter=	**Blank-delimited *directory_keyword_operand=directory_keyword_operand_value* pairs, OR *directory_keyword_parameter_value***
INCLUDE=	• *profilename*
IPL= IPL_PROFILE=	• VDEV=*vdev* (can also contain *nssname*) • LOADPARM=*loadparm* (if specified in directory) • PARM= *parmstring* (if specified in directory)
LINK= LINK_PROFILE=	• USERID=*userid* or * • VDEV1=*vdev1* • VDEV2=*vdev2* • MODE=*mode* | *modesuffix* (will return default R if not specified in directory)
MDISK=	• VDEV=*vdev* • DEVTYPE=*devtype* • DISKTYPE=PERM | DEVNO | V-DISK | TDISK • START=*cyl* | *blk* (if specified in directory) • COUNT=*cyls* | *blks* (if specified in directory) • VOLID=*volid* (if specified in directory) • RDEV=*rdev* (if DISKTYPE=DEVNO is specified in directory) • MODE=W | *mode* | *modesuffix* • READPASSWORD=*pr* • WRITEPASSWORD=*pw* • MULTIPASSWORD=*pm*
NICDEF= NICDEF_PROFILE=	• VDEV=*vdev* • TYPE=QDIO | HIPERSOCKETS • DEVICES=*devs* (if specified in directory) • LAN=* | SYSTEM | *ownerid* (if specified in directory) • LANNAME=*lanname* (if specified in directory) • SWITCHNAME=*switchname* (if LAN=SYSTEM is specified in directory) • CHPID=*chpid* (if specified in directory) • MACID=*macid* (if specified in directory)
OPTION= OPTION_PROFILE=	• A string containing all of the option statements merged together for both the user and the profile. Note that no uppercasing, defaulting, or abbreviating will be performed on this returned data.
PASSWORD=	• *password* (if ESM is installed, the actual password will not be returned and this value will instead be "XXXXXXXX")

Table 5. Output Keywords and Values for Image_Definition_Query_DM (continued)	
directory_keyword_parameter=	**Blank-delimited *directory_keyword_operand=directory_keyword_operand_value* pairs, OR *directory_keyword_parameter_value***
PRIVILEGE_CLASSES= PRIVILEGE_CLASSES_PROFILE=	• *classes*
SHARE= SHARE_PROFILE=	• ABSOLUTE=*y*% (if specified in directory) • RELATIVE=*z* (if specified in directory) • ABSOLUTE_MAX=*a*% (if specified in directory) • RELATIVE_MAX=*b* (if specified in directory) • LIMIT=NOLIMIT | LIMITSOFT | LIMITHARD (if not specified in directory, and neither ABSOLUTE_MAX nor RELATIVE_MAX is specified, the default NOLIMIT will be returned – if not specified in directory and ABSOLUTE_MAX or RELATIVE_MAX is specified, LIMITSOFT will be returned)
SPOOL= SPOOL_PROFILE=	• VDEV=*vdev* • DEVTYPE=PCH | PUNCH | PRINTER | PRT | RDR | READER | VAFP | 1403 | 2501 | 2540_READER | 2540_PUNCH | 3203 |3211 | 3262 | 3505 | 3525 | 3800 | 3800-1 | 3800-3 | 4245 | 4248 • CLASS=0-9 | A-Z | * (if specified in directory) Additional options for the 3800 printer only: • WIDTH=*hexadecimal_value* • LENGTH=*decimal_value_of_half_inches* • CHARACTER_GENERATION_MODULES=4 | 2 • STACKER=CONTINUOUS | BURSTER • CP_PROCESS_DATA_CHECK=YES | NO
STORAGE_INITIAL= STORAGE_INITIAL_PROFILE=	• *stor* (storage and unit)
STORAGE_MAXIMUM= STORAGE_MAXIMUM_PROFILE=	• *mstor* (storage and unit)
VMRELOCATE=	• ENABLED=ON | OFF
VMRELOCATE_PROFILE=	• DOMAIN= SSI | *domain_name*

If RC=8 and RS=3002, the following parameters will be returned:

error_length
(int4) Length of *error_data*.

error_data
One or more of the following:

- (string) "UNKNOWN_DIRECTORY_KEYWORD_PARAMETER=", followed by a blank-delimited series of directory keyword parameters (as specified in the

definition_update_directory_keyword_parameter_list input parameter) that are not recognized by this API (i.e. are not in Table 5 on page 184), followed by a null (ASCIIZ) terminator.

If RC=596, the following parameters will be returned:

error_length
(int4) Length of *error_data*.

error_data
(string) "COMMAND_IN_ERROR=", followed by the specific directory manager command that failed and any accompanying error message text, followed by a null (ASCIIZ) terminator.

Usage Notes

1. The CPU type will be determined from the MACHINE statement or the GLOBALOPTS MACHINE. If no CPU type can be found, "ESA" will be returned. The CPU maximum count will be determined from the machine statement, or if that is missing, by counting unique CPUs in the user and profile directory. If no CPUs are found, "1" will be returned. If all of the CPUs are found in a profile, the maximum with be returned with "CPU_MAXIMUM_PROFILE=". Otherwise, the total will be returned with "CPU_MAXIMUM=".
2. If an asterisk (*) is specified (meaning that all supported directory entries should be queried), any other parameters are ignored.
3. Syntax errors (RC=24 and RS=*pprr*) are only applicable to the common input parameters. Syntax checking is *not* performed on the additional input parameter for this API. If a valid parameter is specified multiple times, the last value specified for that parameter will be used. For more information, see “Key-value pair (KVP) input parameters” on page 51.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
		8	RS_AUTHERR_ESM	Password request not authorized by external security manager
8	RC_ERR	3002	RS_INVALID_PARAMETER	Invalid parameter name
24	RCERR_SYNTAX	*pprr*	*pprr*	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		12	RS_AUTHERR_DM	Request not authorized by directory manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
396	RCERR_INTERNAL	*nnnn*	*psrc*	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
400	RCERR_IMAGEDEF	4	RS_NOT_FOUND	Image or profile definition not found

RC	RC Name	RS	RS Name	Description
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available
592	RCERR_ASYNC_DM	*nnnn*	*opid*	Asynchronous operation started - product-specific asynchronous operation ID (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
596	RCERR_INTERNAL_DM	*nnnn*	*psrc*	Internal directory manager error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Image_Definition_Update_DM

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
definition_update_directory_keyword_parameter_list_length
definition_update_directory_keyword_parameter_list

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
error_length (error only)
error_data (error only)

Purpose

Use Image_Definition_Update_DM to update (replace) a directory statement for a user or profile – or to create one if not found. Note that the image must be unlocked before issuing this API.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 26.

function_name
(string,26,char43) The API function name – in this case, 'Image_Definition_Update_DM'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See “Client Authentication” on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The name of the image or profile for which you wish to update/create a directory record.

definition_update_directory_keyword_parameter_list_length
(int4) Length of *definition_update_directory_keyword_parameter_list*.

definition_update_directory_keyword_parameter_list
(string,1-maxlength,charNA) The remaining set of *directory_keyword_parameter*= input parameters.

The format for specifying the following additional input parameters is either *directory_keyword_parameter*= followed by a blank-delimited series of *directory_keyword_operand=directory_keyword_operand_value* pairs, or *directory_keyword_parameter=directory_keyword_parameter_value*, in both cases followed by a null (ASCIIZ) terminator. They may be specified in any order. For more information, see "Key-value pair (KVP) input parameters" on page 51.

For example:

```
DEDICATE=VDEV=234 RDEV=234 USERACCESSID=FRED'00'x IPL=VDEV=CMS'00'x
   ('00'x = null terminator)
```

Table 6 on page 191 shows the keywords and values that can be specified for each type of directory entry, and note how keywords can be always optional, always required, or required only if certain other conditions are true. See *z/VM: CP Planning and Administration* for more information on how these directory entries work.

Table 6. Input Keywords and Values for Image_Definition_Update_DM

directory_keyword_parameter=	**Blank-delimited *directory_keyword_operand=directory_keyword_operand_value* pairs, OR *directory_keyword_parameter_value***
COMMAND_DEFINE_CPU=	• CPUADDR='*cpuaddr*' (can be any CPU address range, as described in *z/VM: CP Planning and Administration*) (**Required**) • TYPE=CP | IFL | ZIIP | ICF (Optional – if no type is specified when defining a new CPU, it defaults to the type of the primary virtual CPU.)
COMMAND_SET_CPUAFFINITY=	• CPUAFFINITY=ON | OFF (**Required**) • USERID=*userid* or * (Optional)

Table 6. Input Keywords and Values for Image_Definition_Update_DM (continued)	
directory_keyword_parameter=	**Blank-delimited *directory_keyword_operand=directory_keyword_operand_value* pairs, OR *directory_keyword_parameter_value***
COMMAND_SET_SHARE=	• USERID=*userid* (**Required**) • TYPE=ALL | CP | ZIIP | IFL | ICF (Optional, default is ALL) • OPERAND= INITIAL | ABSOLUTE | RELATIVE | NOLIMIT | LIMITSOFT | LIMITHARD (**Required**) • ABSOLUTE=*y*% (**Required** if OPERAND=ABSOLUTE) • RELATIVE=*z* (**Required** if OPERAND=RELATIVE) • ABSOLUTE_MAX=*a*% (Optional) • RELATIVE_MAX=*b* (Optional) • LIMIT=NOLIMIT | LIMITSOFT | LIMITHARD (Optional, default is NOLIMIT)
COMMAND_SET_VCONFIG=	• MODE=ESA390 | LINUX | VM (**Required**)
CONSOLE=	• VDEV=*vdev* (**Required**) • DEVTYPE=*devtype* (**Required**) • CLASS=T | *class* (Optional, default is T) • USERID=*userid* (**Required** if OBSERVER=YES is also specified, otherwise optional) • OBSERVER=YES | NO (Optional, default is NO)
CPU=	• CPUADDR=*cpuaddr* (**Required**) • BASE=YES | NO (Optional, default is NO) • CPUID=*cpuid* (Optional) • DEDICATE=YES | NO (Optional, default is NO) • CRYPTO=YES | NO (Optional, default is NO)
CPU_MAXIMUM=	• COUNT=*mcpu* (Optional, default is 1) • TYPE=ESA | XA | XC (**Required**) **Note:** A MACHINE statement will be created/updated with the information specified.
DEDICATE=	• VDEV=*vdev* (**Required**) • RDEV=*rdev* (**Required** for tape and "other". DASD must have either RDEV=, VOLID=, or both.) • VOLID=*volid* (**Required** for DASD, if no RDEV= was specified.) • R/O=YES | NO (Optional, default is NO) • USERACCESSID=*userid* (Optional) • USERTYPE=SINGLEUSER | MULTIUSER (Optional, default is SINGLEUSER for tape) • ASSIGN=NO (Optional) • QIOASSIST=NO (Optional)

Table 6. Input Keywords and Values for Image_Definition_Update_DM (continued)

directory_keyword_parameter=	**Blank-delimited *directory_keyword_operand=directory_keyword_operand_value* pairs, OR *directory_keyword_parameter_value***
INCLUDE=	• *profilename* (**Required**)
IPL=	• VDEV=*vdev* (can also contain *nssname*) (**Required**) • LOADPARM=*loadparm* (Optional) • PARM= *parmstring* (Optional) See Usage Note "1" on page 197 and Usage Note "2" on page 197.
LINK=	• USERID=*userid* or * (**Required**) • VDEV1=*vdev1* (**Required**) • VDEV2=*vdev2* (Optional) • MODE=*mode* | *modesuffix* (Optional, default is R) • PASSWORD=*password* (Optional)
MDISK=	• VDEV=*vdev* • OPERATION=REDEFINE | ADD | MODEPW | REPLACE • DEVTYPE=*devtype* (Specify DEVTYPE=xxxx if DISKTYPE=AUTOG, AUTOR, or AUTOV) • DISKTYPE=PERM | DEVNO | V-DISK | TDISK | AUTOG | AUTOR | AUTOV • START=*cyl* | *blk* • COUNT=*cyls* | *blks* • VOLID=*volid* • RDEV=*rdev* • MODE=W | *mode* | *modesuffix* • READPASSWORD=*pr* • WRITEPASSWORD=*pw* • MULTIPASSWORD=*pm* • NAME=*groupname* | *regionname* See Usage Note "6" on page 197 for a table of required, optional, and default *directory_keyword_operand* and *directory_keyword_operand_value* pairs for the MDISK= *directory_keyword_parameter*.
NICDEF=	• VDEV=*vdev* (**Required**) • TYPE=QDIO | HIPERSOCKETS (**Required**) • DEVICES=*devs* (Optional) • LAN=* | SYSTEM | *ownerid* (Optional) • LANNAME=*lanname* (**Required** if LAN=* or LAN=*ownerid*, otherwise ignored) • SWITCHNAME=*switchname* (**Required** if LAN=SYSTEM, otherwise ignored) • CHPID=*chpid* (Optional) • MACID=*macid* (Optional)

Table 6. Input Keywords and Values for Image_Definition_Update_DM (continued)	
directory_keyword_parameter=	**Blank-delimited *directory_keyword_operand=directory_keyword_operand_value* pairs, OR *directory_keyword_parameter_value***

OPTION=	• A string containing all of the option statements to be appended to the user or profile. Note that no uppercasing, defaulting, or abbreviating will be performed on the data. (**Required**)
	For example:
	`OPTION=ACCT APPLMON MAXCONN 70'00'x`
	`('00'x = null terminator)`

PASSWORD=	• *password* (**Required**)
PRIVILEGE_CLASSES=	• *classes* (**Required**)
SHARE=	• ABSOLUTE=*y*% (Either ABSOLUTE= or RELATIVE= is **required**) • RELATIVE=*z* (See above) • ABSOLUTE_MAX=*a*% (Optional) • RELATIVE_MAX=*b* (Optional) • LIMIT=NOLIMIT | LIMITSOFT | LIMITHARD (Optional, default is NOLIMIT if ABSOLUTE_MAX or RELATIVE_MAX is *not* specified, otherwise default is LIMITSOFT)
SPOOL=	• VDEV=*vdev* (**Required**) • DEVTYPE=PCH | PUNCH | PRINTER | PRT | RDR | READER | VAFP | 1403 | 2501 | 2540_READER | 2540_PUNCH | 3203 |3211 | 3262 | 3505 | 3525 | 3800 | 3800-1 | 3800-3 | 4245 | 4248 (**Required**) • CLASS=0-9 | A-Z | * (**Required** if WIDTH= and LENGTH= are specified) Additional options for the 3800 printer only: • WIDTH=*hexadecimal_value* (**Required** if any of the following keywords are specified) • LENGTH=*decimal_value_of_half_inches* (**Required** if any of the following keywords are specified) • CHARACTER_GENERATION_MODULES=4 | 2 (Optional, default is 4) • STACKER=CONTINUOUS | BURSTER (Optional, default is CONTINUOUS) • CP_PROCESS_DATA_CHECK=YES | NO (Optional, default is NO)
STORAGE_INITIAL=	• *stor* (storage and unit) (**Required**)
STORAGE_MAXIMUM=	• *mstor* (storage and unit) (**Required**)

Table 6. Input Keywords and Values for Image_Definition_Update_DM (continued)	
directory_keyword_parameter=	**Blank-delimited *directory_keyword_operand=directory_keyword_operand_value* pairs, OR *directory_keyword_parameter_value***
VMRELOCATE=	• ENABLED=ON | OFF (Optional, default is ON) • DOMAIN=SSI | *domain_name* (Optional, default is SSI) **Note:** 1. This parameter is valid only for USER entries, not IDENT entries. 2. If you specify VMRELOCATE= with no values, the two defaults (ENABLED=ON and DOMAIN=SSI) will be assumed.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

If RC=592 and RS=4, the following parameters will be returned:

asynch_length
(int4) Length of *asynch_data*.

asynch_data
(string) "ASYNCH_IDS=", followed by a string enclosed in double quotes. The string contains a blank-delimited series of operation IDs from the directory manager, with the specific command concatenated inside parentheses. The string is followed by a null (ASCIIZ) terminator.

For example:

```
ASYNCH_IDS="1503(AMDISK 201 XXXX AUTOV 10 VOLXYZ)"'00'x
   ('00'x = null terminator)
```

If RC=8 and RS=3002, the following parameters will be returned:

error_length
(int4) Length of *error_data*.

error_data
One of the following:

- (string) "UNKNOWN_DIRECTORY_KEYWORD_PARAMETER=", followed by a blank-delimited series of directory keyword parameters (as specified in the *definition_update_directory_keyword_parameter_list* input parameter) that are not recognized by this API (i.e. are not in Table 6 on page 191), followed by a null (ASCIIZ) terminator.

If RC=8 and RS=3032, the following parameters will be returned:

error_length
(int4) Length of *error_data*.

error_data
One or more of the following:

- (string) "INVALID_DIRECTORY_KEYWORD_OPERAND_VALUE=", followed by a blank-delimited series of directory keyword operands (as specified in the *definition_update_directory_keyword_parameter_list* input parameter) that have invalid values specified, followed by a null (ASCIIZ) terminator.
- (string) "INVALID_DIRECTORY_KEYWORD_PARAMETER_VALUE=", followed by a blank-delimited series of directory keyword parameters (as specified in the *definition_update_directory_keyword_parameter_list* input parameter) that have invalid values specified, followed by a null (ASCIIZ) terminator.
- (string) "UNKNOWN_DIRECTORY_KEYWORD_OPERAND=", followed by a blank-delimited series of directory keyword operands (as specified in the *definition_update_directory_keyword_parameter_list* input parameter) that are not recognized by this API (i.e. are not in Table 6 on page 191), followed by a null (ASCIIZ) terminator.
- (string) "MISSING_DIRECTORY_KEYWORD_PARAMETER=", followed by a blank-delimited series of directory keyword parameters (as specified in the *definition_update_directory_keyword_parameter_list* input parameter) that are not specified, followed by a null (ASCIIZ) terminator.
- (string) "MISSING_DIRECTORY_KEYWORD_OPERAND=", followed by a blank-delimited series of directory keyword operands (as specified in the *definition_update_directory_keyword_parameter_list* input parameter) that are not specified, followed by a null (ASCIIZ) terminator.
- (string) "MISSING_DIRECTORY_KEYWORD_OPERAND_VALUE=", followed by a blank-delimited series of directory keyword operands (as specified in the *definition_update_directory_keyword_parameter_list* input parameter) that do not have values specified, followed by a null (ASCIIZ) terminator.
- (string) "MISSING_DIRECTORY_KEYWORD_PARAMETER_VALUE=", followed by a blank-delimited series of directory keyword parameters (as specified in the *definition_update_directory_keyword_parameter_list* input parameter) that do not have values specified, followed by a null (ASCIIZ) terminator.
- (string) "CONFLICTING_DIRECTORY_KEYWORD_OPERAND=", followed by a blank-delimited series of directory keyword operands (as specified in the *definition_update_directory_keyword_parameter_list* input parameter) that are mutually exclusive, followed by a null (ASCIIZ) terminator.

To make it easier to find the parameter name in error (for example, there may be multiple MDISK statements specified), "(*nnn*)" will be concatenated at the end of each parameter name. This number will correspond to the order of the parameter name as it's found in all of the parameters specified with this API.

For example (in this case a missing SWITCHNAME=*switchname* on the NICDEF= directory keyword parameter):

```
MISSING_DIRECTORY_KEYWORD_OPERAND=NICDEF(1)=SWITCHNAME'00'x
   ('00'x = null terminator)
```

The following illustrates some of the errors that may be returned in the output buffer when RC=8 and RS=3032:

- Input:

```
=XXX=1234(x'00)
STORAGE_INITIAL=64M(x'00)
STORAGE_MAXIMUM=M(x'00)
PRIVILEGE_CLASSES=(x'00)
CONSOLE=XXXX=009 CLASS=T(x'00)
CPU=CPUADDR(x'00)
IPL==XXX(x'00)
INCLUDE==YYY(x'00)
SHARE=ABSOLUTE=10% RELATIVE=10(x'00)
LINK=USERID=MAINT VDE1=XXXX VDEV2=0190 MODE=XX(x'00)
```

- Output:

```
INVALID_DIRECTORY_KEYWORD_OPERAND_VALUE=LINK(10)=MODE=XX(x'00)
INVALID_DIRECTORY_KEYWORD_PARAMETER_VALUE=STORAGE_MAXIMUM(3)=M INCLUDE(8)==YYY(x'00)
UNKNOWN_DIRECTORY_KEYWORD_OPERAND=CONSOLE(5)=XXXX IPL(7)==XXX LINK(10)=VDE1(x'00)
MISSING_DIRECTORY_KEYWORD_PARAMETER=(1)=XXX=1234(x'00)
MISSING_DIRECTORY_KEYWORD_OPERAND=CONSOLE(5)=VDEV CONSOLE(5)=DEVTYPE IPL(7)=VDEV
LINK(10)=VDEV1(x'00)
MISSING_DIRECTORY_KEYWORD_OPERAND_VALUE=CPU(6)=CPUADDR(x'00)
MISSING_DIRECTORY_KEYWORD_PARAMETER_VALUE=PRIVILEGE_CLASSES(4)=(x'00)
CONFLICTING_DIRECTORY_KEYWORD_OPERAND=SHARE(9)=(ABSOLUTE RELATIVE)(x'00)
```

For all other errors, the following parameters will be returned (if available):

error_length
(int4) Length of *error_data*.

error_data
(string) "COMMAND_IN_ERROR=", followed by the specific directory manager command that failed and any accompanying error message text, followed by a null (ASCIIZ) terminator.

Usage Notes

1. If LOADPARM=*loadparm* is specified with IPL=, note that *loadparm* can be a quoted string (as described in *z/VM: CP Planning and Administration*), but in this case, embedded blanks are *not* supported. If you need embedded blanks in *loadparm*, you'll have to update the directory with a GET and REPLACE instead of using this API.
2. If PARM= *parmstring* is specified with IPL=, it must be specified after VDEV=*vdev* and LOADPARM=*loadparm*, so that any characters can be used in *parmstring* (except binary zeroes).
3. A snapshot of the directory will be taken before any of the updates are processed. If a directory manager error occurs, an attempt to restore the original directory will be made. A log record will also be written to the SMAPI LOG with the directory manager command that failed (providing that the log level is set at least to level 3). If you are attempting to do multiple updates (for example, three separate MDISK= specifications) and a failure happens to occur, the reset directory may not correctly reflect the multiple updates. To avoid this potential problem, do only one update per API call, so that the reset directory will be valid.
4. Syntax errors (RC=24 and RS=*pprr*) are only applicable to the common input parameters. Syntax checking is *not* performed on the additional input parameter for this API. If a valid parameter is specified multiple times, the last value specified for that parameter will be used. For more information, see “Key-value pair (KVP) input parameters” on page 51.
5. Using Image_Definition_Update_DM to add a minidisk to a directory **always** results in a minidisk that is **not** formatted.
6. Table 7 on page 198 shows in further detail the required and optional keywords for disk-related directory manager operations associated with the MDISK= parameter name.

Table 7. MDISK= Keywords by Directory Manager Operation

Keyword Parameter	MDISK Operation			
	ADD	**MODEPW**	**REDEFINE**	**REPLACE**
VDEV	**Required**	**Required**	**Required**	**Required**
OPERATION	**Required**	**Required**	**Required**	**Required**
DEVTYPE	**Required**	Ignored	**Required**	**Required**
DISKTYPE	**Required**	Ignored	**Required**	**Required**
If DISKTYPE=PERM:				
COUNT	**Required**	Ignored	**Required**	**Required**
NAME	Ignored	Ignored	Ignored	Ignored
RDEV	Ignored	Ignored	Ignored	Ignored
START	**Required**	Ignored	**Required**	**Required**
VOLID	**Required**	Ignored	**Required**	**Required**
If DISKTYPE=DEVNO:				
COUNT	Ignored	Ignored	Ignored	Ignored
NAME	Ignored	Ignored	Ignored	Ignored
RDEV	**Required**	Ignored	**Required**	**Required**
START	Ignored	Ignored	Ignored	Ignored
VOLID	Ignored	Ignored	Ignored	Ignored
If DISKTYPE=AUTOG | AUTOR:				
COUNT	**Required**	Ignored	**Required**	**Required**
NAME	**Required**	Ignored	**Required**	**Required**
RDEV	Ignored	Ignored	Ignored	Ignored
START	Ignored	Ignored	Ignored	Ignored
VOLID	Ignored	Ignored	Ignored	Ignored
If DISKTYPE=AUTOV:				
COUNT	**Required**	Ignored	**Required**	**Required**
NAME	Ignored	Ignored	Ignored	Ignored
RDEV	Ignored	Ignored	Ignored	Ignored
START	Ignored	Ignored	Ignored	Ignored
VOLID	**Required**	Ignored	**Required**	**Required**
If DISKTYPE=T-DISK | V-DISK:				
COUNT	**Required**	Ignored	**Required**	**Required**
NAME	Ignored	Ignored	Ignored	Ignored
RDEV	Ignored	Ignored	Ignored	Ignored
START	Ignored	Ignored	Ignored	Ignored

Table 7. MDISK= Keywords by Directory Manager Operation (continued)				
	MDISK Operation			
Keyword Parameter	**ADD**	**MODEPW**	**REDEFINE**	**REPLACE**
VOLID	Ignored	Ignored	Ignored	Ignored
Password Options:				
MODE	Optional (default is W)	Optional (default is unchanged)	Ignored	Optional (default is W)
READPASSWORD	Optional	Optional (default is unchanged)	Ignored	Optional
WRITEPASSWORD	Optional (requires read password)	Optional (default is unchanged)	Ignored	Optional (requires read password)
MULTIPASSWORD	Optional (requires write password)	Optional (default is unchanged)	Ignored	Optional (requires write password)

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
8	RC_ERR	3002	RS_INVALID_PARAMETER	Invalid parameter name
		3032	RS_INVALID_INPUT	Invalid input
24	RCERR_SYNTAX	*pprr*	*pprr*	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		12	RS_AUTHERR_DM	Request not authorized by directory manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
396	RCERR_INTERNAL	*nnnn*	*psrc*	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
400	RCERR_IMAGEDEF	4	RS_NOT_FOUND	Image or profile definition not found

RC	RC Name	RS	RS Name	Description
		12	RS_LOCKED	Image or profile definition is locked
404	RCERR_IMAGEDEVD	4	RS_EXISTS	Image device already defined
		8	RS_NOT_DEFINED	Image device not defined
		24	RS_TYPE_NOT_SAME	Image device type not same as source
408	RCERR_IMAGEDISKD	4	RS_EXISTS	Image disk already defined
		8	RS_NOT_DEFINED	Image disk not defined
		12	RS_LOCKED	Image device is locked
		28	RS_PW_NEEDED	Image disk does not have required password
		32	RS_BAD_PW	Incorrect password specified for image disk
444	RCERR_POLICY_PW	0	RS_NONE	Password policy error
500	RCERR_DM	4	RS_NO_UPDATES	Directory manager is not accepting updates
		8	RS_NOT_AVAILABLE	Directory manager is not available
520	RCERR_CPU_DM	28	RS_CPU_OUT_OF_RANGE	Input virtual CPU value out of range
592	RCERR_ASYNC_DM	4	RS_WORK_OUTSTANDING	Asynchronous operation started - product-specific asynchronous operation ID (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
596	RCERR_INTERNAL_DM	*nnnn*	*psrc*	Internal directory manager error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data

RC	RC Name	RS	RS Name	Description
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Image_Delete_DM

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
data_security_erase

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
operation_id

Purpose

Use Image_Delete_DM to delete a virtual image's definition from the directory.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 15.

function_name
(string,15,char43) The API function name – in this case, 'Image_Delete_DM'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See “Client Authentication” on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The name of the image to be deleted.

data_security_erase
(int1) Indicates whether to erase data from the disk(s) being released, as follows:

0
Unspecified (use installation default)

1
Do not erase (override installation default)

2
Erase (override installation default)

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

operation_id
(int4; range -1-2147483647) The identifier of the task. If the operation is asynchronous and has not completed, *return_code* will be 592, *reason_code* will be 0, and *operation_id* will be in the range 0-2147483647. If the operation is complete, *operation_id* will be -1.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
		8	RS_OFFLINE	Request successful; object directory offline
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter

RC	RC Name	RS	RS Name	Description
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		12	RS_AUTHERR_DM	Request not authorized by directory manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
400	RCERR_IMAGEDEF	4	RS_NOT_FOUND	Image definition not defined
		12	RS_LOCKED	Image definition is locked
		16	RS_CANNOT_DELETE	Image definition cannot be deleted
408	RCERR_IMAGEDISKD	12	RS_LOCKED	Image disk is locked
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available
592	RCERR_ASYNC_DM	0	RS_NONE	Asynchronous operation started
596	RCERR_INTERNAL_DM	nnnn	psrc	Internal directory manager error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Image_Device_Dedicate

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
image_device_number_length
image_device_number
real_device_number_length
real_device_number
readonly

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Image_Device_Dedicate to add a dedicated device to an active virtual image's configuration.

See "Image_Device_Dedicate_DM" on page 208 to add a dedicated device to a virtual image's directory entry.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 21.

function_name
(string,21,char43) The API function name – in this case, 'Image_Device_Dedicate'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).

- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The name of the image obtaining a dedicated device.

image_device_number_length
(int4) Length of *image_device_number*.

image_device_number
(string,1-4,char16) The virtual device number of the device.

real_device_number_length
(int4) Length of *real_device_number*.

real_device_number
(string,1-4,char16) A real device number to be dedicated or attached to the specified virtual image.

readonly
(int1) Specify a 1 if the virtual device is to be in read-only mode. Otherwise, specify a 0.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Usage Notes

1. *z/VM: CP Planning and Administration* and *z/VM: CP Commands and Utilities Reference* contain additional information on the parameters used by this function.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
200	RCERR_IMAGEOP	4	RS_NOT_FOUND	Image not found
		12	RS_NOT_ACTIVE	Image not active
204	RCERR_IMAGEDEVU	4	RS_EXISTS	Image device already exists
		8	RS_NOT_EXIST	Image device does not exist
		16	RS_NOT_AVAILABLE	Image device is not available
396	RCERR_INTERNAL	nnnn	psrc	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Image_Device_Dedicate_DM

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
image_device_number_length
image_device_number
real_device_number_length
real_device_number
readonly

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Image_Device_Dedicate_DM to add a dedicated device to a virtual image's directory entry.

See "Image_Device_Dedicate" on page 205 to add a dedicated device to an active virtual image's configuration.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 24.

function_name
(string,24,char43) The API function name – in this case, 'Image_Device_Dedicate_DM'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).

- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The name of the image obtaining a dedicated device.

image_device_number_length
(int4) Length of *image_device_number*.

image_device_number
(string,1-4,char16) The virtual device number of the device.

real_device_number_length
(int4) Length of *real_device_number*.

real_device_number
(string,1-4,char16) A real device number to be dedicated or attached to the specified virtual image.

readonly
(int1) Specify a 1 if the virtual device is to be in read-only mode. Otherwise, specify a 0.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Usage Notes

1. *z/VM: CP Planning and Administration* and *z/VM: CP Commands and Utilities Reference* contain additional information on the parameters used by this function.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
		8	RS_OFFLINE	Request successful; object directory offline
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		12	RS_AUTHERR_DM	Request not authorized by directory manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
400	RCERR_IMAGEDEF	4	RS_NOT_FOUND	Image definition not found
		12	RS_LOCKED	Image definition is locked
404	RCERR_IMAGEDEVD	4	RS_EXISTS	Image device already defined
		12	RS_LOCKED	Image Device Is Locked
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available
592	RCERR_ASYNC_DM	nnnn	opid	Asynchronous operation started - product-specific asynchronous operation ID (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
596	RCERR_INTERNAL_DM	nnnn	psrc	Internal directory manager error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Image_Device_Reset

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
image_device_number_length
image_device_number

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Image_Device_Reset to clear all pending interrupts from the specified virtual device.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 18.

function_name
(string,18,char43) The API function name – in this case, 'Image_Device_Reset'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See “Client Authentication” on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The userid or image name for which the device is being reset.

image_device_number_length
(int4) Length of *image_device_number*.

image_device_number
(string,1-4,char16) The virtual device number of the device to reset.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
200	RCERR_IMAGEOP	4	RS_NOT_FOUND	Image not found
		12	RS_NOT_ACTIVE	Image not active
		16	RS_BEING_DEACT	Image being deactivated

RC	RC Name	RS	RS Name	Description
204	RCERR_IMAGEDEVU	8	RS_NOT_EXIST	Image device does not exist
		24	RS_TAPE_NOT_ASSIGNED	Image device is not a tape drive, or cannot be assigned/ reset
		28	RS_DEV_NOT_SHARED	Image device is not a shared DASD
		32	RS_DEV_NOT_RESERVED	Image device is not a reserved DASD
		36	RS_DEV_IO_ERROR	I/O error on image device
396	RCERR_INTERNAL	nnnn	psrc	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Image_Device_Undedicate

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
image_device_number_length
image_device_number

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Image_Device_Undedicate to delete a dedicated device from an active virtual image's configuration.

See "Image_Device_Undedicate_DM" on page 217 to delete a dedicated device from a virtual image's directory entry.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 23.

function_name
(string,23,char43) The API function name – in this case, 'Image_Device_Undedicate'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The name of the image from which a dedicated device is being removed.

image_device_number_length
(int4) Length of *image_device_number*.

image_device_number
(string,1-4,char16) The virtual device number of the device to be deleted.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Usage Notes

1. *z/VM: CP Planning and Administration* and *z/VM: CP Commands and Utilities Reference* contain additional information on the parameters used by this function.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server

RC	RC Name	RS	RS Name	Description
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
200	RCERR_IMAGEOP	4	RS_NOT_FOUND	Image not found
		12	RS_NOT_ACTIVE	Image not active
204	RCERR_IMAGEDEVU	8	RS_NOT_EXIST	Image device does not exist
		40	RS_NWDEV_NOT_DETACHED	Virtual Network Adapter not deleted
		44	RS_DASD_IN_USE	DASD volume cannot be deleted
396	RCERR_INTERNAL	nnnn	psrc	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Image_Device_Undedicate_DM

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
image_device_number_length
image_device_number

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Image_Device_Undedicate_DM to delete a dedicated device from a virtual image's directory entry.

See "Image_Device_Undedicate" on page 214 to delete a dedicated device from an active virtual image's configuration.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 26.

function_name
(string,26,char43) The API function name – in this case, 'Image_Device_Undedicate_DM'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The name of the image from which a dedicated device is being removed.

image_device_number_length
(int4) Length of *image_device_number*.

image_device_number
(string,1-4,char16) The virtual device number of the device to be deleted.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Usage Notes

1. *z/VM: CP Planning and Administration* and *z/VM: CP Commands and Utilities Reference* contain additional information on the parameters used by this function.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
		8	RS_OFFLINE	Request successful; object directory offline
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager

RC	RC Name	RS	RS Name	Description
		12	RS_AUTHERR_DM	Request not authorized by directory manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
400	RCERR_IMAGEDEF	4	RS_NOT_FOUND	Image definition not found
		12	RS_LOCKED	Image definition is locked
404	RCERR_IMAGEDEVD	8	RS_NOT_DEFINED	Image device not defined
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available
592	RCERR_ASYNC_DM	nnnn	opid	Asynchronous operation started - product-specific asynchronous operation ID (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
596	RCERR_INTERNAL_DM	nnnn	psrc	Internal directory manager error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Image_Disk_Copy

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
image_disk_number_length
image_disk_number

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Image_Disk_Copy to clone a disk in an active virtual image's configuration.

See "Image_Disk_Copy_DM" on page 223 to clone a disk in a virtual image's directory entry.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 15.

function_name
(string,15,char43) The API function name – in this case, 'Image_Disk_Copy'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The userid or image name of the single image for which the disk is being copied.

image_disk_number_length
(int4) Length of *image_disk_number*.

image_disk_number
(string,1-4,char16) The virtual device address of the target disk for the copy.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
200	RCERR_IMAGEOP	4	RS_NOT_FOUND	Image not found
		12	RS_NOT_ACTIVE	Image not active

RC	RC Name	RS	RS Name	Description
204	RCERR_IMAGEDEVU	8	RS_NOT_EXIST	Image device does not exist
208	RCERR_IMAGEDISKU	4	RS_IN_USE	Image disk already in use
		8	RS_NOT_IN_USE	Image disk not in use
		36	RS_NOT_EXIST	Image disk does not exist
396	RCERR_INTERNAL	nnnn	psrc	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Image_Disk_Copy_DM

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
image_disk_number_length
image_disk_number
source_image_name_length
source_image_name
source_image_disk_number_length
source_image_disk_number
image_disk_allocation_type_length
image_disk_allocation_type
allocation_area_name_or_volser_length
allocation_area_name_or_volser
image_disk_mode_length
image_disk_mode
read_password_length
read_password
write_password_length
write_password
multi_password_length
multi_password

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
operation_id

Purpose

Use Image_Disk_Copy_DM to clone a disk in a virtual image's directory entry.

See “Image_Disk_Copy” on page 220 to clone a disk in an active virtual image’s configuration.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 18.

function_name
(string,18,char43) The API function name – in this case, 'Image_Disk_Copy_DM'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See “Client Authentication” on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See “Client Authentication” on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The userid or image name of the single image for which the disk is being copied.

image_disk_number_length
(int4) Length of *image_disk_number*.

image_disk_number
(string,1-4,char16) The virtual device address of the target disk for the copy.

source_image_name_length
(int4) Length of *source_image_name*.

source_image_name
(string,1-8,char42) The name of the virtual image that owns the image disk being copied.

source_image_disk_number_length
(int4) Length of *source_image_disk_number*.

source_image_disk_number
(string,1-4,char16) The image disk number of the virtual image that owns the disk being copied.

image_disk_allocation_type_length
(int4) Length of *image_disk_allocation_type*.

image_disk_allocation_type
One of the following:

- (string,0-10,char10) The starting location.

Note: The maximum value for the starting location depends on the allocation units defined in the *allocation_unit_size* parameter. See Usage Note "5" on page 227 for the maximum values allowed for each allocation unit definition.

- (string,5,AUTOG) Automatic_Group_Allocation
- (string,5,AUTOR) Automatic_Region_Allocation
- (string,5,AUTOV) Automatic_Volume_Allocation
- (string,5,DEVNO) Full Volume Minidisk

allocation_area_name_or_volser_length
(int4) Length of *allocation_area_name_or_volser*.

allocation_area_name_or_volser
One of the following:

- (string,0-8,char42) The group or region where the new image disk is to be created. This is specified when *image_disk_allocation_type* is AUTOG or AUTOR.
- (string,0-6,char42) The label of the DASD volume where the new image disk is to be created. This is specified when *image_disk_allocation_type* is the starting location or AUTOV.
- (string,0-4,char42) The device address of the full volume minidisk where the new image disk is to be created. This is specified when *image_disk_allocation_type* is DEVNO.

image_disk_mode_length
(int4) Length of *image_disk_mode*.

image_disk_mode
(string,0-5,char26) The access mode requested for the disk, as seen by the owner when the virtual image is logged on. Valid modes are:

R
Read-only (R/O) access is desired. Access is not allowed if the owner or any other user has a link to the minidisk in write or any exclusive status.

RR
Read-only (R/O) access is desired, even if the owner or another user has a link to the minidisk in write status. Access is denied if any user has the minidisk linked in exclusive status.

W
Write access is desired. The minidisk is not accessible if the owner or any other user has a link to the minidisk in read or write status.

WR
Write access is desired. Only R/O access is allowed if the owner or any other user has a link to the minidisk in read or write status. Access is denied if any exclusive links exist.

M
Multiple access is desired. A write link is allowed to the minidisk unless another user already has write, stable or exclusive access to it, in which case, the minidisk is not accessible to you.

MR
Write or any exclusive access is allowed to the minidisk unless another user already has write access to it, in which case R/O access is allowed to the minidisk. Access is also denied if any exclusive links exist.

MW
Write access is allowed to the disk unconditionally, except for existing stable or exclusive links. Access is denied if any stable or exclusive links exist.

The following is a complete list of valid inputs for this parameter:

R	RR	W	WR	M	MR	MW
RE	RRE	WE	WRE	ME	MRE	MWE
RS	RRS	WS	WRS	MS	MRS	MWS

RD	RRD	WD	WRD	MD	MRD	MWD
RED	RRED	WED	WRED	MED	MRED	MWED
RSD	RRSD	WSD	WRSD	MSD	MRSD	MWSD
RV	RRV	WV	WRV	MV	MRV	MWV
RVE	RRVE	WVE	WRVE	MVE	MRVE	MWVE
RVS	RRVS	WVS	WRVS	MVS	MRVS	MWVS
RVD	RRVD	WVD	WRVD	MVD	MRVD	MWVD
RVED	RRVED	WVED	WRVED	MVED	MRVED	MWVED
RVSD	RRVSD	WVSD	WRVSD	MVSD	MRVSD	MWVSD

See *z/VM: CP Planning and Administration* and the *z/VM: CP Commands and Utilities Reference* for information on link mode definitions, prefixes, and suffixes.

read_password_length
(int4) Length of *read_password*.

read_password
(string,0-8,charNB) Defines the read password that will be used for accessing the disk.

write_password_length
(int4) Length of *write_password*.

write_password
(string,0-8,charNB) Defines the write password that will be used for accessing the disk. Requires a read password.

multi_password_length
(int4) Length of *multi_password*.

multi_password
(string,0-8,charNB) Defines the multi password that will be used for accessing the disk. Requires a write password.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

operation_id
(int4; range -1-2147483647) The identifier of the task. If the operation is asynchronous and has not completed, *return_code* will be 592, *reason_code* will be 0, and *operation_id* will be in the range 0-2147483647. If the operation is complete, *operation_id* will be -1.

Usage Notes

1. The use of some optional parameters requires that other optional parameters be specified as well. If you are uncertain of these interdependencies, see *z/VM: CP Commands and Utilities Reference* for more information on the parameters used by this function.
2. If the *image_disk_number* already exists for the virtual image specified in *target_identifier*, then the following parameters may not be specified:
   - *image_disk_allocation_type*
   - *allocation_area_name_or_volser*
   - *image_disk_mode*
   - *read_password*
   - *write_password*
   - *multi_password*
3. If the *image_disk_number* does *not* exist for the virtual image specified in *target_identifier*, then the following parameters *must* be specified:
   - *image_disk_allocation_type*
   - *allocation_area_name_or_volser*
   - *image_disk_mode*
4. If *read_password*, *write_password*, and *multi_password* are all not specified, no access passwords will be defined for the disk created by the copy.
5. The following table shows the maximum starting locations allowed for each of the allocation units:

Table 8. Maximum Starting Location Allowed, by Allocation Unit

Allocation Unit	Maximum Starting Location
BLK0512	2147383640
BLK1024	1073741820
BLK2048	536870910
BLK4096	268435455
CYLINDERS	2147483640

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
		8	RS_OFFLINE	Request successful; object directory offline
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		12	RS_AUTHERR_DM	Request not authorized by directory manager
		16	RS_AUTHERR_SERVER	Request not authorized by server

RC	RC Name	RS	RS Name	Description
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
400	RCERR_IMAGEDEF	4	RS_NOT_FOUND	Image definition not found
		12	RS_LOCKED	Image definition is locked
404	RCERR_IMAGEDEVD	24	RS_TYPE_NOT_SAME	Image device type not same as source
		28	RS_SIZE_NOT_SAME	Image device size not same as source
408	RCERR_IMAGEDISKD	4	RS_EXISTS	Image disk already defined
		12	RS_LOCKED	Image device is locked
		24	RS_NO_SPACE	Requested image disk space not available
420	RC_DASD_DM	8	RS_IVS_NAME_NOT_USED	Group, region, or volume name is not defined
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available
592	RCERR_ASYNC_DM	0	RS_NONE	Asynchronous operation started
596	RCERR_INTERNAL_DM	nnnn	psrc	Internal directory manager error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Image_Disk_Create

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
image_disk_number_length
image_disk_number
image_disk_mode_length
image_disk_mode

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Image_Disk_Create to add a disk that is defined in a virtual image's directory entry to that virtual image's active configuration.

See "Image_Disk_Create_DM" on page 233 to add a disk to a virtual image's directory entry.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 17.

function_name
(string,17,char43) The API function name – in this case, 'Image_Disk_Create'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The userid or image name of the single image for which the disk is being created.

image_disk_number_length
(int4) Length of *image_disk_number*.

image_disk_number
(string,1-4,char16) The virtual device address of the disk to be added.

Note that the virtual device address must be defined in the virtual image's static configuration. Virtual device addresses that are defined with type "T-DISK" in the static configuration may not be specified.

image_disk_mode_length
(int4) Length of *image_disk_mode*.

image_disk_mode
(string,0-5,char26) The access mode requested for the disk, as seen by the owner when the virtual image is logged on. Valid modes are:

R
Read-only (R/O) access is desired. Access is not allowed if the owner or any other user has a link to the minidisk in write or any exclusive status.

RR
Read-only (R/O) access is desired, even if the owner or another user has a link to the minidisk in write status. Access is denied if any user has the minidisk linked in exclusive status. (This the default if unspecified.)

W
Write access is desired. The minidisk is not accessible if the owner or any other user has a link to the minidisk in read or write status.

WR
Write access is desired. Only R/O access is allowed if the owner or any other user has a link to the minidisk in read or write status. Access is denied if any exclusive links exist.

M
Multiple access is desired. A write link is allowed to the minidisk unless another user already has write, stable or exclusive access to it, in which case, the minidisk is not accessible to you.

MR
Write or any exclusive access is allowed to the minidisk unless another user already has write access to it, in which case R/O access is allowed to the minidisk. Access is also denied if any exclusive links exist.

MW
Write access is allowed to the disk unconditionally, except for existing stable or exclusive links. Access is denied if any stable or exclusive links exist.

If unspecified, the default mode is RR.

The following is a complete list of valid inputs for this parameter:

R	RR	W	WR	M	MR	MW
RE	RRE	WE	WRE	ME	MRE	MWE
RS	RRS	WS	WRS	MS	MRS	MWS
RD	RRD	WD	WRD	MD	MRD	MWD
RED	RRED	WED	WRED	MED	MRED	MWED
RSD	RRSD	WSD	WRSD	MSD	MRSD	MWSD
RV	RRV	WV	WRV	MV	MRV	MWV
RVE	RRVE	WVE	WRVE	MVE	MRVE	MWVE
RVS	RRVS	WVS	WRVS	MVS	MRVS	MWVS
RVD	RRVD	WVD	WRVD	MVD	MRVD	MWVD
RVED	RRVED	WVED	WRVED	MVED	MRVED	MWVED
RVSD	RRVSD	WVSD	WRVSD	MVSD	MRVSD	MWVSD

See *z/VM: CP Planning and Administration* and the *z/VM: CP Commands and Utilities Reference* for information on link mode definitions, prefixes, and suffixes.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Usage Notes

1. The use of some optional parameters requires that other optional parameters be specified as well. If you are uncertain of these interdependencies, see *z/VM: CP Commands and Utilities Reference* for more information on the parameters used by this function.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager

RC	RC Name	RS	RS Name	Description
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
200	RCERR_IMAGEOP	4	RS_NOT_FOUND	Image not found
		12	RS_NOT_ACTIVE	Image not active
208	RCERR_IMAGEDISKU	4	RS_IN_USE	Image disk already in use
		8	RS_NOT_IN_USE	Image disk not in use
		1157	RS_DEVNO_REQUIRES_FREE_DISK	MDISK DEVNO parameter requires the device to be a free volume
396	RCERR_INTERNAL	nnnn	psrc	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Image_Disk_Create_DM

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
image_disk_number_length
image_disk_number
image_disk_device_type_length
image_disk_device_type
image_disk_allocation_type_length
image_disk_allocation_type
allocation_area_name_or_volser_length
allocation_area_name_or_volser
allocation_unit_size
image_disk_size
image_disk_mode_length
image_disk_mode
image_disk_formatting
image_disk_label_length
image_disk_label
read_password_length
read_password
write_password_length
write_password
multi_password_length
multi_password

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
operation_id

Purpose

Use Image_Disk_Create_DM to add a disk to a virtual image's directory entry.

See “Image_Disk_Create” on page 229 to add a disk that is defined in a virtual image’s directory entry to that virtual image’s active configuration.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 20.

function_name
(string,20,char43) The API function name – in this case, 'Image_Disk_Create_DM'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See “Client Authentication” on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See “Client Authentication” on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The userid or image name of the single image for which the disk is being created.

image_disk_number_length
(int4) Length of *image_disk_number*.

image_disk_number
(string,1-4,char16) The virtual device address of the disk to be added.

image_disk_device_type_length
(int4) Length of *image_disk_device_type*.

image_disk_device_type
(string,1-8,char37) The device type of the volume to which the disk is assigned.

image_disk_allocation_type_length
(int4) Length of *image_disk_allocation_type*.

image_disk_allocation_type
One of the following:

- (string,1-10,char10) The starting location.

  Note: The maximum value for the starting location depends on the allocation units defined in the *allocation_unit_size* parameter. See Usage Note “3” on page 238 for the maximum values allowed for each allocation unit definition.

- (string,5,AUTOG) Automatic_Group_Allocation
- (string,5,AUTOR) Automatic_Region_Allocation
- (string,5,AUTOV) Automatic_Volume_Allocation
- (string,5,DEVNO) Full Volume Minidisk
- (string,6,T-DISK) Automatic Temporary Disk
- (string,6,V-DISK) Automatic Virtual Disk – in this case, *image_disk_device_type* must have value = FB-512.

allocation_area_name_or_volser_length
(int4) Length of *allocation_area_name_or_volser*.

allocation_area_name_or_volser
One of the following:

- (string,1-8,char42) The group or region where the new image disk is to be created. This is specified when *image_disk_allocation_type* is AUTOG or AUTOR.
- (string,1-6,char42) The label of the DASD volume where the new image disk is to be created. This is specified when *image_disk_allocation_type* is the starting location or AUTOV.
- (string,1-4,char42) The device address of the full volume minidisk where the new image disk is to be created. This is specified when *image_disk_allocation_type* is DEVNO.

This parameter is ignored when *image_disk_allocation_type* is T-DISK or V-DISK.

allocation_unit_size
(int1) Supported unit sizes are:

1
CYLINDERS

2
BLK0512

3
BLK1024

4
BLK2048

5
BLK4096

image_disk_size
(int4; range 0-2147483640) The size of the disk to be created. The size value is one of the following:

- Cylinders, if the *allocation_unit_size* is "CYLINDERS"
- Logical disk blocks of size *nnnn* if *allocation_unit_size* is BLK*nnnn*. *nnnn* is either 512 (or 0512), 1024, 2048, or 4096".

This parameter should not be specified when *image_disk_allocation_type* is DEVNO. It will be accepted but ignored in this case. (It is required for all other *image_disk_allocation_type* values.

Note: The maximum value for *image_disk_size* depends on the allocation units defined in the *allocation_unit_size* parameter. See Usage Note “3” on page 238 for the maximum values allowed for each allocation unit definition.

image_disk_mode_length
(int4) Length of *image_disk_mode*.

image_disk_mode
(string,1-5,char26) The access mode requested for the disk, as seen by the owner when the virtual image is logged on. Valid modes are:

R
Read-only (R/O) access is desired. Access is not allowed if the owner or any other user has a link to the minidisk in write or any exclusive status.

RR
Read-only (R/O) access is desired, even if the owner or another user has a link to the minidisk in write status. Access is denied if any user has the minidisk linked in exclusive status.

W
Write access is desired. The minidisk is not accessible if the owner or any other user has a link to the minidisk in read or write status.

WR
Write access is desired. Only R/O access is allowed if the owner or any other user has a link to the minidisk in read or write status. Access is denied if any exclusive links exist.

M
Multiple access is desired. A write link is allowed to the minidisk unless another user already has write, stable or exclusive access to it, in which case, the minidisk is not accessible to you.

MR
Write or any exclusive access is allowed to the minidisk unless another user already has write access to it, in which case R/O access is allowed to the minidisk. Access is also denied if any exclusive links exist.

MW
Write access is allowed to the disk unconditionally, except for existing stable or exclusive links. Access is denied if any stable or exclusive links exist.

The following is a complete list of valid inputs for this parameter:

R	RR	W	WR	M	MR	MW
RE	RRE	WE	WRE	ME	MRE	MWE
RS	RRS	WS	WRS	MS	MRS	MWS
RD	RRD	WD	WRD	MD	MRD	MWD
RED	RRED	WED	WRED	MED	MRED	MWED
RSD	RRSD	WSD	WRSD	MSD	MRSD	MWSD
RV	RRV	WV	WRV	MV	MRV	MWV
RVE	RRVE	WVE	WRVE	MVE	MRVE	MWVE
RVS	RRVS	WVS	WRVS	MVS	MRVS	MWVS
RVD	RRVD	WVD	WRVD	MVD	MRVD	MWVD
RVED	RRVED	WVED	WRVED	MVED	MRVED	MWVED
RVSD	RRVSD	WVSD	WRVSD	MVSD	MRVSD	MWVSD

See *z/VM: CP Planning and Administration* and the *z/VM: CP Commands and Utilities Reference* for information on link mode definitions, prefixes, and suffixes.

image_disk_formatting
(int1) Supported formatting options are:

0
Unspecified

1
NONE – Unformatted

2
CMS0512 – CMS formatted with 512 bytes per block

3
CMS1024 – CMS formatted with 1024 bytes per block

4
CMS2048 – CMS formatted with 2048 bytes per block

5
CMS4096 – CMS formatted with 4096 bytes per block

6
CMS – CMS formatted with the default block size for the allocated device type

If unspecified, "NONE" is presumed if no label is specified, "CMS" is presumed if a label is specified. This parameter is ignored when *image_disk_allocation_type* is T-DISK or V-DISK.

image_disk_label_length
(int4) Length of *image_disk_label*.

image_disk_label
(string,0-6,charNB) The disk label to use when formatting the new extent. The labels are 1 to 6 non-blank characters. This parameter is ignored when *image_disk_allocation_type* is T-DISK or V-DISK.

read_password_length
(int4) Length of *read_password*.

read_password
(string,0-8,charNB) Defines the read password that will be used for accessing the disk. This parameter is ignored when *image_disk_allocation_type* is T-DISK.

write_password_length
(int4) Length of *write_password*.

write_password
(string,0-8,charNB) Defines the write password that will be used for accessing the disk. This parameter is ignored when *image_disk_allocation_type* is T-DISK.

multi_password_length
(int4) Length of *multi_password*.

multi_password
(string,0-8,charNB) Defines the multi password that will be used for accessing the disk. This parameter is ignored when *image_disk_allocation_type* is T-DISK.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

operation_id
(int4; range -1-2147483647) The identifier of the task. If the operation is asynchronous and has not completed, *return_code* will be 592, *reason_code* will be 0, and *operation_id* will be in the range 0-2147483647. If the operation is complete, *operation_id* will be -1.

Usage Notes

1. The use of some optional parameters requires that other optional parameters be specified as well. If you are uncertain of these interdependencies, see *z/VM: CP Commands and Utilities Reference* for more information on the parameters used by this function.

2. If *read_password*, *write_password*, and *multi_password* are all not specified, no access passwords will be defined for the disk created.
3. The following table shows the maximum starting locations and *image_disk_size* values allowed for each of the allocation units:

Table 9. Maximum Starting Location and image_disk_size Value Allowed, by Allocation Unit

Allocation Unit	**Maximum Starting Location and *image_disk_size* Value**
BLK0512	2147383640
BLK1024	1073741820
BLK2048	536870910
BLK4096	268435455
CYLINDERS	2147483640

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
		8	RS_OFFLINE	Request successful; object directory offline
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		12	RS_AUTHERR_DM	Request not authorized by directory manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
400	RCERR_IMAGEDEF	4	RS_NOT_FOUND	Image definition not found
		12	RS_LOCKED	Image definition is locked
408	RCERR_IMAGEDISKD	4	RS_EXISTS	Image disk already defined
		24	RS_NO_SPACE	Requested image disk space not available
420	RC_DASD_DM	8	RS_IVS_NAME_NOT_USED	Group, region, or volume name is not defined
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available
592	RCERR_ASYNC_DM	0	RS_NONE	Asynchronous operation started

RC	RC Name	RS	RS Name	Description
596	RCERR_INTERNAL_DM	nnnn	psrc	Internal directory manager error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Image_Disk_Delete

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
image_disk_number_length
image_disk_number

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Image_Disk_Delete to delete a disk from an active virtual image's configuration.

See "Image_Disk_Delete_DM" on page 243 to delete a disk from a virtual image's directory entry.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 17.

function_name
(string,17,char43) The API function name – in this case, 'Image_Disk_Delete'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The name of the image for which the disk is being deleted.

image_disk_number_length
(int4) Length of *image_disk_number*.

image_disk_number
(string,1-4,char16) The virtual device address of the disk to be deleted.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Usage Notes

1. *z/VM: CP Planning and Administration* and *z/VM: CP Commands and Utilities Reference* contain additional information on the parameters used by this function.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server

RC	RC Name	RS	RS Name	Description
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
200	RCERR_IMAGEOP	4	RS_NOT_FOUND	Image not found
		12	RS_NOT_ACTIVE	Image not active
204	RCERR_IMAGEDEVU	8	RS_NOT_EXIST	Image device does not exist
208	RCERR_IMAGEDISKU	8	RS_NOT_IN_USE	Image disk not in use
		28	RS_DEV_INCOMPATIBLE	Device is not a disk
		36	RS_NOT_EXIST	Image disk does not exist
396	RCERR_INTERNAL	nnnn	psrc	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Image_Disk_Delete_DM

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
image_disk_number_length
image_disk_number
data_security_erase

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
operation_id

Purpose

Use Image_Disk_Delete_DM to delete a disk from a virtual image's directory entry.

See "Image_Disk_Delete" on page 240 to delete a disk from an active virtual image's configuration.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 20.

function_name
(string,20,char43) The API function name – in this case, 'Image_Disk_Delete_DM'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See “Client Authentication” on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See “Client Authentication” on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The name of the image for which the disk is being deleted.

image_disk_number_length
(int4) Length of *image_disk_number*.

image_disk_number
(string,1-4,char16) The virtual device address of the disk to be deleted.

data_security_erase
(int1) Indicates whether to erase data from the disk(s) being released, as follows:

0
Unspecified (use installation default)

1
Do not erase (override installation default)

2
Erase (override installation default)

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

operation_id
(int4; range -1-2147483647) The identifier of the task. If the operation is asynchronous and has not completed, *return_code* will be 592, *reason_code* will be 0, and *operation_id* will be in the range 0-2147483647. If the operation is complete, *operation_id* will be -1.

Usage Notes

1. *z/VM: CP Planning and Administration* and *z/VM: CP Commands and Utilities Reference* contain additional information on the parameters used by this function.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
		8	RS_OFFLINE	Request successful; object directory offline
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		12	RS_AUTHERR_DM	Request not authorized by directory manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
400	RCERR_IMAGEDEF	4	RS_NOT_FOUND	Image definition not found
		12	RS_LOCKED	Image definition is locked
408	RCERR_IMAGEDISKD	12	RS_LOCKED	Image device is locked
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available
592	RCERR_ASYNC_DM	0	RS_NONE	Asynchronous operation started
596	RCERR_INTERNAL_DM	nnnn	psrc	Internal directory manager error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Image_Disk_Query

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
vdasd_id=*value*

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
vdasd_array (1)
    vdasd_structure (2)
        vdasd_vdev
        vdasd_rdev
        vdasd_access_type
        vdasd_devtype
        vdasd_size
        vdasd_unit
        vdasd_volid

Note:

1. An array consists of zero or more of its components.
2. A structure consists of one set of its components.

Purpose

Use Image_Disk_Query to display the status of all DASDs accessible to a virtual image, including temporary disks and virtual disks in storage.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 16.

function_name
(string,16,char43) The API function name – in this case, 'Image_Disk_Query'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The userid being queried.

Note: The format for specifying the following additional input parameters is *parameter_name*=*value*, followed by a null (ASCIIZ) terminator. For more information, see "Key-value pair (KVP) input parameters" on page 51.

vdasd_id=*value*
(string,1-4,char36) The virtual device number, or ALL. This is a required parameter.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

vdasd_array
(array) An array consisting of zero or more instances of *vdasd_structure*, as follows:

vdasd_structure
(structure) A structure consisting of one set of the following parameters:

vdasd_vdev
(string,4,char16) The virtual device number.

vdasd_rdev
One of the following:

- (string,4,char16) The real device number of the volume containing the virtual device.
- (string,4,VDSK) Indicating a virtual disk in storage.

vdasd_access_type
(int1) The type of access the userid has to the disk. The following values are possible:

1
R/O

2
R/W

vdasd_devtype
(string,4,char10) The IBM direct access device type.

vdasd_size
(int8) The size of the device, in units as per *vdasd_unit*.

vdasd_unit
(int1) The following values are possible:

1
Cylinders

2
Blocks

vdasd_volid
One of the following strings, terminated by a null (ASCIIZ) character

- (string,1-6,char37) The volume label of the real device on which the user's virtual DASD resides.
- (string,6,(TEMP)) Indicating a temporary disk.
- (string,6,(VDSK)) Indicating a virtual disk in storage.

Usage Notes

1. You can determine if a DASD is a temporary disk or a virtual disk in storage by examining the value of *vdasd_volid*.
2. Syntax errors (RC = 24 and RS = *pprr*) are only applicable to the common input parameters. Syntax checking is *not* performed on the additional input parameter for this API. If a valid parameter is specified multiple times, the last value specified for that parameter will be used. For more information, see “Key-value pair (KVP) input parameters” on page 51.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
8	RC_ERR	8	RS_DEV_NOT_FOUND	Device does not exist
		3002	RS_INVALID_PARAMETER	Invalid parameter name
		3003	RS_INVALID_OPERAND	Invalid parameter operand
		3004	RS_MISSING_PARAMETER	Required parameter missing

RC	RC Name	RS	RS Name	Description
24	RCERR_SYNTAX	*pprr*	*pprr*	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
200	RCERR_IMAGEOP	12	RS_NOT_ACTIVE	Image not active
396	RCERR_INTERNAL	*nnnn*	*psrc*	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Image_Disk_Share

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
image_disk_number_length
image_disk_number
target_image_name_length
target_image_name
target_image_disk_number_length
target_image_disk_number
read_write_mode_length
read_write_mode
optional_password_length
optional_password

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Image_Disk_Share to add a disk that is defined in a virtual image's directory entry to a different active virtual image's configuration.

See "Image_Disk_Share_DM" on page 254 to add a disk that is defined in a virtual image's directory entry to different virtual image's directory entry.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 16.

function_name
(string,16,char43) The API function name – in this case, 'Image_Disk_Share'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The name of the single image attempting to share the disk.

image_disk_number_length
(int4) Length of *image_disk_number*.

image_disk_number
(string,1-4,char16) The virtual device number to assign to the shared disk for *target_identifier*.

target_image_name_length
(int4) Length of *target_image_name*.

target_image_name
(string,1-8,char42) The name of the target (virtual image) that owns the image disk being shared.

target_image_disk_number_length
(int4) Length of *target_image_disk_number*.

target_image_disk_number
(string,1-4,char16) The *target_image_name*'s virtual device address of the disk to be shared.

read_write_mode_length
(int4) Length of *read_write_mode*.

read_write_mode
(string,0-4,char26) The access mode requested for the disk, as seen by the owner when the virtual image is logged on. Valid modes are:

R
Read-only (R/O) access is desired. Access is not allowed if the owner or any other user has a link to the minidisk in write or any exclusive status.

RR
Read-only (R/O) access is desired, even if the owner or another user has a link to the minidisk in write status. Access is denied if any user has the minidisk linked in exclusive status. (This is the default if unspecified.)

W
Write access is desired. The minidisk is not accessible if the owner or any other user has a link to the minidisk in read or write status.

WR
Write access is desired. Only R/O access is allowed if the owner or any other user has a link to the minidisk in read or write status. Access is denied if any exclusive links exist.

M
Multiple access is desired. A write link is allowed to the minidisk unless another user already has write, stable or exclusive access to it, in which case, the minidisk is not accessible to you.

MR
Write or any exclusive access is allowed to the minidisk unless another user already has write access to it, in which case R/O access is allowed to the minidisk. Access is also denied if any exclusive links exist.

MW
Write access is allowed to the disk unconditionally, except for existing stable or exclusive links. Access is denied if any stable or exclusive links exist.

If unspecified, the default is RR.

See *z/VM: CP Planning and Administration* and the *z/VM: CP Commands and Utilities Reference* for additional link mode definitions, prefixes, and suffixes.

optional_password_length
(int4) Length of *optional_password*.

optional_password
(string,0-8,charNB) The password that may be required to share the disk.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Usage Notes

1. The use of some optional parameters requires that other optional parameters be specified as well. If you are uncertain of these interdependencies, see the *z/VM: CP Commands and Utilities Reference* for more information on the parameters used by this function.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter

RC	RC Name	RS	RS Name	Description
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
200	RCERR_IMAGEOP	4	RS_NOT_FOUND	Image not found
	RCERR_IMAGEOP	12	RS_NOT_ACTIVE	Image not active
208	RCERR_IMAGEDISKU	4	RS_IN_USE	Image disk already in use
	RCERR_IMAGEDISKU	8	RS_NOT_IN_USE	Image disk not in use
	RCERR_IMAGEDISKU	12	RS_NOT_AVAILABLE	Image disk not available
	RCERR_IMAGEDISKU	16	RS_CANNOT_SHARE	Image disk cannot be shared as requested
	RCERR_IMAGEDISKU	20	RS_SHARE_DIFF_MODE	Image disk shared in different mode
	RCERR_IMAGEDISKU	28	RS_PW_NEEDED	Image disk does not have required password
	RCERR_IMAGEDISKU	32	RS_BAD_PW	Incorrect password specified for image disk
396	RCERR_INTERNAL	nnnn	psrc	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Image_Disk_Share_DM

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
target_image_disk_number_length
target_image_disk_number
target_image_namer_length
target_image_name
image_disk_number_length
image_disk_number
read_write_mode_length
read_write_mode
optional_password_length
optional_password

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Image_Disk_Share_DM to add a disk that is defined in a virtual image's directory entry to a different virtual image's directory entry

See "Image_Disk_Share" on page 250 to add a disk that is defined in a virtual image's directory entry to a different active virtual image's configuration.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 19.

function_name
(string,19,char43) The API function name – in this case, 'Image_Disk_Share_DM'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The name of the single image attempting to share the disk.

target_image_disk_number_length
(int4) Length of *target_image_disk_number*.

target_image_disk_number
(string,1-4,char16) The *target_image_name*'s virtual device address of the disk to be shared.

target_image_name_length
(int4) Length of *target_image_name*.

target_image_name
(string,1-8,char42) The name of the target (virtual image) that owns the image disk being shared.

image_disk_number_length
(int4) Length of *image_disk_number*.

image_disk_number
(string,1-4,char16) The virtual device number to assign to the shared disk for *target_identifier*.

read_write_mode_length
(int4) Length of *read_write_mode*.

read_write_mode
(string,0-4,char26) The access mode requested for the disk, as seen by the owner when the virtual image is logged on. Valid modes are:

R
Read-only (R/O) access is desired. Access is not allowed if the owner or any other user has a link to the minidisk in write or any exclusive status.

RR
Read-only (R/O) access is desired, even if the owner or another user has a link to the minidisk in write status. Access is denied if any user has the minidisk linked in exclusive status.

W
Write access is desired. The minidisk is not accessible if the owner or any other user has a link to the minidisk in read or write status.

WR
Write access is desired. Only R/O access is allowed if the owner or any other user has a link to the minidisk in read or write status. Access is denied if any exclusive links exist.

M
Multiple access is desired. A write link is allowed to the minidisk unless another user already has write, stable or exclusive access to it, in which case, the minidisk is not accessible to you.

MR
Write or any exclusive access is allowed to the minidisk unless another user already has write access to it, in which case R/O access is allowed to the minidisk. Access is also denied if any exclusive links exist.

MW
Write access is allowed to the disk unconditionally, except for existing stable or exclusive links. Access is denied if any stable or exclusive links exist.

If unspecified, the default is R.

See *z/VM: CP Planning and Administration* and the *z/VM: CP Commands and Utilities Reference* for additional link mode definitions, prefixes, and suffixes.

optional_password_length
(int4) Length of *optional_password*.

optional_password
(string,0-8,charNB) The password that may be required to share the disk.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Usage Notes

1. The use of some optional parameters requires that other optional parameters be specified as well. If you are uncertain of these interdependencies, see the *z/VM: CP Commands and Utilities Reference* for more information on the parameters used by this function.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
	RC_OK	8	RS_OFFLINE	Request successful; object directory offline

RC	RC Name	RS	RS Name	Description
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
	RCERR_AUTH	12	RS_AUTHERR_DM	Request not authorized by directory manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
400	RCERR_IMAGEDEF	4	RS_NOT_FOUND	Image definition not found
	RCERR_IMAGEDEF	12	RS_LOCKED	Image definition is locked
408	RCERR_IMAGEDISKD	4	RS_EXISTS	Image disk already defined
	RCERR_IMAGEDISKD	8	RS_NOT_DEFINED	Image disk not defined
		12	RS_LOCKED	Image device is locked
	RCERR_IMAGEDISKD	16	RS_NO_SHARING	Image disk sharing not allowed by target image definition
	RCERR_IMAGEDISKD	28	RS_PW_NEEDED	Image disk does not have required password
	RCERR_IMAGEDISKD	32	RS_BAD_PW	Incorrect password specified for image disk
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available
592	RCERR_ASYNC_DM	nnnn	opid	Asynchronous operation started - product-specific asynchronous operation ID (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
596	RCERR_INTERNAL_DM	nnnn	psrc	Internal directory manager error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Image_Disk_Unshare

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
image_disk_number_length
image_disk_number

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Image_Disk_Unshare to delete a shared disk from an active virtual image's configuration.

See "Image_Disk_Unshare_DM" on page 261 to delete a shared disk from a virtual image's directory entry.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 18.

function_name
(string,18,char43) The API function name – in this case, 'Image_Disk_Unshare'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The name of the image from which the previously-shared disk is to be removed from the configuration.

image_disk_number_length
(int4) Length of *image_disk_number*.

image_disk_number
(string,1-4,char16) The virtual device address of the previously-shared disk to be removed from the configuration.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Usage Notes

1. *z/VM: CP Planning and Administration* and *z/VM: CP Commands and Utilities Reference* contain additional information on the parameters used by this function.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager

RC	RC Name	RS	RS Name	Description
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
200	RCERR_IMAGEOP	4	RS_NOT_FOUND	Image not found
		12	RS_NOT_ACTIVE	Image not active
208	RCERR_IMAGEDISKU	8	RS_NOT_IN_USE	Image disk not in use
396	RCERR_INTERNAL	nnnn	psrc	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Image_Disk_Unshare_DM

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
image_disk_number_length
image_disk_number
target_image_name_length
target_image_name
target_image_disk_number_length
target_image_disk_number

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Image_Disk_Unshare_DM to delete a shared disk from a virtual image's directory entry.

See "Image_Disk_Unshare" on page 258 to delete a shared disk from an active virtual image's configuration.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 21.

function_name
(string,21,char43) The API function name – in this case, 'Image_Disk_Unshare_DM'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The name of the image from which the previously-shared disk is to be removed from the configuration.

image_disk_number_length
(int4) Length of *image_disk_number*.

image_disk_number
(string,1-4,char16) The virtual device address of the previously-shared disk to be removed from the configuration.

target_image_name_length
(int4) Length of *target_image_name*.

target_image_name
(string,1-8,char42) The name of the target (virtual image) that owns the previously-shared disk to be removed from the configuration.

target_image_disk_number_length
(int4) Length of *target_image_disk_number*.

target_image_disk_number
(string,1-4,char16) The virtual device number previously assigned to the shared disk for *target_identifier*.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Usage Notes

1. *z/VM: CP Planning and Administration* and *z/VM: CP Commands and Utilities Reference* contain additional information on the parameters used by this function.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
		8	RS_OFFLINE	Request successful; object directory offline
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		12	RS_AUTHERR_DM	Request not authorized by directory manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
400	RCERR_IMAGEDEF	4	RS_NOT_FOUND	Image definition not found
		12	RS_LOCKED	Image definition is locked
408	RCERR_IMAGEDISKD	8	RS_NOT_DEFINED	Image disk not defined
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available
592	RCERR_ASYNC_DM	nnnn	opid	Asynchronous operation started - product-specific asynchronous operation ID (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
596	RCERR_INTERNAL_DM	nnnn	psrc	Internal directory manager error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Image_IPL_Characteristics_Define_DM

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
image_ipl_characteristics_length
device_type=*value*
boot_data=*value*
boot_record=*value*
boot_program=*value*
device_vdev=*value*
SCP_data_type=*value*
SCP_data=*value*
Sec_IPL=*value*
noSec_IPL=*value*

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
error_length
error_data

Purpose

Use Image_IPL_Characteristics_Define_DM to create or change the LOADDEV statements in a virtual image's user directory entry. LOADDEV statements specify the location of a program to load and the data to pass to the loaded program. The statements can identify the virtual device number and specify whether the IPL is a secure IPL. The LOADDEV statements in a virtual image's user directory entry are used when the IPL user directory statement indicates a list-directed IPL from an FCP-attached (SCSI) or ECKD device. The parameters persist after the virtual machine logs on and can be used by an IPL command when the virtual machine is running.

See IPL Directory Statement in *z/VM: CP Planning and Administration*.

Input Parameters

input_length
(int4) The total length of all input parameters (after this parameter).

function_name_length
(int4) Length of *function_name* – in this case, 35.

function_name
(string,35,char43) The API function name – in this case, 'Image_IPL_Characteristics_Define_DM'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following values:

- (string,1-8,char42) The user ID under whose authority the AF_INET requests function is performed.
- (string,0-8,char42) The user ID under whose authority the AF_IUCV requests function is performed.

The *authenticated_userid* parameter is optional for AF_IUCV requests. For more information, see "Client Authentication" on page 36.

password_length
(int4) Length of *password*.

password
One of the following values:

- (string,1-200,charNA) The password or passphrase to be used for authentication for AF_INET requests.
- (string,0-200,charNA) The password or passphrase to be used for authentication for AF_IUCV requests.

The *password* parameter is optional for AF_IUCV requests. For more information, see "Client Authentication" on page 36.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The name of the virtual image for which LOADDEV parameters are set by the Image_IPL_Characteristics_Define_DM function.

image_ipl_characteristics_length
(int4) Length of the remaining parameters. The remaining parameters are image IPL characteristics that modify user directory LOADDEV statements.

Note: The format for specifying the following additional input parameters is *parameter_name=value* followed by a null (ASCIIZ) terminator. For more information, see "Key-value pair (KVP) input parameters" on page 51.

device_type=*value*
(string,4,char26) The device_type parameter identifies the storage device type, which must be SCSI or ECKD.

If the user directory contains any existing LOADDEV statements, then the value of the API device_type parameter must match the device type of the LOADDEV statements (SCSI or ECKD). See usage note "5" on page 270.

If the device_type parameter is omitted, then device_type defaults to SCSI.

Other image-IPL characteristics parameters that are specified on the API call yield user directory LOADDEV statements that contain the device_type value.

The device_type parameter determines what other IPL characteristics parameters are valid. When device_type is SCSI, the following parameter that is specific to ECKD devices is not valid:

- boot_record

When device_type is ECKD, the following parameter that is specific to SCSI devices is not valid:

- boot_data

boot_data=*value*

The boot_data parameter is allowed only when both of the following conditions are satisfied:

- The API device_type parameter has a value of SCSI or is not specified.
- If any LOADDEV parameters exist, they indicate SCSI device type.

The boot_data value can contain up to three parts, which correspond to the values of the user directory LOADDEV BR_LBA, LOADDEV LUN, and LOADDEV PORTNAME statements. The parts must be separated by commas and must be in the following order:

1. ***BR_LBA***
   The API *BR_LBA* parameter adds, changes, or deletes the user directory LOADDEV LUN statement. Specify one of the following values:
   - (string,0-16,char16) The logical block address of the boot record of the SCSI device. The valid range is 0 - FFFFFFFFFFFFFFFF. No interior blank space is allowed. The existing LOADDEV BR_LBA statement in the directory entry is deleted and a new LOADDEV BR_LBA statement is added.

     If the *BR_LBA* value is null, then the LOADDEV BR_LBA statement is not changed.
   - (string,6,DELETE) The existing LOADDEV BR_LBA statement in the directory entry is deleted. If no LOADDEV BR_LBA statement exists in the directory entry, then an error is returned.
2. ***LUN***
   The API *LUN* parameter adds, changes, or deletes the user directory LOADDEV LUN statement. Specify one of the following values:
   - (string,0-16,char16) The 8-byte logical unit number of the SCSI device. The valid range is 0 - FFFFFFFFFFFFFFFF. No interior blank space is allowed. The existing LOADDEV LUN statement in the directory entry is deleted and a new LOADDEV LUN statement is added.

     If the *LUN* value is null, then the LOADDEV LUN statement is not changed.
   - (string,6,DELETE) The existing LOADDEV LUN statement in the directory entry is deleted. If no LOADDEV LUN statement exists in the directory entry, then an error is returned.

   If the user directory does not contain a LOADDEV LUN statement, then a list-directed IPL uses a default value of 0. Because 0 is not a valid LUN name, a list-directed IPL attempt that uses the default value for the LOADDEV LUN parameter will fail.
3. ***portname***
   The API *portname* parameter adds, changes, or deletes the user directory LOADDEV PORTNAME statement. Specify one of the following values:
   - (string,0-16,char16) The 8-byte fibre channel port name of the SCSI device. The valid range is 0 - FFFFFFFFFFFFFFFF. No interior blank space is allowed. The existing LOADDEV PORTNAME statement in the directory entry is deleted and a new LOADDEV PORTNAME statement is added.

     If the *portname* value is null, then the LOADDEV PORTNAME statement is not changed.
   - (string,6,DELETE) The existing LOADDEV PORTNAME statement in the directory entry is deleted. If no LOADDEV PORTNAME statement exists in the directory entry, then an error is returned.

   If the user directory does not contain a LOADDEV PORTNAME statement, then a list-directed IPL uses a default value of 0. Because 0 is not a valid device port name, a list-directed IPL attempt that uses the default value for the LOADDEV PORTNAME parameter will fail.

If the boot_data parameter has less than three values separated by commas, then the first value is the *BR_LBA* value and the second value is the *LUN* value.

Blank spaces are tolerated before and after the commas.

The following examples are valid boot_data parameters:

Table 10. boot_data parameter examples

Example	Effect
`boot_data=123ABC456DEF890,12,DELETE`	• The LOADDEV BR_LBA value is changed to 123ABC456DEF890. • The LOADDEV LUN value is changed to 12. • The existing LOADDEV PORTNAME statement is deleted.
`boot_data = A04, 12,`	• The LOADDEV BR_LBA value is changed to A04. • The LOADDEV LUN value is changed to 12. • The LOADDEV PORTNAME value is not changed.
`boot_data = ,,45`	• The LOADDEV BR_LBA and LOADDEV LUN values are not changed. • The LOADDEV PORTNAME value is changed to 45.

If the boot_data parameter is omitted, then the LOADDEV BR_LBA, LOADDEV LUN, and LOADDEV PORTNAME statements are not changed.

boot_record=*value*

The boot_record parameter specifies the boot record location on the ECKD IPL device. The API boot_record parameter adds, changes, or deletes the user directory LOADDEV BOOTREC statement. The boot_record parameter is allowed only when both of the following conditions are satisfied:

- The API device_type parameter specifies a value of ECKD.
- If any LOADDEV parameters exist, they indicate ECKD device type.

Specify one of the following values:

- (string,0-12,char16,plus ' ') The hexadecimal numbers of the cylinder, head, and record where the boot record is located. Leading zeros are not required. The three values must be separated by blank spaces and listed in the following order:

  1. Cylinder
  2. Head
  3. Record

  Values must be in the following ranges:

  – The valid cylinder range is 0 - FFFFFFF.
  – The valid head (or track) range is 0 - F.
  – The valid record range is 1 - FF.

  The existing LOADDEV BOOTREC statement in the directory entry is deleted and a new LOADDEV BOOTREC *cylinder head record* statement is added .
- (string,5,LABEL) A value of LABEL specifies the boot record that is specified in the volume label. The existing LOADDEV BOOTREC statement in the directory entry is deleted and the LOADDEV BOOTREC LABEL statement is added.
- (string,6,DELETE) The current LOADDEV BOOTREC statement in the directory entry is deleted. If no LOADDEV BOOTREC statement exists in the directory entry, then an error is returned.

The following examples are valid boot_record parameters:

Table 11. boot_record parameter examples

Example	Effect
`boot_record = 1234567 A 34`	• The LOADDEV BOOTREC cylinder value is changed to 1234567. • The LOADDEV BOOTREC head value is changed to A. • The LOADDEV BOOTREC record value is changed to 34.
`boot_record = 0004 0 1A`	• The LOADDEV BOOTREC cylinder value is changed to 4. • The LOADDEV BOOTREC head value is changed to 0. • The LOADDEV BOOTREC record value is changed to 1A.
`boot_record = LABEL`	The LOADDEV BOOTREC value is changed to LABEL.
`boot_record = DELETE`	The existing LOADDEV BOOTREC statement is deleted.

If the boot_record parameter is omitted, then the LOADDEV BOOTREC statement is not changed.

boot_program=*value*
The boot_program parameter specifies the boot program that is loaded from the IPL device. The API boot_program parameter adds, changes, or deletes the user directory LOADDEV BOOTPROG statement. Specify one of the following values:

- (string,0-2,char10) A boot program number. The value must be a decimal number in the range 0 - 30. The existing LOADDEV BOOTPROG statement in the directory entry is deleted and a new LOADDEV BOOTPROG statement is added.
- (string,9,AUTOMATIC) Loads the first operating system program (not a dump program) that is listed in the program table. The existing LOADDEV BOOTPROG statement in the directory entry is deleted and the LOADDEV BOOTPROG AUTOMATIC statement is added.
- (string,6,DELETE) The existing LOADDEV BOOTPROG statement in the directory entry is deleted. If no LOADDEV BOOTPROG statement exists in the directory entry, then an error is returned.

If the boot_program parameter is omitted, then the LOADDEV BOOTPROG statement is not changed.

device_vdev=*value*
The device_vdev parameter specifies the virtual device number of the device to IPL. The API device_vdev parameter adds, changes, or deletes the user directory LOADDEV DEVICE statement. Specify one of the following values:

- (string,0-4,char16) The virtual device number of the device to IPL. The existing LOADDEV DEVICE statement in the directory entry is deleted and a new LOADDEV DEVICE statement is added.
- (string,6,DELETE) The existing LOADDEV DEVICE statement in the directory entry is deleted. If no LOADDEV DEVICE statement exists in the directory entry, then an error is returned.

