Program Directory for

TCP/IP for z/VM®

Level 530

Program Number 5741-A05

for Use with
z/VM Version 5 Release 3

Document Date: June 2007

GI10-0784-00

Note!

Before using this information and the product it supports, be sure to read the general information under “Notices” on page 163.

This program directory, dated June 2007, applies to TCP/IP for z/VM, level 530, Program Number 5741-A05.

A form for reader's comments appears at the back of this publication. When you send information to IBM®, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

Contents

Figures

1.0 Introduction

This program directory is intended for the system programmer responsible for program installation and maintenance. It contains information that corresponds to the material and procedures for installation and service of the following:

- TCP/IP for z/VM

Note: It is recommended that you review this program directory in its entirety before you install or service this program, then keep this document for future reference.

The program directory contains the following sections:

- 2.0, “Program Materials” on page 5 identifies basic (and optional) TCP/IP for z/VM program materials and documentation
- 3.0, “Program Support” on page 8 describes the IBM support available for TCP/IP for z/VM
- 4.0, “Program and Service Level Information” on page 11 lists APARs (program level fixes) that have been incorporated within TCP/IP for z/VM
- 5.0, “Installation Requirements and Considerations” on page 13 identifies resources and considerations for installing and using TCP/IP for z/VM
- 6.0, “Installation Instructions” on page 52 provides detailed installation instructions for TCP/IP for z/VM
- 7.0, “Service Instructions” on page 73 provides detailed servicing instructions for TCP/IP for z/VM
- Appendix A, “TCP/IP Installation, Service and Migration Utilities” on page 104 provides information about the TCP2PROD command (supplied for placing TCP/IP for z/VM files into production), its related CATALOG files, and other TCP/IP service and migration aids
- Appendix B, “Modifying TCP/IP for z/VM CATALOG Files” on page 127 provides information about how to use VMSES/E local modifications to alter TCP/IP for z/VM CATALOG files
- Appendix C, “Modifying the TCP/IP for z/VM Default Installation” on page 131 presents considerations for changes that affect the installation (and service of) TCP/IP for z/VM
- Appendix D, “Making Local Modifications to TCP/IP for z/VM Modules” on page 133 provides information to help you implement local modifications to TCP/IP for z/VM modules
- Appendix E, “Modifying TCP/IP for z/VM VMNFS Code” on page 143 provides information specific to TCP/IP NFS server module local modifications
- Appendix F, “TCP/IP for z/VM Build Lists” on page 145 provides information about the VMSES/E build lists used to maintain TCP/IP for z/VM
- Appendix G, “Moving TCP/IP for z/VM to SFS Directories” on page 147 provides instructions for changing the TCP/IP for z/VM service environment to use Shared File System directories instead of default minidisks
- Appendix H, “Copying TCP/IP for z/VM Client Code to the Y-Disk” on page 152 provides considerations and optional instructions for copying client files to the system Product Code minidisk

- Appendix I, “Managing TCP/IP Files with Unique Service Requirements” on page 155 provides information about TCP/IP files for which extenuating service considerations and procedures are applicable.

Obtaining Updated Planning Information

Before you install TCP/IP for z/VM, read 3.1, “Preventive Service Planning” on page 8. This section describes how to obtain any updates to the information and procedures presented within this program directory.

1.1 Program Description

TCP/IP (Transmission Control Protocol/Internet Protocol) enables z/VM customers to participate in a multivendor, open networking environment using the TCP/IP protocol suite for communications and interoperability. The applications included in TCP/IP provide the ability to transfer files, send mail, log on a remote host, allow access from any other TCP/IP node in the network, and perform other network client and server functions.