If the device_vdev parameter is omitted, then the LOADDEV DEVICE statement is not changed.

SCP_data_type=*value*
(string,0-3,char10) The SCP_data_type parameter identifies the type of data that is specified in the SCP_data parameter. The API SCP_data_type parameter adds, changes, or deletes the user directory LOADDEV SCPDATA statement. Specify one of the following values:

1
DELETE - The existing LOADDEV SCPDATA statement in the directory entry is deleted. If no LOADDEV SCPDATA statement exists in the directory entry, then an error is returned.

2
EBCDIC - EBCDIC data (code page 924) data. A value of 2 removes any existing HEX operand from the LOADDEV SCPDATA statement. 2 is the default.

3
Hexadecimal - UTF-8 encoded hexadecimal data. A value of 3 adds the HEX operand to the LOADDEV SCPDATA statement.

Notes:

1. If the value of SCP_data_type is 1 (DELETE), then you cannot specify the SCP_data parameter.
2. If the value of SCP_data_type is 2 (EBCDIC) or 3 (hexadecimal), then you must specify the SCP_data parameter.
3. If you omit the SCP_data_type parameter, then the value defaults to 2 (EBCDIC).

SCP_data=*value*
(string,0-4096,charNA,minus ') The SCP_data parameter specifies the SCP data that is passed to the loaded program. The API SCP_data parameter provides SCP data for the user directory LOADDEV SCPDATA statement. The value of SCP_data is restricted by the SCP_data_type parameter:

- If the value of SCP_data_type is 2 (EBCDIC) or if the SCP_data_type parameter is not used, then the value of SCP_data must be a string of 1 - 4096 EBCDIC (code page 924) characters. Single quotation marks (') are not allowed.
- If the value of SCP_data_type is 3 (hexadecimal), then the value of SCP_data is a string of 2 - 4096 hexadecimal characters, 0 - 9 and A - F.

  Remember:

  Valid UTF-8 characters are specified as 2-8 hexadecimal characters. These hexadecimal values are always in pairs (2,4,6 or 8 hexadecimal characters). When the SCP_data_type is 3 (hexadecimal), the SCP_data must be specified as hexadecimal pairs. Therefore, the length of the hexadecimal data must be an even number of hexadecimal numbers that represent valid UTF-8 values.

The SCP_data parameter must consist of binary EBCDIC codes when transmitted on the wire.

If the SCP_data parameter is omitted, then the LOADDEV SCPDATA statement is not changed.

Sec_IPL=*value*
(string,0-6,char26) The Sec_IPL parameter specifies the SECURE IPL option. The API sec_IPL parameter adds or deletes the user directory LOADDEV SECURE statement and can delete the LOADDEV NOSECURE statement. Specify one of the following values:

- YES - The IPL uses the SECURE option. An existing LOADDEV NOSECURE statement in the directory entry is deleted and the LOADDEV SECURE statement is added.
- DELETE - The existing LOADDEV SECURE statement in the directory entry is deleted. If no LOADDEV SECURE statement exists in the directory entry, then an error is returned.

If the API sec_IPL parameter is omitted, then an existing LOADDEV SECURE statement is changed (deleted) only if the API noSec_IPL parameter specifies YES.

noSec_IPL=*value*
(string,0-6,char26) The noSec_IPL parameter specifies the NOSECURE IPL option. The API sec_IPL parameter adds or deletes the user directory LOADDEV NOSECURE statement and can delete the LOADDEV SECURE statement. Specify one of the following values:

- YES - The IPL uses the NOSECURE option. An existing LOADDEV SECURE statement in the directory entry is deleted and the LOADDEV NOSECURE statement is added.
- DELETE - The existing LOADDEV NOSECURE statement in the directory entry is deleted. If no LOADDEV SECURE statement exists in the directory entry, then an error is returned.

If the API noSec_IPL parameter is omitted, then an existing LOADDEV NOSECURE statement is changed (deleted) only if the API sec_IPL parameter specifies YES.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this parameter).

request_id
(int4) The identifier of the request (The same value as returned in "Response 1 -- Immediate Request Verification" on page 270).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

error_length
(int4) The length of the error_data parameter. The error_length parameter is returned only when RC=8 and RS=24, 3002, 3003, or 3004.

error_data
(int4) A blank-delimited string that describes the error cause. The string is followed by a null (ASCIIZ) terminator. The error_data parameter is returned only when RC=8 and RS=24, 3002, 3003, or 3004.

Usage Notes

1. The image-IPL characteristics parameters are expressed as key-value pairs (KVPs). The format for specifying the parameters is *parameter_name=value*, followed by a null (ASCIIZ) terminator. For more information, see "Key-value pair (KVP) input parameters" on page 51.
2. If a KVP parameter is not specified, then the corresponding LOADDEV directory statement is not changed. There are two exceptions:
   - If the device_type parameter is not specified, then a value of SCSI is assumed.
   - If the SCP_data_type parameter is not specified, then a value of 2 (EBCDIC) is assumed.
3. The KVP parameters accept a null value except for the following parameters:
   - device_type
   - SCP_data_type
   - SCP_data, when the value of SCP_data_type is 2 or 3.

   When a null value is specified for a parameter that accepts a null value, the corresponding LOADDEV statement is not changed.
4. The KVP parameters are optional in most cases. Parameters are required in the following cases:
   - If the value of the SCP_data_type parameter is 2 (EBCDIC) or 3 (hexadecimal), then the SCP_data parameter must be specified with a non-null value.
   - The device_type parameter cannot be the only input parameter. At least one other input parameter is required when device_type is specified.
5. If the user directory contains any existing LOADDEV statements, then the value of the API device_type parameter must match the device type of the LOADDEV statements (SCSI or ECKD). If the device types do not match, then existing LOADDEV statements are not changed.

Unlike the SET LOADDEV command, the Image_IPL_Characteristics_Define_DM API call has no CLEAR operand that clears all existing LOADDEV parameters and resets the device type. You can use the Image_IPL_Characteristics_Define_DM API call to change the LOADDEV device type:

a. Delete all existing LOADDEV statements.

b. Specify the new device type and characteristics of that IPL device.

6. If a parameter specifies DELETE, then there must an existing LOADDEV statement for that parameter. If no existing LOADDEV statement for that parameter exists, then an error occurs.
7. More information on the directory format and on specific LOADDEV directory statements is available. See LOADDEV Directory Statement and Creating and Updating a User Directory in *z/VM: CP Planning and Administration*.

Return and Reason Codes

RC	RC Name	RS Value	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
8	RC_ERR	24	RS_CONFLICTING_PARMS	Conflicting parameters
		3002	RS_INVALID_PARAMETER	Invalid parameter name
		3003	RS_INVALID_OPERAND	Invalid parameter operand
		3004	RS_MISSING_PARAMETER	Missing a required parameter
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		12	RS_AUTHERR_DM	Request not authorized by directory manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; user ID or password not valid
400	RCERR_IMAGEDEF	4	RS_NOT_FOUND	Image definition not found
		12	RS_LOCKED	Image or profile definition is locked
404	RCERR_IMAGEDEVD	8	RS_DEV_NOT_FOUND	The parameter to be deleted is not defined.
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available
520	RCERR_UTF_8	2826	RS_INVALID_UTF_8	SCPDATA contains invalid UTF-8 data
592	RCERR_ASYNC_DM	nnnn	opid	Asynchronous operation started - product-specific asynchronous operation ID. (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)

RC	RC Name	RS Value	RS Name	Description
596	RCERR_INTERNAL_DM	nnnn	psrc	Internal directory manager error - product-specific return code. (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Image_IPL_Characteristics_Query_DM

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
ipl_array_length
ipl_array

Purpose

Use Image_IPL_Characteristics_Query_DM to query the LOADDEV statements in a virtual image's user directory entry. LOADDEV statements specify the location of a program to load and the data to pass to the loaded program. The statements can identify the virtual device number and specify whether the IPL is a secure IPL. The LOADDEV statements in a virtual image's user directory entry are used when the IPL user directory statement indicates a list-directed IPL from an FCP-attached (SCSI) or ECKD device. The parameters persist after the virtual machine logs on and can be used by an IPL command when the virtual machine is running.

See IPL Directory Statement in *z/VM: CP Planning and Administration*.

Input Parameters

input_length
(int4) The total length of all input parameters (after this parameter).

function_name_length
(int4) Length of *function_name* – in this case, 34.

function_name
(string,34,char43) The API function name – in this case, Image_IPL_Characteristics_Query_DM.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following values:

- (string,1-8,char42) The user ID under whose authority the AF_INET requests function is performed.

- (string,0-8,char42) The user ID under whose authority the AF_IUCV requests function is performed.

The *authenticated_userid* parameter is optional for AF_IUCV requests. For more information, see "Client Authentication" on page 36.

password_length
(int4) Length of *password*.

password
One of the following values:

- (string,1-200,charNA) The password or passphrase to be used for authentication of AF_INET requests.
- (string,0-200,charNA) The password or passphrase to be used for authentication of AF_IUCV requests.

The *password* parameter is optional for AF_IUCV requests. For more information, see "Client Authentication" on page 36.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The name of the virtual image for which LOADDEV parameters are queried.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this parameter).

request_id
(int4) The identifier of the request (same as returned in "Response 1 -- Immediate Request Verification" on page 274.

return_code
(int4) The return code.

reason_code
(int4) The reason code.

ipl_array_length
(int4) The length of the *ipl_array* parameter.

ipl_array
(array) An array consisting of one or more instances of *ipl_structure*.

ipl_structure
(structure) A structure that consists of one set of the following parameters:

loaddev_length
(int4) The length of the *loaddev_statement* parameter.

loaddev_statement
(string,1-4115) The value of the LOADDEV directory statement.

Usage Notes

1. The output data returns the LOADDEV statement data without the "LOADDEV" word at the beginning of each line.

2. More information on the directory format and on specific LOADDEV directory statements is available. See LOADDEV Directory Statement and Creating and Updating a User Directory in *z/VM: CP Planning and Administration*.

Return and Reason Codes

RC	RC Name	RS Value	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
		28	RS_NONE_FOUND	Return buffer is empty
		3002	RS_INVALID_PARAMETER	Invalid parameter name
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		12	RS_AUTHERR_DM	Request not authorized by directory manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; user ID or password not valid
400	RCERR_IMAGEDEF	4	RS_NOT_FOUND	Image definition not found
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available
592	RCERR_ASYNC_DM	nnnn	opid	Asynchronous operation started - product-specific asynchronous operation ID. (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
596	RCERR_INTERNAL_DM	nnnn	psrc	Internal directory manager error - product-specific return code. (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Image_IPL_Delete_DM

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Image_IPL_Delete_DM to delete the IPL statement from a virtual image's directory entry or a profile directory entry.

If there is no IPL statement in a virtual image's directory entry, then no operating system is automatically loaded and started when the virtual image is activated.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 19.

function_name
(string,19,char43) The API function name – in this case, 'Image_IPL_Delete_DM'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) Specifies the name of the user or profile for which the IPL statement is to be deleted.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Return and Reason Codes

RC	RC Name	RS Value	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
		8	RS_OFFLINE	Request successful; object directory offline
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter pprr
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		12	RS_AUTHERR_DM	Request not authorized by directory manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
400	RCERR_IMAGEDEF	4	RS_NOT_FOUND	Image definition not defined

RC	RC Name	RS Value	RS Name	Description
		12	RS_LOCKED	Image definition is locked
408	RCERR_IMAGEDISKD	8	RS_NOT_DEFINED	Image disk not defined
460	RC_IPL_DM	4	RS_IPL_NOT_FOUND	Image does not have an IPL statement
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available
592	RCERR_ASYNC_DM	nnnn	opid	Asynchronous operation started - product-specific asynchronous operation ID (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
596	RCERR_INTERNAL_DM	nnnn	psrc	Internal directory manager error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Image_IPL_Query_DM

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
saved_system_length
saved_system
load_parameter_length
load_parameter
parameter_string_length
parameter_string

Purpose

Use Image_IPL_Query_DM to query the information about the operating system, or device that contains the operating system, that is specified on the IPL statement in a virtual image's directory entry or a profile directory entry. This operating system is automatically loaded and started when the virtual image is activated.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 18.

function_name
(string,18,char43) The API function name – in this case, 'Image_IPL_Query_DM'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) Specifies the name of the user or profile for which the IPL statement is to be queried.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

saved_system_length
(int4) Length of *saved_system*.

saved_system
(string,1-8,char42) Specifies the name of the saved system or virtual device address of the device containing the system to be loaded or the LOADDEV keyword.

load_parameter_length
(int4) Length of *load_parameter*.

load_parameter
(string,0-10,char) Specifies the load parameter (up to 8 characters) that is used by the IPL'd system. Note that the load parameter may be enclosed in single quotes.

parameter_string_length
(int4) Length of *parameter_string*.

parameter_string
(string,0-64,char) Specifies the parameters to be passed to the IPL'd operating system. Although the IPL command allows for 64 bytes of parameters, the string on the directory statement is limited to the number of characters that can be specified in the first 72 positions of the statement.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
		28	RS_NONE_FOUND	No matching entries found. Return buffer is empty.
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter pprr
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		12	RS_AUTHERR_DM	Request not authorized by directory manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
400	RCERR_IMAGEDEF	4	RS_NOT_FOUND	Image definition not defined
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available
592	RCERR_ASYNC_DM	nnnn	opid	Asynchronous operation started - product-specific asynchronous operation ID (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
596	RCERR_INTERNAL_DM	nnnn	psrc	Internal directory manager error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Image_IPL_Set_DM

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
saved_system_length
saved_system
load_parameter_length
load_parameter
parameter_string_length
parameter_string

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Image_IPL_Set_DM to add an IPL statement to a virtual image's directory entry or a profile directory entry. The IPL statement identifies an operating system, or a device that contains an operating system, which is automatically loaded and started when the virtual image is activated.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 16.

function_name
(string,16,char43) The API function name – in this case, 'Image_IPL_Set_DM'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).

- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) Specifies the name of the user or profile for which the IPL statement is to be set.

saved_system_length
(int4) Length of *saved_system*.

saved_system
(string,1-8,char42) Specifies the name of the saved system or virtual device address of the device that contains the system to be loaded or the LOADDEV keyword.

load_parameter_length
(int4) Length of *load_parameter*.

load_parameter
(string,0-10,char) Specifies the load parameter (up to 8 characters) that is used by the IPL'd system. It might be necessary to enclose the load parameter in single quotation marks.

parameter_string_length
(int4) Length of *parameter_string*.

parameter_string
(string,0-64,char) Specifies the parameters to be passed to the IPL'd operating system. Although the IPL command allows for 64 bytes of parameters, the string on the directory statement is limited to the number of characters that can be specified in the first 72 positions of the statement.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
		8	RS_OFFLINE	Request successful; object directory offline
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter pprr
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		12	RS_AUTHERR_DM	Request not authorized by directory manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
400	RCERR_IMAGEDEF	4	RS_NOT_FOUND	Image definition not defined
		12	RS_LOCKED	Image definition is locked
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available
592	RCERR_ASYNC_DM	nnnn	opid	Asynchronous operation started - product-specific asynchronous operation ID (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
596	RCERR_INTERNAL_DM	nnnn	psrc	Internal directory manager error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Image_Lock_DM

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
device_address_length
device_address

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Image_Lock_DM to lock a virtual image's directory entry or a specific device in a virtual image's directory entry so that it cannot be changed.

Use this function before replacing a virtual image's directory entry with Image_Replace_DM.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 13.

function_name
(string,13,char43) The API function name – in this case, 'Image_Lock_DM'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The name of the image to be locked.

device_address_length
(int4) Length of *device_address*.

device_address
(string,0-4,char16) The virtual address of the device being locked.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Usage Notes

1. Use Image_Lock_DM before an Image_Replace_DM operation. The Image_Replace_DM operation will unlock the image upon completion. If, after locking the image, you do not perform the Image_Replace_DM, use Image_Unlock_DM to unlock the image.
2. To lock an entire image, omit the device address. To lock a specific device, specify the device address.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager

RC	RC Name	RS	RS Name	Description
		12	RS_AUTHERR_DM	Request not authorized by directory manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
400	RCERR_IMAGEDEF	4	RS_NOT_FOUND	Image definition not defined
		12	RS_LOCKED	Image definition is locked
404	RCERR_IMAGEDEVD	8	RS_NOT_DEFINED	Image device not defined
		12	RS_LOCKED	Image device is locked
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available
592	RCERR_ASYNC_DM	nnnn	opid	Asynchronous operation started - product-specific asynchronous operation ID (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
596	RCERR_INTERNAL_DM	nnnn	psrc	Internal directory manager error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Image_Lock_Query_DM

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
lock_info_structure (2)
  lock_info_structure_length
  locked_type
  image_locked_by
locked_dev_array_length
locked_dev_array (1)
  dev_lock_info_structure (2)
    dev_address
    dev_locked_by

Note:

1. An array consists of zero or more of its components.
2. A structure consists of one set of its components.

Purpose

Use Image_Lock_Query_DM to query the status of directory manager locks in effect for a specific virtual image.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 19.

function_name
(string,13,char43) The API function name – in this case, 'Image_Lock_Query_DM'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The name of the image for which the directory lock status is being queried.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

lock_info_structure
(structure) A structure consisting of the following blank-delimited components (this structure will be absent when RS=RS_UNLOCKED):

lock_info_structure_length
(int4) The combined length of the remaining parameters in *lock_info_structure* (not including this parameter). This will be zero when RS=RS_UNLOCKED.

locked_type
(string,5-6,char26) One of the following:

IMAGE
Image locked

DEVICE
Device(s) locked

image_locked_by
(string,0-8,char42) The image that performed the image lock. This will be absent if *locked_type*=DEVICE.

locked_dev_array_length
(int4) Length of *locked_dev_array*. This array will be absent if RS = RS_UNLOCKED or *locked_type* = IMAGE.

locked_dev_array
(array) An array consisting of zero or more instances of *dev_lock_info_structure*, as follows:

dev_lock_info_structure
(structure) A structure consisting of one set of the following parameters:

dev_address
(string,1-4,char16) The address of the locked device.

dev_locked_by
(string,1-8,char42) The image that performed the device lock action.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	12	RS_LOCKED	Image or device(s) locked
		24	RS_UNLOCKED	Image or device(s) unlocked
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		12	RS_AUTHERR_DM	Request not authorized by directory manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
400	RCERR_IMAGEDEF	4	RS_NOT_FOUND	Image definition not defined
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available
592	RCERR_ASYNC_DM	nnnn	opid	Asynchronous operation started - product-specific asynchronous operation ID (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
596	RCERR_INTERNAL_DM	nnnn	psrc	Internal directory manager error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist

RC	RC Name	RS	RS Name	Description
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Image_MDISK_Link_Query

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
vdev=*value*

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
link_array_length
link_array (1)
link_structure (2)
system_name
user
vaddr
access_mode

Note:

1. An array consists of zero or more of its components.
2. A structure consists of one set of its components.

Purpose

Use Image_MDISK_Link_Query to query the links to an image's MDISK.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 22.

function_name
(string,13,char43) The API function name – in this case, 'Image_MDISK_Link_Query'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The name of the image for which a virtual dasd link is being queried.

Note: The format for specifying the following additional input parameters is *parameter_name=value*, followed by a null (ASCIIZ) terminator. For more information, see "Key-value pair (KVP) input parameters" on page 51.

vdev=*value*
(string,1-4,char16) The VDEV address of the virtual DASD which is being queried for links. This is a required parameter.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

link_array_length
(int4) Length of *link_array*.

link_array
(array) An array consisting of zero or more instances of *link_structure*, with each structure terminated by a null (ASCIIZ) character, as follows:

link_structure
(structure) A structure consisting of one set of the following parameters, with a blank separating each parameter:

system_name
(string,1-8,char42) The name of the system where the user linked to the MDISK is active.

user
(string,1-8,char42) The user that is linked to the MDISK.

vaddr
(string,1-4,char16) The virtual address the MDISK is linked as by the user.

access_mode
(string,4-5,char26) One of the following:

WRITE
The disk is linked in read-write mode.

READ
The disk is linked in read-only mode.

Usage Notes

1. Syntax errors (RC = 24 and RS = *pprr*) are only applicable to the common input parameters. Syntax checking is *not* performed on the additional input parameters for this API. If a valid parameter is specified multiple times, the last value specified for that parameter will be used. For more information, see “Key-value pair (KVP) input parameters” on page 51.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
		28	RS_LINK_NOT_FOUND	No links to disk found
8	RC_ERR	8	RS_DEV_NOT_FOUND	Device does not exist
		12	RS_NOT_LOGGED_ON	*target_identifier* not logged on
		24	RS_CONFLICTING_PARMS	Conflicting parameters
		3002	RS_INVALID_PARAMETER	Invalid parameter name
		3003	RS_INVALID_OPERAND	Invalid parameter operand
		3004	RS_MISSING_PARAMETER	Required parameter missing
24	RCERR_SYNTAX	*pprr*	*pprr*	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
396	RCERR_INTERNAL	*nnnn*	*psrc*	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)

RC	RC Name	RS	RS Name	Description
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Image_Name_Query_DM

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
image_name_array_length
image_name_array (1)
image_name_structure (2)
image_name_length
image_name

Note:

1. An array consists of zero or more of its components.
2. A structure consists of one set of its components.

Purpose

Use Image_Name_Query_DM to obtain a list of defined virtual images.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 19.

function_name
(string,19,char43) The API function name – in this case, 'Image_Name_Query_DM'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) This must match an entry in the authorization file that also contains the *authenticated_userid* and the *function_name* (Image_Name_Query_DM).

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

image_name_array_length
(int4) Length of *image_name_array*.

image_name_array
(array) An array consisting of zero or more instances of *image_name_structure*, as follows:

image_name_structure
(structure) A structure consisting of one set of *image_name_length* and *image_name*, as follows:

image_name_length
(int4) Length of *image_name*.

image_name
(string,1-8,char42) The name of the image.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful

RC	RC Name	RS	RS Name	Description
		28	RS_NONE_FOUND	No matching entries found. Return buffer is empty.
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		12	RS_AUTHERR_DM	Request not authorized by directory manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available
592	RCERR_ASYNC_DM	nnnn	opid	Asynchronous operation started - product-specific asynchronous operation ID (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
596	RCERR_INTERNAL_DM	nnnn	psrc	Internal directory manager error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Image_Password_Set_DM

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
image_password_length
image_password

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Image_Password_Set_DM to set or change a virtual image's password.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 21.

function_name
(string,21,char43) The API function name – in this case, 'Image_Password_Set_DM'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The name of the image for which the password is being set.

image_password_length
(int4) Length of *image_password*.

image_password
(string,1-200,charNA) The password or passphrase to set for the image.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
		8	RS_OFFLINE	Request successful; object directory offline
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		12	RS_AUTHERR_DM	Request not authorized by directory manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid

RC	RC Name	RS	RS Name	Description
400	RCERR_IMAGEDEF	4	RS_NOT_FOUND	Image definition not found
		12	RS_LOCKED	Image definition is locked
444	RCERR_POLICY_PW	0	RS_NONE	Password policy error
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available
		20	RS_PW_FORMAT_NOT _SUPPORTED	Password format not supported
592	RCERR_ASYNC_DM	nnnn	opid	Asynchronous operation started - product-specific asynchronous operation ID (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
596	RCERR_INTERNAL_DM	nnnn	psrc	Internal directory manager error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Image_Pause

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
`action`=*value*

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Image_Pause to pause a running image and to restart a paused image.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 18.

function_name
(string,18,char43) The API function name – in this case, 'Image_Pause'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The user ID under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The user ID under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See “Client Authentication” on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) A string that must match an entry in the authorization file that also contains the *authenticated_userid* and the *function_name* (Image_Pause).

`action`=*value*
This is a required parameter. The format for specifying a required parameter is *`parameter_name=value`*, followed by a null (ASCIIZ) terminator. In this case, *value* can be one of the following:

- (string,5,char26) PAUSE (this is the default).
- (string,7,char26) UNPAUSE

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
8	RC_ERR	12	RS_NOT_ACTIVE	Image not logged on
		3002	RS_INVALID_PARAMETER	Invalid parameter name
		3003	RS_INVALID_OPERAND	Invalid parameter operand
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; user ID or password not valid

RC	RC Name	RS	RS Name	Description
396	RCERR_INTERNAL	nnnn	psrc	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Image_Query_Activate_Time

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
date_format_indicator

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
image_name_length
image_name
activation_date_length
activation_date
activation_time_length
activation_time

Purpose

Use Image_Query_Activate_Time to obtain the date and time when a virtual image was activated.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 25.

function_name
(string,25,char43) The API function name – in this case, 'Image_Query_Activate_Time'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).

- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See “Client Authentication” on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See “Client Authentication” on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) To specify which virtual image's activation date and time is being queried.

date_format_indicator
(int1) The format of the date stamp that is returned:

1
mm/dd/yy

2
mm/dd/yyyy

3
yy-mm-dd

4
yyyy-mm-dd

5
dd/mm/yy

6
dd/mm/yyyy

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

image_name_length
(int4) Length of *image_name*.

image_name
(string,1-8,char42) Name of the image.

activation_date_length
(int4) Length of *activation_date*.

activation_date
(string,8-10,char) Date the virtual image was activated.

activation_time_length
(int4) Length of *activation_time*. This value will always be 8.

activation_time
(string,8,char) Time the virtual image was activated.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter pp
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not Authorized by External Security Manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
200	RCERR_IMAGEOP	12	RS_NOT_ACTIVE	Image not active
396	RCERR_INTERNAL	nnnn	psrc	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Image_Query_DM

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
image_record_array_length
image_record_array (1)
image_record_structure (2)
image_record_length
image_record

Note:

1. An array consists of zero or more of its components.
2. A structure consists of one set of its components.

Purpose

Use Image_Query_DM to obtain a virtual image's directory entry.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 14.

function_name
(string,14,char43) The API function name – in this case, 'Image_Query_DM'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The name of the image being queried.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

image_record_array_length
(int4) Length of *image_record_array*.

image_record_array
(array) An array consisting of zero or more instances of *image_record_structure*, as follows:

image_record_structure
(structure) A structure consisting of one set of *image_record_length* and *image_record*, as follows:

image_record_length
(int4) Length of *image_record*.

image_record
(string,1-80,charNA) A record from the virtual image's directory entry.

Usage Notes

1. See the "Creating and Updating a User Directory" chapter in *z/VM: CP Planning and Administration* for more information on the directory format and on specific directory statements.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		12	RS_AUTHERR_DM	Request not authorized by directory manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
400	RCERR_IMAGEDEF	4	RS_NOT_FOUND	Image definition not found
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available
592	RCERR_ASYNC_DM	nnnn	opid	Asynchronous operation started - product-specific asynchronous operation ID (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
596	RCERR_INTERNAL_DM	nnnn	psrc	Internal directory manager error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Image_Recycle

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
recycled
not_recycled
failing_array_length
failing_array (1)
failing_structure (2)
failing_structure_length
image_name_length
image_name
return_code
reason_code

Note:

1. An array consists of zero or more of its components.
2. A structure consists of one set of its components.

Purpose

Use Image_Recycle to deactivate and then reactivate a virtual image or list of virtual images. If the specified virtual image (or a virtual image in the specified list) is not active, it remains inactive.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 13.

function_name
(string,13,char43) The API function name – in this case, 'Image_Recycle'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
One of the following:

- (string,1-8,char42) The name of the image being recycled.
- (string,1-64,char43) The name of a list containing names of images to be recycled.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

recycled
(int4) The number of images recycled successfully.

not_recycled
(int4) The number of images *not* recycled successfully.

failing_array_length
(int4) Length of *failing_array*.

failing_array
(array) An array consisting of zero or more instances of *failing_structure* for every image that failed, as follows:

failing_structure
(structure) A structure consisting of one set of the following parameters:

failing_structure_length
(int4) The combined length of the remaining parameters in *failing_structure* (not including this parameter).

image_name_length
(int4) Length of *image_name*.

image_name
(string,1-8,char42) The name of the image.

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Usage Notes

1. This function checks the name to determine whether it is a list, and if not, processes the name as a single image name. Therefore, lists should be given names that cannot be confused with image names.
2. During authorization checking and function processing, name lists are only expanded once; although a name within a list may also be the name of a list, the second (nested) list will not be expanded.
3. By default, this function waits a maximum of 120 seconds (2 minutes) per specified image for each image to deactivate before attempting to reactivate that image. Images that take longer than 120 seconds to deactivate are not reactivated. The maximum deactivation wait time can be increased (or decreased) by changing the value of the `Max_Image_Wait_Time =` attribute in the DMSSICNF COPY file. For more information, see “Configuring SMAPI” on page 30.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
200	RCERR_IMAGEOP	4	RS_NOT_FOUND	Image not found
		12	RS_NOT_ACTIVE	Image not active
		16	RS_BEING_DEACT	Image being deactivated
		24	RS_LIST_NOT_FOUND	List not found
		36	RS_SOME_NOT_RECYC	Some images in list not recycled

RC	RC Name	RS	RS Name	Description
396	RCERR_INTERNAL	nnnn	psrc	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Image_Replace_DM

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
image_record_array_length
image_record_array (1)
image_record_structure (2)
image_record_length
image_record

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Note:

1. An array consists of zero or more of its components.
2. A structure consists of one set of its components.

Purpose

Use Image_Replace_DM to replace a virtual image's directory entry.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 16.

function_name
(string,16,char43) The API function name – in this case, 'Image_Replace_DM'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The name of the image to be replaced.

image_record_array_length
(int4) Length of *image_record_array*.

image_record_array
(array) An array consisting of zero or more instances of *image_record_structure*, as follows:

image_record_structure
(structure) A structure consisting of one set of *image_record_length* and *image_record*, as follows:

image_record_length
(int4) Length of *image_record*.

image_record
(string,1-72,charNA) A record from the virtual image's directory entry.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Usage Notes

1. Use Image_Lock_DM before an Image_Replace_DM operation. The Image_Replace_DM operation will unlock the image upon completion. If, after locking the image, you do not perform the Image_Replace_DM, use Image_Unlock_DM to unlock the image.

2. See the "Creating and Updating a User Directory" chapter in *z/VM: CP Planning and Administration* for more information on the directory format and on specific directory statements.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
		8	RS_OFFLINE	Request successful; object directory offline
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		12	RS_AUTHERR_DM	Request not authorized by directory manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
400	RCERR_IMAGEDEF	4	RS_NOT_FOUND	Image definition not found
		24	RS_NOT_LOCKED	Image name is not locked
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available
592	RCERR_ASYNC_DM	nnnn	opid	Asynchronous operation started - product-specific asynchronous operation ID (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
596	RCERR_INTERNAL_DM	nnnn	psrc	Internal directory manager error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Image_SCSI_Characteristics_Define_DM

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
boot_program_length
boot_program
BR_LBA_length
BR_LBA
LUN_length
LUN
port_name_length
port_name
SCP_data_type
SCP_data_length
SCP_data

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Image_SCSI_Characteristics_Define_DM to update the LOADDEV statements in a virtual image's user directory source file. LOADDEV statements specify the location of a program to load and the data to pass to the loaded program. The statements can identify the virtual device number and specify whether the IPL is a secure IPL. The parameters that are specified by the LOADDEV statements are used when the IPL user directory statement indicates a list-directed IPL.

Parameters for a secure IPL (*fcp_vdev* and SECURE) are not supported. See “Image_IPL_Characteristics_Define_DM” on page 264.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 36.

function_name
(string,36,char43) The API function name – in this case, 'Image_SCSI_Characteristics_Define_DM'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The target image name whose LOADDEV is being set.

boot_program_length
(int4) The length of *boot_program*.

boot_program
(string,0-6,char10) The boot program number (which must be a value in the range 0 to 30), or the keyword "DELETE" to delete the existing boot program number. If null, the boot program number will be unchanged.

BR_LBA_length
(int4) The length of *BR_LBA*.

BR_LBA
(string,0-16,char16) The logical-block address of the boot record, or the keyword "DELETE" to delete the existing logical-block address. If null, the logical-block address will be unchanged.

LUN_length
(int4) The length of *LUN*.

LUN
(string,0-16,char16) The logical unit number, or the keyword "DELETE" to delete the existing logical unit number. If null, the logical unit number will be unchanged.

port_name_length
(int4) The length of *port_name*.

port_name
(string,0-16,char16) The port name, or the keyword "DELETE" to delete the existing port name. If null, the port name will be unchanged.

SCP_data_type
(int1) The type of data specified in the *SCP_data* parameter, as follows:

0
Unspecified

1
DELETE – delete the *SCP_data* for the image

2
EBCDIC – EBCDIC (codepage 924) data

3
HEX – UTF-8 encoded hex data

Note:

1. If *SCP_data_type* is 0 (unspecified) or 1 (DELETE), then *SCP_data* must *not* be specified.
2. If *SCP_data_type* is 2 (EBCDIC) or 3 (HEX), then *SCP_data must* be specified.

SCP_data_length
(int4) The length of *SCP_data*.

SCP_data
(string,0-4096,charNA) The SCP data, which can be any of the following:

- If *SCP_data_type* is 2 (EBCDIC), then *SCP_data* is a string of up to 4096 EBCDIC (codepage 924) characters.
- If *SCP_data_type* is 3 (HEX), then *SCP_data* is a string of up to 4096 EBCDIC characters '0'-'9' and 'A'-'F', therefore representing up to 2048 UTF-8 data bytes. (Two hexadecimal characters are required to represent one UTF-8 data byte.)

Note that *SCP_data* must consist of binary EBCDIC codes on the wire.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Usage Notes

1. See the "Creating and Updating a User Directory" chapter in *z/VM: CP Planning and Administration* for more information on the directory format and on specific directory statements.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
		8	RS_OFFLINE	Request successful; object directory offline

RC	RC Name	RS	RS Name	Description
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter pp
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		12	RS_AUTHERR_DM	Request not authorized by directory manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
400	RCERR_IMAGEDEF	4	RS_NOT_FOUND	Image definition not found
456	RCERR_SCSI	4	RS_LOADDEV_NOT_FOUND	LOADDEV statement not found
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available
520	RCERR_UTF8	2826	RS_INVALID_UTF_DATA	SCPDATA contains invalid UTF-8 data
592	RCERR_ASYNC_DM	nnnn	opid	Asynchronous operation started - product-specific asynchronous operation ID (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
596	RCERR_INTERNAL_DM	nnnn	psrc	Internal directory manager error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Image_SCSI_Characteristics_Query_DM

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
boot_program_length
boot_program
BR_LBA_length
BR_LBA
LUN_length
LUN
port_name_length
port_name
SCP_data_type
SCP_data_length
SCP_data

Purpose

Use Image_SCSI_Characteristics_Query_DM to query the LOADDEV statements in a virtual image's user directory source file. LOADDEV statements specify the location of a program to load and the data to pass to the loaded program. The statements can identify the virtual device number and specify whether the IPL is a secure IPL. The LOADDEV statements in a virtual image's user directory entry are used when the IPL user directory statement indicates a list-directed IPL from an FCP-attached (SCSI) or ECKD device. The parameters persist after the virtual machine logs on and can be used by an IPL command when the virtual machine is running.

Parameters for a secure IPL (*fcp_vdev* and SECURE) are not supported. See "Image_IPL_Characteristics_Query_DM" on page 273.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 35.

function_name
(string,35,char43) The API function name – in this case, 'Image_SCSI_Characteristics_Query_DM'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The target userid whose LOADDEV is being queried.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

boot_program_length
(int4) The length of *boot_program*.

boot_program
(string,0-6,char10) The boot program number.

BR_LBA_length
(int4) The length of *BR_LBA*.

BR_LBA
(string,0-16,char16) The logical-block address of the boot record.

LUN_length
(int4) The length of *LUN*.

LUN
(string,0-16,char16) The logical unit number.

port_name_length
(int4) The length of *port_name*.

port_name
(string,0-16,char16) The port name.

SCP_data_type
(int1) The type of data specified in the *SCP_data* parameter, as follows:

0
Unspecified

2
EBCDIC – EBCDIC (codepage 924) data

3
HEX – UTF-8 encoded hex data

SCP_data_length
(int4) The length of *SCP_data*.

SCP_data
(string,0-4096,charNA) The SCP data.

Usage Notes

1. See the "Creating and Updating a User Directory" chapter in *z/VM: CP Planning and Administration* for more information on the directory format and on specific directory statements.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
0	RC_OK	28	RS_EMPTY	There are no SCSI characteristics for this image.
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter pp
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		12	RS_AUTHERR_DM	Request not authorized by directory manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
400	RCERR_IMAGEDEF	4	RS_NOT_FOUND	Image definition not found

RC	RC Name	RS	RS Name	Description
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available
592	RCERR_ASYNC_DM	nnnn	opid	Asynchronous operation started - product-specific asynchronous operation ID (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
596	RCERR_INTERNAL_DM	nnnn	psrc	Internal directory manager error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Image_Status_Query

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
image_name_array_length
image_name_array (1)
image_name_structure (2)
image_name_length
image_name

Note:

1. An array consists of zero or more of its components.
2. A structure consists of one set of its components.

Purpose

Use Image_Status_Query to determine whether virtual images are active (logged on or logged on disconnected) or inactive.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 18.

function_name
(string,18,char43) The API function name – in this case, 'Image_Status_Query'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
One of the following:

- (string,1-8,char42) The userid or image name.
- (string,1-64,char43) The name of a list of userids or images.
- (string,1,*) All active images.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

image_name_array_length
(int4) Length of *image_name_array*.

image_name_array
(array) An array consisting of zero or more instances of *image_name_structure*, as follows:

image_name_structure
(structure) A structure consisting of one set of *image_name_length* and *image_name*, as follows:

image_name_length
(int4) Length of *image_name*.

image_name
(string,1-8,char42) The name of an active image, from the set of images specified by *target_identifier*.

Usage Note

The asterisk (*) is not supported in the *target_identifier* field, and will result in a 100/16 reason code/return code if the SMAPI authorization policy is set to either of the following:

Authorization_Policy_ESMAuthlist
Authorization_Policy_ESMOnly

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
		12	RS_NOT_ACTIVE	Image not active
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
396	RCERR_INTERNAL	nnnn	psrc	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Image_Unlock_DM

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
device_address_length
device_address

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Image_Unlock_DM to unlock a virtual image's directory entry or a specific device in a virtual image's directory entry.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 15.

function_name
(string,15,char43) The API function name – in this case, 'Image_Unlock_DM'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The name of the image to be unlocked.

device_address_length
(int4) Length of *device_address*.

device_address
(string,0-4,char16) The virtual address of the device being unlocked.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Usage Notes

1. Use Image_Unlock_DM to unlock a locked image if you do not perform an Image_Replace_DM operation.
2. To unlock an entire image, omit the device address. To unlock a specific device, specify the device address.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		12	RS_AUTHERR_DM	Request not authorized by directory manager

RC	RC Name	RS	RS Name	Description
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
400	RCERR_IMAGEDEF	4	RS_NOT_FOUND	Image definition not defined
		24	RS_NOT_LOCKED	Image definition is not locked
404	RCERR_IMAGEDEVD	24	RS_NOT_LOCKED	Image device is not locked
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available
592	RCERR_ASYNC_DM	nnnn	opid	Asynchronous operation started - product-specific asynchronous operation ID (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
596	RCERR_INTERNAL_DM	nnnn	psrc	Internal directory manager error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Image_Volume_Add

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
image_device_number_length
image_device_number
image_vol_id_length
image_vol_id
system_config_name_length
system_config_name
system_config_type_length
system_config_type
parm_disk_owner_length
parm_disk_owner
parm_disk_number_length
parm_disk_number
parm_disk_password_length
parm_disk_password
alt_system_config_name_length
alt_system_config_name
alt_system_config_type_length
alt_system_config_type
alt_parm_disk_owner_length
alt_parm_disk_owner
alt_parm_disk_number_length
alt_parm_disk_number
alt_parm_disk_password_length
alt_parm_disk_password

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Image_Volume_Add to add a DASD volume to be used by virtual images to the z/VM system configuration file.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 16.

function_name
(string,16,char43) The API function name – in this case, 'Image_Volume_Add'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See “Client Authentication” on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See “Client Authentication” on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The name of the image to which a volume is being added.

image_device_number_length
(int4) Length of *image_device_number*.

image_device_number
(string,1-4,char16) The virtual device number of the device.

image_vol_id_length
(int4) Length of *image_vol_id*.

image_vol_id
(string,1-6,char42) The DASD volume label.

system_config_name_length
(int4) Length of *system_config_name*.

system_config_name
(string,0-8,char42) File name of system configuration file. The default is set by the "System_Config_File_Name =" statement in the DMSSICNF COPY file. (See the "Default SYSTEM CONFIG Link Values" section in “Configuring SMAPI” on page 30.)

system_config_type_length
(int4) Length of *system_config_type*.

system_config_type
(string,0-8,char42) File type of system configuration file. The default is set by the "System_Config_File_Type =" statement in the DMSSICNF COPY file. (See the "Default SYSTEM CONFIG Link Values" section in “Configuring SMAPI” on page 30.)

parm_disk_owner_length
(int4) Length of *parm_disk_owner*.

parm_disk_owner
(string,0-8,char42) Owner of the parm disk. The default is set by the "Parm_Disk_Owner =" statement in the DMSSICNF COPY file. (See the "Default SYSTEM CONFIG Link Values" section in “Configuring SMAPI” on page 30.)

parm_disk_number_length
(int4) Length of *parm_disk_number*.

parm_disk_number
(string,0-4,char16) Number of the parm disk as defined in the VSMWORK1 directory. (See Usage Note “4” on page 335.) The default is set by the "Parm_Disk_Number =" statement in the DMSSICNF COPY file. (See “Configuring SMAPI” on page 30.)

parm_disk_password_length
(int4) Length of *parm_disk_password*.

parm_disk_password
(string,0-8,charNB) Multiwrite password for the parm disk. The default is "," and should not be changed. Any value other the default is ignored. (See “Configuring SMAPI” on page 30.)

Note: The character "," is used to indicate no password. Therefore "," cannot be the password.

alt_system_config_name_length
(int4) Length of *alt_system_config_name*.

alt_system_config_name
(string,0-8,char42) No longer valid, maintained for backward compatibility. See Usage Note “1” on page 335.

alt_system_config_type_length
(int4) Length of *alt_system_config_type*.

alt_system_config_type
(string,0-8,char42) No longer valid, maintained for backward compatibility. See Usage Note “1” on page 335.

alt_parm_disk_owner_length
(int4) Length of *alt_parm_disk_owner*.

alt_parm_disk_owner
(string,0-8,char42) No longer valid, maintained for backward compatibility. See Usage Note “1” on page 335.

alt_parm_disk_number_length
(int4) Length of *alt_parm_disk_number*.

alt_parm_disk_number
(string,0-4,char16) No longer valid, maintained for backward compatibility. See Usage Note “1” on page 335.

alt_parm_disk_password_length
(int4) Length of *alt_parm_disk_password*.

alt_parm_disk_password
(string,0-8,charNB) No longer valid, maintained for backward compatibility. See Usage Note “1” on page 335.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Usage Notes

1. This API will only update the system configuration file on the parm disk specified, and *not* on the alternate parm disk. To maintain backward compatibility, however, the parameters for the alternate parm disk must still be specified. (The easiest way to do this is to simply specify the same values for the alternate parm disk parameters that were specified for the primary parm disk.)
2. If the system administrator has changed the default location of the system configuration file, or has renamed the file, then the input parameters must be used to specify the new file information.
3. Updates for the VSMWORK1 user in the VM directory are required to link and access the CP parm disks. A link option for PMAINT CF0 must be added. If the system administrator changed the default locations of the parm disks, the VSMWORK1 userid must be granted the appropriate authority and links to the new locations.

   The following links are provided in the user directory of VSMWORK1:

   ```
   .
   IDENTITY VSMWORK1 .....
   .
   .
   LINK PMAINT CF0 CF0 MD
   ```

4. If you want a different parm disk, add links to the VSMWORK1 user directory. For example:

   ```
   .
   USER VSMWORK1 .....
   .
   .
   LINK SMAPIC5 C00 FC00 MD
   ```

5. Your DASD volume must be initialized before you issue the Image_Volume_Add function. Note that ICKDSF initialization is not required if your DASD volume is one of the following
   - Enterprise Storage Server (ESCON, FICON, or FCP attached)
   - SCSI disks emulated as FBA DASD

   The above DASD volumes are initialized when they are set up.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid

RC	RC Name	RS	RS Name	Description
300	RCERR_VOLUME	0	RS_NONE	Image volume operation successful
		8	RS_DEV_NOT_FOUND	Device not found
		10	RS_DEV_NOT_AVAIL_TO _ATTACH	Device not available for attachment
		12	RS_DEV_NOT_VOLUME	Device not a volume
		14	RS_FREE_MODE_NOT_AVAIL	Free modes not available
		16	RS_DEV_NOT_ONLINE	Device vary online failed
		18	RS_VOLID_NOT_FOUND	Volume label not found in system configuration
		20	RS_VOLID_IN_USE	Volume label already in system configuration
		22	RS_PDISKS_SAME	Parm disks 1 and 2 are same
		24	RS_PARM_DISK_LINK_ERROR	Error linking parm disk (1 or 2)
		28	RS_PARM_DISK_NOT_RW	Parm disk (1 or 2) not RW
		32	RS_SYS_CONF_NOT_FOUND	System configuration not found on parm disk 1
		34	RS_SYS_CONF_BAD_DATA	System configuration has bad data
		36	RS_SYS_CONF_SYNTX_ERR	Syntax errors updating system configuration file
		38	RS_CPDISK_MODE_NOT _AVAIL	CP disk modes not available
		40	RS_PARM_DISK_FULL	Parm disk (1 or 2) is full
		42	RS_PDISK_ACC_NOT _ALLOWED	Parm disk (1 or 2) access not allowed
		44	RS_PDISK_PW_NOT _SUPPLIED	Parm disk (1 or 2) PW not supplied
		46	RS_PDISK_PW_INCORRECT	Parm disk (1 or 2) PW is incorrect
		48	RS_PDISK_NOT_IN_SERVER _DIRECTORY	Parm disk (1 or 2) is not in server's user directory
		50	RS_CP_RELEASE_ERROR	Error in release of CPRELEASE parm disk (1 or 2)
		52	RS_CP_ACCESS_ERROR	Error in access of CPACCESS parm disk (1 or 2)
396	RCERR_INTERNAL	nnnn	psrc	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)

RC	RC Name	RS	RS Name	Description
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Image_Volume_Delete

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
image_device_number_length
image_device_number
image_vol_id_length
image_vol_id
system_config_name_length
system_config_name
system_config_type_length
system_config_type
parm_disk_owner_length
parm_disk_owner
parm_disk_number_length
parm_disk_number
parm_disk_password_length
parm_disk_password
alt_system_config_name_length
alt_system_config_name
alt_system_config_type_length
alt_system_config_type
alt_parm_disk_owner_length
alt_parm_disk_owner
alt_parm_disk_number_length
alt_parm_disk_number
alt_parm_disk_password_length
alt_parm_disk_password

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Image_Volume_Delete to delete a DASD volume definition from the z/VM system configuration file.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 19.

function_name
(string,19,char43) The API function name – in this case, 'Image_Volume_Delete'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See “Client Authentication” on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See “Client Authentication” on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The name of the image from which a volume is being deleted.

image_device_number_length
(int4) Length of *image_device_number*.

image_device_number
(string,1-4,char16) The virtual device number of the device.

image_vol_id_length
(int4) Length of *image_vol_id*.

image_vol_id
(string,1-6,char42) The DASD volume label.

system_config_name_length
(int4) Length of *system_config_name*.

system_config_name
(string,0-8,char42) File name of system configuration file. The default is set by the "System_Config_File_Name =" statement in the DMSSICNF COPY file. (See the "Default SYSTEM CONFIG Link Values" section in “Configuring SMAPI” on page 30.)

system_config_type_length
(int4) Length of *system_config_type*.

system_config_type
(string,0-8,char42) File type of system configuration file. The default is set by the "System_Config_File_Type =" statement in the DMSSICNF COPY file. (See the "Default SYSTEM CONFIG Link Values" section in “Configuring SMAPI” on page 30.)

parm_disk_owner_length
(int4) Length of *parm_disk_owner*.

parm_disk_owner
(string,0-8,char42) Owner of the parm disk. The default is set by the "Parm_Disk_Owner =" statement in the DMSSICNF COPY file. (See the "Default SYSTEM CONFIG Link Values" section in “Configuring SMAPI” on page 30.)

parm_disk_number_length
(int4) Length of *parm_disk_number*.

parm_disk_number
(string,0-4,char16) Number of the parm disk as defined in the VSMWORK1 directory. (See Usage Note “4” on page 341.) The default is set by the "Parm_Disk_Number =" statement in the DMSSICNF COPY file. (See the "Default SYSTEM CONFIG Link Values" section in “Configuring SMAPI” on page 30.)

parm_disk_password_length
(int4) Length of *parm_disk_password*.

parm_disk_password
(string,0-8,charNB) Multiwrite password for the parm disk. The default is "," and should not be changed. Any value other the default is ignored. (See “Configuring SMAPI” on page 30.)

Note: The character "," is used to indicate no password. Therefore "," cannot be the password.

alt_system_config_name_length
(int4) Length of *alt_system_config_name*.

alt_system_config_name
(string,0-8,char42) No longer valid, maintained for backward compatibility. See Usage Note “1” on page 341.

alt_system_config_type_length
(int4) Length of *alt_system_config_type*.

alt_system_config_type
(string,0-8,char42) No longer valid, maintained for backward compatibility. See Usage Note “1” on page 341.

alt_parm_disk_owner_length
(int4) Length of *alt_parm_disk_owner*.

alt_parm_disk_owner
(string,0-8,char42) No longer valid, maintained for backward compatibility. See Usage Note “1” on page 341.

alt_parm_disk_number_length
(int4) Length of *alt_parm_disk_number*.

alt_parm_disk_number
(string,0-4,char16) No longer valid, maintained for backward compatibility. See Usage Note “1” on page 341.

alt_parm_disk_password_length
(int4) Length of *alt_parm_disk_password*.

alt_parm_disk_password
(string,0-8,charNB) No longer valid, maintained for backward compatibility. See Usage Note “1” on page 341.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Usage Notes

1. This API will only update the system configuration file on the parm disk specified, and *not* on the alternate parm disk. To maintain backward compatibility, however, the parameters for the alternate parm disk must still be specified. (The easiest way to do this is to simply specify the same values for the alternate parm disk parameters that were specified for the primary parm disk.)
2. If the system administrator has changed the default location of the system configuration file, or has renamed the file, then the input parameters must be used to specify the new file information.
3. Updates for the VSMWORK1 user in the VM directory are required to link and access the CP parm disks. A link option for PMAINT CF0 must be added. If the system administrator changed the default locations of the parm disks, the VSMWORK1 userid must be granted the appropriate authority and links to the new locations.

   The following links are provided in the user directory of VSMWORK1:

   ```
   .
   IDENTITY VSMWORK1 .....
   .
   .
   LINK PMAINT CF0 CF0 MD
   ```

4. If you want a different parm disk, add links to the VSMWORK1 user directory. For example:

   ```
   .
   USER VSMWORK1 .....
   .
   .
   LINK SMAPIC5 C00 FC00 MD
   ```

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
300	RCERR_VOLUME	0	RS_NONE	Image volume operation successful
		8	RS_DEV_NOT_FOUND	Device not found
		10	RS_DEV_NOT_AVAIL_TO _ATTACH	Device not available for attachment
		12	RS_DEV_NOT_VOLUME	Device not a volume

RC	RC Name	RS	RS Name	Description
		14	RS_FREE_MODE_NOT_AVAIL	Free modes not available
		16	RS_DEV_NOT_ONLINE	Device vary online failed
		18	RS_VOLID_NOT_FOUND	Volume label not found in system configuration
		20	RS_VOLID_IN_USE	Volume label already in system configuration
		22	RS_PDISKS_SAME	Parm disks 1 and 2 are same
		24	RS_PARM_DISK_LINK _ERROR	Error linking parm disk (1 or 2)
		28	RS_PARM_DISK_NOT_RW	Parm disk (1 or 2) not RW
		32	RS_SYS_CONF_NOT_FOUND	System configuration not found on parm disk 1
		34	RS_SYS_CONF_BAD_DATA	System configuration has bad data
		36	RS_SYS_CONF_SYNTX_ERR	Syntax errors updating system configuration file
		38	RS_CPDISK_MODE_NOT _AVAIL	CP disk modes not available
		40	RS_PARM_DISK_FULL	Parm disk (1 or 2) is full
		42	RS_PDISK_ACC_NOT _ALLOWED	Parm disk (1 or 2) access not allowed
		44	RS_PDISK_PW_NOT _SUPPLIED	Parm disk (1 or 2) PW not supplied
		46	RS_PDISK_PW_INCORRECT	Parm disk (1 or 2) PW is incorrect
		48	RS_PDISK_NOT_IN_SERVER _DIRECTORY	Parm disk (1 or 2) is not in server's user directory
		50	RS_CP_RELEASE_ERROR	Error in release of CPRELEASE parm disk (1 or 2)
		52	RS_CP_ACCESS_ERROR	Error in access of CPACCESS parm disk (1 or 2)
396	RCERR_INTERNAL	nnnn	psrc	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Image_Volume_Share

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
`img_vol_addr=`*value*
`share_enable=`*value*

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Image_Volume_Share to indicate a full-pack minidisk is to be shared by the users of many real and virtual systems.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 18.

function_name
(string,18,char43) The API function name – in this case, 'Image_Volume_Share'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See “Client Authentication” on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) This must match an entry in the authorization file that also contains the *authenticated_userid* and the *function_name* (Image_Volume_Share).

Note: The format for specifying the following additional input parameter is *parameter_name=value*, followed by a null (ASCIIZ) terminator. For more information, see "Key-value pair (KVP) input parameters" on page 51.

`img_vol_addr=`*value*
(string,1-4,char16) The real device number of the volume to be shared. This is a required parameter.

`share_enable=`*value*
(string,0-3,char26) One of the following:

ON
Turns on sharing of the specified full-pack minidisk.

OFF
Turns off sharing of the specified full-pack minidisk.

If unspecified, the default is ON.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Usage Notes

1. Do *not* use this API if you are sharing the full-pack minidisk between two or more systems that are members of the same SSI cluster.
2. This API must be executed on all systems that intend to share the minidisk.
3. Syntax errors (RC = 24 and RS = *pprr*) are only applicable to the common input parameters. Syntax checking is *not* performed on the additional input parameters for this API. If a valid parameter is specified multiple times, the last value specified for that parameter will be used. For more information, see "Key-value pair (KVP) input parameters" on page 51.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
8	RC_ERR	18	RS_VOLUME_NOT_FOUND	Volume does not exist
		19	RS_CP_OWNED	Volume is CP owned and cannot be used
		20	RS_CP_SYSTEM	Volume is CP system and cannot be used
		3002	RS_INVALID_PARAMETER	Invalid parameter name
		3003	RS_INVALID_OPERAND	Invalid parameter operand
		3004	RS_MISSING_PARAMETER	Required parameter missing
		3012	RS_VOLUME_NOT_FOUND	Volume does not exist
		3013	RS_VOLUME_OFFLINE	Volume is offline
		3014	RS_SHARE_UNSPPORTED	Volume does not support sharing
24	RCERR_SYNTAX	*pprr*	*pprr*	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
396	RCERR_INTERNAL	*nnnn*	*psrc*	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Image_Volume_Space_Define_DM

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
function_type
region_name_length
region_name
image_vol_id_length
image_vol_id
start_cylinder
size
group_name_length
group_name
device_type

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Image_Volume_Space_Define_DM to define space on a DASD volume to be allocated by the directory manager for use by virtual images.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 28.

function_name
(string,28,char43) The API function name – in this case, 'Image_Volume_Space_Define_DM'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) This must match an entry in the authorization file that also contains the *authenticated_userid* and the *function_name* (Image_Volume_Space_Define_DM).

function_type
(int1) One of the following numeric values, determining which additional parameters are required:

1
Define region as specified. *image_volid*, *region_name*, *start_cylinder*, and *size* are required for this function.

2
Define region as specified and add to group. *image_vol_id*, *region_name*, *start_cylinder*, *size*, and *group_name* are required for this function.

3
Define region as full volume. *image_vol_id* and *region_name* are required for this function.

4
Define region as full volume and add to group. *image_vol_id*, *region_name*, and *group_name* are required for this function.

5
Add existing region to group. (This function also defines the group if it does not already exist.) *region_name* and *Group* are required for this function.

Note: Refer to your directory manager documentation for more information on which function types are supported.

region_name_length
(int4) Length of *region_name*.

region_name
(string,0-8,char42) The region to be defined.

image_vol_id_length
(int4) Length of *image_vol_id*.

image_vol_id
(string,0-6,char42) The DASD volume label.

start_cylinder
(int4; range 0-2147483640) The starting point of the region. If the device is not mounted and attached to the system, then the *start_cylinder* parameter is required along with the *size* and *device_type* parameters.

size
(int4; range 1-2147483640) The number of cylinders to be used by region. If the device is not mounted and attached to the system, then the *Size* parameter is required along with the *start_cylinder* and *device_type* parameters.

group_name_length
(int4) Length of *group_name*.

group_name
(string,0-8,char42) The name of the group to which the region is assigned.

device_type
(int1) The device type designation. Valid values are:

0
Unspecified

1
3390

2
9336

3
3380

4
FB-512

If unspecified, the device must already be mounted and attached to the system, and the directory manager exit will query the device to determine the device type. If specified and the device is not mounted and attached to the system, the *start_cylinder* and *size* parameters must also be specified. If specified and the device is mounted and attached to the system, the query will be done to determine the *start_cylinder* and *size* parameters if these parameters are not specified.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Usage Notes

1. Depending on the value of *function_type*, the indicated optional parameters are required, while all other optional parameters are prohibited. Refer to your directory manager documentation for more information on which function types are supported.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
		8	RS_OFFLINE	Request successful; object directory offline
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		12	RS_AUTHERR_DM	Request not authorized by directory manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
420	RC_DASD_DM	4	RS_IVS_NAME_USED	Group, region, or volume name is already defined
		8	RS_IVS_NAME_NOT_USED	That group, region, or volume name is not defined.
		36	RS_IVS_NAME_NOT_DASD	The requested volume is offline or is not a DASD device
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available
592	RCERR_ASYNC_DM	nnnn	opid	Asynchronous operation started - product-specific asynchronous operation ID (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
596	RCERR_INTERNAL_DM	nnnn	psrc	Internal directory manager error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Image_Volume_Space_Define_Extended_DM

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
image_volume_space_define_names_length
`function_type=`*value*
`region_name=`*value*
`image_vol_id=`*value*
`start_cylinder=`*value*
`size=`*value*
`group_name=`*value*
`device_type=`*value*
`alloc_method=`*value*

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Image_Volume_Space_Define_Extended_DM to define space on a DASD volume to be allocated by the directory manager for use by virtual images.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 37.

function_name
(string,37,char43) The API function name – in this case, 'Image_Volume_Space_Define_Extended_DM'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) This must match an entry in the authorization file that also contains the *authenticated_userid* and the *function_name* (Image_Volume_Space_Define_Extended_DM).

image_volume_space_define_names_length
(int4) Length of the remaining set of *parameter_name=value* input parameters.

Note: The format for specifying the following additional input parameters is *parameter_name=value*, followed by a null (ASCIIZ) terminator. They may be specified in any order. For more information, see "Key-value pair (KVP) input parameters" on page 51. .

`function_type=`*value*
(string,1,char10) One of the following numeric values, determining which additional parameters are required:

1
Define region as specified. Additional parameters required for this function:

- `image_vol_id=`*value*
- `region_name=`*value*
- `start_cylinder=`*value*
- `size=`*value*

2
Define region as specified and add to group. Additional parameters required for this function:

- `image_vol_id=`*value*
- `region_name=`*value*
- `start_cylinder=`*value*
- `size=`*value*
- `group_name=`*value*

3
Define region as full volume. Additional parameters required for this function:

- `image_vol_id=`*value*
- `region_name=`*value*

4
Define region as full volume and add to group. Additional parameters required for this function:

- `image_vol_id=`*value*
- `region_name=`*value*
- `group_name=`*value*

5
Add existing region to group. (This function also defines the group if it does not already exist.) Additional parameters required for this function:

- `region_name=`*value*
- `group_name=`*value*

Note: Refer to your directory manager documentation for more information on which function types are supported.

`region_name=`*value*
(string,0-8,char42) The region to be defined.

`image_vol_id=`*value*
(string,0-6,char42) The DASD volume label.

`start_cylinder=`*value*
(string,0-10,char10) The starting point of the region. If the device is not mounted and attached to the system, then this parameter is required along with the `size=`*value* and `device_type=`*value* parameters.

`size=`*value*
(string,0-10,char10) The number of cylinders to be used by region. If the device is not mounted and attached to the system, then this parameter is required along with the `start_cylinder=`*value* and `device_type=`*value* parameters.

`group_name=`*value*
(string,0-8,char42) The name of the group to which the region is assigned.

`device_type=`*value*
(string,0-1,char10) The device type designation. Valid values are:

0
Unspecified

1
3390

2
9336

3
3380

4
FB-512

If unspecified, the device must already be mounted and attached to the system, and the directory manager exit will query the device to determine the device type.

If specified and the device is not mounted and attached to the system, the `start_cylinder=`*value* and `size=`*value* parameters must also be specified.

If specified and the device is mounted and attached to the system, the query will be done to determine the `start_cylinder=`*value* and `size=`*value* parameters if these parameters are not specified.

`alloc_method=`*value*
(string,0-1,char10) The allocation method. Valid values are:

0
Unspecified

1
Specifies the linear scanning method, in which the first region within a group is scanned for allocation until full, then the second region, and so on until the last region is reached.

2
Specifies the rotating scanning method, in which the first region within a group is scanned for the first allocation, then the second region for the second allocation, and so on with each new allocation starting at the next region.

Values 1 and 2 are used when a new group is created. For an existing group, value 0 should be specified.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Usage Notes

1. Depending on the `function_type=`*value* parameter, some further input parameters will be required, while others will be prohibited. Refer to your directory manager documentation for more information on which function types are supported.
2. Syntax errors (RC = 24 and RS = *pprr*) are only applicable to the common input parameters. Syntax checking is *not* performed on the additional input parameters for this API. If a valid parameter is specified multiple times, the last value specified for that parameter will be used. For more information, see “Key-value pair (KVP) input parameters” on page 51.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
8	RC_ERR	3002	RS_INVALID_PARAMETER	Invalid parameter name
		3003	RS_INVALID_OPERAND	Invalid parameter operand
		3004	RS_MISSING_PARAMETER	Required parameter missing
24	RCERR_SYNTAX	*pprr*	*pprr*	Syntax error in function parameter

RC	RC Name	RS	RS Name	Description
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		12	RS_AUTHERR_DM	Request not authorized by directory manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
420	RC_DASD_DM	4	RS_IVS_NAME_USED	Group, region, or volume name is already defined
		8	RS_IVS_NAME_NOT_USED	Group, region, or volume name is not defined.
		36	RS_IVS_NAME_NOT_DASD	The requested volume is offline or is not a DASD device
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available
592	RCERR_ASYNC_DM	*nnnn*	*opid*	Asynchronous operation started - product-specific asynchronous operation ID (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
596	RCERR_INTERNAL_DM	*nnnn*	*psrc*	Internal directory manager error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory

RC	RC Name	RS	RS Name	Description
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Image_Volume_Space_Query_DM

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
query_type
entry_type
entry_names_length
entry_names

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
record_array_length
record_array (1)
    record_structure (2)
        record_length
        record

Note:

1. An array consists of zero or more of its components.
2. A structure consists of one set of its components.

Purpose

Use Image_Volume_Space_Query_DM to query how space on a DASD volume is allocated by the directory manager.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 27.

function_name
(string,27,char43) The API function name – in this case, 'Image_Volume_Space_Query_DM'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) This must match an entry in the authorization file that also contains the *authenticated_userid* and the *function_name* (Image_Volume_Space_Query_DM).

query_type
(int1) One of the following values:

1
DEFINITION – Query volume definition for the specified image device.

2
FREE – Query amount of free space available on the specified image device.

3
USED – Query amount of space used on the specified image device.

entry_type
(int1) One of the following values:

1
VOLUME – Query specified volume.

2
REGION – Query specified region.

3
GROUP – Query specified group.

entry_names_length
(int4) Length of *entry_names*.

entry_names
One of the following:

- (string,0-255,char42 plus blank) Names of groups, regions or volumes to be queried, separated by blanks.
- (string,1,*) Specifies all areas of the requested type.

If unspecified, an asterisk (*) is assumed, to specify all areas of the requested type.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

record_array_length
(int4) Length of *record_array*.

record_array
(array) An array consisting of zero or more instances of *record_structure*, as follows:

record_structure
(structure) A structure consisting of one set of *record_length* and *record*, as follows:

record_length
(int4) Length of *record*.

record
(string,1-*,charNA) The record containing the queried information. See Usage Note “1” on page 358.

Usage Notes

1. If the call completes successfully, each *record* will contain the following information, depending of the type of query.
   - For VOLUME DEFINITION:

     volid
     devtype
     size
     region_names one or more names separated by blanks
   - For REGION DEFINITION:

     region_name
     volid
     start_cyl
     devtype
     size
     group_names (blank, or one or more names separated by blanks)
   - For GROUP DEFINITION:

     group_name
     region_names (blank, or one or more names separated by blanks)
   - For USED space query:

     volid

devtype
start
size
owner
vaddr
ssinode
group_name (or * for region or volid query)
region_name (or * for volid)

- For FREE space query:

  volid
  devtype
  start
  size
  group_name (or * for region or volid query)
  region_name (or * for volid)

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		12	RS_AUTHERR_DM	Request not authorized by directory manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
420	RC_DASD_DM	8	RS_IVS_NAME_NOT_USED	That group, region, or volume name is not defined.
		12	RS_IVS_NAME_NOT _INCLUDED	That region name is not included in the group.
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available
592	RCERR_ASYNC_DM	nnnn	opid	Asynchronous operation started - product-specific asynchronous operation ID (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
596	RCERR_INTERNAL_DM	nnnn	psrc	Internal directory manager error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)

RC	RC Name	RS	RS Name	Description
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Image_Volume_Space_Query_Extended_DM

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
image_volume_space_query_names_length
`query_type=`*value*
`entry_type=`*value*
`entry_names=`*value*

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
record_array (1)
record

Note:

1. An array consists of zero or more of its components.

Purpose

Use Image_Volume_Space_Query_Extended_DM to query how space on a DASD volume is allocated by the directory manager.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 36.

function_name
(string,36,char43) The API function name – in this case, 'Image_Volume_Space_Query_Extended_DM'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) This must match an entry in the authorization file that also contains the *authenticated_userid* and the *function_name* (Image_Volume_Space_Query_Extended_DM).

image_volume_space_query_names_length
(int4) Length of the remaining set of *parameter_name=value* input parameters.

Note: The format for specifying the following additional input parameters is *parameter_name=value*, followed by a null (ASCIIZ) terminator. They may be specified in any order. For more information, see "Key-value pair (KVP) input parameters" on page 51.

query_type=*value*
(string,1,char10) One of the following values:

1
DEFINITION – Query volume definition for the specified image device.

2
FREE – Query amount of free space available on the specified image device.

3
USED – Query amount of space used on the specified image device.

This is a required parameter.

entry_type=*value*
(string,1,char10) One of the following values:

1
VOLUME – Query specified volume.

2
REGION – Query specified region.

3
GROUP – Query specified group.

This is a required parameter.

entry_names=*value*
(string,0-255,char42 plus blank) Names of groups, regions or volumes to be queried, separated by blanks. An asterisk (*) specifies all areas of the requested type. If unspecified, * is the default.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

record_array
(array) An array consisting of zero or more instances of *record_structure*, with each structure terminated by a null (ASCIIZ) character, as follows:

record
(string,1-*,charNA) A record containing the queried information. See Usage Note "1" on page 363.

Usage Notes

1. If the call completes successfully, each record will contain the following information, depending of the type of query.
   - For `query_type=DEFINITION`, `entry_type=VOLUME`:
     - *volid*
     - *devtype*
     - *size*
     - *region_names* (one or more names separated by blanks)
   - For `query_type=DEFINITION`, `entry_type=REGION`:
     - *region_name*
     - *volid*
     - *start_cyl*
     - *devtype*
     - *size*
     - *group_names* (blank, or one or more names separated by blanks)
   - For `query_type=DEFINITION`, `entry_type=GROUP`:
     - *group_name*
     - *alloc_method*
     - *region_names* (blank, or one or more names separated by blanks)
   - For `query_type=USED`:
     - *volid*
     - *devtype*
     - *start*
     - *size*
     - *owner*
     - *vaddr*

  - *group_name* (or * if `entry_type`=VOLUME or `entry_type`=REGION)
  - *region_name* (or * if `entry_type`=VOLUME)
  - *ssinode*
- For `query_type`=FREE:
  - *volid*
  - *devtype*
  - *start*
  - *size*
  - *group_name* (or * if `entry_type`=VOLUME or `entry_type`=REGION)
  - *region_name* (or * if `entry_type`=VOLUME)

2. Syntax errors (RC = 24 and RS = *pprr*) are only applicable to the common input parameters. Syntax checking is *not* performed on the additional input parameters for this API. If a valid parameter is specified multiple times, the last value specified for that parameter will be used. For more information, see “Key-value pair (KVP) input parameters” on page 51.
3. If the DASD is an unknown size, the information returned for the device type is "*deviceType*-?" (for example, "3390-?" or "9336-?").

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
8	RC_ERR	3002	RS_INVALID_PARAMETER	Invalid parameter name
		3003	RS_INVALID_OPERAND	Invalid parameter operand
		3004	RS_MISSING_PARAMETER	Required parameter missing
24	RCERR_SYNTAX	*pprr*	*pprr*	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		12	RS_AUTHERR_DM	Request not authorized by directory manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
420	RC_DASD_DM	8	RS_IVS_NAME_NOT_USED	Group, region, or volume name is not defined.
		12	RS_IVS_NAME_NOT_INCLUDED	Region name is not included in the group

RC	RC Name	RS	RS Name	Description
		36	RS_IVS_NAME_NOT_DASD	The requested volume is offline or is not a DASD device
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available
592	RCERR_ASYNC_DM	*nnnn*	*opid*	Asynchronous operation started - product-specific asynchronous operation ID (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
596	RCERR_INTERNAL_DM	*nnnn*	*psrc*	Internal directory manager error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Image_Volume_Space_Remove_DM

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
function_type
region_name_length
region_name
image_vol_id_length
image_vol_id
group_name_length
group_name

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Image_Volume_Space_Remove_DM to remove the directory manager's space allocations from a DASD volume.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 28.

function_name
(string,28,char43) The API function name – in this case, 'Image_Volume_Space_Remove_DM'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).

- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) This must match an entry in the authorization file that also contains the *authenticated_userid* and the *function_name* (Image_Volume_Space_Remove_DM).

function_type
(int1) One of the following numeric values, determining which additional parameters are required:

1
Remove named region. *RegionName* is required for this function.

2
Remove named region from group. *RegionName* and *GroupName* are required for this function.

3
Remove named region from all groups. *RegionName* is required for this function.

4
Remove all regions from specific volume. *ImageVolid* is required for this function.

5
Remove all regions from specific volume and group. *ImageVolid* and *GroupName* are required for this function.

6
Remove all regions from specific volume and all groups. *ImageVolid* is required for this function.

7
Remove entire group. *GroupName* is required for this function.

region_name_length
(int4) Length of *region_name*.

region_name
(string,0-8,char42) The region to be defined.

image_vol_id_length
(int4) Length of *image_vol_id*.

image_vol_id
(string,0-6,char42) The DASD volume label.

group_name_length
(int4) Length of *group_name*.

group_name
(string,0-8,char42) The name of the group to which the region is assigned.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Usage Notes

1. Depending on the value of *FunctionType*, the indicated optional parameters are required, while all other optional parameters are prohibited.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
		8	RS_OFFLINE	Request successful; object directory offline
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		12	RS_AUTHERR_DM	Request not authorized by directory manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
420	RC_DASD_DM	8	RS_IVS_NAME_NOT_USED	Group, region, or volume name is not defined
		12	RS_IVS_NAME_NOT _INCLUDED	Region name is not included in the group
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available
592	RCERR_ASYNC_DM	nnnn	opid	Asynchronous operation started - product-specific asynchronous operation ID (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)

RC	RC Name	RS	RS Name	Description
596	RCERR_INTERNAL_DM	nnnn	psrc	Internal directory manager error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Metadata_Delete

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
metadata_name_list

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Metadata_Delete to delete metadata values associated with a textual identifier (typically a directory entry name).

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 15.

function_name
(string,15,char43) The API function name – in this case, 'Metadata_Delete'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See “Client Authentication” on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) A textual identifier (typically a directory entry name).

metadata_name_list
(string,1-maxlength,charNB) A blank-delimited list of metadata names, followed by a null (ASCIIZ) terminator. Note that these metadata names are case sensitive.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
8	RC_ERR	3002	RS_INVALID_PARAMETER	Invalid parameter name
24	RCERR_SYNTAX	*pprr*	*pprr*	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		12	RS_AUTHERR_DM	Request not authorized by directory manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid

RC	RC Name	RS	RS Name	Description
396	RCERR_INTERNAL	*nnnn*	*psrc*	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range
		68	RS_DATABASE	Unable to access LOHCOST server

Metadata_Get

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
metadata_name_list

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
metadata_entry_array_length
metadata_entry_array (1)
metadata_entry_structure (2)
metadata_entry_structure_length
metadata_entry_name_length
metadata_entry_name
metadata_length
metadata

Note:

1. An array consists of zero or more of its components.
2. A structure consists of one set of its components.

Purpose

Use Metadata_Get to obtain metadata values associated with a textual identifier (typically a directory entry name).

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 12.

function_name
(string,12,char43) The API function name – in this case, 'Metadata_Get'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) A textual identifier (typically a directory entry name).

metadata_name_list
(string,1-maxlength,charNB) A blank-delimited list of metadata names, followed by a null (ASCIIZ) terminator. Note that these metadata names are case sensitive.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

metadata_entry_array_length
(int4) Length of *metadata_entry_array*.

metadata_entry_array
(array) An array consisting of zero or more instances of *metadata_entry_structure*, as follows:

metadata_entry_structure
(structure) A structure consisting of one set of the following parameters:

metadata_entry_structure_length
(int4) The combined length of the remaining parameters in *metadata_entry_structure* (not including this parameter).

metadata_entry_name_length
(int4) Length of *metadata_entry_name*.

metadata_entry_name
(string,1-1024,charNB) The metadata entry name.

metadata_length
(int4) Length of *metadata*.

metadata
(string,1-maxlength,charNA) The metadata.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
8	RC_ERR	3002	RS_INVALID_PARAMETER	Invalid parameter name
24	RCERR_SYNTAX	*pprr*	*pprr*	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		12	RS_AUTHERR_DM	Request not authorized by directory manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
396	RCERR_INTERNAL	*nnnn*	*psrc*	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range
		68	RS_DATABASE	Unable to access LOHCOST server

Metadata_Set

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
metadata_entry_array_length
metadata_entry_array (1)
metadata_entry_structure (2)
metadata_entry_structure_length
metadata_entry_name_length
metadata_entry_name
metadata_length
metadata

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Note:

1. An array consists of zero or more of its components.
2. A structure consists of one set of its components.

Purpose

Use Metadata_Set to set metadata values associated with a textual identifier (typically a directory entry name).

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 12.

function_name
(string,12,char43) The API function name – in this case, 'Metadata_Set'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) A textual identifier (typically a directory entry name).

metadata_entry_array_length
(int4) Length of *metadata_entry_array*.

metadata_entry_array
(array) An array consisting of zero or more instances of *metadata_entry_structure*, as follows:

metadata_entry_structure
(structure) A structure consisting of one set of the following parameters:

metadata_entry_structure_length
(int4) The combined length of the remaining parameters in *metadata_entry_structure* (not including this parameter).

metadata_entry_name_length
(int4) Length of *metadata_entry_name*.

metadata_entry_name
(string,1-1024,charNB) The metadata entry name.

metadata_length
(int4) Length of *metadata*.

metadata
(string,1-maxlength,charNA) The metadata.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
8	RC_ERR	3002	RS_INVALID_PARAMETER	Invalid parameter name
24	RCERR_SYNTAX	13	RS_LONG	Metadata entry name value length exceeds allowable length (1024)
		pprr	*pprr*	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		12	RS_AUTHERR_DM	Request not authorized by directory manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
396	RCERR_INTERNAL	*nnnn*	*psrc*	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range
		68	RS_DATABASE	Unable to access LOHCOST server

Metadata_Space_Query

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
`searchkey=value`

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
output_data

Note:

1. An array consists of zero or more of its components.

Purpose

Use Metadata_Space_Query to obtain information about metadata space used and available.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 18.

function_name
(string,26,char43) The API function name – in this case, 'Metadata_Space_Query'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) This must match an entry in the authorization file that also contains the *authenticated_userid* and the *function_name* (Configuration_Read).

searchkey=value
Value is a null terminated string, including wildcards. See "Event_Subscribe" on page 111, which describes the metadata names of interest.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

output_data
A set of null terminated strings describing the metadata space specified by the searchkey.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful – at least one null terminated string is returned. If no searchkey is specified, the string contains the total metadata space followed by the amount of available metadata space, both in 1K blocks. If a searchkey is specified, there are a set of null terminated strings specifying the length in bytes followed by the qualifying metadata name for each qualifying name.
4	RC_WNG	4	RS_NOT_FOUND	No qualifying metadata was found.
8	RC_ERR	8	RS_NOT_EXIST	No metadata exists.
		11	RS_UNSUPPORTED	Unsupported parameter.
		12	RS_NOT_ACTIVE	The metadata server is inactive.
		3004	RS_MISSING_PARAMETER	Required parameter missing

Name_List_Add

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
name_length
name

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Name_List_Add to add a name to a list in the name list file. If the list that is specified in *target_identifier* does not exist, a new list will be created.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 13.

function_name
(string,13,char43) The API function name – in this case, 'Name_List_Add'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See “Client Authentication” on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-64,char43) The name of the list that is being updated.

name_length
(int4) Length of *name*.

name
One of the following:

- (string,1-8,char42) A userid.
- (string,1-64,char43) A function name.

This is the name to be added to the list specified in *target_identifier*. Mixed case names are permitted as input but case is ignored when the name is processed. (All names are converted to upper case.)

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
		12	RS_NEW_LIST	Request successful; new list created
		36	RS_NAME_IN_LIST	Name is already in list
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
28	RCERR_FILE_NOT_FOUND	0	RS_NONE	Namelist file not found
36	RCERR_FILE_CANNOT _BE_UPDATED	0	RS_NONE	Namelist file cannot be updated

RC	RC Name	RS	RS Name	Description
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
396	RCERR_INTERNAL	nnnn	psrc	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Name_List_Destroy

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Name_List_Destroy to destroy a list from the name list file.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 17.

function_name
(string,17,char43) The API function name – in this case, 'Name_List_Destroy'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See “Client Authentication” on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-64,char43) The name of the list being destroyed.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
28	RCERR_FILE_NOT_FOUND	0	RS_NONE	Namelist file not found
36	RCERR_FILE_CANNOT _BE_UPDATED	0	RS_NONE	Namelist file cannot be updated
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
200	RCERR_IMAGEOP	24	RS_LIST_NOT_FOUND	List not found
396	RCERR_INTERNAL	nnnn	psrc	Internal system error - product-specific return code (See "Internal Return Codes (RC = 396, 592, or 596)" on page 834)

RC	RC Name	RS	RS Name	Description
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Name_List_Query

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
name_array_length
name_array (1)
name_structure (2)
name_length
name

Note:

1. An array consists of zero or more of its components.
2. A structure consists of one set of its components.

Purpose

Use Name_List_Query to query the names that are in a list in the name list file.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 15.

function_name
(string,15,char43) The API function name – in this case, 'Name_List_Query'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
One of the following:

- (string,1-64,char43) The name of the list being queried.
- (string,1,*) All existing lists.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

name_array_length
(int4) Length of *name_array*.

name_array
(array) An array consisting of zero or more instances of *name_structure*, as follows:

name_structure
(structure) A structure consisting of one set of *name_length* and *name*, as follows:

name_length
(int4) Length of *name*.

name
One of the following:

- (string,1-8,char42) Images (userids).
- (string,1-64,char43) Function names.

Usage Note

The asterisk (*) is not supported in the *target_identifier* field, and will result in a 100/16 reason code/return code if the SMAPI authorization policy is set to either of the following:

Authorization_Policy_ESMAuthlist
Authorization_Policy_ESMOnly

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
28	RCERR_FILE_NOT_FOUND	0	RS_NONE	Namelist file not found
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
200	RCERR_IMAGEOP	24	RS_LIST_NOT_FOUND	List not found
396	RCERR_INTERNAL	nnnn	psrc	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Name_List_Remove

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
name_length
name

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Name_List_Remove to delete a name from a list in the name list file. If there are no names remaining in the list, the list is also deleted.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 16.

function_name
(string,16,char43) The API function name – in this case, 'Name_List_Remove'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-64,char43) The name of the list that is being updated.

name_length
(int4) Length of *name*.

name
One of the following:

- (string,1-8,char42) A userid.
- (string,1-64,char43) A function name or list.

This is the name to be removed from the list specified in *target_identifier*.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
		16	RS_LIST_DESTROYED	Request successful; no more entries, list destroyed
		32	RS_NOT_IN_LIST	Name was not in list
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
28	RCERR_FILE_NOT_FOUND	0	RS_NONE	Namelist file not found
36	RCERR_FILE_CANNOT _BE_UPDATED	0	RS_NONE	Namelist file cannot be updated

RC	RC Name	RS	RS Name	Description
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
	RCERR_AUTH	16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
200	RCERR_IMAGEOP	24	RS_LIST_NOT_FOUND	List not found
396	RCERR_INTERNAL	nnnn	psrc	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Network_IP_Interface_Create

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
`tcpip_stack=`*value*
`interface_id=`*value*
`permanent=`*value*
`primary_ipv4=`*value*
`primary_ipv6=`*value*
`interface=`*value*
`buffers=`*value*
`cpu=`*value*
`transport_type=`*value*
`mtu=`*value*
`noforward=`*value*
`pathmtu=`*value*
`p2p=`*value*
`port_name=`*value*
`port_number=`*value*
`vlan=`*value*

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
error_data_length (error only)
error_data (error only)

Purpose

Use Network_IP_Interface_Create to create the initial network interface configuration for the z/VM TCP/IP stack.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 27.

function_name
(string,13,char43) The API function name – in this case, 'Network_IP_Interface_Create'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) This must match an entry in the authorization file that also contains the *authenticated_userid* and the *function_name* (Network_IP_Interface_Create).

Note: The format for specifying the following additional input parameters is *parameter_name=value*, followed by a null (ASCIIZ) terminator. For more information, see "Key-value pair (KVP) input parameters" on page 51.

`tcpip_stack=`*value*
(string,1-8,char42) The TCP/IP stack to which the new interface applies. This input parameter is required.

`interface_id=`*value*
(string,1-16,charNB) The identifier of the new interface. Note that this value cannot begin with a dash (–), end with a colon (:), or contain a semicolon (;). This input parameter is required.

`permanent=`*value*
(string,0-3,char26) One of the following:

YES
The added interface will be permanent.

NO
The added interface will be temporary (created only for the current session). This is the default.

`primary_ipv4=`*value*
(string,7-18,char10 plus '.' and '/') The primary IPv4 address. The address should be specified in dot-decimal notation, with a mask length separated by a slash delimiter ('/'). (For example:

192.168.0.9/24.) The mask length is optional, and its value should be in the range 1-30. Specifying a port number here (:port) is not allowed. One of the IP input parameters (primary_ipv4=,primary_ipv6=) is required. Only one may be specified.

primary_ipv6=*value*
(string,3-43,char16 plus ':' and '/') The primary IPv6 address. The address should be specified by 8 groups of 16-bit hexadecimal values, separated by colons (:), with a prefix length separated by a slash delimiter ('/'). (For example: 1080:0:0:0:AB32:800:FF83:10/64.) The prefix length is optional, and its value should be in the range 1-128. One group of consecutive zeroes within an address may be replaced by a double colon ('::'). IPv4-embedded IPv6 addresses are not allowed. One of the IP input parameters (primary_ipv4=, primary_ipv6=) is required. Only one may be specified.

interface=*value*
(string,4-37,char) Type of interface to be created. Only one of the following types can be specified per value, and only one interface can be created per API call. The options for each type are blank-delimited, and are required unless otherwise stated. This input parameter is required.

ETH *rdevno ipv4router ipv6router*
Defines an LCS, QDIO, or EQDIO Ethernet interface.

rdevno
(string,1-4,char16) The real device address.

ipv4router
(string,0-3,char26) Optional, the router interface type for IPv4. Possible values are: PRI, SEC, NON.

ipv6router
(string,0-7,char26) Optional, the router interface type for IPv6. Possible values are: IPV6PRI, IPV6SEC, IPV6NON.

HS *rdevno*
Defines a real HyperSocket connection.

rdevno
(string,1-4,char16) The real device address.

IUCV *userid*
Defines an IUCV interface.

userid
(string,1-8,char42) The communication partner userid.

CTC *rdevno*
Defines a real channel-to-channel interface.

rdevno
(string,1-4,char16) The real device address.

VETH *vdevno ownerid lanname*
Defines a virtual QDIO Ethernet connection to the named guest LAN or virtual switch.

rdevno
(string,1-4,char16) The real device address.

ownerid
(string,1-8,char42) The owner of the LAN/VSWITCH. If a VSWITCH name is specified, the *ownerid* must be SYSTEM.

lanname
(string,1-8,char42) The LAN or VSWITCH name.

If no guest LAN or VSWITCH exists with the specified *ownerid*/*lanname* combination, a QDIO guest LAN will be created. The *ownerid* and *lanname* are limited to a maximum of 8 characters each.

VCTC *vdevno1 userid vdevno2*
Defines a virtual channel-to-channel interface. A virtual CTC is defined and coupled to the specified user's virtual device.

vdevno1
vdevno2
(string,1-4,char16) The virtual device addresses.

userid
(string,1-8,char42) The owner of *vdevno1*.

VHS *vdevno ownerid lanname*
Defines a virtual HyperSocket connection. A HyperSockets guest LAN will be created.

vdevno
(string,1-4,char16) The virtual device address.

ownerid
(string,1-8,char42) The LAN owner.

lanname
(string,1-8,char42) The LAN name.

If no guest LAN exists with the specified *ownerid* and *lanname* combination, a HiperSockets guest LAN will be created. The *ownerid* and *lanname* are limited to a maximum of 8 characters each.

buffers=*value*
(string,5-11,char10 plus -; range 2048-65535) The value defines the number of buffers to be allocated for an EQDIO connection. The value can be one of the following:

- A single value between 2048 and 65535. This specifies the initial/maximum number of buffers.
- A range indicated by two numbers separated by a dash (-). This specifies the initial number of buffers along with the maximum number of buffers that may be used. The range of numbers must be between 2048 and 65535. The minimum and maximum values cannot be the same value.

  Example: 2048-4096

Note: The actual minimum and maximum value may be different due to rounding by the device. For more information, see the documentation of the BUFFERS operand for the IFCONFIG command in *z/VM: TCP/IP Planning and Customization*.

cpu=*value*
(string,0-1,char10) Specifies the virtual processor to be used to run the device driver for the interface. The value must be an integer in the range 0-6. The default is 0.

transport_type=*value*
(string,2-8,char26) One of the following:

IP
The transport for the link is IP.

ETHERNET
The transport for the link is Ethernet.

This parameter can be specified only for real or virtual QDIO Ethernet devices.

mtu=*value*
(string,0-5,char10) Defines the maximum transmission unit (MTU) size that is to be used on the interface. To determine the recommended MTU size, refer to the hardware documentation associated with the device. If you specify 0 or omit this option, the TCP/IP stack will select an intelligent default.

noforward=*value*
(string,0-3,char26) One of the following:

ON
Specifies that packets received on this link are *not* to be forwarded to another host (that is, packets destined for a foreign host are to be discarded) and that packets transmitted on this link must originate from the local host. Packets received for another host on this link are to be dropped, as are packets received for another host on any link and forwarded through this one.

OFF
Specifies that packets received or transmitted on the link can be forwarded to another host. This is the default.

pathmtu=*value*
(string,0-3,char26) One of the following:

YES
Specifies that path MTU discovery will be used on IPv4 routes for a given link.

NO
Specifies that path MTU discovery will *not* be used on IPv4 routes for a given link.

YES is the default when the PATHMTU operand is specified on the ASSORTEDPARMS statement in the TCP/IP configuration file. Otherwise, NO is the default.

Note that these operands have no effect on IPv6 routes. Path MTU discovery is always enabled for IPv6 and cannot be disabled.

p2p=*value*
(string,7-15,char10 plus '.') Defines the IPv4 address associated with the other end of a point-to-point interface. The value should be specified in dot-decimal notation. This is a required parameter for IUCV and CTC interfaces.

port_name=*value*
(string,1-8,charNB) Specifies the queued direct I/O (QDIO) port name when it is being defined for use by this interface.

port_number=*value*
(string,1-2,char10) Specifies the physical port or adapter number on the device, when it is being defined to be used by this interface. This number depends on the device type, as follows:

- For channel-to-channel (CTC) connections, specify 0 or 1.
- For an IBM Open Systems Adapter-Express operating in LCS mode, specify 0 or 1.
- For an IBM Open Systems Adapter-Express operating in QDIO mode, specify a decimal number in the range 0-15. The value of the port number depends on how many ports the OSA-Express hardware feature supports. If the port_number=*value* is not specified, it will default to port 0.

Do *not* specify a port number for other devices.

vlan=*value*
(string,1-9,char10 plus blank) Specifies the identifier for a virtual local area network (VLAN). The format of the value is either *ipv4vlan* or *ipv4vlan ipv6vlan* (blank delimited), as follows:

- For a QDIO Ethernet device, *ipv4vlan* specifies the IPv4 VLAN ID. You can optionally specify a separate VLAN ID for your IPv6 network by using the second subvalue, *ipv6vlan*. If *ipv6vlan* is not specified, *ipv4vlan* will also be used for the IPv6 network.
- For a HiperSockets device, only one VLAN ID, *ipv4vlan*, may be specified.

Note that this input parameter can be specified only for the above devices.

Both *ipv4vlan* and *ipv6vlan* must be numbers in the range 1-4094.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

The following parameters will be returned only when the IFCONFIG command returns an error or warning (RC/RS=4/4, 8/12, or 8/16):

error_data_length
(int4) Length of *error_data*.

error_data
(string) The output of the IFCONFIG command.

Usage Notes

1. This API issues the IFCONFIG command, which makes use of the NETSTAT and OBEYFILE commands to facilitate its operations. The SMAPI worker server IDs must therefore be included in the OBEY list for all TCPIP stacks they manage. Additionally, the SMAPI worker servers need to have links to the 198 TCPIP disk.
2. Syntax errors (RC = 24 and RS = *pprr*) are only applicable to the common input parameters. Syntax checking is *not* performed on the additional input parameters for this API. If a valid parameter is specified multiple times, the last value specified for that parameter will be used. For more information, see “Key-value pair (KVP) input parameters” on page 51.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
4	RC_WNG	4	RS_IFCONFIG_WARNING	The command completed successfully, but a warning condition was detected on IFCONFIG command
8	RC_ERR	8	RS_DEV_NOT_FOUND	Device does not exist
		12	RS_IFCONFIG_ERROR	An error was encountered on IFCONFIG command
		16	RS_IFCONFIG_UNEXPECTED	An unexpected condition was encountered on IFCONFIG command
		3002	RS_INVALID_PARAMETER	Invalid parameter name
		3003	RS_INVALID_OPERAND	Invalid parameter operand
		3004	RS_MISSING_PARAMETER	Required parameter missing
24	RCERR_SYNTAX	*pprr*	*pprr*	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid

RC	RC Name	RS	RS Name	Description
396	RCERR_INTERNAL	*nnnn*	*psrc*	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Network_IP_Interface_Modify

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
`tcpip_stack=`*value*
`interface_id=`*value*
`permanent=`*value*
`delete_ip=`*value*
`add_ip=`*value*
`change_mask=`*value*
`change_mtu=`*value*
`change_p2p=`*value*

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
error_data_length (error only)
error_data (error only)

Purpose

Use Network_IP_Interface_Modify to change the configuration of the existing network interface.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 27.

function_name
(string,13,char43) The API function name – in this case, 'Network_IP_Interface_Modify.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) This must match an entry in the authorization file that also contains the *authenticated_userid* and the *function_name* (Network_IP_Interface_Modify).

Note: The format for specifying the following additional input parameters is *parameter_name=value*, followed by a null (ASCIIZ) terminator. For more information, see "Key-value pair (KVP) input parameters" on page 51.

`tcpip_stack=`*value*
(string,1-8,char42) The TCP/IP stack to which the interface change applies. This input parameter is required.

`interface_id=`*value*
(string,1-16,charNB) The identifier of the interface to be modified. Note that this value cannot begin with a dash (–), end with a colon (:), or contain a semicolon (;). This input parameter is required.

`permanent=`*value*
(string,0-3,char26) One of the following:

YES
The changes to the interface configuration will be permanent.

NO
The changes to the interface configuration will be temporary (created only for the current session). This is the default.

You must specify *exactly one* of the next five modify input parameters (delete_ip=, add_ip=, change_mask=, change_mtu=. or change_p2p=).

`delete_ip=`*value*
(string,3-43,char16 plus ':', '.' and '/') The IPv4 or IPv6 address to be deleted.

The IPv4 address should be specified in dot-decimal notation with a mask length separated by a slash delimiter ('/'). (For example: 192.168.0.9/24.) The mask length is optional and its value should be in the range 1-30.

An IPv6 address should be specified by 8 groups of 16-bit hexadecimal values separated by colons (:) with a prefix length separated by a slash delimiter ('/'). (For example: 1080:0:0:0:AB32:800:FF83:10/64.) The prefix length is optional and its value should be in the range 1-128. One group of consecutive zeroes within an address may be replaced by a double colon ('::'). IPv4-embedded IPv6 addresses are not allowed.

add_ip=*value*
(string,3-43,char16 plus ':', '.' and '/') The IPv4 or IPv6 address to be added.

The IPv4 address should be specified in dot-decimal notation with a mask length separated by a slash delimiter ('/'). (For example: 192.168.0.9/24.) The mask length is optional and its value should be in the range 1-30.

An IPv6 address should be specified by 8 groups of 16-bit hexadecimal values separated by colons (:) with a prefix length separated by a slash delimiter ('/'). (For example: 1080:0:0:0:AB32:800:FF83:10/64.) The prefix length is optional and its value should be in the range 1-128. One group of consecutive zeroes within an address may be replaced by a double colon ('::'). IPv4-embedded IPv6 addresses are not allowed.

change_mask=*value*
(string,7-15,charNB) The subnet mask which will be associated with interface. This value should be specified in dot-decimal notation.

change_mtu=*value*
(string,1-5,char10) The maximum transmission unit (MTU) size that is to be used on the interface.

change_p2p=*value*
(string,7-15,charNB) Changes the peer IP address to the specified value. This value should be specified in dot-decimal notation.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

The following parameters will be returned only when the IFCONFIG command returns an error or warning (RC=4, RS=4, or RC=8, RS=12/16):

error_data_length
(int4) Length of *error_data*.

error_data
(string) The output of the IFCONFIG command.

Usage Notes

1. This API issues the IFCONFIG command, which makes use of the NETSTAT and OBEYFILE commands to facilitate its operations. The SMAPI worker server IDs must therefore be included in the OBEY list for all TCPIP stacks they manage. Additionally, the SMAPI worker servers need to have links to the 198 TCPIP disk.
2. Syntax errors (RC = 24 and RS = *pprr*) are only applicable to the common input parameters. Syntax checking is *not* performed on the additional input parameters for this API. If a valid parameter is specified multiple times, the last value specified for that parameter will be used. For more information, see “Key-value pair (KVP) input parameters” on page 51.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
4	RC_WNG	4	RS_IFCONFIG_WARNING	The command completed successfully, but a warning condition was detected on IFCONFIG command
8	RC_ERR	8	RS_DEV_NOT_FOUND	Device does not exist
		12	RS_IFCONFIG_ERROR	An error was encountered on IFCONFIG command
		16	RS_IFCONFIG_UNEXPECTED	An unexpected condition was encountered on IFCONFIG command
		3002	RS_INVALID_PARAMETER	Invalid parameter name
		3003	RS_INVALID_OPERAND	Invalid parameter operand
		3004	RS_MISSING_PARAMETER	Required parameter missing
24	RCERR_SYNTAX	*pprr*	*pprr*	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
396	RCERR_INTERNAL	*nnnn*	*psrc*	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Network_IP_Interface_Query

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
`tcpip_stack=`*value*
`interface_all=`*value*
`interface_id=`*value*

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
interface_configuration_array_length
interface_configuration_array (1)
    interface_configuration_structure (2)

Note:

1. An array consists of zero or more of its components.
2. A structure consists of one set of its components.

Purpose

Use Network_IP_Interface_Query to obtain interface configurations for a specified TCP/IP stack virtual machine.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 26.

function_name
(string,13,char43) The API function name – in this case, 'Network_IP_Interface_Query.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) This must match an entry in the authorization file that also contains the *authenticated_userid* and the *function_name* (Network_IP_Interface_Query).

Note: The format for specifying the following additional input parameters is *parameter_name=value*, followed by a null (ASCIIZ) terminator. For more information, see "Key-value pair (KVP) input parameters" on page 51.

`tcpip_stack=`*value*
(string,1-8,char42) The TCP/IP stack whose interfaces are to be queried. This input parameter is required.

`interface_all=`*value*
(string,0-3,char26) One of the following:

YES
Return configurations of all interfaces, both active and inactive.

NO
Return configurations of active interfaces only. This is the default.

Note: You cannot specify both `interface_all=`YES and `interface_id=`*value*.

`interface_id=`*value*
(string,0-16,charNB) The identifier of the interface to be queried. Note that this value cannot begin with a dash (–), end with a colon (:), or contain a semicolon (;). If it is not specified, configurations for all interfaces for the specified TCP/IP stack will be returned.

Note: You cannot specify both `interface_all=`YES and `interface_id=`*value*.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

interface_configuration_array_length
(int4) Length of *interface_configuration_array*.

interface_configuration_array
(array) An array consisting of zero or more instances of *interface_configuration_structure*, with each structure terminated by a null (ASCIIZ) character, as follows:

interface_configuration_structure
(structure) A structure consisting of one set of the following *output_keyword=value* pairs, according to interface type, with a blank separating each pair. Note that each interface type (IUCV, CTC, HIPERS, QDIO_ETHERNET, EQDIO_ETHERNET, and LCS_ETHERNET) has a different set of potential *output_keyword=value* pairs, as follows:

- **IUCV:**
  - INTERFACE_ID=*value*
  - INET_ADDR=*value* (IPv4 address of the interface)
  - P_T_P=*value* (Peer IP address, omitted if not configured)
  - MASK=*value*
  - STATUS= UP | DOWN
  - BROADCAST=YES (Omitted if not configured)
  - MULTICAST=YES (Omitted if not configured)
  - POINTOPOINT=YES (Omitted if not configured)
  - MTU=*value*
  - VDEV=*value* (Omitted if not configured)
  - RDEV=*value* (Omitted if not configured)
  - TYPE=IUCV
  - CONNECTS_TO=*value*
  - LAN_OWNER=*value* (Omitted if not configured)
  - LAN_NAME=*value* (Omitted if not configured)
  - VSWITCH_NAME=*value* (Omitted if not configured)
  - VLAN=*value* (VLAN ID, omitted if not configured)
  - IPV6_VLAN=*value* (IPv6 VLAN ID, omitted if not configured)
  - CPU=*value*
  - FORWARDING=*value*
  - IPV4_PATH_MTU_DISCOVERY=*value*
  - RX_BYTES=*value* (Omitted if not configured)
  - TX_BYTES=*value* (Omitted if not configured)
  - IPV4_TAKEOVER_LINK=*value* (Omitted if not configured)
  - IPV6_TAKEOVER_LINK=*value* (Omitted if not configured)
  - ADDITIONAL_IPV4_ADDRESS_LENGTH=*value* (Length of additional IPv4 addresses, omitted if ADDITIONAL_IPV4_ADDRESS= not configured)
  - ADDITIONAL_IPV4_ADDRESS=*value* (May be more than one additional blank-delimited IPv4 addresses listed here, omitted if not configured)

  - IPV6_ADDRESS_LENGTH=*value* (Length of IPv6 addresses, omitted if IPV6_ADDRESS= not configured)
  - IPV6_ADDRESS=*value* (May be one or more blank-delimited IPv6 addresses, omitted if not configured.)
- **CTC:**
  - INTERFACE_ID=*value*
  - INET_ADDR=*value* (IPv4 address of the interface)
  - P_T_P=*value* (Peer IP address, omitted if not configured)
  - MASK=*value*
  - STATUS= UP | DOWN
  - BROADCAST=YES (Omitted if not configured)
  - MULTICAST=YES (Omitted if not configured)
  - POINTOPOINT=YES (Omitted if not configured)
  - MTU=*value*
  - VDEV=*value* (Omitted if not configured)
  - RDEV=*value* (Omitted if not configured)
  - TYPE=CTC
  - CONNECTS_TO=*value* (Omitted if not configured)
  - PORTNUMBER=*value*
  - LAN_OWNER=*value* (Omitted if not configured)
  - LAN_NAME=*value* (Omitted if not configured)
  - VSWITCH_NAME=*value* (Omitted if not configured)
  - VLAN=*value* (VLAN ID, omitted if not configured)
  - IPV6_VLAN=*value* (IPv6 VLAN ID, omitted if not configured)
  - CPU=*value*
  - FORWARDING=*value*
  - IPV4_PATH_MTU_DISCOVERY=*value*
  - RX_BYTES=*value* (Omitted if not configured)
  - TX_BYTES=*value* (Omitted if not configured)
  - IPV4_TAKEOVER_LINK=*value* (Omitted if not configured)
  - IPV6_TAKEOVER_LINK=*value* (Omitted if not configured)
  - ADDITIONAL_IPV4_ADDRESS_LENGTH=*value* (Length of additional IPv4 addresses, omitted if ADDITIONAL_IPV4_ADDRESS= not configured)
  - ADDITIONAL_IPV4_ADDRESS=*value* (May be more than one additional blank-delimited IPv4 addresses listed here, omitted if not configured)
  - IPV6_ADDRESS_LENGTH=*value* (Length of IPv6 addresses, omitted if IPV6_ADDRESS= not configured)
  - IPV6_ADDRESS=*value* (May be one or more blank-delimited IPv6 addresses, omitted if not configured.)
- **HIPERS:**
  - INTERFACE_ID=*value*
  - INET_ADDR=*value* (IPv4 address of the interface)
  - P_T_P=*value* (Peer IP address, omitted if not configured)
  - MASK=*value*
  - STATUS= UP | DOWN

- BROADCAST=YES (Omitted if not configured)
- MULTICAST=YES (Omitted if not configured)
- POINTOPOINT=YES (Omitted if not configured)
- MTU=*value*
- VDEV=*value* (Omitted if not configured)
- RDEV=*value* (Omitted if not configured)
- TYPE=HIPERS
- IPV6_STATE=*value*
- LAN_OWNER=*value* (Omitted if not configured)
- LAN_NAME=*value* (Omitted if not configured)
- VSWITCH_NAME=*value* (Omitted if not configured)
- VLAN=*value* (VLAN ID, omitted if not configured)
- IPV6_VLAN=*value* (IPv6 VLAN ID, omitted if not configured)
- CPU=*value*
- FORWARDING=*value*
- IPV4_PATH_MTU_DISCOVERY=*value*
- RX_BYTES=*value* (Omitted if not configured)
- TX_BYTES=*value* (Omitted if not configured)
- IPV4_TAKEOVER_LINK=*value* (Omitted if not configured)
- IPV6_TAKEOVER_LINK=*value* (Omitted if not configured)
- ADDITIONAL_IPV4_ADDRESS_LENGTH=*value* (Length of additional IPv4 addresses, omitted if ADDITIONAL_IPV4_ADDRESS= not configured)
- ADDITIONAL_IPV4_ADDRESS=*value* (May be more than one additional blank-delimited IPv4 addresses listed here, omitted if not configured)
- IPV6_ADDRESS_LENGTH=*value* (Length of IPv6 addresses, omitted if IPV6_ADDRESS= not configured)
- IPV6_ADDRESS=*value* (May be one or more blank-delimited IPv6 addresses, omitted if not configured.)

QDIO_ETHERNET or EQDIO_ETHERNET:

- INTERFACE_ID=*value*
- INET_ADDR=*value* (IPv4 address of the interface)
- P_T_P=*value* (Peer IP address, omitted if not configured)
- MASK=*value*
- STATUS= UP | DOWN
- BROADCAST=YES (Omitted if not configured)
- MULTICAST=YES (Omitted if not configured)
- POINTOPOINT=YES (Omitted if not configured)
- MTU=*value*
- VDEV=*value* (Omitted if not configured)
- RDEV=*value* (Omitted if not configured)
- TYPE=QDIO_ETHERNET | EQDIO_ETHERNET
- PORTNAME=*value*
- PORTNUMBER=*value*
- IPV6_STATE=*value*

  - TRANSPORT_TYPE=*value*
  - MAC_ADDR=*value* (Omitted if not configured)
  - IPV4_ROUTER_TYPE=*value* (Omitted if not configured)
  - IPV6_ROUTER_TYPE=*value* (Omitted if not configured)
  - LAN_OWNER=*value* (Omitted if not configured)
  - LAN_NAME=*value* (Omitted if not configured)
  - VSWITCH_NAME=*value* (Omitted if not configured)
  - VLAN=*value* (VLAN ID, omitted if not configured)
  - IPV6_VLAN=*value* (IPv6 VLAN ID, omitted if not configured)
  - CPU=*value*
  - FORWARDING=*value*
  - IPV4_PATH_MTU_DISCOVERY=*value*
  - RX_BYTES=*value* (Omitted if not configured)
  - TX_BYTES=*value* (Omitted if not configured)
  - IPV4_TAKEOVER_LINK=*value* (Omitted if not configured)
  - IPV6_TAKEOVER_LINK=*value* (Omitted if not configured)
  - ADDITIONAL_IPV4_ADDRESS_LENGTH=*value* (Length of additional IPv4 addresses, omitted if ADDITIONAL_IPV4_ADDRESS= not configured)
  - ADDITIONAL_IPV4_ADDRESS=*value* (May be more than one additional blank-delimited IPv4 addresses listed here, omitted if not configured)
  - IPV6_ADDRESS_LENGTH=*value* (Length of IPv6 addresses, omitted if IPV6_ADDRESS= not configured)
  - IPV6_ADDRESS=*value* (May be one or more blank-delimited IPv6 addresses, omitted if not configured.)
  - BUFFERS=value

- **LCS_ETHERNET:**
  - INTERFACE_ID=*value*
  - INET_ADDR=*value* (IPv4 address of the interface)
  - P_T_P=*value* (Peer IP address, omitted if not configured)
  - MASK=*value*
  - STATUS= UP | DOWN
  - BROADCAST=YES (Omitted if not configured)
  - MULTICAST=YES (Omitted if not configured)
  - POINTOPOINT=YES (Omitted if not configured)
  - MTU=*value*
  - VDEV=*value* (Omitted if not configured)
  - RDEV=*value* (Omitted if not configured)
  - TYPE=LCS_ETHERNET
  - PORTNUMBER=*value*
  - LAN_OWNER=*value* (Omitted if not configured)
  - LAN_NAME=*value* (Omitted if not configured)
  - VSWITCH_NAME=*value* (Omitted if not configured)
  - VLAN=*value* (VLAN ID, omitted if not configured)
  - IPV6_VLAN=*value* (IPv6 VLAN ID, omitted if not configured)

- CPU=*value*
- FORWARDING=*value*
- IPV4_PATH_MTU_DISCOVERY=*value*
- RX_BYTES=*value* (Omitted if not configured)
- TX_BYTES=*value* (Omitted if not configured)
- IPV4_TAKEOVER_LINK=*value* (Omitted if not configured)
- IPV6_TAKEOVER_LINK=*value* (Omitted if not configured)
- ADDITIONAL_IPV4_ADDRESS_LENGTH=*value* (Length of additional IPv4 addresses, omitted if ADDITIONAL_IPV4_ADDRESS= not configured)
- ADDITIONAL_IPV4_ADDRESS=*value* (May be more than one additional blank-delimited IPv4 addresses listed here, omitted if not configured)
- IPV6_ADDRESS_LENGTH=*value* (Length of IPv6 addresses, omitted if IPV6_ADDRESS= not configured)
- IPV6_ADDRESS=*value* (May be one or more blank-delimited IPv6 addresses, omitted if not configured.)

Usage Notes

1. This API can return the configurations for the following types of interfaces:
   - IUCV
   - CTC
   - HIPERS (HiperSocket)
   - EQDIO_ETHERNET
   - QDIO_ETHERNET
   - LCS_ETHERNET
2. Syntax errors (RC = 24 and RS = *pprr*) are only applicable to the common input parameters. Syntax checking is *not* performed on the additional input parameters for this API. If a valid parameter is specified multiple times, the last value specified for that parameter will be used. For more information, see "Key-value pair (KVP) input parameters" on page 51.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
4	RC_WNG	3022	RS_NO_INTERFACE_EXIST	No interface configured on specified TCP/IP stack virtual machine
8	RC_ERR	4	RS_NOT_FOUND	Specified interface not found
		24	RS_CONFLICTING_PARMS	Conflicting parameters
		3002	RS_INVALID_PARAMETER	Invalid parameter name
		3003	RS_INVALID_OPERAND	Invalid parameter operand
		3004	RS_MISSING_PARAMETER	Required parameter missing
		3020	RS_TCPIP_STACK_NOT_VALID	Specified TCP/IP stack is not available

RC	RC Name	RS	RS Name	Description
		3021	RS_NOT_IN_OBEYLIST	SMAPI worker server not in the obey list of specified TCP/IP stack
24	RCERR_SYNTAX	*pprr*	*pprr*	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
396	RCERR_INTERNAL	*nnnn*	*psrc*	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Network_IP_Interface_Remove

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
`tcpip_stack=`*value*
`interface_id=`*value*
`permanent=`*value*

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
error_data_length (error only)
error_data (error only)

Purpose

Use Network_IP_Interface_Remove to remove the existing network interface.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 27.

function_name
(string,13,char43) The API function name – in this case, 'Network_IP_Interface_Remove.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) This must match an entry in the authorization file that also contains the *authenticated_userid* and the *function_name* (Network_IP_Interface_Remove).

Note: The format for specifying the following additional input parameters is *parameter_name=value*, followed by a null (ASCIIZ) terminator. For more information, see "Key-value pair (KVP) input parameters" on page 51.

tcpip_stack=*value*
(string,1-8,char42) The TCP/IP stack to which the interface removal applies. This input parameter is required.

interface_id=*value*
(string,1-16,charNB) The identifier of the interface to be removed. Note that this value cannot begin with a dash (–), end with a colon (:), or contain a semicolon (;). This input parameter is required.

permanent=*value*
(string,0-3,char26) One of the following:

YES
The changes to the interface configuration will be permanent.

NO
The changes to the interface configuration will be temporary (created only for the current session). This is the default.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

The following parameters will be returned only when the IFCONFIG command returns an error or warning (RC=4, RS=4, or RC=8, RS=12/16):

error_data_length
(int4) Length of *error_data*.

error_data
(string) The output of the IFCONFIG command.

Usage Notes

1. This API issues the IFCONFIG command, which makes use of the NETSTAT and OBEYFILE commands to facilitate its operations. The SMAPI worker server IDs must therefore be included in the OBEY list for all TCPIP stacks they manage. Additionally, the SMAPI worker servers need to have links to the 198 TCPIP disk.
2. Syntax errors (RC = 24 and RS = *pprr*) are only applicable to the common input parameters. Syntax checking is *not* performed on the additional input parameters for this API. If a valid parameter is specified multiple times, the last value specified for that parameter will be used. For more information, see "Key-value pair (KVP) input parameters" on page 51.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
4	RC_WNG	4	RS_IFCONFIG_WARNING	The command completed successfully, but a warning condition was detected on IFCONFIG command
8	RC_ERR	8	RS_DEV_NOT_FOUND	Device does not exist
		12	RS_IFCONFIG_ERROR	An error was encountered on IFCONFIG command
		16	RS_IFCONFIG_UNEXPECTED	An unexpected condition was encountered on IFCONFIG command
		3002	RS_INVALID_PARAMETER	Invalid parameter name
		3003	RS_INVALID_OPERAND	Invalid parameter operand
		3004	RS_MISSING_PARAMETER	Required parameter missing
24	RCERR_SYNTAX	*pprr*	*pprr*	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
396	RCERR_INTERNAL	*nnnn*	*psrc*	Internal system error - product-specific return code (See "Internal Return Codes (RC = 396, 592, or 596)" on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist

RC	RC Name	RS	RS Name	Description
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Page_or_Spool_Volume_Add

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
`vol_addr=`*value*
`volume_label=`*value*
`volume_use=`*value*
`system_config_name=`*value*
`system_config_type=`*value*
`parm_disk_owner=`*value*
`parm_disk_number=`*value*
`parm_disk_password=`*value*

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Page_or_Spool_Volume_Add to add a full volume page or spool disk to the system.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 24.

function_name
(string,24,char43) The API function name – in this case, 'Page_or_Spool_Volume_Add'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See “Client Authentication” on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See “Client Authentication” on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) This must match an entry in the authorization file that also contains the *authenticated_userid* and the *function_name* (Page_or_Spool_Volume_Add).

Note: The format for specifying the following additional input parameters is *parameter_name=value*, followed by a null (ASCIIZ) terminator. They may be specified in any order. For more information, see “Key-value pair (KVP) input parameters” on page 51.

vol_addr=*value*
(string,1-4,char16) The real address of the volume to be used for page or spool space. This is a required parameter.

volume_label=*value*
(string,1-6,char36) The name to be associated with the newly formatted volume. This is a required parameter.

volume_use=*value*
(string,4-5,char26) One of the following:

PAGE
The volume is to be formatted and used as a page volume.

SPOOL
The volume is to be formatted and used as a spool volume.

This is a required parameter.

system_config_name=*value*
(string,0-8,char42) The file name of the system configuration file. The default is set by the "System_Config_File_Name =" statement in the DMSSICNF COPY file. (See the "Default SYSTEM CONFIG Link Values" section in “Configuring SMAPI” on page 30.)

system_config_type=*value*
(string,0-8,char42) The file type of the system configuration file. The default is set by the "System_Config_File_Type =" statement in the DMSSICNF COPY file. (See the "Default SYSTEM CONFIG Link Values" section in “Configuring SMAPI” on page 30.)

parm_disk_owner=*value*
(string,0-8,char42) The owner of the parm disk. The default is set by the "Parm_Disk_Owner =" statement in the DMSSICNF COPY file. (See the "Default SYSTEM CONFIG Link Values" section in “Configuring SMAPI” on page 30.)

parm_disk_number=*value*
(string,0-4,char16) Number of the parm disk, as defined in the VSMWORK1 directory. (See Usage Note “4” on page 419.) The default is set by the "Parm_Disk_Number =" statement in the DMSSICNF COPY file. (See the "Default SYSTEM CONFIG Link Values" section in “Configuring SMAPI” on page 30.)

parm_disk_password=*value*
(string,0-8,charNB) The multiwrite password for the parm disk. The default is "," and should not be changed. Any value other the default is ignored. (See “Configuring SMAPI” on page 30.)

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Usage Notes

1. Page_or_Spool_Volume_Add will occupy a long call server until a requested format is complete and the volume is brought online. It is recommended that there be at least two long call SMAPI servers defined in installations where this API may be used. The volume being added will be formatted, and all existing data will be lost. Take care to ensure that the volume is not already in use on this system or another system, such as another member of an SSI.
2. If the volume being added will be used as a SPOOL volume:
   - The volume *must* be available on all the systems in the SSI.
   - The SSI *must* be in a STABLE state.
3. The volume being added will be added to the CP_OWNED list for the current system session, and to the system configuration file for availability to all future system IPLs.
4. If the system is a member of an SSI:
   - The SSI *must* use a shared system configuration file.
   - The volume will be formatted with OWNER information set to the system and the SSI where this API is executed.
5. Syntax errors (RC = 24 and RS = *pprr*) are only applicable to the common input parameters. Syntax checking is *not* performed on the additional input parameters for this API. If a valid parameter is specified multiple times, the last value specified for that parameter will be used. For more information, see “Key-value pair (KVP) input parameters” on page 51.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
8	RC_ERR	8	RS_DEV_NOT_FOUND	Device does not exist

RC	RC Name	RS	RS Name	Description
		10	RS_DEV_NOT_AVAIL_TO_ ATTACH	Device not available for attachment
		12	RS_DEV_NOT_VOLUME	Device not a volume
		14	RS_FREE_MODE_NOT_AVAIL	Free modes not available
		20	RS_VOLID_IN_USE	Volume label already CP_OWNED on this system or in this system's configuration
		24	RS_PARM_DISK_LINK_ERR	Error linking parm disk
		28	RS_PARM_DISK_NOT_RW	Parm disk not RW
		32	RS_SYS_CONF_NOT_FOUND	System configuration not found on parm disk
		34	RS_SYS_CONF_BAD_DATA	System configuration has bad data
		38	RS_CPDISK_MODE_NOT_ AVAIL	CP disk modes not available
		40	RS_PARM_DISK_FULL	Parm disk is full
		42	RS_PDISK_ACC_NOT_ ALLOWED	Parm disk access not allowed
		44	RS_PDISK_PW_NOT_ SUPPLIED	Parm disk password not supplied
		46	RS_PDISK_PW_INCORRECT	Parm disk password is incorrect
		48	RS_PDISK_NOT_IN_SERVER _DIRECTORY	Parm disk is not in server's user directory
		50	RS_CPRELEASE_ERROR	Error with CPRELEASE of parm disk
		52	RS_CP_ACCESS_ERROR	Error in access of CPACCESS parm disk
		3002	RS_INVALID_PARAMETER	Invalid parameter name
		3003	RS_INVALID_OPERAND	Invalid parameter operand
		3004	RS_MISSING_PARAMETER	Required parameter missing
		3006	RS_SSI_UNSTABLE	SSI is not in a STABLE state
		3007	RS_SSI_CPOWNED_CONFLICT	The volume ID or slot is not available on all systems in the SSI
		3011	RS_NO_SLOT_AVAILABLE	No unique CP_OWNED slot available on system and in System Config
24	RCERR_SYNTAX	*pprr*	*pprr*	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager

RC	RC Name	RS	RS Name	Description
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
396	RCERR_INTERNAL	*nnnn*	*psrc*	Internal system error - product-specific return code (See "Internal Return Codes (RC = 396, 592, or 596)" on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Process_ABEND_Dump

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
`spoolid=`*value*

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Process_ABEND_Dump to instruct the dump processing userid to process one or more ABEND dumps from its reader and place them in the dump processing location specified in the DMSSICNF COPY file. (See the `Dump_Processing_Location =` entry in "Configuring SMAPI" on page 30 for more information.)

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 18.

function_name
(string,18,char43) The API function name – in this case, 'Process_ABEND_Dump'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) This must match an entry in the authorization file that also contains the *authenticated_userid* and the *function_name* (Process_ABEND_Dump).

Note: The format for specifying the following additional input parameter is *parameter_name=value*, followed by a null (ASCIIZ) terminator. For more information, see "Key-value pair (KVP) input parameters" on page 51.

spoolid=*value*
(string,0-8,char42) The spool ID of the ABEND dump to be processed, or "ALL" to process all remaining ABEND dumps. If not specified, the next ABEND dump is processed.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Usage Notes

1. The actual processing of the dump occurs asynchronously. When it has completed, a type 2008 event will be transmitted indicating success or failure.
2. Under normal circumstances, OPERATNS will automatically attempt to process any dumps that appear in its reader. In the event that a dump is found, a type 2010 event will be transmitted to indicate the success or failure of this automatic processing.
3. If a dump file is successfully loaded to SFS, it will be purged from the OPERATNS userid's reader.
4. If a dump does not appear to be processed during automatic processing or when a spool ID of "ALL" is given, and no errors are generated, it is likely in HOLD status. Dumps in this state can be processed only if their spool ID is explicitly provided.
5. Syntax errors (RC = 24 and RS = *pprr*) are only applicable to the common input parameters. Syntax checking is *not* performed on the additional input parameters for this API. If a valid parameter is specified multiple times, the last value specified for that parameter will be used. For more information, see "Key-value pair (KVP) input parameters" on page 51.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
8	RC_ERR	3002	RS_INVALID_PARAMETER	Invalid parameter name
		3003	RS_INVALID_OPERAND	Invalid parameter operand
		3004	RS_MISSING_PARAMETER	Required parameter missing
24	RCERR_SYNTAX	*pprr*	*pprr*	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
200	RCERR_IMAGEOP	12	RS_NOT_ACTIVE	The dump processing userid (OPERATNS) is either not logged on or is busy processing a dump
396	RCERR_INTERNAL	*nnnn*	*psrc*	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Profile_Create_DM

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
profile_record_array_length
profile_record_array (1)
profile_record_structure (2)
profile_record_length
profile_record

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Note:

1. An array consists of zero or more of its components.
2. A structure consists of one set of its components.

Purpose

Use Profile_Create_DM to create a profile directory entry to be included in the definition of a virtual image in the directory.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 17.

function_name
(string,36,char43) The API function name – in this case, 'Profile_Create_DM'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The name of the profile to be created.

profile_record_array_length
(int4) Length of *profile_record_array*.

profile_record_array
(array) An array consisting of zero or more instances of *profile_record_structure*, as follows:

profile_record_structure
(structure) A structure consisting of one set of *profile_record_length* and *profile_record*, as follows:

profile_record_length
(int4) Length of *profile_record*.

profile_record
(string,1-72,charNA) A record of the profile directory entry.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
		8	RS_OFFLINE	Request successful; object directory offline
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter pprr
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		12	RS_AUTHERR_DM	Request not authorized by directory manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
400	RCERR_IMAGEDEF	8	RS_NAME_EXISTS	Profile name already defined
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available
592	RCERR_ASYNC_DM	nnnn	opid	Asynchronous operation started - product-specific asynchronous operation ID (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
596	RCERR_INTERNAL_DM	nnnn	psrc	Internal directory manager error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Profile_Delete_DM

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Profile_Delete_DM to delete a profile directory entry.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 17.

function_name
(string,36,char43) The API function name – in this case, 'Profile_Delete_DM'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See “Client Authentication” on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The name of the profile to be deleted.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
		8	RS_OFFLINE	Request successful; object directory offline
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter pprr
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		12	RS_AUTHERR_DM	Request not authorized by directory manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
400	RCERR_IMAGEDEF	4	RS_NOT_FOUND	Profile definition not defined
		12	RS_LOCKED	Profile definition is locked
		16	RS_CANNOT_DELETE	Profile definition cannot be deleted

RC	RC Name	RS	RS Name	Description
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available
592	RCERR_ASYNC_DM	nnnn	opid	Asynchronous operation started - product-specific asynchronous operation ID (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
596	RCERR_INTERNAL_DM	nnnn	psrc	Internal directory manager error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Profile_Lock_DM

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Profile_Lock_DM to lock a profile directory entry so that it cannot be changed.

Use this function before replacing a profile directory entry with Profile_Replace_DM.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 15.

function_name
(string,15,char43) The API function name – in this case, 'Profile_Lock_DM'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See “Client Authentication” on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See “Client Authentication” on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The name of the profile to be locked.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Usage Notes

1. Use Profile_Lock_DM before a Profile_Replace_DM operation. The Profile_Replace_DM operation will unlock the profile directory entry upon completion. If, after locking the profile directory entry, you do not perform the Profile_Replace_DM, use Profile_Unlock_DM to unlock the profile directory entry.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		12	RS_AUTHERR_DM	Request not authorized by directory manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
400	RCERR_IMAGEDEF	4	RS_NOT_FOUND	Profile definition not defined
		12	RS_LOCKED	Profile definition is locked

RC	RC Name	RS	RS Name	Description
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available
592	RCERR_ASYNC_DM	nnnn	opid	Asynchronous operation started - product-specific asynchronous operation ID (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
596	RCERR_INTERNAL_DM	nnnn	psrc	Internal directory manager error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Profile_Lock_Query_DM

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
lock_info_structure (2)
  lock_info_structure_length
  locked_type
  profile_locked_by
locked_dev_array_length
locked_dev_array (1)
  dev_lock_info_structure (2)
    dev_address
    dev_locked_by

Note:

1. An array consists of zero or more of its components.
2. A structure consists of one set of its components.

Purpose

Use Profile_Lock_Query_DM to query the status of whether a directory manager lock is in effect for a specific profile.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 21.

function_name
(string,13,char43) The API function name – in this case, 'Profile_Lock_Query_DM'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The name of the profile for which the directory lock status is being queried.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

lock_info_structure
(structure) A structure consisting of the following blank-delimited components (this structure will be absent when RS = RS_UNLOCKED):

lock_info_structure_length
(int4) The combined length of the remaining parameters in *lock_info_structure* (not including this parameter). This will be zero when RS=RS_UNLOCKED.

locked_type
(string,6-7,char26) One of the following:

PROFILE
Profile locked

DEVICE
Device(s) locked

profile_locked_by
(string,0-8,char42) The image that performed the profile lock. This will be absent if *locked_type*=DEVICE.

locked_dev_array_length
(int4) Length of *locked_dev_array*. This array will be absent if RS = RS_UNLOCKED or *locked_type* = PROFILE.

locked_dev_array
(array) An array consisting of zero or more instances of *dev_lock_info_structure*, as follows:

dev_lock_info_structure
(structure) A structure consisting of one set of the following parameters:

dev_address
(string,1-4,char16) The address of locked device.

dev_locked_by
(string,1-8,char42) The image that performed the device lock action.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	12	RS_LOCKED	Image or device(s) locked
		24	RS_UNLOCKED	Image or device(s) unlocked
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		12	RS_AUTHERR_DM	Request not authorized by directory manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
400	RCERR_IMAGEDEF	4	RS_NOT_FOUND	Image definition not defined
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available
592	RCERR_ASYNC_DM	nnnn	opid	Asynchronous operation started - product-specific asynchronous operation ID (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
596	RCERR_INTERNAL_DM	nnnn	psrc	Internal directory manager error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist

RC	RC Name	RS	RS Name	Description
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Profile_Query_DM

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
profile_record_array_length
profile_record_array (1)
profile_record_structure (2)
profile_record_length
profile_record

Note:

1. An array consists of zero or more of its components.
2. A structure consists of one set of its components.

Purpose

Use Profile_Query_DM to query a profile directory entry.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 16.

function_name
(string,36,char43) The API function name – in this case, 'Profile_Query_DM'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The name of the profile being queried.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

profile_record_array_length
(int4) Length of *profile_record_array*.

profile_record_array
(array) An array consisting of zero or more instances of *profile_record_structure*, as follows:

profile_record_structure
(structure) A structure consisting of one set of *profile_record_length* and *profile_record*, as follows:

profile_record_length
(int4) Length of *profile_record*.

profile_record
(string,1-80,charNA) A record of the profile directory entry.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful

RC	RC Name	RS	RS Name	Description
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter pprr
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		12	RS_AUTHERR_DM	Request not authorized by directory manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
400	RCERR_IMAGEDEF	4	RS_NOT_FOUND	Profile definition not defined
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available
592	RCERR_ASYNC_DM	nnnn	opid	Asynchronous operation started - product-specific asynchronous operation ID (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
596	RCERR_INTERNAL_DM	nnnn	psrc	Internal directory manager error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Profile_Replace_DM

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
profile_record_array_length
profile_record_array (1)
profile_record_structure (2)
profile_record_length
profile_record

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Note:

1. An array consists of zero or more of its components.
2. A structure consists of one set of its components.

Purpose

Use Profile_Replace_DM to replace the definition of a profile to be included in a virtual image in the directory.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 18.

function_name
(string,36,char43) The API function name – in this case, 'Profile_Replace_DM'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The name of the profile directory entry to be replaced.

profile_record_array_length
(int4) Length of *profile_record_array*.

profile_record_array
(array) An array consisting of zero or more instances of *profile_record_structure*, as follows:

profile_record_structure
(structure) A structure consisting of one set of *profile_record_length* and *profile_record*, as follows:

profile_record_length
(int4) Length of *profile_record*.

profile_record
(string,1-72,charNA) A record of the profile directory entry.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Usage Notes

1. Use Image_Lock_DM before a Profile_Replace_DM operation. The Profile_Replace_DM operation will unlock the profile directory entry upon completion. If, after locking the profile directory entry, you do not perform the Profile_Replace_DM, use Image_Unlock_DM to unlock the profile directory entry.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
		8	RS_OFFLINE	Request successful; object directory offline
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter pprr
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		12	RS_AUTHERR_DM	Request not authorized by directory manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
400	RCERR_IMAGEDEF	4	RS_NOT_FOUND	Profile definition not defined
		24	RS_NOT_LOCKED	Profile name is not locked
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available
592	RCERR_ASYNC_DM	nnnn	opid	Asynchronous operation started - product-specific asynchronous operation ID (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
596	RCERR_INTERNAL_DM	nnnn	psrc	Internal directory manager error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Profile_Unlock_DM

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Profile_Unlock_DM to unlock a profile directory entry.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 17.

function_name
(string,17,char43) The API function name – in this case, 'Profile_Unlock_DM'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See “Client Authentication” on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The name of the profile to be unlocked.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Usage Notes

1. Use Profile_Unlock_DM to unlock a locked profile directory entry if you do not perform a Profile_Replace_DM operation.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		12	RS_AUTHERR_DM	Request not authorized by directory manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
400	RCERR_IMAGEDEF	4	RS_NOT_FOUND	Profile definition not defined
		24	RS_NOT_LOCKED	Profile definition is not locked

RC	RC Name	RS	RS Name	Description
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available
592	RCERR_ASYNC_DM	nnnn	opid	Asynchronous operation started - product-specific asynchronous operation ID (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
596	RCERR_INTERNAL_DM	nnnn	psrc	Internal directory manager error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Prototype_Create_DM

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
prototype_record_array_length
prototype_record_array (1)
prototype_record_structure (2)
prototype_record_length
prototype_record

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Note:

1. An array consists of zero or more of its components.
2. A structure consists of one set of its components.

Purpose

Use Prototype_Create_DM to create a new virtual image prototype.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 19.

function_name
(string,19,char43) The API function name – in this case, 'Prototype_Create_DM'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The name of the prototype being created.

prototype_record_array_length
(int4) Length of *prototype_record_array*.

prototype_record_array
(array) An array consisting of zero or more instances of *prototype_record_structure*, as follows:

prototype_record_structure
(structure) A structure consisting of one set of *prototype_record_length* and *prototype_record*, as follows:

prototype_record_length
(int4) Length of *prototype_record*.

prototype_record
(string,1-72,charNA) A single record to be added to the new virtual image prototype.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Usage Notes

1. See the "Creating and Updating a User Directory" chapter in *z/VM: CP Planning and Administration* for more information on the directory format and on specific directory statements.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		12	RS_AUTHERR_DM	Request not authorized by directory manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
416	RCERR_PROTODEF	0	RS_NONE	Prototype definition error
		4	RS_NOT_FOUND	Prototype definition not found
		8	RS_NAME_EXISTS	Prototype already exists
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available
592	RCERR_ASYNC_DM	nnnn	opid	Asynchronous operation started - product-specific asynchronous operation ID (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
596	RCERR_INTERNAL_DM	nnnn	psrc	Internal directory manager error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Prototype_Delete_DM

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Prototype_Delete_DM to delete an image prototype.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 19.

function_name
(string,19,char43) The API function name – in this case, 'Prototype_Delete_DM'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See “Client Authentication” on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The name of the prototype to be deleted.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		12	RS_AUTHERR_DM	Request not authorized by directory manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
416	RCERR_PROTODEF	4	RS_NOT_FOUND	Prototype definition not found
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available

RC	RC Name	RS	RS Name	Description
592	RCERR_ASYNC_DM	nnnn	opid	Asynchronous operation started - product-specific asynchronous operation ID (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
596	RCERR_INTERNAL_DM	nnnn	psrc	Internal directory manager error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Prototype_Name_Query_DM

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
prototype_name_array_length
prototype_name_array (1)
prototype_name_structure (2)
prototype_name_length
prototype_name

Note:

1. An array consists of zero or more of its components.
2. A structure consists of one set of its components.

Purpose

Use Prototype_Name_Query_DM to obtain a list of names of defined prototypes.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 23.

function_name
(string,23,char43) The API function name – in this case, 'Prototype_Name_Query_DM'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) This must match an entry in the authorization file that also contains the *authenticated_userid* and the *function_name* (Prototype_Name_Query_DM).

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

prototype_name_array_length
(int4) Length of *prototype_name_array*.

prototype_name_array
(array) An array consisting of zero or more instances of *prototype_name_structure*, as follows:

prototype_name_structure
(structure) A structure consisting of one set of *prototype_name_length* and *prototype_name*, as follows:

prototype_name_length
(int4) Length of *prototype_name*.

prototype_name
(string,1-8,char42) The name of the prototype.

Usage Notes

1. See the "Creating and Updating a User Directory" chapter in *z/VM: CP Planning and Administration* for more information on the directory format and on specific directory statements.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		12	RS_AUTHERR_DM	Request not authorized by directory manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
416	RCERR_PROTODEF	4	RS_NOT_FOUND	Prototype definition not found
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available
592	RCERR_ASYNC_DM	nnnn	opid	Asynchronous operation started - product-specific asynchronous operation ID (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
596	RCERR_INTERNAL_DM	nnnn	psrc	Internal directory manager error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Prototype_Query_DM

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
prototype_record_array_length
prototype_record_array (1)
prototype_record_structure (2)
prototype_record_length
prototype_record

Note:

1. An array consists of zero or more of its components.
2. A structure consists of one set of its components.

Purpose

Use Prototype_Query_DM to query the characteristics of an image prototype.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 18.

function_name
(string,18,char43) The API function name – in this case, 'Prototype_Query_DM'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The name of the prototype to be queried

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

prototype_record_array_length
(int4) Length of *prototype_record_array*.

prototype_record_array
(array) An array consisting of zero or more instances of *prototype_record_structure*, as follows:

prototype_record_structure
(structure) A structure consisting of one set of *prototype_record_length* and *prototype_record*, as follows:

prototype_record_length
(int4) Length of *prototype_record*.

prototype_record
(string,1-72,charNA) A record from the virtual image prototype.

Usage Notes

1. See the "Creating and Updating a User Directory" chapter in *z/VM: CP Planning and Administration* for more information on the directory format and on specific directory statements.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		12	RS_AUTHERR_DM	Request not authorized by directory manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
416	RCERR_PROTODEF	4	RS_NOT_FOUND	Prototype definition not found
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available
592	RCERR_ASYNC_DM	nnnn	opid	Asynchronous operation started - product-specific asynchronous operation ID (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
596	RCERR_INTERNAL_DM	nnnn	psrc	Internal directory manager error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Prototype_Replace_DM

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
prototype_record_array_length
prototype_record_array (1)
prototype_record_structure (2)
prototype_record_length
prototype_record

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Note:

1. An array consists of zero or more of its components.
2. A structure consists of one set of its components.

Purpose

Use Prototype_Replace_DM to replace an existing prototype.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 20.

function_name
(string,20,char43) The API function name – in this case, 'Prototype_Replace_DM'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The name of the prototype to be replaced.

prototype_record_array_length
(int4) Length of *prototype_record_array*.

prototype_record_array
(array) An array consisting of zero or more instances of *prototype_record_structure*, as follows:

prototype_record_structure
(structure) A structure consisting of one set of *prototype_record_length* and *prototype_record*, as follows:

prototype_record_length
(int4) Length of *prototype_record*.

prototype_record
(string,1-72,charNA) A single record to be added to the new virtual image prototype.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Usage Notes

1. See the "Creating and Updating a User Directory" chapter in *z/VM: CP Planning and Administration* for more information on the directory format and on specific directory statements.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		12	RS_AUTHERR_DM	Request not authorized by directory manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
416	RCERR_PROTODEF	4	RS_NOT_FOUND	Prototype definition not found
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available
592	RCERR_ASYNC_DM	nnnn	opid	Asynchronous operation started - product-specific asynchronous operation ID (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
596	RCERR_INTERNAL_DM	nnnn	psrc	Internal directory manager error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Query_ABEND_Dump

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
`location`=*value*

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
abend_dump_array (1)
abend_dump_structure (2)
abend_dump_loc
abend_dump_id
abend_dump_date
abend_dump_dist

Note:

1. An array consists of zero or more of its components.
2. A structure consists of one set of its components.

Purpose

Use Query_ABEND_Dump to display the current ABEND dumps that appear in the OPERATNS userid's reader or have already been processed to the dump processing location specified in the DMSSICNF COPY file. (See the `Dump_Processing_Location =` entry in "Configuring SMAPI" on page 30 for more information.)

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 16.

function_name
(string,16,char43) The API function name – in this case, Query_ABEND_Dump'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) This must match an entry in the authorization file that also contains the *authenticated_userid* and the *function_name* (Query_ABEND_Dump).

Note: The format for specifying the following additional input parameter is *parameter_name=value*, followed by a null (ASCIIZ) terminator. For more information, see "Key-value pair (KVP) input parameters" on page 51.

`location=`*value*
(string,0-3,char26) One of the following:

RDR
Query ABEND dumps in the reader (unprocessed).

SFS
Query ABEND dumps in the VMSYSU:OPERATNS. SFS directory (processed).

ALL
Query ABEND dumps both in the reader and the SFS directory.

If not specified, ALL is the default.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

abend_dump_array
(array) An array consisting of zero or more instances of *abend_dump_structure*, as follows:

abend_dump_structure
(structure) A structure consisting of one set of the following parameters:

abend_dump_loc
(int1) The location of the ABEND dump file, as follows:

1
Reader (unprocessed)

2
SFS directory (processed)

abend_dump_id
(string,8,char42) The spool ID (for a reader file) or file name (for an SFS file) of the abend dump.

abend_dump_date
(string,10,char42) The date of the ABEND dump in ISO format: *yyyy-mm-dd*

abend_dump_dist
(string,8,char42 plus blank) For reader files, this is the DIST of the ABEND dump. For a file in the SFS directory, this field consists of eight blank spaces.

Usage Notes

1. Syntax errors (RC = 24 and RS = *pprr*) are only applicable to the common input parameters. Syntax checking is *not* performed on the additional input parameters for this API. If a valid parameter is specified multiple times, the last value specified for that parameter will be used. For more information, see “Key-value pair (KVP) input parameters” on page 51.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
		28	RS_EMPTY	Return buffer is empty
8	RC_ERR	3002	RS_INVALID_PARAMETER	Invalid parameter name
		3003	RS_INVALID_OPERAND	Invalid parameter operand
24	RCERR_SYNTAX	*pprr*	*pprr*	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
200	RCERR_IMAGEOP	12	RS_NOT_ACTIVE	The dump processing userid (OPERATNS) is either not logged on or is busy processing a dump

RC	RC Name	RS	RS Name	Description
396	RCERR_INTERNAL	*nnnn*	*psrc*	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Query_All_DM

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
query_keyword_parameter_list_length
query_keyword_parameter_list

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters (if FORMAT=YES specified):

output_length
request_id
return_code
reason_code
directory_entries_array_length
directory_entries_array (1)
    directory_entry_structure (2)
        directory_entry_structure_length
        directory_entry_type
        directory_entry_id_length
        directory_entry_id
        directory_entry_data_length
        directory_entry_data

Response 2 – Output Parameters (if FORMAT=NO specified):

output_length
request_id
return_code
reason_code
directory_entries_array_length
directory_entries_array (1)
    directory_entry_structure (2)
        directory_entry_structure_length
        directory_entry_type
        directory_entry_id_length
        directory_entry_id
        directory_entry_data_array_length
        directory_entry_data_array (1)
            directory_entry_data_structure (2)
                directory_entry_record_length
                directory_entry_record

Note:

1. An array consists of zero or more of its components.
2. A structure consists of one set of its components.

Purpose

Use Query_All_DM to obtain the contents of the entire system directory.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 12.

function_name
(string,12,char43) The API function name – in this case, 'Query_All_DM'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) Used strictly for authorization – i.e. the authenticated user must have authorization to perform this function for this target.

Note: The format for specifying the following additional input parameter is *parameter_name*=*value*, followed by a null (ASCIIZ) terminator. For more information, see "Key-value pair (KVP) input parameters" on page 51.

query_keyword_parameter_list_length
(int4) Length of *query_keyword_parameter_list*.

query_keyword_parameter_list
(string,1-maxlength,charNA) The remaining set of *keyword_parameter*=*value* input parameters.

Note: The format for specifying this additional input parameter is *keyword_parameter*=*value*, followed by a null (ASCIIZ) terminator. For more information, see "Key-value pair (KVP) input parameters" on page 51.

Table 12. Input Keywords and Values for Query_All_DM	
keyword_parameter=	***value***
FORMAT=	**YES** Output data formatted. **NO** Output data unformatted. If unspecified, YES is the default. See Usage Note "1" on page 469.
MATCHKEY=	(char) Character match key, either exact or fuzzy, to be used for determining which directory entries are to be seen. The entire string that constitutes the MATCHKEY *value* must be in ASCII. See Usage Note "3" on page 469.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

directory_entries_array_length
(int4) Length of *directory_entries_array*.

directory_entries_array
(array) An array of ASCII-format data consisting of zero or more instances of *directory_entry_structure*, as follows:

directory_entry_structure
(structure) A structure consisting of one set of the following parameters:

directory_entry_structure_length
(int4) The combined length of the remaining parameters in *directory_entry_structure* (not including this parameter).

directory_entry_type
(int4) One of the following:

0
USER

1
PROFILE

2
USER defined via POOL

3
POOL

4
DIRECTORY

5
GLOBAL

6
IDENTITY

7
SUBCONFIG

8
OTHER

See Usage Note "1" on page 469.

directory_entry_id_length
(int4) Length of *directory_entry_id*.

directory_entry_id
(string,1-10,charNA) The directory entry ID.

Rest of output contingent on FORMAT=YES/NO. If FORMAT=YES was specified:

directory_entry_data_length
(int4) Length of *directory_entry_data*.

directory_entry_data
(string,1-maxlength,charNA) A series of null-terminated strings, each containing "*directory_keyword_parameter*=" followed by a series of blank-delimited "*directory_keyword_operand=directory_keyword_operand_value*" pairs, similar to the output for "Image_Definition_Query_DM" on page 182.

If FORMAT=NO was specified:

directory_entry_data_array_length
(int4) Length of *directory_entry_data_array*.

directory_entry_data_array
(array) An array consisting of zero or more instances of *directory_entry_data_structure*, as follows:

directory_entry_data_structure
(structure) A structure consisting of one set of *directory_entry_record_length* and *directory_entry_record*, as follows:

directory_entry_record_length
(int4) Length of *directory_entry_record*.

directory_entry_record
(string,1-80,charNA) A record from a directory entry, similar to the output for "Image_Query_DM" on page 308.

Usage Notes

1. If `format`=YES is specified, the CP directory entry data returned is formatted internally via Image_Definition_Query_DM processing. (See "Image_Definition_Query_DM" on page 182 for more information on the output format.) All returned *directory_entry_type* values are categorized as type 0 (USER) or type 1 (PROFILE). For IDENTITYs, the definitions will be returned from the IDENTITY entry, any included PROFILE, and any existing member-specific SUBCONFIG for the system executing the API. IDENTITYs will be returned as *directory_entry_type* 0 (USER).

   If `format`=NO is specified, all CP directory entries are returned unformatted, as per Image_Query_DM. (See "Image_Query_DM" on page 308 for more information on the output format.)
2. For more information on the directory format and on specific directory statements, see the "Creating and Updating a User Directory" chapter in *z/VM: CP Planning and Administration*.
3. A MATCHKEY=*value* can be either exact or fuzzy, as follows:

Exact *value*
The *value* is exact if it contains no wildcard characters. Potential output records against which the *value* is compared must match the pattern exactly (they must have the same length and the same data).

Fuzzy *value*
The *value* is fuzzy if it contains wildcard characters. Potential output records against which the *value* is compared must match the pattern specified by the *value*, allowing for the wildcards.

The allowable wildcard characters are * (asterisk), % (percent), and ' (apostrophe). They are interpreted in a similar way as the wildcard characters in CMS file names and file types.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
8	RC_ERR	28	RS_EMPTY	Return buffer is empty
		3002	RS_INVALID_PARAMETER	Invalid parameter name
		3003	RS_INVALID_OPERAND	Invalid parameter operand
24	RCERR_SYNTAX	*pprr*	*pprr*	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		12	RS_AUTHERR_DM	Request not authorized by directory manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available
592	RCERR_ASYNC_DM	*nnnn*	*opid*	Asynchronous operation started - product-specific asynchronous operation ID (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
596	RCERR_INTERNAL_DM	*nnnn*	*psrc*	Internal directory manager error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Query_API_Functional_Level

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Query_API_Functional_Level to obtain the support level of the server and functions, as follows:

- For z/VM 7.2, this API provides this return code and reason code: 0, 720.
- For z/VM 7.3, this API provides this return code and reason code: 0, 730.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 26.

function_name
(string,26,char43) The API function name – in this case, 'Query_API_Functional_Level'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See “Client Authentication” on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) This must match an entry in the authorization file that also contains the *authenticated_userid* and the *function_name* (Query_API_Functional_Level).

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	720	RS_720	The API functional level is z/VM 7.2
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter pprr
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
396	RCERR_INTERNAL	nnnn	psrc	Internal system error - product-specific return code (See "Internal Return Codes (RC = 396, 592, or 596)" on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist

RC	RC Name	RS	RS Name	Description
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Query_Asynchronous_Operation_DM

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
operation_id

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Query_Asynchronous_Operation_DM to query the status of an asynchronous directory manager operation.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 31.

function_name
(string,31,char43) The API function name – in this case, 'Query_Asynchronous_Operation_DM'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See “Client Authentication” on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) This must match an entry in the authorization file that also contains the *authenticated_userid* and the *function_name* (Query_Asynchronous_Operation_DM).

operation_id
(int4; range 0-2147483647) The identifier of the operation to be queried.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Usage Notes

1. If a nonexistent *operation_id* is specified, a return code of 0 with a reason code of 100 will be returned.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
		100	RS_ASYNC_OP_SUCCEEDED	Asynchronous operation succeeded
		104	RS_ASYNC_OP_IN _PROGRESS	Asynchronous operation in progress
		108	RS_ASYNC_OP_FAILED	Asynchronous operation failed
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager

RC	RC Name	RS	RS Name	Description
		12	RS_AUTHERR_DM	Request not authorized by directory manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available
592	RCERR_ASYNC_DM	nnnn	opid	Asynchronous operation started - product-specific asynchronous operation ID (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
596	RCERR_INTERNAL_DM	nnnn	psrc	Internal directory manager error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Query_Directory_Manager_Level_DM

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
directory_manager_level_length
directory_manager_level

Purpose

Use Query_Directory_Manager_Level_DM to query the directory manager that is being used and its functional level.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 32.

function_name
(string,32,char43) The API function name – in this case, 'Query_Directory_Manager_Level_DM'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See “Client Authentication” on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) This must match an entry in the authorization file that also contains the *authenticated_userid* and the *function_name* (Query_Directory_Manager_Level_DM).

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

directory_manager_level_length
(int4) Length of *directory_manager_level*.

directory_manager_level
(string,1-100,charNA) The directory manager name and level.

Usage Notes

1. See the "Creating and Updating a User Directory" chapter in *z/VM: CP Planning and Administration* for more information on the directory format and on specific directory statements.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		12	RS_AUTHERR_DM	Request not authorized by directory manager
		16	RS_AUTHERR_SERVER	Request not authorized by server

RC	RC Name	RS	RS Name	Description
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available
592	RCERR_ASYNC_DM	nnnn	opid	Asynchronous operation started - product-specific asynchronous operation ID (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
596	RCERR_INTERNAL_DM	nnnn	psrc	Internal directory manager error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Response_Recovery

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
failed_request_ID

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
response_data

Purpose

Use Response_Recovery to obtain response data from previous calls that may have failed.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 17.

function_name
(string,17,char43) The API function name – in this case, 'Response_Recovery'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See “Client Authentication” on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) This must match an entry in the authorization file that also contains the *authenticated_userid* and the *function_name* (Response_Recovery).

failed_request_ID
(int4) Previously-failed *request_id* for which you wish to recover response data.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

If RC=0 and RS=0, the following is returned:

response_data
(string) Recovered response, as associated with the specified *failed_request_ID*.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RC_NONE	Request successful
4	RC_WNG	4	RS_NOT_FOUND	Request does not exist
24	RCERR_SYNTAX	*pprr*	*pprr*	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid

RC	RC Name	RS	RS Name	Description
396	RCERR_INTERNAL	nnnn	psrc	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Shared_Memory_Access_Add_DM

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
memory_segment_name_length
memory_segment_name

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Shared_Memory_Access_Add_DM to add restricted (RSTD) access to a shared memory segment.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 27.

function_name
(string,27,char43) The API function name – in this case, 'Shared_Memory_Access_Add_DM'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See “Client Authentication” on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The userid or list of userids being granted access to the memory segment.

memory_segment_name_length
(int4) Length of *memory_segment_name*.

memory_segment_name
(string,1-8,char42) The name of the memory segment to which access is being granted.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Usage Notes

1. This function checks the name to determine whether it is a list, and if not, processes the name as a single image name. Therefore, lists should be given names that cannot be confused with image names.
2. During authorization checking and function processing, name lists are only expanded once; although a name within a list may also be the name of a list, the second (nested) list will not be expanded.
3. If *target_identifier* is a list of userids, and
   - Processing for all entries in the list is successful, then RC=0.
   - Processing for any entry in the list is **not** successful, then RC=504 (RCERR_LIST_DM) and RS is set to the position in the list where the error occurred. This is where processing stops. IDs located earlier in the list are processed but no IDs located later in the list are processed.
4. If *target_identifier* is a list, then you may have special considerations for checking authorizations, depending on your directory manager. Refer to your directory manager documentation for more information.
5. Note that while a list name specified for *target_identifier* is generally limited to 64 characters (in the char43 character set) for other APIs, here a list name is limited by the IBM DirMaint directory manager to 8 characters in the char42 character set (meaning that no underscores are allowed). This same restriction applies to Shared_Memory_Access_Query_DM and Shared_Memory_Access_Remove_DM.

6. The shared memory segment specified in *memory_segment_name* does not need to be defined before restricted access is added.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
		8	RS_OFFLINE	Request successful; object directory offline
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		12	RS_AUTHERR_DM	Request not authorized by directory manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
400	RCERR_IMAGEDEF	4	RS_NOT_FOUND	Image definition not defined
		12	RS_LOCKED	Image definition is locked
424	RCERR_SEGMENT_DM	4	RS_SEG_NAME_DUPLICATE	Namesave statement already exists
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available
504	RCERR_LIST_DM	nnnn	psrc	Target ID not added
592	RCERR_ASYNC_DM	nnnn	opid	Asynchronous operation started - product-specific asynchronous operation ID (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
596	RCERR_INTERNAL_DM	nnnn	psrc	Internal directory manager error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
600	RCERR_SHSTOR	20	RS_NOT_AUTHORIZED	Not authorized to issue internal system command or is not authorized for RSTD segment
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory

RC	RC Name	RS	RS Name	Description
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Shared_Memory_Access_Query_DM

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
memory_segment_name_length
memory_segment_name

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
name_array_length
name_array (1)
name_structure (2)
name_length
name

Note:

1. An array consists of zero or more of its components.
2. A structure consists of one set of its components.

Purpose

Use Shared_Memory_Access_Query_DM to query the restricted (RSTD) access to a shared memory segment.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 29.

function_name
(string,29,char43) The API function name – in this case, 'Shared_Memory_Access_Query_DM'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The userid or list of userids being queried for restricted access to the specified segment.

memory_segment_name_length
(int4) Length of *memory_segment_name*.

memory_segment_name
(string,1-8,char42) The name of the memory segment being queried.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

name_array_length
(int4) Length of *name_array*.

name_array
(array) An array consisting of zero or more instances of *name_structure*, as follows:

name_structure
(structure) A structure consisting of one set of *name_length* and *name*, as follows:

name_length
(int4) Length of *name*.

name
(string,1-8,char42) A userid.

Usage Notes

1. If *target_identifier* is a list, then you may have special considerations for checking authorizations, depending on your directory manager. Refer to your directory manager documentation for more information.
2. Note that while a list name specified for *target_identifier* is generally limited to 64 characters (in the char43 character set) for other APIs, here a list name is limited by the IBM DirMaint directory manager to 8 characters in the char42 character set (meaning that no underscores are allowed). This same restriction applies to Shared_Memory_Access_Add_DM and Shared_Memory_Access_Remove_DM.
3. The shared memory segment specified in *memory_segment_name* does not need to be defined before restricted access is queried.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
		20	RS_NOT_AUTHORIZED	No output; user(s) not authorized for specified segment
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		12	RS_AUTHERR_DM	Request not authorized by directory manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
400	RCERR_IMAGEDEF	4	RS_NOT_FOUND	Image definition not defined
		12	RS_LOCKED	Image definition is locked
424	RCERR_SEGMENT_DM	8	RS_SEG_NAME_NOT_FOUND	Segment name not found
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available
592	RCERR_ASYNC_DM	nnnn	opid	Asynchronous operation started - product-specific asynchronous operation ID (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)

RC	RC Name	RS	RS Name	Description
596	RCERR_INTERNAL_DM	nnnn	psrc	Internal directory manager error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Shared_Memory_Access_Remove_DM

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
memory_segment_name_length
memory_segment_name

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Shared_Memory_Access_Remove_DM to remove restricted (RSTD) access from a shared memory segment.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 30.

function_name
(string,30,char43) The API function name – in this case, 'Shared_Memory_Access_Remove_DM'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See “Client Authentication” on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The userid or list of IDs for which access is being removed.

memory_segment_name_length
(int4) Length of *memory_segment_name*.

memory_segment_name
(string,1-8,char42) The name of the memory segment to which access is being removed.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Usage Notes

1. This function checks the name to determine whether it is a list, and if not, processes the name as a single image name. Therefore, lists should be given names that cannot be confused with image names.
2. During authorization checking and function processing, name lists are only expanded once; although a name within a list may also be the name of a list, the second (nested) list will not be expanded.
3. If *target_identifier* is a list of userids, and
   - Processing for all entries in the list is successful, then RC=0.
   - Processing for any entry in the list is **not** successful, then RC=504 (RCERR_LIST_DM) and RS is set to the position in the list where the error occurred. This is where processing stops. IDs located earlier in the list are processed but no IDs located later in the list are processed.
4. If *target_identifier* is a list, then you may have special considerations for checking authorizations, depending on your directory manager. Refer to your directory manager documentation for more information.
5. Note that while a list name specified for *target_identifier* is generally limited to 64 characters (in the char43 character set) for other APIs, here a list name is limited by the IBM DirMaint directory manager to 8 characters in the char42 character set (meaning that no underscores are allowed). This same restriction applies to Shared_Memory_Access_Add_DM and Shared_Memory_Access_Query_DM.

6. The shared memory segment specified in *memory_segment_name* does not need to be defined before restricted access is removed.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		12	RS_AUTHERR_DM	Request not authorized by directory manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
400	RCERR_IMAGEDEF	4	RS_NOT_FOUND	Image definition not defined
		12	RS_LOCKED	Image definition is locked
424	RCERR_SEGMENT_DM	8	RS_SEG_NAME_NOT_FOUND	Segment name not found
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available
504	RCERR_LIST_DM	nnnn	psrc	Target ID Not Removed
592	RCERR_ASYNC_DM	nnnn	opid	Asynchronous operation started - product-specific asynchronous operation ID (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
596	RCERR_INTERNAL_DM	nnnn	psrc	Internal directory manager error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
600	RCERR_SHSTOR	20	RS_NOT_AUTHORIZED	Not authorized to issue internal system command or is not authorized for RSTD segment
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Shared_Memory_Create

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
memory_segment_name_length
memory_segment_name
begin_page
end_page
page_access_descriptor
memory_attributes
memory_access_identifier_length
memory_access_identifier

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Shared_Memory_Create to create a memory segment that can be shared among virtual images.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 20.

function_name
(string,20,char43) The API function name – in this case, 'Shared_Memory_Create'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The userid for which memory is being saved in the segment.

memory_segment_name_length
(int4) Length of *memory_segment_name*.

memory_segment_name
(string,1-8,char42) The name of the memory segment being created.

begin_page
(int8; range 0-524031) The beginning page to be saved.

end_page
(int8; range 0-524031) The ending page to be saved.

page_access_descriptor
(int1) The type of page access. Valid values are:

1
SW – Shared read/write access.

2
EW – Exclusive read/write access.

3
SR – Shared read-only access.

4
ER – Exclusive read-only access.

5
SN – Shared read/write access, no data saved.

6
EN – Exclusive read/write access, no data saved.

7
SC – Shared read-only access, no data saved, CP writeable pages.

Note:

1. Only exclusive access (**EW, EN, ER**) may be specified when *begin_page* starts in segment zero.
2. Shared read-only access (**SC**) may *not* be specified when *memory_attributes* is set to UNRSTD. (Note that this is the default for *memory_attributes*, so you must specifically set the value to RSTD.)

memory_attributes
(int1) Valid values are:

0
Unspecified

1
RSTD – Restricted saved memory

2
UNRSTD – Unrestricted saved memory. This is the default.

memory_access_identifier_length
(int4) Length of *memory_access_identifier*.

memory_access_identifier
(string,0-8,char42) The name of the image or list of images authorized to access the **RSTD** segment. This parameter is optional. If specified, it is used only when **RSTD** is specified in the *memory_attributes* parameter. See Usage Note "3" on page 496.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Usage Notes

1. This function checks the name to determine whether it is a list, and if not, processes the name as a single image name. Therefore, lists should be given names that cannot be confused with image names.
2. During authorization checking and function processing, name lists are only expanded once; although a name within a list may also be the name of a list, the second (nested) list will not be expanded.
3. This function calls the Shared_Memory_Access functions internally. The optional *memory_access_identifier* parameter adds access to a **RSTD** segment by adding a NAMESAVE statement in the user directory for the specified segment name. The userid of the segment's creator must be in the *memory_access_identifier* list (or be added later with Shared_Memory_Access_Add_DM) in order for that ID to access, purge, or query the segment. Any userid being granted access may have to log off and log back on in order for the new NAMESAVE statement to take effect.
4. The *target_identifier* userids must have CP privileges to save segments in order to complete this function, otherwise RC 600 RS 16 will be returned.
5. This function is *not* supported for shared memory segments above the 2GB memory boundary.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful

RC	RC Name	RS	RS Name	Description
		4	RS_NOT_FOUND	Segment was created or replaced, but specified userid in *memory_access_identifier* could not be found to give RSTD access
		8	RS_OFFLINE	Request successful; object directory offline
		12	RS_NAMESAVE_EXISTS	Request successful; NAMESAVE statement already exists in directory
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
396	RCERR_INTERNAL	nnnn	psrc	Internal system error - product-specific return code (See "Internal Return Codes (RC = 396, 592, or 596)" on page 834)
400	RCERR_IMAGEDEF	4	RS_NOT_FOUND	Image definition not defined
		12	RS_LOCKED	Image definition is locked
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available
504	RCERR_LIST_DM	nnnn	psrc	Target ID not added
592	RCERR_ASYNC_DM	nnnn	opid	Asynchronous operation started - product-specific asynchronous operation ID (See "Internal Return Codes (RC = 396, 592, or 596)" on page 834)
596	RCERR_INTERNAL_DM	nnnn	psrc	Internal directory manager error - product-specific return code (See "Internal Return Codes (RC = 396, 592, or 596)" on page 834)
600	RCERR_SHSTOR	8	RS_BAD_RANGE	Bad page range
		12	RS_NOT_LOGGED_ON	User not logged on
		16	RS_NOSAVE	Could not save segment

RC	RC Name	RS	RS Name	Description
		20	RS_NOT_AUTHORIZED	Not authorized to issue internal system command or is not authorized for RSTD segment
		24	RS_CONFLICTING_PARMS	Conflicting parameters
		28	RS_SEGMENT_NOT_FOUND	Segment not found or does not exist
		299	RS_CLASS_S_ALREADY _DEFINED	Class S (skeleton) segment file already exists
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Shared_Memory_Delete

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
memory_segment_name_length
memory_segment_name

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Shared_Memory_Delete to delete a shared memory segment.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 20.

function_name
(string,20,char43) The API function name – in this case, 'Shared_Memory_Delete'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See “Client Authentication” on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) This must match an entry in the authorization file that also contains the *authenticated_userid* and the *function_name* (Shared_Memory_Delete).

memory_segment_name_length
(int4) Length of *memory_segment_name*.

memory_segment_name
(string,1-8,char42) The name of the memory segment being deleted.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Usage Notes

1. This function calls the Shared_Memory_Access functions internally.
2. During authorization checking and function processing, name lists are only expanded once; although a name within a list may also be the name of a list, the second (nested) list will not be expanded.
3. Shared_Memory_Delete will only purge a DCSS that matches the segment name requested for deletion. An NSS by that name will not be deleted.
4. For RSTD segments, the *authenticated_userid* on this call must match an authorized userid having a NAMESAVE statement in their directory for the specified segment name. The APIs provide Shared_Memory_Access_Add or the *memory_access_identifier* parameter on Shared_Memory_Create and Shared_Memory_Replace calls to add NAMESAVE statements for a user.
5. This function is *not* supported for shared memory segments above the 2GB memory boundary.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful

RC	RC Name	RS	RS Name	Description
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
396	RCERR_INTERNAL	nnnn	psrc	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
400	RCERR_IMAGEDEF	4	RS_NOT_FOUND	Image definition not defined
		16	RS_CANNOT_DELETE	DCSS Segment Could Not Be Deleted
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available
592	RCERR_ASYNC_DM	nnnn	opid	Asynchronous operation started - product-specific asynchronous operation ID (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
596	RCERR_INTERNAL_DM	nnnn	psrc	Internal directory manager error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
600		12	RS_NOT_LOGGED_ON	User not logged on
		20	RS_NOT_AUTHORIZED	Not authorized to issue internal system command or is not authorized for RSTD segment
		28	RS_SEGMENT_NOT_FOUND	Segment not found or does not exist
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Shared_Memory_Query

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
memory_segment_name_length
memory_segment_name

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
memory_segment_array_length
memory_segment_array (1)
  memory_segment_structure (2)
    memory_segment_structure_length
    memory_segment_name_length
    memory_segment_name
    memory_segment_status
    page_range_array_length
    page_range_array (1)
      page_range_structure (2)
        page_range_structure_length
        begin_page
        end_page
        page_access_descriptor

Note:

1. An array consists of zero or more of its components.
2. A structure consists of one set of its components.

Purpose

Use Shared_Memory_Query to query information about system data files that are contained in the saved memory segment.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 19.

function_name
(string,19,char43) The API function name – in this case, 'Shared_Memory_Query'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) This must match an entry in the authorization file that also contains the *authenticated_userid* and the *function_name* (Shared_Memory_Query).

memory_segment_name_length
(int4) Length of *memory_segment_name*.

memory_segment_name
One of the following:

- (string,1-8,char42) The name of the memory segment being queried.
- (string,1,*) Specifies all defined memory segments for query.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

memory_segment_array_length
(int4) Length of *memory_segment_array*.

memory_segment_array
(array) An array consisting of zero or more instances of *memory_segment_structure*, as follows:

memory_segment_structure
(structure) A structure consisting of one set of the following parameters:

memory_segment_structure_length
(int4) The combined length of the remaining parameters in *memory_segment_structure* (not including this parameter).

memory_segment_name_length
(int4) Length of *memory_segment_name*.

memory_segment_name
(string,1-8,char42) The name of a memory segment.

memory_segment_status
(int1) One of the following values:

1
Skeleton

2
Available and nonrestricted

3
Available and restricted

4
Pending purge

page_range_array_length
(int4) Length of *page_range_array*.

page_range_array
(array) An array consisting of zero or more instances of *page_range_structure*, as follows:

page_range_structure
(structure) A structure consisting of one set of the following parameters:

page_range_structure_length
(int4) The combined length of the remaining parameters in *page_range_structure* (not including this parameter).

begin_page
(int8; range 0-524031) The beginning page of the segment.

end_page
(int8; range 0-524031) The ending page of the segment.

page_access_descriptor
(int1) The type of page access, as follows:

1
SW – Shared read/write access.

2
EW – Exclusive read/write access.

3
SR – Shared read-only access.

4
ER – Exclusive read-only access.

5
SN – Shared read/write access, no data saved.

6
EN – Exclusive read/write access, no data saved.

7
SC – Shared read-only access, no data saved, CP writeable pages.

Usage Notes

1. This function is *not* supported for shared memory segments above the 2GB memory boundary.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
		28	RS_SEGMENT_NOT_FOUND	Request successful But Segment Not Found
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
396	RCERR_INTERNAL	nnnn	psrc	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
400	RCERR_IMAGEDEF	4	RS_NOT_FOUND	Image definition not defined
600		12	RS_NOT_LOGGED_ON	User not logged on
		20	RS_NOT_AUTHORIZED	Not authorized to issue internal system command or is not authorized for RSTD segment
		28	RS_SEGMENT_NOT_FOUND	Segment not found or does not exist
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data

RC	RC Name	RS	RS Name	Description
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Shared_Memory_Replace

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
memory_segment_name_length
memory_segment_name
memory_access_identifier_length
memory_access_identifier

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Shared_Memory_Replace to replace a shared memory segment previously defined by Shared_Memory_Create.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 21.

function_name
(string,21,char43) The API function name – in this case, 'Shared_Memory_Replace'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See “Client Authentication” on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The userid for whom the memory is being replaced.

memory_segment_name_length
(int4) Length of *memory_segment_name*.

memory_segment_name
(string,1-8,char42) The name of the memory segment being replaced.

memory_access_identifier_length
(int4) Length of *memory_access_identifier*.

memory_access_identifier
(string,0-8,char42) The image name or the name of a list of new users who have access to the **RSTD** memory segment. See Usage Note "3" on page 508.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Usage Notes

1. This function checks the name to determine whether it is a list, and if not, processes the name as a single image name. Therefore, lists should be given names that cannot be confused with image names.
2. During authorization checking and function processing, name lists are only expanded once; although a name within a list may also be the name of a list, the second (nested) list will not be expanded.
3. This function calls the Shared_Memory_Access functions internally. The optional *memory_access_identifier* parameter adds access to a **RSTD** segment by adding a NAMESAVE statement in the user directory for the specified segment name. The userid of the segment's creator must be in the *memory_access_identifier* list (or be added later with Shared_Memory_Access_Add_DM) in order for that ID to access, purge, or query the segment. Any

userid being granted access may have to log off and log back on in order for the new NAMESAVE statement to take effect.