Transmission Control Protocol/Internet Protocol for z/VM, level 530, (TCP/IP for z/VM) contains the functions provided by TCP/IP for z/VM, level 520, and provides the following enhancements:

The TCP/IP stack has been updated to help maintain high availability in the event of an ethernet QDIO (IPv4 or IPv6) or LCS (IPv4 only)) device failure. These enhancements include:

  - **IPv4 ARP Takeover support**, which provides redundant ethernet (LCS or QDIO) adapters (two or more adapters connected to the same LAN segment) the ability to take over for each other in the event of a device failure.
  - **IPv6 Neighbor Discovery Takeover support**, which provides redundant QDIO ethernet adapters (two or more adapters connected to the same LAN segment) the ability to take over for each other in the event of a device failure.
  - **Enhanced OSA Address Table (OAT) management**, which restricts the IP addresses reported to an OSA adapter to those associated with relevant HOME statement definitions, IPv4 or IPv6 VIPA definitions, IPv4 Proxy ARP addresses, or those that arise through IP takeover actions.

- **IPv6 VIPA support** has been added, which gives hosts the ability to use the VIPA address as a target for IP traffic, allowing such traffic to be routed to a working adapter on the TCP/IP stack, should one of several applicable adapters fail.
- TCP/IP has been updated to allow both an IPv4 and IPv6 VLAN to be associated with a **QDIOETHERNET** link. Changes include:
  - acceptance of an IPv6 VLAN ID (in addition to an IPv4 VLAN ID) for the **QDIOETHERNET LINK** statement
  - **IFCONFIG** command changes that allow both an IPv4 VLAN ID and an IPv6 VLAN ID to be specified as part of the **VLAN** interface operand

- A new socket API IOCTL subcommand (**SIOCDINTERFACE**) has been introduced that allows an inactive interface to be dynamically removed from the TCP/IP server configuration. In addition, the **IFCONFIG** command has been updated to exploit this API, through support of a new **-Remove** command option.
- Addition of an System Network Management Protocol (SNMP) Sub-Agent server (**SNMPSUBA**), which extends the functionality of the SNMP agent by supporting a specific set of Management Information Base (MIB) variables. A pre-configured subagent and exit routine are provided that will supply bridge Management Information Base (BRIDGE-MIB) data, as documented in RFC 1493, for the z/VM Virtual Switch (VSWITCH).
- Addition of a Lightweight Directory Access Protocol (LDAP) server (**LDAPSRV**) which provides client access to data maintained in an LDAP directory. An LDAP directory provides an easy way to maintain directory information in a central location for storage, update, retrieval, and exchange.

  For more detailed information about what functions have been implemented in the initial version of the z/VM LDAP server, see *TCP/IP LDAP Administration Guide* (SC24-6140).
- Secure Sockets Layer/Transport Layer Security (SSL/TLS) support for secure FTP (RFC 4217), Telnet (draft specification #6), and SMTP (RFC 3207) sessions. With this support, APIs are provided that permit a Pascal or Assembler client or server application to control the acceptance and establishment of TCP/IP sessions encrypted using SSL/TLS.

  The TCP/IPservers and applications enhanced with, or to support, this capability are the:
  - TCP/IP server
  - SSL server
  - FTP server
  - FTP client
  - Telnet server (Internal to the TCP/IP server)
  - Telnet client
  - SMTP server
- Updated SSL RPM packages, which support running the SSL server using these Linux distributions:
  - SUSE SLES9 Service Pack 3 (31-bit)
  - SUSE SLES9 Service Pack 3 (64-bit)
  - Red Hat Enterprise Linux AS (Version 4 U4) (31-bit)
  - Red Hat Enterprise Linux AS (Version 4 U4) (64-bit)
- SSL Server changes that allow additional security levels to be exempted from use, to more easily eliminate weak ciphers, and which allow the SSL server Linux guest to remain active after a critical error is encountered during server operations.
- Updates to the SSL server administrative command (**SSLADMIN**) that:
  - allow the specification of the number of days that a self-signed certificate is to be valid
  - improve the management of SSL server LOG files, by providing the ability to:
    - maintain log information in a file named other than SSLADMIN LOG
    - specify a maximum size to be established for the SSL server log
    - purge log information accumulated by the SSL server.

Note: Support for the BOOTPD and RouteD servers has been withdrawn from TCP/IP for z/VM, level 530.

2.0 Program Materials

An IBM program is identified by a program number. The program number for TCP/IP for z/VM is 5741-A05.

The z/VM version 5 release 3 program announcement material provides detailed information about features supported by TCP/IP for z/VM. If you have not already received a copy of this information, contact your IBM marketing representative.