4. The *target_identifier* userids must have CP privileges to save segments in order to complete this function, otherwise RC 600 RS 16 will be returned.
5. This function is *not* supported for shared memory segments above the 2GB memory boundary.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
		4	RS_NOT_FOUND	Segment was created or replaced, but specified userid in *memory_access_identifier* could not be found to give RSTD access
		8	RS_OFFLINE	Request successful; object directory offline
		12	RS_NAMESAVE_EXISTS	Request successful; NAMESAVE statement already exists in directory
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
396	RCERR_INTERNAL	nnnn	psrc	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
400	RCERR_IMAGEDEF	4	RS_NOT_FOUND	Image definition not defined
		12	RS_LOCKED	Image definition is locked
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available
504	RCERR_LIST_DM	nnnn	psrc	Target ID not added
592	RCERR_ASYNC_DM	nnnn	opid	Asynchronous operation started - product-specific asynchronous operation ID (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)

RC	RC Name	RS	RS Name	Description
596	RCERR_INTERNAL_DM	nnnn	psrc	Internal directory manager error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
600		12	RS_NOT_LOGGED_ON	User not logged on
		16	RS_NOSAVE	Could not save segment
		20	RS_NOT_AUTHORIZED	Not authorized to issue internal system command or is not authorized for RSTD segment
		28	RS_SEGMENT_NOT_FOUND	Segment not found or does not exist
		299	RS_CLASS_S_ALREADY _DEFINED	Class S (skeleton) segment file already exists
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

SMAPI_Status_Capture

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use SMAPI_Status_Capture to capture data to assist with identification and resolution of a problem with the SMAPI servers.

You can use the stand-alone SMSTATUS EXEC to perform this same function when SMAPI_Status_Capture cannot be executed because SMAPI is not responsive. For more information, see Appendix G, “Capturing SMAPI Data for Problem Resolution,” on page 883.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 20.

function_name
(string,13,char43) The API function name – in this case, 'SMAPI_Status_Capture'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See “Client Authentication” on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) This must match an entry in the authorization file that also contains the *authenticated_userid* and the *function_name* (SMAPI_Status_Capture).

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Usage Notes

1. This API will capture the status of several system settings, as well as copies of the SMAPI-related log files and SMAPI server console logs. The files will be compressed into a single file, SMSTATUS *xxxxxxx* (where *xxxxxxx* is a unique identifier), which will be saved in COPYFILE PACKED format and placed in the directory VMSYS:VSMWORK1.STATUS, as specified by the `Server_STATUS =` attribute in the DMSSICNF COPY file. SMAPI will retain the *n* most recent output files from invocations of SMSTATUS, where *n* is determined by the `Server_StatusLog_Max =` attribute. See "Configuring SMAPI" on page 30 for more information.
2. The specific data collected by this API will include:
   - The contents of the following:
     - SMAPI logs
     - Console logs from all SMAPI servers
     - * NOTEBOOK files
     - DMSSISVR NAMES file
     - DMSSICNF COPY file
     - VSMWORK1 AUTHLIST
     - DIRECTORY MANAGER CONSOLE LOG
     - SMAPINET files

  - VSMAPI SV%LOG% files
- Output from:
  - CP QUERY CPLEVEL
  - CP QUERY NAMES
  - CP QUERY VMLAN
  - CMS LISTFILE output of files on MAINT's 193 that have filenames starting with DMSRS*, DMSS*, DMSWS*, and VSM*
  - CMS LISTFILE output of VMSYS:VSMWORK1. and VMSYS:VSMWORK1.DATA
  - DMSWSCHK
  - CP QUERY USERID
  - CP QUERY STORAGE
  - CP INDICATE
  - CP QUERY SECUSER ALL
  - CP QUERY SSI
  - CP QUERY PRODUCT
  - CP QUERY LIMITS FOR VMSYS
  - CMS LISTFILE output of files on MAINT's 193 disk

3. Be aware that the console output from some SMAPI servers (such as LOHCOST) may be large, and that the default VMSYS: filepool size may not be sufficient to handle it, causing SMSTATUS to fail. For more information on increasing the filepool size, see *z/VM: CMS File Pool Planning, Administration, and Operation*.
4. SMAPI automatically issues SMAPI_Status_Capture when an internal error is detected.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
8	RC_ERR	14	RS_FREE_MODE_NOT_AVAIL	Free modes not available
		3015	RS_FILE_SAVE_ERROR	File could not be saved
24	RCERR_SYNTAX	*pprr*	*pprr*	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
396	RCERR_INTERNAL	*nnnn*	*psrc*	Internal system error - product-specific return code (See "Internal Return Codes (RC = 396, 592, or 596)" on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist

RC	RC Name	RS	RS Name	Description
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

SSI_Query

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
ssi_name
ssi_mode
cross_system_timeouts
ssi_pdr
ssi_info_array (1)
ssi_info_structure (2)
member_slot
member_system_id
member_state
member_pdr_heartbeat
member_received_heartbeat

Note:

1. An array consists of zero or more of its components.
2. A structure consists of one set of its components.

Purpose

Use SSI_Query to obtain the SSI and system status.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 9.

function_name
(string,9,char43) The API function name – in this case, 'SSI_Query'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) This must match an entry in the authorization file that also contains the *authenticated_userid* and the *function_name* (SSI_Query).

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

The following output parameters, each followed by a by a null (ASCIIZ) character, are returned if RC=0.

ssi_name
(string,1-8,char42) The name of SSI cluster.

ssi_mode
(string,4-6,char26) One of the following, indicating this member's view of the SSI mode:

Stable
All IPLed systems in the SSI cluster are joined and participating in all SSI services.

Influx
A member is in a transition state.

Safe
A member is in an unknown state.

cross_system_timeouts
(string,7-8,char26) Indicates the status of cross-system timeouts. Possible values are:

Enabled

Disabled

ssi_pdr
(string,6-14,char42) The SSI persistent data record device, returned in this format:

```
volid_on_rdev
```

where *volid* is the volume label, and *rdev* is the real device address of the device that contains the SSI persistent data record.

ssi_info_array
(array) An array consisting of zero or more instances of *ssi_info_structure*, with each structure terminated by a null (ASCIIZ) character, as follows:

ssi_info_structure
(structure) A structure consisting of one set of the following parameters, with a blank separating each parameter:

member_slot
(string,1,char10) Indicates the slot number in the SSI member list.

member_system_id
(string,1-8,char42) The system identifier of the member occupying that slot.

member_state
(string,4-9,char26) The issuing member's view of the state of the associated member. Valid states are:

Down
A member is in the down state when any of the following are true:

- It has not been IPLed as a member of the SSI cluster.
- It has left the SSI cluster due to a system shutdown or abend.
- It has not attempted to join the SSI cluster after an IPL.
- It has been declared down by use of the SET SSI command.

Joining
A member is in joining state when it is in the process of joining an SSI cluster that already has one or more joined members. Only one member can be in joining state at a time.

Joined
A member is in joined state when it has successfully joined the SSI cluster and is participating in SSI-wide operations.

Leaving
A member is in leaving state when it was joined to an SSI cluster and is now shutting down.

Isolated
A member is in isolated state when it cannot join the SSI cluster due to a failure in the enablement of SSI-wide operations, or due to a failure occurring while attempting to join the SSI cluster.

Suspended
A member is in suspended state when it does not have connectivity to another member in the SSI cluster that is in a state other than down or isolated.

Unknown
A remote member is in unknown state when the connectivity from the local member is lost.

member_pdr_heartbeat
(string,19,char43 plus /) The timestamp (in the local member's time zone) of the heartbeat in the SSI persistent data record for the specified member. The record is returned in the following format:

```
mm/dd/yyyy_hh:mm:ss
```

member_received_heartbeat
(string,19,char43 plus /) The timestamp (in the local member's time zone) of the last received heartbeat from the specified member. The record is returned in the following format (same format as *member_pdr_heartbeat*):

```
mm/dd/yyyy_hh:mm:ss
```

Usage Notes

1. Syntax errors (RC = 24 and RS = *pprr*) are only applicable to the common input parameters. Syntax checking is *not* performed on the additional input parameters for this API. If a valid parameter is specified multiple times, the last value specified for that parameter will be used. For more information, see “Key-value pair (KVP) input parameters” on page 51.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
4	RC_WNG	3008	RS_NOT_SSI_MEMBER	System is not a member of an SSI cluster
		3009	RS_REPAIR_IPL_PARAM	System was IPLed with the REPAIR IPL parameter
8	RC_ERR	3002	RS_INVALID_PARAMETER	Invalid parameter name
		3003	RS_INVALID_OPERAND	Invalid parameter operand
24	RCERR_SYNTAX	*pprr*	*pprr*	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
396	RCERR_INTERNAL	*nnnn*	*psrc*	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist

RC	RC Name	RS	RS Name	Description
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Static_Image_Changes_Activate_DM

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Static_Image_Changes_Activate_DM to enable changes to the source directory to be made available to virtual images.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 32.

function_name
(string,32,char43) The API function name – in this case, 'Static_Image_Changes_Activate_DM'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See “Client Authentication” on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) This must match an entry in the authorization file that also contains the *authenticated_userid* and the *function_name* (Static_Image_Changes_Activate_DM).

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Usage Notes

1. Although a user's directory may be updated as a result of calling this API, it may still be necessary for the user to log off and back on for any new directory statements to take effect.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_OK	Request successful
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		12	RS_AUTHERR_DM	Request not authorized by directory manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available

RC	RC Name	RS	RS Name	Description
592	RCERR_ASYNC_DM	nnnn	opid	Asynchronous operation started - product-specific asynchronous operation ID (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
596	RCERR_INTERNAL_DM	nnnn	psrc	Internal directory manager error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Static_Image_Changes_Deactivate_DM

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Static_Image_Changes_Deactivate_DM to prevent changes to the source directory from being made available to virtual images.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 34.

function_name
(string,34,char43) The API function name – in this case, 'Static_Image_Changes_Deactivate_DM'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See “Client Authentication” on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) This must match an entry in the authorization file that also contains the *authenticated_userid* and the *function_name* (Static_Image_Changes_Deactivate_DM).

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		12	RS_AUTHERR_DM	Request not authorized by directory manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available
592	RCERR_ASYNC_DM	nnnn	opid	Asynchronous operation started - product-specific asynchronous operation ID (See "Internal Return Codes (RC = 396, 592, or 596)" on page 834)

RC	RC Name	RS	RS Name	Description
596	RCERR_INTERNAL_DM	nnnn	psrc	Internal directory manager error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Static_Image_Changes_Immediate_DM

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Static_Image_Changes_Immediate_DM to make changes to the source directory immediately available to virtual images regardless of the current status of static image changes (active or inactive).

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 33.

function_name
(string,33,char43) The API function name – in this case, 'Static_Image_Changes_Immediate_DM'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See “Client Authentication” on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) This must match an entry in the authorization file that also contains the *authenticated_userid* and the *function_name* (Static_Image_Changes_Immediate_DM).

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Usage Notes

1. Although a user's directory may be updated as a result of calling this API, it may still be necessary for the user to log off and back on for any new directory statements to take effect.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		12	RS_AUTHERR_DM	Request not authorized by directory manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available

RC	RC Name	RS	RS Name	Description
592	RCERR_ASYNC_DM	nnnn	opid	Asynchronous operation started - product-specific asynchronous operation ID (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
596	RCERR_INTERNAL_DM	nnnn	psrc	Internal directory manager error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

System_Config_Syntax_Check

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
`system_config_name=`*value*
`system_config_type=`*value*
`parm_disk_owner=`*value*
`parm_disk_number=`*value*
`parm_disk_password=`*value*

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
system_config_syntax_error_array (1) (error only)
    system_config_syntax_error_record

Note:

1. An array consists of zero or more of its components.

Purpose

Use System_Config_Syntax_Check to check the syntax of a system configuration file located on a system parm disk.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 26.

function_name
(string,26,char43) The API function name – in this case, 'System_Config_Syntax_Check'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See “Client Authentication” on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See “Client Authentication” on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) This must match an entry in the authorization file that also contains the *authenticated_userid* and the *function_name* (System_Config_Syntax_Check).

Note: The format for specifying the following additional input parameters is *parameter_name=value*, followed by a null (ASCIIZ) terminator. They may be specified in any order. For more information, see “Key-value pair (KVP) input parameters” on page 51.

`system_config_name=`*value*
(string,0-8,char42) File name of the system configuration file. The default is set by the "System_Config_File_Name =" statement in the DMSSICNF COPY file. (See the "Default SYSTEM CONFIG Link Values" section in “Configuring SMAPI” on page 30.)

`system_config_type=`*value*
(string,0-8,char42) File type of the system configuration file. The default is set by the "System_Config_File_Type =" statement in the DMSSICNF COPY file. (See the "Default SYSTEM CONFIG Link Values" section in “Configuring SMAPI” on page 30.)

`parm_disk_owner=`*value*
(string,0-8,char42) Owner of the parm disk. The default is set by the "Parm_Disk_Owner =" statement in the DMSSICNF COPY file. (See the "Default SYSTEM CONFIG Link Values" section in “Configuring SMAPI” on page 30.)

`parm_disk_number=`*value*
(string,0-4,char16) Number of the parm disk as defined in the VSMWORK1 directory. (See Usage Notes “3” on page 531 and “4” on page 531.) The default is set by the "Parm_Disk_Number =" statement in the DMSSICNF COPY file. (See the "Default SYSTEM CONFIG Link Values" section in “Configuring SMAPI” on page 30.)

`parm_disk_password=`*value*
(string,0-8,char42) Multiwrite password for the parm disk. The default is "," and should not be changed. Any value other the default is ignored. (See “Configuring SMAPI” on page 30.)

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

If RC=8 and RS=34, the following parameters will be returned:

system_config_syntax_error_array
(array) An array consisting of zero or more instances of *system_config_syntax_error_record*, as follows:

system_config_syntax_error_record
(string) A record containing the error message number and the text of the CPSYNTAX error encountered. Each record is terminated with a null (ASCIIZ) terminator.

Usage Notes

1. Syntax errors (RC=24 and RS=*pprr*) are only applicable to the common input parameters. Syntax checking is *not* performed on the additional input parameter for this API. If a valid parameter is specified multiple times, the last value specified for that parameter will be used. For more information, see “Key-value pair (KVP) input parameters” on page 51.
2. If the system administrator has either changed the default location of the system configuration file or renamed the file, then the input parameters must be used to specify the new file information.
3. Updates for the VSMWORK1 user in the VM directory are required to link and access the CP parm disks. A link option for PMAINT CF0 must be added. If the system administrator changed the default locations of the parm disks, the VSMWORK1 userid must be granted the appropriate authority and links to the new locations.

   The following links are provided in the user directory of VSMWORK1:

   ```
   .
   IDENTITY VSMWORK1 .....
   .
   .
   LINK PMAINT CF0 CF0 MD
   ```

4. If you want a different parm disk, add links to the VSMWORK1 user directory. For example:

   ```
   .
   USER VSMWORK1 .....
   .
   .
   LINK SMAPIC5 C00 FC00 MD
   ```

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
8	RC_ERR	14	RS_FREE_MODE_NOT_AVAIL	Free modes not available
		24	RS_PARM_DISK_LINK_ERR	Error linking parm disk
		32	RS_SYS_CONF_NOT_FOUND	System configuration not found on parm disk

RC	RC Name	RS	RS Name	Description
		34	RS_SYS_CONF_BAD_DATA	System configuration has bad data
		38	RS_CPDISK_MODE_NOT_ AVAIL	CP disk modes not available
		44	RS_PDISK_PW_NOT_ SUPPLIED	No link password for parm disk was provided
		46	RS_PDISK_PW_INCORRECT	Parm disk password is incorrect
		48	RS_PDISK_NOT_IN_SERVER _DIRECTORY	Parm disk is not in server's user directory
		50	RS_CPRELEASE_ERROR	Error with CPRELEASE of parm disk
		3002	RS_INVALID_PARAMETER	Invalid parameter name
24	RCERR_SYNTAX	*pprr*	*pprr*	Syntax error in function parameter
396	RCERR_INTERNAL	*nnnn*	*psrc*	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

System_Compliance_Information_Query

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
system_compliance_information_data_length
system_compliance_information_data

Purpose

Use System_Compliance_Information_Query to obtain information about audit– and secure-configuration–related configuration values available from CP and various software products (like ESMs). All data is extracted in real time when the call is started. No historical data is provided.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 24.

function_name
(string,13,char43) The API function name – in this case, 'System_Compliance_Information_Query'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See “Client Authentication” on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) This must match an entry in the authorization file that also contains the *authenticated_userid* and the *function_name* (System_Compliance_Information_Query).

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

system_compliance_information_data_length
(int4) Length of system_compliance_information_data.

system_compliance_information_data
(string) A sequence of YAML records that are each prefixed by an (int4) length field.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
4	RC_WNG	*nnnn*	RS_ROUTINE_ERROR	Process partially failed. The return code is provided as the reason code. More information may be provided in the "Warnings:" section of the output file.
8	RC_ERR	28	RS_OUTPUT_NOT_VALID	Environment validation failed
8	RC_ERR	*nnnn*	RS_ROUTINE_ERROR	The routine failed. The return code is provided as the reason code.

System_Disk_Accessibility

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
dev_num=*value*

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use System_Disk_Accessibility to verify that the specified device is available to be attached. If RC=0/RS=0 is received, then the device is available.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 25.

function_name
(string,25,char43) The API function name – in this case, 'System_Disk_Accessibility'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) This must match an entry in the authorization file that also contains the *authenticated_userid* and the *function_name* (System_Disk_Accessibility).

Note: The format for specifying the following additional input parameter is *parameter_name=value*, followed by a null (ASCIIZ) terminator. For more information, see "Key-value pair (KVP) input parameters" on page 51.

dev_num=*value*
(string,1-4,char16) The disk device number. This is a required input parameter.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful (the device is available to be attached)
8	RC_ERR	8	RS_DEV_NOT_FOUND	Device does not exist
		10	RS_DEV_NOT_AVAIL_TO _ATTACH	Device is not available to be attached
		3002	RS_INVALID_PARAMETER	Invalid parameter name
		3003	RS_INVALID_OPERAND	Invalid parameter operand
		3004	RS_MISSING_PARAMETER	Required parameter missing
24	RCERR_SYNTAX	*pprr*	*pprr*	Syntax error in function parameter

RC	RC Name	RS	RS Name	Description
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
396	RCERR_INTERNAL	*nnnn*	*psrc*	Internal system error - product-specific return code (See "Internal Return Codes (RC = 396, 592, or 596)" on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

System_Disk_Add

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
dev_num=*value*

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use System_Disk_Add to dynamically add an ECKD disk to a running z/VM system.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 15.

function_name
(string,15,char43) The API function name – in this case, 'System_Disk_Add'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See “Client Authentication” on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The name of the image to which a disk is being added.

Note: The format for specifying the following additional input parameters is *parameter_name=value*, followed by a null (ASCIIZ) terminator. For more information, see "Key-value pair (KVP) input parameters" on page 51.

dev_num=*value*
(string,1-4,char16) The disk device number. This is a required parameter.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Usage Notes

1. This function is used to sense and add a new plugged-in disk for a running z/VM system as an offline disk. The Image_Volume_Add function should be used to format and add the new disk to be used by virtual images to the z/VM system configuration file.
2. Syntax errors (RC = 24 and RS = *pprr*) are only applicable to the common input parameters. Syntax checking is *not* performed on the additional input parameter for this API. If a valid parameter is specified multiple times, the last value specified for that parameter will be used. For more information, see "Key-value pair (KVP) input parameters" on page 51.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
8	RC_ERR	3002	RS_INVALID_PARAMETER	Invalid parameter name
		3003	RS_INVALID_OPERAND	Invalid parameter operand
		3004	RS_MISSING_PARAMETER	Required parameter missing

RC	RC Name	RS	RS Name	Description
24	RCERR_SYNTAX	*pprr*	*pprr*	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
396	RCERR_INTERNAL	*nnnn*	*psrc*	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

System_Disk_IO_Query

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
system_disk_IO_list_length
system_disk_IO_list

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
DASD_information_length
DASD_information
error_data_length (error only)
error_data (error only)

Purpose

Use System_Disk_IO_Query to obtain DASD read and write byte counts for SCSI EDEV and ECKD volumes owned by z/VM, and for which the control units have information. This information will be obtained from DCSS data that has been formatted from CP MONITOR records.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 20.

function_name
(string,13,char43) The API function name – in this case, 'System_Disk_IO_Query'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).

- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See “Client Authentication” on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See “Client Authentication” on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) This must match an entry in the authorization file that also contains the *authenticated_userid* and the *function_name* (System_Disk_IO_Query).

system_disk_IO_list_length
(int4) Length of *system_disk_IO_list*. Zero must be specified if no *system_disk_IO_list* is specified.

system_disk_IO_list
(string,1-maxlength,char36 plus * blank) One of the following:

RDEV=*
Return information for all RDEVs. (This is the default.)

RDEV=*rdev1 rdev2...*
Return information for a blank-delimited list of RDEVs.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

DASD_information_length
(int4) Length of *DASD_information*.

DASD_information
(string) A series of null-terminated strings, each containing the "DASD_IO=" output keyword parameter followed by a series of blank-delimited "*output_subkeyword=value*" pairs for each volume/EDEV, as shown in Table 13 on page 543.

Table 13. Output Keywords and Values for System_Disk_IO_Query

output_keyword_parameter=	**Blank-delimited *output_subkeyword=value* pairs**
DASD_IO=	• TYPE=SCSI | ECKD | NOT_FOUND • RDEV=*rdev* • READ_BYTES=*bytes* (*rate*) • WRITE_BYTES=*bytes* (*rate*) If TYPE=SCSI, then the following path information is also included: • PATH1_READ_BYTES=*bytes* (*rate*) • PATH1_WRITE_BYTES=*bytes* (*rate*) • PATH2_READ_BYTES=*bytes* (*rate*) • PATH2_WRITE_BYTES=*bytes* (*rate*) • PATH3_READ_BYTES=*bytes* (*rate*) • PATH3_WRITE_BYTES=*bytes* (*rate*) • PATH4_READ_BYTES=*bytes* (*rate*) • PATH4_WRITE_BYTES=*bytes* (*rate*) • PATH5_READ_BYTES=*bytes* (*rate*) • PATH5_WRITE_BYTES=*bytes* (*rate*) • PATH6_READ_BYTES=*bytes* (*rate*) • PATH6_WRITE_BYTES=*bytes* (*rate*) • PATH7_READ_BYTES=*bytes* (*rate*) • PATH7_WRITE_BYTES=*bytes* (*rate*) • PATH8_READ_BYTES=*bytes* (*rate*) • PATH8_WRITE_BYTES=*bytes* (*rate*) **Note:** 1. Byte values are in decimal. (Note that these values can wrap.) 2. Rates follow inside parentheses and are the number of blocks changed from previous interval divided by interval size. For ECKD, the blocks are in multiples of 128K. For SCSI, the blocks are usually 512. 3. If information is not available for a DASD, the byte counts will be -1.

If RC=8 and RS=3002, the following parameters will be returned:

error_data_length
(int4) Length of *error_data*.

error_data
(string) "UNKNOWN_PARAMETER_NAMES=", followed by a blank-delimited list of input parameter names that are not valid, then followed by a null terminator.

If RC=8 and RS=3003, the following parameters will be returned:

error_data_length
(int4) Length of *error_data*.

error_data
(string) "INVALID_PARAMETER_NAME_VALUES=", followed by a blank-delimited list of input parameter names that have invalid values specified, followed by a null terminator.

If RC=8 and RS=3004, the following parameters will be returned:

error_data_length
(int4) Length of *error_data*.

error_data
(string) "MISSING PARAMETER=", followed by a blank-delimited list of input parameter names that are missing, then followed by a null terminator.

For all other errors, the following parameters will be returned:

error_data_length
(int4) Length of *error_data*.

error_data
(string) "COMMAND_IN_ERROR=", followed by the command that failed and any accompanying error message and/or return code, then followed by a null terminator.

Usage Notes

1. The *DASD_information_length* and *DASD_information* output parameters are returned only if RC=0, or if RC=8 and RS=8.
2. ECKD volume information is obtained from the control unit (if available). EDEV information starts from the IPL of the zVM system.
3. Syntax errors (RC = 24 and RS = *pprr*) are only applicable to the common input parameters. Syntax checking is *not* performed on the additional input parameters for this API. If a valid parameter is specified multiple times, the last value specified for that parameter will be used. For more information, see “Key-value pair (KVP) input parameters” on page 51.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
8	RC_ERR	8	RS_DEV_NOT_FOUND	Device does not exist
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range
		3002	RS_INVALID_PARAMETER	Invalid parameter name
		3003	RS_INVALID_OPERAND	Invalid parameter operand
		3004	RS_MISSING_PARAMETER	Required parameter missing
		3016	RS_SEGMENT_EMPTY	SMAPIOUT segment empty
		3017	RS_SEGMENT_DATA_INVALID	SMAPIOUT segment does not contain valid data
		3018	RS_SEGMENT_NOT_FOUND	SMAPIOUT segment not found and loaded
24	RCERR_SYNTAX	*pprr*	*pprr*	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager

RC	RC Name	RS	RS Name	Description
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
200	RCERR_IMAGEOP	12	RS_NOT_ACTIVE	Image not active
396	RCERR_INTERNAL	*nnnn*	*psrc*	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

System_Disk_Query

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
dev_num=*value*
disk_size=*value*

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
disk_info_array (1)
    disk_info_structure (2)
        dev_id
        dev_type
        dev_status
        dev_volser
        disk_size

Note:

1. An array consists of zero or more of its components.
2. A structure consists of one set of its components.

Purpose

Use System_Disk_Query to query a real ECKD disk or all real ECKD disks.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 17.

function_name
(string,17,char43) The API function name – in this case, 'System_Disk_Query'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) This must match an entry in the authorization file that also contains the *authenticated_userid* and the *function_name* (System_Disk_Query).

Note: The format for specifying the following additional input parameters is *parameter_name*=*value*, followed by a null (ASCIIZ) terminator. For more information, see "Key-value pair (KVP) input parameters" on page 51.

dev_num=*value*
(string,1-4,char36) The device number, or ALL. This is a required parameter.

disk_size=*value*
(string,0-3,char26) One of the following:

YES
Indicates that the output of this query should include the disk size, in cylinders, for each ECKD DASD.

NO
Indicates that no disk size information should be returned. This is the default.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

disk_info_array
(array) An array consisting of zero or more instances of *disk_info_structure*, with each structure terminated by a null (ASCIIZ) character, as follows:

disk_info_structure
(structure) A structure consisting of one set of the following parameters, with a blank separating each parameter:

dev_id
(string,4,char16) The device number.

dev_type
(string,7,char17) The device type.

dev_status
(string,1-8,char42) The following values are possible:

FREE
Indicates a free device.

OFFLINE
Indicates an offline device.

SYSTEM
Indicates the device is used as users' minidisks.

OWNED
Indicates the device is used by the system for paging and spooling activity.

userid
Userid to which the DASD is attached.

dev_volser
(string,0-6,char36) The device volume serial number.

disk_size
(string,1-8,char10) The size of the disk (in cylinders).

Note that this value is returned only if `disk_size`=YES was specified.

Usage Notes

1. If the device status is OFFLINE, the *dev_volser* output field may not be specified.
2. Syntax errors (RC = 24 and RS = *pprr*) are only applicable to the common input parameters. Syntax checking is *not* performed on the additional input parameters for this API. If a valid parameter is specified multiple times, the last value specified for that parameter will be used. For more information, see “Key-value pair (KVP) input parameters” on page 51.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
4	RC_WNG	28	RS_EMPTY	Return buffer is empty
8	RC_ERR	8	RS_DEV_NOT_FOUND	Device does not exist
		3002	RS_INVALID_PARAMETER	Invalid parameter name
		3003	RS_INVALID_OPERAND	Invalid parameter operand
		3004	RS_MISSING_PARAMETER	Required parameter missing

RC	RC Name	RS	RS Name	Description
24	RCERR_SYNTAX	*pprr*	*pprr*	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
396	RCERR_INTERNAL	*nnnn*	*psrc*	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

System_EQID_Query

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
`eqid_for`=*value*
`eqid_target`=*value*

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
eqid_array_length
eqid_array (1)
eqid_structure (if `eqid_for=EQID`) (2)
eqid_name
eqid_rdev
eqid_structure (if `eqid_for=ALL` or `eqid_for=RDEV`) (2)
eqid_rdev
eqid_name
error_data_length (error only)
error_data (error only)

Note:

1. An array consists of zero or more of its components.
2. A structure consists of one set of its components.

Purpose

Use System_EQID_Query to obtain a list of system devices assigned a device equivalency ID.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 17.

function_name
(string,13,char43) The API function name – in this case, 'System_EQID_Query'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) This must match an entry in the authorization file that also contains the *authenticated_userid* and the *function_name* (System_EQID_Query).

Note: The format for specifying the following additional input parameters is *parameter_name=value*, followed by a null (ASCIIZ) terminator. For more information, see "Key-value pair (KVP) input parameters" on page 51.

`eqid_for=`*value*
(string,3-4,char26) One of the following:

EQID
Returns all RDEVs that have an EQID equal to the value specified by `eqid_target=`.

ALL
Returns all RDEVs that have been assigned a user-defined EQID, along with the EQIDs for those RDEVs.

RDEV
Returns the EQIDs for the RDEVs within the range specified by `eqid_target=`.

This is a required parameter.

`eqid_target=`*value*
One of the following must be specified if `eqid_for=EQID` or `eqid_for=RDEV`:

eqid_name
(string,1-maxlength,char36) A string of 1-8 alphanumeric characters for a user-defined EQID, or a string of 50 alphanumeric characters plus a dash ("-") for a system-generated EQID. Multiple EQID names may be specified, separated by blanks.

eqid_rdev
(string,1-maxlength,char37) A single RDEV, a range of RDEVs, or a series of both. Only RDEVs that have an EQID (either system-generated or user-defined) are returned. RDEVs that do not exist or have no EQID are ignored.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

eqid_array_length
(int4) Length of *eqid_array*.

eqid_array
(array) An array consisting of zero or more instances of *eqid_structure*, with each structure terminated by a null (ASCIIZ) character, as follows:

eqid_structure
(structure) A structure consisting of one set of the following parameters, with a blank separating each parameter:

If `eqid_for=EQID`:

eqid_name
(string,1-50,char26) The EQID name.

eqid_rdev
(string,1-maxlength,char16) One or more RDEVs associated with the *eqid_name*. Each RDEV is blank-delimited.

If `eqid_for=ALL` or `eqid_for=RDEV`:

eqid_rdev
(string,1-4,char16 plus -) The RDEV of the device with an associated EQID.

eqid_name
(string,1-50,char36) The EQID associated with the *eqid_rdev*.

If RC=8 and RS=3002, the following parameters will be returned:

error_data_length
(int4) Length of *error_data*.

error_data
(string) "UNKNOWN_PARAMETER_NAMES=", followed by the first parameter name that is not valid, followed by a null terminator.

If RC=8 and RS=3003, the following parameters will be returned:

error_data_length
(int4) Length of *error_data*.

error_data
(string) "UNKNOWN_PARAMETER_OPERAND=", followed by the first parameter value that is not valid, followed by a null terminator.

Usage Notes

1. The *eqid_array_length* and *eqid_array* output parameters are returned only if RC=0.
2. If `eqid_target=`*value* is specified when `eqid_for=ALL`, then `eqid_target=`*value* will be ignored.
3. Syntax errors (RC = 24 and RS = *pprr*) are only applicable to the common input parameters. Syntax checking is *not* performed on the additional input parameters for this API. If a valid parameter is specified multiple times, the last value specified for that parameter will be used. For more information, see "Key-value pair (KVP) input parameters" on page 51.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
4	RC_WNG	8	RS_NOT_EXIST	No device EQIDs found
8	RC_ERR	3002	RS_INVALID_PARAMETER	Invalid parameter name
		3003	RS_INVALID_OPERAND	Invalid parameter operand
		3004	RS_MISSING_PARAMETER	Required parameter missing
24	RCERR_SYNTAX	*pprr*	*pprr*	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
396	RCERR_INTERNAL	*nnnn*	*psrc*	Internal system error - product-specific return code (See "Internal Return Codes (RC = 396, 592, or 596)" on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

System_FCP_EQID_Set

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
`rdev=`*value*
`eqid=`*value*
`persist=`*value*

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
error_data_length (error only)
error_data (error only)

Purpose

Use System_FCP_EQID_Set to set the EQID (device equivalency ID) for a specific FCP device.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 19.

function_name
(string,13,char43) The API function name – in this case, 'System_FCP_EQID_Set'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) This must match an entry in the authorization file that also contains the *authenticated_userid* and the *function_name* (System_FCP_EQID_Set).

Note: The format for specifying the following additional input parameters is *parameter_name=value*, followed by a null (ASCIIZ) terminator. For more information, see "Key-value pair (KVP) input parameters" on page 51.

rdev=*value*
(string,1-4,char16) The real device address of the FCP for which the EQID is being set. This is a required parameter.

eqid=*value*
One of the following:

- (string,1-8,char36) The EQID to be set for the FCP device
- (string,6,char26) `NOEQID` removes a previously assigned EQID from this RDEV and reverts back to a system-generated EQID.

This is a required parameter.

persist=*value*
(string,0-3,char26) This can be one of the following values:

NO
The FCP device EQID is updated on the active system but is *not* updated in the permanent configuration for the system.

YES
The FCP device EQID is updated on the active system and in the permanent configuration for the system.

If not specified, the default is NO.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

If RC=8 and RS=3002, the following parameters will be returned:

error_data_length

(int4) Length of *error_data*.

error_data

(string) "INVALID_PARAMETER_NAME=", followed by the first parameter name that is not valid, followed by a null terminator.

If RC=8 and RS=3003, the following parameters will be returned:

error_data_length
(int4) Length of *error_data*.

error_data
(string) "INVALID_PARAMETER_VALUE=", followed by the first parameter value that is not valid, followed by a null terminator.

Usage Notes

1. Syntax errors (RC = 24 and RS = *pprr*) are only applicable to the common input parameters. Syntax checking is *not* performed on the additional input parameters for this API. If a valid parameter is specified multiple times, the last value specified for that parameter will be used. For more information, see “Key-value pair (KVP) input parameters” on page 51.
2. If the EQID is successfully set for an FCP device by this API, that FCP device will be in an ONLINE status when this API completes.
3. If the EQID is successfully updated, but the subsequent VARY ON of the FCP device fails, a warning (RC_WNG) is returned.
4. NOEQID removes a previously assigned EQID from the RDEV and reverts back to a system-generated EQID. The device will have no EQID if the device cannot be varied online. Once the device is varied online, a system-generated EQID is assigned.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
4	RC_WNG	8	RS_NOT_EXIST	Specified device not found
		3034	RS_VARY_ON_FAILED	Vary on device failed
8	RC_ERR	28	RS_DEV_INCOMPATIBLE	Device incompatible (not FCP)
		3002	RS_INVALID_PARAMETER	Invalid parameter name
		3003	RS_INVALID_OPERAND	Invalid parameter operand
		3004	RS_MISSING_PARAMETER	Required parameter missing
		3033	RS_VARY_OFF_FAILED	Vary off device failed
		3035	RS_EQID_SET_FAILED	Error setting EQID
24	RCERR_SYNTAX	*pprr*	*pprr*	Syntax error in function parameter

RC	RC Name	RS	RS Name	Description
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
396	RCERR_INTERNAL	*nnnn*	*psrc*	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

System_FCP_Free_Query

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
`fcp_dev=`*value*

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
fcp_array (1)
    fcp_structure (2)
        fcp_dev
        wwpn
        lun
        uuid
        vendor
        prod
        model
        serial
        code
        blk_size
        diskblks
        lun_size

Note:

1. An array consists of zero or more of its components.
2. A structure consists of one set of its components.

Purpose

Use System_FCP_Query to query free FCP disk information.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 21.

function_name
(string,21,char43) The API function name – in this case, 'System_FCP_Free_Query'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) This must match an entry in the authorization file that also contains the *authenticated_userid* and the *function_name* (System_FCP_Free_Query).

Note: The format for specifying the following additional input parameter is *parameter_name*=*value*, followed by a null (ASCIIZ) terminator. For more information, see "Key-value pair (KVP) input parameters" on page 51.

`fcp_dev`=*value*
(string,1-4,char16) The FCP device number. This is a required parameter.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

fcp_array
(array) An array consisting of zero or more instances of *fcp_structure*, with each structure terminated by a null (ASCIIZ) character, as follows:

fcp_structure
(structure) A structure consisting of one set of the following parameters, with a semicolon separating each parameter:

fcp_dev
(string,4,char16) FCP device number.

wwpn
(string,16,char16) World wide port number.

lun
(string,16,char16) Logical unit number.

uuid
(string,32-64,char16) Universally unique number in printed hex.

vendor
(string,1-8,char42) Vendor name.

prod
(string,1-4,char10) Product number.

model
(string,1-4,char10) Model number.

serial
(string,1-8,char10) Serial number.

code
(string,1-4,char10) Device code.

blk_size
(string,1-10,char10) Block size, in bytes.

diskblks
(string,1-10,char10) Number of blocks residing on the logical unit.

lun_size
(string,1-20,char10) Number of bytes residing on the logical unitt.

Usage Notes

1. Syntax errors (RC = 24 and RS = *pprr*) are only applicable to the common input parameters. Syntax checking is *not* performed on the additional input parameters for this API. If a valid parameter is specified multiple times, the last value specified for that parameter will be used. For more information, see “Key-value pair (KVP) input parameters” on page 51.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
8	RC_ERR	8	RS_DEV_NOT_FOUND	Device does not exist
		3002	RS_INVALID_PARAMETER	Invalid parameter name
		3003	RS_INVALID_OPERAND	Invalid parameter operand
		3004	RS_MISSING_PARAMETER	Required parameter missing

RC	RC Name	RS	RS Name	Description
24	RCERR_SYNTAX	*pprr*	*pprr*	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
396	RCERR_INTERNAL	*nnnn*	*psrc*	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

System_Image_Performance_Query

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use System_Image_Performance_Query to obtain virtual machine performance data.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 18.

function_name
(string,18,char43) The API function name – in this case, 'System_Image_Performance_Query'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The user ID under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The user ID under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See “Client Authentication” on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
One of the following:

- (string,1-8,char42) The name of the image being queried.
- (string,1-64,char43) The name of a list containing names of images to be queried.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

image_performance_array
Array of image_performance_structure as follows. This parameter is outputted only if return code=0 or return code=4.

- 4-byte integer number of entries.
- 4-byte integer length of entry data.
- Entry data (in the format described by DIAGNOSE code X'2FC').

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
4	RC_WNG	4	RS_NOT_FOUND	At least one of the specified target IDs did not have performance data to return
		10	RS_TOO_MANY_PARM	At least one extra parameter was specified. All such parameters are ignored.
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server

RC	RC Name	RS	RS Name	Description
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; user ID or password not valid
396	RCERR_INTERNAL	nnnn	psrc	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

System_Information_Query

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
system_information_data_length
system_information_data

Purpose

Use System_Information_Query to obtain information about a CP instance, including time, storage, system level, IPL time, system generation time, language, CPU ID, and CPU capability information. (Note that some capability information may not be available due to hardware dependency. A zero will be returned in this case).

See the following commands in *z/VM: CP Commands and Utilities Reference* and *z/VM: CMS Commands and Utilities Reference* for more information on the specific details of the returned information:

- QUERY CPLEVEL ISO
- QUERY TIMEZONE
- QUERY STORAGE
- QUERY TIME ISO
- QUERY CMSLEVEL
- QUERY CPLANGUAGE
- QUERY CAPABILITY
- QUERY LANGUAGE

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 24.

function_name
(string,13,char43) The API function name – in this case, 'System_Information_Query'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) This must match an entry in the authorization file that also contains the *authenticated_userid* and the *function_name* (System_Information_Query).

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

system_information_data_length
(int4) Length of *system_information_data*.

system_information_data
(string) A series of null-terminated strings, each containing "*output_keyword_parameter*=" followed by a series of blank-delimited "*output_subkeyword=value*" pairs, as shown in Table 14 on page 567. See the corresponding commands in *z/VM: CP Commands and Utilities Reference* or *z/VM: CMS Commands and Utilities Reference* for more information on the specific details returned with each *output_subkeyword=value* pair.

Table 14. Output Keywords and Values for System_Information_Query

output_keyword_parameter=	**Blank-delimited *output_subkeyword=value* pairs**
CMS_LEVEL=	• LEVEL=*nn* • SERVICE_LEVEL=*nnn* • LANGUAGE=*langid*
CPU_CAPABILITY=	• PRIMARY=*pppppppp* • SECONDARY=*ssssssss* • NOMINAL=*0* | *nnnnnnnn* • ADJUSTMENT_INDICATION=0 | *cai* • CHANGE_REASON=0 | *ccr* • CHANGE_EXPLANATION_LENGTH=*nnnn* (length of *text* in CHANGE_EXPLANATION="*text*") • CHANGE_EXPLANATION="" | "*text*" **Note:** 1. 0 indicates information is not available for NOMINAL, ADJUSTMENT_INDICATION, and CHANGE_REASON. 2. "" indicates no CHANGE_EXPLANATION is available.
CPUID=	• ID=*aasssssscccddddd*
CP_LEVEL=	• VERSION=*v.r.m* • SERVICE_LEVEL=*nnnn* • GENERATION_TIME=*hh*:*mm*:*ss* • GENERATION_DATE=*yyyy-mm-dd* (or in the format configured for the system) • GENERATION_TIME_ZONE=*zone* • IPL_TIME=*hh*:*mm*:*ss* • IPL_DATE=*yyyy-mm-dd* (or in the format configured for the system) • IPL_TIME_ZONE=*zone* • LANGUAGE=*langid*
STORAGE=	• ONLINE=*nn* • CONFIGURED=*nn* • INCREMENT=*nn* • STANDBY=*nn* • RESERVED=0 | *nn*

Table 14. Output Keywords and Values for System_Information_Query (continued)	
output_keyword_parameter=	**Blank-delimited *output_subkeyword=value* pairs**
TIME=	• TIMEZONE=*zone* • TIME=*hh*:*mm*:*ss* • DATE=*yyyy-mm-dd* (or in the format configured for the system) • CONNECT=*hh*:*mm*:*ss* • VIRTUAL_CPU=*mm*:*ss*:*hh* • TOTAL_CPU=*mm*:*ss*:*hh*

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
24	RCERR_SYNTAX	*pprr*	*pprr*	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
396	RCERR_INTERNAL	*nnnn*	*psrc*	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

System_Page_Utilization_Query

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
page_file_structure (2)
- *page_file_structure_length*
- *total_paging_pages*
- *total_paging_pages_in_use*
- *total_paging_percent_used*
- *paging_volume_array* (1)
  - *paging_volume_structure* (2)
    - *volid*
    - *rdev*
    - *total_pages*
    - *pages_in_use*
    - *percent_used*
    - *drained*

Note:

1. An array consists of zero or more of its components.
2. A structure consists of one set of its components.

Purpose

Use System_Page_Utilization_Query to obtain information about the z/VM paging space defined on the system.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 29.

function_name
(string,13,char43) The API function name – in this case, 'System_Page_Utilization_Query'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) This must match an entry in the authorization file that also contains the *authenticated_userid* and the *function_name* (System_Page_Utilization_Query).

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

page_file_structure
(structure) A structure consisting of one set of the following parameters, with a blank separating each parameter:

page_file_structure_length
(int4) The combined length of the remaining parameters in *page_file_structure* (not including this parameter).

total_paging_pages
(string,1-*,char10 plus 'K') The total number of pages allocated for paging use on the system.

total_paging_pages_in_use
(string,1-*,char10) The total number of pages in use for paging on the system.

total_paging_percent_used
(string,1-3,char10) The percentage of the available paging space currently in use on the system.

paging_volume_array
(array) An array consisting of zero or more instances of *paging_volume_structure*, with each structure terminated by a null (ASCIIZ) character, as follows:

paging_volume_structure
(structure) A structure consisting of one set of the following parameters, with a blank separating each parameter:

volid
(string,1-6,char42) The volume ID of the page volume.

rdev
(string,1-4,char16) The RDEV of the page volume.

total_pages
(string,1-*,char10) The total number of pages on the volume available for paging use.

pages_in_use
(string,1-*,char10) The total number pages in use on the volume for page files.

percent_used
(string,1-3,char10) The percentage of the available page space on the volume in use.

drained
(string,7-10,char26) One of the following:

NOTDRAINED
This paging space is *not* drained and CP is allocating new page space to this volume.

DRAINED
This paging space is drained and CP is not allocating new page space to this volume.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
24	RCERR_SYNTAX	*pprr*	*pprr*	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
396	RCERR_INTERNAL	*nnnn*	*psrc*	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)

RC	RC Name	RS	RS Name	Description
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

System_Performance_Information_Query

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
system_performance_information_list_length
system_performance_information_list

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
system_performance_information_data_length
system_performance_information_data
error_data_length (error only)
error_data (error only)

Purpose

Use System_Performance_Information_Query to gather hypervisor performance data, including available/used, processor number, total processor percentages, and optional detailed CPU information for all visible LPARs on the CEC.

This API allows users to query, set, or stop the monitor rate, and to set the interval value. The data is returned from CP QUERY FRAMES, CP INDICATE LOAD, CP QUERY MONITOR RATE, and monitor data for CPUs.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 35.

function_name
(string,13,char43) The API function name – in this case, 'System_Performance_Information_Query'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) This must match an entry in the authorization file that also contains the *authenticated_userid* and the *function_name* (System_Performance_Information_Query).

system_performance_information_list_length
(int4) Length of *system_performance_information_list*. Zero must be specified if no *system_performance_information_list* is specified.

system_performance_information_list
(string,0-maxlength,charNA) A series of null-terminated strings, each containing "*keyword_parameter*=" followed by a series of blank-delimited "*subkeyword=value*" pairs, as shown in Table 15 on page 574.

Table 15. Input Keywords and Values for System_Performance_Information_Query

keyword_parameter=	**Blank-delimited *subkeyword=value* pairs**
MONITOR_RATE=	One or both of the following can be specified: • QUERY=YES | NO (YES is the default) • SET=STOP | *nn* (in seconds) **Note:** 1. The monitor rate is how often the CP system data is written to the *MONITOR stream. 2. The QUERY option will display the current monitor rate *before* setting a new value. 3. The valid range for the monitor rate is from .01 to 30 seconds. Note, however, that no more than two digits may be specified after the decimal point. Leading zeros are not required, and trailing zeros are not required after the decimal point.

Table 15. Input Keywords and Values for System_Performance_Information_Query (continued)	
***keyword_parameter*=**	**Blank-delimited *subkeyword*=*value* pairs**
MONITOR_INTERVAL=	One or both of the following can be specified: • QUERY=YES | NO (YES is the default) • SET=*nn*SECONDS | *nn*MINUTES | *nn* (MINUTES is the default) **Note:** 1. The monitor interval is how often the CP *MONITOR data is analyzed. 2. The QUERY option will display the current monitor interval *before* setting a new value. 3. The valid range the monitor interval is 6-3600 seconds, or 1-60 minutes. In both cases, you can only use whole numbers. The default is 1 minute. The monitor interval must always be greater than the monitor rate.
MONITOR_EVENT=	• QUERY=YES | NO (YES is the default) • SET=ENABLE | DISABLE (**Required**, along with DOMAIN=, when enabling or disabling a domain) • DOMAIN= ALL | APPLDATA_ALL | I/O_ALL | ISFC | NETWORK | PROCESSOR | SCHEDULER_ALL | SEEKS_ALL | SSI | STORAGE | USER_ALL (**Required**, along with SET=, when enabling or disabling a domain) **Note:** The QUERY option will display the current monitor events *before* setting any new values.
MONITOR_SAMPLING=	• SET=ENABLE | DISABLE • DOMAIN=ALL (ALL is the default if DOMAIN is not specified)
DETAILED_CPU=	• SHOW=YES | NO (NO is the default. YES returns much more detailed CPU information, based on the specified monitor rate and interval.)

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

system_performance_information_data_length
(int4) Length of *system_performance_information_data*.

system_performance_information_data
(string) A series of null-terminated strings, each containing "*output_keyword_parameter*=" followed by either a *value* or a series of blank-delimited or "*output_subkeyword*=*value*" pairs, as shown in Table 16 on page 576.

Table 16. Output Keywords and Values for System_Performance_Information_Query

***output_keyword_parameter*=**	***value* or blank-delimited *output_subkeyword*=*value* pairs**
These values will always be returned:	
CPU_COUNT=	*nnnn*
CPU_AVERAGE_USE=	*nn*%
PAGING_RATE=	*nnn* (pages read and written per second)
MEMORY_IN_USE=	*nnn* (4096 byte pages)
MEMORY_TOTAL=	*nnn* (4096 byte pages)
This value will be returned only if MONITOR_RATE=QUERY=YES was specified:	
MONITOR_RATE=	*nnn* SECONDS (PENDING *nnn* SECONDS) (if any pending changes)
This value will be returned only if MONITOR_INTERVAL=QUERY=YES was specified:	
MONITOR_INTERVAL=	*nnn* MINUTES | SECONDS (PENDING nnn MINUTES | SECONDS) (if any pending changes)
These values will be returned only if MONITOR_EVENT=QUERY=YES was specified:	
MONITOR_EVENT_COUNT=	*nnn* (a count of the following DOMAIN=*domainname* event records)
DOMAIN=*domainname* (can be MONITOR, PROCESSOR, STORAGE, SCHEDULER, SEEKS, USER, I/O, NETWORK, ISFC, APPLDATA, SSI, etc.)	ENABLED | DISABLED *additional_data_list* (optional) *additional_data_list* can be one of the following: • EXCEPT_USERS | DEVICES *list* • ENABLED_USERS | DEVICES *list*
This value will be returned only if DETAILED_CPU=SHOW=YES was specified:	
MY_LPAR_NAME=	*lparname* (the active LPAR name on which zVM is running)
DETAILED_CPU_COUNT=	*nnn* The number of null-terminated DETAILED_CPU= records that follow below.

Table 16. Output Keywords and Values for System_Performance_Information_Query (continued)	
output_keyword_parameter=	***value* or blank-delimited *output_subkeyword=value* pairs**
DETAILED_CPU=	Each record will contain all of the *output_subkeyword=value* pairs below, separated by blanks, then followed by a null terminator. Note that percentage values will use two decimal places (for example, 5.25%). See Usage Note "1" on page 578 for the list of SEGTPRC DSECT field names that correspond to these *output_subkeyword=value* pairs. • LPAR_NAME=*name* • LPAR_NUMBER=*nnnn* • LPAR_ID=*nnnn* • LPAR_CPU_COUNT=*nnnn* • LPAR_CAP=YES | NO • LPAR_WEIGHT=*nnnn* • LPAR_WAIT=YES|NO • LPAR_LOAD=*nnnn* • LPAR_STATUS=ACTIVE | INACTIVE • LPAR_OVERHEAD=*nnnn* • CPU_TYPE=UNKNOWN | CP | ICF | IFL | ZIIP | ZAPP | OTHER • CPU_ID=*nnnn* • CPU_SUSPEND_TIME=*nn*% • CPU_LP_OVERHEAD_TIME=*nn*% • CPU_BUSY_TIME=*nn*% • PHYSICAL_CPU_BUSY=*nn*% • LOGICAL_CPU_LOAD=*nn*% • VM_CPU_LOAD=*nn*%

If RC=8 and RS=3002, the following parameters will be returned:

error_data_length
(int4) Length of *error_data*.

error_data
(string) "UNKNOWN_PARAMETER_NAMES=", followed by a blank-delimited list of input parameter names that are not valid, then followed by a null terminator.

If RC=8 and RS=3003, the following parameters will be returned:

error_data_length
(int4) Length of *error_data*.

error_data
(string) "INVALID_PARAMETER_NAME_VALUES=", followed by a blank-delimited list of input parameter names that have invalid values specified, followed by a null terminator.

If RC=8 and RS=3004, the following parameters will be returned:

error_data_length
(int4) Length of *error_data*.

error_data
(string) "MISSING PARAMETER=", followed by a blank-delimited list of input parameter names that are missing, then followed by a null terminator.

For all other errors, the following parameters will be returned:

error_data_length
(int4) Length of *error_data*.

error_data
(string) "COMMAND_IN_ERROR=", followed by the command that failed and any accompanying error message and/or return code, then followed by a null terminator.

Usage Notes

1. The DETAILED_CPU=*output_subkeyword=value* pairs correspond to the fields in SEGTPRC DSECT, as shown in Table 17 on page 578.

Table 17. SEGTPRC DSECT field names corresponding to System_Performance_Information_Query DETAILED_CPU=output_subkeyword=value pairs

output_subkeyword=value	**SEGTPRC DSECT field name**
LPAR_NAME=*name*	LPENAME
LPAR_NUMBER=*nnnn*	LPENUM
LPAR_ID=*nnnn*	LPENUPID
LPAR_CPU_COUNT=*nnnn*	LPENLPCT
LPAR_CAP=YES | NO	LPENCAP
LPAR_WEIGHT=*nnnn*	LPENWGHT
LPAR_WAIT=YES|NO	LPENWAIT
LPAR_LOAD=*nnnn*	LPENLOAD
LPAR_STATUS=ACTIVE | INACTIVE	LPENSTAT
LPAR_OVERHEAD=*nnnn*	LPENOVHD
CPU_TYPE=UNKNOWN | CP | ICF | IFL | ZIIP | ZAPP | OTHER	LPENPTY
CPU_ID=*nnnn*	LPENCPU
CPU_SUSPEND_TIME=*nn*%	LPENMSPC
CPU_LP_OVERHEAD_TIME=*nn*%	LPENOVER
CPU_BUSY_TIME=*nn*%	LPENBUSY
PHYSICAL_CPU_BUSY=*nn*%	LPENPHYS
LOGICAL_CPU_LOAD=*nn*%	LPENLGLD
VM_CPU_LOAD=*nn*%	LPENVMLD

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
8	RC_ERR	36	RS_LENGTH_NOT_VALID	Specified length is not valid
		3002	RS_INVALID_PARAMETER	Invalid parameter name

RC	RC Name	RS	RS Name	Description
		3003	RS_INVALID_OPERAND	Invalid parameter operand
		3004	RS_MISSING_PARAMETER	Required parameter missing
		3016	RS_SEGMENT_EMPTY	SMAPIOUT segment empty
		3017	RS_SEGMENT_DATA_INVALID	SMAPIOUT segment does not contain valid data
		3018	RS_SMAPIOUT_NOT_FOUND	SMAPIOUT segment not found
		3019	RS_CPU_DATA_UNAVAILABLE	SMAPIOUT CPU data not found
24	RCERR_SYNTAX	*pprr*	*pprr*	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
396	RCERR_INTERNAL	*nnnn*	*psrc*	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

System_Performance_Threshold_Disable

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
event_type

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use System_Performance_Threshold_Disable to disable thresholds for asynchronous event production.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 36.

function_name
(string,35,char43) The API function name – in this case, 'System_Performance_Threshold_Disable'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See “Client Authentication” on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) Used strictly for authorization – i.e. the authenticated user must have authorization to perform this function for this target.

event_type
(string,1-17,char42 plus blank) One of the following, followed by a null (ASCIIZ) terminator:

- System_CPU
- System_Virtual_IO
- System_Paging
- System_DASD_IO
- User_CPU *userid*
- User_IO *userid*

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
8	RC_ERR	3002	RS_INVALID_PARAMETER	Invalid parameter name
24	RCERR_SYNTAX	*pprr*	*pprr*	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid

RC	RC Name	RS	RS Name	Description
396	RCERR_INTERNAL	*nnnn*	*psrc*	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

System_Performance_Threshold_Enable

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
event_type

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use System_Performance_Threshold_Enable to enable thresholds for asynchronous event production.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 35.

function_name
(string,35,char43) The API function name – in this case, 'System_Performance_Threshold_Enable'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) Used strictly for authorization – i.e. the authenticated user must have authorization to perform this function for this target.

event_type
(string,1-26,char42 plus blank plus /) One of the following, with the appropriate value(s) specified, followed by a null (ASCIIZ) terminator:

- System_CPU = *percentage*
- System_Virtual_IO = *rate/sec*
- System_Paging = *rate/sec*
- System_DASD_IO = *rate/sec*
- User_CPU = *userid percentage*
- User_IO = *userid rate/sec*

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
8	RC_ERR	4	RS_NOT_FOUND	Performance monitoring virtual server not found
		3002	RS_INVALID_PARAMETER	Invalid parameter name
24	RCERR_SYNTAX	*pprr*	*pprr*	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager

RC	RC Name	RS	RS Name	Description
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
396	RCERR_INTERNAL	*nnnn*	*psrc*	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

System_Processor_Query

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

The following output parameters will be returned if there is no error:

system_processor_partition_mode_length
system_processor_partition_mode
system_processor_array_length
system_processor_array
system_processor_structure_length
system_processor_structure
address
status
type
core_id

The following output parameters will be returned if there is an error:

error_length
error_data

Notes:

1. *core_id* will only be returned when multithreading is enabled. *core_id* identifies the core ID of the core that contains the specified processor.
2. An array consists of zero or more of its components.
3. A structure consists of one set of its components.

Purpose

Use System_Processor_Query to list all real processors accessible to the system and indicate the way in which each processor is being used. This API also returns the mode of the logical partition.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 22.

function_name
(string,22,char43) The API function name – in this case, 'System_Processor_Query'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

  Note: *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

  Note: *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) This must match an entry in the authorization file that also contains the authenticated_userid and the function_name (System_Processor_Query).

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in "Response 1 -- Immediate Request Verification" on page 587).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

If no errors are encountered, the following parameters will be returned:

system_processor_partition_mode_length
(int4) The length of *systems_processor_partition_mode*.

system_processor_partition_mode
(string,4-10,char26 plus - /) One of the following:

ESA/390
GENERAL
LINUX-ONLY
Z/VM

system_processor_array_length
(int4) The length of the *system_processor_array*.

system_processor_array
(array) An array consisting of zero or more instances of *system_processor_structure*.

system_processor_structure_length
(int4) The length of the *system_processor_structure*

system_processor_structure
(structure) A structure consisting of one set of the following parameters, with a blank separating each parameter:

address
(string,4,char16) The processor address.

status
(string,6-9,char26) The processor status. One of the following:

Master-Processor
Alternate
Parked
Standby

type
(string,2-4,char26) The processor status. One of the following:

CP
IFL
ZIIP
ICF

core_id
(string,0-4,char16) When multithreading is enabled, identifies the core ID of the core that contains the specified processor. When multithreading is not enabled, this output parameter will be omitted.

If RC=8, the following parameters will be returned:

error_length
(int4) Length of error_data.

error_data
(string) "UNKNOWN_PARAMETER_NAMES=", followed by the specific input parameter that failed.

System_RDR_File_Manage

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
`spoolids=`*value*
`action=`*value*

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use System_RDR_File_Manage to manipulate the reader files of the target.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 18.

function_name
(string,18,char43) The API function name – in this case, 'System_RDR_File_Manage'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The user ID under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The user ID under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See “Client Authentication” on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) A string that must match an entry in the authorization file that also contains the *authenticated_userid* and the *function_name* ('System_RDR_File_Manage').

spoolids=*value*
This is a required parameter. The format for specifying a required parameter is *parameter_name=value*, followed by a null (ASCIIZ) terminator. In this case, *value* can be one of the following:

- (string,1-4,char10) The spool IDs that are to be manipulated. Multiple spool IDs must be separated with a blank.
- (string,3,char26) ALL, indicating all the target's reader files.

action=*value*
One of the following:

- (string,5,char26) PURGE (this is the default).
- (string,5,char26) ORDER
- (string,8,char26) TRANSFER

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Usage Notes

1. RC= 24 and RS=pprr (syntax error) are only applicable for the first nine input parameters. SMAPI syntax checking is not performed on the other input parameters.
2. When multiple spool IDs are specified, the actions take place in the order the individual spool IDs are listed. If an error occurs with one or more spool IDs, the actions for valid spool IDs are still successful.
3. When TRANSFER is specified, the reader files are transferred to the authenticated user virtual machine.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
8	RC_ERR	3002	RS_INVALID_PARAMETER	Invalid parameter name
		3003	RS_INVALID_OPERAND	Invalid parameter operand
		3004	RS_MISSING_PARAMETER	Invalid parameter missing
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter pprr
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
396	RCERR_INTERNAL	nnnn	psrc	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

System_RDR_File_Query

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
rdr_file_info

Purpose

Use System_RDR_File_Query to query the reader files of the target.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 18.

function_name
(string,18,char43) The API function name – in this case, 'System_RDR_File_Query'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The user ID under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The user ID under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See “Client Authentication” on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) A string that must match an entry in the authorization file that also contains the *authenticated_userid* and the *function_name* (System_RDR_File_Query).

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

rdr_file_info
(string, 1-80, char44) An array of null terminated (ASCIIZ) strings. Each string represents one line as returned by the QUERY RDR *targetid* ALL command. This parameter is outputted only if return code=0.

Usage Note

- This API has no input parameters. Any submitted input parameters are ignored.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
8	RC_ERR	4	RS_NOT_FOUND	No reader files found.
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter.
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; user ID or password not valid

RC	RC Name	RS	RS Name	Description
396	RCERR_INTERNAL	nnnn	psrc	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist

System_SCSI_Disk_Add

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
`dev_num=`*value*
`dev_path_array=`*value*
`option=`*value*
`persist=`*value*

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use System_SCSI_Disk_Add to dynamically add a SCSI disk to a running z/VM system.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 20.

function_name
(string,20,char43) The API function name – in this case, 'System_SCSI_Disk_Add'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See “Client Authentication” on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) This must match an entry in the authorization file that also contains the *authenticated_userid* and the *function_name* (System_SCSI_Disk_Add).

Note: The format for specifying the following additional input parameters is *parameter_name=value*, followed by a null (ASCIIZ) terminator. They may be specified in any order. For more information, see "Key-value pair (KVP) input parameters" on page 51.

dev_num=*value*
(string,1-4,char16) The SCSI disk device number. This is a required parameter.

dev_path_array=*value*
An array of device path structures. Each structure has the following fields (each field is separated by a blank and the structures are separated by semicolons):

fcp_dev_num
(string,1-4,char16) The FCP device number.

fcp_wwpn
(string,1-16,char16) The world wide port number.

fcp_lun
(string,1-16,char16) The logical unit number.

This is a required parameter.

option=*value*
(string,0-1,char10) One of the following:

1
Add a new SCSI disk. This is the default if unspecified.

2
Add new paths to an existing SCSI disk.

3
Delete paths from an existing SCSI disk.

persist=*value*
(string,0-3,char42) This can be one of the following values:

NO
The SCSI device is updated on the active system, but is *not* updated in the permanent configuration for the system.

YES
The SCSI device is updated on the active system and also in the permanent configuration for the system.

If not specified, the default is NO.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Usage Notes

1. This function is used to sense and add a new plugged-in disk for a running z/VM system as an offline disk. The Image_Volume_Add function should be used to format and add the new disk to be used by virtual images to the z/VM system configuration file.
2. Syntax errors (RC = 24 and RS = *pprr*) are only applicable to the common input parameters. Syntax checking is *not* performed on the additional input parameters for this API. If a valid parameter is specified multiple times, the last value specified for that parameter will be used. For more information, see "Key-value pair (KVP) input parameters" on page 51.
3. This API uses the generic SCSI attribute when creating the SCSI DISK. CP will attempt to determine the real device attributes and update the edevice attributes accordingly. For example, an edevice created on a DS8000 enterprise data system would return the 2107 attribute when queried.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
4	RC_WNG	8	RS_DEV_NOT_FOUND	Device does not exist
8	RC_ERR	3002	RS_INVALID_PARAMETER	Invalid parameter name
		3003	RS_INVALID_OPERAND	Invalid parameter operand
		3004	RS_MISSING_PARAMETER	Required parameter missing
24	RCERR_SYNTAX	*pprr*	*pprr*	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
396	RCERR_INTERNAL	*nnnn*	*psrc*	Internal system error - product-specific return code (See "Internal Return Codes (RC = 396, 592, or 596)" on page 834)

RC	RC Name	RS	RS Name	Description
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

System_SCSI_Disk_Delete

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
`dev_num=`*value*
`persist=`*value*

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use System_SCSI_Disk_Delete to delete a real SCSI disk.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 23.

function_name
(string,23,char43) The API function name – in this case, 'System_SCSI_Disk_Delete'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See “Client Authentication” on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) This must match an entry in the authorization file that also contains the *authenticated_userid* and the *function_name* (System_SCSI_Disk_Delete).

Note: The format for specifying the following additional input parameter is *parameter_name=value*, followed by a null (ASCIIZ) terminator. For more information, see "Key-value pair (KVP) input parameters" on page 51.

dev_num=*value*
(string,1-4,char16) The SCSI disk device number. This is a required parameter.

persist=*value*
(string,0-3,char42) This can be one of the following values:

NO
The SCSI device is deleted from the active system, but is *not* deleted from the permanent configuration for the system.

YES
The SCSI device is deleted from the active system and also from the permanent configuration for the system.

If not specified, the default is NO.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Usage Notes

1. Syntax errors (RC = 24 and RS = *pprr*) are only applicable to the common input parameters. Syntax checking is *not* performed on the additional input parameters for this API. If a valid parameter is specified multiple times, the last value specified for that parameter will be used. For more information, see "Key-value pair (KVP) input parameters" on page 51.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
8	RC_ERR	8	RS_DEV_NOT_FOUND	Device does not exist
		3002	RS_INVALID_PARAMETER	Invalid parameter name
		3003	RS_INVALID_OPERAND	Invalid parameter operand
		3004	RS_MISSING_PARAMETER	Required parameter missing
24	RCERR_SYNTAX	*pprr*	*pprr*	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
396	RCERR_INTERNAL	*nnnn*	*psrc*	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

System_SCSI_Disk_Query

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
dev_num=*value*

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
scsi_info_array (1)
  scsi_info_structure (2)
    dev_id
    dev_type
    dev_attr
    dev_size
    fcp_array (1)
      fcp_structure (2)
        fcp_dev_id
        fcp_dev_wwpn
        fcp_dev_lun

Note:

1. An array consists of zero or more of its components.
2. A structure consists of one set of its components.

Purpose

Use System_SCSI_Disk_Query to query a real SCSI disk or all real SCSI disks..

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 22.

function_name
(string,22,char43) The API function name – in this case, 'System_SCSI_Disk_Query'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) This must match an entry in the authorization file that also contains the *authenticated_userid* and the *function_name* (System_SCSI_Disk_Query).

Note: The format for specifying the following additional input parameters is *parameter_name=value*, followed by a null (ASCIIZ) terminator. For more information, see "Key-value pair (KVP) input parameters" on page 51.

dev_num=*value*
(string,1-4,char36) The device number, or 'ALL'. This is a required parameter.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

scsi_info_array
(array) An array consisting of zero or more instances of *scsi_info_structure*, with each structure terminated by a null (ASCIIZ) character, as follows:

scsi_info_structure
(structure) A structure consisting of one set of the following parameters, with a blank separating each parameter:

dev_id
(string,4,char16) The device number.

dev_type
(string,3,char36) The device type.

dev_attr
(string,4,char36) The device attribute.

dev_size
(string,1-8,char10) The device size, in blocks. (The block size is 512.)

fcp_array
(array) An array consisting of zero or more instances of *fcp_structure*, as follows:

fcp_structure
(structure) A structure consisting of one set of the following parameters:

fcp_dev_id
(string,4,char16) The FCP device number.

fcp_dev_wwpn
(string,16,char16) The world wide port number.

fcp_dev_lun
(string,16,char16) The logical unit number.

Usage Notes

1. Syntax errors (RC = 24 and RS = *pprr*) are only applicable to the common input parameters. Syntax checking is *not* performed on the additional input parameter for this API. If a valid parameter is specified multiple times, the last value specified for that parameter will be used. For more information, see “Key-value pair (KVP) input parameters” on page 51.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
4	RC_WNG	28	RS_EMPTY	Return buffer is empty
8	RC_ERR	8	RS_DEV_NOT_FOUND	Device does not exist
		3002	RS_INVALID_PARAMETER	Invalid parameter name
		3003	RS_INVALID_OPERAND	Invalid parameter operand
		3004	RS_MISSING_PARAMETER	Required parameter missing
24	RCERR_SYNTAX	*pprr*	*pprr*	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server

RC	RC Name	RS	RS Name	Description
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
396	RCERR_INTERNAL	*nnnn*	*psrc*	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

System_Service_Query

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
system_service_query_list_length
system_service_query_list

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
system_service_query_data_length
system_service_query_data
error_data_length (error only)
error_data (error only)

Purpose

Use System_Service_Query to query the status of an APAR, PTF, or RSU for a zVM component. Note that the status is based on information returned from the SERVICE EXEC, not from querying the running components.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 20.

function_name
(string,13,char43) The API function name – in this case, 'System_Service_Query'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).

- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) This must match an entry in the authorization file that also contains the *authenticated_userid* and the *function_name* (System_Service_Query).

system_service_query_list_length
(int4) Length of *system_service_query_list*.

system_service_query_list
(string,1-maxlength,charNA) A series of null-terminated strings, each containing "COMPONENT=" followed by a series of blank-delimited "*subkeyword=value*" pairs, as shown in Table 18 on page 607.

Table 18. Input Keywords and Values for System_Service_Query

keyword_parameter=	**Blank-delimited *subkeyword=value* pairs**
COMPONENT=	• NAMECOMPONENT=*compname* (**Required.** Refer to the Service Guide for component names. Examples: VMSES, REXX, LE, CMS, CP, GCS, DV, TSAF, AVS, RSCS, TCPIP, DIRM, RACF, PERFTK, VMHCD.) • TYPE=APAR | PTF | RSU (**Required.**) • NUMBER=*APAR_number* | *PTF_number* (**Required** for APAR or PTF, ignored for RSU.)

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

system_service_query_data_length
(int4) Length of *system_service_query_data*.

system_service_query_data
(string) A series of null-terminated strings, each containing "COMPONENT=" followed by a series of blank-delimited "*output_subkeyword=value*" pairs, as shown in Table 19 on page 608.

Table 19. Output Keywords and Values for System_Service_Query

output_keyword_parameter=	**Blank-delimited *output_subkeyword=value* pairs**
COMPONENT=	**Output subkeywords for all types:** • NAME=*compname* • TYPE=APAR | PTF | RSU **Additional output subkeywords for APAR or PTF only:** • SERVICE_STATUS=UNKNOWN | RECEIVED | APPLIED | BUILT | PUT2PROD • TIME=*hh*:*mm*:*ss* • DATE=*yyyy-mm-dd* (or in the format configured for the system) • SYSTEM=*name* **Additional output subkeywords for RSU only:** • SERVICE_LEVEL="*level_string*" (example: "000-0000") • PRODUCTION_LEVEL="*level_string*" (example: "GDLVMK4.000-0000")

If RC=8 and RS=3002, the following parameters will be returned:

error_data_length
(int4) Length of *error_data*.

error_data
(string) "UNKNOWN_PARAMETER_NAMES=", followed by a blank-delimited list of input parameter names that are not valid, then followed by a null terminator.

If RC=8 and RS=3003, the following parameters will be returned:

error_data_length
(int4) Length of *error_data*.

error_data
(string) "UNKNOWN_PARAMETER_OPERAND=", followed by a blank-delimited list of input parameter operands that are not valid, then followed by a null terminator.

For all other errors, the following parameters will be returned:

error_data_length
(int4) Length of *error_data*.

error_data
(string) "COMMAND_IN_ERROR=", followed by the command that failed and any accompanying error message and/or return code, then followed by a null terminator.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
8	RC_ERR	4	RS_NOT_FOUND	APAR or PTF not found

RC	RC Name	RS	RS Name	Description
		28	RS_OUTPUT_NOT_VALID	Unexpected error obtaining information. See error data for details.
		3002	RS_INVALID_PARAMETER	Invalid parameter name
		3003	RS_INVALID_OPERAND	Invalid parameter operand
		3004	RS_MISSING_PARAMETER	Required parameter missing
24	RCERR_SYNTAX	*pprr*	*pprr*	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
396	RCERR_INTERNAL	*nnnn*	*psrc*	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

System_Shutdown

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
`Within=`*value*
`By=`*value*
`Immediate=`*value*
`Reipl=`*value*
`Cancel=`*value*

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
error_data_length (error only)
error_data (error only)

Purpose

Use System_Shutdown to systematically end all system function.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 15.

function_name
(string,13,char43) The API function name – in this case, 'System_Shutdown'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).

- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) This must match an entry in the authorization file that also contains the *authenticated_userid* and the *function_name* (System_Shutdown).

Note: The format for specifying the following additional input parameters is *parameter_name*=*value*, followed by a null (ASCIIZ) terminator. For more information, see "Key-value pair (KVP) input parameters" on page 51.

`Within`=*value*
(string,0-5,char10) Send a shutdown signal to enabled users and delay the shutdown until either the specified interval (minus the amount of time reserved for a CP shutdown) has elapsed, or until all signaled user machines indicate that they have shut down, whichever occurs first. The interval is specified as a number of seconds in the range of 1-65535. The default is that no `Within`=*value* is submitted.

By=*value*
(string,0-8,char10 plus :) Sends a shutdown signal to enabled users and delay the shutdown until either the designated time of day (minus the amount of time reserved for a CP shutdown) is reached, or until all signaled user machines indicate that they have shut down, whichever occurs first. The time can be specified as *hh*:*mm* or *hh*:*mm*:*ss*. The equivalent interval in seconds must be in the range 1-65535. The default is that no By=*value* is submitted.

`Immediate`=*value*
(string,0-11,char36) One of the following:

IMMEDIATE
Shut down the system immediately without sending shutdown signals to enabled users, even if a previous SHUTDOWN command is pending. If a previous SHUTDOWN command is pending, its operands are not used and must be specified with IMMEDIATE on the new command if they are required.

NOIMMEDIATE
Do *not* issue the SHUTDOWN with the IMMEDIATE option. This is the default.

`Reipl`=*value*
(string,0-7,char26) One of the following:

REIPL
Specifies that the system is to be restarted immediately after the SHUTDOWN command completes. This is the default.

NOREIPL
Specifies that the system is *not* to be restarted immediately after the SHUTDOWN command completes.

Cancel=*value*
(string,0-8,char26) One of the following:

CANCEL
This causes a scheduled shutdown to be terminated. Any guests that received termination signals when the original SHUTDOWN command was issued continue to process those signals.

NOCANCEL
This does *not* cause a scheduled shutdown to be terminated. This is the default.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

If RC=8 and RS=3002, the following parameters will be returned:

error_data_length
(int4) Length of *error_data*.

error_data
(string) "UNKNOWN_PARAMETER_NAMES=", followed by a blank-delimited list of input parameter names that are not valid, then followed by a null terminator.

If RC=8 and RS=3003, the following parameters will be returned:

error_data_length
(int4) Length of *error_data*.

error_data
(string) "UNKNOWN_PARAMETER_OPERAND=", followed by a blank-delimited list of input parameter operands that are not valid, then followed by a null terminator.

Usage Notes

1. Syntax errors (RC = 24 and RS = *pprr*) are only applicable to the common input parameters. Syntax checking is *not* performed on the additional input parameters for this API. If a valid parameter is specified multiple times, the last value specified for that parameter will be used. For more information, see “Key-value pair (KVP) input parameters” on page 51.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
8	RC_ERR	24	RS_CONFLICTING_PARMS	Conflicting parameters
		3002	RS_INVALID_PARAMETER	Invalid parameter name
		3003	RS_INVALID_OPERAND	Invalid parameter operand

RC	RC Name	RS	RS Name	Description
24	RCERR_SYNTAX	*pprr*	*pprr*	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
396	RCERR_INTERNAL	*nnnn*	*psrc*	Internal system error - product-specific return code (See "Internal Return Codes (RC = 396, 592, or 596)" on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

System_Spool_Utilization_Query

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
spool_information_structure (2)
- *spool_information_structure_length*
- *total_spool_pages*
- *total_spool_pages_in_use*
- *total_spool_percent_used*
- *spool_volume_array* (1)
  - *spool_volume_structure* (2)
    - *volid*
    - *rdev*
    - *total_pages*
    - *pages_in_use*
    - *percent_used*
    - *dump*
    - *drained*

Note:

1. An array consists of zero or more of its components.
2. A structure consists of one set of its components.

Purpose

Use System_Spool_Utilization_Query to obtain information about the z/VM spool space defined on the system.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 30.

function_name
(string,13,char43) The API function name – in this case, 'System_Spool_Utilization_Query'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) This must match an entry in the authorization file that also contains the *authenticated_userid* and the *function_name* (System_Spool_Utilization_Query).

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

spool_information_structure
(structure) A structure consisting of one set of the following parameters, with a blank separating each parameter:

spool_information_structure_length
(int4) The combined length of the remaining parameters in *spool_information_structure* (not including this parameter).

total_spool_pages
(string,1-8,char10 plus 'K') The total number of pages allocated for spool use on the system.

total_spool_pages_in_use
(string,1-8,char10) The total number of pages in use for spool on the system.

total_spool_percent_used
(string,1-3,char10) The percentage of the available spool space currently in use on the system.

spool_volume_array
(array) An array consisting of zero or more instances of *spool_volume_structure*, with each structure terminated by a null (ASCIIZ) character, as follows:

spool_volume_structure
(structure) A structure consisting of one set of the following parameters, with a blank separating each parameter:

volid
(string,1-6,char42) The volume ID of the spool volume.

rdev
(string,1-4,char16) The RDEV of the spool volume.

total_pages
(string,1-8,char10) The total number of pages on the volume available for spool use.

pages_in_use
(string,1-8,char10) The total number pages in use on the volume for spool files.

percent_used
(string,1-3,char10) The percentage of the available spool space on the volume in use.

dump
(string,4-7,char26) One of the following:

NOTDUMP
This spool space is *not* reserved for DUMP space only.

DUMP
This spool space is reserved for DUMP space only.

drained
(string,7-10,char26) One of the following:

NOTDRAINED
This spool space is *not* drained and CP is allocating new spool space to this volume.

DRAINED
This spool space is drained and CP is not allocating new spool space to this volume.

Usage Notes

1. In an SSI, this API will return values only for those CPOWNED volumes with SPOOL space that are owned by the system where the API is executed.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
24	RCERR_SYNTAX	*pprr*	*pprr*	Syntax error in function parameter

RC	RC Name	RS	RS Name	Description
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
396	RCERR_INTERNAL	*nnnn*	*psrc*	Internal system error - product-specific return code (See "Internal Return Codes (RC = 396, 592, or 596)" on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

System_WWPN_Query

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
`owner=`*value*

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
wwpn_array (1)
  wwpn_structure (2)
    fcp_dev_id
    npiv_wwpn
    chpid
    perm_wwpn
    dev_status
    owner

Note:

1. An array consists of zero or more of its components.
2. A structure consists of one set of its components.

Purpose

Use System_WWPN_Query to query all FCPs on a z/VM system and return a list of WWPNs.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 17.

function_name
(string,17,char43) The API function name – in this case, 'System_WWPN_Query'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) This must match an entry in the authorization file that also contains the *authenticated_userid* and the *function_name* (System_WWPN_Query).

Note: The format for specifying the following additional input parameters is *parameter_name=value*, followed by a null (ASCIIZ) terminator. For more information, see "Key-value pair (KVP) input parameters" on page 51.

owner=*value*
(string,0-3,char26) One of the following:

YES
Indicates that the output of this query should include the owner of the WWPN, if is it attached to a user.

NO
Indicates that no owner information should be returned. This is the default.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

wwpn_array
(array) An array consisting of zero or more instances of *wwpn_structure*, with each structure terminated by a null (ASCIIZ) character, as follows:

wwpn_structure
(structure) A structure consisting of one set of the following parameters, with a blank separating each parameter:

fcp_dev_id
(string,4,char16) The FCP device number.

npiv_wwpn
(string,4-16,char16) NPIV world wide port number or "NONE".

chpid
(string,2,char16) Channel path ID

perm_wwpn
(string,16,char16) Physical world wide port number

dev_status
(string,1,char10) FCP device status. The following values are possible:

1
Active

2
Free

3
Offline

owner
(string,1-8,char42) The owner of the WWPN, if is it attached to a user. If the WWPN is not attached, this value will be "NONE".

Note that this value is returned only if `owner`=YES was specified.

Usage Notes

1. Syntax errors (RC = 24 and RS = *pprr*) are only applicable to the common input parameters. Syntax checking is *not* performed on the additional input parameters for this API. If a valid parameter is specified multiple times, the last value specified for that parameter will be used. For more information, see “Key-value pair (KVP) input parameters” on page 51.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
4	RC_WNG	28	RS_EMPTY	Return buffer is empty
8	RC_ERR	3002	RS_INVALID_PARAMETER	Invalid parameter name
24	RCERR_SYNTAX	*pprr*	*pprr*	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid

RC	RC Name	RS	RS Name	Description
396	RCERR_INTERNAL	*nnnn*	*psrc*	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Virtual_Channel_Connection_Create

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
image_device_number_length
image_device_number
coupled_image_name_length
coupled_image_name
coupled_image_device_number_length
coupled_image_device_number

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Virtual_Channel_Connection_Create to establish a virtual network connection between two active virtual images. A virtual network connector (CTCA) is added to each virtual image's configuration if one is not already defined.

See "Virtual_Channel_Connection_Create_DM" on page 625 to add a virtual network connection between two virtual images to their directory entries.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 33.

function_name
(string,33,char43) The API function name – in this case, 'Virtual_Channel_Connection_Create'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The name of the image obtaining a connection device.

image_device_number_length
(int4) Length of *image_device_number*.

image_device_number
(string,1-4,char16) Specifies the virtual device number of the network device in the active virtual image.

coupled_image_name_length
(int4) Length of *coupled_image_name*.

coupled_image_name
(string,1-8,char42) The virtual image name of the target virtual image that is to be connected. This parameter is required here (for an active instance).

coupled_image_device_number_length
(int4) Length of *coupled_image_device_number*.

coupled_image_device_number
(string,1-4,char16) The virtual device number of the network device in another virtual image.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Usage Notes

1. The use of some optional parameters requires that other optional parameters be specified as well. If you are uncertain of these interdependencies, see *z/VM: CP Commands and Utilities Reference* for more information on the parameters used by this function.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
200	RCERR_IMAGEOP	4	RS_NOT_FOUND	Image not found
		12	RS_NOT_ACTIVE	Image not active
204	RCERR_IMAGEDEVU	12	RS_BUSY	Image device is busy
212	RCERR_IMAGECONN	4	RS_NO_PARTNER	Partner image not found
		28	RS_DEV_INCOMPATIBLE	Image device not correct type for requested connection
396	RCERR_INTERNAL	nnnn	psrc	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Virtual_Channel_Connection_Create_DM

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
image_device_number_length
image_device_number
coupled_image_name_length
coupled_image_name

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Virtual_Channel_Connection_Create_DM to add a virtual network connection between two virtual images to their directory entries. A virtual network connector (CTCA) is added to each virtual image's directory entry if one is not already defined.

See "Virtual_Channel_Connection_Create" on page 622 to establish a virtual network connection between two active virtual images.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 36.

function_name
(string,36,char43) The API function name – in this case, 'Virtual_Channel_Connection_Create_DM'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).

- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The name of the image obtaining a connection device.

image_device_number_length
(int4) Length of *image_device_number*.

image_device_number
(string,1-4,char16) Specifies the virtual device number of the network device in the active virtual image.

coupled_image_name_length
(int4) Length of *coupled_image_name*.

coupled_image_name
(string,0-8,char42) The virtual image name of the target virtual image that is to be connected. This parameter is optional here (for a static instance), required in Virtual_Channel_Connection_Create (for an active instance).

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Usage Notes

1. The use of some optional parameters requires that other optional parameters be specified as well. If you are uncertain of these interdependencies, see *z/VM: CP Commands and Utilities Reference* for more information on the parameters used by this function.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
		8	RS_OFFLINE	Request successful; object directory offline
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		12	RS_AUTHERR_DM	Request not authorized by directory manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
400	RCERR_IMAGEDEF	4	RS_NOT_FOUND	Image definition not found
		12	RS_LOCKED	Image definition is locked
404	RCERR_IMAGEDEVD	4	RS_EXISTS	Image device already defined
		12	RS_LOCKED	Image device is locked
412	RCERR_IMAGECONND	4	RS_NO_PARTNER	Partner image not found
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available
592	RCERR_ASYNC_DM	nnnn	opid	Asynchronous operation started - product-specific asynchronous operation ID (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
596	RCERR_INTERNAL_DM	nnnn	psrc	Internal directory manager error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Virtual_Channel_Connection_Delete

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
image_device_number_length
image_device_number

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Virtual_Channel_Connection_Delete to terminate a virtual network connection between two active virtual images and to remove the virtual network connector (CTCA) from the virtual image's configuration. The specified network connector will be removed whether or not there is an active connection.

See "Virtual_Channel_Connection_Delete_DM" on page 631 to remove a virtual network connection from a virtual image's directory entry and to remove the virtual network connector (CTCA) from the virtual image's directory entry.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 33.

function_name
(string,33,char43) The API function name – in this case, 'Virtual_Channel_Connection_Delete'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The name of the image for which the connection device is being removed.

image_device_number_length
(int4) Length of *image_device_number*.

image_device_number
(string,1-4,char16) The virtual device number of the device to be deleted.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid

RC	RC Name	RS	RS Name	Description
200	RCERR_IMAGEOP	4	RS_NOT_FOUND	Image not found
		12	RS_NOT_ACTIVE	Image not active
204	RCERR_IMAGEDEVU	8	RS_NOT_EXIST	Image device does not exist
396	RCERR_INTERNAL	nnnn	psrc	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Virtual_Channel_Connection_Delete_DM

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
image_device_number_length
image_device_number

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Virtual_Channel_Connection_Delete_DM to remove a virtual network connection from a virtual image's directory entry and to remove the virtual network connector (CTCA) from the virtual image's directory entry.

See "Virtual_Channel_Connection_Delete" on page 628 to terminate a virtual network connection between two active virtual images and to remove the virtual network connector (CTCA) from the virtual image's configuration.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 36.

function_name
(string,36,char43) The API function name – in this case, 'Virtual_Channel_Connection_Delete_DM'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The name of the image for which the connection device is being removed.

image_device_number_length
(int4) Length of *image_device_number*.

image_device_number
(string,1-4,char16) The virtual device number of the device to be deleted.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
		8	RS_OFFLINE	Request successful; object directory offline
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		12	RS_AUTHERR_DM	Request not authorized by directory manager

RC	RC Name	RS	RS Name	Description
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
400	RCERR_IMAGEDEF	4	RS_NOT_FOUND	Image definition not found
		12	RS_LOCKED	Image definition is locked
404	RCERR_IMAGEDEVD	8	RS_NOT_DEFINED	Image device not defined
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available
592	RCERR_ASYNC_DM	nnnn	opid	Asynchronous operation started - product-specific asynchronous operation ID (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
596	RCERR_INTERNAL_DM	nnnn	psrc	Internal directory manager error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Virtual_Network_Adapter_Connect_LAN

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
image_device_number_length
image_device_number
lan_name_length
lan_name
lan_owner_length
lan_owner

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Virtual_Network_Adapter_Connect_LAN to connect an existing virtual network adapter on an active virtual image to an existing virtual network LAN.

See "Virtual_Network_Adapter_Connect_LAN_DM" on page 638 to define a virtual network LAN connection for an existing virtual network adapter in a virtual image's directory entry.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 35.

function_name
(string,35,char43) The API function name – in this case, 'Virtual_Network_Adapter_Connect_LAN'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The name of the image for which a LAN connection is being created.

image_device_number_length
(int4) Length of *image_device_number*.

image_device_number
(string,1-4,char16) The virtual device address for the new adapter.

lan_name_length
(int4) Length of *lan_name*.

lan_name
(string,1-8,char36 plus $#@) The name of the guest LAN segment to connect the virtual image.

lan_owner_length
(int4) Length of *lan_owner*.

lan_owner
(string,1-8,char42) The virtual image owning the guest LAN segment to be connected.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Usage Notes

1. A virtual network adapter may be created using Virtual_Network_Adapter_Create or Virtual_Network_Adapter_Create_DM. A virtual network LAN may be created using Virtual_Network_LAN_Create.
2. The value specified for *image_device_number* must take into account the number of network adapter devices requested to ensure that there will be enough addresses between the address specified and the high address range of FFFF. Otherwise, return code 396 reason code 9 may be received from this function. For example, if the value of *network_adapter_devices* is 3, then the largest valid value for *image_device_number* is FFFD. This would accommodate three device addresses: FFFD, FFFE, and FFFF.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
200	RCERR_IMAGEOP	12	RS_NOT_ACTIVE	Image not active
204	RCERR_IMAGEDEVU	8	RS_NOT_EXIST	Image device does not exist
212	RCERR_IMAGECONN	8	RS_AUTHERR_CONNECT	Image not authorized to connect
		12	RS_LAN_NOT_EXIST	LAN does not exist
		20	RS_OWNER_NOT_ACTIVE	Requested LAN owner not active
		28	RS_DEV_INCOMPATIBLE	Image device not correct type for requested connection
		52	RS_MAX_CONN	Maximum number of connections reached
		96	RS_UNKNOWN	Unknown reason
396	RCERR_INTERNAL	nnnn	psrc	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory

RC	RC Name	RS	RS Name	Description
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Virtual_Network_Adapter_Connect_LAN_DM

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
image_device_number_length
image_device_number
lan_name_length
lan_name
lan_owner_length
lan_owner

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Virtual_Network_Adapter_Connect_LAN_DM to define a virtual network LAN connection for an existing virtual network adapter in a virtual image's directory entry.

See "Virtual_Network_Adapter_Connect_LAN" on page 634 to connect an existing virtual network adapter on an active virtual image to an existing virtual network LAN.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 38.

function_name
(string,38,char43) The API function name – in this case, 'Virtual_Network_Adapter_Connect_LAN_DM'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The name of the image for which a LAN connection is being created.

image_device_number_length
(int4) Length of *image_device_number*.

image_device_number
(string,1-4,char16) The virtual device address for the new adapter.

lan_name_length
(int4) Length of *lan_name*.

lan_name
(string,1-8,char36 plus $#@) The name of the guest LAN segment to connect the virtual image.

lan_owner_length
(int4) Length of *lan_owner*.

lan_owner
(string,1-8,char42) The virtual image owning the guest LAN segment to be connected.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Usage Notes

1. A virtual network adapter may be created using Virtual_Network_Adapter_Create_DM.
2. The value specified for *image_device_number* must take into account the number of network adapter devices requested to ensure that there will be enough addresses between the address specified and the high address range of FFFF. Otherwise, return code 396 reason code 9 may be received from this function. For example, if the value of *network_adapter_devices* is 3, then the largest valid value for *image_device_number* is FFFD. This would accommodate three device addresses: FFFD, FFFE, and FFFF.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
		8	RS_OFFLINE	Request successful; object directory offline
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		12	RS_AUTHERR_DM	Request not authorized by directory manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
400	RCERR_IMAGEDEF	4	RS_NOT_FOUND	Image definition not found
		12	RS_LOCKED	Image definition is locked
404	RCERR_IMAGEDEVD	8	RS_NOT_EXIST	Image device does not exist
		12	RS_LOCKED	Image device is locked
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available
592	RCERR_ASYNC_DM	nnnn	opid	Asynchronous operation started - product-specific asynchronous operation ID (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
596	RCERR_INTERNAL_DM	nnnn	psrc	Internal directory manager error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data

RC	RC Name	RS	RS Name	Description
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Virtual_Network_Adapter_Connect_Vswitch

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
image_device_number_length
image_device_number
switch_name_length
switch_name

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Virtual_Network_Adapter_Connect_Vswitch to connect an existing virtual network adapter on an active virtual image to an existing virtual switch.

See "Virtual_Network_Adapter_Connect_Vswitch_DM" on page 645 to define a virtual switch connection for an existing virtual network adapter in a virtual image's directory entry.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 39.

function_name
(string,39,char43) The API function name – in this case, 'Virtual_Network_Adapter_Connect_Vswitch'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The name of the user to which virtual network adapter virtual switch connection information will be added.

image_device_number_length
(int4) Length of *image_device_number*.

image_device_number
(string,1-4,char16) The virtual device address for the new adapter.

switch_name_length
(int4) Length of *switch_name*.

switch_name
(string,1-8,char36 plus @#$_) The name of the virtual switch segment to connect to the virtual image.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager

RC	RC Name	RS	RS Name	Description
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
200	RCERR_IMAGEOP	12	RS_NOT_ACTIVE	Image not active
204	RCERR_IMAGEDEVU	8	RS_NOT_EXIST	Image device does not exist
212	RCERR_IMAGECONN	8	RS_AUTHERR_CONNECT	Image not authorized to connect
		28	RS_DEV_INCOMPATIBLE	Image device not correct type for requested connection
396	RCERR_INTERNAL	nnnn	psrc	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Virtual_Network_Adapter_Connect_Vswitch_DM

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
image_device_number_length
image_device_number
switch_name_length
switch_name

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Virtual_Network_Adapter_Connect_Vswitch_DM to define a virtual switch connection for an existing virtual network adapter in a virtual image's directory entry.

See "Virtual_Network_Adapter_Connect_Vswitch" on page 642 to connect an existing virtual network adapter on an active virtual image to an existing virtual switch.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 42.

function_name
(string,42,char43) The API function name – in this case, 'Virtual_Network_Adapter_Connect_Vswitch_DM'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).

- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The name of the user or profile to which virtual network adapter virtual switch connection information will be added.

image_device_number_length
(int4) Length of *image_device_number*.

image_device_number
(string,1-4,char16) The virtual device address for the new adapter.

switch_name_length
(int4) Length of *switch_name*.

switch_name
(string,1-8,char36 plus @#$_) The name of the virtual switch segment to connect to the virtual image.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
		8	RS_OFFLINE	Request successful; object directory offline

RC	RC Name	RS	RS Name	Description
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		12	RS_AUTHERR_DM	Request not authorized by directory manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
400	RCERR_IMAGEDEF	4	RS_NOT_FOUND	Image definition not found
		12	RS_LOCKED	Image definition is locked
404	RCERR_IMAGEDEVD	8	RS_NOT_EXIST	Image device does not exist
		12	RS_LOCKED	Image device is locked
412	RCERR_IMAGECONND	28	RS_DEV_INCOMPATIBLE	Image device not correct type for requested connection
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available
592	RCERR_ASYNC_DM	nnnn	opid	Asynchronous operation started - product-specific asynchronous operation ID (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
596	RCERR_INTERNAL_DM	nnnn	psrc	Internal directory manager error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Virtual_Network_Adapter_Connect_Vswitch_Extended

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
`image_device_number=`*value*
`switch_name=`*value*
`port_selection=`*value*

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Virtual_Network_Adapter_Connect_Vswitch_Extended to connect an existing virtual network adapter on an active virtual image to an existing virtual switch.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 48.

function_name
(string,48,char43) The API function name – in this case, 'Virtual_Network_Adapter_Connect_Vswitch_Extended'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The name of the user to which virtual network adapter virtual switch connection information will be added.

Note: The format for specifying the following additional input parameters is *parameter_name=value*, followed by a null (ASCIIZ) terminator. They may be specified in any order. For more information, see "Key-value pair (KVP) input parameters" on page 51.

image_device_number=*value*
(string,1-4,char16) The virtual device address for the new adapter. This input parameter is required.

switch_name=*value*
(string,1-8,char36 plus @#$_) The name of the virtual switch segment to connect to the virtual image. This input parameter is required.

port_selection=*value*
One of the following:

- (string,4,AUTO) CP will choose the port.
- (string,0-5,char16; range 0-65535) The port number to be used.

If unspecified, AUTO is the default.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
8	RC_ERR	3002	RS_INVALID_PARAMETER	Invalid parameter name

RC	RC Name	RS	RS Name	Description
		3003	RS_INVALID_OPERAND	Invalid parameter operand
		3004	RS_MISSING_PARAMETER	Required parameter missing
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
		20	RS_TARGET_IMG_NOT _AUTHORIZED	Target image not authorized for function
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
200	RCERR_IMAGEOP	12	RS_NOT_ACTIVE	Image not active
204	RCERR_IMAGEDEVU	8	RS_NOT_EXIST	Image device does not exist
		20	RS_IS_CONNECTED	Image device already connected
212	RCERR_IMAGECONN	8	RS_AUTHERR_CONNECT	Image not authorized to connect
		28	RS_DEV_INCOMPATIBLE	Image device not correct type for requested connection
		40	RS_VSWITCH_NOT_EXISTS	Virtual switch does not exist
396	RCERR_INTERNAL	nnnn	psrc	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Virtual_Network_Adapter_Create

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
image_device_number_length
image_device_number
adapter_type
network_adapter_devices
channel_path_id_length
channel_path_id

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Virtual_Network_Adapter_Create to add a virtual network interface card (NIC) to an active virtual image.

See "Virtual_Network_Adapter_Create_DM" on page 654 to add a virtual network interface card to a virtual image's directory entry.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 30.

function_name
(string,30,char43) The API function name – in this case, 'Virtual_Network_Adapter_Create'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The name of the image for which a network adapter is being defined.

image_device_number_length
(int4) Length of *image_device_number*.

image_device_number
(string,1-4,char16) The virtual device address for the new adapter.

adapter_type
(int1) The adapter type must be one of the following:

1
Defines this adapter as a simulated HiperSockets NIC. This adapter will function like the HiperSockets internal adapter (model 1732-05). A HiperSockets NIC can function without a guest LAN connection, or it can be coupled to a HiperSockets guest LAN.

2
Defines this adapter as a simulated QDIO NIC. This adapter will function like the OSA Direct Express (QDIO) adapter (model 1731-01). A QDIO NIC is functional when it is coupled either to a QDIO guest LAN or a virtual switch using Virtual_Network_Vswitch_Connect.

network_adapter_devices
(int4; range 3-3072) The number of virtual devices associated with this adapter. For a simulated HiperSockets adapter, this must be a decimal value between 3 and 3,072 (inclusive). For a simulated QDIO adapter, this must be a decimal value between 3 and 240 (inclusive).

channel_path_id_length
(int4) Length of *channel_path_id*.

channel_path_id
(string,0-2,char16) For use only when configuring a second-level z/OS system, where it is used to specify the hex CHPID numbers for the first- and second-level systems. Do not specify this parameter for z/VM, which allocates available CHPIDs by default.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
200	RCERR_IMAGEOP	4	RS_NOT_FOUND	Image not found
		12	RS_NOT_ACTIVE	Image not active
204	RCERR_IMAGEDEVU	4	RS_EXISTS	Image device already exists
		28	RS_DEV_INCOMPATIBLE	Image device already defined as type other than network adapter
396	RCERR_INTERNAL	nnnn	psrc	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Virtual_Network_Adapter_Create_DM

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
image_device_number_length
image_device_number
adapter_type
network_adapter_devices
channel_path_id_length
channel_path_id
mac_id_length
mac_id

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Virtual_Network_Adapter_Create_DM to add a virtual network interface card (NIC) to a virtual image's directory entry.

See "Virtual_Network_Adapter_Create" on page 651 to add a virtual network interface card to an active virtual image.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 33.

function_name
(string,33,char43) The API function name – in this case, 'Virtual_Network_Adapter_Create_DM'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The name of the image for which a network adapter is being defined.

image_device_number_length
(int4) Length of *image_device_number*.

image_device_number
(string,1-4,char16) The virtual device address for the new adapter.

adapter_type
(int1) The adapter type must be one of the following:

1
Defines this adapter as a simulated HiperSockets NIC. This adapter will function like the HiperSockets internal adapter (model 1732-05). A HiperSockets NIC can function without a guest LAN connection, or it can be coupled to a HiperSockets guest LAN.

2
Defines this adapter as a simulated QDIO NIC. This adapter will function like the OSA Direct Express (QDIO) adapter (model 1731-01). A QDIO NIC is functional when it is coupled either to a QDIO guest LAN or a virtual switch using Virtual_Network_Vswitch_Connect.

network_adapter_devices
(int4; range 3-3072) The number of virtual devices associated with this adapter. For a simulated HiperSockets adapter, this must be a decimal value between 3 and 3,072 (inclusive). For a simulated QDIO adapter, this must be a decimal value between 3 and 240 (inclusive).

channel_path_id_length
(int4) Length of *channel_path_id*.

channel_path_id
(string,0-2,char16) For use only when configuring a second-level z/OS system, where it is used to specify the hex CHPID numbers for the first- and second-level systems. Do not specify this parameter for z/VM, which allocates available CHPIDs by default.

mac_id_length
(int4) Length of *mac_id*.

mac_id
(string,0-6,char16) The MAC identifier.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
		8	RS_OFFLINE	Request successful; object directory offline
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		12	RS_AUTHERR_DM	Request not authorized by directory manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
400	RCERR_IMAGEDEF	4	RS_NOT_FOUND	Image definition not found
		12	RS_LOCKED	Image definition is locked
404	RCERR_IMAGEDEVD	4	RS_EXISTS	Image device already defined
		12	RS_LOCKED	Image device is locked
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available
592	RCERR_ASYNC_DM	nnnn	opid	Asynchronous operation started - product-specific asynchronous operation ID (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)

RC	RC Name	RS	RS Name	Description
596	RCERR_INTERNAL_DM	nnnn	psrc	Internal directory manager error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Virtual_Network_Adapter_Create_Extended

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
`image_device_number=`*value*
`adapter_type=`*value*
`devices=`*value*
`channel_path_id=`*value*

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Virtual_Network_Adapter_Create_Extended to add a virtual network interface card (NIC) to an active virtual image.

See "Virtual_Network_Adapter_Create_Extended_DM" on page 662 to add a virtual network interface card to a virtual image's directory entry.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 39.

function_name
(string,39,char43) The API function name – in this case, 'Virtual_Network_Adapter_Create_Extended'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The name of the image for which a network adapter is being defined.

Note: The format for specifying the following additional input parameters is *parameter_name*=*value*, followed by a null (ASCIIZ) terminator. They may be specified in any order. For more information, see "Key-value pair (KVP) input parameters" on page 51.

`image_device_number=`*value*
(string,1-4,char16) The virtual device address for the new adapter. This is a required parameter.

`adapter_type=`*value*
(string,4-12,char26) One of the following:

HIPERsockets
Defines this adapter as a simulated HiperSockets NIC. This adapter will function like the HiperSockets internal adapter (device model 1732-05). A HiperSockets NIC can function without a guest LAN connection, or it can be coupled to a HiperSockets guest LAN. This is the default if `adapter_type=`*value* is not specified.

Note: You will receive an error if you attempt to connect a simulated HiperSockets adapter to a virtual switch.

QDIO
Defines this adapter as a simulated QDIO NIC. This adapter will function like the OSA Direct Express (QDIO) adapter (device model 1732-01). A QDIO NIC is functional when it is coupled to a QDIO guest LAN or a QDIO virtual switch.

This is a required parameter.

`devices=`*value*
(string,0-4,char16; range 3-3072) The number of virtual devices associated with this adapter. For a simulated HiperSockets adapter, this must be a decimal value between 3 and 3,072 (inclusive). For a simulated QDIO adapter, this must be a decimal value between 3 and 240 (inclusive). If omitted, the default is 3.

`channel_path_id=`*value*
(string,0-2,char16) For use only when configuring a second-level z/OS system, where it is used to specify the hex CHPID numbers for the first- and second-level systems. Do not specify this parameter for z/VM, which allocates available CHPIDs by default.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Usage Notes

1. Syntax errors (RC = 24 and RS = *pprr*) are only applicable to the common input parameters. Syntax checking is *not* performed on the additional input parameters for this API. If a valid parameter is specified multiple times, the last value specified for that parameter will be used. For more information, see “Key-value pair (KVP) input parameters” on page 51.
2. If the value for an optional input parameter is not specified, the default value for the parameter, if one exists, is used.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
8	RC_ERR	3002	RS_INVALID_PARAMETER	Invalid parameter name
		3003	RS_INVALID_OPERAND	Invalid parameter operand
		3004	RS_MISSING_PARAMETER	Required parameter missing
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
		20	RS_TARGET_IMG_NOT _AUTHORIZED	Target image not authorized for function
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
200	RCERR_IMAGEOP	4	RS_NOT_FOUND	Image not found
		12	RS_NOT_ACTIVE	Image not active
204	RCERR_IMAGEDEVU	2	RS_INVALID_DEVICE	Input image device number not valid
		4	RS_EXISTS	Image device already exists
		28	RS_DEV_INCOMPATIBLE	Image device already defined as type other than network adapter

RC	RC Name	RS	RS Name	Description
396	RCERR_INTERNAL	nnnn	psrc	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Virtual_Network_Adapter_Create_Extended_DM

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
adapter_create_names_length
`image_device_number=`*value*
`adapter_type=`*value*
`devices=`*value*
`channel_path_id=`*value*
`mac_id=`*value*

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Virtual_Network_Adapter_Create_Extended_DM to add a virtual network interface card (NIC) to a virtual image's directory entry.

See "Virtual_Network_Adapter_Create_Extended" on page 658 to add a virtual network interface card to an active virtual image.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 42.

function_name
(string,42,char43) The API function name – in this case, 'Virtual_Network_Adapter_Create_Extended_DM'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The name of the image for which a network adapter is being defined.

adapter_create_names_length
(int4) Length of the remaining set of *parameter_name=value* input parameters.

Note: The format for specifying the following additional input parameters is *parameter_name=value*, followed by a null (ASCIIZ) terminator. They may be specified in any order. For more information, see "Key-value pair (KVP) input parameters" on page 51.

image_device_number=*value*
(string,1-4,char16) The virtual device address for the new adapter. This is a required parameter.

adapter_type=*value*
(string,4-12,char26) You must specify one of the following:

HIPERsockets
Defines this adapter as a simulated HiperSockets NIC. This adapter will function like the HiperSockets internal adapter (device model 1732-05). A HiperSockets NIC can function without a guest LAN connection, or it can be coupled to a HiperSockets guest LAN.

Note: You will receive an error if you attempt to connect a simulated HiperSockets adapter to a virtual switch.

QDIO
Defines this adapter as a simulated QDIO NIC. This adapter will function like the OSA Direct Express (QDIO) adapter (device model 1732-01). A QDIO NIC is functional when it is coupled either to a QDIO guest LAN or to a QDIO, virtual switch.

devices=*value*
(string,0-4,char16; range 3-3072) The number of virtual devices associated with this adapter. For a simulated HiperSockets adapter, this must be a decimal value between 3 and 3,072 (inclusive). For a simulated QDIO adapter, this must be a decimal value between 3 and 240 (inclusive). If omitted, the default is 3.

channel_path_id=*value*
(string,0-2,char16) For use only when configuring a second-level z/OS system, where it is used to specify the hex CHPID numbers for the first- and second-level systems. Do not specify this parameter for z/VM, which allocates available CHPIDs by default.

mac_id=*value*
(string,0-6,char16) The MAC identifier.

Note: This should only be specified for NIC adapter types of QDIO or Hipersockets.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Usage Notes

1. Syntax errors (RC = 24 and RS = *pprr*) are only applicable to the common input parameters. Syntax checking is *not* performed on the additional input parameters for this API. If a valid parameter is specified multiple times, the last value specified for that parameter will be used. For more information, see “Key-value pair (KVP) input parameters” on page 51.
2. Unlike Virtual_Network_Adapter_Create_Extended (where HIPERsockets is the default adapter type, if not specified), in this API you must specify an `adapter_type=`*value*.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
		8	RS_OFFLINE	Request successful; object directory offline
8	RC_ERR	3002	RS_INVALID_PARAMETER	Invalid parameter name
		3003	RS_INVALID_OPERAND	Invalid parameter operand
		3004	RS_MISSING_PARAMETER	Required parameter missing
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		12	RS_AUTHERR_DM	Request not authorized by directory manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
400	RCERR_IMAGEDEF	4	RS_NOT_FOUND	Image definition not found

RC	RC Name	RS	RS Name	Description
		12	RS_LOCKED	Image definition is locked
404	RCERR_IMAGEDEVD	4	RS_EXISTS	Image device already defined
		12	RS_LOCKED	Image device is locked
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available
592	RCERR_ASYNC_DM	nnnn	opid	Asynchronous operation started - product-specific asynchronous operation ID (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
596	RCERR_INTERNAL_DM	nnnn	psrc	Internal directory manager error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Virtual_Network_Adapter_Delete

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
image_device_number_length
image_device_number

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Virtual_Network_Adapter_Delete to remove a virtual network interface card (NIC) from an active virtual image.

See "Virtual_Network_Adapter_Delete_DM" on page 669 to remove a virtual network interface card from a virtual image's directory entry.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 30.

function_name
(string,30,char43) The API function name – in this case, 'Virtual_Network_Adapter_Delete'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The name of the image for which the network adapter is being removed.

image_device_number_length
(int4) Length of *image_device_number*.

image_device_number
(string,1-4,char16) The virtual device number of the base address for the adapter to be deleted.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
200	RCERR_IMAGEOP	4	RS_NOT_FOUND	Image not found
		12	RS_NOT_ACTIVE	Image not active

RC	RC Name	RS	RS Name	Description
204	RCERR_IMAGEDEVU	8	RS_NOT_EXIST	Image device does not exist
		28	RS_DEV_INCOMPATIBLE	Image device already defined as type other than network adapter
396	RCERR_INTERNAL	nnnn	psrc	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Virtual_Network_Adapter_Delete_DM

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
image_device_number_length
image_device_number

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Virtual_Network_Adapter_Delete_DM to remove a virtual network interface card (NIC) from a virtual image's directory entry.

See "Virtual_Network_Adapter_Delete" on page 666 to remove a virtual network interface card from an active virtual image.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 33.

function_name
(string,33,char43) The API function name – in this case, 'Virtual_Network_Adapter_Delete_DM'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The name of the image for which the network adapter is being removed.

image_device_number_length
(int4) Length of *image_device_number*.

image_device_number
(string,1-4,char16) The virtual device number of the base address for the adapter to be deleted.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
		8	RS_OFFLINE	Request successful; object directory offline
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		12	RS_AUTHERR_DM	Request not authorized by directory manager
		16	RS_AUTHERR_SERVER	Request not authorized by server

RC	RC Name	RS	RS Name	Description
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
400	RCERR_IMAGEDEF	4	RS_NOT_FOUND	Image definition not found
		12	RS_LOCKED	Image definition is locked
404	RCERR_IMAGEDEVD	8	RS_NOT_DEFINED	Image device not defined
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available
592	RCERR_ASYNC_DM	nnnn	opid	Asynchronous operation started - product-specific asynchronous operation ID (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
596	RCERR_INTERNAL_DM	nnnn	psrc	Internal directory manager error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Virtual_Network_Adapter_Disconnect

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
image_device_number_length
image_device_number

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Virtual_Network_Adapter_Disconnect to disconnect a virtual network adapter on an active virtual image from a virtual network LAN or virtual switch.

See "Virtual_Network_Adapter_Disconnect_DM" on page 675 to remove virtual network LAN or virtual switch connection from a virtual network adapter definition in a virtual image's directory entry.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 34.

function_name
(string,34,char43) The API function name – in this case, 'Virtual_Network_Adapter_Disconnect'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The name of the user or profile from which virtual network adapter guest LAN connection information will be removed.

image_device_number_length
(int4) Length of *image_device_number*.

image_device_number
(string,1-4,char16) Specifies the virtual device address of the connected adapter.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
200	RCERR_IMAGEOP	12	RS_NOT_ACTIVE	Image not active

RC	RC Name	RS	RS Name	Description
204	RCERR_IMAGEDEVU	8	RS_NOT_EXIST	Image device does not exist
		28	RS_DEV_INCOMPATIBLE	Image device already defined as type other than network adapter
		48	RS_IS_DISCONNECTED	Virtual network adapter is already disconnected
212	RCERR_IMAGECONN	20	RS_OWNER_NOT_ACTIVE	Requested LAN owner not active
396	RCERR_INTERNAL	nnnn	psrc	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Virtual_Network_Adapter_Disconnect_DM

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
image_device_number_length
image_device_number

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Virtual_Network_Adapter_Disconnect_DM to remove a virtual network LAN or virtual switch connection from a virtual network adapter definition in a virtual image's directory entry.

See "Virtual_Network_Adapter_Disconnect" on page 672 to disconnect a virtual network adapter on an active virtual image from a virtual network LAN or virtual switch.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 37.

function_name
(string,37,char43) The API function name – in this case, 'Virtual_Network_Adapter_Disconnect_DM'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The name of the user or profile from which virtual network adapter guest LAN connection information will be removed.

image_device_number_length
(int4) Length of *image_device_number*.

image_device_number
(string,1-4,char16) Specifies the virtual device address of the connected adapter.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
		8	RS_OFFLINE	Request successful; object directory offline
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		12	RS_AUTHERR_DM	Request not authorized by directory manager
		16	RS_AUTHERR_SERVER	Request not authorized by server

RC	RC Name	RS	RS Name	Description
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
400	RCERR_IMAGEDEF	4	RS_NOT_FOUND	Image definition not found
		12	RS_LOCKED	Image definition is locked
404	RCERR_IMAGEDEVD	8	RS_NOT_EXIST	Image device does not exist
412	RCERR_IMAGECONND	16	RS_NO_MATCH	Parameters do not match existing directory statement
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available
592	RCERR_ASYNC_DM	nnnn	opid	Asynchronous operation started - product-specific asynchronous operation ID (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
596	RCERR_INTERNAL_DM	nnnn	psrc	Internal directory manager error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Virtual_Network_Adapter_Query

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
image_device_number_length
image_device_number

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
adapter_array_length
adapter_array (1)
    adapter_structure (2)
        adapter_structure_length
        image_device_number_length
        image_device_number
        adapter_type
        network_adapter_devices
        adapter_status
        lan_owner_length
        lan_owner
        lan_name_length
        lan_name

Note:

1. An array consists of zero or more of its components.
2. A structure consists of one set of its components.

Purpose

Use Virtual_Network_Adapter_Query to obtain information about the specified adapter for an active virtual image.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 29.

function_name
(string,29,char43) The API function name – in this case, 'Virtual_Network_Adapter_Query'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The virtual image name of the owner of the adapter.

image_device_number_length
(int4) Length of *image_device_number*.

image_device_number
(string,1-4,char16) The virtual device address of the adapter.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

adapter_array_length
(int4) Length of *adapter_array_array*.

adapter_array
(array) An array consisting of zero or more instances of *adapter_structure*, as follows:

adapter_structure
(structure) A structure consisting of one set of the following parameters:

adapter_structure_length
(int4) The combined length of the remaining parameters in *adapter_structure* (not including this parameter).

image_device_number_length
(int4) Length of *image_device_number*.

image_device_number
(string,1-4,char16) The virtual device address of the adapter.

adapter_type
(int1) The adapter type. The possible values are:

1
HiperSockets

2
QDIO

network_adapter_devices
(int4) The number of devices associated with the adapter.

adapter_status
(int1) The adapter status. The possible values are:

0
Not coupled.

1
Coupled but not active.

2
Coupled and active.

lan_owner_length
(int4) Length of *lan_owner*.

lan_owner
(string,0-8,char42 plus blank) The name of virtual image owning the guest LAN to which the adapter is connected. This value will be blanks if the adapter is not connected.

lan_name_length
(int4) Length of *lan_name*.

lan_name
(string,0-8,char36 plus blank $#@) The name of the guest LAN to which the adapter is connected. This value will be blanks if the adapter is not connected.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager

RC	RC Name	RS	RS Name	Description
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
200	RCERR_IMAGEOP	4	RS_NOT_FOUND	Image not found
		12	RS_NOT_ACTIVE	Image not active
204	RCERR_IMAGEDEVU	8	RS_NOT_EXIST	Image device does not exist
212	RCERR_IMAGECONN	8	RS_ADAPTER_NOT_EXIST	Adapter does not exist
396	RCERR_INTERNAL	nnnn	psrc	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Virtual_Network_Adapter_Query_Extended

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
virtual_network_adapter_query_names_length
`image_device_number=`*value*

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
virtual_network_adapter_data_length
virtual_network_adapter_data

Purpose

Use Virtual_Network_Adapter_Query_Extended to obtain information about the specified adapter for an active virtual image.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 29.

function_name
(string,29,char43) The API function name – in this case, 'Virtual_Network_Adapter_Query'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See “Client Authentication” on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The virtual image name of the owner of the adapter.

virtual_network_adapter_query_names_length
(int4) Length of the remaining set of *parameter_name=value* input parameters.

Note: The format for specifying the following additional input parameters is *parameter_name=value*, followed by a null (ASCIIZ) terminator. They may be specified in any order. For more information, see "Key-value pair (KVP) input parameters" on page 51.

`image_device_number=`*value*
(string,0-4,char16) The virtual device address of the adapter, or an asterisk which indicates that information for all network adapter devices defined for the target image should be returned.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

virtual_network_adapter_data_length
(int4) Length of *virtual_network_adapter_data*.

virtual_network_adapter_data
(string) A series of null-terminated strings, each containing "*output_subkeyword=*" followed by either a value or a keyword, as shown in Table 20 on page 683.

Table 20. Output Keywords and Values for Virtual_Network_Adapter_Query_Extended

output_keyword_parameter=	**Blank-delimited *output_subkeyword=value* pairs**
adapter_count=	Number of adapter devices returned in the response. Each adapter begins with the adapter_address keyword.
adapter_address=	The virtual device address of the adapter.

Table 20. Output Keywords and Values for Virtual_Network_Adapter_Query_Extended (continued)	
output_keyword_parameter=	**Blank-delimited *output_subkeyword=value* pairs**
adapter_info_end	Indicates the end of the adapter information. No value or equal sign follows this keyword.
adapter_status=	The adapter port or NIC status, as a 2-digit hexadecimal value (for example, 02). The possible values are the following. For a guest NIC: **X'00'** NIC is not coupled **X'01'** Coupled but not active **X'02'** Coupled and active For a virtual switch network connection or a HiperSockets Bridge port: **X'01'** Attached to a controller but not active **X'02'** Attached and active
adapter_type=	The adapter type. Possible values: **1** HiperSockets NIC **2** QDIO NIC **3** QDIO VSWITCH **6** IQD Bridge Port

Table 20. Output Keywords and Values for Virtual_Network_Adapter_Query_Extended (continued)

output_keyword_parameter=	**Blank-delimited *output_subkeyword=value* pairs**
device_options=	Device options are returned as an 8-character string representing 4 hexadecimal bytes. Byte 1: **80** Broadcast **40** Ethernet **20** IPv4 **10** IPv6 **08** Multicast **04** Promiscuous enabled **02** Promiscuous denied **01** VLAN enabled Bytes 2-4 are reserved for future use. For a HiperSockets Logical Port, these bytes are not used.
extended_port_status=	Extended port status is a 2-character string representing a hexadecimal byte: **80** Isolation status ON (0 if guest NIC not coupled) **40** VEPA status ON **20** Uplink NIC Port (0 if guest NIC not coupled)
guest_trans_priority_level=	Guest transmission priority level is a 2-character string representing a hexadecimal byte: **00** z/VM **01** High **02** Normal **03** Low
lan_name=	The name of the guest LAN to which the adapter is connected. This keyword/value pair is not returned if the adapter is not connected.
lan_owner=	The name of virtual image owning the guest LAN to which the adapter is connected. This keyword/value pair is not returned if the adapter is not connected.

Table 20. Output Keywords and Values for Virtual_Network_Adapter_Query_Extended (continued)	
output_keyword_parameter=	**Blank-delimited *output_subkeyword=value* pairs**
mac_count=	The number of sets of IP information which follow this keyword. One set is provided for each MAC address associated with an IP address. If there is no MAC address information, then the MAC count is zero and the mac_count= keyword will not be produced. No sets of IP information related to MACs will follow, and neither will a mac_info_end keyword follow. Each set of information begins with the mac_address= keyword.
mac_address=	The 6-digit MAC address as a 12-character string representing 6 hexadecimal bytes. This is the first keyword in the set for a particular MAC address.
mac_address_type=	The MAC or IP address type as a 2-character string: **00** Unicast MAC or IP address **01** Multicast MAC or IP address **02** Broadcast MAC
mac_info_end	Indicates the end of the MAC information. No value or equal sign follows this keyword.
mac_ip_address=	IP address associated with this MAC address. This key/value pair is not returned if the IP address is not known.
mac_ip_version=	The MAC or IP address value: **4** IPv4 **6** IPv6 This key/value pair is not returned if the IP address is not known.
mac_status=	The IP or MAC address status: **80** Local **40** Remote **20** Manual **10** Owner **08** Error or External Conflict **04-01** Reserved for future use
network_device_count=	The number of devices associated with the adapter.

Table 20. Output Keywords and Values for Virtual_Network_Adapter_Query_Extended (continued)	
output_keyword_parameter=	**Blank-delimited *output_subkeyword=value* pairs**
port_type=	Port type: **0** Undefined (VLAN Unaware or guest NIC not coupled) **1** Access **2** Trunk
port_nic_flags=	Port or NIC flags is a 2-character string representing a hexadecimal byte: **80** NIC distribution ON The remainder of the byte is reserved for future use.
router_status=	Router Status: **80** Primary **40** Secondary **20** Multicast **10** MAC address protection ON **08-01** Reserved for future use

The following is an example of response for a single adapter response with one IP address:

```
adapter_count=2
adapter_address=0600
port_type=0
extended_port_status=00
guest_trans_priority_level=00
port_nic_flags=80
adapter_type=2
network_device_count=3
adapter_status=02
lan_owner=SYSTEM
lan_name=VSW1
device_options=B9000000
router_status=00
mac_count=1
mac_address=02000B000036
mac_address_type=00
mac_status=10
mac_ip_version=4
mac_ip_address=9.60.18.147
mac_info_end
adapter_info_end
```

The following is an example of a response for multiple adapters with the first one having multiple IP addresses associated with it and the second having a single IP address:

```
adapter_count=2
adapter_address=0600
port_type=0
extended_port_status=00
guest_trans_priority_level=01
port_nic_flags=00
adapter_type=2
network_device_count=3
adapter_status=02
lan_owner=SYSTEM
lan_name=VSW1
device_options=B9000000
router_status=00
mac_count=3
mac_address=02000B000036
mac_address_type=00
mac_status=10
mac_ip_version=4
mac_ip_address=9.60.18.147
mac_address=02000B000036
mac_address_type=00
mac_status=90
mac_ip_version=6
mac_ip_address=fe80::200:b00:400:36
mac_address=3333FF000036
mac_address_type=01
mac_status=80
mac_ip_version=6
mac_ip_address=ff02::1:ff00:36
mac_info_end
adapter_info_end
adapter_address=0800
port_type=1
extended_port_status=00
guest_trans_priority_level=00
port_nic_flags=00
adapter_type=2
network_device_count=3
adapter_status=02
lan_owner=SYSTEM
lan_name=VSW2
device_options=C1000000
router_status=00
mac_count=1
mac_address=01005E000001
mac_address_type=01
mac_status=00
mac_info_end
adapter_info_end
```

Usage Notes

1. The number of keywords provided for the Adapter information and the IP information can be changed in the future and additional sets of information can be added. The user of the API should code to be insensitive to additions of keywords or additional sections.

2. Syntax errors (RC = 24 and RS = *pprr*) are only applicable to the common input parameters. Syntax checking is *not* performed on the additional input parameters for this API. If a valid parameter is specified multiple times, the last value specified for that parameter will be used. For more information, see "Key-value pair (KVP) input parameters" on page 51.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
8	RC_ERR	3004	RS_MISSING_PARAMETER	Required parameter missing
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
200	RCERR_IMAGEOP	4	RS_NOT_FOUND	Image not found
		12	RS_NOT_ACTIVE	Image not active
204	RCERR_IMAGEDEVU	8	RS_NOT_EXIST	Image device does not exist
212	RCERR_IMAGECONN	8	RS_ADAPTER_NOT_EXIST	Adapter does not exist
396	RCERR_INTERNAL	nnnn	psrc	Internal system error - product-specific return code (See "Internal Return Codes (RC = 396, 592, or 596)" on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Virtual_Network_LAN_Access

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
lan_name
lan_owner
access_op
access_user
promiscuity

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Virtual_Network_Lan_Access to grant users access to a restricted virtual network LAN.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 26.

function_name
(string,26,char43) The API function name – in this case, 'Virtual_Network_LAN_Access'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) This must match an entry in the authorization file that also contains the *authenticated_userid* and the *function_name* (Virtual_Network_Lan_Access).

lan_name
(string,1-8,char36 plus $#@) The name of the LAN to which access is being granted or revoked, followed by a null (ASCIIZ) terminator.

lan_owner
(string,1-8,char36) The virtual image owning the guest LAN segment to be created, followed by a null (ASCIIZ) terminator.

access_op
One of the following, followed by a null (ASCIIZ) terminator:

- (string,5,GRANT) Grant access.
- (string,6,REVOKE) Revoke access.

access_user
(string,1-8,char36) Virtual image to be granted access to the LAN, followed by a null (ASCIIZ) terminator.

promiscuity
One of the following, followed by a null (ASCIIZ) terminator:

- (string,14,NONPROMISCUOUS) Nonpromiscuous access.
- (string,11,PROMISCUOUS) Promiscuous access.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Usage Notes

1. Because a LAN is defined by both its *lan_name* and *lan_owner*, an error in either one of these fields may result in an "Invalid LAN ID" return code.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
8	RC_ERR	2	RS_INVALID_USER	Invalid access user
		3	RS_INVALID_OP	Invalid op value
		4	RS_INVALID_PRO	Invalid promiscuity value
		2783	RS_INVALID_LANID	Invalid LAN ID
		2795	RS_INVALID_LAN_PARM	Invalid LAN parameter
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
396	RCERR_INTERNAL	nnnn	psrc	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Virtual_Network_LAN_Access_Query

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
lan_name
lan_owner

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
authorized_users_array (1)
    authorized_user_record

Note:

1. An array consists of zero or more of its components.

Purpose

Use Virtual_Network_LAN_Access_Query to query which users are authorized to access a specified restricted virtual network LAN.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 32.

function_name
(string,26,char43) The API function name – in this case, 'Virtual_Network_LAN_Access_Query'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).

- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) This must match an entry in the authorization file that also contains the *authenticated_userid* and the *function_name* (Virtual_Network_Lan_Access_Query).

lan_name
(string,1-8,char36 plus $#@) The name of the LAN being queried, followed by a null (ASCIIZ) terminator.

lan_owner
(string,1-8,char36) The owner of the LAN being queried, followed by a null (ASCIIZ) terminator.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

authorized_users_array
(array) An array consisting of zero or more instances of *authorized_user_record*, as follows:

authorized_user_record
(string,1-23,char36) An authorized user name, followed by a blank, then the promiscuity value – for example, "LANUSER1 PROMISCUOUS" or "LANUSER2 NONPROMISCUOUS". Each record is followed by a null (ASCIIZ) terminator.

Usage Notes

1. Because a LAN is defined by both its *lan_name* and *lan_owner*, an error in either one of these fields may result in an "Invalid LAN ID" return code.

2. If the LAN whose access is being queried is unrestricted, an RC=4 (RC_WNG), RS=5 (RS_UNRESTRICTED_LAN) return code will be returned. There will be *no* list of authorized users returned in this case, as *all* users are authorized to access an unrestricted LAN.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
4	RC_WNG	5	RS_UNRESTRICTED_LAN	Unrestricted LAN
		6	RS_NO_USERS	No authorized users
8	RC_ERR	2783	RS_INVALID_LANID	Invalid LAN ID
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
212	RCERR_IMAGECONN	12	RS_LAN_NOT_EXIST	LAN does not exist
		16	RS_NOT_EXIST	LAN owner LAN name does not exist
396	RCERR_INTERNAL	nnnn	psrc	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Virtual_Network_LAN_Create

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
lan_name_length
lan_name
lan_owner_length
lan_owner
lan_type
transport_type

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Virtual_Network_LAN_Create to create a virtual network LAN.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 26.

function_name
(string,26,char43) The API function name – in this case, 'Virtual_Network_LAN_Create'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The name of the image for which a LAN connection is being created.

lan_name_length
(int4) Length of *lan_name*.

lan_name
(string,1-8,char36 plus $#@) The name of the guest LAN segment to be created.

lan_owner_length
(int4) Length of *lan_owner*.

lan_owner
One of the following:

- (string,1-8,char42) The virtual image owning the guest LAN segment to be created. Note that specifying a virtual image as the *lan_owner* will result in a LAN creation that is *not* persistent across IPLs of the system (CP).
- (string,6,SYSTEM) Specifying 'SYSTEM' as *lan_owner* will result in a LAN creation that is persistent across IPLs of the system.

lan_type
(int1) The type of guest LAN segment. This must be one of the following:

1
Defines this adapter as an unrestricted simulated HiperSockets NIC. This adapter will function like the HiperSockets internal adapter (model 1732-05). A HiperSockets NIC can function without a guest LAN connection, or it can be coupled to a HiperSockets guest LAN.

2
Defines this adapter as an unrestricted simulated QDIO NIC. This adapter will function like the OSA Direct Express (QDIO) adapter (model 1731-01). A QDIO NIC is functional when it is coupled either to a QDIO guest LAN or to a virtual switch using Virtual_Network_Vswitch_Connect.

3
Defines this adapter as a restricted simulated HiperSockets NIC. This adapter will function like the HiperSockets internal adapter (model 1732-05). A HiperSockets NIC can function without a guest LAN connection, or it can be coupled to a HiperSockets guest LAN.

4
Defines this adapter as a restricted simulated QDIO NIC. This adapter will function like the OSA Direct Express (QDIO) adapter (model 1731-01). A QDIO NIC is functional when it is coupled either to a QDIO guest LAN or to a virtual switch using Virtual_Network_Vswitch_Connect.

transport_type
(int1) Specifies the transport mechanism to be used for guest LANs and virtual switches, as follows:

0
Unspecified

1
IP – Reference all target nodes on LAN or switch using IP addresses.

2
Ethernet – Reference all target nodes on LAN or switch using MAC addresses.

If not specified, IP is assumed.

Note: If *lan_type* is specified as HIPERSOCKETS then *transport_type* can only be specified as IP (and not as ETHERNET). If you specify *transport_type* as ETHERNET, a syntax error will be returned.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
		20	RS_VMLAN_CREATED	Request successful; new virtual network LAN created
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
212	RCERR_IMAGECONN	20	RS_OWNER_NOT_ACTIVE	Requested LAN owner not active
		24	RS_LAN_NAME_EXISTS	LAN name already exists with different attributes
396	RCERR_INTERNAL	nnnn	psrc	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)

RC	RC Name	RS	RS Name	Description
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Virtual_Network_LAN_Delete

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
lan_name_length
lan_name
lan_owner_length
lan_owner

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Virtual_Network_LAN_Delete to delete a virtual network LAN.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 26.

function_name
(string,26,char43) The API function name – in this case, 'Virtual_Network_LAN_Delete'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See “Client Authentication” on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The name of the image for which a LAN connection is being deleted.

lan_name_length
(int4) Length of *lan_name*.

lan_name
(string,1-8,char36 plus $#@) The name of the guest LAN segment to be deleted.

lan_owner_length
(int4) Length of *lan_owner*.

lan_owner
One of the following:

- (string,1-8,char42) The virtual image owning the guest LAN segment to be deleted.
- (string,6,SYSTEM) Specifying 'SYSTEM' as *lan_owner* will result in deletion of the LAN now and from all future system (CP) IPLs.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager

RC	RC Name	RS	RS Name	Description
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
212	RCERR_IMAGECONN	12	RS_LAN_NOT_EXIST	LAN does not exist
		96	RS_UNKNOWN	Unknown reason
396	RCERR_INTERNAL	nnnn	psrc	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Virtual_Network_LAN_Query

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
lan_name_length
lan_name
lan_owner_length
lan_owner

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
lan_array_length
lan_array (1)
    lan_structure (2)
        lan_structure_length
        lan_name_length
        lan_name
        lan_owner_length
        lan_owner
        lan_type
        connected_adapter_array_length
        connected_adapter_array (1)
            connected_adapter_structure (2)
                connected_adapter_structure_length
                adapter_owner_length
                adapter_owner
                image_device_number_length
                image_device_number

Note:

1. An array consists of zero or more of its components.
2. A structure consists of one set of its components.

Purpose

Use Virtual_Network_LAN_Query to obtain information about a virtual network LAN.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 25.

function_name
(string,25,char43) The API function name – in this case, 'Virtual_Network_LAN_Query'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) This must match an entry in the authorization file that also contains the *authenticated_userid* and the *function_name* (Virtual_Network_LAN_Query).

lan_name_length
(int4) Length of *lan_name*.

lan_name
One of the following:

- (string,1-8,char36 plus $#@) The name of the guest LAN segment to be queried.
- (string,1,*) A request is made for information about *all* guest LAN segments.

lan_owner_length
(int4) Length of *lan_owner*.

lan_owner
One of the following:

- (string,0-8,char42) The name of the virtual image owning the guest LAN segment.
- (string,1,*) A request is made for *all* qualified guest LAN segments.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

lan_array_length
(int4) Length of *lan_array*.

lan_array
(array) An array consisting of zero or more instances of *lan_structure*, as follows:

lan_structure
(structure) A structure consisting of one set of the following parameters:

lan_structure_length
(int4) The combined length of the remaining parameters in *lan_structure* (not including this parameter).

lan_name_length
(int4) Length of *lan_name*.

lan_name
(string,1-8,char36 plus $#@) The name of the guest LAN.

lan_owner_length
(int4) Length of *lan_owner*.

lan_owner
(string,1-8,char42) The name of the virtual image owning the guest LAN segment.

lan_type
(int1) The type of guest LAN segment. This will be one of the following:

1
Defines this adapter as a simulated HiperSockets NIC. This adapter will function like the HiperSockets internal adapter (model 1732-05). A HiperSockets NIC can function without a guest LAN connection, or it can be coupled to a HiperSockets guest LAN.

2
Defines this adapter as a simulated QDIO NIC. This adapter will function like the OSA Direct Express (QDIO) adapter (model 1731-01). A QDIO NIC is functional when it is coupled either to a QDIO guest LAN or to a virtual switch using Virtual_Network_Vswitch_Connect.

connected_adapter_array_length
(int4) Length of *connected_adapter_array*.

connected_adapter_array
(array) An array consisting of zero or more instances of *connected_adapter_structure*, as follows:

connected_adapter_structure
(structure) A structure consisting of one set of the following parameters:

connected_adapter_structure_length
(int4) The combined length of the remaining parameters in *connected_adapter_structure* (not including this parameter).

adapter_owner_length
(int4) Length of *adapter_owner*.

adapter_owner
(string,1-8,char42) The owner of the connected adapter.

image_device_number_length
(int4) Length of *image_device_number_owner*.

image_device_number
(string,1-4,char16) The virtual device address of the connected adapter.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
212	RCERR_IMAGECONN	12	RS_LAN_NOT_EXIST	LAN does not exist
		16	RS_NOT_EXIST	LAN owner LAN name does not exist
		96	RS_UNKNOWN	Unknown reason
396	RCERR_INTERNAL	nnnn	psrc	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Virtual_Network_OSA_Query

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
osa_info_array (1)
  osa_info_structure (2)
    osa_address
    osa_status
    osa_type
    chpid_address
    agent_status
    trace_size

Note:

1. An array consists of zero or more of its components.
2. A structure consists of one set of its components.

Purpose

Use Virtual_Network_OSA_Query to query data about real OSA devices.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 25.

function_name
(string,25,char43) The API function name – in this case, 'Virtual_Network_OSA_Query'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) This must match an entry in the authorization file that also contains the *authenticated_userid* and the *function_name* (Virtual_Network_OSA_Query).

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

osa_info_array
(array) An array consisting of zero or more instances of *osa_info_structure*, with each structure terminated by a null (ASCIIZ) character, as follows:

osa_info_structure
(structure) A structure consisting of one set of the following parameters, with a blank separating each parameter:

osa_address
(string,4,char16) The device address.

osa_status
(string,4-16,char42) The following values are possible:

FREE

OFFLINE

BOXED

ATTACHED*userid*
ATTACHED and *userid* (the userid of the device) are concatenated.

BOX/ATTC*userid*
BOX/ATTC and *userid* (the userid of the device) are concatenated.

osa_type
(string,3-7,char26) The following values are possible:

HIPER

OSA

OSH

OSN

UNKNOWN
If *osa_type* is UNKNOWN, the only information returned will be *osa_address*, *osa_status*, *osa_type* (UNKNOWN), and *agent_status*. For example:

```
1111 OFFLINE UNKNOWN   NO
```

chpid_address
(string,2,char16) The CHPID address.

agent_status
(string,2-3,char42) The following values are possible:

YES

NO

trace_size
(string,0-4,char10) A number between 0 and 4095 specifying the number of pages allocated for an internal trace table to track trace events pertaining to EQDIO (OSH type) on the OSA device. Non-OSH type OSA devices will have a value of 0.

Usage Notes

1. Syntax errors (RC = 24 and RS = *pprr*) are only applicable to the common input parameters. Syntax checking is *not* performed on the additional input parameters for this API. If a valid parameter is specified multiple times, the last value specified for that parameter will be used. For more information, see “Key-value pair (KVP) input parameters” on page 51.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
4	RC_WNG	4	RS_NO_OSAS	No OSAs on system
8	RC_ERR	3002	RS_INVALID_PARAMETER	Invalid parameter name
		3003	RS_INVALID_OPERAND	Invalid parameter operand
		3004	RS_MISSING_PARAMETER	Required parameter missing
24	RCERR_SYNTAX	*pprr*	*pprr*	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager

RC	RC Name	RS	RS Name	Description
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
396	RCERR_INTERNAL	*nnnn*	*psrc*	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Virtual_Network_VLAN_Query_Stats

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
`userid=`*value*
`VLAN_id=`*value*
`device=`*value*
`fmt_version=`*value*

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
port_nic_array_length
port_nic_array (1)
    port_nic_structure (2)
        port_nic_structure_length
        port_nic_info_structure (2)
            type
            port_name or *nic_addr*
            port_nic_num
        pseg_array_length
        pseg_array (1)
            pseg_structure (2)
                pseg_vlanid
                pseg_rx
                pseg_rx_disc
                pseg_tx
                pseg_tx_disc

Note:

1. An array consists of zero or more of its components.
2. A structure consists of one set of its components.

Purpose

Use Virtual_Network_VLAN_Query_Stats to query a virtual LAN's statistics.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 32.

function_name
(string,32,char43) The API function name – in this case, 'Virtual_Network_VLAN_Query_Stats'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) Used strictly for authorization, i.e. the authenticated user must have authorization to perform this function for this target.

Note: The format for specifying the following additional input parameters is *parameter_name=value*, followed by a null (ASCIIZ) terminator. They may be specified in any order. For more information, see "Key-value pair (KVP) input parameters" on page 51.

`fmt_version=`*value*
(string,0-10,char10) The format version of this API, for calls to DIAGNOSE X'26C'. For V6.2, the supported format version value is 4. This is an optional parameter.

`userid=`*value*
(string,1-8,char42) The name of the virtual machine. This input parameter is required.

`VLAN_id=`*value*
(string,0-8,char42) The VLAN ID for which you are querying information. If not specified, information for *all* VLANs will be returned.

`device=`*value*
(string,0-4,char26) Specifies whether information is requested for the ports, the virtual NICs or both, as follows:

PORT

NIC

BOTH

If not specified, BOTH is the default.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

port_nic_array_length
(int4) Length of *port_nic_array*.

port_nic_array
(array) An array consisting of zero or more instances of *port_nic_structure*, as follows:

port_nic_structure
(structure) A structure consisting of one set of the following parameters:

port_nic_structure_length
(int4) The combined length of the remaining parameters in *port_nic_structure* (not including this parameter).

port_nic_info_structure
(structure) A null-terminated structure consisting of one set of the following parameters, with a blank separating each parameter:

type
(string,3-4,char26) One of the following:

PORT

NIC

port_name* or *nic_addr
(string,1-8,char36 plus $#@*) If *type*=PORT, the name of the port. (If the port name is blank, then the string "*noname*" will be returned here.)

If *type*=NIC, the virtual address of the port.

port_nic_num
(string,1-10,char10) The port number (0 if never coupled).

pseg_array_length
(int4) Length of *pseg_array*.

pseg_array
(array) An array consisting of zero or more instances of *pseg_structure*, with each structure terminated by a null (ASCIIZ) character, as follows:

pseg_structure
(structure) A null-terminated structure consisting of one set of the following parameters, with a blank separating each parameter:

pseg_vlanid
(string,1-10,char10) The VLAN ID, or 0. (0 is returned for a VLAN UNAWARE virtual switch, or for a VLAN AWARE virtual switch with the *VLAN_counters* attribute set to OFF.)

pseg_rx
(string,1-10,char10) Received frames.

pseg_rx_disc
(string,1-10,char10) Received frames discarded.

pseg_tx
(string,1-10,char10) Transmitted frames.

pseg_tx_disc
(string,1-10,char10) Transmitted frames discarded.

Usage Notes

1. Syntax errors (RC = 24 and RS = *pprr*) are only applicable to the common input parameters. Syntax checking is *not* performed on the additional input parameters for this API. If a valid parameter is specified multiple times, the last value specified for that parameter will be used. For more information, see "Key-value pair (KVP) input parameters" on page 51.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
8	RC_ERR	8	RS_NOT_AVAILABLE	Input parameter value not supported
		36	RS_LENGTH_NOT_VALID	Specified length is not valid
		3002	RS_INVALID_PARAMETER	Invalid parameter name
		3003	RS_INVALID_OPERAND	Invalid parameter operand
		3004	RS_MISSING_PARAMETER	Required parameter missing
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
212	RCERR_IMAGECONN	48	RS_VLAN_NOT_FOUND	VLAN does not exist
396	RCERR_INTERNAL	nnnn	psrc	Internal system error - product-specific return code (See "Internal Return Codes (RC = 396, 592, or 596)" on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data

RC	RC Name	RS	RS Name	Description
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Virtual_Network_Vswitch_Create

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
switch_name_length
switch_name
real_device_address_length
real_device_address
port_name_length
port_name
controller_name_length
controller_name
connection_value
queue_memory_limit
routing_value
transport_type
vlan_id
port_type
update_system_config_indicator
system_config_name_length
system_config_name
system_config_type_length
system_config_type
parm_disk_owner_length
parm_disk_owner
parm_disk_number_length
parm_disk_number
parm_disk_password_length
parm_disk_password
alt_system_config_name_length
alt_system_config_name
alt_system_config_type_length
alt_system_config_type
alt_parm_disk_owner_length
alt_parm_disk_owner
alt_parm_disk_number_length
alt_parm_disk_number
alt_parm_disk_password_length
alt_parm_disk_password
gvrp_value
native_vlanid

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length

request_id

return_code

reason_code

Purpose

Use Virtual_Network_Vswitch_Create to create a virtual switch.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 30.

function_name
(string,30,char43) The API function name – in this case, 'Virtual_Network_Vswitch_Create'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The virtual image name of the owner of the virtual switch.

switch_name_length
(int4) Length of *switch_name*.

switch_name
(string,1-8,char36 plus @#$_) The name of the virtual switch segment.

real_device_address_length
(int4) Length of *real_device_address*.

real_device_address
(string,0-14,char16 plus blank) The real device address of a real OSA-Express QDIO device used to create the switch to the virtual adapter. A maximum of three device addresses, all 1-4 characters in length, may be specified, delimited by blanks. "NONE" may also be specified. (The default value is "NONE".)

port_name_length
(int4) Length of *port_name*.

port_name
(string,0-26,char42 plus blank) The name used to identify the OSA Expanded adapter. A maximum of three port names, all 1-8 characters in length, may be specified, delimited by blanks. The default value is *switch_name*.

controller_name_length
(int4) Length of *controller_name*.

controller_name
One of the following:

- (string,0-8,char42) The userid controlling the real device.
- (string,1,*) Specifies that any available controller may be used.

The default value is '*'.

connection_value
(int1) This can be one of the following values:

0
Unspecified

1
Activate the real device connection.

2
Do *not* activate the real device connection.

If not specified, a value of 1 (activate) is assumed.

queue_memory_limit
(int4) A number between 1 and 8 specifying the QDIO buffer size in megabytes. If unspecified, the default is 8.

routing_value
(int1) Specifies whether the OSA-Express QDIO device will act as a router to the virtual switch, as follows:

0
Unspecified

Note that when *transport_type* is 2 (ETHERNET), *routing_value* must be unspecified.

1
NONROUTER – The OSA-Express device identified in *real_device_address* will *not* act as a router to the virtual switch.

2
PRIROUTER – The OSA-Express device identified in *real_device_address* will act as a primary router to the virtual switch.

transport_type
(int1) Specifies the transport mechanism to be used for the virtual switch, as follows:

0
Unspecified

1
IP

2
ETHERNET

vlan_id
(int4) The VLAN ID. This can be any of the following values:

-1
The VLAN ID is not specified.

0
UNAWARE

1 - 4094
Any number in this range is a valid VLAN ID.

Note: If neither *vlan_id* nor *port_type* are specified, then *vlan_id* defaults to UNAWARE.

port_type
(int1) Specifies the port type, as follows:

0
Unspecified

1
ACCESS

2
TRUNK

Note:

1. If *vlan_id* is specified but *port_type* is not specified, then *port_type* will default to ACCESS.
2. If *vlan_id* is specified as UNAWARE, then you cannot specify *port_type*, *gvrp_value* or *native_vlanid*.

update_system_config_indicator
(int1) This can be one of the following values:

0
Unspecified.

1
Create a virtual switch on the active system.

2
Create a virtual switch on the active system and add the virtual switch definition to the system configuration file.

3
Add the virtual switch definition to the system configuration file.

If not specified, the default is 1.

system_config_name_length
(int4) Length of *system_config_name*.

system_config_name
(string,0-8,char42) File name of the system configuration file. The default is set by the "System_Config_File_Name =" statement in the DMSSICNF COPY file. (See the "Default SYSTEM CONFIG Link Values" section in “Configuring SMAPI” on page 30.)

system_config_type_length
(int4) Length of *system_config_type*.

system_config_type
(string,0-8,char42) File type of the system configuration file. The default is set by the "System_Config_File_Type =" statement in the DMSSICNF COPY file. (See the "Default SYSTEM CONFIG Link Values" section in “Configuring SMAPI” on page 30.)

parm_disk_owner_length
(int4) Length of *parm_disk_owner*.

parm_disk_owner
(string,0-8,char42) Owner of the parm disk. The default is set by the "Parm_Disk_Owner =" statement in the DMSSICNF COPY file. (See the "Default SYSTEM CONFIG Link Values" section in “Configuring SMAPI” on page 30.)

parm_disk_number_length
(int4) Length of *parm_disk_number*.

parm_disk_number
(string,0-4,char16) Number of the parm disk, as defined in the server's directory. The default is set by the "Parm_Disk_Number =" statement in the DMSSICNF COPY file. (See the "Default SYSTEM CONFIG Link Values" section in “Configuring SMAPI” on page 30.)

parm_disk_password_length
(int4) Length of *parm_disk_password*.

parm_disk_password
(string,0-8,charNB) Multiwrite password for the parm disk. The default is "," and should not be changed. Any value other the default is ignored. (See “Configuring SMAPI” on page 30.)

alt_system_config_name_length
(int4) Length of *alt_system_config_name*.

alt_system_config_name
(string,0-8,char42) No longer valid, maintained for backward compatibility. See Usage Note “1” on page 721.

alt_system_config_type_length
(int4) Length of *alt_system_config_type*.

alt_system_config_type
(string,0-8,char42) No longer valid, maintained for backward compatibility. See Usage Note “1” on page 721.

alt_parm_disk_owner_length
(int4) Length of *alt_parm_disk_owner*.

alt_parm_disk_owner
(string,0-8,char42) No longer valid, maintained for backward compatibility. See Usage Note “1” on page 721.

alt_parm_disk_number_length
(int4) Length of *alt_parm_disk_number*.

alt_parm_disk_number
(string,0-4,char16) No longer valid, maintained for backward compatibility. See Usage Note “1” on page 721.

alt_parm_disk_password_length
(int4) Length of *alt_parm_disk_password*.

alt_parm_disk_password
(string,0-8,charNB) No longer valid, maintained for backward compatibility. See Usage Note “1” on page 721.

gvrp_value
(int1) This can be one of the following values:

0
Unspecified

1
GVRP

2
NOGVRP

Note: If *vlan_id* is specified as UNAWARE, then you cannot specify *port_type*, *gvrp_value* or *native_vlanid*.

native_vlanid
(int4) The native VLAN ID. This can be any of the following values:

-1
The native VLAN ID is not specified.

1 - 4094
Any number in this range is a valid native VLAN ID.

Note: If *vlan_id* is specified as UNAWARE, then you cannot specify *port_type*, *gvrp_value* or *native_vlanid*.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Usage Notes

1. This API updates only the system configuration file on the parm disk specified, and *not* on the alternate parm disk. To maintain backward compatibility, however, the parameters for the alternate parm disk must still be specified. (The easiest way to do this is to simply specify the same values for the alternate parm disk parameters that were specified for the primary parm disk.)
2. If the system administrator has changed the default location of the system configuration file, or has renamed the file, then the input parameters must be used to specify the new file information.
3. Updates for the VSMWORK1 user in the VM directory are required to link and access the CP parm disks. A link option for PMAINT CF0 must be added. If the system administrator changed the default locations of the parm disks, the VSMWORK1 userid must be granted the appropriate authority and links to the new locations.

   The following links are provided in the user directory of VSMWORK1:

   ```
   .
   IDENTITY VSMWORK1 .....
   .
   .
   LINK PMAINT CF0 CF0 MD
   ```

4. If you want a different parm disk, add links to the VSMWORK1 user directory. For example:

   ```
   .
   USER VSMWORK1 .....
   .
   .
   LINK SMAPIC5 C00 FC00 MD
   ```

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
212	RCERR_IMAGECONN	36	RS_VSWITCH_EXISTS	Virtual switch already exists
396	RCERR_INTERNAL	nnnn	psrc	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
620	RCERR_VIRTUALNETWORKD	14	RS_FREE_MODE_NOT_AVAIL	Free modes not available
		22	RS_PARM_DISKS_SAME	System config parm disks 1 and 2 are same
		24	RS_PARM_DISK_LINK_ERROR	Error linking parm disk (1 or 2)
		28	RS_PARM_DISK_NOT_RW	Parm disk (1 or 2) not RW
		32	RS_SYS_CONF_NOT_FOUND	System config not found on parm disk 1
		34	RS_SYS_CONF_BAD_DATA	System config has bad data
		36	RS_SYS_CONF_SYNTX_ERR	Syntax errors updating system config
		38	RS_CPDISK_MODE_NOT _AVAIL	CP disk modes not available
		40	RS_PARM_DISK_FULL	Parm disk (1 or 2) is full
		42	RS_PARM_DISK_ACC_NOT _ALLOWED	Parm disk (1 or 2) access not allowed
		44	RS_PDISK_PW_NOT _SUPPLIED	Parm disk (1 or 2) PW not supplied
		46	RS_PDISK_PW_INCORRECT	Parm disk (1 or 2) PW is incorrect
		48	RS_PDISK_NOT_IN_SERVER _DIRECTORY	Parm disk (1 or 2) is not in server's directory
		50	RS_CP_RELEASE_ERROR	Error in release of CPRELEASE parm disk (1 or 2)
		52	RS_CP_ACCESS_ERROR	Error in access of CPACCESS parm disk (1 or 2)

RC	RC Name	RS	RS Name	Description
		54	RS_DEF_VSWITCH_EXISTS	DEFINE VSWITCH statement already exists in system config
		64	RS_DEF_MOD_MULTI_FOUND	Multiple DEFINE or MODIFY statements found in system config
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Virtual_Network_Vswitch_Create_Extended

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
`switch_name=`*value*
`real_device_address=`*value*
`port_name=`*value*
`controller_name=`*value*
`connection_value=`*value*
`queue_memory_limit=`*value*
`routing_value=`*value*
`transport_type=`*value*
`vlan_id=`*value*
`port_type=`*value*
`persist=`*value*
`gvrp_value=`*value*
`native_vlanid=`*value*
`vswitch_type=`*value*
`iptimeout=`*value*
`port_selection=`*value*
`vswitch_domain=`*value*
`vswitch_global=`*value*
`group_name=`*value*
`log=`*value*

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Virtual_Network_Vswitch_Create_Extended to create a virtual switch.

Note: This API invokes the CP DEFINE VSWITCH and SET VSWITCH commands. See *z/VM: CP Commands and Utilities Reference* for additional information about the values of the key-value input parameters.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 39.

function_name
(string,39,char43) The API function name – in this case, 'Virtual_Network_Vswitch_Create_Extended'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The virtual image name of the owner of the virtual switch.

Note: The format for specifying the following additional input parameters is *parameter_name=value*, followed by a null (ASCIIZ) terminator. They may be specified in any order. For more information, see "Key-value pair (KVP) input parameters" on page 51.

`switch_name=`*value*
(string,1-8,char36 plus @#$_) The name of the virtual switch segment. This is a required parameter.

`real_device_address=`*value*
(string,0-23,char16 plus blank . P p) The real device address or the real device address and OSA Express port number of a QDIO OSA Express device to be used to create the switch to the virtual adapter. If using a real device and an OSA Express port number, specify the real device number followed by a period (.), the letter 'P' (or 'p'), followed by the port number as a hexadecimal number. A maximum of three device addresses, all 1-7 characters in length, may be specified, delimited by blanks. "None" may also be specified. (The default value is "None".)

`port_name=`*value*
(string,0-26,char42 plus blank) The name used to identify the OSA Expanded adapter. A maximum of three port names, all 1-8 characters in length, may be specified, delimited by blanks.

`controller_name=`*value*
One of the following:

- (string,0-8,char42) The userid controlling the real device.
- (string,1,*) Specifies that any available controller may be used.

The default value is '*'.

connection_value=*value*
(string,0-10,char42) One of the following:

CONnect
Activate the real device connection.

DISCONnect
Do *not* activate the real device connection.

NOUPLINK
The virtual switch will never have connectivity through the UPLINK port. This option removes the UPLINK port from the virtual switch. Once the UPLINK port is removed, it can never be added back to the virtual switch.

If not specified, the default is CONNECT.

queue_memory_limit=*value*
(string,0-7,char10 plus -; range 8-256) indicates the amount of memory in megabytes that CP and Queued Direct I/O Hardware Facility should use for buffers for each data connection. The operating range is determined by the active connected adapter and can vary from the range specified. The value is specified as a single number or a range of two numbers separated by a dash (-). The allowed values depend on the type of device that the VSWITCH is connected:

- When a Network-Express (EQDIO) Adapter is active, the amount of memory allocated for EQDIO buffers should vary within the specified range (min-max). The current value is adjusted based on the level of network demand.

  The range specifies the initial number of buffers along with the maximum number of buffers that may be used. The range should be between 8 and 256.

  Example: 8-256

  A single value may be specified and is treated as the minimum value of a range with 256 as the maximum value. If `queue_memory_limit` is omitted, the default is treated by CP as 8-256.
- When connected to an active OSA-Express (QDIO) Adapter, the value determines the amount of memory that can be consumed for QDIO buffers on a single data connection. The OSA-Express (QDIO) interface can operate with up to 8M of memory.

  The value can be specified as a single value between 1 and 8. This is the current value.

  In addition, the value can be specified as a range, as for Network-Express (EQDIO) Adapter. The range is between 8 and 256. The minimum value in the specified range is treated as the current value.

  When the specified minimum is higher than 8, the current value will be 8 instead of the minimum (outside the configured range). This is supported in case the VSWITCH is being configured to support a future Network-Express (EQDIO) adapter.

routing_value=*value*
(string,0-9,char42) Specifies whether the OSA-Express QDIO device will act as a router to the virtual switch, as follows:

NONrouter
The OSA-Express device identified in `real_device_address=` will *not* act as a router to the virtual switch.

PRIrouter
The OSA-Express device identified in `real_device_address=` will act as a primary router to the virtual switch.

If `transport_type=ETHERNET` is specified, this value *must* be unspecified. For other transport types, if this value is unspecified, the default is NONROUTER.

transport_type=*value*
(string,0-8,char42) Specifies the transport mechanism to be used for the virtual switch, as follows:

IP

ETHernet

The default for this value is IP.

vlan_id=*value*
(string,0-8,char42) The VLAN ID. This can be any of the following values:

UNAWARE

AWARE

1 - 4094
Any number in this range is a valid VLAN ID.

If neither vlan_id= nor port_type= are specified, then vlan_id= defaults to UNAWARE.

The default for this value is UNAWARE.

port_type=*value*
(string,0-6,char42) Specifies the port type, as follows:

ACCESS

TRUNK

If vlan_id= is specified but port_type= is not specified, then port_type= will default to ACCESS.

If vlan_id==UNAWARE is specified, then you *cannot* specify port_type=, gvrp_value= or native_vlanid=.

persist=*value*
(string,0-3,char42) This can be one of the following values:

NO
The vswitch is updated on the active system, but is not updated in the permanent configuration for the system.

YES
The vswitch is updated on the active system and also in the permanent configuration for the system.

If not specified, the default is NO.

gvrp_value=*value*
(string,0-6,char42) This can be one of the following values:

GVRP

NOGVRP

If vlan_id=UNAWARE is *not* specified, then the default for this value is GVRP.

If vlan_id=UNAWARE is specified, then you *cannot* specify port_type=, gvrp_value= or native_vlanid=.

native_vlanid=*value*
(string,0-4,char42) The native VLAN ID. This can be any of the following values:

NONE

1 - 4094
Any number in this range is a valid native VLAN ID.

If vlan_id=UNAWARE is specified, then you *cannot* specify port_type=, gvrp_value= or native_vlanid=.

vswitch_type=*value*
(string,0-4,char42) The type of virtual switch to be created. This value can be either of the following:

QDIO
defines a simulated Ethernet or IP virtual switch. A QDIO virtual switch creates a network comprised of both simulated QDIO devices residing on the same z/VM system, with real network devices located on an external or physical network.

IVL
defines an Inter-VSwitch Link which provides the communication facility to implement an IVL Domain. An IVL domain is a grouping of up to 16 systems running z/VM connected by an IVL LAN segment. All the active members within an IVL domain provide control operations that support the creation and management of shared virtual networking components such as Shared Port Groups.

If not specified, the default is QDIO. For more information on these values, see the description of the DEFINE VSWITCH command in *z/VM: CP Commands and Utilities Reference*.

iptimeout=*value*
(string,0-3,char10) A number between 1 and 240 specifying the length of time in minutes that a remote IP address table entry remains in the IP address table for the virtual switch.

If not specified, the default is 5.

port_selection=*value*
(string,0-9,char26) Indicates whether the vswitch is port-based or user-based, as follows:

PORTBASED
The virtual switch configuration and authorization will be on a port basis. Each port must be configured using VIRTUAL_NETWORK_VSWITCH_SET_EXTENDED.

USERBASED
The virtual switch configuration and authorization will be on a user ID basis. Port numbers for guests will be assigned by CP. This is the default if not specified.

group_name=*value*
(string,1-8,char36 plus @#$_) Indicates that the virtual switch UPLINK port is to be configured to use IEEE 802.3ad Link Aggregation. The groupname is a 1- to 8-character name that identifies the group. This option can only be specified when the virtual switch is going to be defined as **transport_type**=ETHERNET.

vswitch_domain=*value*
(string,0-1,char26; range A-H) Defines the domain to which the IVL virtual switch belongs. A is the default value for an IVL vswitch.

vswitch_global=*value*
(string,3-6,char26) One of the following:

GLObal
Identifies this virtual switch as a member of a global virtual switch. A global virtual switch is a collection of virtual switches that share the same name and the same networking characteristics. This collection of virtual switches spans multiple systems running z/VM, but logically operates as a single switch.

LOCal
Indicates that the virtual switch is *not* a member of a global virtual switch.

DOMain
Specified for an IVL switch. An IVL domain is a grouping of up to 16 systems running z/VM, connected by an IVL LAN segment

log=*value*
(string,1-7,char26) defines the default setting for messages displayed for native controller processing when an EQDIO device is connected for this VSwitch. One of the following:

OFF
sets VSwitch activity logging OFF. Normal activity will not be logged to the system operator console. This is the default VSwitch logging mode. Note, however, that errors and unusual events will always be logged to the system operator console regardless of the LOG setting.

ON
sets VSwitch message logging to display brief messages describing each significant event associated with EQDIO device operation. This setting is intended to be used by IBM personnel when diagnosing EQDIO interface problems.

Verbose
sets VSwitch message logging to display more detailed messages describing each significant event associated with EQDIO device operation. This setting is intended to be used by IBM personnel when diagnosing EQDIO interface problems.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Usage Notes

1. Syntax errors (RC = 24 and RS = *pprr*) are only applicable to the common input parameters. Syntax checking is *not* performed on the additional input parameters for this API. If a valid parameter is specified multiple times, the last value specified for that parameter will be used. For more information, see “Key-value pair (KVP) input parameters” on page 51.
2. If the value for an optional input parameter is not specified, the default value for the parameter, if one exists, is used.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
8	RC_ERR	3002	RS_INVALID_PARAMETER	Invalid parameter name
		3003	RS_INVALID_OPERAND	Invalid parameter operand
		3004	RS_MISSING_PARAMETER	Required parameter missing
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
212	RCERR_IMAGECONN	36	RS_VSWITCH_EXISTS	Virtual switch already exists

RC	RC Name	RS	RS Name	Description
396	RCERR_INTERNAL	nnnn	psrc	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Virtual_Network_Vswitch_Delete

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
switch_name_length
switch_name
update_system_config_indicator
system_config_name_length
system_config_name
system_config_type_length
system_config_type
parm_disk_owner_length
parm_disk_owner
parm_disk_number_length
parm_disk_number
parm_disk_password_length
parm_disk_password
alt_system_config_name_length
alt_system_config_name
alt_system_config_type_length
alt_system_config_type
alt_parm_disk_owner_length
alt_parm_disk_owner
alt_parm_disk_number_length
alt_parm_disk_number
alt_parm_disk_password_length
alt_parm_disk_password

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Virtual_Network_Vswitch_Delete to delete a virtual switch.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 30.

function_name
(string,30,char43) The API function name – in this case, 'Virtual_Network_Vswitch_Delete'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The virtual image name of the owner of the virtual switch.

switch_name_length
(int4) Length of *switch_name*.

switch_name
(string,1-8,char36 plus @#$_) The name of the virtual switch segment.

update_system_config_indicator
(int1) This can be any of the following values:

0
Unspecified.

1
Delete the virtual switch from the active system.

2
Delete the virtual switch from the active system and delete the virtual switch definition from the system configuration file.

3
Delete the virtual switch definition from the system configuration file.

If not specified, the default is 1.

system_config_name_length
(int4) Length of *system_config_name*.

system_config_name
(string,0-8,char42) File name of the system configuration file. The default is set by the "System_Config_File_Name =" statement in the DMSSICNF COPY file. (See the "Default SYSTEM CONFIG Link Values" section in “Configuring SMAPI” on page 30.)

system_config_type_length
(int4) Length of *system_config_type*.

system_config_type
(string,0-8,char42) File type of the system configuration file. The default is set by the "System_Config_File_Type =" statement in the DMSSICNF COPY file. (See the "Default SYSTEM CONFIG Link Values" section in “Configuring SMAPI” on page 30.)

parm_disk_owner_length
(int4) Length of *parm_disk_owner*.

parm_disk_owner
(string,0-8,char42) Owner of the parm disk. The default is set by the "Parm_Disk_Owner =" statement in the DMSSICNF COPY file. (See the "Default SYSTEM CONFIG Link Values" section in “Configuring SMAPI” on page 30.)

parm_disk_number_length
(int4) Length of *parm_disk_number*.

parm_disk_number
(string,0-4,char16) Number of the parm disk, as defined in the server's directory. The default is set by the "Parm_Disk_Number =" statement in the DMSSICNF COPY file. (See the "Default SYSTEM CONFIG Link Values" section in “Configuring SMAPI” on page 30.)

parm_disk_password_length
(int4) Length of *parm_disk_password*.

parm_disk_password
(string,0-8,charNB) Multiwrite password for the parm disk. The default is ",".