The following sections identify:

- basic and optional program materials that are applicable to this program
- publications useful during installation and service.

2.1 Basic Machine-Readable Material

TCP/IP for z/VM is distributed as part of the z/VM version 5 release 3 System Deliverable. Refer to the *IBM z/VM V5.3 - Improving scalability, security, and virtualization technology* Software Announcement, 207-019, for information about ordering z/VM version 5 release 3 and its features.

2.2 Optional Machine-Readable Material

There are no features or optional machine-readable materials associated with TCP/IP for z/VM.

2.3 Program Publications

The following sections identify the basic publications associated with TCP/IP for z/VM. There are no optional publications for this component of z/VM.

2.3.1 Basic Program Publications

Publications associated with TCP/IP for z/VM are listed in Figure 1:

Figure 1 (Page 1 of 2). Basic Material: Unlicensed Publications

Publication Title	Form Number
TCP/IP Planning and Customization	SC24-6125
TCP/IP LDAP Administration Guide	SC24-6140
TCP/IP User's Guide	SC24-6127
TCP/IP Messages and Codes	GC24-6124

Figure 1 (Page 2 of 2). Basic Material: Unlicensed Publications

Publication Title	Form Number
TCP/IP Programmer's Reference	SC24-6126
TCP/IP Diagnosis Guide	GC24-6123

One copy of the following program publication is provided at no charge to licencees of TCP/IP for z/VM:

- TCP/IP Planning and Customization (SC24-6125)

2.3.2 Softcopy Publications

TCP/IP for z/VM publications are supplied in softcopy form as part of the *IBM Online Library: z/VM Collection*, using both BookManager® and Adobe™ Portable Document Format (PDF) file formats. One copy of the *IBM Online Library: z/VM Collection* CD-ROM and DVD is included when you order the basic materials for z/VM version 5 release 3.

In addition, TCP/IP for z/VM softcopy publications, including this program directory, are available as Adobe PDF files at the TCP/IP for z/VM home page on the World Wide Web. The URL for this home page is:

www.vm.ibm.com/related/tcpip/

z/VM publications also can be separately ordered through the IBM Publications Center (for a fee), by using specific publication numbers. The URL for the IBM Publications Center is:

www.ibm.com/shop/publications/order

The IBM Publications Center is a world wide central repository for IBM product publications and marketing material. Note that a large number of publications are available as on-line files (in various formats, such as Adobe PDF), which currently can be downloaded free of charge.

2.4 Program Source Materials

No viewable program listings are provided for TCP/IP for z/VM.

2.5 Publications Useful During Installation and Service

The publications listed in Figure 2 may be useful during the installation and servicing of TCP/IP for z/VM. To obtain copies of these publications, contact your IBM representative or access the IBM Publications Center on the World Wide Web; the URL for the home page of this site is:

www.ibm.com/shop/publications/order

Figure 2. Publications Useful During Installation / Service on z/VM version 5 release 3

Publication Title	Form Number
TCP/IP Planning and Customization	SC24-6125
TCP/IP LDAP Administration Guide	SC24-6140
TCP/IP User's Guide	SC24-6127
z/VM: Guide for Automated Installation and Service	GC24-6099
z/VM: Service Guide	GC24-6117
z/VM: VMSES/E Introduction and Reference	GC24-6130
z/VM: CMS Planning and Administration	SC24-6078
z/VM: CMS File Pool Planning, Administration, and Operation	SC24-6074
z/VM: CP Planning and Administration	SC24-6083
C/C++ for z/VM Run-Time Library Reference	SC09-7624
z/VM: CMS Callable Services Reference	SC24-6072
z/VM: CMS Commands and Utilities Reference	SC24-6073
z/VM: REXX/VM Reference	SC24-6113
z/VM: CMS and REXX/VM Messages and Codes	GC24-6118
z/VM: Other Components Messages and Codes	GC24-6120

3.0 Program Support

This section describes the IBM support available for TCP/IP for z/VM.