Note:

1. The character "," is used to indicate no password. Therefore "," cannot be the password.
2. A password is not required if appropriate ESM permissions are granted for the appropriate minidisks.

alt_system_config_name_length
(int4) Length of *alt_system_config_name*.

alt_system_config_name
(string,0-8,char42) No longer valid, maintained for backward compatibility. See Usage Note “1” on page 734.

alt_system_config_type_length
(int4) Length of *alt_system_config_type*.

alt_system_config_type
(string,0-8,char42) No longer valid, maintained for backward compatibility. See Usage Note “1” on page 734.

alt_parm_disk_owner_length
(int4) Length of *alt_parm_disk_owner*.

alt_parm_disk_owner
(string,0-8,char42) No longer valid, maintained for backward compatibility. See Usage Note “1” on page 734.

alt_parm_disk_number_length
(int4) Length of *alt_parm_disk_number*.

alt_parm_disk_number
(string,0-4,char16) No longer valid, maintained for backward compatibility. See Usage Note “1” on page 734.

alt_parm_disk_password_length
(int4) Length of *alt_parm_disk_password*.

alt_parm_disk_password
(string,0-8,charNB) No longer valid, maintained for backward compatibility. See Usage Note "1" on page 734.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Usage Notes

1. This API updates only the system configuration file on the parm disk specified, and *not* on the alternate parm disk. To maintain backward compatibility, however, the parameters for the alternate parm disk must still be specified. (The easiest way to do this is to simply specify the same values for the alternate parm disk parameters that were specified for the primary parm disk.)
2. If the system administrator has changed the default location of the system configuration file, or has renamed the file, then the input parameters must be used to specify the new file information.
3. Updates for the VSMWORK1 user in the VM directory are required to link and access the CP parm disks. A link option for PMAINT CF0 must be added. If the system administrator changed the default locations of the parm disks, the VSMWORK1 userid must be granted the appropriate authority and links to the new locations.

   The following links are provided in the user directory of VSMWORK1:

   ```
   .
   IDENTITY VSMWORK1 .....
   .
   .
   LINK PMAINT CF0 CF0 MD
   ```

4. If you want a different parm disk, add links to the VSMWORK1 user directory. For example:

   ```
   .
   USER VSMWORK1 .....
   .
   .
   LINK SMAPIC5 C00 FC00 MD
   ```

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
		44	RS_VSWITCH_REMOVED	Request successful; virtual switch removed

RC	RC Name	RS	RS Name	Description
		66	RS_DEF_MOD_MULTI _ERASED	Multiple DEFINE or MODIFY statements are erased in system config
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
212	RCERR_IMAGECONN	40	RS_VSWITCH_NOT_EXISTS	Virtual switch does not exist
396	RCERR_INTERNAL	nnnn	psrc	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
620	RCERR_VIRTUALNETWORKD	14	RS_FREE_MODE_NOT_AVAIL	Free modes not available
		22	RS_PARM_DISKS_SAME	System config parm disks 1 and 2 are same
		24	RS_PARM_DISK_LINK_ERROR	Error linking parm disk (1 or 2)
		28	RS_PARM_DISK_NOT_RW	Parm disk (1 or 2) not RW
		32	RS_SYS_CONF_NOT_FOUND	System config not found on parm disk 1
		34	RS_SYS_CONF_BAD_DATA	System config has bad data
		36	RS_SYS_CONF_SYNTX_ERR	Syntax errors updating system config
		38	RS_CPDISK_MODE_NOT _AVAIL	CP disk modes not available
		40	RS_PARM_DISK_FULL	Parm disk (1 or 2) is full
		42	RS_PARM_DISK_ACC_NOT _ALLOWED	Parm disk (1 or 2) access not allowed
		44	RS_PDISK_PW_NOT _SUPPLIED	Parm disk (1 or 2) PW not supplied
		46	RS_PDISK_PW_INCORRECT	Parm disk (1 or 2) PW is incorrect
		48	RS_PDISK_NOT_IN_SERVER _DIRECTORY	Parm disk (1 or 2) is not in server's directory
		50	RS_CP_RELEASE_ERROR	Error in release of CPRELEASE parm disk (1 or 2)
		52	RS_CP_ACCESS_ERROR	Error in access of CPACCESS parm disk (1 or 2)

RC	RC Name	RS	RS Name	Description
		54	RS_DEF_VSWITCH_EXISTS	DEFINE VSWITCH statement already exists in system config
		60	RS_DEF_SWITCH_NOT_EXIST	DEFINE VSWITCH statement does not exist in system config
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Virtual_Network_Vswitch_Delete_Extended

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
`switch_name=`*value*
`persist=`*value*

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use Virtual_Network_Vswitch_Delete_Extended to delete a virtual switch.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 39.

function_name
(string,39,char43) The API function name – in this case, 'Virtual_Network_Vswitch_Delete_Extended'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See “Client Authentication” on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The virtual image name of the owner of the virtual switch.

Note: The format for specifying the following additional input parameters is *parameter_name=value*, followed by a null (ASCIIZ) terminator. They may be specified in any order. For more information, see "Key-value pair (KVP) input parameters" on page 51.

switch_name=*value*
(string,1-8,char36 plus @#$_) The name of the virtual switch segment. This is a required parameter.

persist=*value*
(string,0-3,char42) This can be one of the following values:

NO
The vswitch is deleted on the active system, but is not deleted from the permanent configuration for the system.

YES
The vswitch is deleted from the active system and also from the permanent configuration for the system.

If not specified, the default is NO.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Usage Notes

1. Syntax errors (RC = 24 and RS = *pprr*) are only applicable to the common input parameters. Syntax checking is *not* performed on the additional input parameters for this API. If a valid parameter is specified multiple times, the last value specified for that parameter will be used. For more information, see "Key-value pair (KVP) input parameters" on page 51.
2. If the value for an optional input parameter is not specified, the default value for the parameter, if one exists, is used.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
8	RC_ERR	3002	RS_INVALID_PARAMETER	Invalid parameter name
		3003	RS_INVALID_OPERAND	Invalid parameter operand
		3004	RS_MISSING_PARAMETER	Required parameter missing
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
212	RCERR_IMAGECONN	40	RS_VSWITCH_NOT_EXISTS	Virtual switch does not exist
396	RCERR_INTERNAL	nnnn	psrc	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Virtual_Network_Vswitch_Query

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
switch_name_length
switch_name

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
vswitch_array_length

Response 2 – Output Parameters (continued):

vswitch_array (1)
vswitch_structure (2)
vswitch_structure_length
switch_name_length
switch_name
transport_type
port_type
queue_memory_limit
routing_value
vlan_id
native_vlan_id
mac_id
gvrp_request_attribute
gvrp_enabled_attribute
switch_status
real_device_array_length
real_device_array (1)
real_device_structure (2)
real_device_structure_length
real_device_address
controller_name_length
controller_name
port_name_length
port_name
device_status
device_error_status
authorized_user_array_length
authorized_user_array (1)
authorized_user_structure (2)
authorized_user_structure_length
grant_userid_length
grant_userid
vlan_array_length
vlan_array (1)
vlan_structure (2)
vlan_structure_length
user_vlan_id
connected_adapter_array_length
connected_adapter_array (1)
connected_adapter_structure (2)
connected_adapter_structure_length
adapter_owner_length
adapter_owner
image_device_number_length
image_device_number

Note:

1. An array consists of zero or more of its components.
2. A structure consists of one set of its components.

Purpose

Use Virtual_Network_Vswitch_Query to obtain information about the specified virtual switch or switches.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 29.

function_name
(string,29,char43) The API function name – in this case, 'Virtual_Network_Vswitch_Query'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) This must match an entry in the authorization file that also contains the *authenticated_userid* and the *function_name* (Virtual_Network_Vswitch_Query).

switch_name_length
(int4) Length of *switch_name*.

switch_name
One of the following:

- (string,1-8,char36 plus @#$_) The name of the new virtual switch.
- (string,1,*) All virtual switches.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

vswitch_array_length
(int4) Length of *vswitch_array*.

vswitch_array
(array) An array consisting of zero or more instances of *vswitch_structure*, as follows:

vswitch_structure
(structure) A structure consisting of one set of the following parameters:

vswitch_structure_length
(int4) The combined length of the remaining parameters in *vswitch_structure* (not including this parameter).

switch_name_length
(int4) Length of *switch_name*.

switch_name
One of the following:

- (string,1-8,char36 plus @#$_) The name of the virtual switch.
- (string,1,*) All virtual switches.

transport_type
(int1) The transport mechanism. The possible values are:

1
IP

2
Ethernet

port_type
(int1) The port type. The possible values are:

1
Access

2
Trunk

queue_memory_limit
(int4) The QDIO buffer size in megabytes.

routing_value
(int1) Indicates if the QDIO device will act as a router. The possible values are:

1
The device will not act as a router.

2
The device will act as a router.

vlan_id
(int4) The default VLAN ID. A value of 32768 indicates that the virtual switch was created as VLAN AWARE, without a default VLAN ID.

native_vlan_id
(int4) The native VLAN ID. A value of 32768 indicates that the virtual switch was created as VLAN AWARE, with a native VLAN ID of NONE.

mac_id
(int8) The MAC identifier.

gvrp_request_attribute
(int1) The attribute indicating if GVRP was requested. The possible values are:

1
GVRP requested

2
GVRP not requested

gvrp_enabled_attribute
(int1) The attribute indicating if GVRP is enabled. The possible values are:

1
GVRP enabled

2
GVRP not enabled

switch_status
(int1) The status of the virtual switch. The possible values are:

1
Virtual switch defined.

2
Controller not available.

3
Operator intervention required.

4
Disconnected.

5
Virtual devices attached to controller. Normally a transient state.

6
OSA initialization in progress. Normally a transient state.

7
OSA device not ready.

8
OSA device ready.

9
OSA devices being detached. Normally a transient state.

10
Virtual switch delete pending. Normally a transient state.

11
Virtual switch failover recovering. Normally a transient state.

12
Autorestart in progress. Normally a transient state.

real_device_array_length
(int4) Length of *real_device_array*.

real_device_array
(array) An array consisting of zero or more instances of *real_device_structure*, as follows:

real_device_structure
(structure) A structure consisting of one set of the following parameters:

real_device_structure_length
(int4) The combined length of the remaining parameters in *real_device_structure* (not including this parameter).

real_device_address
(int4) The real device address of the OSA-Express QDIO device.

controller_name_length
(int4) Length of *controller_name*.

controller_name
(string,0-71,char42 plus blank) The userid controlling the real device. This may be a maximum of eight userids, all 1-8 characters in length, delimited by blanks.

port_name_length
(int4) Length of *port_name*.

port_name
(string,0-16,char16) The port name.

device_status
(int1) The status of the real device. The possible values are:

0
Device is not active.

1
Device is active.

2
Device is a backup device.

device_error_status
(int1) The error status of the real device. The possible values are:

0
No error.

1
Port name conflict.

2
No layer 2 support.

3
Real device does not exist.

4
Real device is attached elsewhere.

5
Real device is not compatible type.

6
Initialization error.

7
Stalled OSA.

8
Stalled controller.

9
Controller connection severed.

10
Primary or secondary routing conflict.

11
Device is offline.

12
Device was detached.

13
IP/Ethernet type mismatch.

14
Insufficient memory in controller virtual machine.

15
TCP/IP configuration conflict.

16
No link aggregation support.

17
OSA-E attribute mismatch.

18
Reserved for future use.

19
OSA-E is not ready.

20
Reserved for future use.

21
Attempting restart for device.

22
Exclusive user error.

23
Device state is invalid.

24
Port number is invalid for device.

25
No OSA connection isolation.

26
EQID mismatch.

27
Incompatible controller.

28
BACKUP detached.

29
BACKUP not ready.

30
BACKUP attempting restart.

31
EQID mismatch.

32
No HiperSockets bridge support.

33
HiperSockets bridge error.

authorized_user_array_length
(int4) Length of *authorized_user_array*.

authorized_user_array
(array) An array consisting of zero or more instances of *authorized_user_structure*, as follows:

authorized_user_structure
(structure) A structure consisting of one set of the following parameters:

authorized_user_structure_length
(int4) The combined length of the remaining parameters in *authorized_user_structure* (not including this parameter).

grant_userid_length
(int4) Length of *grant_userid*.

grant_userid
(string,1-8.char42) The userid authorized to connect to the virtual switch.

vlan_array_length
(int4) Length of *vlan_array*.

vlan_array
(array) An array consisting of zero or more instances of *vlan_structure*, as follows:

vlan_structure
(structure) A structure consisting of one set of the following parameters:

vlan_structure_length
(int4) The combined length of the remaining parameters in *vlan_structure* (not including this parameter).

user_vlan_id
(int4) The authorized VLAN ID.

connected_adapter_array_length
(int4) Length of *connected_adapter_array*.

connected_adapter_array
(array) An array consisting of zero or more instances of *connected_adapter_structure*, as follows:

connected_adapter_structure_length
(int4) The combined length of the remaining parameters in *connected_adapter_structure* (not including this parameter).

connected_adapter_structure
(structure) A structure consisting of one set of the following parameters:

adapter_owner_length
(int4) Length of *adapter_owner*.

adapter_owner
(string,1-8,char42) The userid owning the adapter.

image_device_number_length
(int4) Length of *image_device_number*.

image_device_number
(string,1-4,char16) The virtual device address of the adapter.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
212	RCERR_IMAGECONN	40	RS_VSWITCH_NOT_EXISTS	Virtual switch does not exist
396	RCERR_INTERNAL	nnnn	psrc	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)

RC	RC Name	RS	RS Name	Description
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Virtual_Network_Vswitch_Query_Byte_Stats

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
`switch_name=`*value*

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

- *output_length*
- *request_id*
- *return_code*
- *reason_code*
- *vswitch_array_size*
- *vswitch_array*
- *vswitch_structure*
  - *switch_name*
  - *uplink_array_size*
  - *uplink_array*
  - *uplink_structure*
    - *unplink_conn*
    - *uplink_fr_rx*
    - *uplink_fr_rx_dsc*
    - *uplink_fr_rx_err*
    - *uplink_fr_tx*
    - *uplink_fr_tx_dsc*
    - *uplink_fr_tx_dsc_err*
    - *uplink_rx*
    - *uplink_tx*
  - *bridge_fr_rx*
  - *bridge_fr_rx_dsc*
  - *bridge_fr_rx_err*
  - *bridge_fr_tx*
  - *bridge_fr_tx_dsc*
  - *bridge_fr_tx_err*
  - *bridge_rx*
  - *bridge_tx*
  - *nic_array_size*
  - *nic_array*
  - *nic_structure*
    - *nic_id*
    - *nic_fr_rx*
    - *nic_fr_rx_dsc*
    - *nic_fr_rx_err*
    - *nic_fr_tx*
    - *nic_fr_tx_dsc*
    - *nicr_tx_dsc_err*
    - *nic_rx*
    - *nic_tx*
  - *vlan_array_size*
  - *vlan_array*
  - *vlan_structure*
    - *vlan_id*
    - *vlan_rx*
    - *vlan_tx*

Purpose

Use Virtual_Network_Vswitch_Query_Byte_Stats to query the byte information statistics for a virtual switch.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name*.

function_name
(string,15,char43) The API function name – in this case, 'Virtual_Network_Vswitch_Query_Byte_Stats'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) Used strictly for authorization, i.e. the authenticated user must have authorization to perform this function for this target.

Note: The format for specifying the following additional input parameter is *parameter_name=value*, followed by a null (ASCIIZ) terminator. They may be specified in any order. For more information, see "Key-value pair (KVP) input parameters" on page 51.

switch_name=value
The name of the virtual switch, or "*" for all vswitches.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Note:

The following output values are all character strings, blank-delimited if there is more than one value within the field, and null (ASCIIZ) terminated to designate the end of each field.

vswitch_array_size
Number of elements in the vswitch_array.

vswitch_array
An array consisting of one or more instances of *vswitch_structure*, as follows:

vswitch_structure
A structure consisting of one set of the following parameters:

switch_name
The name of the virtual switch.

uplink_array_size
Number of elements in *uplink_array*.

uplink_array
An array consisting of zero or more instances of *uplink_structure* as follows:

uplink_structure
A structure consisting of one set of the following parameters:

uplink_conn
Either the real device number or the *userid vdev* associated with this uplink.

uplink_fr_rx
Received frames on this uplink port.

uplink_fr_rx_dsc
Received frames that were discarded.

uplink_fr_rx_err
Errors on received frames.

uplink_fr_tx
Transmitted frames on this uplink port.

uplink_fr_tx_dsc
Transmitted frames that were discarded.

uplink_fr_tx_dcs_err
Errors on transmitted frames.

uplink_rx
Received bytes on this uplink port.

uplink_tx
Transmitted bytes on this uplink port.

bridge_fr_rx
Received frames on the bridge port.

bridge_fr_rx_dsc
Received frames on the bridge port that were discarded.

bridge_fr_rx_err
Errors on received frames on the bridge port.

bridge_fr_tx
Transmitted frames on the bridge port.

bridge_fr_tx_dsc
Transmitted frames on the bridge port that were discarded.

bridge_fr_tx_err
Errors on transmitted frames on the bridge port.

bridge_rx
Received bytes on the bridge port.

bridge_tx
Transmitted bytes on the bridge port.

nic_array_size
The number of elements in *nic_array*.

nic_array
An array consisting of zero or more instances of the *nic_structure* as follows:

nic_structure
A structure consisting of one set of the following parameters:

nic_id
NIC owner, then a blank, then virtual device address (including port). The virtual device address is in the form: *nnnn*P*nn*

nic_fr_rx
Received frames on this NIC.

nic_fr_rx_dsc
Received frames on this NIC that were discarded.

nic_fr_rx_err
Errors on received frames on this NIC.

nic_fr_tx
Transmitted frames on this NIC.

nic_fr_tx_dsc
Transmitted frames on this NIC that were discarded.

nic_tx_err
Errors on transmitted frames on this NIC.

nic_rx
Received bytes on this NIC.

nic_tx
Transmitted bytes on this NIC.

vlan_array_size
The number of elements in *vlan_array*.

vlan_array
An array consisting of zero or more instances of the *vlan_structure* as follows:

vlan_structure
A structure consisting of one set of the following parameters:

vlan_id
The VLAN ID.

vlan_rx
Received bytes on this VLAN.

vlan_tx
Transmitted bytes on this VLAN.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
8	RC_ERR	999	RS_NOT_AVAILABLE	This function is not available on this system
		3002	RS_INVALID_PARAMETER	Invalid parameter name
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
396	RCERR_INTERNAL	nnnn	psrc	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Virtual_Network_Vswitch_Query_Extended

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
`switch_name=`*value*
`vepa_status=`*value*

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
vswitch_count
vswitch_array (1)
vswitch_structure (2)
vswitch_structure_length
vswitch_attr_info_structure_length
vswitch_attr_info_structure (2)
switch_name
transport_type
port_type
queue_memory_limit
routing_value
vlan_awareness
vlan_id
native_vlan_id
mac_address
gvrp_request_attribute
gvrp_enabled_attribute
switch_status
link_ag
lag_interval
lag_group
IP_timeout
switch_type
isolation_status
MAC_protect
user_port_based
VLAN_counters
vepa_status
spg_scope
queue_memory_limit_min
queue_memory_limit_max
load_balance_type
priority_queuing_state
log_setting

Response 2 – Output Parameters (continued):

- *real_device_info_array_length*
- *real_device_info_array (1)*
  - *real_device_info_structure (2)*
    - *real_device_address*
    - *virtual_device_address*
    - *controller_name*
    - *port_name*
    - *device_status*
    - *device_error_status*
    - *nic_distribution*
    - *path_mtu_discovery_setting*
    - *path_mtu_discovery_value*
    - *mfs*
    - *buffer_limit*
    - *buffer_in_use*
    - *async_data_xfer_count*
    - *siga_busy_count*
    - *partner_switch_capability*
    - *adapter_cceid*
    - *adapter_pchid*
- *authorized_user_array_length*
- *authorized_user_array (1)*
  - *authorized_user_structure (2)*
    - *port_num*
    - *grant_userid*
    - *promiscuous_mode*
    - *osd_sim*
    - *vlan_count*
    - *vlan_info*
      - *user_vlan_id*
- *connected_adapter_array_length*
- *connected_adapter_array (1)*
  - *connected_adapter_structure (2)*
    - *adapter_owner*
    - *adapter_vdev*
    - *adapter_macaddr*
    - *adapter_type*
- *uplink_NIC_structure_length*
- *uplink_NIC_structure (2)*
  - *uplink_NIC_userid*
  - *uplink_NIC_vdev*
  - *uplink_NIC_error_status*
- *global_member_array_length*
- *global_member_array (1)*
  - *global_member_structure (2)*
    - *member_name*
    - *member_state*
    - *sync_error_code*

Note:

1. An array consists of zero or more of its components.
2. A structure consists of one set of its components.

Purpose

Use Virtual_Network_Vswitch_Query_Extended to obtain information about the specified virtual switch or switches.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 38.

function_name
(string,38,char43) The API function name – in this case, 'Virtual_Network_Vswitch_Query_Extended'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) This must match an entry in the authorization file that also contains the *authenticated_userid* and the *function_name* (Virtual_Network_Vswitch_Query_Extended).

Note: The format for specifying the following additional input parameters is *parameter_name=value*, followed by a null (ASCIIZ) terminator. They may be specified in any order. For more information, see "Key-value pair (KVP) input parameters" on page 51.

`switch_name=`*value*
One of the following.

- (string,1-8,char36 plus @#$_) The name of the new virtual switch segment.
- (string,1,*) All virtual switches.

`vepa_status=`*value*
(string,2-3,char26) One of the following:

YES
Indicates that the *vepa_status* output parameter will be included in the *vswitch_attr_info_structure*.

NO
Indicates that the *vepa_status* output parameter will *not* be included in the *vswitch_attr_info_structure*. This is the default.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

vswitch_count
(int4) Number of null-delimited entries in *vswitch_array*.

vswitch_array
(array) An array consisting of zero or more instances of *vswitch_structure*, as follows:

vswitch_structure
(structure) A structure consisting of one set of the following parameters:

vswitch_structure_length
(int4) The combined length of the remaining parameters in *vswitch_structure* (not including this parameter).

vswitch_attr_info_structure_length
(int4) Length of *vswitch_attr_info_structure*.

vswitch_attr_info_structure
(structure) A null-terminated structure consisting of one set of the following parameters, with a blank separating each parameter:

switch_name
(string,1-8,char36 plus @#$_) The name of the virtual switch.

transport_type
(string,2-8,char26) The transport mechanism. The possible values are:

IP

ETHERNET

port_type
(string,4-6,char26) The port type. The possible values are:

ACCESS

TRUNK

NONE

queue_memory_limit
(string,1-3,char10) The current QDIO buffer size in megabytes (0-255).

routing_value
(string,2-9,char26) Indicates if the QDIO device will act as a router. The possible values are:

PRIROUTER
The device will act as a router.

NONROUTER
The device will *not* act as a router.

NA
This attribute does not apply to ETHERNET switches.

vlan_awareness
(string,5-7,char26) Indicates if the switch is VLAN aware. The possible values are:

AWARE

UNAWARE

vlan_id
(string,1-8,char42) The default VLAN ID. A value of 32768 indicates that the virtual switch was created without a default VLAN ID. (In this case, you will also see *vlan_awareness*=AWARE returned.)

native_vlan_id
(string,1-8,char42) The native VLAN ID. A value of 32768 indicates that the virtual switch was created with a native VLAN ID of NONE. (In this case, you will also see *vlan_awareness*=AWARE returned.)

mac_address
(string,17,char16 plus -) The MAC address.

gvrp_request_attribute
(string,4-6,char26) The attribute indicating if GVRP was requested. The possible values are:

GVRP
GVRP requested

NOGVRP
GVRP not requested

gvrp_enabled_attribute
(string,4-6,char26) The attribute indicating if GVRP is enabled. The possible values are:

GVRP
GVRP enabled

NOGVRP
GVRP not enabled

switch_status
(string,1-2,char10) The status of the virtual switch. The possible values are:

1
Virtual switch defined.

2
Controller not available.

3
Operator intervention required.

4
Disconnected.

5
Virtual devices attached to controller. Normally a transient state.

6
OSA initialization in progress. Normally a transient state.

7
OSA device not ready.

8
OSA device ready.

9
OSA devices being detached. Normally a transient state.

10
Virtual switch delete pending. Normally a transient state.

11
Virtual switch failover recovering. Normally a transient state.

12
Autorestart in progress. Normally a transient state.

link_ag
(string,4-41,char26) Indicates the current status of link aggregation, using a combination of the following values to form the string 'LAG:-*xxx*-...-*xxx*'

GROUP
GROUP attribute specified

LACP
LACP active

SHARED
Shared port group

EXCLUSIVE
Exclusive port group

lag_interval
(string.1-3,char10) Link aggregation time interval. (This will be 0 if link aggregation is not active.)

lag_group
(string,1-8,char42) Link aggregation group name.

IP_timeout
(string.1-3,char10) IP timeout interval.

switch_type
(string,4,char26) The vswitch type. This can be only QDIO.

isolation_status
(string,9-11,char26) Indicates whether port isolation is active, as follows:

ISOLATION
Port isolation is active

NOISOLATION
Port isolation is not active

MAC_protect
(string,10-13,char26) Indicates whether MAC address protection is active, as follows:

MACPROTECT
MAC address protection is active

NOMACPROTECT
MAC address protection is not active

UNSPECIFIED

user_port_based
(string,9,char26) Indicates if vswitch is user-based or port-based, as follows:

USERBASED

PORTBASED

VLAN_counters
(string,8-10,char26) One of the following:

COUNTERS

NOCOUNTERS

vepa_status
(string,2-3,char26) One of the following:

ON
Indicates that guests are prohibited from sending traffic to other guests on the same virtual switch, without going through an external entity by forwarding all traffic from the guest through the OSA uplink to an adjacent switch. In addition, no direct LPAR communications sharing the same OSA port are permitted with the guest ports of the virtual switch. All traffic from the virtual switch destined for any sharing hosts/LPARs on the same OSA port will be forwarded, as well. Any traffic destined for the virtual switch guest ports from hosts/LPARs sharing the same OSA port will also be forwarded to the adjacent switch.

OFF
Indicates that guest ports are allowed to communicate with each other and with any hosts and/or LPARs that share the same OSA port.

This output parameter is present only if `vepa_status`=YES was specified.

spg_scope
(string,5-7,char26) The shared port group scope, one of the following values:

NoSyn
Not synchronized

Error
Error

Pending
Pending synchronization

Synced
Synchronized

(NONE)
Not shared

queue_memory_limit_min
(string,1-3,char10) The minimum QDIO buffer size in megabytes (0-255).

queue_memory_limit_max
(string,1-3,char10) The maximum QDIO buffer size in megabytes (0-255).

load_balance_type
(string,6-13,char26) Load balance type, one of the following values:

INDEPENDENT
Independent

FORCED
Forced independent

COLLABORATIVE
Collaborative

(UNKNOWN)
Unknown value

priority_queuing_state
(string,2-6,char26) Priority queuing state, one of the following values:

OFF
Off

FORCED
Forced off

ON
On

(UNKNOWN)
Unknown value

log_setting
(string,2-7,char26) Log setting, one of the following values:

OFF
Off

ON
On

VERBOSE
Verbose

(UNKNOWN)
Unknown value

real_device_info_array_length
(int4) Length of *real_device_info_array*.

real_device_info_array
(array) An array consisting of zero or more instances of *real_device_info_structure*, as follows:

real_device_info_structure
(structure) A null-terminated structure consisting of one set of the following parameters, with a blank separating each parameter:

real_device_address
(string,4,char16) The real device address of the OSA-Express QDIO device.

virtual_device_address
(string,4,char16) The virtual device address of the device.

controller_name
(string,1-71,char42 plus _) The userid controlling the real device. This may be a maximum of eight userids, all 1-8 characters in length, delimited by underscores ('_').

port_name
(string,1-8,char42) The port name.

device_status
(string,1,char10) The status of the real device. The possible values are:

0
Device is not active.

1
Device is active.

2
Device is a backup device.

device_error_status
(string,1-2,char10) The error status of the real device. The possible values are:

0
No error.

1
Port name conflict.

2
No layer 2 support.

3
Real device does not exist.

4
Real device is attached elsewhere.

5
Real device is not compatible type.

6
Initialization error.

7
Stalled OSA.

8
Stalled controller.

9
Controller connection severed.

10
Primary or secondary routing conflict.

11
Device is offline.

12
Device was detached.

13
IP/Ethernet type mismatch.

14
Insufficient memory in controller virtual machine.

15
TCP/IP configuration conflict.

16
No link aggregation support.

17
OSA-E attribute mismatch.

18
Reserved for future use.

19
OSA-E is not ready.

20
Reserved for future use.

21
Attempting restart for device.

22
Exclusive user error.

23
Device state is invalid.

24
Port number is invalid for device.

25
No OSA connection isolation.

26
EQID mismatch.

27
Incompatible controller.

28
BACKUP detached.

29
BACKUP not ready.

30
BACKUP attempting restart.

31
EQID mismatch.

32
No HiperSockets bridge support.

33
HiperSockets bridge error.

nic_distribution
(string,2-3,char26) One of the following:

ON
NIC distribution enabled.

OFF
NIC distribution disabled.

path_mtu_discovery_setting
(string,3-8,char26) One of the following:

OFF
The Path MTU Discovery process for this virtual switch is disabled by CP.

VALUE
A user-defined value is used by CP as the Path MTU Discovery value.

EXTERNAL
CP should determine the Path MTU Discovery value to assign.

(UNKNOWN)
Unknown value.

path_mtu_discovery_value
(string,5,char10) The number of bytes used by CP as the Path MTU discovery value.

mfs
(string,2,char10) The maximum frame size expressed in kilobytes.

buffer_limit
(string,1-10,char10) Buffer limit for asynchronous transmissions.

buffer_in_use
(string,1-10,char10) Number of asynchronous transmission buffers currently in use.

async_data_xfer_count
(string,1-10,char10) Count of async data transfers.

siga_busy_count
(string,1-10,char10) Number of times the synchronous (unicast) transmit function encountered a condition code 2 error (indicating the destination host was busy and could not accept a synchronous buffer transfer).

partner_switch_capability
(string,8-11,char26) One of the following:

UNAVAILABLE
Not available.

STANDARD
Standard - VEB.

REFLECTIVE
Reflective Relay.

(UNKNOWN)
Unknown value.

adapter_cceid
(string,6,char26) Six bytes of data that uniquely identifies the adapter processor.

adapter_pchid
(string,2,char16) Two bytes of data that identifies the Physical Channel ID representing the OSA_express adapter.

authorized_user_array_length
(int4) Length of *authorized_user_array*.

authorized_user_array
(array) An array consisting of zero or more instances of *authorized_user_structure*, as follows:

authorized_user_structure
(structure) A null-terminated structure consisting of one set of the following parameters, with a blank separating each parameter:

port_num
(string,1-16,char16) The port number.

grant_userid
(string,1-8,char42) The userid authorized to connect to the virtual switch, if user-based, or the port if port-based.

promiscuous_mode
(string,4-6,char26) Indicates if user or port is authorized for promiscuous mode, as follows:

PROM
Authorized for promiscuous mode

NOPROM
Not authorized for promiscuous mode

osd_sim
(string,6-8,char26) Indicates if user or port is authorized for OSDSIM, as follows:

OSDSIM
Authorized for OSDSIM

NOOSDSIM
Not authorized for OSDSIM

vlan_count
(string,1-2,char10) Number of null-delimited VLAN entries in *vlan_info*.

vlan_info
(array) A set of blank-delimited strings (one string per device as per the number defined in *vlan_count*), each string consisting of one set of the following values:

user_vlan_id
(string,1-8,char42) The authorized VLAN ID.

connected_adapter_array_length
(int4) Length of *connected_adapter_array*.

connected_adapter_array
(array) An array consisting of zero or more instances of *connected_adapter_structure*, as follows:

connected_adapter_structure
(structure) A null-terminated structure consisting of one set of the following parameters, with a blank separating each parameter:

adapter_owner
(string,1-8,char42) The userid owning the adapter.

adapter_vdev
(string,4,char16) The virtual device address of the adapter.

adapter_macaddr
(string,6-17,char36) The unicast MAC address of the adapter. If no such address is found, this value will be "(NONE)".

adapter_type
(string,4-12,char26) The adapter type. This can be only QDIO.

uplink_NIC_structure_length
(int4) Length of *uplink_NIC_structure*.

uplink_NIC_structure
(structure) A null-terminated structure consisting of one set of the following parameters, with a blank separating each parameter:

uplink_NIC_userid
(string,1-8,char42) The userid owning the uplink adapter.

uplink_NIC_vdev
(string,4,char16) The virtual device address of the uplink adapter.

uplink_NIC_error_status
(string,1-3,char10) The error status, as follows:

0
No error

1
Userid not logged on

2
Not authorized

3
VDEV does not exist

4
VDEV is attached elsewhere

5
VDEV not compatible type

6
VLAN conflict

7
No MAC address

8
Not managed

9
Port Error

13
Type mismatch

255
Unknown error

global_member_array_length
(int4) Length of *global_member_array*.

global_member_array
(array) An array consisting of zero or more instances of *global_member_structure*, as follows:

global_member_structure
(structure) A null-terminated structure consisting of one set of the following parameters, with a blank separating each parameter:

member_name
(string,1-8,char36 plus @#$_) Global member name.

member_state
(string,5-7,char26) Member state, one of the following values:

NoSyn
Not synchronized

Error
Error

Pending
Pending synchronization

Synced
Synchronized

sync_error_code
(string,8,char16) Synchronization error code.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
8	RC_ERR	3002	RS_INVALID_PARAMETER	Invalid parameter name
		3003	RS_INVALID_OPERAND	Invalid parameter operand
		3004	RS_MISSING_PARAMETER	Required parameter missing
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
212	RCERR_IMAGECONN	40	RS_VSWITCH_NOT_EXISTS	Virtual switch does not exist
396	RCERR_INTERNAL	nnnn	psrc	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range
		99	RS_RETRY	A system change occurred during the API call – reissue the API call to obtain the data.

Virtual_Network_Vswitch_Query_Stats

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
`switch_name=`*value*
`fmt_version=`*value*

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
vswitch_array_length
vswitch_array (1)
vswitch_structure (2)
switch_name_length
switch_name
segment_array_length
segment_array (1)
segment_structure (2)
seg_vlanid
seg_rx
seg_rx_disc
seg_tx
seg_tx_disc
seg_activated_TOD
seg_config_update_TOD
seg_vlan_interfaces
seg_vlan_deletes
seg_device_type
seg_device_addr
seg_device_status

Note:

1. An array consists of zero or more of its components.
2. A structure consists of one set of its components.

Purpose

Use Virtual_Network_Vswitch_Query_Stats to query a virtual switch's statistics.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 35.

function_name
(string,35,char43) The API function name – in this case, 'Virtual_Network_Vswitch_Query_Stats'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) Used strictly for authorization, i.e. the authenticated user must have authorization to perform this function for this target.

Note: The format for specifying the following additional input parameters is *parameter_name=value*, followed by a null (ASCIIZ) terminator. They may be specified in any order. For more information, see "Key-value pair (KVP) input parameters" on page 51.

`switch_name=`*value*
(string,1-8,char36 plus @#$_) The name of the virtual switch segment. This is a required parameter.

`fmt_version=`*value*
(string,0-10,char10) The format version of this API, for calls to DIAGNOSE X'26C'. For V6.2, the supported format version value is 4. This is an optional parameter.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

vswitch_array_length
(int4) Length of *vswitch_array*.

vswitch_array
(array) An array consisting of zero or more instances of *vswitch_structure*, as follows:

vswitch_structure
(structure) A structure consisting of one set of the following parameters:

switch_name_length
(int4) Length of *switch_name*.

switch_name
(string,0-8,char36 plus @#$_) The name of the virtual switch.

segment_array_length
(int4) Length of *segment_array*.

segment_array
(array) An array consisting of zero or more instances of *segment_structure*, as follows:

segment_structure
(structure) A null-terminated structure consisting of one set of the following parameters, with a blank separating each parameter:

seg_vlanid
(string,1-10,char10) The VLAN ID, or 0. (0 is returned for a VLAN UNAWARE virtual switch, or for a VLAN AWARE virtual switch with the *VLAN_counters* attribute set to OFF.)

seg_rx
(string,1-10,char10) Received frames.

seg_rx_disc
(string,1-10,char10) Received frames discarded.

seg_tx
(string,1-10,char10) Transmitted frames.

seg_tx_disc
(string,1-10,char10) Transmitted frames discarded.

seg_activated_TOD
(string,1-10,char10 plus *) Timestamp (in TOD clock format) representing the time at which point the VLAN most recently became active, or else an asterisk (*) if *seg_device_type*=CONN.

A VLAN ID is considered to be activated when at least one guest initialized a port on which the VLAN ID may flow. This value will be zero for a VLAN UNAWARE virtual switch, or for a VLAN AWARE virtual switch with *VLAN_counters* set to OFF.

seg_config_update_TOD
(string,1-10,char10 plus *) Timestamp (in TOD clock format) representing the time of the most recent change to the VLAN configuration, or else an asterisk (*) if *seg_device_type*=CONN.

A VLAN configuration change occurs when a port is added or removed from the list of ports on which the VLAN ID may flow. This value will be zero for a VLAN UNAWARE virtual switch, or for a VLAN AWARE virtual switch with *VLAN_counters* set to OFF.

seg_vlan_interfaces
(string,1-10,char10 plus *) Number of interfaces on which the VLAN is active, or else an asterisk (*) if *seg_device_type*=CONN.

seg_vlan_deletes
(string,1-10,char10) Number of times VLAN was deleted, when the VLAN ID is non-zero, or else an asterisk (*) if *seg_device_type*=CONN.

seg_device_type
(string,4,char26) One of the following:

CONN
Connected adaptor.

RDEV
Virtual switch.

seg_device_addr
(string,4,char16) Device address.

seg_device_status
(string,1,char10) Device status, as follows:

- If *seg_device_type*=CONN, this field will correspond to the "Port or NIC Status" field in the Port or NIC information returned by DIAGNOSE Code X'26C' (Subcode X'00000024', Return Virtual Port or Virtual NIC Information).
- If *seg_device_type*=RDEV, this field will correspond to the "Error Status" field in the RDEV information returned by DIAGNOSE Code X'26C' (Subcode X'00000020', Return Virtual Switch Information).

See the DIAGNOSE Code X'26C' documentation in *z/VM: CMS Commands and Utilities Reference* for more information on the possible values for these fields.

Usage Notes

1. Syntax errors (RC = 24 and RS = *pprr*) are only applicable to the common input parameters. Syntax checking is *not* performed on the additional input parameters for this API. If a valid parameter is specified multiple times, the last value specified for that parameter will be used. For more information, see “Key-value pair (KVP) input parameters” on page 51.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
8	RC_ERR	8	RS_NOT_AVAILABLE	This function is not available on this system
		3002	RS_INVALID_PARAMETER	Invalid parameter name
		3003	RS_INVALID_OPERAND	Invalid parameter operand
		3004	RS_MISSING_PARAMETER	Required parameter missing
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
212	RCERR_IMAGECONN	40	RS_VSWITCH_NOT_EXISTS	Virtual switch does not exist

RC	RC Name	RS	RS Name	Description
396	RCERR_INTERNAL	nnnn	psrc	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Virtual_Network_Vswitch_Set

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
switch_name_length
switch_name
grant_userid_length
grant_userid
user_vlan_id_length
user_vlan_id
revoke_userid_length
revoke_userid
real_device_address_length
real_device_address
port_name_length
port_name
controller_name_length
controller_name
connection_value
queue_memory_limit
routing_value
port_type
update_system_config_indicator
system_config_name_length
system_config_name
system_config_type_length
system_config_type
parm_disk_owner_length
parm_disk_owner
parm_disk_number_length
parm_disk_number
parm_disk_password_length
parm_disk_password
alt_system_config_name_length
alt_system_config_name
alt_system_config_type_length
alt_system_config_type
alt_parm_disk_owner_length
alt_parm_disk_owner
alt_parm_disk_number_length
alt_parm_disk_number
alt_parm_disk_password_length
alt_parm_disk_password
gvrp_value
mac_id_length
mac_id

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length

request_id

return_code

reason_code

Purpose

Use Virtual_Network_Vswitch_Set to change the configuration of an existing virtual switch.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 27.

function_name
(string,27,char43) The API function name – in this case, 'Virtual_Network_Vswitch_Set'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The virtual image name of the owner of the virtual switch.

switch_name_length
(int4) Length of *switch_name*.

switch_name
(string,1-8,char36 plus @#$_) The name of the virtual switch segment.

grant_userid_length
(int4) Length of *grant_userid*.

grant_userid
(string,0-8,char42) A userid to be added to the access list for the specified virtual switch. This userid will be allowed to connect to the switch through a QDIO device.

user_vlan_id_length
(int4) Length of *user_vlan_id*.

user_vlan_id
(string,0-19,char10 plus blank -) The user VLAN ID can be specified in the following ways:

- As single values between 1 and 4094. A maximum of four values may be specified, separated by blanks.

  Example: 1010 2020 3030 4040
- As a range of two numbers, separated by a dash (-). A maximum of two ranges may be specified.

  Example: 10-12 20-22

revoke_userid_length
(int4) Length of *revoke_userid*.

revoke_userid
(string,0-8,char42) A userid to be removed from the access list for the specified virtual switch. This userid will no longer be allowed to connect to the switch but existing connections will not be broken.

real_device_address_length
(int4) Length of *real_device_address*.

real_device_address
(string,0-14,char16 plus blank) The real device address of a real OSA-Express QDIO device used to create the switch to the virtual adapter. A maximum of three device addresses, all 1-4 characters in length, may be specified, delimited by blanks. "NONE" may also be specified.

port_name_length
(int4) Length of *port_name*.

port_name
(string,0-26,char42 plus blank) The name used to identify the OSA Expanded adapter. A maximum of three port names, all 1-8 characters in length, may be specified, delimited by blanks.

controller_name_length
(int4) Length of *controller_name*.

controller_name
One of the following:

- (string,0-71,char42 plus blank) The userid controlling the real device. A maximum of eight userids, all 1-8 characters in length, may be specified, delimited by blanks.
- (string,1,*) Specifies that any available controller may be used.

connection_value
(int1) This can be one of the following values:

0
Unspecified

1
Activate the real device connection.

2
Do *not* activate the real device connection.

queue_memory_limit
(int4) A number between 1 and 8 specifying the QDIO buffer size in megabytes. If unspecified, the default is 8.

routing_value
(int1) Specifies whether the OSA-Express QDIO device will act as a router to the virtual switch, as follows:

0
Unspecified

1
NONROUTER – The OSA-Express device identified in *real_device_address* will *not* act as a router to the virtual switch.

2
PRIROUTER – The OSA-Express device identified in *real_device_address* will act as a primary router to the virtual switch.

port_type
(int1) Specifies the port type, as follows:

0
Unspecified

1
ACCESS

2
TRUNK

update_system_config_indicator
(int1) This can be one of the following values:

0
Unspecified.

1
Update the virtual switch definition on the active system.

2
Update the virtual switch definition on the active system and in the system configuration file.

3
Update the virtual switch definition in the system configuration file.

If not specified, the default is 1.

system_config_name_length
(int4) Length of *system_config_name*.

system_config_name
(string,0-8,char42) File name of the system configuration file. The default is set by the "System_Config_File_Name =" statement in the DMSSICNF COPY file. (See the "Default SYSTEM CONFIG Link Values" section in “Configuring SMAPI” on page 30.)

system_config_type_length
(int4) Length of *system_config_type*.

system_config_type
(string,0-8,char42) File type of the system configuration file. The default is set by the "System_Config_File_Type =" statement in the DMSSICNF COPY file. (See the "Default SYSTEM CONFIG Link Values" section in “Configuring SMAPI” on page 30.)

parm_disk_owner_length
(int4) Length of *parm_disk_owner*.

parm_disk_owner
(string,0-8,char42) Owner of the parm disk. The default is set by the "Parm_Disk_Owner =" statement in the DMSSICNF COPY file. (See the "Default SYSTEM CONFIG Link Values" section in “Configuring SMAPI” on page 30.)

parm_disk_number_length
(int4) Length of *parm_disk_number*.

parm_disk_number
(string,0-4,char16) Number of the parm disk, as defined in the server's directory. The default is set by the "Parm_Disk_Number =" statement in the DMSSICNF COPY file. (See the "Default SYSTEM CONFIG Link Values" section in “Configuring SMAPI” on page 30.)

parm_disk_password_length
(int4) Length of *parm_disk_password*.

parm_disk_password
(string,0-8,charNB) Multiwrite password for the parm disk. The default is "," and should not be changed. Any value other the default is ignored. (See “Configuring SMAPI” on page 30.)

alt_system_config_name_length
(int4) Length of *alt_system_config_name*.

alt_system_config_name
(string,0-8,char42) No longer valid, maintained for backward compatibility. See Usage Note “1” on page 779.

alt_system_config_type_length
(int4) Length of *alt_system_config_type*.

alt_system_config_type
(string,0-8,char42) No longer valid, maintained for backward compatibility. See Usage Note “1” on page 779.

alt_parm_disk_owner_length
(int4) Length of *alt_parm_disk_owner*.

alt_parm_disk_owner
(string,0-8,char42) No longer valid, maintained for backward compatibility. See Usage Note “1” on page 779.

alt_parm_disk_number_length
(int4) Length of *alt_parm_disk_number*.

alt_parm_disk_number
(string,0-4,char16) No longer valid, maintained for backward compatibility. See Usage Note “1” on page 779.

alt_parm_disk_password_length
(int4) Length of *alt_parm_disk_password*.

alt_parm_disk_password
(string,0-8,charNB) No longer valid, maintained for backward compatibility. See Usage Note “1” on page 779.

gvrp_value
(int1) This can be any of the following two values:

0
Unspecified

1
GVRP

2
NOGVRP

mac_id_length
(int4) Length of *mac_id*.

mac_id
(string,0-6,char16) The MAC identifier.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Usage Notes

1. This API updates only the system configuration file on the parm disk specified, and *not* on the alternate parm disk. To maintain backward compatibility, however, the parameters for the alternate parm disk must still be specified. (The easiest way to do this is to simply specify the same values for the alternate parm disk parameters that were specified for the primary parm disk.)
2. Exactly one of the following optional parameters must be specified:
   - *grant_userid*
   - *user_vlan_id*
   - *revoke_userid*
   - *port_name*
   - *real_device_address*
   - *connection_value*
   - *queue_memory_limit*
   - *controller_name*
   - *routing_value*
   - *gvrp_value*
   - *mac_id*
3. You cannot change the characteristics of a GRANT dynamically. You must revoke the granted userid and then re-GRANT it.
4. The only attributes that may be changed in the system configuration file by this API are:
   - Addition of a userid to a virtual switch's access list
   - Removal of a userid from a virtual switch's access list.

   Refer to "Virtual_Network_Vswitch_Create" on page 716 if you wish to change other virtual switch attributes in the system configuration file.
5. If you receive return code 620, then:
   - The modification to the virtual switch is valid only during this system IPL
   - The modification to the virtual switch authorization is not updated in the z/VM system configuration file.
6. If the system administrator has changed the default location of the system configuration file, or has renamed the file, then the appropriate input parameters must be used to specify the new file information.
7. Updates for the VSMWORK1 user in the VM directory are required to link and access the CP parm disks. A link option for PMAINT CF0 must be added. If the system administrator changed the default locations of the parm disks, the VSMWORK1 userid must be granted the appropriate authority and links to the new locations.

   The following links are provided in the user directory of VSMWORK1:

```
.
IDENTITY VSMWORK1 .....
.
.
LINK PMAINT CF0 CF0 MD
```

8. If you want a different parm disk, add links to the VSMWORK1 user directory. For example:

```
.
USER VSMWORK1 .....
.
.
LINK SMAPIC5 C00 FC00 MD
```

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
		66	RS_DEF_MOD_MULTI _ERASED	Multiple DEFINE or MODIFY statements are erased in system config
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
212	RCERR_IMAGECONN	40	RS_VSWITCH_NOT_EXISTS	Virtual switch does not exist
396	RCERR_INTERNAL	nnnn	psrc	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
620	RCERR_VIRTUALNETWORKD	14	RS_FREE_MODE_NOT_AVAIL	Free modes not available
		22	RS_PARM_DISKS_SAME	System config parm disks 1 and 2 are same
		24	RS_PARM_DISK_LINK_ERROR	Error linking parm disk (1 or 2)
		28	RS_PARM_DISK_NOT_RW	Parm disk (1 or 2) not RW
		32	RS_SYS_CONF_NOT_FOUND	System config not found on parm disk 1
		34	RS_SYS_CONF_BAD_DATA	System config has bad data
		36	RS_SYS_CONF_SYNTX_ERR	Syntax errors updating system config
		38	RS_CPDISK_MODE_NOT _AVAIL	CP disk modes not available
		40	RS_PARM_DISK_FULL	Parm disk (1 or 2) is full

RC	RC Name	RS	RS Name	Description
		42	RS_PARM_DISK_ACC_NOT _ALLOWED	Parm disk (1 or 2) access not allowed
		44	RS_PDISK_PW_NOT _SUPPLIED	Parm disk (1 or 2) PW not supplied
		46	RS_PDISK_PW_INCORRECT	Parm disk (1 or 2) PW is incorrect
		48	RS_PDISK_NOT_IN_SERVER _DIRECTORY	Parm disk (1 or 2) is not in server's directory
		50	RS_CP_RELEASE_ERROR	Error in release of CPRELEASE parm disk (1 or 2)
		52	RS_CPACCESS_ERROR	Error in access of CPACCESS parm disk (1 or 2)
		54	RS_DEF_VSWITCH_EXISTS	DEFINE VSWITCH statement already exists in system config
		58	RS_REVOKE_FAILED	MODIFY VSWITCH statement to userid not found in system config
		60	RS_DEF_VSWITCH_NOT _EXIST	DEFINE VSWITCH statement does not exist in system config
		62	RS_VSWITCH_CONFLICT	DEFINE operands conflict, cannot be updated in the system config
		64	RS_DEF_MOD_MULTI _FOUND	Multiple DEFINE or MODIFY statements found in system config
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Virtual_Network_Vswitch_Set_Extended

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
switch_name=*value*
grant_userid=*value*
user_vlan_id=*value*
revoke_userid=*value*
real_device_address=*value*
port_name=*value*
controller_name=*value*
connection_value=*value*
queue_memory_limit=*value*
routing_value=*value*
port_type=*value*
persist=*value*
gvrp_value=*value*
mac_id=*value*
uplink=*value*
osd_sim=*value*
nic_userid=*value*
nic_vdev=*value*
lacp=*value*
interval=*value*
group_rdev=*value*
iptimeout=*value*
port_isolation=*value*
promiscuous=*value*
MAC_protect=*value*
VLAN_counters=*value*
nic_portselection=*value*
portnum=*value*
portnum_modify=*value*
portnum_remove=*value*
vlan_port_add=*value*
vlan_port_remove=*value*
vlan_delete=*value*
vepa=*value*
trace_size=*value*
ivl_vlanid=*value*
ivl_heartbeat=*value*
lacp_group_type=*value*
log=*value*

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length

request_id

return_code

reason_code

Purpose

Use Virtual_Network_Vswitch_Set_Extended to change the configuration of an existing virtual switch.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 36.

function_name
(string,36,char43) The API function name – in this case, 'Virtual_Network_Vswitch_Set_Extended'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The virtual image name of the owner of the virtual switch.

Note: The format for specifying the following additional input parameters is *parameter_name=value*, followed by a null (ASCIIZ) terminator. They may be specified in any order. For more information, see "Key-value pair (KVP) input parameters" on page 51.

`switch_name=`*value*
(string,1-8,char36 plus @#$_) The name of the virtual switch segment. This is a required parameter.

grant_userid=*value*
(string,0-8,char42) A userid to be added to the access list for the specified virtual switch. This userid will be allowed to connect to the switch through a QDIO device.

user_vlan_id=*value*
(string,0-19,char10 plus blank -) The user VLAN ID can be specified in the following ways:

- As single values between 1 and 4094. A maximum of four values may be specified, separated by blanks.

  Example: 1010 2020 3030 4040

- As a range of two numbers, separated by a dash (-). A maximum of two ranges may be specified.

  Example: 10-12 20-22

revoke_userid=*value*
(string,0-8,char42) A userid to be removed from the access list for the specified virtual switch. This userid will no longer be allowed to connect to the switch but existing connections will not be broken.

real_device_address=*value*
(string,0-23,char16 plus blank . P p) The real device address or the real device address and OSA Express port number of a QDIO OSA Express device to be used to create the switch to the virtual adapter. If using a real device and an OSA Express port number, specify the real device number followed by a period (.), the letter 'P' (or 'p'), followed by the port number as a hexadecimal number. A maximum of three device addresses, all 1-7 characters in length, may be specified, delimited by blanks. "None" may also be specified.

port_name=*value*
(string,0-26,char42 plus blank) The name used to identify the OSA Expanded adapter. A maximum of three port names, all 1-8 characters in length, may be specified, delimited by blanks.

controller_name=*value*
One of the following:

- (string,0-71,char42 plus blank) The userid controlling the real device. A maximum of eight userids, all 1-8 characters in length, may be specified, delimited by blanks.
- (string,1,*) Specifies that any available controller may be used.

connection_value=*value*
(string,0-10,char42) This can be one of the following values:

CONnect
Activate the real device connection.

DISCONnect
Do *not* activate the real device connection.

queue_memory_limit=*value*
(string,0-1,char16) indicates the amount of memory CP and Queued Direct I/O Hardware Facility should use for buffers for each data connection. This is defined as a range (min-max). When only one value is specified, the default max value is 256M.

Example:
8
8-256

When an OSA-Express (QDIO) Adapter is active, the lower limit (min) determines the amount of memory that can be consumed for QDIO buffers on a single data connection. The OSA-Express (QDIO) interface can operate with up to 8M of memory, so if the QUEUESTORAGE lower limit (min) is higher than 8M, the current value will be 8M instead of min (outside the configured range).

When a Network-Express (EQDIO) Adapter is active, the amount of memory allocated for EQDIO buffers should vary within the specified range (min-max). The current value is adjusted based on the level of network demand.

When multiple network devices are defined in a link aggregation group, each device port within the group will follow the same specifications.

routing_value=*value*
(string,0-9,char42) Specifies whether the OSA-Express QDIO device will act as a router to the virtual switch, as follows:

NONrouter
The OSA-Express device identified in `real_device_address=` will *not* act as a router to the virtual switch.

PRIrouter
The OSA-Express device identified in `real_device_address=` will act as a primary router to the virtual switch.

port_type=*value*
(string,0-6,char42) Specifies the port type, as follows:

ACCESS

TRUNK

persist=*value*
(string,0-3,char42) This can be one of the following values:

NO
The vswitch is updated on the active system, but is not updated in the permanent configuration for the system.

YES
The vswitch is updated on the active system and also in the permanent configuration for the system.

If not specified, the default is NO.

gvrp_value=*value*
(string,0-6,char42) This can be one of the following values:

GVRP

NOGVRP

mac_id=*value*
(string,0-6,char16) The MAC identifier.

Note: This value should only be specified for virtual switch type of QDIO.

uplink=*value*
(string,0-3,char42) One of the following:

NO
The port being enabled is not the virtual switch's UPLINK port.

YES
The port being enabled is the virtual switch's UPLINK port.

osd_sim=*value*
(string,0-3,char42) One of the following:

NO
The userid on the grant must use an IEDN or INMN type NIC adapter when coupling to a IEDN or INMN type virtual switch (respectively).

YES
A virtual NIC created by a DEFINE NIC TYPE QDIO CP command is allowed to couple to an IEDN or INMN type virtual switch.

nic_userid=*value*
One of the following:

- (string,0-8,char42) The userid of the port to/from which the UPLINK port will be connected or disconnected.
- (string,1,*) Disconnect the currently connected guest port to/from the special virtual switch UPLINK port. (This is equivalent to specifying NIC NONE on CP SET VSWITCH).

Note: If a userid (not *) is specified, then `nic_vdev=` must also be specified.

`nic_vdev=`*value*
(string,0-4,char16) The virtual device to/from which the the UPLINK port will be connected/disconnected.

Note: If this value is specified, `nic_userid=` must also be specified, with a userid.

`lacp=`*value*
(string,0-8,char42) One of the following values:

ACTIVE
Indicates that the virtual switch will initiate negotiations with the physical switch via the link aggregation control protocol (LACP) and will respond to LACP packets sent by the physical switch.

INACTIVE
Indicates that aggregation is to be performed, but without LACP.

`interval=`*value*
(string,0-8,char42) The interval to be used by the control program (CP) when doing load balancing of conversations across multiple links in the group. This can be any of the following values:

1 - 9990
Indicates the number of seconds between load balancing operations across the link aggregation group.

OFF
Indicates that no load balancing is done.

`group_rdev=`*value*
(string,0-63,char16 plus blank . P p) The real device address or the real device address and OSA Express port number of a QDIO OSA Express device to be affected within the link aggregation group associated with this vswitch. If using a real device and an OSA Express port number, specify the real device number followed by a period (.), the letter 'P' (or 'p'), followed by the port number as a hexadecimal number. A maximum of eight device addresses, all 1-7 characters in length, may be specified, delimited by blanks.

Note: If a real device address is specified, this device will be added to the link aggregation group associated with this vswitch. (The link aggregation group will be created if it does not already exist.)

`iptimeout=`*value*
(string,0-3,char10) A number between 1 and 240 specifying the length of time in minutes that a remote IP address table entry remains in the IP address table for the virtual switch.

`port_isolation=`*value*
(string,0-3,char26) One of the following:

ON

OFF

`promiscuous=`*value*
(string,0-3,char26) One of the following:

NO
The userid or port on the grant is *not* authorized to use the vswitch in promiscuous mode

YES
The userid or port on the grant is authorized to use the vswitch in promiscuous mode.

`MAC_protect=`*value*
(string,0-11,char26) One of the following:

ON

OFF

UNSPECified

VLAN_counters=*value*
(string,0-3,char26) One of the following:

ON

OFF

nic_portselection=*value*
(string,0-7,char26) One of the following:

AUTO
CP will assign the port number

PORTNUM
The application specifies the port number.

If not specified, AUTO is the default. If specified, `nic_userid=` must also be specified.

portnum=*value*
(string,0-16,char42 plus blank) Port number, followed by the user ID. This parameter may be specified with one or more of the following:

- `port_type=`*value*
- `promiscuous=`*value*
- `osd_sim=`*value*
- `user_vlan_id=`*value*

portnum_modify=*value*
(string,0-16,char16) Port number to modify. This parameter *must* be specified with one or more of the following:

- `port_type=`*value*
- `promiscuous=`*value*
- `osd_sim=`*value*
- `user_vlan_id=`*value*

portnum_remove=*value*
(string,0-16,char16) Port number to remove.

vlan_port_add=*value*
(string,0-maxlength,char42 plus blank) The VLAN ID, followed by a set of valid port numbers (between 1 and 2048, inclusive). This set may contain ranges.

vlan_port_remove=*value*
(string,0-maxlength,char42 plus blank) The VLAN ID, followed by a set of valid port numbers (between 1 and 2048, inclusive). See examples above in `vlan_port_add=`*value*.

vlan_delete=*value*
(string,0-8,char42) The VLAN ID to be deleted.

vepa=*value*
(string,0-3,char26) The operational mode of the virtual switch with regard to forwarding guest-to-guest and guest-to-external destination communications, as follows:

ON
Prohibits guests from sending traffic to other guests on the same virtual switch, without going through an external entity by forwarding all traffic from the guest through the OSA uplink to an adjacent switch. In addition, no direct LPAR communications sharing the same OSA port are permitted with the guest ports of the virtual switch. All traffic from the virtual switch destined for any sharing hosts/LPARs on the same OSA port will be forwarded, as well. Any traffic destined for the virtual switch guest ports from hosts/LPARs sharing the same OSA port will also be forwarded to the adjacent switch.

Note:

1. vepa=ON requires an ETHERNET virtual switch (without a bridge port), with OSA uplink(s) that supports VEPA. Also, the partner switch must support reflective relay.
2. You may *not* specify vepa=ON if port_isolation=ON is also specified.

OFF
Allows guests to communicate with each other and with any hosts and/or LPARs that share the same OSA port. This is the default setting for a QDIO or an IEDN virtual switch.

trace_size=*value*
(string,0-4,char10; range 0-4095) Specifies the number of pages to be allocated for an internal trace table to keep track of trace events pertaining to this UPLINK port. If not specified, the default value is 8.

ivl_vlanid=*value*
(string,0-4,char10; range 1-4094) Specifies the VLAN ID associated with the IVL port on an IVL virtual switch that is VLAN AWARE.

ivl_heartbeat=*value*
(string,0-4,char10; range 10-3600) Specifies the length of time that is allowed to expire before the local system declares a communication problem with another system in the IVL domain if heartbeat signals are missing. If not specified, the default value is 30 seconds.

lacp_group_type=*value*
(string,0-9,char42) Specifies the port group type as one of the following:

EXClusive
An exclusive port group where each OSA-E feature supports only a single VSWITCH QDIO connection. (In other words, no sharing of the OSA-Express feature is permitted.)

SHAred
A shared port group in which one or more global virtual switches and global virtual switch members share the OSA-Express feature(s) that make up the port group.

Notes:

1. SHAred may not be specified when **lacp**=INACTIVE is specified.
2. If not specified, when **lacp**=ACTIVE, the default value is EXClusive. When **lacp**=INACTIVE, the default (and only possible) value is SHAred.
3. If specified, **lacp**=*value* must also be specified and must not be null in the parameter list. Otherwise, it will result in RC=8/RS=3004 (Missing Parameter) if **lacp**=*value* is not specified and RC=8/RS=3003 (Invalid Parameter) if **lacp**=*value* is specified as a null value.

log=*value*
(string,1-7,char26) sets VSwitch activity logging behavior for an active Network-Express (EQDIO) interface. Activity associated with EQDIO device operation is displayed on the system operator console (not on a controller virtual machine console). The LOG parameter has no effect when an OSA-Express (QDIO) device is active. In that case, the associated messages are logged on the active controller virtual machine console.

OFF
sets VSwitch activity logging OFF. Normal activity will not be logged to the system operator console. This is the default VSwitch logging mode. Note, however, that errors and unusual events will always be logged to the system operator console regardless of the LOG setting.

ON
sets VSwitch message logging to display brief messages describing each significant event associated with EQDIO device operation. This setting is intended to be used by IBM personnel when diagnosing EQDIO interface problems.

Verbose
Sets VSwitch message logging to display more detailed messages describing each significant event associated with EQDIO device operation. This setting is intended to be used by IBM personnel when diagnosing EQDIO interface problems.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Usage Notes

1. Exactly one parameter can be specified on any one call to this function. To set multiple attributes, multiple calls to this function will be necessary.
2. Syntax errors (RC = 24 and RS = *pprr*) are only applicable to the common input parameters. Syntax checking is *not* performed on the additional input parameters for this API. If a valid parameter is specified multiple times, the last value specified for that parameter will be used. For more information, see "Key-value pair (KVP) input parameters" on page 51.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
8	RC_ERR	24	RS_CONFLICTING_PARMS	Conflicting input parameters
		3002	RS_INVALID_PARAMETER	Invalid parameter name
		3003	RS_INVALID_OPERAND	Invalid parameter operand
		3004	RS_MISSING_PARAMETER	Required parameter missing
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
212	RCERR_IMAGECONN	40	RS_VSWITCH_NOT_EXISTS	Virtual switch does not exist
		44	RS_ALREADY_AUTH	Image already authorized
396	RCERR_INTERNAL	nnnn	psrc	Internal system error - product-specific return code (See "Internal Return Codes (RC = 396, 592, or 596)" on page 834)

RC	RC Name	RS	RS Name	Description
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

VMRELOCATE

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
`destination=`*value*
`action=`*value*
`force=`*value*
`immediate=`*value*
`max_total=`*value*
`max_quiesce=`*value*

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
VMRELOCATE_error_record

Purpose

Use VMRELOCATE to relocate, test relocation eligibility, or cancel the relocation of the specified virtual machine, while it continues to run, to the specified system within the z/VM SSI cluster.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 10.

function_name
(string,25,char43) The API function name – in this case, 'VMRELOCATE'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).

- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The name of the virtual machine whose relocation to another system within the z/VM SSI cluster will be initiated, tested, or canceled.

Note: The format for specifying the following additional input parameters is *parameter_name*=*value*, followed by a null (ASCIIZ) terminator. They may be specified in any order. For more information, see "Key-value pair (KVP) input parameters" on page 51.

`destination`=*value*
(string,1-8,char42) The z/VM SSI cluster name of the destination system to which the specified virtual machine will be relocated. This is a required parameter.

Note that this parameter is not used when canceling a relocation. If it is specified with `action`=CANCEL, it will be ignored.

`action`=*value*
(string,0-6,char42) One of the following:

MOVE
Initiate a VMRELOCATE MOVE of the virtual machine specified in *target_identifier*.

TEST
Test the specified virtual machine and determine if it is eligible to be relocated to the specified system.

If TEST is specified, all other valid additional input parameters except `destination`= are ignored. If `action`= is not specified, TEST is the default.

CANCEL
Stop the relocation of the specified virtual machine.

If CANCEL is specified, all other additional input parameter are ignored.

`force`=*value*
(string,0-27,char42 plus blank) Any combination of the following may be specified, in any order:

ARCHITECTURE
Indicates that relocation is to be attempted even though the virtual machine is currently running on a system with hardware architecture facilities or CP-supplied features not available on the destination system (for example, when relocating to a system running an earlier release of CP).

DOMAIN
Indicates that relocation is to be attempted even though the virtual machine would be moved outside of its domain.

STORAGE
Indicates that relocation should proceed even if CP determines that there are insufficient storage resources available on the destination, following memory capacity assessment checks.

For example, to choose all three options, specify `force=ARCHITECTURE DOMAIN STORAGE`.

`immediate=`*value*
(string,0-3,char42) One of the following:

NO
Specifies regular processing. This is the default. The VMRELOCATE command will go through several passes of virtual memory before going to the quiesce stage. The default for `max_total=` is NOLIMIT, and the default for `max_quiesce=` is 10 seconds when `immediate=`NO is specified.

YES
The VMRELOCATE command will do one early pass through virtual machine storage and then go directly to the quiesce stage. The defaults for both `max_total=` and `max_quiesce=` are NOLIMIT when `immediate=`YES is specified.

`max_total=`*value*
(string,0-8,char42) One of the following:

NOLIMIT
Specifies that there is no limit on the total amount of time the system should allow for this relocation. The relocation will therefore not be canceled due to time constraints. This is the default.

value
The maximum total time (in seconds) that the command issuer is willing to wait for the entire relocation to complete. The range for this value is 1-99999999.

`max_quiesce=`*value*
(string,0-8,char42) One of the following:

NOLIMIT
Specifies that there is no limit on the total quiesce time the system should allow for this relocation.

value
The maximum quiesce time (in seconds) a virtual machine may be stopped during a relocation attempt. The range for this value is 1-99999999.

The default is NOLIMIT if `immediate=`YES is specified, or 10 seconds if not.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

VMRELOCATE_error_record
(string,1-maxlength,char42 plus blank) The error/information message number of the errors preventing the virtual machine from being relocatable. Each 4-digit number is extracted from the

HCP*nnnn*E, HCP*nnnn*W, or HCP*nnnn*I message generated, separated by blanks, and then the entire record is terminated with a null (ASCIIZ) terminator.

Note that this error record will be returned only if:

- A VMRELOCATE MOVE results in RC = 8 and RS = 3000, or
- A VMRELOCATE TEST results in RC = 4 and RS = 3000.

Usage Notes

1. The virtual machine being relocated must be active on the system on which this API is issued.
2. The `action=TEST` option may used to verify that the target virtual machine is eligible for relocation prior to requesting that a relocation be executed.
3. Syntax errors (RC = 24 and RS = *pprr*) are only applicable to the common input parameters. Syntax checking is *not* performed on the additional input parameters for this API. If a valid parameter is specified multiple times, the last value specified for that parameter will be used. For more information, see "Key-value pair (KVP) input parameters" on page 51.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
4	RC_WNG	3000	RS_RELOCATION_ERRORS	VMRELOCATE TEST error
8	RC_ERR	3000	*psrc*	VMRELOCATE MOVE error - product-specific return code (reason code is the number of CP error code(s) returned)
		3002	RS_INVALID_PARAMETER	Invalid parameter name
		3003	RS_INVALID_OPERAND	Invalid parameter operand
		3004	RS_MISSING_PARAMETER	Parameter is missing
24	RCERR_SYNTAX	*pprr*	*pprr*	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
396	RCERR_INTERNAL	*nnnn*	*psrc*	Internal system error - product-specific return code (See "Internal Return Codes (RC = 396, 592, or 596)" on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory

RC	RC Name	RS	RS Name	Description
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

VMRELOCATE_Image_Attributes

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
`relocation_setting=`*value*
`domain_name=`*value*
`archforce=`*value*

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code

Purpose

Use VMRELOCATE_Image_Attributes to modify the relocation setting for a specified image.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 27.

function_name
(string,27,char43) The API function name – in this case, 'VMRELOCATE_Image_Attributes'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See “Client Authentication” on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The user ID whose relocation capability is being set. If "*" is specified, the target user is the command issuer.

Note: The format for specifying the following additional input parameters is *parameter_name=value*, followed by a null (ASCIIZ) terminator. They may be specified in any order. For more information, see "Key-value pair (KVP) input parameters" on page 51.

`relocation_setting=`*value*
(string,2-3,char26) One of the following:

ON
Enables relocation for the specified user.

OFF
Disables relocation for the specified user.

This is a required parameter.

`domain_name=`*value*
(string,0-8,char42) The domain currently associated with a user. If unspecified, the currently associated domain is assumed.

`archforce=`*value*
(string,0-3,char26) One of the following:

YES
Specifies the FORCE ARCHITECTURE option, in which the virtual machine is assigned to the new domain even if it means the guest's virtual architecture will be set to a level with less capability than it had in its original domain.

NO
The guest's virtual machine will *not* be set to a new domain.

If unspecified, the default is NO.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

Usage Notes

1. The user whose relocation capability is being set must be logged on.
2. The relocation attribute cannot be set for an IDENTITY.
3. Syntax errors (RC = 24 and RS = *pprr*) are only applicable to the common input parameters. Syntax checking is *not* performed on the additional input parameters for this API. If a valid parameter is specified multiple times, the last value specified for that parameter will be used. For more information, see "Key-value pair (KVP) input parameters" on page 51.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
8	RC_ERR	12	RS_NOT_LOGGED_ON	*target_identifier* not logged on
		1821	RS_NONEXISTENT_DOMAIN	Relocation domain *domain_name* does not exist
		1822	RS_NO_FORCE_ ARCHITECTURE	User *target_identifier* cannot be set to a new relocation domain *domain_name* without the FORCE ARCHITECTURE option
		1823	RS_IDENTITY_RELOCATION	A multiconfiguration virtual machine cannot be relocated
		3002	RS_INVALID_PARAMETER	Invalid parameter name
		3003	RS_INVALID_OPERAND	Invalid parameter operand
		3004	RS_MISSING_PARAMETER	Required parameter missing
		3008	RS_NOT_SSI_MEMBER	System is not a member of an SSI cluster
24	RCERR_SYNTAX	*pprr*	*pprr*	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
396	RCERR_INTERNAL	*nnnn*	*psrc*	Internal system error - product-specific return code (See "Internal Return Codes (RC = 396, 592, or 596)" on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist

RC	RC Name	RS	RS Name	Description
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

VMRELOCATE_Modify

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
`max_total=`*value*
`max_quiesce=`*value*

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
VMRELOCATE_error_record (error only)

Purpose

Use VMRELOCATE_Modify to modify the time limits associated with a relocation already in progress for the specified image.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 17.

function_name
(string,17,char43) The API function name – in this case, 'VMRELOCATE_Modify'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See “Client Authentication” on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) The name of the image, already in the process of relocation, for which the time limits should be modified.

Note: The format for specifying the following additional input parameters is *parameter_name=value*, followed by a null (ASCIIZ) terminator. They may be specified in any order. For more information, see "Key-value pair (KVP) input parameters" on page 51.

`max_total=`*value*
One of the following:

- (string,0-7,NOLIMIT) Indicates that there is no limit on the total amount of time the system should allow for this relocation. The relocation will not be canceled due to time constraints. This is the default if unspecified.
- (string,0-8,char10; range 1-99999999) The maximum total time (in seconds) that the command issuer is willing to wait for the entire relocation to complete.

See Usage Note "1" on page 802.

`max_quiesce=`*value*
One of the following:

- (string,7,NOLIMIT) Indicates that there is no limit on the total quiesce time the system should allow for this relocation.
- (string,1-8,char10; range 1-99999999) The maximum quiesce time (in seconds) a virtual machine may be stopped during a relocation attempt. The default, if unspecified, is 10 seconds.

See Usage Note "1" on page 802.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

If RC=8 and RS=3010, the following parameter will also be returned:

VMRELOCATE_error_record
(string) The 4-digit error/information message number of the errors specifying why the relocation time limits cannot be modified. These 4-digit numbers are extracted from each HCP*xxxx*E, HCP*xxxx*W, and HCP*xxxx*I message generated. Each 4-digit error message is separated by a blank, and the record is ended with a null (ASCIIZ) terminator.

Usage Notes

1. `max_total=`*value* and `max_quiesce=`*value* are both optional input parameters, but at least one of them must be specified.
2. The virtual machine being moved must be logged on to the system on which this command is issued.
3. Syntax errors (RC = 24 and RS = *pprr*) are only applicable to the common input parameters. Syntax checking is *not* performed on the additional input parameters for this API. If a valid parameter is specified multiple times, the last value specified for that parameter will be used. For more information, see "Key-value pair (KVP) input parameters" on page 51.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
8	RC_ERR	3002	RS_INVALID_PARAMETER	Invalid parameter name
		3003	RS_INVALID_OPERAND	Invalid parameter operand
		3004	RS_MISSING_PARAMETER	Required parameter missing
		3010	RS_RELOCATION_MODIFY_ERROR	VMRELOCATE modify error
24	RCERR_SYNTAX	*pprr*	*pprr*	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
396	RCERR_INTERNAL	*nnnn*	*psrc*	Internal system error - product-specific return code (See "Internal Return Codes (RC = 396, 592, or 596)" on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

VMRELOCATE_Status

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
`status_target=`*value*

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
VMRELOCATE_status_array (1)
  VMRELOCATE_status_structure (2)
    VMRELOCATE_image
    VMRELOCATE_source_system
    VMRELOCATE_destination_system
    VMRELOCATE_by
    VMRELOCATE_elapsed
    VMRELOCATE_status

Note:

1. An array consists of zero or more of its components.
2. A structure consists of one set of its components.

Purpose

Use VMRELOCATE_Status to obtain information about relocations currently in progress.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 17.

function_name
(string,25,char43) The API function name – in this case, 'VMRELOCATE_Status'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) This must match an entry in the authorization file that also contains the *authenticated_userid* and the *function_name* (VMRELOCATE_Status).

Note: The format for specifying the following additional input parameter is *parameter_name*=*value*, followed by a null (ASCIIZ) terminator. For more information, see "Key-value pair (KVP) input parameters" on page 51.

`status_target=`*value*
(string,0-13,char42 plus blank) One of the following:

ALL
Specifies that the status of all relocations currently in progress on this system are displayed.

USER *userid*
Display relocation status of the virtual machine with name *userid*.

INCOMING
Display status of all incoming relocations.

OUTGOING
Display status of all outgoing relocations.

If unspecified, ALL is the default.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

VMRELOCATE_status_array
(array) An array consisting of one or more instances of *VMRELOCATE_status_structure*, as follows:

VMRELOCATE_status_structure
(structure) A structure consisting of one set of the following parameters, with a blank separating each parameter. Each structure is then terminated by a null (ASCIIZ) character.

VMRELOCATE_image
(string,1-8,char42) The virtual machine being relocated.

VMRELOCATE_source_system
(string,1-8,char42) The system from which this image is being moved.

VMRELOCATE_destination_system
(string,1-8,char42) The system to which the image is being moved.

VMRELOCATE_by
(string,1-8,char42) The userid that initiated the relocate

VMRELOCATE_elapsed
(string,8,char42) Time elapsed (*hh*:*mm*:*ss*) since this relocation started

VMRELOCATE_status
(string,0-15,char26 plus / _) The point in the relocation process that the image has currently reached. The following values are possible:

CONNECTING
The source system is connecting to the destination system.

ELIG_CHECKS
Relocation eligibility checking is in progress.

CREATING_GUEST
Creating skeleton guest on the destination system.

MOVING_MEMORY
The virtual machine's memory is being transferred to the destination system.

STOPPING_GUEST
The virtual machine is being stopped on the source system.

MOVING_GUEST
The virtual machine state is being moved to the destination system.

FINAL_MEM_COPY
The final pass of memory transfer is in progress.

FINAL_I/O_CHECK
The final I/O check is in progress.

RESUME_GUEST
The virtual machine is being started on the destination system.

CLEANING_UP
The relocation is finished and cleanup work is being done.

TERMINATING
The relocation had an error or was canceled and is in the process of terminating.

TEST
A VMRELOCATE TEST is in progress for this virtual machine.

COMM_ERROR
The VMRELOCATE STATUS command was issued on the destination system and a communications error occurred when attempting to retrieve the current status from the source system

Usage Notes

1. Status is only available for relocations involving the system where the status request is issued.
2. Syntax errors (RC = 24 and RS = *pprr*) are only applicable to the common input parameters. Syntax checking is *not* performed on the additional input parameters for this API. If a valid parameter is specified multiple times, the last value specified for that parameter will be used. For more information, see "Key-value pair (KVP) input parameters" on page 51.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
4	RC_WNG	3001	RS_NO_RELOCATION_ACTIVE	No active relocations found
8	RC_ERR	3002	RS_INVALID_PARAMETER	Invalid parameter name
		3003	RS_INVALID_OPERAND	Invalid parameter operand
		nnnn	RS_VMRELOCATE_ERROR	VMRELOCATE_Status returned an error. The RS *nnnn* represents the HCP*nnnn* message.
24	RCERR_SYNTAX	*pprr*	*pprr*	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
396	RCERR_INTERNAL	*nnnn*	*psrc*	Internal system error - product-specific return code (See "Internal Return Codes (RC = 396, 592, or 596)" on page 834)
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

VMRM_Configuration_Query

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
configuration_file_name_length
configuration_file_name
configuration_file_type_length
configuration_file_type
configuration_dir_name_length
configuration_dir_name

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
configuration_file_length
configuration_file

Purpose

Use VMRM_Configuration_Query to query the contents of the VMRM configuration file.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 24.

function_name
(string,24,char43) The API function name – in this case, 'VMRM_Configuration_Query'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) This must match an entry in the authorization file that also contains the *authenticated_userid* and the *function_name* (VMRM_Configuration_Query).

configuration_file_name_length
(int4) Length of *configuration_file_name*.

configuration_file_name
(string,1-8,char43) The name of the configuration file.

configuration_file_type_length
(int4) Length of *configuration_file_type*.

configuration_file_type
(string,1-8,char43) The file type of the configuration file.

configuration_dir_name_length
(int4) Length of *configuration_dir_name*.

configuration_dir_name
(string,1-153,char43 plus .) The fully-qualified Shared File System (SFS) directory name where the configuration file is located. In addition to <char43>, a period (.) can also be used. See *z/VM: CMS Commands and Utilities Reference* for more information about SFS directory names.

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

configuration_file_length
(int4) Length of *configuration_file*. See description below.

configuration_file
(string,1-maxlength,charNA) The contents of the specified configuration file. There is no limit to its length, so long as you have enough virtual storage.

Usage Notes

1. To query the current configuration file, specify $CURRCFG for the *configuration_file_name*, $SAVE for the *configuration_file_type*, and "VMSYS:VMRMSVM." (the default filepool) for the *configuration_dir_name*.
2. The SFS directory used by VMRM is the default filepool and directory shipped with z/VM unless changed by an administrator. The constant VMRM_SFSDir is set to "VMSYS:VMRMSVM." in the IRMCONS COPY file used by VMRM, and DMSSICNF COPY used by the VSMWORK1 userid (SMAPI server). If the administrator changes the default filepool for these userids, then the constant must be updated as well to match the changed directory name. See "Naming Shared File System (SFS) Directories" in *z/VM: CMS Commands and Utilities Reference* for more information on directories. The updates should be made as local modifications using the automated local modification procedure. Refer to *z/VM: Service Guide* for more information on using this procedure.
3. Access to the Shared File System is required for the *authenticated_userid* to execute this function. The VMRMSVM (Virtual Machine Resource Manager) server virtual machine must be started and managing the workload or workloads specified in the configuration file.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
396	RCERR_INTERNAL	nnnn	psrc	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
800	RCERR_VMRM	16	RS_CANNOT_ACCESS_DATA	Not authorized to access file
		28	RS_FILE_NOT_FOUND	Specified configuration file not found
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

VMRM_Configuration_Update

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
configuration_file_name_length
configuration_file_name
configuration_file_type_length
configuration_file_type
configuration_dir_name_length
configuration_dir_name
syncheck_only
update_file_length
update_file

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
log_record_array_length
log_record_array (1)
    log_record_structure (2)
        log_record_length
        log_record

Note:

1. An array consists of zero or more of its components.
2. A structure consists of one set of its components.

Purpose

Use VMRM_Configuration_Update to add, delete, and change VMRM configuration file statements.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 25.

function_name
(string,25,char43) The API function name – in this case, 'VMRM_Configuration_Update'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) This must match an entry in the authorization file that also contains the *authenticated_userid* and the *function_name* (VMRM_Configuration_Update).

configuration_file_name_length
(int4) Length of *configuration_file_name*.

configuration_file_name
(string,1-8,char43) The name of the configuration file.

configuration_file_type_length
(int4) Length of *configuration_file_type*.

configuration_file_type
(string,1-8,char43) The file type of the configuration file.

configuration_dir_name_length
(int4) Length of *configuration_dir_name*.

configuration_dir_name
(string,1-153,char43 plus .) The fully-qualified Shared File System (SFS) directory name where the configuration file is located. In addition to <char43>, a period (.) can also be used. See *z/VM: CMS Commands and Utilities Reference* for more information about SFS directory names.

syncheck_only
(int1) Specify a 1 to choose the **SYNCHECK** option, meaning that only a syntax check of the configuration is done, without processing a configuration file update. Otherwise, specify a 0 to indicate that both a syntax check and a configuration file update should occur.

Note that when **SYNCHECK** is specified, the updates are validated for correct syntax, but the configuration file is not changed. This can be useful when the configuration file specified is the same as that named in an **ADMIN NEWCFG** statement.

update_file_length
(int4) Length of *update_file*. See description below.

update_file
(string,1-maxlength,charNA) A new, complete VMRM configuration file to syntax-check or to replace the old file. (For information about VMRM configuration file formats, see "VMRM SVM Tuning Parameters" in *z/VM: Performance*.) There is no limit to its length, so long as you have enough virtual storage.

Note: Records in the *update_file* must be separated by nulls (X'00').

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

log_record_array_length
(int4) Length of *log_record_array*.

log_record_array
(array) An array consisting of zero or more instances of *log_record_structure*, as follows:

log_record_structure
(structure) A structure consisting of one set of *log_record_length* and *log_record*, as follows:

log_record_length
(int4) Length of *log_record*. See description below.

log_record
(string,1-maxlength,charNA) Records in the VMRM log file. These are the actual messages received from the VMRM server machine. There is no limit to its length, so long as you have enough virtual storage.

Usage Notes

1. Access to the Shared File System is required for the *authenticated_userid* to execute this function. The VMRMSVM (Virtual Machine Resource Manager) server virtual machine must be started and managing the workload or workloads.
2. When specifying the location of the configuration to be updated, ensure that the VSMWORK1 server has write access to that directory.
3. The SFS directory used by VMRM is the default filepool and directory shipped with z/VM unless changed by an administrator. The constant VMRM_SFSDir is set to "VMSYS:VMRMSVM." in the IRMCONS COPY file used by VMRM, and DMSSICNF COPY used by the VSMWORK1 userid (SMAPI server). If the administrator changes the default filepool for these userids, then the constant must be updated as well to match the changed directory name. See "Naming Shared File System (SFS) Directories" in *z/VM: CMS Commands and Utilities Reference* for more information on directories. The

updates should be made as local modifications using the automated local modification procedure. Refer to *z/VM: Service Guide* for more information on using this procedure.

4. Records in the *update_file* must be separated by nulls (X'00').

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
396	RCERR_INTERNAL	nnnn	psrc	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
800	RCERR_VMRM	12	RS_UPDATE_SYNTAX_ERROR	Incorrect Syntax In Update Data
		24	RS_UPDATE_WRITE_ERROR	Error writing file(s) to directory
		28	RS_FILE_NOT_FOUND	Specified configuration file not found
		32	RS_UPDATE_PROCESS _ERROR	Internal error processing updates
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

VMRM_Measurement_Query

Input Parameters:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier

Response 1 – Immediate Request Verification:

request_id

Response 2 – Output Parameters:

output_length
request_id
return_code
reason_code
query_timestamp_length
query_timestamp
file_spec_length
file_spec
file_timestamp_length
file_timestamp
workload_array_length
workload_array (1)
workload_structure (2)
workload_length
workload

Note:

1. An array consists of zero or more of its components.
2. A structure consists of one set of its components.

Purpose

Use VMRM_Measurement_Query to obtain current VMRM measurement values.

Input Parameters

input_length
(int4) The total length of all input parameters (after this one).

function_name_length
(int4) Length of *function_name* – in this case, 22.

function_name
(string,22,char43) The API function name – in this case, 'VMRM_Measurement_Query'.

authenticated_userid_length
(int4) Length of *authenticated_userid*.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

password_length
(int4) Length of *password*.

password
One of the following:

- (string,1-200,charNA) The password or passphrase to be used for authentication (AF_INET requests).
- (string,0-200,charNA) The password or passphrase to be used for authentication (AF_IUCV requests).

Note that *password* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier_length
(int4) Length of *target_identifier*.

target_identifier
(string,1-8,char42) This must match an entry in the authorization file that also contains the *authenticated_userid* and the *function_name* (VMRM_Measurement_Query).

Response 1 -- Immediate Request Verification

request_id
(int4) The identifier of the request.

Response 2 -- Output Parameters

output_length
(int4) The total length of all output parameters (after this one).

request_id
(int4) The identifier of the request (same as returned in immediate request verification above).

return_code
(int4) The return code.

reason_code
(int4) The reason code.

query_timestamp_length
(int4) Length of *query_timestamp*.

query_timestamp
(string,1-17,char42) The timestamp when the query was issued.

file_spec_length
(int4) Length of *file_spec*.

file_spec
(string,1-20,char43) The file name of the active configuration file.

file_timestamp_length
(int4) Length of *file_timestamp*.

file_timestamp
(string,1-12, char42 plus / ,) The timestamp of the active configuration file.

workload_array_length
(int4) Length of *workload_array*.

workload_array
(array) An array consisting of zero or more instances of *workload_structure*, as follows:

workload_structure
(structure) A structure consisting of one set of *workload_length* and *workload*, as follows:

workload_length
(int4) Length of *workload*.

workload
(string,1-35,charNA) Each workload entry will contain the following:

- *workload_name*
- *CPU keyword and actual_value*
- *DASD keyword and actual_value*
- For example:

```
WORKNAME1        CPU 10 DASD 20
WORKNAME2        CPU NULL DASD NULL
```

Usage Notes

1. Access to the Shared File System is required for the *authenticated_userid* to execute this function. The VMRMSVM (Virtual Machine Resource Manager) service virtual machine must be started and managing the workload or workloads specified in the configuration file.
2. The SFS directory used by VMRM is the default filepool and directory shipped with z/VM unless changed by an administrator. The constant VMRM_SFSDir is set to "VMSYS:VMRMSVM." in the IRMCONS COPY file used by VMRM, and DMSSICNF COPY used by the VSMWORK1 userid (SMAPI server). If the administrator changes the default filepool for these userids, then the constant must be updated as well to match the changed directory name. See "Naming Shared File System (SFS) Directories" in *z/VM: CMS Commands and Utilities Reference* for more information on directories. The updates should be made as local modifications using the automated local modification procedure. Refer to *z/VM: Service Guide* for more information on using this procedure.
3. Even if the VMRM machine is not creating new measurement data, it is possible that the VMRM_Measurement_Query function will return old data; that is, data from the last time actual goal measurement data was available. The configuration file name, date, and timestamp of when the data was collected will appear at the beginning of the file or return buffer.

Return and Reason Codes

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager

RC	RC Name	RS	RS Name	Description
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
396	RCERR_INTERNAL	nnnn	psrc	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
800	RCERR_VMRM	8	RS_NO_MEASUREMENT_DATA	No measurement data exists
		16	RS_CANNOT_ACCESS_DATA	Not authorized to access file
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Chapter 7. Return and Reason Code Summary

The following return codes and reason codes are used by the Systems Management APIs.

All Return Codes (Including Internal)

Table 21. All Return Codes (Including Internal)

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
0	RC_OK	4	RS_NOT_FOUND	Segment was created or replaced, but specified userid in *memory_access_identifier* could not be found to give RSTD access
0	RC_OK	4	RS_AFFINITY_SUPPRESSED	CPU defined, but CPU affinity suppressed
0	RC_OK	8	RS_OFFLINE	Request successful; object directory offline
0	RC_OK	8	RS_AUTHERR_ESM	Password request not authorized by external security manager
0	RC_OK	12	RS_NAMESAVE_EXISTS	Request successful; NAMESAVE statement already exists in directory
0	RC_OK	12	RS_NEW_LIST	Request successful; new list created
0	RC_OK	12	RS_NOT_ACTIVE	Image not active
0	RC_OK	12	RS_LOCKED	Image or device(s) locked
0	RC_OK	16	RS_LIST_DESTROYED	Request successful; no more entries, list destroyed
0	RC_OK	20	RS_VMLAN_CREATED	Request successful; new virtual network LAN created
0	RC_OK	20	RS_NOT_AUTHORIZED	No output; user(s) not authorized for specified segment
0	RC_OK	24	RS_VMLAN_REMOVED	Request successful; virtual network LAN removed
0	RC_OK	24	RS_UNLOCKED	Image or device(s) unlocked
0	RC_OK	28	RS_NONE_FOUND	No matching entries found. Return buffer is empty.
0	RC_OK	28	RS_EMPTY	There are no SCSI characteristics for this image.

© Copyright IBM Corp. 2003, 2025

Table 21. All Return Codes (Including Internal) (continued)

RC	RC Name	RS	RS Name	Description
0	RC_OK	28	RS_SEGMENT_NOT_FOUND	Query request successful, but segment not found
0	RC_OK	28	RS_NOTIFY_NOT_FOUND	No matching entries found
0	RC_OK	28	RS_LINK_NOT_FOUND	No links to disk found
0	RC_OK	32	RS_NOT_IN_LIST	Name was not in list
0	RC_OK	36	RS_NAME_IN_LIST	Name is already in list
0	RC_OK	40	RS_VSWITCH_CREATED	Request successful; new virtual switch created
0	RC_OK	44	RS_VSWITCH_REMOVED	Request successful; virtual switch removed
0	RC_OK	66	RS_DEF_MOD_MULTI _ERASED	Multiple DEFINE or MODIFY statements are erased in system config
0	RC_OK	100	RS_ASYNC_OP_SUCCEEDED	Asynchronous operation succeeded
0	RC_OK	104	RS_ASYNC_OP_IN _PROGRESS	Asynchronous operation in progress
0	RC_OK	108	RS_ASYNC_OP_FAILED	Asynchronous operation failed
0	RC_OK	720	RS_720	The API functional level is z/VM 7.2.
4	RC_WNG	4	RS_NOT_FOUND	Request does not exist
4	RC_WNG	4	RS_IFCONFIG_WARNING	The command completed successfully, but a warning condition was detected on IFCONFIG command
4	RC_WNG	5	RS_UNRESTRICTED_LAN	Unrestricted LAN
4	RC_WNG	6	RS_NO_USERS	No authorized users
4	RC_WNG	8	RS_DEV_NOT_FOUND	Device does not exist
4	RC_WNG	8	RS_NOT_EXIST	No device EQIDs found
4	RC_WNG	28	RS_EMPTY	Return buffer is empty
4	RC_WNG	3000	RS_RELOCATION_ERRORS	VMRELOCATE TEST error
4	RC_WNG	3001	RS_NO_RELOCATION_ACTIVE	No active relocations found
4	RC_WNG	3008	RS_NOT_SSI_MEMBER	System is not a member of an SSI cluster
4	RC_WNG	3009	RS_REPAIR_IPL_PARAM	System was IPLed with the REPAIR IPL parameter
4	RC_WNG	3022	RS_NO_INTERFACE_EXIST	No interface configured on specified TCP/IP stack virtual machine
8	RC_ERR	2	RS_INVALID_USER	Invalid access user

Table 21. All Return Codes (Including Internal) (continued)

RC	RC Name	RS	RS Name	Description
8	RC_ERR	3	RS_INVALID_OP	Invalid op value
8	RC_ERR	4	RS_INVALID_PRO	Invalid promiscuity value
8	RC_ERR	4	RS_NOT_FOUND	Directory entry to be deleted not found
8	RC_ERR	4	RS_NOT_FOUND	Performance monitoring virtual server not found
8	RC_ERR	4	RS_NOT_FOUND	APAR or PTF not found
8	RC_ERR	4	RS_NOT_FOUND	Specified interface not found
8	RC_ERR	8	RS_DEV_NOT_FOUND	Device does not exist
8	RC_ERR	8	RS_NOT_AVAILABLE	Input parameter value not supported
8	RC_ERR	10	RS_DEV_NOT_AVAIL_TO_ ATTACH	Device not available for attachment
8	RC_ERR	12	RS_DEV_NOT_VOLUME	Device not a volume
8	RC_ERR	12	RS_NOT_LOGGED_ON	*target_identifier* not logged on
8	RC_ERR	12	RS_IFCONFIG_ERROR	An error was encountered on IFCONFIG command
8	RC_ERR	13	RS_INVALID_KEY	Match key length does not match the match key specified
8	RC_ERR	14	RS_FREE_MODE_NOT_AVAIL	Free modes not available
8	RC_ERR	16	RS_IFCONFIG_UNEXPECTED	An unexpected condition was encountered on IFCONFIG command
8	RC_ERR	18	RS_VOLUME_NOT_FOUND	Volume does not exist
8	RC_ERR	19	RS_CP_OWNED	Volume is CP owned and cannot be used
8	RC_ERR	20	RS_CP_SYSTEM	Volume is CP system and cannot be used
8	RC_ERR	20	RS_VOLID_IN_USE	Volume label already CP_OWNED on this system or in this system's configuration
8	RC_ERR	24	RS_PARM_DISK_LINK_ERR	Error linking parm disk
8	RC_ERR	24	RS_UPDATE_WRITE_ERROR	Unable to write ASYNCH file
8	RC_ERR	24	RS_CONFLICTING_PARMS	Conflicting parameters
8	RC_ERR	28	RS_PARM_DISK_NOT_RW	Parm disk not RW
8	RC_ERR	28	RS_OUTPUT_NOT_VALID	Unexpected error obtaining information. See error data for details.
8	RC_ERR	32	RS_SYS_CONF_NOT_FOUND	System configuration not found on parm disk

Table 21. All Return Codes (Including Internal) (continued)

RC	RC Name	RS	RS Name	Description
8	RC_ERR	34	RS_SYS_CONF_BAD_DATA	System configuration has bad data
8	RC_ERR	36	RS_LENGTH_NOT_VALID	Specified length is not valid
8	RC_ERR	38	RS_CPDISK_MODE_NOT_ AVAIL	CP disk modes not available
8	RC_ERR	40	RS_PARM_DISK_FULL	Parm disk is full
8	RC_ERR	42	RS_PDISK_ACC_NOT_ ALLOWED	Parm disk access not allowed
8	RC_ERR	44	RS_PDISK_PW_NOT_ SUPPLIED	No link password for parm disk was provided
8	RC_ERR	46	RS_PDISK_PW_INCORRECT	Parm disk password is incorrect
8	RC_ERR	48	RS_PDISK_NOT_IN_SERVER _DIRECTORY	Parm disk is not in server's user directory
8	RC_ERR	50	RS_CPRELEASE_ERROR	Error with CPRELEASE of parm disk
8	RC_ERR	52	RS_CP_ACCESS_ERROR	Error in access of CPACCESS parm disk
8	RC_ERR	241	RS_VM_IPC_COMM_LOST	Internal communication error
8	RC_ERR	1821	RS_NONEXISTENT_DOMAIN	Relocation domain *domain_name* does not exist
8	RC_ERR	1822	RS_NO_FORCE _ARCHITECTURE	User *target_identifier* cannot be set to a new relocation domain *domain_name* without the FORCE ARCHITECTURE option
8	RC_ERR	1823	RS_IDENTITY_RELOCATION	A multiconfiguration virtual machine cannot be relocated
8	RC_ERR	2783	RS_INVALID_LANID	Invalid LAN ID
8	RC_ERR	2795	RS_INVALID_LAN_PARM	Invalid LAN parameter
8	RC_ERR	3000	RS_RELOCATION_ERRORS	VMRELOCATE MOVE error
8	RC_ERR	3002	RS_INVALID_PARAMETER	Invalid parameter name
8	RC_ERR	3003	RS_INVALID_OPERAND	Invalid parameter operand
8	RC_ERR	3004	RS_MISSING_PARAMETER	Required parameter missing
8	RC_ERR	3006	RS_SSI_UNSTABLE	SSI is not in a STABLE state
8	RC_ERR	3007	RS_SSI_CPOWNED_CONFLICT	The volume ID or slot is not available on all systems in the SSI
8	RC_ERR	3008	RS_NOT_SSI_MEMBER	System is not a member of an SSI cluster

Table 21. All Return Codes (Including Internal) (continued)

RC	RC Name	RS	RS Name	Description
8	RC_ERR	3010	RS_RELOCATION_MODIFY _ERROR	VMRELOCATE modify error
8	RC_ERR	3011	RS_NO_SLOT_AVAILABLE	No unique CP_OWNED slot available on system and in System Config
8	RC_ERR	3012	RS_VOLUME_NOT_FOUND	Volume does not exist
8	RC_ERR	3013	RS_VOLUME_OFFLINE	Volume is offline
8	RC_ERR	3014	RS_SHARE_UNSPPORTED	Volume does not support sharing
8	RC_ERR	3015	RS_FILE_SAVE_ERROR	File could not be saved
8	RC_ERR	3016	RS_SEGMENT_EMPTY	SMAPIOUT segment empty
8	RC_ERR	3017	RS_SEGMENT_DATA_INVALID	SMAPIOUT segment does not contain valid data
8	RC_ERR	3018	RS_SEGMENT_NOT_FOUND	SMAPIOUT segment not found and loaded
8	RC_ERR	3019	RS_CPU_DATA_UNAVAILABLE	SMAPIOUT CPU data not found
8	RC_ERR	3020	RS_TCPIP_STACK_NOT_VALID	Specified TCP/IP stack is not available
8	RC_ERR	3021	RS_NOT_IN_OBEYLIST	SMAPI worker server not in the obey list of specified TCP/IP stack
8	RC_ERR	3032	RS_INVALID_INPUT	Invalid input
8	RC_ERR	*nnnn*	RS_VMRELOCATE_ERROR	VMRELOCATE_Status returned an error. The RS *nnnn* represents the HCP*nnnn* message.
24	RCERR_SYNTAX	13	RS_LONG	Metadata entry name value length exceeds allowable length (1024)
24	RCERR_SYNTAX	19	RS_UNRECOG	Parameter value not recognized
24	RCERR_SYNTAX	pprr[1]	pprr[1]	Syntax error in function parameter
28	RCERR_FILE_NOT_FOUND	0	RS_NONE	Namelist file not found
36	RCERR_FILE_CANNOT_BE _UPDATED	0	RS_NONE	Namelist file cannot be updated
100	RCERR_AUTH	0	RS_NONE	Request is authorized
100	RCERR_AUTH	4	RS_DEFERRED_SERVER	Authorization deferred to directory manager
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager

Table 21. All Return Codes (Including Internal) (continued)

RC	RC Name	RS	RS Name	Description
100	RCERR_AUTH	12	RS_AUTHERR_DM	Request not authorized by directory manager
100	RCERR_AUTH	16	RS_AUTHERR_SERVER	Request not authorized by server
100	RCERR_AUTH	20	RS_TARGET_IMG_NOT _AUTHORIZED	Target image not authorized for function
104	RCERR_NO_AUTHFILE	0	RS_NONE	Authorization file not found
106	RCERR_AUTHFILE_RO	0	RS_NONE	Authorization file cannot be updated
108	RCERR_EXISTS	0	RS_NONE	Authorization file entry already exists
112	RCERR_NO_ENTRY	0	RS_NONE	Authorization file entry does not exist
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
188	RCERR_ESM	psrc[2]	psrc[2]	Internal server error; ESM failure
192	RCERR_PW_CHECK	psrc[2]	psrc[2]	Internal server error; cannot authenticate user/password
200	RCERR_IMAGEOP	0	RS_NONE	Image operation error
200	RCERR_IMAGEOP	4	RS_NOT_FOUND	Image not found
200	RCERR_IMAGEOP	8	RS_ALREADY_ACTIVE	Image already active
200	RCERR_IMAGEOP	12	RS_NOT_ACTIVE	Image not active
200	RCERR_IMAGEOP	16	RS_BEING_DEACT	Image being deactivated
200	RCERR_IMAGEOP	24	RS_LIST_NOT_FOUND	List not found
200	RCERR_IMAGEOP	28	RS_NOT_ALL	Some images in list not activated
200	RCERR_IMAGEOP	32	RS_SOME_NOT_DEACT	Some images in list not deactivated
200	RCERR_IMAGEOP	36	RS_SOME_NOT_RECYC	Some images in list not recycled
200	RCERR_IMAGEOP	36	RS_TIME_NOT_VALID	Specified time results in interval greater than max allowed
204	RCERR_IMAGEDEVU	0	RS_NONE	Image device usage error
204	RCERR_IMAGEDEVU	2	RS_INVALID_DEVICE	Input image device number not valid
204	RCERR_IMAGEDEVU	4	RS_EXISTS	Image device already exists
204	RCERR_IMAGEDEVU	8	RS_NOT_EXIST	Image device does not exist
204	RCERR_IMAGEDEVU	12	RS_BUSY	Image device is busy

Table 21. All Return Codes (Including Internal) (continued)

RC	RC Name	RS	RS Name	Description
204	RCERR_IMAGEDEVU	16	RS_NOT_AVAILABLE	Image device is not available
204	RCERR_IMAGEDEVU	20	RS_IS_CONNECTED	Image device already connected
204	RCERR_IMAGEDEVU	24	RS_TAPE_NOT_ASSIGNED	Image device is not a tape drive, or cannot be assigned/reset
204	RCERR_IMAGEDEVU	28	RS_DEV_NOT_SHARED	Image device is not a shared DASD
204	RCERR_IMAGEDEVU	28	RS_DEV_INCOMPATIBLE	Image device already defined as type other than network adapter
204	RCERR_IMAGEDEVU	32	RS_DEV_NOT_RESERVED	Image device is not a reserved DASD
204	RCERR_IMAGEDEVU	36	RS_DEV_IO_ERROR	I/O error on image device
204	RCERR_IMAGEDEVU	40	RS_NWDEV_NOT_DETACHED	Virtual Network Adapter not deleted
204	RCERR_IMAGEDEVU	44	RS_DASD_IN_USE	DASD volume cannot be deleted
204	RCERR_IMAGEDEVU	48	RS_IS_DISCONNECTED	Virtual network adapter is already disconnected
208	RCERR_IMAGEDISKU	0	RS_NONE	Image disk usage error
208	RCERR_IMAGEDISKU	4	RS_IN_USE	Image disk already in use
208	RCERR_IMAGEDISKU	8	RS_NOT_IN_USE	Image disk not in use
208	RCERR_IMAGEDISKU	12	RS_NOT_AVAILABLE	Image disk not available
208	RCERR_IMAGEDISKU	16	RS_CANNOT_SHARE	Image disk cannot be shared as requested
208	RCERR_IMAGEDISKU	20	RS_SHARE_DIFF_MODE	Image disk shared in different mode
208	RCERR_IMAGEDISKU	28	RS_PW_NEEDED	Image disk does not have required password
208	RCERR_IMAGEDISKU	28	RS_DEV_INCOMPATIBLE	Device is not a disk
208	RCERR_IMAGEDISKU	32	RS_BAD_PW	Incorrect password specified for image disk
208	RCERR_IMAGEDISKU	36	RS_NOT_EXIST	Image disk does not exist
208	RCERR_IMAGEDISKU	1157	RS_DEVNO_REQUIRES_FREE_DISK	MDISK DEVNO parameter requires the device to be a free volume
212	RCERR_IMAGECONN	0	RS_NONE	Active image connectivity error
212	RCERR_IMAGECONN	4	RS_NO_PARTNER	Partner image not found

Table 21. All Return Codes (Including Internal) (continued)

RC	RC Name	RS	RS Name	Description
212	RCERR_IMAGECONN	8	RS_AUTHERR_CONNECT	Image not authorized to connect
212	RCERR_IMAGECONN	8	RS_ADAPTER_NOT_EXIST	Adapter does not exist
212	RCERR_IMAGECONN	12	RS_LAN_NOT_EXIST	LAN does not exist
212	RCERR_IMAGECONN	16	RS_NOT_EXIST	LAN owner LAN name does not exist
212	RCERR_IMAGECONN	20	RS_OWNER_NOT_ACTIVE	Requested LAN owner not active
212	RCERR_IMAGECONN	24	RS_LAN_NAME_EXISTS	LAN name already exists with different attributes
212	RCERR_IMAGECONN	28	RS_DEV_INCOMPATIBLE	Image device not correct type for requested connection
212	RCERR_IMAGECONN	32	RS_NOT_CONNECTED	Image device not connected to LAN
212	RCERR_IMAGECONN	36	RS_VSWITCH_EXISTS	Virtual switch already exists
212	RCERR_IMAGECONN	40	RS_VSWITCH_NOT_EXISTS	Virtual switch does not exist
212	RCERR_IMAGECONN	44	RS_ALREADY_AUTH	Image already authorized
212	RCERR_IMAGECONN	48	RS_VLAN_NOT_FOUND	VLAN does not exist
212	RCERR_IMAGECONN	52	RS_MAX_CONN	Maximum number of connections reached
212	RCERR_IMAGECONN	96	RS_UNKNOWN	Unknown reason
216	RCERR_IMAGECPU	2	RS_INVALID_DEVICE	Input virtual CPU value out of range
216	RCERR_IMAGECPU	4	RS_NOT_FOUND	Virtual CPU not found
216	RCERR_IMAGECPU	12	RS_NOT_ACTIVE	Image not active
216	RCERR_IMAGECPU	24	RS_VCPU_ALREADY_EXISTS	Virtual CPU already exists
216	RCERR_IMAGECPU	28	RS_VCPU_OUT_OF_RANGE	Virtual CPU address beyond allowable range defined in directory
216	RCERR_IMAGECPU	40	RS_TYPE_NOT_SUPPORTED	Processor type not supported on your system
300	RCERR_VOLUME	0	RS_NONE	Image volume operation successful
300	RCERR_VOLUME	8	RS_DEV_NOT_FOUND	Device not found
300	RCERR_VOLUME	10	RS_DEV_NOT_AVAIL_TO _ATTACH	Device not available for attachment
300	RCERR_VOLUME	12	RS_DEV_NOT_VOLUME	Device not a volume
300	RCERR_VOLUME	14	RS_FREE_MODE_NOT_AVAIL	Free modes not available
300	RCERR_VOLUME	16	RS_DEV_NOT_ONLINE	Device vary online failed

Table 21. All Return Codes (Including Internal) (continued)

RC	RC Name	RS	RS Name	Description
300	RCERR_VOLUME	18	RS_VOLID_NOT_FOUND	Volume label not found in system configuration
300	RCERR_VOLUME	20	RS_VOLID_IN_USE	Volume label already in system configuration
300	RCERR_VOLUME	22	RS_PDISKS_SAME	Parm disks 1 and 2 are same
300	RCERR_VOLUME	24	RS_PARM_DISK_LINK_ERROR	Error linking parm disk (1 or 2)
300	RCERR_VOLUME	28	RS_PARM_DISK_NOT_RW	Parm disk (1 or 2) not RW
300	RCERR_VOLUME	32	RS_SYS_CONF_NOT_FOUND	System configuration not found on parm disk 1
300	RCERR_VOLUME	34	RS_SYS_CONF_BAD_DATA	System configuration has bad data
300	RCERR_VOLUME	36	RS_SYS_CONF_SYNTX_ERR	Syntax errors updating system configuration file
300	RCERR_VOLUME	38	RS_CPDISK_MODE_NOT _AVAIL	CP disk modes not available
300	RCERR_VOLUME	40	RS_PARM_DISK_FULL	Parm disk (1 or 2) is full
300	RCERR_VOLUME	42	RS_PDISK_ACC_NOT _ALLOWED	Parm disk (1 or 2) access not allowed
300	RCERR_VOLUME	44	RS_PDISK_PW_NOT _SUPPLIED	Parm disk (1 or 2) PW not supplied
300	RCERR_VOLUME	46	RS_PDISK_PW_INCORRECT	Parm disk (1 or 2) PW is incorrect
300	RCERR_VOLUME	48	RS_PDISK_NOT_IN_SERVER _DIRECTORY	Parm disk (1 or 2) is not in server's user directory
300	RCERR_VOLUME	50	RS_CP_RELEASE_ERROR	Error in release of CPRELEASE parm disk (1 or 2)
300	RCERR_VOLUME	52	RS_CP_ACCESS_ERROR	Error in access of CPACCESS parm disk (1 or 2)
396	RCERR_INTERNAL	0	RS_NONE	Internal system error
396	RCERR_INTERNAL	20	RS_REXX_SYNTAX	REXX syntax error in a SMAPI executable
396	RCERR_INTERNAL	nnnn	psrc[2]	Internal system error - product-specific return code (See "Internal Return Codes (RC = 396, 592, or 596)" on page 834)
400	RCERR_IMAGEDEF	0	RS_NONE	Image or profile definition error
400	RCERR_IMAGEDEF	4	RS_NOT_FOUND	Image or profile definition not found
400	RCERR_IMAGEDEF	8	RS_NAME_EXISTS	Image or profile name already defined

Table 21. All Return Codes (Including Internal) (continued)

RC	RC Name	RS	RS Name	Description
400	RCERR_IMAGEDEF	12	RS_LOCKED	Image or profile definition is locked
400	RCERR_IMAGEDEF	16	RS_CANNOT_DELETE	Image or profile definition cannot be deleted
400	RCERR_IMAGEDEF	20	RS_NOT_DEFINED	Image prototype is not defined
400	RCERR_IMAGEDEF	24	RS_NOT_LOCKED	Image or profile definition is not locked
400	RCERR_IMAGEDEF	40	RS_MULTIPLE	Multiple user statements
404	RCERR_IMAGEDEVD	0	RS_NONE	Image device definition error
404	RCERR_IMAGEDEVD	4	RS_EXISTS	Image device already defined
404	RCERR_IMAGEDEVD	8	RS_NOT_DEFINED	Image device not defined
404	RCERR_IMAGEDEVD	12	RS_LOCKED	Image device is locked
404	RCERR_IMAGEDEVD	24	RS_TYPE_NOT_SAME	Image device type not same as source
404	RCERR_IMAGEDEVD	24	RS_NOT_LOCKED	Image device is not locked
404	RCERR_IMAGEDEVD	28	RS_SIZE_NOT_SAME	Image device size not same as source
408	RCERR_IMAGEDISKD	0	RS_NONE	Image disk definition error
408	RCERR_IMAGEDISKD	4	RS_EXISTS	Image disk already defined
408	RCERR_IMAGEDISKD	8	RS_NOT_DEFINED	Image disk not defined
408	RCERR_IMAGEDISKD	12	RS_LOCKED	Image device is locked
408	RCERR_IMAGEDISKD	16	RS_NO_SHARING	Image disk sharing not allowed by target image definition
408	RCERR_IMAGEDISKD	24	RS_NO_SPACE	Requested image disk space not available
408	RCERR_IMAGEDISKD	28	RS_PW_NEEDED	Image disk does not have required password
408	RCERR_IMAGEDISKD	32	RS_BAD_PW	Incorrect password specified for image disk
412	RCERR_IMAGECONND	0	RS_NONE	Image connectivity definition error
412	RCERR_IMAGECONND	4	RS_NO_PARTNER	Partner image not found
412	RCERR_IMAGECONND	16	RS_NO_MATCH	Parameters do not match existing directory statement
412	RCERR_IMAGECONND	28	RS_DEV_INCOMPATIBLE	Image device not correct type for requested connection
416	RCERR_PROTODEF	0	RS_NONE	Prototype definition error

Table 21. All Return Codes (Including Internal) (continued)

RC	RC Name	RS	RS Name	Description
416	RCERR_PROTODEF	4	RS_NOT_FOUND	Prototype definition not found
416	RCERR_PROTODEF	8	RS_NAME_EXISTS	Prototype already exists
420	RC_DASD_DM	4	RS_IVS_NAME_USED	Group, region, or volume name is already defined
420	RC_DASD_DM	8	RS_IVS_NAME_NOT_USED	Group, region, or volume name is not defined
420	RC_DASD_DM	12	RS_IVS_NAME_NOT _INCLUDED	Region name is not included in the group
420	RC_DASD_DM	36	RS_IVS_NAME_NOT_DASD	The requested volume is offline or is not a DASD device
424	RCERR_SEGMENT_DM	4	RS_SEG_NAME_DUPLICATE	Namesave statement already exists
424	RCERR_SEGMENT_DM	8	RS_SEG_NAME_NOT_FOUND	Segment name not found
428	RCERR_NOTIFY	4	RS_NOTIFY_DUP	Duplicate subscription
428	RCERR_NOTIFY	8	RS_NOTIFY_NOT_FOUND	No matching entries
432	RCERR_TAG	4	RS_DUP_NAME	Tag name is already defined
432	RCERR_TAG	8	RS_NOT_DEFINED	Tag name is not defined
432	RCERR_TAG	12	RS_DUP_ORDINAL	Tag ordinal is already defined
432	RCERR_TAG	16	RS_CANNOT_REVOKE	Tag is in use in one or more directory entries, can not be revoked
432	RCERR_TAG	20	RS_NOT_AUTHORIZED	Use not allowed by exit routine
436	RCERR_PROFILED	4	RS_NOT_FOUND	Profile included not found
436	RCERR_PROFILED	40	RS_MULTIPLE	Multiple profiles included
444	RCERR_POLICY_PW	0	RS_NONE	Password policy error
444	RCERR_POLICY_PW	4	RS_LONG	Password too long
444	RCERR_POLICY_PW	8	RS_SHORT	Password too short
444	RCERR_POLICY_PW	12	RS_CONTENT	Password content does not match policy
448	RCERR_POLICY_ACCT	0	RS_NONE	Account policy error
448	RCERR_POLICY_ACCT	4	RS_LONG	Account number too long
448	RCERR_POLICY_ACCT	8	RS_SHORT	Account number too short
448	RCERR_POLICY_ACCT	12	RS_CONTENT	Account number content does not match policy
452	RCERR_TASK	4	RS_NOT_FOUND	Task not found
456	RCERR_SCSI	4	RS_LOADDEV_NOT_FOUND	LOADDEV statement not found

Table 21. All Return Codes (Including Internal) (continued)

RC	RC Name	RS	RS Name	Description
460	RC_IPL_DM	4	RS_IPL_NOT_FOUND	Image does not have an IPL statement
500	RCERR_DM	0	RS_NONE	Directory manager request could not be completed
500	RCERR_DM	4	RS_NO_UPDATES	Directory manager is not accepting updates
500	RCERR_DM	8	RS_NOT_AVAILABLE	Directory manager is not available
500	RCERR_DM	12	RS_DISABLED	Directory manager has been disabled
500	RCERR_DM	16	RS_INTERRUPTED	Directory manager was interrupted
500	RCERR_DM	20	RS_PW_FORMAT_NOT _SUPPORTED	Password format not supported
504	RCERR_LIST_DM	nnnn	psrc[2]	Target ID not added
520	RCERR_CPU_DM	24	RS_ONLY1_BASE_ALLOWED	Only one base CPU may be defined
520	RCERR_CPU_DM	28	RS_CPU_OUT_OF_RANGE	Input virtual CPU value out of range
520	RCERR_CPU_DM	30	RS_CPU_NOT_FOUND	CPU not found
520	RCERR_CPU_DM	32	RS_MAX_EXCEEDED	Maximum allowable number of virtual CPUs is exceeded
520	RCERR_CPU_DM	45	RS_CRYPTO_NOT_INSTALLED	The Cryptographic Coprocessor Facility (CCF) is not installed on this system
520	RCERR_UTF8	2826	RS_INVALID_UTF_DATA	SCPDATA contains invalid UTF-8 data
592	RCERR_ASYNC_DM	0	RS_NONE	Asynchronous operation started
592	RCERR_ASYNC_DM	nnnn	opid[3]	Asynchronous operation started - product-specific asynchronous operation ID (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
596	RCERR_INTERNAL_DM	nnnn	psrc[2]	Internal directory manager error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
600	RCERR_SHSTOR	8	RS_BAD_RANGE	Bad page range
600	RCERR_SHSTOR	12	RS_NOT_LOGGED_ON	User not logged on
600	RCERR_SHSTOR	16	RS_NOSAVE	Could not save segment

Table 21. All Return Codes (Including Internal) (continued)

RC	RC Name	RS	RS Name	Description
600	RCERR_SHSTOR	20	RS_NOT_AUTHORIZED	Not authorized to issue internal system command or is not authorized for RSTD segment
600	RCERR_SHSTOR	24	RS_CONFLICTING_PARMS	Conflicting parameters
600	RCERR_SHSTOR	28	RS_SEGMENT_NOT_FOUND	Segment not found or does not exist
600	RCERR_SHSTOR	299	RS_CLASS_S_ALREADY _DEFINED	Class S (skeleton) segment file already exists
620	RCERR_VIRTUALNETWORKD	14	RS_FREE_MODE_NOT_AVAIL	Free modes not available
620	RCERR_VIRTUALNETWORKD	22	RS_PARM_DISKS_SAME	System config parm disks 1 and 2 are same
620	RCERR_VIRTUALNETWORKD	24	RS_PARM_DISK_LINK_ERROR	Error linking parm disk (1 or 2)
620	RCERR_VIRTUALNETWORKD	28	RS_PARM_DISK_NOT_RW	Parm disk (1 or 2) not RW
620	RCERR_VIRTUALNETWORKD	32	RS_SYS_CONF_NOT_FOUND	System config not found on parm disk 1
620	RCERR_VIRTUALNETWORKD	34	RS_SYS_CONF_BAD_DATA	System config has bad data
620	RCERR_VIRTUALNETWORKD	36	RS_SYS_CONF_SYNTX_ERR	Syntax errors updating system config
620	RCERR_VIRTUALNETWORKD	38	RS_CPDISK_MODE_NOT _AVAIL	CP disk modes not available
620	RCERR_VIRTUALNETWORKD	40	RS_PARM_DISK_FULL	Parm disk (1 or 2) is full
620	RCERR_VIRTUALNETWORKD	42	RS_PARM_DISK_ACC_NOT _ALLOWED	Parm disk (1 or 2) access not allowed
620	RCERR_VIRTUALNETWORKD	44	RS_PDISK_PW_NOT _SUPPLIED	Parm disk (1 or 2) PW not supplied
620	RCERR_VIRTUALNETWORKD	46	RS_PDISK_PW_INCORRECT	Parm disk (1 or 2) PW is incorrect
620	RCERR_VIRTUALNETWORKD	48	RS_PDISK_NOT_IN_SERVER _DIRECTORY	Parm disk (1 or 2) is not in server's directory
620	RCERR_VIRTUALNETWORKD	50	RS_CP_RELEASE_ERROR	Error in release of CPRELEASE parm disk (1 or 2)
620	RCERR_VIRTUALNETWORKD	52	RS_CPACCESS_ERROR	Error in access of CPACCESS parm disk (1 or 2)
620	RCERR_VIRTUALNETWORKD	54	RS_DEF_VSWITCH_EXISTS	DEFINE VSWITCH statement already exists in system config
620	RCERR_VIRTUALNETWORKD	58	RS_REVOKE_FAILED	MODIFY VSWITCH statement to userid not found in system config
620	RCERR_VIRTUALNETWORKD	60	RS_DEF_VSWITCH_NOT _EXIST	DEFINE VSWITCH statement does not exist in system config

Table 21. All Return Codes (Including Internal) (continued)

RC	RC Name	RS	RS Name	Description
620	RCERR_VIRTUALNETWORKD	62	RS_VSWITCH_CONFLICT	DEFINE operands conflict, cannot be updated in the system config
620	RCERR_VIRTUALNETWORKD	64	RS_DEF_MOD_MULTI_FOUND	Multiple DEFINE or MODIFY statements found in system config
800	RCERR_VMRM	8	RS_NO_MEASUREMENT _DATA	No measurement data exists
800	RCERR_VMRM	12	RS_UPDATE_SYNTAX_ERROR	Error in update buffer or processing syntax check
800	RCERR_VMRM	16	RS_CANNOT_ACCESS_DATA	Not authorized to access file
800	RCERR_VMRM	24	UPDATE_WRITE_ERROR	Error writing file(s) to directory
800	RCERR_VMRM	28	RS_FILE_NOT_FOUND	Specified configuration file not found
800	RCERR_VMRM	32	RS_UPDATE_PROCESS _ERROR	Internal error processing updates
900	RCERR_SERVER	4	RS_NOT_FOUND	Custom exec not found
900	RCERR_SERVER	8	RS_WORKER_NOT_FOUND	Worker server was not found
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
900	RCERR_SERVER	16	RS_PTS_ENTRY_NOT_VALID	Internal server error - DMSSIPTS entry for function is invalid
900	RCERR_SERVER	20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
900	RCERR_SERVER	24	RS_SFS_ERROR	Error accessing SFS directory
900	RCERR_SERVER	28	RS_OUTPUT_NOT_VALID	Internal server error - error with format of function output
900	RCERR_SERVER	32	RS_REQRESP_INVALID	Internal server error - response from worker server was not valid
900	RCERR_SERVER	36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range
900	RCERR_SERVER	40	RS_SOCKET	Internal server socket error
900	RCERR_SERVER	68	RS_DATABASE	Unable to access LOHCOST server
900	RCERR_SERVER	99	RS_RETRY	A system change occurred during the API call – reissue the API call to obtain the data.

[1] *pprr* = parameter in error, and nature of error. See "Syntax Error Reason Codes (RC = 24)" on page 833.
[2] *psrc* = product-specific return code.
[3] *opid* = operation ID.

Syntax Error Reason Codes (RC = 24)

Return code 24 signifies a syntax error in the supplied parameters. The reason code identifies the parameter in error and the nature of the error.

Important

Parameter length specifications must exactly match the actual length of the data provided for each parameter. Length errors for a specific parameter (such as rr = 13 below) could result from an incorrect length specification of a previous parameter.

Note, also, that even parameters that are ignored (or can be left unspecified) must still be syntactically correct. If an entry does not conform to the character set specified for that parameter, an error is generated.

The reason code is a decimal integer value in the format "pprr", where:

- pp = reason code / 100
- rr = remainder (reason code / 100)

pp
This identifies the *n*th parameter in the input argument structure that is in error (only significant digits are included in the returned value).

rr
This identifies the nature of the error, usually the number of characters in the valid character set, as follows:

01
First character of listname is a colon ":"

10
Characters not "0123456789"

11
Unsupported function

13
Length is greater than maximum or exceeds total length

14
Length is less than minimum

15
Numeric value less than minimum or null value encountered

16
Characters not "0123456789ABCDEF"

17
Characters not "0123456789ABCDEF-"

18
Numeric value greater than maximum

19
Unrecognized value

23
Conflicting parameter specified

24
Unspecified required parameter

25
Extraneous parameter specified

26
Characters not "ABCDEFGHIJKLMNOPQRSTUVWXYZ"

36
Characters not "ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789"

37
Characters not "ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789-"

42
Characters not "ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789@#$+-:"

43
Characters not "ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789@#$+-:_"

44
Characters not "ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789@#$+-:_="

45
Invalid SFS syntax

88
Unexpected end of data

99
Non-breaking characters: non-blank, non-null, non-delete, non-line-end, non-carriage return, non-line-feed

Internal Return Codes (RC = 396, 592, or 596)

Table 22. Internal Return Codes (RC = 396, 592, or 596)

RC	RC Name	RS	RS Name	Description
396	RCERR_INTERNAL	0	RS_NONE	Internal system error
396	RCERR_INTERNAL	20	RS_REXX_SYNTAX	REXX syntax error in a SMAPI executable
396	RCERR_INTERNAL	nnnn	psrc	Internal system error - product-specific return code
592	RCERR_ASYNC_DM	0	RS_NONE	Asynchronous operation started
592	RCERR_ASYNC_DM	nnnn	opid	Asynchronous operation started - product-specific asynchronous operation ID
596	RCERR_INTERNAL_DM	nnnn	psrc	Internal directory manager error - product-specific return code

Return Code 396

If an error occurs in a function exec while processing a function request for which no other specified return code is applicable, the reason code will be set to the return code of the failing routine and the return code will be 396.

Table 23 on page 834 shows which CP commands are used by the Systems Management APIs. More information on CP command return codes can be found in *z/VM: CP Commands and Utilities Reference*.

Table 23. CP Commands Used by Systems Management APIs

API	CP Command
Delete_ABEND_Dump	LINK
Image_Activate	XAUTOLOG

Table 23. CP Commands Used by Systems Management APIs (continued)

API	CP Command
Image_Active_Configuration_Query	FOR QUERY SHARE QUERY USER QUERY USERID QUERY VIRTUAL
Image_CPU_Define	DEFINE CPU QUERY USER QUERY USERID
Image_CPU_Delete	DETACH CPU QUERY USER QUERY USERID
Image_CPU_Query	QUERY USER QUERY USERID QUERY VIRTUAL CPUS
Image_Deactivate	FORCE QUERY NAMES QUERY SIGNAL SHUTDOWN SIGNAL SHUTDOWN
Image_Device_Dedicate	ATTACH
Image_Device_Reset	FOR QUERY USERID RESET
Image_Device_Undedicate	DETACH FOR QUERY USERID
Image_Disk_Copy	FOR LINK QUERY MDISK QUERY USERID
Image_Disk_Create	FOR LINK QUERY MDISK QUERY USERID
Image_Disk_Delete	DETACH FOR QUERY MDISK QUERY USERID

Table 23. CP Commands Used by Systems Management APIs (continued)

API	CP Command
Image_Disk_Query	FOR QUERY DASD DETAILS QUERY USERID QUERY VIRTUAL DASD
Image_Disk_Share	FOR LINK QUERY MDISK QUERY PASSWORD LINK QUERY USERID SET PASSWORD LINK
Image_Disk_Unshare	DETACH FOR QUERY MDISK QUERY USERID
Image_Lock_Query_DM	QUERY QUERY USERS
Image_MDISK_Link_Query	QUERY LINKS QUERY SSI STATUS QUERY USERS
Image_Pause	QUERY USER SLEEP BEGIN FOR
Image_Query_Activate_Time	DISPLAY LOCATE VMDBK QUERY USER
Image_Recycle	FORCE QUERY USERID XAUTOLOG
Image_Status_Query	QUERY NAMES
Image_Volume_Add	ATTACH CPACCESS CPFMTXA (utility) CPRELEASE DETACH LINK QUERY CPDISK QUERY MDISK QUERY VIRTUAL DASD SET SMSG

Table 23. CP Commands Used by Systems Management APIs (continued)

API	CP Command
Image_Volume_Delete	ATTACH CPACCESS CPRELEASE DETACH LINK QUERY CPDISK QUERY MDISK QUERY VIRTUAL DASD
Image_Volume_Share	SET SHARED
Page_or_Spool_Volume_Add	ATTACH CPACCESS CPFMTXA (utility) CPRELEASE CPSYNTAX (utility) DETACH QUERY DASD ACTIVE QUERY DASD DETAILS QUERY SSI QUERY VIRTUAL SET EMSG VARY ON
Process_ABEND_Dump	CHANGE READER LINK ORDER READER PURGE READER QUERY READER
Query_ABEND_Dump	LINK QUERY READER
Shared_Memory_Create	DEFSEG FOR PURGE NSS QUERY NSS QUERY USER QUERY USERID SAVESEG
Shared_Memory_Delete	FOR PURGE NSS QUERY NSS QUERY USERID
Shared_Memory_Query	QUERY NSS

Table 23. CP Commands Used by Systems Management APIs (continued)

API	CP Command
Shared_Memory_Replace	DEFSEG FOR PURGE NSS QUERY NSS QUERY USER QUERY USERID SAVESEG
SMAPI_Status_Capture	QUERY ACCESSED QUERY CPLEVEL QUERY NAMES QUERY OSA QUERY RDR QUERY TIME QUERY USERS QUERY VIRTUAL QUERY VMLAN QUERY VSWITCH TRANSFER
SSI_Query	QUERY SSI
System_Disk_Accessibility	ATTACH DETACH
System_Disk_Add	SET RDEV VARY ON
System_Disk_Query	QUERY DASD
System_EQID_Query	QUERY EQID
System_FCP_EQID_Set	VARY QUERY rdevice SET REDEVICE Special Devices
System_FCP_Query	ATTACH DETACH QUERY FCP
System_Information_Query	QUERY CAPABILITY QUERY CPLANGUAGE QUERY CPLEVEL QUERY CPUID QUERY STORAGE QUERY TIME QUERY TIMEZONE
System_Page_Utilization_Query	QUERY ALLOC PAGE

Table 23. CP Commands Used by Systems Management APIs (continued)

API	CP Command
System_Performance_Information_Query	INDICATE MONITOR QUERY FRAMES QUERY LPARS QUERY MONITOR
System_Processor_Query	QUERY PROCESSORS EXPANDED
System_RDR_File_Manage	ORDER PURGE TRANSFER
System_RDR_File_Query	QUERY RDR
System_SCSI_Disk_Add	SET EDEV
System_SCSI_Disk_Delete	DELETE EDEV QUERY EDEV SET EDEV VARY OFF
System_SCSI_Disk_Query	QUERY EDEV
System_Service_Query	QUERY SSI
System_Shutdown	SHUTDOWN
System_Spool_Utilization_Query	QUERY ALLOC SPOOL
System_WWPN_Query	QUERY FCP WWPN
Virtual_Channel_Connection_Create	COUPLE DEFINE CTCA FOR LOCATE VDEV
Virtual_Channel_Connection_Delete	DETACH FOR LOCATE VDEV
Virtual_Network_Adapter_Connect_LAN	COUPLE FOR QUERY LAN QUERY VIRTUAL NIC
Virtual_Network_Adapter_Connect_Vswitch	COUPLE FOR QUERY VIRTUAL NIC

Table 23. CP Commands Used by Systems Management APIs (continued)

API	CP Command
Virtual_Network_Adapter_Connect_Vswitch _Extended	COUPLE FOR QUERY VIRTUAL NIC
Virtual_Network_Adapter_Create	DEFINE NIC FOR QUERY VIRTUAL NIC
Virtual_Network_Adapter_Create_Extended	DEFINE NIC FOR QUERY VIRTUAL NIC
Virtual_Network_Adapter_Delete	DETACH NIC FOR QUERY VIRTUAL NIC
Virtual_Network_Adapter_Disconnect	FOR QUERY VIRTUAL NIC UNCOUPLE
Virtual_Network_LAN_Access	SET LAN
Virtual_Network_LAN_Create	DEFINE LAN
Virtual_Network_LAN_Delete	DETACH LAN
Virtual_Network_OSA_Query	QUERY OSA
Virtual_Network_Vswitch_Create	CPACCESS CPRELEASE DEFINE VSWITCH DETACH LINK QUERY CPDISK QUERY MDISK SET VSWITCH
Virtual_Network_Vswitch_Create_Extended	DEFINE VSWITCH SET VSWITCH
Virtual_Network_Vswitch_Delete	CPACCESS CPRELEASE DETACH DETACH VSWITCH LINK QUERY CPDISK QUERY MDISK

Table 23. CP Commands Used by Systems Management APIs (continued)	
API	**CP Command**
Virtual_Network_Vswitch_Delete_Extended	DETACH VSWITCH QUERY VSWITCH SET PORT GROUP
Virtual_Network_Vswitch_Set	CPACCESS CPRELEASE DETACH LINK QUERY CPDISK QUERY MDISK SET VSWITCH
Virtual_Network_Vswitch_Set_Extended	SET PORT GROUP SET VSWITCH QUERY VSWITCH
VMRELOCATE	VMRELOCATE
VMRELOCATE_Image_Attributes	SET VMRELOCATE
VMRELOCATE_Modify	VMRELOCATE MODIFY
VMRELOCATE_Status	QUERY VMRELOCATE

RC=396 RS=40

If RC 396 is returned with RS 40 from one of these image device functions, the following may be helpful in determining the cause:

Image_Disk_Create
Image_Disk_Copy
Image_Disk_Share

- **For any of these functions:**
  - The image disk to be accessed dynamically may be defined as a T-DISK, or may be defined on a DASD volume that is not currently attached to the system.
- **For the Image_Disk_Create function:** If the *Image_Disk_Formatting* parameter was set to any value other than NONE, or if a value for *Image_Disk_Label* was specified, then the format operation may not have completed yet.
- **For the Image_Disk_Copy function:** The copy may not be completed yet.
- **For the Image_Disk_Share function:**
  - One or more active images may already have an active link to the disk to be shared, and the modes may not be compatible with concurrent sharing.
  - The image disk may not have a password, which is required for sharing, or the actual password may be different from the password specified on the call to the Image_Disk_Share function.
  - The image disk to be shared may not be defined to the External Security Manager (ESM), or, if it is defined to the ESM, permission for the target image to access the requested image disk may be denied.

Return Code 592, 596

If an error occurs in a directory manager routine while processing a function request for which no other specified return code is applicable, the return code will be 596 (RCERR_INTERNAL_DM) and the reason code will be the return code passed back from the failing routine.

If you are running the IBM Directory Maintenance Facility (DirMaint) and the directory manager code is 1119, this indicates a CMS or CP command error. Check the console of the directory manager or the consoles of the SMAPI long call worker servers (VSMWORK2, VSMWORK3, etc).

If the directory manager routine begins an asynchronous operation and then returns control to the calling program before the operation completes, the return code will be set to 592 (RCERR_ASYNC_DM). If the routine has an *operation_id* output parameter (i.e., if it is a truly asynchronous operation), then the reason code will be set to 0 (RS_NONE), and the *operation_id* output parameter will contain the operation ID (as shown in “Internal Return Codes (RC = 396, 592, or 596)” on page 834). If the routine does not have an *operation_id* output parameter (i.e., if it is not technically an asynchronous operation, but is rather a directory manager operation that can potentially be delayed for other reasons), then the reason code will be set to an arbitrary operation ID value. If DirMaint is your directory manager, you can find more information in the "DirMaint Support for Systems Management APIs" appendix of the *z/VM: Directory Maintenance Facility Tailoring and Administration Guide*, and also in the *z/VM: Directory Maintenance Facility Messages*.

Appendix A. The Directory Manager Exit

For directory manager APIs, a special exit must be called to create and process the directory manager commands associated with the API and to return the API output. The directory manager exit is written in interpreted or compiled REXX. The REXX EXEC name for the exit is defined by the DM_exit parameter in the DMSSICNF COPY file, as described in "Configuring SMAPI" on page 30. The original default value is DMSSIXDM, the IBM-supplied exit. The following sections describe the input and output interface information needed to create a customized directory manager exit.

Note: Directory manager APIs are designed as "long call" APIs and are processed by a "long call" worker server as described in Chapter 3, "Defining the Servers," on page 21.

Directory Manager Exit Input Interface

The directory manager exit is called with the following REXX input arguments, as per those of the same name in the related API documentation in Chapter 6, "Socket Application Programming Interfaces," on page 55 (unless otherwise noted in this appendix):

total_parms

function_name

authenticated_userid

target_identifier

additional_input_parameter_1

...

additional_input_parameter_n

Each parameter is described below:

total_parms
(int4;range 3-(*n*+3)) The total number of parameters that follow this parameter on a given directory manager exit call, where *n* is the number of additional input parameters (see below).

function_name
(string,1-64,char43) The API function name.

authenticated_userid
One of the following:

- (string,1-8,char42) The userid under whose authority to perform the function (AF_INET requests).
- (string,0-8,char42) The userid under whose authority to perform the function (AF_IUCV requests).

Note that *authenticated_userid* is optional for AF_IUCV requests. See "Client Authentication" on page 36 for more information.

target_identifier
The userid for which the function will be performed.

additional_input_parameter_1* to *additional_input_parameter_n
The function-specific input arguments. Table 24 on page 844 indicates which function-specific arguments are supplied for each directory manager API. These input arguments are supplied as described in the related API documentation for the associated input parameters of the same name in Chapter 6, "Socket Application Programming Interfaces," on page 55. Any differences between the arguments supplied to the directory manager exit and the related API documentation in are indicated in the table.

Note:

1. Unlike standard APIs, string length parameters are *not* provided to the directory manager exit.

© Copyright IBM Corp. 2003, 2025

2. Int4 values are provided to the exit in character format unless otherwise noted.
3. Int1 enumeration types are provided as the string values specified in Table 24 on page 844 unless otherwise noted.
4. Arrays are provided to the exit as single input parameters, with no preceding length parameters.
5. Integers supplied in array input arguments are not provided as character data, but rather as binary data, and are indicated in Table 24 on page 844 as int1 or int4, not string.
6. Unspecified strings, int4, or int1 enumeration types are provided as an empty string (‘’) to the directory manager exit.
7. The input arguments are either provided in EBCDIC or as-provided by the Systems Management API client. Any exceptions are noted in Table 24 on page 844.

Table 24. Directory Manager Function-Specific Arguments

Function Name	Input Argument	Input Type*	Possible Input Values	Input Encoding
Asynchronous_Notification_Disable_DM	*entity_type*	string	‘DIRECTORY’	EBCDIC
	communication_type	string	‘TCP’ ‘UDP’	EBCDIC
	port_number	string		EBCDIC
	ip_address			EBCDIC
	encoding	string	‘’ ‘ASCII’ ‘EBCDIC’	EBCDIC
	subscriber_data			As provided
Asynchronous_Notification_Enable_DM	*entity_type*	string	‘DIRECTORY’	EBCDIC
	subscription_type	string	‘INCLUDE’ ‘EXCLUDE’	EBCDIC
	communication_type	string	‘TCP’ ‘UDP’	EBCDIC
	port_number	string		EBCDIC
	ip_address			EBCDIC
	encoding	string	‘’ ‘ASCII’ ‘EBCDIC’	EBCDIC
	subscriber_data			As provided
Asynchronous_Notification_Query_DM	*entity_type*	string	‘DIRECTORY’	EBCDIC
	communication_type	string	‘’ ‘TCP’ ‘UDP’	EBCDIC
	port_number	string		EBCDIC
	ip_address			EBCDIC
	encoding	string	‘’ ‘ASCII’ ‘EBCDIC’	EBCDIC
	subscriber_data			As provided

Table 24. Directory Manager Function-Specific Arguments (continued)

Function Name	Input Argument	Input Type*	Possible Input Values	Input Encoding
Directory_Manager_Local_Tag_Define_DM	*tag_name*			EBCDIC
	tag_ordinal	string		EBCDIC
	define_action	string	‘’ ‘CREATE’ ‘CHANGE’	EBCDIC
Directory_Manager_Local_Tag_Delete_DM	*tag_name*			EBCDIC
Directory_Manager_Local_Tag_Query_DM	*tag_name*			EBCDIC
Directory_Manager_Local_Tag_Set_DM	*tag_name*			EBCDIC
	tag_value			EBCDIC
Directory_Manager_Search_DM	*search_pattern*			EBCDIC
Directory_Manager_Task_Cancel_DM	*operation_id*	string		EBCDIC
Image_CPU_Define_DM	*cpu_address*			EBCDIC
	base_cpu	string	‘’ ‘BASE’	EBCDIC
	cpuid			EBCDIC
	dedicate_cpu	string	‘’ ‘NODEDICATE’ ‘DEDICATE’	EBCDIC
	crypto	string	‘’ ‘CRYPTO’	EBCDIC
Image_CPU_Delete_DM	*cpu_address*			EBCDIC
Image_CPU_Query_DM	*cpu_address*			EBCDIC
Image_CPU_Set_Maximum_DM	*max_cpu*	string		EBCDIC
Image_Create_DM	*prototype_name*			EBCDIC
	initial_password			EBCDIC
	initial_account_number			EBCDIC
	image_record_array *image_record_length* *image_record*	int4		EBCDIC
Image_Definition_Create_DM	*definition_create_keyword_parameter_list*			EBCDIC
Image_Definition_Delete_DM	*definition_delete_keyword_parameter_list*			EBCDIC
Image_Definition_Query_DM	*definition_query_keyword_parameter_list*			EBCDIC
Image_Definition_Update_DM	*definition_update_keyword_parameter_list*			EBCDIC
Image_Delete_DM	*data_security_erase*	string	‘’ ‘1’ ‘2’	EBCDIC

Table 24. Directory Manager Function-Specific Arguments (continued)

Function Name	Input Argument	Input Type*	Possible Input Values	Input Encoding
Image_Device_Dedicate_DM	*image_device_number*			EBCDIC
	real_device_number			EBCDIC
	readonly	string	‘’ ‘READONLY’	EBCDIC
Image_Device_Undedicate_DM	*image_device_number*			EBCDIC
Image_Disk_Copy_DM	*image_disk_number*			EBCDIC
	source_image_name			EBCDIC
	source_image_disk_number			EBCDIC
	image_disk_allocation_type			EBCDIC
	allocation_area_name_or_volser			EBCDIC
	image_disk_mode			EBCDIC
	read_password			EBCDIC
	write_password			EBCDIC
	multi_password			EBCDIC
Image_Disk_Create_DM	*image_disk_number*			EBCDIC
	image_disk_device_type			EBCDIC
	image_disk_allocation_type			EBCDIC
	allocation_area_name_or_volser			EBCDIC
	allocation_unit_size	string	‘CYLINDERS’ ‘BLK0512’ ‘BLK1024’ ‘BLK2048’ ‘BLK4096’	EBCDIC
	image_disk_size	string		EBCDIC
	image_disk_mode			EBCDIC
	image_disk_formatting	string	‘’ ‘NONE’ ‘CMS0512’ ‘CMS1024’ ‘CMS2048’ ‘CMS4096’ ‘CMS’	EBCDIC
	image_disk_label			EBCDIC
	read_password			EBCDIC
	write_password			EBCDIC
	multi_password			EBCDIC
Image_Disk_Delete_DM	*image_disk_number*			EBCDIC
	data_security_erase	string	‘’ ‘1’ ‘2’	EBCDIC

Table 24. Directory Manager Function-Specific Arguments (continued)

Function Name	Input Argument	Input Type*	Possible Input Values	Input Encoding
Image_Disk_Share_DM	*target_disk_number*			EBCDIC
	target			EBCDIC
	image_disk_number			EBCDIC
	read_write_mode			EBCDIC
	optional_password			EBCDIC
Image_Disk_Unshare_DM	*image_disk_number*			EBCDIC
	target			EBCDIC
	target_disk_number			EBCDIC
Image_IPL_Delete_DM	(No additional arguments)			
Image_IPL_Query_DM	(No additional arguments)			
Image_IPL_Set_DM	*saved_system*			EBCDIC
	load_parameter			EBCDIC
	parameter_string			EBCDIC
Image_Lock_DM	*device_address*			EBCDIC
Image_Lock_Query_DM	(No additional arguments)			
Image_Name_Query_DM	(No additional arguments)			
Image_Password_Set_DM	*image_password*			EBCDIC
Image_Query_DM	(No additional arguments)			
Image_Replace_DM	*image_record_array* *image_record_length* *image_record*	int4		EBCDIC
Image_SCSI_Characteristics_Define_DM	*boot_program*			EBCDIC
	BR_LBA			EBCDIC
	LUN			EBCDIC
	port_name			EBCDIC
	SCP_data_type	string	" 'DELETE' 'EBCDIC' 'HEX'	EBCDIC
	SCP_data			As provided
Image_SCSI_Characteristics_Query_DM	(No additional arguments)			
Image_Unlock_DM	*device_address*			EBCDIC

Table 24. Directory Manager Function-Specific Arguments (continued)

Function Name	Input Argument	Input Type*	Possible Input Values	Input Encoding
Image_Volume_Space_Define_DM	*function_type*	string	1-5	EBCDIC
	region_name			EBCDIC
	image_vol_id			EBCDIC
	start_cylinder	string		EBCDIC
	size	string		EBCDIC
	group_name			EBCDIC
	device_type	string	‘’ ‘3390’ ‘9336’ ‘3380’ ‘FB-512’	EBCDIC
Image_Volume_Space_Define_Extended_DM	`function_type=`*value*			EBCDIC
	`region_name=`*value*			EBCDIC
	`image_vol_id=`*value*			EBCDIC
	`start_cylinder=`*value*			EBCDIC
	`size=`*value*			EBCDIC
	`group_name=`*value*			EBCDIC
	`device_type=`*value*			EBCDIC
	`alloc_method=`*value*			EBCDIC
Image_Volume_Space_Query_DM	*query_type*	string	‘DEFINITION’ ‘FREE’ ‘USED’	EBCDIC
	entry_type	string	‘VOLUME’ ‘REGION’ ‘GROUP’	EBCDIC
	entry_names			EBCDIC
Image_Volume_Space_Query_Extended_DM	`query_type=`*value*			EBCDIC
	`entry_type=`*value*			EBCDIC
	`entry_names=`*value*			EBCDIC
Image_Volume_Space_Remove_DM	*function_type*	string	1-7	EBCDIC
	region_name			EBCDIC
	image_vol_id			EBCDIC
	group_name			EBCDIC
Profile_Create_DM	*profile_record_array* *profile_record_length* *profile_record*	int4		EBCDIC
Profile_Delete_DM	(No additional arguments)			
Profile_Lock_DM	(No additional arguments)			
Profile_Lock_Query_DM	(No additional arguments)			
Profile_Query_DM	(No additional arguments)			

Table 24. Directory Manager Function-Specific Arguments (continued)

Function Name	Input Argument	Input Type*	Possible Input Values	Input Encoding
Profile_Replace_DM	*profile_record_array* *profile_record_length* *profile_record*	int4		EBCDIC
Profile_Unlock_DM	(No additional arguments)			
Prototype_Create_DM	*prototype_record_array* *prototype_record_length* *prototype_record*	int4		EBCDIC
Prototype_Delete_DM	(No additional arguments)			
Prototype_Name_Query_DM	(No additional arguments)			
Prototype_Query_DM	(No additional arguments)			
Prototype_Replace_DM	*prototype_record_array* *prototype_record_length* *prototype_record*	int4		EBCDIC
Query_All_DM	`query_parameter_name_list=`*value*			EBCDIC
Query_Asynchronous_Operation_DM	*operation_id*	string		EBCDIC
Query_Directory_Manager_Level_DM	(No additional arguments)			
Shared_Memory_Access_Add_DM	*memory_segment_name*			EBCDIC
Shared_Memory_Access_Query_DM	*memory_segment_name*			EBCDIC
Shared_Memory_Access_Remove_DM	*memory_segment_name*			EBCDIC
Static_Image_Changes_Activate_DM	(No additional arguments)			
Static_Image_Changes_Deactivate_DM	(No additional arguments)			
Static_Image_Changes_Immediate_DM	(No additional arguments)			
Virtual_Channel_Connection_Create_DM	*image_device_number*			EBCDIC
	coupled_image_name			EBCDIC
Virtual_Channel_Connection_Delete_DM	*image_device_number*			EBCDIC
System_FCP_EQID_Set	*rdev*	string	char16	EBCDIC
	eqid	string	char36 plus -	EBCDIC
	persist	string	"NO" "YES"	EBCDIC
Virtual_Network_Adapter_Connect_LAN_DM	*image_device_number*			EBCDIC
	lan_name			EBCDIC
	lan_owner			EBCDIC
Virtual_Network_Adapter_Connect_Vswitch_DM	*image_device_number*			EBCDIC
	switch_name			EBCDIC
Virtual_Network_Adapter_Create_DM	*image_device_number*			EBCDIC
	adapter_type			As provided
	network_adapter_devices	string		EBCDIC
	channel_path_id			EBCDIC
	mac_id			EBCDIC

Table 24. Directory Manager Function-Specific Arguments (continued)

Function Name	Input Argument	Input Type*	Possible Input Values	Input Encoding
Virtual_Network_Adapter_Create_Extended_DM	`image_device_number=`*value*			EBCDIC
	`adapter_type=`*value*			EBCDIC
	`devices=`*value*			EBCDIC
	`channel_path_id=`*value*			EBCDIC
	`mac_id=`*value*			EBCDIC
Virtual_Network_Adapter_Delete_DM	*image_device_number*			EBCDIC
Virtual_Network_Adapter_Disconnect_DM	*image_device_number*			EBCDIC

* Input type is the same as in the specific API, except where noted in this table.

Directory Manager Exit Output Interface

After performing the function for a given directory manager API, the directory manager exit must return the following output arguments (concatenated as a single return string) to the Systems Management API server:

return_code

reason_code

additional_output_parameter_1

...

additional_output_parameter_n

Each parameter is described below:

return_code
(int4) The return code.

reason_code
(int4) The reason code.

additional_output_parameter_1* to *additional_output_parameter_n

The function-specific output arguments. These arguments are the output parameters documented after the *reason_code* parameter in the "Response 2 Output Parameters" section in the API-specific documentation in Chapter 6, "Socket Application Programming Interfaces," on page 55.

Note: The *output_length* and *request_id* output parameters documented in each "Response 2 Output Parameters" section in Chapter 6, "Socket Application Programming Interfaces," on page 55 must *not* be returned by the directory manager exit. These output parameters are added by the Systems Management API server.

Appendix B. Creating Custom APIs

This appendix shows how an installation can add APIs to the server to provide capabilities that are not provided by the IBM-supplied APIs. This is done through a three-step process, as follows:

1. Determine the API design (name, input and output parameters, and type).
2. Write a custom exec (in REXX) to perform the desired task.
3. Install the custom exec on the server.

A client program calls the locally-defined API in the same manner as an IBM-supplied API, with the appropriate input parameters. The server checks the authorization file and, if the client is authorized, a worker server calls the custom exec with certain input arguments. The custom exec performs the desired task and returns with a certain return value. The server then returns the appropriate output parameters to the client program.

Designing the Custom API

In this first step, you determine the API's name, input and output parameters, and type (short-call or long-call). Note that the API cannot have the same name as an IBM-supplied API. If such an API is created and installed, it will be ignored by the server.

The input and output parameters of the API follow the same structure used by the IBM-supplied APIs, as shown in “Call Format” on page 51. Specifically, the input parameters must consist of the nine common input parameters, optionally followed by any number of additional input parameters, as follows:

input_length
function_name_length
function_name
authenticated_userid_length
authenticated_userid
password_length
password
target_identifier_length
target_identifier
additional_input_parameter_1
...
additional_input_parameter_n

The output parameters must consist of the four common output parameters, optionally followed by any number of additional output parameters, as follows:

output_length
request_id
return_code
reason_code
additional_output_parameter_1
...
additional_output_parameter_n

The type is either short-call or long-call. If the API is declared as short-call, the custom exec will be called by the one short-call worker server (as defined in the configuration file DMSSISVR NAMES). If the API is

© Copyright IBM Corp. 2003, 2025

declared as long-call, the custom exec will be called by one of the long-call worker servers. For proper operation of the server, if the custom exec performs a task that might have a significant delay, such as interacting with a directory manager, the API should be declared as long.

Writing the Custom EXEC

In this step, the custom EXEC is written in REXX (interpreted or compiled). The name of the custom EXEC cannot start with the reserved characters “DMS”.

When the custom EXEC is called, it is passed the following three input arguments:

1. The *authenticated_userid* parameter supplied by the client program, translated from ASCII to EBCDIC.
2. The *target_identifier* parameter supplied by the client program, translated from ASCII to EBCDIC.
3. The concatenation of the *additional_input_parameter_1* to *additional_input_parameter_n* parameters supplied by the client program, with no translation of any type.

Note that these are the *only* API input parameters available to the custom EXEC.

The custom EXEC must return a value that is the concatenation of the *return_code*, *reason_code*, and *additional_output_parameter_1* to *additional_output_parameter_n* parameters, to return to the client program. No translation of any type occurs when these parameters are returned to the client program. API output parameters other than these are controlled by the server, not the custom EXEC.

The custom EXEC can use privileged CP commands as defined by the privilege classes listed in the worker server’s directory entry. Care must be taken to avoid unintended results.

The server checks the authorization file before calling the custom EXEC. Therefore, the custom EXEC does not need to check the authorization file.

Each custom EXEC must call a unique EXEC to process it. The EXEC listed in DMSSIPRM NAMES (for non-Directory Manager APIs) and DMSSJBST COPY (for Directory Manager APIs) is part of the ESM resource name constructed when you configure an ESM to manage API authorization individually. The EXEC name must be unique to guarantee the ability to authorize APIs. SMAPI initialization will warn the operator if this requirement is violated. The following tools help you adhere to this requirement:

- The DMSAPISD EXEC checks for duplicate names. SMAPI initialization runs DMSAPISD and notifies the operator if duplicates are found. See “DMSAPISD” on page 886.
- The DMSAPISL EXEC lists the EXEC/documented API/ESM profile mappings. When creating new profiles to restrict access to a particular API, you pass DMSAPISL EXEC the API name you want to control and it provides the EXEC name to use in the ESM profile. To determine what existing profiles are used for, you pass DMSAPISL EXEC the EXEC name to find out the corresponding API. See “DMSAPISL” on page 887.

Installing the Custom EXEC

Follow these four steps to install the custom EXEC into the server:

1. Place the custom EXEC on the MAINT 193 disk.
2. Add an entry to the DMSSIUSR NAMES configuration file, defining the API name and the custom exec name. The entry must have the following format:

```
:nick.api_name
:program.custom_exec_filename
```

3. If the type of the API is long-call, add the API name to the value of the *ulong* variable in the configuration file DMSSICNF COPY. The assignment statement must have the following format:

```
ulong = 'api_name_1 api_name_2 ... api_name_n'
```

4. If necessary, authorize the request. For more information, see “Authorizing API Requests” on page 36 and “Configuring SMAPI to use an ESM to Authorize Requests” on page 36.

Return and Reason Codes

If the custom exec is called successfully, the return and reason codes of the API are the values given by the custom exec. Otherwise, if there is an error in calling the custom exec, some common return and reason codes are as follows:

RC	RC Name	RS	RS Name	Description
24	RCERR_SYNTAX	pprr	pprr	Syntax error in function parameter
100	RCERR_AUTH	8	RS_AUTHERR_ESM	Request not authorized by external security manager
		12	RS_AUTHERR_DM	Request not authorized by directory manager
		16	RS_AUTHERR_SERVER	Request not authorized by server
120	RCERR_USER_PW_BAD	0	RS_NONE	Authentication error; userid or password not valid
396	RCERR_INTERNAL	nnnn	psrc	Internal system error - product-specific return code (See “Internal Return Codes (RC = 396, 592, or 596)” on page 834)
596	RCERR_INTERNAL_DM	nnnn	psrc	Internal directory manager error - product-specific return code
900	RCERR_SERVER	12	RS_FUNCTION_NOT_VALID	Specified function does not exist
		20	RS_PARM_LIST_NOT_VALID	Total length does not match the specified input data
		24	RS_SFS_ERROR	Error accessing SFS directory
		36	RS_LENGTH_NOT_VALID	Specified length was not valid, out of valid server data range

Step-by-step Example

This section contains a step-by-step example, using a locally-defined API that simply sets three output parameters to the values of three input parameters.

Example: Designing the API

The name of the API is Echo_Parameters.

The API input parameters are as follows:

Parameter Name	Data Type	Example Value, in Hex	Interpretation of Example Value
input_length	int4	00000041	65
function_name_length	int4	0000000F	15

Parameter Name	Data Type	Example Value, in Hex	Interpretation of Example Value
function_name	string	4563686F 5F506172 616D6574 657273	“Echo_Parameters”
authenticated_userid_length	int4	00000008	8
authenticated_userid	string	534D434C 49454E54	“SMCLIENT”
password_length	int4	00000006	6
password	string	53454352 4554	“SECRET”
target_identifier_length	int4	00000005	5
target_identifier	string	55534552 32	“USER2”
switch_name_length	int4	00000007	7
switch_name	string	53574954 434831	“SWITCH1”
queue_limit	int4	00000014	20

The API output parameters are as follows:

Parameter Name	Data Type	Example Value, in Hex	Interpretation of Example Value
output_length	int4	0000001B	27
request_id	int4	0000054D	1357
return_code	int4	00000000	0
reason_code	int4	00000000	0
switch_name_length	int4	00000007	7
switch_name	string	53574954 434831	“SWITCH1”
queue_limit	int4	00000014	20

Note: Although this would be a short-call API by nature, it will be defined here as long-call for the sake of illustration.

Example: Writing the Custom Exec

Here is ECHOPARM EXEC, the custom exec:

```
/*********************************************************************/
/* ECHOPARM EXEC -- Custom exec for Echo_Parameters API               */
/*********************************************************************/

/*-------------------------------------------------------------------*/
/* Get the input arguments                                           */
/*-------------------------------------------------------------------*/
parse arg authenticatedUserid, targetIdentifier, inParms
                                /* authenticatedUserid is "SMCLIENT" */
                                /* targetIdentifier is "USER2" */

/*-------------------------------------------------------------------*/
/* Get the switch_name_length input parameter into a usable form     */
/*-------------------------------------------------------------------*/
parse value inParms with 1 inSwitchNameLen 5 inParms
inSwitchNameLen = c2d(inSwitchNameLen)
                                /* inSwitchNameLen is 7 */

/*-------------------------------------------------------------------*/
/* Get the switch_name input parameter into a usable form            */
/*-------------------------------------------------------------------*/
```

```
p = inSwitchNameLen + 1
parse value inParms with 1 inSwitchName =(p) inParms
'PIPE VAR INSWITCHNAME|XLATE A2E|VAR INSWITCHNAME'
                                /* inSwitchName is "SWITCH1" */

/*--------------------------------------------------------------------*/
/* Get the queue_limit input parameter into a usable form             */
/*--------------------------------------------------------------------*/
parse value inParms with 1 inQueueLimit 5 inParms
inQueueLimit = c2d(inQueueLimit)  /* inQueueLimit is 20 */

/*--------------------------------------------------------------------*/
/* Set the return_code output parameter, then convert it to the form  */
/* to return                                                          */
/*--------------------------------------------------------------------*/
returnCode = 0
returnCode = d2c(returnCode, 4)

/*--------------------------------------------------------------------*/
/* Set the reason_code output parameter, then convert it to the form  */
/* to return                                                          */
/*--------------------------------------------------------------------*/
reasonCode = 0
reasonCode = d2c(reasonCode, 4)

/*--------------------------------------------------------------------*/
/* Set the switch_name output parameter, then convert it to the form  */
/* to return                                                          */
/*--------------------------------------------------------------------*/
outSwitchName = inSwitchName
'PIPE VAR OUTSWITCHNAME|XLATE E2A|VAR OUTSWITCHNAME'

/*--------------------------------------------------------------------*/
/* Set the switch_name_length output parameter, then convert it to the */
/* form to return                                                     */
/*--------------------------------------------------------------------*/
outSwitchNameLen = length(outSwitchName)
outSwitchNameLen = d2c(outSwitchNameLen, 4)

/*--------------------------------------------------------------------*/
/* Set the queue_limit output parameter, then convert it to the form  */
/* to return                                                          */
/*--------------------------------------------------------------------*/
outQueueLimit = inQueueLimit
outQueueLimit = d2c(outQueueLimit, 4)

/*--------------------------------------------------------------------*/
/* Exit with the appropriate return value                             */
/*--------------------------------------------------------------------*/
outParms = returnCode || reasonCode || outSwitchNameLen || outSwitchName,
      || outQueueLimit
exit outParms
```

The input arguments are as follows:

Input Argument	Example Value in Hex	Interpretation of Example Value
authenticatedUserid	E2D4C3D3 C9C5D5E3	“SMCLIENT”
targetIdentifier	E4E2C5D9 F2	“USER2”
inParms	00000007 53574954 43483100 000014	7 || “SWITCH1” || 20

The return value is as follows:

Return Value	Example Value in Hex	Interpretation of Example Value
outParms	00000000 00000000 00000007 53574954 43483100 000014	0 || 0 || 7 || “SWITCH1” || 20

Example: Installing the Custom Exec

ECHOPARM EXEC is placed on the MAINT 193 disk.

The following entry is added to DMSSIUSR NAMES:

```
:nick.Echo_Parameters
:program.ECHOPARM
```

The following value is assigned in DMSSICNF COPY (assuming this is the only long-call API installed in the server):

```
ulong = 'Echo_Parameters'
```

The authorization file is updated to contain an entry with a *requested_function* field that includes “Echo_Parameters” (as described in “Authorizing API Requests” on page 36).

Appendix C. ENROLL and GRANT Commands Performed Automatically During z/VM Installation

The following lists show both the ENROLL and GRANT commands that are performed automatically during z/VM installation. They are shown here for verification and testing purposes. Also, if you are adding a new worker or request server, you can use the appropriate commands from these lists as a guide to enrolling your new server in the correct file pool and then granting SFS authorizations.

- ENROLL commands automatically performed during z/VM installation:

```
ENROLL USER VSMWORK1 VMSYS: (BLOCKS  6000 STORGROUP 2
ENROLL USER VSMWORK2 VMSYS:
ENROLL USER VSMWORK3 VMSYS:
ENROLL USER VSMREQIN VMSYS:
ENROLL USER VSMREQIU VMSYS:
ENROLL USER VSMGUARD VMPSFS: (BLOCKS  1000 STORGROUP 2
ENROLL USER VSMGUARD VMSYS:
ENROLL USER VSMREQI6 VMSYS:
ENROLL USER VSMEVSRV VMSYS:
ENROLL USER DTCSMAPI VMSYS:
ENROLL USER OPERATNS VMSYS:
ENROLL USER PERSMAPI VMSYS: (BLOCKS 24000 STORGROUP 2
```

- GRANT commands automatically performed during z/VM installation:

```
GRANT AUTHORITY VMSYS:VSMWORK1. TO MAINT    (WRITE NEWWRITE
GRANT AUTHORITY VMSYS:VSMWORK1.DATA TO MAINT    (WRITE NEWWRITE
GRANT AUTHORITY VMSYS:VSMWORK1. TO VSMGUARD (WRITE NEWWRITE
GRANT AUTHORITY VMSYS:VSMWORK1.DATA TO VSMGUARD (WRITE NEWWRITE
GRANT AUTHORITY VMSYS:VSMWORK1.STATUS TO VSMGUARD (WRITE NEWWRITE
GRANT AUTHORITY VMSYS:VSMWORK1.STATUS TO VSMWORK2 (WRITE NEWWRITE
GRANT AUTHORITY VMSYS:VSMWORK1.STATUS TO VSMWORK3 (WRITE NEWWRITE
GRANT AUTHORITY * * VMSYS:VSMWORK1. TO VSMGUARD (READ
GRANT AUTHORITY VMSYS:VSMWORK1. TO PERSMAPI (READ NEWREAD
GRANT AUTHORITY VMPSFS:VSMGUARD. TO DIRMAINT (READ NEWREAD
GRANT AUTHORITY VMPSFS:VSMGUARD. TO DIRMSAT  (READ NEWREAD
GRANT AUTHORITY VMPSFS:VSMGUARD. TO DIRMSAT2 (READ NEWREAD
GRANT AUTHORITY VMPSFS:VSMGUARD. TO DIRMSAT3 (READ NEWREAD
GRANT AUTHORITY VMPSFS:VSMGUARD. TO DIRMSAT4 (READ NEWREAD
GRANT AUTHORITY VMPSFS:VSMGUARD. TO DATAMOVE (READ NEWREAD
GRANT AUTHORITY VMPSFS:VSMGUARD. TO DATAMOV2 (READ NEWREAD
GRANT AUTHORITY VMPSFS:VSMGUARD. TO DATAMOV3 (READ NEWREAD
GRANT AUTHORITY VMPSFS:VSMGUARD. TO DATAMOV4 (READ NEWREAD
GRANT AUTHORITY VMPSFS:VSMGUARD. TO AUTOLOG1 (WRITE NEWWRITE
GRANT AUTHORITY VMPSFS:VSMGUARD. TO AUTOLOG2 (WRITE NEWWRITE
```

© Copyright IBM Corp. 2003, 2025

Appendix D. Sample Code

This section shows two sample programs using various APIs. The first is written in C, the second in Java.