3.1 Preventive Service Planning

Before you install TCP/IP for z/VM, check with your IBM Support Center or use IBMLink™ to determine if Preventive Service Planning (PSP) information is available that you should know. To obtain this information, specify the appropriate UPGRADE and SUBSET values listed in Figure 3:

Figure 3. PSP Upgrade and Subset ID

RETAIN™				
COMPID	**Release**	**Upgrade**	**Subset**	**Component Name**
5735FAL00	530	TCPIP530	VM530	TCP/IP for z/VM
5735FAL00	530	TCPIP530	*yynn*RSU	RSU Service Recommendations

RSU-BY-LVL information can be obtained from the VM service RSU web site at this URL:
http://www.ibm.com/eserver/zseries/zvm/service/rsu

3.2 Statement of Support Procedures

With TCP/IP for z/VM, you are entitled to support under the basic warranty for z/VM version 5 release 3. Also, note that z/VM Software Subscription and Support is *automatically* added when you order z/VM — this provides zSeries service to which you are likely accustomed.

Note: You must take specific action when you order z/VM to decline z/VM Software Subscription and Support.

Report any difficulties you have using this program to your IBM Support Center. If an APAR (Authorized Program Analysis Report) is required, the Support Center will provide the address to which any needed documentation can be sent.

Figure 4 identifies IBM RETAIN information — the Component ID (COMPID), Release, and Field Engineering Service Number (FESN) — that corresponds to TCP/IP for z/VM:

Figure 4. Component IDs

RETAIN			
COMPID	**Release**	**Component Name**	**FESN**
5735FAL00	530	TCP/IP for z/VM	0461035

3.3 Service Information

The IBM Software Support Center provides telephone assistance for problem diagnosis and resolution. You can call the IBM Software Support Center at any time; you will receive a return call within eight business hours (Monday—Friday, 8:00 a.m.—5:00 p.m., local customer time). The number to call is:

1-800-426-7378 (or, **1-800-IBM-SERV**)

Outside of the United States or Puerto Rico, contact your local IBM representative or your authorized supplier.

Various installation and service-related items, such as the Preventive Service Planning (PSP) “bucket” and current RSU status/content information, are available at the TCP/IP for z/VM home page on the World Wide Web. The URL for this home page is:

www.vm.ibm.com/related/tcpip/

3.3.1 Problem Documentation

When working with TCP/IP for z/VM support personnel on problems associated with an active Problem Management Record (PMR), diagnostic information may occasionally be requested. In such cases, the support staff will work with you to determine how to best provide any requested documentation. For reference, addresses that can be used to submit various documentation formats are listed in this section.

Note: The addresses that follow should not be used as a problem reporting facility. All requests for problem assistance must be processed through the IBM Software Support Center, as previously described. Documentation submitted to any of these addresses will be reviewed only if it is associated with an active PMR.

Figure 5. Problem Documentation Addresses

Format	Address
Internet	vmtcpdoc@vnet.ibm.com
IBMLink	GDLVM7(TCPLVL2)
Carrier Service	IBM Corporation Attention: *IBM contact name* Dept. G79G 1701 North St. Endicott, NY 13760

3.3.2 Communicating Your Comments to IBM

If you have comments about or suggestions for improving the TCP/IP for z/VM program product, or this Program Directory, please provide them to IBM through one of the following means:

- If you prefer to send comments by mail, use the address provided with the Reader's Comments form (RCF) at the back of this document
- If you prefer to send comments electronically, use this Internet e-mail address:

  vmtcpip@vnet.ibm.com

If you send documentation-related comments, please include:

- the title of this publication
- the section title, section number, or topic to which your comment applies.

4.0 Program and Service Level Information

This section identifies the program level and any relevant service levels of TCP/IP for z/VM. In this context, *program level* refers to APAR fixes incorporated within the TCP/IP for z/VM program; *service level* refers to PTFs that are supplied with this product. Information about the TCP/IP for z/VM cumulative service deliverable is provided as well.