Sample C Program

```
/*********************************************************************************/
/*                                                                               */
/* SAMPLE.C - Sample SMAPI client code.                                          */
/*                                                                               */
/* Some C-based tips to assist client application programmers can be found       */
/* in the following simple client coding example, which illustrates how to       */
/* call a representative SMAPI.                                                  */
/*                                                                               */
/* Set the constants and variables in the CONFIGURABLE VALUES section below,     */
/* and in the CONFIGURABLE CODE sections throughout the code, as needed to       */
/* to execute this code on your system.                                          */
/*                                                                               */
/* Code and comments that are marked "FOR z/VM" are applicable if this code      */
/* is executed on the z/VM platform.  Otherwise, this code should generally      */
/* execute correctly on any platform without major modifications.                */
/*                                                                               */
/* FOR z/VM: This code was compiled as follows.                                  */
/* c89 //sample.c //dmscsl.text -Wc,list -Wb,p,map -D_OE_SOCKETS -DVM            */
/*     -l//VMLIB                                                                 */
/*                                                                               */
/*********************************************************************************/

/* FOR z/VM: CSL linkage for ASCII/EBCDIC translation. */
#ifdef __VM__
#pragma linkage(DMSCSL,OS)
#endif

/* FOR z/VM: Clear the __POSIX_SYSTEM environment variable, because otherwise */
/* commands from the system() function will get routed to the OE Shell        */
/* instead of to the CMS command interpreter in a POSIX application.          */
#ifdef __VM__
#pragma runopts(POSIX(ON),ENVAR("__POSIX_SYSTEM=NO"))
#endif

/* FOR z/VM: External function definition for CSL. */
#ifdef __VM__
extern int DMSCSL(const char *RTNNAME,int *rc,...);
#endif

/* LE Sockets DEFINEs. */
#define _XOPEN_SOURCE_EXTENDED 1
#define _OPEN_THREADS
#define _OPEN_SYS
#define _OE_SOCKETS 1
#define _OPEN_MSGQ_EXT
#define _ALL_SOURCE
#define _OPEN_SYS_SOCK_EXT

#include <pthread.h>
#include <errno.h>
#include <vmcmt.h>
#include <features.h>
#include <sys/types.h>
#include <stdlib.h>
#include <stdio.h>
#include <stdarg.h>
#include <ctype.h>
#include <string.h>
#include <strings.h>
#include <sys/time.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <unistd.h>
#include <netdb.h>
#include <uio.h>
#include <arpa/nameser.h>
#include <net/if.h>
```

© Copyright IBM Corp. 2003, 2025

Sample Code

```
#include <sys/msg.h>

/* FOR z/VM: IUCV sockaddr structure. */
#ifdef __VM__
#include <saiucv.h>
#endif

/* CONFIGURABLE VALUES */

/* Set AUTHID to the name of the authenticated_userid (authorized client */
/* image for the API call).                                              */
char AUTHID[8] = "SMAPIC1\0";

/* Set FNAME to the name of the function for the API call.  For this example, */
/* we will call Authorization_List_Query, because this API illustrates the    */
/* use of binary (int), string, structure, and array SMAPI data types, and    */
/* the use of optional parameters.                                            */
char FNAME[25] = "Authorization_List_Query\0";

/* Set I_OR_V to 0 if you want to send a request to the SMAPI INET server. */
/* FOR z/VM: Set it to 1 if you want to send a request to the SMAPI IUCV   */
/* server.                                                                 */
int I_OR_V = 0;

/* Set IPADDRESS to the IPv4 IP address of your SMAPI server. */
char IPADDR[16] = "127.0.0.1";

/* FOR z/VM: Set IUCVUID to the name of your IUCV request server machine. */
char IUCVUID[9] = "VSMREQIU\0";

/* Max number of bytes for socket recv().  We are not likely to receive this */
/* much data from a single API call.                                         */
#define LIMIT 1000000

/* Size in bytes of I/O plist buffers (almost 1M, which should be more than  */
/* enough for input and output plists for virtually any API call).           */
#define PSIZE 1000000

/* Set PW to the password or passphrase of the AUTHID. */
char PW[9] = "PASSWORD\0";

/* Default SMAPI INET server port.  This may need to be changed depending */
/* on how the server is configured.                                       */
int SINETPORT = 44444;

/* Default timeout for select() in seconds.  This may need to be increased */
/* for some of the more complex/time-consuming APIs.                       */
#define STIMEOUT 20

/* Set TARGID to the name of the target image for the API call. */
char TARGID[8] = "SMAPIC1\0";

/* END CONFIGURABLE VALUES */

/* Other variables. */

int                      afamily = AF_INET;
char                     blanks[8] = "        ";
int                      bytetotal;
char                     ch;
int                      cinetport;  /* Client's INET port. */
int                      cnt1;
int                      flags = 0;
int                      i,j;
char                    *inplist_p = NULL;
char                     iucvprog[9] = "DMSRSRQU\0";  /* FOR z/VM: For IUCV. */
int                      len1,len2,len3;
time_t                   ltime;
int                      numfds;
int                      option = SO_REUSEADDR;
int                      optionval = 1;
int                      optionlen = sizeof(optionval);
char                    *outplist_p = NULL;
int                      plen;
int                      protocol = IPPROTO_IP;
int                      protolevel = SOL_SOCKET;
int                      rc = 0;
fd_set                   readfds;
struct sockaddr_in       saddr_in;
struct sockaddr_iucv     saddr_iucv;  /* FOR z/VM: For IUCV. */
char                    *save_p;
int                      sockaddrlen;
```

```
int                      sockid;
int                      socktype = SOCK_STREAM;
struct sockaddr_in      *sptr_in = &saddr_in;
struct sockaddr_iucv    *sptr_iucv = &saddr_iucv;  /* FOR z/VM: For IUCV. */
char                    *temp_p;
struct timeval           timeout;

/* FOR z/VM: ASCII/EBCDIC translation function. */
#ifdef __VM__
int xlate(char *,char *,int);
#endif

/*********************************************************************************/
/* MAIN                                                                          */
/*********************************************************************************/

main()
{
  /*-------------------------------------------------------------------------*/
  /* Setup.                                                                  */
  /*-------------------------------------------------------------------------*/

  printf("SAMPLE SMAPI CLIENT APPLICATION\n");

  /* Set up timeout for socket select() calls. */

  timeout.tv_sec = STIMEOUT;
  timeout.tv_usec = 0;

  /* Get storage for input and output API plists. */

  if ((inplist_p = (char *)calloc(PSIZE,1)) == NULL)
  {
    printf("ERROR> calloc() input plist error.\n");
    goto main_exit;
  }
  printf("Input plist address = %08p\n",inplist_p);

  if ((outplist_p = (char *)calloc(PSIZE,1)) == NULL)
  {
    printf("ERROR> calloc() output plist error.\n");
    goto main_exit;
  }
  printf("Output plist address = %08p\n",outplist_p);

  /*-------------------------------------------------------------------------*/
  /* Create the input plist.                                                 */
  /*-------------------------------------------------------------------------*/

  printf("Creating input plist.\n");

  /* Initialize temp pointer to 4 bytes past the start of the plist.  We will */
  /* fill in the plist input_length parm later, after we see how big the      */
  /* plist turns out to be.                                                   */

  /* FOR z/VM: z/VM uses EBCDIC, but the SMAPI server expects input to be in  */
  /* ASCII, because the data is always ASCII on the wire.  Since the server   */
  /* runs on z/VM, it translates incoming data from ASCII to EBCDIC.  Thus a  */
  /* z/VM client must translate its input from EBCDIC to ASCII before sending */
  /* it to the server, using the code pages specified for the SMAPI server.   */
  /* Note that binary (int) parms do not need translation; only string parms  */
  /* get translated.                                                          */

  temp_p = (char *)((int)inplist_p + 4);

  /* Fill in input plist header parameters (except input_length). */

  *((int *)temp_p) = strlen(FNAME);  /* function_name_length */
  temp_p = (char *)((int)temp_p + 4);

  memcpy(temp_p,FNAME,strlen(FNAME));  /* function_name */

  /* FOR x/VM: Translate string from EBCDIC to ASCII. */
#ifdef __VM__
  if ((rc = xlate(temp_p,"A",strlen(FNAME))) != 0)
  {
    printf("ERROR> xlate() error.\n");
    goto main_exit;
  }
#endif

  temp_p = (char *)((int)temp_p + strlen(FNAME));
```

```
  *((int *)temp_p) = strlen(AUTHID);  /* authenticated_userid_length */
  temp_p = (char *)((int)temp_p + 4);

  memcpy(temp_p,AUTHID,strlen(AUTHID));  /* authenticated_userid */

  /* FOR x/VM: Translate string from EBCDIC to ASCII. */
#ifdef __VM__
  if ((rc = xlate(temp_p,"A",strlen(AUTHID))) != 0)
  {
    printf("ERROR> xlate() error.\n");
    goto main_exit;
  }
#endif

  temp_p = (char *)((int)temp_p + strlen(AUTHID));

  *((int *)temp_p) = strlen(PW);  /* password_length */
  temp_p = (char *)((int)temp_p + 4);

  memcpy(temp_p,PW,strlen(PW));  /* password */

  /* FOR x/VM: Translate string from EBCDIC to ASCII. */
#ifdef __VM__
  if ((rc = xlate(temp_p,"A",strlen(PW))) != 0)
  {
    printf("ERROR> xlate() error.\n");
    goto main_exit;
  }
#endif

  temp_p = (char *)((int)temp_p + strlen(PW));

  /* CONFIGURABLE CODE */

  /* Note that we choose in this example to not specify the optional  */
  /* target_id string (as indicated by string length = 0), so that we */
  /* can query the entire contents of the server's AUTHLIST file via  */
  /* Authorization_List_Query.                                        */

  *((int *)temp_p) = 0;  /* target_identifier_length */
  temp_p = (char *)((int)temp_p + 4);

  /* Fill in function-specific parameters.  For this example, we will fill */
  /* in parms for Authorization_List_Query.  We will choose to not specify */
  /* the optional for_id and function_id strings (as indicated by string   */
  /* length = 0), so that we can query the entire contents of the server's */
  /* AUTHLIST file.                                                        */

  *((int *)temp_p) = 0;  /* for_id_length */
  temp_p = (char *)((int)temp_p + 4);

  *((int *)temp_p) = 0;  /* function_id_length */
  temp_p = (char *)((int)temp_p + 4);

  /* END CONFIGURABLE CODE */

  /* Fill in the first header parameter (input_length), now that we know how */
  /* big the plist is.  Note that input_length is the total length of all of */
  /* the parms that follow it (it does not include itself in that total).    */

  i = (int)temp_p - (int)inplist_p - 4;
  *((int *)inplist_p) = i;

  /* Save total plist length (including the input_length parm) for when we */
  /* send() the request later.                                             */

  plen = i + 4;
  printf("Input plist length = %08x = %id\n",plen,plen);

  /* Display the whole input plist. */

  printf("Input plist contents (ASCII hex):\n");
  temp_p = inplist_p;

  for (i = 1; i <= plen; i++)
  {
    printf("%02x",*temp_p);
    temp_p = (char *)((int)temp_p + 1);
  }
  printf("\n");
```

```
/*--------------------------------------------------------------------------*/
/* Create a TCP (stream) socket and connect to a SMAPI server.              */
/*--------------------------------------------------------------------------*/

if (I_OR_V == 0)
{ /* INET SERVER */

  /* Choose a client INET port using a random number between 1024 and */
  /* RAND_MAX (32767).                                                */

  time(&ltime);
  srand(ltime);
  cinetport = rand();

  if (cinetport < 1024)
    cinetport = cinetport + 1024;  /* Don't use a reserved port. */

  if (cinetport == SINETPORT)
    cinetport = cinetport++;  /* Don't use the server's port. */

  /* Open a socket. */

  printf("Opening an AF_INET socket.\n");

  if ((sockid = socket(AF_INET,socktype,protocol)) < 0)
  {
    printf("ERROR> AF_INET socket() errno = %i\n",errno);
    goto main_exit;
  }
  else
    printf("socket() succeeded for socket id %i.\n",sockid);

  /* Set SO_REUSEADDR option so port can be reused if necessary. */

  printf("Setting SO_REUSEADDR.\n");

  if ((rc = setsockopt(sockid,protolevel,option,&optionval,optionlen)) < 0)
  {
    printf("ERROR> AF_INET setsockopt() errno = %i\n",errno);
    goto closesock;
  }
  else
    printf("setsockopt() succeeded.\n");

  /* Bind the socket. */

  printf("Binding the socket.\n");

  memset(&saddr_in,0,sizeof(saddr_in));
  saddr_in.sin_len = sizeof(saddr_in);
  saddr_in.sin_family = afamily;
  saddr_in.sin_port = htons(cinetport);
  saddr_in.sin_addr.s_addr = inet_addr(IPADDR);
  sockaddrlen = sizeof(saddr_in);

  if ((rc = bind(sockid,sptr_in,sockaddrlen)) < 0)
  {
    printf("ERROR> AF_INET bind() errno = %i\n",errno);
    goto closesock;
  }
  else
    printf("bind() succeeded.\n");

  /* Connect to the server. */

  printf("Connecting to the server.\n");

  saddr_in.sin_port = htons(SINETPORT);

  if ((rc = connect(sockid,sptr_in,sockaddrlen)) < 0)
  {
    printf("ERROR> AF_INET connect() errno = %i\n",errno);
    goto closesock;
  }
  else
    printf("connect() succeeded.\n");

} /* USING INET SERVER */

else
if (I_OR_V == 1)
{ /* FOR z/VM: USING IUCV SERVER */
```

```
  /* Open a socket. */

  printf("Opening an AF_IUCV socket.\n");

  if ((sockid = socket(AF_IUCV,socktype,protocol)) < 0)
  {
    printf("ERROR> AF_IUCV socket() errno = %i\n",errno);
    goto closesock;
  }
  else
    printf("socket() succeeded for socket id %i.\n",sockid);

  /* Bind the socket. */

  printf("Binding the socket.\n");

  memset(&saddr_iucv,0,sizeof(saddr_iucv));
  saddr_iucv.siucv_len = sizeof(saddr_iucv);
  saddr_iucv.siucv_family = AF_IUCV;
  saddr_iucv.siucv_port = 0;
  saddr_iucv.siucv_addr = 0;
  memcpy(&saddr_iucv.siucv_nodeid,&blanks,8);
  memcpy(&saddr_iucv.siucv_userid,&blanks,8);
  memcpy(&saddr_iucv.siucv_name,&blanks,8);
  sockaddrlen = sizeof(saddr_iucv);

  if ((rc = bind(sockid,sptr_iucv,sockaddrlen)) < 0)
  {
    printf("ERROR> AF_IUCV bind() errno = %i\n",errno);
    goto closesock;
  }
  else
    printf("bind() succeeded.\n");

  /* Connect to the server. */

  printf("Connecting to the server.\n");

  memcpy(&saddr_iucv.siucv_userid,&IUCVUID,8);
  memcpy(&saddr_iucv.siucv_name,&iucvprog,8);

  if ((rc = connect(sockid,sptr_iucv,sockaddrlen)) < 0)
  {
    printf("ERROR> AF_IUCV connect() errno = %i\n",errno);
    goto closesock;
  }
  else
    printf("connect() succeeded.\n");

} /* USING IUCV SERVER */

else
{
  printf("ERROR> Invalid I_OR_V value.\n");
  goto closesock;
}

/*-------------------------------------------------------------------------*/
/* Send the request to the server.                                         */
/*-------------------------------------------------------------------------*/

/* Note that a simple send() is used for this example.  sendto()/recvfrom(),*/
/* sendmsg()/recvmsg(), and write()/read() could also be used with our      */
/* connected socket instead of send()/recv() (with a zeroed flags parameter,*/
/* send() is equivalent to write() ).  Also note that the underlying        */
/* protocol for this socket should generally be able to handle any size     */
/* individual SMAPI message (input plist).  However, if multiple input      */
/* plists are to be sent at once, and the message becomes too big to pass   */
/* atomically through the underlying protocol, then send() will return      */
/* errno EMSGSIZE.  If at a given time the message is too big to fit in the */
/* socket's send buffer, then a blocking socket will block until enough     */
/* buffer space becomes available, and a nonblocking socket will return     */
/* errno EAGAIN or EWOULDBLOCK.  In the latter case, select() or poll() can */
/* be used to determine when it becomes possible to send the message.  Our  */
/* socket in this example is a blocking socket.                             */

/* Send the request. */

printf("Sending API request on socket %i.\n",sockid);

if ((rc = send(sockid,inplist_p,plen,flags)) < 0)
```

```
{
  printf("ERROR> send() errno = %i\n",errno);
  goto closesock;
}
else
  printf("send() succeeded.\n");

/*-------------------------------------------------------------------------*/
/* Receive output from the server.                                         */
/*-------------------------------------------------------------------------*/

/* There are 2 logical responses per socket - first the request id, and    */
/* then the API output plist.  For clarity, separate recv()'s are done in  */
/* this example for the request id and the output plist.                   */

/* See if the socket is ready for recv(). */

printf("Checking the socket.\n");

FD_ZERO(&readfds);
FD_SET(sockid,&readfds);
numfds = sockid + 1;  /* This parm is highest socket descriptor + 1. */

if ((rc = select(numfds,&readfds,NULL,NULL,&timeout)) < 0)
{
  printf("ERROR> select() errno = %i\n",errno);
  goto closesock;
}
else
if (rc == 0)
{
  printf("ERROR> No response from server, client timed out (%i sec).\n",
         timeout.tv_sec);
  goto closesock;
}
else
  printf("select() succeeded.\n");

/* Receive the request id. */

printf("Receiving request id on socket %i.\n",sockid);

if ((rc = recv(sockid,outplist_p,4,flags)) < 0)
{
  printf("ERROR> Request id recv() errno = %i\n",errno);
  goto closesock;
}
else
  printf("Request id recv() succeeded.\n");

/* Display the request id in hex.  Note that this is always the first 4  */
/* bytes of the output.                                                  */

/* FOR z/VM: Note that ASCII/EBCDIC translation is not needed, because   */
/* the request id is a binary value.                                     */

printf("Request id = %08x = %id\n",*((int *)outplist_p),*((int *)outplist_p));

/* See if the socket is ready for recv(). */

printf("Checking the socket.\n");

FD_ZERO(&readfds);
FD_SET(sockid,&readfds);

if ((rc = select(numfds,&readfds,NULL,NULL,&timeout)) < 0)
{
  printf("ERROR> select() errno = %i\n",errno);
  goto closesock;
}
else
if (rc == 0)
{
  printf("ERROR> No response from server, client timed out (%i sec).\n",
         timeout.tv_sec);
  goto closesock;
}
else
  printf("select() succeeded.\n");

/* Receive the API output plist. */
```

```
printf("Receiving API output plist on socket %i.\n",sockid);

/* First, receive the first 4 bytes of the output plist, which give the */
/* total length of the remainder of the output plist.                   */

if ((rc = recv(sockid,outplist_p,4,flags)) < 0)
{
  printf("ERROR> Output plist length recv() errno = %i\n",errno);
  goto closesock;
}
else
{
  printf("Output plist length recv() succeeded.\n");
  plen = *((int *)outplist_p);
}

/* Now loop (if necessary) to receive the rest of the output plist. */

bytetotal = 0;
temp_p = (char *)((int)outplist_p + 4);

while (bytetotal < plen)
{
  if ((rc = recv(sockid,temp_p,LIMIT,flags)) < 0)
  {
    printf("ERROR> Output plist recv() errno = %i\n",errno);
    goto closesock;
  }
  else
  {
    bytetotal += rc;
    temp_p = (char *)((int)temp_p + rc);
  }
}

printf("Output plist recv() succeeded.\n");
printf("Output plist length = %08x = %id\n",plen,plen);

/*--------------------------------------------------------------------------*/
/* Display the output.                                                      */
/*--------------------------------------------------------------------------*/

/* Display the whole output plist. */

printf("Output plist contents (ASCII hex):\n");
temp_p = outplist_p;

for (i = 1; i <= plen; i++)
{
  printf("%02x",*temp_p);
  temp_p = (char *)((int)temp_p + 1);
}
printf("\n");

/* Initialize temp pointer to the start of the output plist.                */

/* FOR z/VM: As per the explanation given earlier when we filled in the     */
/* input plist, the z/VM client must now translate the output data from     */
/* ASCII to EBCDIC.  Again note that binary (int) parms do not need trans-  */
/* lation; only string string parms get translated.                         */

temp_p = outplist_p;

/* Display the output header parms. */

printf("output_length = %08x = %id\n",*((int *)temp_p),*((int *)temp_p));
temp_p = (char *)((int)temp_p + 4);

printf("request_id = %08x = %id\n",*((int *)temp_p),*((int *)temp_p));
temp_p = (char *)((int)temp_p + 4);

printf("return_code = %08x = %id\n",*((int *)temp_p),*((int *)temp_p));
temp_p = (char *)((int)temp_p + 4);

printf("reason_code = %08x = %id\n",*((int *)temp_p),*((int *)temp_p));
temp_p = (char *)((int)temp_p + 4);

/* CONFIGURABLE CODE */

/* Display the function-specific output parms for Authorization_List_Query. */

len1 = *((int *)temp_p);  /* Save array length. */
```

```
  temp_p = (char *)((int)temp_p + 4);

  /* Process array of structures. */

  cnt1 = 0;

  if (len1 > 0)
  {
    do
    {
      printf("auth_record_array_length = %08x = %id\n",len1,len1);

      /* Display as many requesting_userid/for_userid/function_name     */
      /* lengths, strings, and associated list_indicators as necessary. */

      printf("auth_record_structure_length = %08x = %id\n",*((int *)temp_p),
             *((int *)temp_p));
      len2 = *((int *)temp_p);  /* Save structure length. */
      temp_p = (char *)((int)temp_p + 4);
      cnt1 = cnt1 + 4;

      if (len2 > 0)
      {
        for (i = 1; i <= 3; i++)
        {
          /* Get length of string, move to next field. */

          printf("string_length = %08x = %id\n",*((int *)temp_p),
                 *((int *)temp_p));
          len3 = *((int *)temp_p);  /* Save string length. */
          temp_p = (char *)((int)temp_p + 4);
          cnt1 = cnt1 + 4;

          if (len3 > 0)
          {
            /* FOR x/VM: Translate string from ASCII to EBCDIC. */
#ifdef __VM__
            if ((rc = xlate(temp_p,"E",len3)) != 0)
            {
              printf("ERROR> xlate() error.\n");
              goto closesock;
            }
#endif

            save_p = temp_p;
            printf("string (hex) = ");
            for (j = 1; j <= len3; j++)
            {
              printf("%02x",*temp_p);
              temp_p = (char *)((int)temp_p + 1);
            }
            printf("\n");

            temp_p = save_p;
            printf("string (char) = ");
            for (j = 1; j <= len3; j++)
            {
              printf("%c",*temp_p);
              temp_p = (char *)((int)temp_p + 1);
            }
            printf("\n");

            cnt1 = cnt1 + len3;

            printf("list_indicator = %02x\n",*temp_p);
            temp_p = (char *)((int)temp_p + 1);
            cnt1 = cnt1 + 1;
          }
          else
          {
            printf("ERROR> Output error.\n");
            goto closesock;
          }
        }  /* End of for loop. */
      }
      else
      {
        printf("ERROR> Output error.\n");
        goto closesock;
      }
    } while (cnt1 < len1);
  }
```

```
  /* END CONFIGURABLE CODE */

#ifdef __VM__
  printf("Output plist contents (EBCDIC hex):\n");
  temp_p = outplist_p;

  for (i = 1; i <= plen; i++)
  {
    printf("%02x",*temp_p);
    temp_p = (char *)((int)temp_p + 1);
  }
  printf("\n");
#endif

  /*-------------------------------------------------------------------------*/
  /* Close the socket.                                                       */
  /*-------------------------------------------------------------------------*/

  closesock:

  printf("Closing the socket.\n");

  if ((rc = shutdown(sockid,SHUT_RDWR)) < 0)
  {
    printf("ERROR> shutdown() errno = %i\n",errno);
    goto main_exit;
  }
  else
    printf("shutdown() succeeded.\n");

  if ((rc = close(sockid)) < 0)
  {
    printf("ERROR> close() errno = %i\n",errno);
    goto main_exit;
  }
  else
    printf("close() succeeded.\n");

  /*-------------------------------------------------------------------------*/
  /* End of main().                                                          */
  /*-------------------------------------------------------------------------*/

  main_exit:

  free(inplist_p);
  free(outplist_p);

}  /* End of main().

/*****************************************************************************/
/* SUBROUTINE FUNCTIONS                                                      */
/*****************************************************************************/

/* FOR z/VM: ASCII <-> EBCDIC translation. */

#ifdef __VM__
int xlate(char *bufp,char *eora,int num)
{
  char    a2etab[257] = "\0";
  char    ch;
  char   *chp;
  char    crlf[3] = "\0";
  char    e2atab[257] = "\0";
  int     i,ind,val;
  int     rc = 0;
  int     rs = 0;
  char    tabname[9] = "09240923\0";

  /* Get translation tables. */

  DMSCSL("DTCXLATE",&rc,&rs,tabname,&a2etab,&e2atab,&crlf,"QUIET",5);

  if ((rc != 0) || (rs != 0))
  {
    printf("ERROR> DTCXLATE error: rc=%i, rs=%i.\n",rc,rs);
    goto xlate_exit;
  }

  /* Translate contents of buffer in-place. */

  chp = bufp;
```

```
  for (i = 0; i < num; i++)
  {
    ch = *chp;
    ind = ch;

    if (strncmp(eora,"E",1) == 0)
      ch = a2etab[ind];  /* Translate ASCII to EBCDIC. */

    if (strncmp(eora,"A",1) == 0)
      ch = e2atab[ind];  /* Translate EBCDIC to ASCII. */

    *chp = ch;
    chp = (char *)((int)chp + 1);
  }

  xlate_exit:
  return(rc);
}
#endif

/*****************************************************************************/

/* EOF */
```

Sample Java Program

```
/**
 * Test code showing an invocation of the SMAPI Image_Active_Configuration_Query API.
 * Issues a query and then uses toString() on the response object to dump the response to
 * stdout.
 *
 * Note several request parms and response parms are assumed to be in the default code page.
 *
import java.net.*;
import java.util.*;
import java.io.*;
public class SmapiConfigQueryRequest {

   public static final byte[] FUNCTION_IMAGE_QUERY = "Image_Active_Configuration_Query".getBytes();
   public static final int    RC_OK = 0;

   private static final String USE_MSG =
      "Syntax: SmapiConfigQueryRequest target-machine "
      +"port auth-uid auth-pw uid\n"
      +" Where:\n"
      +"  target-machine   is the ip address of the machine hosting the SMAPI server\n"
      +"  port             is the port on which the SMAPI server is listening\n"
      +"  auth-uid         is the name of a user on the machine that will be used "
      +"to authenticate the request\n"
      +"  auth-pw          is the password of auth-uid\n"
      +"  uid              is the name of userid being queried";

   /**
    *
    */
   private static void usemsg() {
      System.err.println( USE_MSG );
      System.exit( 1 );
   }

   /**
    * Main
    */
   public static void main( String[] args ) {
      if ( args.length != 5 ) {
         usemsg();
      }
      try {
         SmapiConfigQueryRequest qvr =
            new SmapiConfigQueryRequest( args[0], Integer.parseInt( args[1] ), args[2], args[3] );
         System.out.println( qvr.query( args[4] ) );
      } catch ( Exception e ) {
         System.err.println( "Failure building or processing request" );
         e.printStackTrace();
      }
   }

   /**
```

```
    * Given a length and the inbound stream, read a string
    * NOTE: Assuming default code page here.
    */
   private static String readString( int length, DataInputStream in ) throws IOException {
      byte[] bytes = new byte[ length ];
      in.readFully( bytes );
      return new String( bytes );
   }

   /**
    * An object that holds the response
    */
   public class ConfigQueryResponse {
      /**
       * Info about a specific CPU
       */
      public class CpuInfo {
         int         number;
         String      id;
         byte        status;
         /**
          * Create yourself given the inbound data stream
          */
         CpuInfo( DataInputStream in ) throws IOException {
            int structLength = in.readInt();
            number = in.readInt();
            id = readString( in.readInt(), in );
            status = in.readByte();
         }
         /**
          * Format contents into the provided StringBuffer
          */
         public void toString( StringBuffer sb ) {
            sb.append( "\tid     " ).append( id ).append( "\n" );
            sb.append( "\tnumber " ).append( number ).append( "\n" );
            sb.append( "\tstatus " ).append( status ).append( "\n" );
         }
         /**
          * How much did we consume off the wire?
          * Required because the returned plist specifies the entire
          * length of the arrays so each specific object could take different
          * amounts of data off the wire.  This will be called after we've
          * been created to decrement the 'array length' to ensure we stop
          * at the appropriate point in time.
          */
         int size() {
            return 4          // struct length
               + 4            // number
               + 4            // string length
               + id.length()  // the string itself
               + 1;           // status byte
         }
      }
      /**
       * An object that contains information about a device.
       */
      public class DeviceInfo {
         byte     type;
         String   address;

         /**
          * Given an inbound DataStream, consume the appropriate amount
          */
         DeviceInfo( DataInputStream in ) throws IOException {
            int structLength = in.readInt();
            type = in.readByte();
            address = readString( in.readInt(), in );
         }
         /**
          * Format contents into the provided StringBuffer
          */
         public void toString( StringBuffer sb ) {
             sb.append( "\taddress " )
               .append( address )
               .append( " (type ")
               .append( type ).append( ")\n" );
         }
         /**
          * How much did we consume off the wire?
          * Required because the returned plist specifies the entire
          * length of the arrays so each specific object could take different
          * amounts of data off the wire.  This will be called after we've
```

```
         * been created to decrement the 'array length' to ensure we stop
         * at the appropriate point in time.
         */
        int size() {
           return 1               // Type
              +   4               // struct length
              +   4               // address length
              + address.length();  // the string itself
        }
     }

     int immedRequestVerification;

     int outputLength;
     int requestId;
     int returnCode;
     int reasonCode;
     int memorySize;

     byte memoryUnit;
     byte shareType;

     int shareValueLength;
     int numberOfCPUs;

     String shareValue;

     List cpuInfo = new ArrayList();     // of CpuInfo objects
     List deviceInfo = new ArrayList();  // of DeviceInfo objects

     /**
      * Read the response
      */
     ConfigQueryResponse( DataInputStream in ) throws IOException {
        immedRequestVerification = in.readInt();
        outputLength = in.readInt();

        requestId = in.readInt();
        returnCode = in.readInt();
        reasonCode = in.readInt();

        if ( returnCode != RC_OK ) {
           throw new RuntimeException( "Query failed, return code: "
                                       + returnCode + " reason code: " + reasonCode );
        }

        memorySize = in.readInt();
        memoryUnit = in.readByte();
        shareType = in.readByte();
        shareValueLength = in.readInt();
        byte[] shareValueByteArray = new byte[shareValueLength];
        in.readFully(shareValueByteArray);
        shareValue = new String(shareValueByteArray);
        numberOfCPUs = in.readInt();

        int cpuInfoArrayLength = in.readInt();

        while ( cpuInfoArrayLength > 0 ) {
           CpuInfo newCpuInfo = new CpuInfo( in );
           cpuInfo.add( newCpuInfo );
           cpuInfoArrayLength -= newCpuInfo.size();

        }

        int deviceInfoArrayLength = in.readInt();
        while ( deviceInfoArrayLength > 0 ) {
           DeviceInfo newDeviceInfo = new DeviceInfo( in );
           deviceInfo.add( newDeviceInfo );
           deviceInfoArrayLength -= newDeviceInfo.size();
        }
     }
     /**
      * Format our state.
      */
     public String toString() {
        StringBuffer sb = new StringBuffer( 256 );

        sb.append( "Verification " ).append( immedRequestVerification ).append( "\n" );
        sb.append( "RequestId    " ).append( requestId ).append( "\n" );
        sb.append( "MemorySize   " ).append( memorySize ).append( "\n" );
        sb.append( "MemoryUnit   " ).append( memoryUnit ).append( "\n" );
        sb.append( "ShareType    " ).append( shareType ).append( "\n" );
```

```
         sb.append( "ShareValue   " ).append( shareValue ).append( "\n" );
         sb.append( "MumberOfCPUs " ).append( numberOfCPUs ).append( "\n" );

         sb.append( (cpuInfo.size()>0 ? "CPU info follows" : "no CPU info returned") ).append( "\n" );
         Iterator iter = cpuInfo.iterator();
         while ( iter.hasNext() ) {
            ((CpuInfo)iter.next()).toString(sb);
         }
         sb.append(
            (deviceInfo.size()>0 ? "device info follows" : "no device info returned") )
           .append( "\n" );
         iter = deviceInfo.iterator();
         while ( iter.hasNext() ) {
            ((DeviceInfo)iter.next()).toString(sb);
         }
         return sb.toString();
      }
   }


   protected byte[]  host;
   protected String  hostStr;
   protected int     port;
   protected byte[]  user;
   protected byte[]  password;

   /**
    * Create a query request
    */
   public SmapiConfigQueryRequest( String host, int port, String user, String password ) {
      this.host = host.getBytes();
      this.hostStr = host;
      this.port = port;
      this.user = user.getBytes();
      this.password = password.getBytes();
   }

   /**
    * Issue the query
    */
   public ConfigQueryResponse query( String targetUser )
            throws SocketException, UnknownHostException, IOException {
      Socket sock = null;
      DataOutputStream out = null;
      DataInputStream in = null;
      try {
         byte[] target = targetUser.getBytes();
         sock = new Socket( hostStr, port );
         out = new DataOutputStream( new BufferedOutputStream( sock.getOutputStream() ) );
         in = new DataInputStream( new BufferedInputStream( sock.getInputStream() ) );
         int inputParmLen =   4 + FUNCTION_IMAGE_QUERY.length
                              + 4 + user.length
                              + 4 + password.length
                              + 4 + target.length;

         // Write plist
         out.writeInt( inputParmLen );
         out.writeInt( FUNCTION_IMAGE_QUERY.length );
         out.write( FUNCTION_IMAGE_QUERY );
         out.writeInt( user.length );
         out.write( user );
         out.writeInt( password.length );
         out.write( password );
         out.writeInt( target.length );
         out.write( target );
         out.flush();

         // consume the response
         return new ConfigQueryResponse( in );
      } finally {
         try {out.close();} catch ( Exception e ) {}
         try {in.close();} catch ( Exception e ) {}
         try {sock.close();} catch ( Exception e ) {}
      }
   }
}
```

Appendix E. Diagnosing Configuration Errors During Server Startup

In the following table, use the following key to interpret the **Affected Areas** column:

SMAPI
The SMAPI server will not function due to the server console output being displayed. In this case, the server console output will be followed by the line:

```
SMAPI will not work due to config errors
```

API Functions
The specific API functions listed will not function due to the server console output being displayed. In this case, the server console output will be followed by the line:

```
Some APIs will not work due to config incomplete
```

Table 25. Configuration Errors, With Explanation and Affected Areas

Server Console Output	Explanation	Affected Areas

Invalid number of guards	Must have *exactly one* guard server entry:
	`:type.WORKER`
	`:short.GUARD`
	as described in "The Server Names File" on page 27.
		SMAPI

Must have at least one short call server	Must have *at least one* short call server entry:
	`:type.WORKER`
	`:short.YES`
	as described in "The Server Names File" on page 27.
		SMAPI

Must have exactly one event server	Must have *exactly one* event server entry:
	`:type.WORKER`
	`:short.AF_EVNT`
	as described in "The Server Names File" on page 27, or else some API functions will not work.
		API functions:
		• Event_Stream_Add
		• Event_Subscribe
		• Event_Unsubscribe

© Copyright IBM Corp. 2003, 2025

Table 25. Configuration Errors, With Explanation and Affected Areas (continued)

Server Console Output	Explanation	Affected Areas

Must have exactly one performance monitoring machine	Must have *exactly one* performance monitoring machine entry:
	`:type.WORKER`
	`:short.PMM`
	as described in "The Server Names File" on page 27, or else some API functions will not work.
		API functions:
		• System_Performance_Threshold_Disable
		• System_Performance_Threshold_Enable

• Must have exactly one private TCP/IP stack
• Private TCP/IP stack name is incorrect
	• Must have *exactly one* private TCP/IP stack entry:
	`:type.WORKER`
	`:short.PSTK`
	as described in "The Server Names File" on page 27.
	• The private TCP/IP stack entry must be:
	`:server.DTCSMAPI`
		SMAPI

Must have exactly one dump handler to use ABEND_Dump-related APIs	Must have *exactly one* dump handler entry:
	`:type.WORKER`
	`:short.DMPH`
	as described in "The Server Names File" on page 27, or else some API functions will not work.
		API functions:
		• Delete_ABEND_Dump
		• Process_ABEND_Dump
		• Query_ABEND_Dump

Should have exactly one database server	Should have *exactly one* database server entry:
	`:type.WORKER`
	`:short.DBS`
	as described in "The Server Names File" on page 27.
		SMAPI
		(There may be a significant performance impact if no database server is configured. No more than one database server may be configured).

Must have at least one long call server	Must have *at least one* long call server entry:
	`:type.WORKER`
	`:short.NO`
	as described in "The Server Names File" on page 27.
		SMAPI

Directory Manager not configured correctly	A directory manager is not up and running, or has incorrect configuration setup. See "The Directory Manager" on page 4.	**SMAPI**

Table 25. Configuration Errors, With Explanation and Affected Areas (continued)

Server Console Output	Explanation	Affected Areas
Must have no more than one directory manager	Must have *no more than one* directory manager entry: `:type.DMGR` as described in "The Server Names File" on page 27.	**SMAPI**
TCP/IP is not working	TCP/IP is not operating or is configured incorrectly.	**SMAPI**
Cannot check TCPMAINT for correct TCP/IP configuration	Not able to link and/or access TCPMAINTs 198 and/or 591 disks due to changes to the installed defaults.	Will not allow TCP/IP configuration checking that may catch possible problems as shown in next table entry.
filename filetype filemode not found	The VSMWORK1 AUTHLIST, DMSSICNF COPY, or DMSSISVR NAMES file is missing. (Note that the name of the VSMWORK1 AUTHLIST file is configurable.)	**SMAPI**

Appendix F. Using SMAPI with an External Security Manager

An External Security Manager (ESM) controls who can have access, and what kind of access they can have, to specific data files and disks. If an ESM is implemented at your installation, SMAPI must be given the appropriate access to the disks and files you want it to manage. This can be done using an ESM such as RACF (Resources Access Control Facility).

This appendix describes how to enable the proper RACF authorizations for use with SMAPI:

- Guidance for defining the SMAPI service machines to your ESM
- Granting the necessary authority to the various SMAPI service machines.

These recommendations are optional and whether you follow them depends on the level of security that your installation requires.

If you add additional SMAPI server machines to your system at a later time, remember to review this chapter and perform the necessary steps for the new service machines.

The use of an ESM is optional. If you do not have an ESM installed on your system, you may skip this appendix.

During initialization, SMAPI attempts to verify the RACROUTE configuration described in "Enabling RACROUTE" on page 877. SMAPI will not start if there are configuration errors. In this case, the appropriate messages are issued to the system operator.

Using SMAPI with RACF

RACF for z/VM can be used to enhance the security and integrity of your system by:

- Helping your installation implement its security policy
- Identifying and authenticating each user
- Controlling each user's access to sensitive data
- Logging and reporting events that are relevant to the system's security.

For more information on RACF for z/VM, see the RACF publications listed in the "Bibliography" on page 895.

For information on setting up DIRMAINT with RACF, see "Appendix A. External Security Manager Considerations" in the *z/VM: Directory Maintenance Facility Tailoring and Administration Guide*.

Enabling RACROUTE

Enable the appropriate SMAPI service machines to use RACROUTE services. Enter the following:

```
RAC SETROPTS CLASSACT(FACILITY)
RAC SETROPTS RACLIST(FACILITY)
RAC RDEFINE FACILITY ICHCONN UACC(NONE)
RAC PERMIT ICHCONN CLASS(FACILITY) ID(VSMREQI6) ACCESS(UPDATE)
RAC PERMIT ICHCONN CLASS(FACILITY) ID(VSMREQIN) ACCESS(UPDATE)
RAC PERMIT ICHCONN CLASS(FACILITY) ID(VSMREQIU) ACCESS(UPDATE)
RAC PERMIT ICHCONN CLASS(FACILITY) ID(VSMEVSRV) ACCESS(UPDATE)
RAC PERMIT ICHCONN CLASS(FACILITY) ID(DTCSMAPI) ACCESS(UPDATE)
RAC PERMIT ICHCONN CLASS(FACILITY) ID(VSMGUARD) ACCESS(UPDATE)
RAC PERMIT ICHCONN CLASS(FACILITY) ID(VSMWORK1) ACCESS(UPDATE)
RAC PERMIT ICHCONN CLASS(FACILITY) ID(VSMWORK2) ACCESS(UPDATE)
RAC PERMIT ICHCONN CLASS(FACILITY) ID(VSMWORK3) ACCESS(UPDATE)
RAC SETROPTS RACLIST(FACILITY) REFRESH
```

Note: These commands may fail if they have already been issued before.

© Copyright IBM Corp. 2003, 2025

The directory entry for the SMAPI service machines using this capability must all contain this statement:

```
IUCV ANY PRIORITY MSGLIMIT 255
```

Note: A MSGLIMIT value of 255 is initially suggested. It may be adjusted as your experience dictates.

Note each SMAPI service machine has access to the RACF SERVMACH file, which is located on MAINT's 19E disk. This file identifies which RACFVM service machine RACROUTE requests will be sent to.

Making the SMAPI Service Machines Exempt From Certain Command Checking

The SMAPI service machines, (VSMWORK1, VSMWORK2, VSMWORK3), DTCSMAPI and MAINT, should be made exempt from access checking. Even if access checking is not active on your system, make the SMAPI service machines exempt from access checking for the FOR (privilege class C), and LINK commands:

```
RAC SETROPTS CLASSACT(VMXEVENT)

RAC RDEFINE VMXEVENT USERSEL.DTCSMAPI
RAC RALTER  VMXEVENT USERSEL.DTCSMAPI ADDMEM(FOR.C/NOCTL)
RAC RALTER  VMXEVENT USERSEL.DTCSMAPI ADDMEM(LINK/NOCTL)
RAC SETEVENT REFRESH USERSEL.DTCSMAPI

RAC RDEFINE VMXEVENT USERSEL.VSMWORK1
RAC RALTER  VMXEVENT USERSEL.VSMWORK1 ADDMEM(FOR.C/NOCTL)
RAC RALTER  VMXEVENT USERSEL.VSMWORK1 ADDMEM(LINK/NOCTL)
RAC SETEVENT REFRESH USERSEL.VSMWORK1

RAC RDEFINE VMXEVENT USERSEL.VSMWORK2
RAC RALTER  VMXEVENT USERSEL.VSMWORK2 ADDMEM(FOR.C/NOCTL)
RAC RALTER  VMXEVENT USERSEL.VSMWORK2 ADDMEM(LINK/NOCTL)
RAC SETEVENT REFRESH USERSEL.VSMWORK2

RAC RDEFINE VMXEVENT USERSEL.VSMWORK3
RAC RALTER  VMXEVENT USERSEL.VSMWORK3 ADDMEM(FOR.C/NOCTL)
RAC RALTER  VMXEVENT USERSEL.VSMWORK3 ADDMEM(LINK/NOCTL)
RAC SETEVENT REFRESH USERSEL.VSMWORK3


RAC RDEFINE VMXEVENT USERSEL.MAINT
RAC RALTER  VMXEVENT USERSEL.MAINT ADDMEM(FOR.C/NOCTL)
RAC RALTER  VMXEVENT USERSEL.MAINT ADDMEM(LINK/NOCTL)
RAC SETEVENT REFRESH USERSEL.MAINT
```

Note:

- These commands may fail if they have been issued previously.
- The RAC SETEVENT REFRESH commands shown above will fail with the following error message:

  ```
  RPISET133E SETEVENT FAILED. USER IS NOT CURRENTLY LOGGED ON.
  ```

  if the user ID is not currently logged on. This is acceptable, as the user ID will refresh the next time it is logged on. To view the list of events, enter:

  ```
  RAC SETEVENT LIST USERSEL.DTCSMAPI
  ```

  These commands will also fail with the error message:

  ```
  RPISET133E SETEVENT FAILED. USER IS NOT CURRENTLY LOGGED ON.
  ```

  if the user ID is not logged on. Log on the user ID and reissue the command if you wish to see this information.
- You should consider auditing LINK and FOR.C requests -- the default is NOAUDIT. To enable auditing of the requests, RALTER each VMXEVENT profile above with the ADDMEM(AUDIT) keyword from a user authorized to control auditing, then REFRESH the profile. Depending upon your organization's

separation of duties with regard to security policies, you might need to have a different person enable auditing.

Enabling SMAPI to Access DIAGNOSE X'88'

You must enable the SMAPI service machines for DIAGNOSE X'88' access. If RACF is being used to control DIAGNOSE X'88' access, enable DIAGNOSE X'88' access for SMAPI by completing the following steps:

Step 1. Enable RACF/VM profile protection for DIAGNOSE X'88':

1. Create a profile called DIAG088 in the VMCMD class with a default access of NONE:

```
RAC RDEFINE VMCMD DIAG088 UACC(NONE)
```

2. Ensure that the VMCMD class is active:

```
RAC SETROPTS CLASSACT(VMCMD)
```

Note: Each SMAPI server has the OPTION DIAG88 statement in its directory entry. If you do not enable RACF protection, the checking defaults to the CP directory OPTION DIAG88 entry, which tells CP that the server is authorized to use DIAGNOSE code X'88'.

Step 2. Give the SMAPI server permission to perform password validation (which uses DIAGNOSE X'88' subcode 8):

1. Give authority to the following request servers: VSMREQIN, VSMREQI6, VSMREQIU, and VSMEVSRV.

```
RAC PERMIT DIAG088 CLASS(VMCMD) ID(VSMREQIN) ACCESS(READ)
RAC PERMIT DIAG088 CLASS(VMCMD) ID(VSMREQI6) ACCESS(READ)
RAC PERMIT DIAG088 CLASS(VMCMD) ID(VSMREQIU) ACCESS(READ)
RAC PERMIT DIAG088 CLASS(VMCMD) ID(VSMEVSRV) ACCESS(READ)
```

2. Give authority to the worker servers: VSMGUARD, VSMWORK1, VSMWORK2, and VSMWORK3.

```
RAC PERMIT DIAG088 CLASS(VMCMD) ID(VSMGUARD) ACCESS(READ)
RAC PERMIT DIAG088 CLASS(VMCMD) ID(VSMWORK1) ACCESS(READ)
RAC PERMIT DIAG088 CLASS(VMCMD) ID(VSMWORK2) ACCESS(READ)
RAC PERMIT DIAG088 CLASS(VMCMD) ID(VSMWORK3) ACCESS(READ)
```

3. Give authority to these SMAPI user IDs: LOHCOST, DTCSMAPI PERSMAPI and OPERATNS.

```
RAC PERMIT DIAG088 CLASS(VMCMD) ID(LOHCOST) ACCESS(READ)
RAC PERMIT DIAG088 CLASS(VMCMD) ID(DTCSMAPI) ACCESS(READ)
RAC PERMIT DIAG088 CLASS(VMCMD) ID(PERSMAPI) ACCESS(READ)
RAC PERMIT DIAG088 CLASS(VMCMD) ID(OPERATNS) ACCESS(READ)
```

For more information, see *z/VM: RACF Security Server Security Administrator's Guide*.

Enabling SMAPI to Access Needed Resources

You must enable the SMAPI service machine for minidisk, reader, and/or VMBATCH access.

Minidisk Access

If RACF is being used to control minidisk access:

```
RAC PERMIT MAINT630.5E5 CLASS(VMMDISK) ID(VSMWORK1) ACCESS(READ)
RAC PERMIT MAINT630.51D CLASS(VMMDISK) ID(VSMWORK1) ACCESS(READ)
RAC PERMIT PMAINT.551   CLASS(VMMDISK) ID(VSMGUARD) ACCESS(READ)
```

Allow VSMWORK1 minidisk authority to the following:

```
RAC PERMIT PMAINT.CF0 CLASS(VMMDISK) ACC(CONTROL) ID(VSMWORK1)
RAC PERMIT MAINT.CF1  CLASS(VMMDISK) ACC(CONTROL) ID(VSMWORK1)
```

Allow SMAPI worker servers to read the TCPMAINT 198 disk:

```
RAC PERMIT TCPMAINT.198 CLASS(VMMDISK) ACC(READ) ID(VSMGUARD)
RAC PERMIT TCPMAINT.198 CLASS(VMMDISK) ACC(READ) ID(VSMWORK1)
RAC PERMIT TCPMAINT.198 CLASS(VMMDISK) ACC(READ) ID(VSMWORK2)
RAC PERMIT TCPMAINT.198 CLASS(VMMDISK) ACC(READ) ID(VSMWORK3)
```

Reader Access

If RACF is being used to control reader access, enable reader access to DTCSMAPI for the MAINT and TCPMAINT user IDs:

```
RAC PERMIT MAINT CLASS(VMRDR) ID(DTCSMAPI) ACCESS(UPDATE)
RAC PERMIT TCPMAINT CLASS(VMRDR) ID(DTCSMAPI) ACCESS(UPDATE)
```

If RACF is being used to control reader access, and the Directory Maintenance Facility (DirMaint) is being used as your directory manager, enable reader access to VSMWORK2 and VSMWORK3 for the DIRMAINT user ID:

```
RAC PERMIT DIRMAINT CLASS(VMRDR) ID(VSMWORK2) ACCESS(UPDATE)
RAC PERMIT DIRMAINT CLASS(VMRDR) ID(VSMWORK3) ACCESS(UPDATE)
```

VMBATCH Access

Permit the SMAPI servers CONTROL access to a generic VMBATCH, or else to an existing discrete VMBATCH profile to use the SMAPI services:

- To give CONTROL access if you have an existing generic VMBATCH profile:

```
RAC PERMIT ** CLASS(VMBATCH) ID(VSMWORK1) ACCESS(CONTROL)
RAC PERMIT ** CLASS(VMBATCH) ID(VSMWORK2) ACCESS(CONTROL)
RAC PERMIT ** CLASS(VMBATCH) ID(VSMWORK3) ACCESS(CONTROL)
RAC PERMIT ** CLASS(VMBATCH) ID(DTCSMAPI) ACCESS(CONTROL)
```

- To give CONTROL authority using the discrete VMBATCH profile:

```
RAC PERMIT CLASS(VMBATCH) ID(VSMWORK1) ACCESS(CONTROL)
RAC PERMIT CLASS(VMBATCH) ID(VSMWORK2) ACCESS(CONTROL)
RAC PERMIT CLASS(VMBATCH) ID(VSMWORK3) ACCESS(CONTROL)
RAC PERMIT CLASS(VMBATCH) ID(DTCSMAPI) ACCESS(CONTROL)
```

Migrating to Using the ESM Policies for Authorizing APIs

You need the information in this section only when you are using an ESM to authorize SMAPI requests. See “Configuring SMAPI” on page 30 for additional information.

Note that the ESM profile name structure is the following:

```
SMAPI_Instance_Name.target.SMAPI_exec_name.sysid
```

A REXX EXEC, DMSAPISL EXEC, is provided on the MAINT 193 disk to translate, in either direction, between the SMAPI EXEC name, such as DMSSSFLQ EXEC, and the corresponding API name, such as Query_API_Functional_Level. See “DMSAPISL” on page 887 for more information.

IBM recommends that you initially use the default authorization policy. If you are starting with no applicable ESM profiles, then for example when your ESM uses an unshared security database, this policy will cause the existing SMAPI authorization method to handle all requests. However, SMAPI will call the ESM to record the results of all authorization decisions that SMAPI makes as general event security audit log records. SMAPI always records the results of authorization decisions, regardless of the authorization policy in effect, in SMAPI's log files. For more information, see the description of the Server Log Level property in “SMAPI Configuration Properties” on page 30.

You can use the audit records created to help you set up your ESM profiles, which will allow you to eventually have the ESM control authorization decisions. Until you are ready to switch to full ESM control,

avoid creating profiles that include SMAPI.** or SMAPI.*.*.*sysid*. Only add profiles containing wildcards after ensuring that Generics are enabled for the FACILITY class. For more information, see the description of the SETROPTS command in *z/VM: RACF Security Server Command Language Reference*.

The DMSAPISD EXEC and the DMSAPISL EXEC can be used to help you set up your profiles. For more information, see "DMSAPISD" on page 886 and "DMSAPISL" on page 887. SMAPI uses general purpose CMS services when it calls the ESM. For more information, see *z/VM: CMS Macros and Functions Reference*.

Appendix G. Capturing SMAPI Data for Problem Resolution

SMSTATUS is a special stand-alone EXEC that captures data regarding the status of the various SMAPI servers and system settings that are useful for investigating suspected problems involving SMAPI. Use it to perform the same function as SMAPI_Status_Capture when that API cannot be executed because SMAPI is not responsive. See "SMAPI_Status_Capture" on page 511 for more information on that API.

To use this EXEC, follow these steps:

1. The SMSTATUS EXEC is designed to be run by MAINT. To run the exec:
   a. Log on as MAINT.
   b. Access the `vmsys:vsmwork1.data` directory.
   c. Access MAINT's 193 disk. It must be accessed in your search order **after** the `vmsys:vsmwork1.data` directory.
   d. Enter SMSTATUS.
2. Running SMSTATUS may prompt you for a password, in order to test that the directory manager is configured correctly. You will be prompted to check if you are in a VMREAD state. If you are, then enter your logon password to continue.
3. When the SMSTATUS EXEC completes, there will be an output file created in the VMSYS:VSMWORK1.STATUS directory, as specified by the `Server_STATUS` = attribute in the DMSSICNF COPY file. Note the following about this file:
   - The EXEC itself will indicate the name and location of this file.
   - It will be a COPYFILE PACKED text file.
   - Do not attempt to further compress or pack this file. The SMSTATUS file must be downloaded to your workstation as a BINARY FIXED LRECL 1024 file.
   - Once downloaded, the file can be sent to IBM Service to assist with diagnosing suspected problems.
   - SMAPI will retain the *n* most recent output files from invocations of SMSTATUS, where *n* is determined by the `Server_StatusLog_Max` = attribute.

   See "Configuring SMAPI" on page 30 for more information.

The return and reason codes returned by the SMSTATUS EXEC are as follows:

RC	RC Name	RS	RS Name	Description
0	RC_OK	0	RS_NONE	Request successful
8	RC_ERR	14	RS_FREE_MODE_NOT_AVAIL	Free modes not available
		3015	RS_FILE_SAVE_ERROR	File could not be saved
396	RCERR_INTERNAL	*nnnn*	*psrc*	Internal system error - product-specific return code (See "Internal Return Codes (RC = 396, 592, or 596)" on page 834)

Usage Notes

- SMSTATUS does not clear or rotate logs after it collects them.

© Copyright IBM Corp. 2003, 2025

- If you are running an External Security Manager (ESM), SMSTATUS can fail to collect console logs even if you configured SMAPI as indicated in Appendix F, “Using SMAPI with an External Security Manager,” on page 877. This failure will be noted in the SMSTATUS output.
- The SMSTATUS output for some ESM-related problems will be incomplete. To diagnose a problem related to an ESM, you might need to provide all relevant profiles, all group membership for groups authorized by those profiles, and the list of all groups that any user listed in those profiles (either directly or via another group) is a member of. One way to provide this information is via a data base dump.

Appendix H. Utilities and Common Procedures

© Copyright IBM Corp. 2003, 2025

DMSAPISD

The DMSAPISD EXEC checks for duplicate names between an API and a EXEC, including those in SMAPI's API definitions as well as any user-defined APIs you created. (See Appendix B, “Creating Custom APIs,” on page 851.) It resides on the MAINT 193 disk.

Input

The input to DMSAPISD is a single parameter, and can be:

- Nothing (produces no output if no duplicates are found)
- Any value (produces output for each API to EXEC mapping)

Output

The return code is the count of EXECs that implement more than one API.

- RC=0 indicates that no duplicate API and EXEC matches were found
- RC=*nn* indicates that *nn* duplicates were found. A message is displayed for each duplicate.

DMSAPISL

The DMSAPISL EXEC lists the EXEC/documented API/ESM profile mappings. It resides on the MAINT 193 disk. When creating new ESM profiles to restrict access to a particular API, you pass DMSAPISL EXEC the API name you want to control and it provides the EXEC name to use in the ESM profile. To determine what existing profiles are used for, you pass DMSAPISL EXEC the EXEC name to find out the corresponding API.

Input

The following positional parameters are the input to the DMSAPISL EXEC:

- *prefix* or * (optional; used to match the SMAPI API or EXEC name)
- `esm` (optional; `esm` causes the ESM profile name and the API name to be displayed. Otherwise, the API and EXEC name are displayed.)

Output

- RC=0 indicates there were no errors, and the appropriate output is displayed.
- RC>0 indicates an error.

DMSAPISP

The DMSAPISP EXEC retrieves the SMAPI API configuration mappings (API, EXEC name, ESM profile) from the DMSSIPRM NAMES file and the DMSSIUSR NAMES file. This EXEC is used by the DMSAPISL EXEC and the DMSAPISD EXEC.

Input

The following parameters are the input to the DMSAPISP EXEC:

- The REXX stem variable name
- Trace flag (optional)

Output

- Return code
- Stem variable name with this suffix: `.0` (the number of values in the `name`, `exec`, and `profile` stem variables below)
- Stem name variable with this suffix: `.0name.` (the SMAPI API name)
- Stem name variable with this suffix: `.0exec.` (the SMAPI EXEC name called for this API)
- Stem name variable with this suffix: `.0profile.` (the ESM profile name for this API)

For example, if the input stem name is "esm" and the files it examines define only one mapping, between the API named "TestAPI" and an EXEC named "TestEXEC", that mapping results in the following variables being set in the caller's variable pool. (This example assumes that the SMAPI instance name is "SMAPI" and the DMSAPISP EXEC is run on a z/VM system with a system ID of "VM001".)

```
esm.0=1
esm.0name.1 = "TestAPI"
esm.0exec.1 = "TestEXEC"
esm.0profile.1 = "SMAPI.targetvm.TestEXEC.VM001"
```

SMCFGDM EXEC

Purpose

The SMCFGDM EXEC configures DirMaint so that DirMaint is in sync with SMAPI. The newly configured DirMaint is then ready to be used as the SMAPI directory manager.

Synopsis

```
SBCFGDM
```

Return Value

0
The EXEC ran successfully.

8
The EXEC ran unsuccessfully. Error information is sent to SMCFGDM ERRLOG A.

Notes

1. The EXEC writes a console file containing processing information to the reader of the user ID issuing the EXEC.
2. The EXEC must be issued from a user ID that has the following authorizations with respect to the installed DirMaint instance:

```
CMDSET ADGHMOPS CMDL 140A
CMDSET ADGHMOPS CMDL 150A
ALLOW_ASUSER_NOPASS
```

Notices

This information was developed for products and services offered in the US. This material might be available from IBM in other languages. However, you may be required to own a copy of the product or product version in that language in order to access it.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive, MD-NC119
Armonk, NY 10504-1785
US

For license inquiries regarding double-byte character set (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan Ltd.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 103-8510, Japan

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some jurisdictions do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

IBM may use or distribute any of the information you provide in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Director of Licensing
IBM Corporation
North Castle Drive, MD-NC119
Armonk, NY 10504-1785
US

© Copyright IBM Corp. 2003, 2025

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

The performance data and client examples cited are presented for illustrative purposes only. Actual performance results may vary depending on specific configurations and operating conditions.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

Statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information may contain examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to actual people or business enterprises is entirely coincidental.

COPYRIGHT LICENSE:

This information may contain sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

Programming Interface Information

This book documents intended Programming Interfaces that allow the customer to write programs to obtain services of z/VM.

Trademarks

IBM, the IBM logo, and ibm.com® are trademarks or registered trademarks of International Business Machines Corp., in the United States and/or other countries. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on IBM Copyright and trademark information (https://www.ibm.com/legal/copytrade).

The registered trademark Linux is used pursuant to a sublicense from the Linux Foundation, the exclusive licensee of Linus Torvalds, owner of the mark on a worldwide basis.

Terms and Conditions for Product Documentation

Permissions for the use of these publications are granted subject to the following terms and conditions.

Applicability

These terms and conditions are in addition to any terms of use for the IBM website.

Personal Use

You may reproduce these publications for your personal, noncommercial use provided that all proprietary notices are preserved. You may not distribute, display or make derivative work of these publications, or any portion thereof, without the express consent of IBM.

Commercial Use

You may reproduce, distribute and display these publications solely within your enterprise provided that all proprietary notices are preserved. You may not make derivative works of these publications, or reproduce, distribute or display these publications or any portion thereof outside your enterprise, without the express consent of IBM.

Rights

Except as expressly granted in this permission, no other permissions, licenses or rights are granted, either express or implied, to the publications or any information, data, software or other intellectual property contained therein.

IBM reserves the right to withdraw the permissions granted herein whenever, in its discretion, the use of the publications is detrimental to its interest or, as determined by IBM, the above instructions are not being properly followed.

You may not download, export or re-export this information except in full compliance with all applicable laws and regulations, including all United States export laws and regulations.

IBM MAKES NO GUARANTEE ABOUT THE CONTENT OF THESE PUBLICATIONS. THE PUBLICATIONS ARE PROVIDED "AS-IS" AND WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, AND FITNESS FOR A PARTICULAR PURPOSE.

IBM Online Privacy Statement

IBM Software products, including software as a service solutions, ("Software Offerings") may use cookies or other technologies to collect product usage information, to help improve the end user experience, to tailor interactions with the end user, or for other purposes. In many cases no personally identifiable information is collected by the Software Offerings. Some of our Software Offerings can help enable you to collect personally identifiable information. If this Software Offering uses cookies to collect personally identifiable information, specific information about this offering's use of cookies is set forth below.

This Software Offering does not use cookies or other technologies to collect personally identifiable information.

If the configurations deployed for this Software Offering provide you as customer the ability to collect personally identifiable information from end users via cookies and other technologies, you should seek your own legal advice about any laws applicable to such data collection, including any requirements for notice and consent.

For more information about the use of various technologies, including cookies, for these purposes, see:

- The section entitled **IBM Websites** at IBM Privacy Statement (https://www.ibm.com/privacy)
- Cookies and Similar Technologies (https://www.ibm.com/privacy#Cookies_and_Similar_Technologies)

Bibliography

This topic lists the publications in the z/VM library. For abstracts of the z/VM publications, see *z/VM: General Information*.

Where to Get z/VM Information

The current z/VM product documentation is available in IBM Documentation - z/VM (https://www.ibm.com/docs/en/zvm).

z/VM Base Library

Overview

- *z/VM: License Information*, GI13-4377
- *z/VM: General Information*, GC24-6286

Installation, Migration, and Service

- *z/VM: Installation Guide*, GC24-6292
- *z/VM: Migration Guide*, GC24-6294
- *z/VM: Service Guide*, GC24-6325
- *z/VM: VMSES/E Introduction and Reference*, GC24-6336

Planning and Administration

- *z/VM: CMS File Pool Planning, Administration, and Operation*, SC24-6261
- *z/VM: CMS Planning and Administration*, SC24-6264
- *z/VM: Connectivity*, SC24-6267
- *z/VM: CP Planning and Administration*, SC24-6271
- *z/VM: Getting Started with Linux on IBM Z*, SC24-6287
- *z/VM: Group Control System*, SC24-6289
- *z/VM: I/O Configuration*, SC24-6291
- *z/VM: Running Guest Operating Systems*, SC24-6321
- *z/VM: Saved Segments Planning and Administration*, SC24-6322
- *z/VM: Secure Configuration Guide*, SC24-6323

Customization and Tuning

- *z/VM: CP Exit Customization*, SC24-6269
- *z/VM: Performance*, SC24-6301

Operation and Use

- *z/VM: CMS Commands and Utilities Reference*, SC24-6260
- *z/VM: CMS Primer*, SC24-6265
- *z/VM: CMS User's Guide*, SC24-6266
- *z/VM: CP Commands and Utilities Reference*, SC24-6268

© Copyright IBM Corp. 2003, 2025

- *z/VM: System Operation*, SC24-6326
- *z/VM: Virtual Machine Operation*, SC24-6334
- *z/VM: XEDIT Commands and Macros Reference*, SC24-6337
- *z/VM: XEDIT User's Guide*, SC24-6338

Application Programming

- *z/VM: CMS Application Development Guide*, SC24-6256
- *z/VM: CMS Application Development Guide for Assembler*, SC24-6257
- *z/VM: CMS Application Multitasking*, SC24-6258
- *z/VM: CMS Callable Services Reference*, SC24-6259
- *z/VM: CMS Macros and Functions Reference*, SC24-6262
- *z/VM: CMS Pipelines User's Guide and Reference*, SC24-6252
- *z/VM: CP Programming Services*, SC24-6272
- *z/VM: CPI Communications User's Guide*, SC24-6273
- *z/VM: ESA/XC Principles of Operation*, SC24-6285
- *z/VM: Language Environment User's Guide*, SC24-6293
- *z/VM: OpenExtensions Advanced Application Programming Tools*, SC24-6295
- *z/VM: OpenExtensions Callable Services Reference*, SC24-6296
- *z/VM: OpenExtensions Commands Reference*, SC24-6297
- *z/VM: OpenExtensions POSIX Conformance Document*, GC24-6298
- *z/VM: OpenExtensions User's Guide*, SC24-6299
- *z/VM: Program Management Binder for CMS*, SC24-6304
- *z/VM: Reusable Server Kernel Programmer's Guide and Reference*, SC24-6313
- *z/VM: REXX/VM Reference*, SC24-6314
- *z/VM: REXX/VM User's Guide*, SC24-6315
- *z/VM: Systems Management Application Programming*, SC24-6327
- *z/VM: z/Architecture Extended Configuration (z/XC) Principles of Operation*, SC27-4940

Diagnosis

- *z/VM: CMS and REXX/VM Messages and Codes*, GC24-6255
- *z/VM: CP Messages and Codes*, GC24-6270
- *z/VM: Diagnosis Guide*, GC24-6280
- *z/VM: Dump Viewing Facility*, GC24-6284
- *z/VM: Other Components Messages and Codes*, GC24-6300
- *z/VM: VM Dump Tool*, GC24-6335

z/VM Facilities and Features

Data Facility Storage Management Subsystem for z/VM

- *z/VM: DFSMS/VM Customization*, SC24-6274
- *z/VM: DFSMS/VM Diagnosis Guide*, GC24-6275
- *z/VM: DFSMS/VM Messages and Codes*, GC24-6276
- *z/VM: DFSMS/VM Planning Guide*, SC24-6277

- *z/VM: DFSMS/VM Removable Media Services*, SC24-6278
- *z/VM: DFSMS/VM Storage Administration*, SC24-6279

Directory Maintenance Facility for z/VM

- *z/VM: Directory Maintenance Facility Commands Reference*, SC24-6281
- *z/VM: Directory Maintenance Facility Messages*, GC24-6282
- *z/VM: Directory Maintenance Facility Tailoring and Administration Guide*, SC24-6283

Open Systems Adapter

- Open Systems Adapter/Support Facility on the Hardware Management Console (https://www.ibm.com/docs/en/SSLTBW_2.3.0/pdf/SC14-7580-02.pdf), SC14-7580
- Open Systems Adapter-Express ICC 3215 Support (https://www.ibm.com/docs/en/zos/2.3.0?topic=osa-icc-3215-support), SA23-2247
- Open Systems Adapter Integrated Console Controller User's Guide (https://www.ibm.com/docs/en/SSLTBW_2.3.0/pdf/SC27-9003-02.pdf), SC27-9003
- Open Systems Adapter-Express Customer's Guide and Reference (https://www.ibm.com/docs/en/SSLTBW_2.3.0/pdf/ioa2z1f0.pdf), SA22-7935

Performance Toolkit for z/VM

- *z/VM: Performance Toolkit Guide*, SC24-6302
- *z/VM: Performance Toolkit Reference*, SC24-6303

The following publications contain sections that provide information about z/VM Performance Data Pump, which is licensed with Performance Toolkit for z/VM.

- *z/VM: Performance*, SC24-6301. See z/VM Performance Data Pump.
- *z/VM: Other Components Messages and Codes*, GC24-6300. See Data Pump Messages.

RACF® Security Server for z/VM

- *z/VM: RACF Security Server Auditor's Guide*, SC24-6305
- *z/VM: RACF Security Server Command Language Reference*, SC24-6306
- *z/VM: RACF Security Server Diagnosis Guide*, GC24-6307
- *z/VM: RACF Security Server General User's Guide*, SC24-6308
- *z/VM: RACF Security Server Macros and Interfaces*, SC24-6309
- *z/VM: RACF Security Server Messages and Codes*, GC24-6310
- *z/VM: RACF Security Server Security Administrator's Guide*, SC24-6311
- *z/VM: RACF Security Server System Programmer's Guide*, SC24-6312
- *z/VM: Security Server RACROUTE Macro Reference*, SC24-6324

Remote Spooling Communications Subsystem Networking for z/VM

- *z/VM: RSCS Networking Diagnosis*, GC24-6316
- *z/VM: RSCS Networking Exit Customization*, SC24-6317
- *z/VM: RSCS Networking Messages and Codes*, GC24-6318
- *z/VM: RSCS Networking Operation and Use*, SC24-6319
- *z/VM: RSCS Networking Planning and Configuration*, SC24-6320

TCP/IP for z/VM

- *z/VM: TCP/IP Diagnosis Guide*, GC24-6328
- *z/VM: TCP/IP LDAP Administration Guide*, SC24-6329
- *z/VM: TCP/IP Messages and Codes*, GC24-6330
- *z/VM: TCP/IP Planning and Customization*, SC24-6331
- *z/VM: TCP/IP Programmer's Reference*, SC24-6332
- *z/VM: TCP/IP User's Guide*, SC24-6333

Prerequisite Products

Device Support Facilities

- Device Support Facilities (ICKDSF): User's Guide and Reference (https://www.ibm.com/docs/en/SSLTBW_2.5.0/pdf/ickug00_v2r5.pdf), GC35-0033

Environmental Record Editing and Printing Program

- Environmental Record Editing and Printing Program (EREP): Reference (https://www.ibm.com/docs/en/SSLTBW_2.5.0/pdf/ifc2000_v2r5.pdf), GC35-0152
- Environmental Record Editing and Printing Program (EREP): User's Guide (https://www.ibm.com/docs/en/SSLTBW_2.5.0/pdf/ifc1000_v2r5.pdf), GC35-0151

Related Products

XL C++ for z/VM

- *XL C/C++ for z/VM: Runtime Library Reference*, SC09-7624
- *XL C/C++ for z/VM: User's Guide*, SC09-7625

z/OS

IBM Documentation - z/OS (https://www.ibm.com/docs/en/zos)

Index

Numerics

2U04386Virtual_Network_Adapter_Connect_Vswitch 10

A

ABEND dump management functions 7
activating a server 44
AF_EVNT 21
AF_INET 21
AF_IUCV 21
AF_MGMT 21
AF_SCLP 21
array 51
asynch update port 32
Asynchronous_Notification_Disable_DM 8, 56
Asynchronous_Notification_Enable_DM 8, 60
Asynchronous_Notification_Query_DM 8, 65
authorization exit 31
authorization file 36, 38
authorization functions 7
authorization list configuration 33
authorization policy 30
Authorization_List_Add 7, 70
Authorization_List_Query 7, 74
Authorization_List_Remove 7, 79
authorizing API requests 36, 38

C

C program, sample 859
call format 51
character sets 49
Check_Authentication 16, 82
client authentication 36
code, sample 859
Configuration_Read 379
creating custom APIs 851
custom APIs 34, 851
custom EXEC 852

D

data types 49
DCSS, server 32
deactivating a server 44
default SYSTEM CONFIG link values 35
defining additional servers 44
defining servers 21
Delete_ABEND_Dump 7, 85
directory manager 4
directory manager control functions 7
directory manager exit 31, 843
directory manager local tag and scan functions 8
directory profiles 15
directory updates subscription functions 8
Directory_Manager_Local_Tag_Define_DM 8, 88
Directory_Manager_Local_Tag_Delete_DM 8, 91
Directory_Manager_Local_Tag_Query_DM 8, 94
Directory_Manager_Local_Tag_Set_DM 8, 97
Directory_Manager_Search_DM 8, 101
Directory_Manager_Task_Cancel_DM 7, 105
DMSAPISD EXEC 886
DMSAPISL EXEC 887
DMSAPISP EXEC 888
DMSSICNF COPY file 30
DMSSICNF file
    properties 30
DMSSISVR NAMES file 27
DTCSMAPI 24
dump processing interval 35
dump processing location 35
dynamically activating or deactivating a worker server 44

E

ENROLL command 857
ESM policies 880
Event_Stream_Add 9, 108
Event_Subscribe 9, 111
Event_Unsubscribe 9, 115
EXECs
    SMCFGDM 889
external security manager 877
External Security Manager (ESM) 36, 880

G

Getting started 4
GRANT command 857

I

image characteristic functions 9
image connectivity 10
image CPUs 12
image devices 12
image IPL management 13
image operations 13
IMAGE RECYCLE maximum wait time 35
image volume management functions 14
Image_Activate 13, 117
Image_Active_Configuration_Query 13, 121
Image_Console_Get 126
Image_CPU_Define 12, 128
Image_CPU_Define_DM 12, 131
Image_CPU_Delete 12, 135
Image_CPU_Delete_DM 12, 138
Image_CPU_Query 12, 141
Image_CPU_Query_DM 12, 145
Image_CPU_Set_Maximum_DM 12, 149
Image_Create_DM 9, 152

Image_Deactivate 13, 157
Image_Definition_Async_Updates 8, 161
Image_Definition_Create_DM 8, 164
Image_Definition_Delete_DM 8, 175
Image_Definition_Query_DM 8, 182
Image_Definition_Update_DM 8, 190
Image_Delete_DM 9, 202
Image_Device_Dedicate 12, 205
Image_Device_Dedicate_DM 12, 208
Image_Device_Reset 12, 211
Image_Device_Undedicate 12, 214
Image_Device_Undedicate_DM 12, 217
Image_Disk_Copy 12, 220
Image_Disk_Copy_DM 12, 223
Image_Disk_Create 12, 229
Image_Disk_Create_DM 12, 233
Image_Disk_Delete 12, 240
Image_Disk_Delete_DM 12, 243
Image_Disk_Query 12, 246
Image_Disk_Share 12, 250
Image_Disk_Share_DM 12, 254
Image_Disk_Unshare 12, 258
Image_Disk_Unshare_DM 12, 261
Image_IPL_Characteristics_Define_DM 14, 264
Image_IPL_Characteristics_Query_DM 14, 273
Image_IPL_Delete_DM 13, 276
Image_IPL_Query_DM 13, 279
Image_IPL_Set_DM 13, 282
Image_Lock_DM 9, 285
Image_Lock_Query_DM 9, 288
Image_MDISK_Link_Query 12, 292
Image_Name_Query_DM 9, 296
Image_Password_Set_DM 9, 299
Image_Pause 13, 302
Image_Query_Activate_Time 13, 305
Image_Query_DM 9, 308
Image_Recycle 13, 311
Image_Replace_DM 9, 315
Image_SCSI_Characteristics_Define_DM 14, 318
Image_SCSI_Characteristics_Query_DM 14, 322
Image_Status_Query 13, 326
Image_Unlock_DM 9, 329
Image_Volume_Add 14, 332
Image_Volume_Delete 14, 338
Image_Volume_Query_DM 356
Image_Volume_Share 14, 343
Image_Volume_Space_Define_DM 14, 346
Image_Volume_Space_Define_Extended_DM 14, 350
Image_Volume_Space_Query_DM 14
Image_Volume_Space_Query_Extended_DM 14, 361
Image_Volume_Space_Remove_DM 14, 366
input interface, directory manager exit 843
installation 857
instance name 31
integer 49
introduction 3

J

java program, sample 869

L

list-directed IPL functions 14
LOHCOST 23
LOHCOST server defaults 32
long call worker 22

M

max image wait time 35
Metadata_Delete 8, 370
Metadata_Get 8, 373
Metadata_Set 8, 375
Metadata_Space_Query 8

N

name list configuration 33
name list functions 14
name lists 39
Name_List_Add 14, 382
Name_List_Destroy 14, 385
Name_List_Query 14, 388
Name_List_Remove 14, 391
Network_IP_Interface_Create 15, 394
Network_IP_Interface_Modify 15, 401
Network_IP_Interface_Query 15, 405
Network_IP_Interface_Remove 15, 413

O

OPERATNS 25
output interface, directory manager exit 850
overview 3

P

Page_or_Spool_Volume_Add 9, 417
PERSMAPI 24
problem resolution 883
Process_ABEND_Dump 7, 422
profile management 15
Profile_Create_DM 15, 425
Profile_Delete_DM 15, 428
Profile_Lock_DM 15, 431
Profile_Lock_Query_DM 15, 434
Profile_Query_DM 15, 438
Profile_Replace_DM 15, 441
Profile_Unlock_DM 15, 444
prototype management functions 15
Prototype_Create_DM 15, 447
Prototype_Delete_DM 15, 450
Prototype_Name_Query_DM 15, 453
Prototype_Query_DM 15, 456
Prototype_Replace_DM 15, 459

Q

Query_ABEND_Dump 7, 462
Query_All_DM 8, 466
Query_API_Functional_Level 16, 471
Query_Asynchronous_Operation_DM 474
Query_Directory_Manager_Level_DM 7, 477

Quick start guide 4

R

RACROUTE 877
reason codes, summarized 819
request servers 21
Response_Recovery 16, 480
return code
    24 833
    396 834
    592 842
    596 842
return codes, summarized 819
RPIVAL program name 31

S

sample code 859
server configuration file
    properties 30
server DCSS 32
server functions 16
server log file size 35
server log level 32
server names file 27
server shutdown 43
server startup 43
SFS Garbage Collection Periodic Check 34
shared memory management functions 16
Shared_Memory_Access_Add_DM 16, 483
Shared_Memory_Access_Query_DM 16, 487
Shared_Memory_Access_Remove_DM 16, 491
Shared_Memory_Create 16, 494
Shared_Memory_Delete 16, 499
Shared_Memory_Query 16, 502
Shared_Memory_Replace 16, 507
short call worker 22
shutting down the server 43
Single System Image Clusters 16
SMAPI configuration file
    properties 30
SMAPI_Status_Capture 16, 511
SMSTATUS 16
SMSTATUS EXEC 883
socket-based server environment 4
sockets overview 49
SSI Clusters 16
SSI_Query 16, 515
starting the server 43
Static_Image_Changes_Activate_DM 7, 520
Static_Image_Changes_Deactivate_DM 7, 523
Static_Image_Changes_Immediate_DM 7, 526
string 49
structure 51
syntax errors 833
SYSTEM CONFIG link values 35
System_Compliance_Information_Query 533
System_Config_Syntax_Check 17, 529
System_Disk_Accessibility 9, 535
System_Disk_Add 9, 538
System_Disk_IO_Query 9, 541
System_Disk_Query 9, 546
System_EQID_Query 9, 550
System_FCP_EQID_Set 554
System_FCP_Free_Query 9, 558
System_Image_Performance_Query 13, 562
System_Information_Query 17, 565
System_Page_Utilization_Query 17, 569
System_Performance_Information_Query 17, 573
System_Performance_Threshold_Disable 9, 580
System_Performance_Threshold_Enable 9, 583
System_Processor_Query 17, 586
System_RDR_File_Manage 17, 589
System_RDR_File_Query 17, 592
System_SCSI_Disk_Add 9, 595
System_SCSI_Disk_Delete 9, 599
System_SCSI_Disk_Query 9, 602
System_Service_Query 17, 606
System_Shutdown 17, 610
System_Spool_Utilization_Query 17, 614
System_WWPN_Query 9, 618

T

TCP/IP requirements 36
temporary virtual device number and access mode 35

V

virtual machine reader operations 17
Virtual_Channel_Connection_Create 10, 622
Virtual_Channel_Connection_Create_DM 10, 625
Virtual_Channel_Connection_Delete 10, 628
Virtual_Channel_Connection_Delete_DM 10, 631
Virtual_Network_Adapter_Connect_LAN 10, 634
Virtual_Network_Adapter_Connect_LAN_DM 10, 638
Virtual_Network_Adapter_Connect_Vswitch 642
Virtual_Network_Adapter_Connect_Vswitch_DM 10, 645
Virtual_Network_Adapter_Connect_Vswitch_Extended 10, 648
Virtual_Network_Adapter_Create 10, 651
Virtual_Network_Adapter_Create_DM 10, 654
Virtual_Network_Adapter_Create_Extended 10, 658
Virtual_Network_Adapter_Create_Extended_DM 10, 662
Virtual_Network_Adapter_Delete 10, 666
Virtual_Network_Adapter_Delete_DM 10, 669
Virtual_Network_Adapter_Disconnect 10, 672
Virtual_Network_Adapter_Disconnect_DM 10, 675
Virtual_Network_Adapter_Query 10, 678
Virtual_Network_Adapter_Query_Extended 682
Virtual_Network_LAN_Access 10, 690
Virtual_Network_LAN_Access_Query 10, 693
Virtual_Network_LAN_Create 10, 696
Virtual_Network_LAN_Delete 10, 700
Virtual_Network_LAN_Query 10, 703
Virtual_Network_OSA_Query 10, 707
Virtual_Network_VLAN_Query_Stats 10, 711
Virtual_Network_Vswitch_Create 10, 716
Virtual_Network_Vswitch_Create_Extended 10, 724
Virtual_Network_Vswitch_Delete 10, 731
Virtual_Network_Vswitch_Delete_Extended 10, 737
Virtual_Network_Vswitch_Query 740
Virtual_Network_VSwitch_Query 10
Virtual_Network_Vswitch_Query_Byte_Stats 749
Virtual_Network_Vswitch_Query_Extended 10, 755

Virtual_Network_Vswitch_Query_Stats 10, 769
Virtual_Network_Vswitch_Set 10, 774
Virtual_Network_Vswitch_Set_Extended 10, 782
VMRELOCATE 16, 791
VMRELOCATE_Image_Attributes 16, 796
VMRELOCATE_Modify 16, 800
VMRELOCATE_Status 16, 803
VMRM configuration 34
VMRM configuration update functions 17
VMRM_Configuration_Query 17, 807
VMRM_Configuration_Update 17, 810
VMRM_Measurement_Query 17, 814
VSMGUARD 22
VSMWORK1 22
VSMWORK2 22
VSMWORK3 22

W

worker servers 22

Product Number: 5741-A09

Printed in USA

SC24-6327-73