4.1 Program Level Information - TCP/IP for z/VM

APAR fixes (for previous levels of IBM TCP/IP for VM) that have been incorporated into this level of TCP/IP for z/VM are:

PK01120 PK02500 PK04088 PK07003 PK09863 PK10073 PK11392 PK13124
PK14089 PK14462 PK15521 PK16654 PK16916 PK17330 PK17568 PK18025
PK18868 PK20097 PK20336 PK20629 PK20644 PK21154 PK21497 PK22014
PK22191 PK22609 PK23103 PK23286 PK24289 PK24307 PK24638 PK24639
PK25058 PK25362 PK25441 PK27201 PK27491 PK27669 PK27932 PK28021
PK28297 PK28447 PK28708 PK28763 PK29547 PK30302 PK30609 PK31600
PK32005 PK32094 PK33023 PK34158 PK35220 PK36029 PK40449 PK40875
PK41224 PK42210 PK42976

4.2 Service Level Information

Before you install and configure TCP/IP for z/VM, you should review the TCPIP530 PSP (Preventive Service Planning) “bucket” for updated installation information that you should be aware of, or for information about PTFs that should be installed. Specify upgrade and subset values of **TCPIP530** and **VM530**, respectively, when you request or obtain this information.

4.3 Cumulative Service (RSU) Information

Cumulative service for TCP/IP for z/VM is available through a periodic, preventive service deliverable, the Recommended Service Upgrade (RSU). The RSU is used to provide service updates for multiple z/VM components (including TCP/IP for z/VM) and is often referred to as a *stacked* RSU.

The current-level of the stacked z/VM RSU can be obtained using the information provided in Figure 6:

Figure 6. Cumulative Service (RSU) Information

RETAIN		
COMPID	**Release**	**PTF**
568411202	RSU	**UM97530**

Note: Current RSU status and content information is available at the TCP/IP for z/VM home page on the World Wide Web. The URL for this home page is:

www.vm.ibm.com/related/tcpip/

5.0 Installation Requirements and Considerations

The following sections identify system requirements for installing TCP/IP for z/VM.

5.1 Hardware Requirements

There are no special hardware requirements to install TCP/IP for z/VM. Additional hardware requirements for exploiting specific functions of TCP/IP for z/VM are documented in the announcement material and in *TCP/IP Planning and Customization* (SC24-6125).

5.2 Program Considerations

The following sections list programming considerations for installing TCP/IP for z/VM.

5.2.1 Operating System Requirements

TCP/IP for z/VM requires the following operating system:

- z/VM version 5 release 3
- CMS Level 23

5.2.2 Other Program Product Requirements

5.2.2.1 Other Program Product Requirements - TCP/IP for z/VM

IBM C/C++ for z/VM (5654-A22) has been used to build the C components that provide the TCP/IP services listed here. The Language Environment® for z/VM (supplied as an installed component of z/VM version 5 release 3) is necessary to use these services:

- Domain Name Server virtual machine (NAMESRV)
- Internet Message Access Protocol server (IMAP)
- Lightweight Directory Access Protocol server (LDAPSRV)
- Multiple Protocol ROUTE server (MPRoute)
- Network Data Base servers (NDBPMGR and NDBSRV*nn*)
- Portmapper server (PORTMAP)
- Remote Execution daemon (REXECD and RXAGENT*n*)
- SNMP Query Engine, Agent and Subagent (SNMPD, SNMPQE and SNMPSUBA)
- Sockets Applications Programming Interface
- Network File System server (VMNFS)
- Kerberos Authentication and Administrator Servers (VMKERB and ADMSERV)

Various client functions require Language Environment for z/VM support as well. Representative of these are:

- CMSRESOL and CMSRESXA
- DIG
- LDAP Client Applications
- NDBSRVS
- NFS (client)
- NSLOOKUP
- PING
- RPCGEN and RPCINFO
- TRACERTE

Notes:

1. By default, the code for the above services is installed when you install TCP/IP for z/VM, regardless of whether you intend to use all, or only a subset, of these services.
2. Language Environment (LE) **APAR VM64055** must be applied on any system where TCP/IP level 530 is used. While this APAR is incorporated within the z/VM version 5 release 3 System Deliverable, this requirement is cited here for consideration in the event one attempts to use TCP/IP for z/VM on a level of CP and CMS other than level 530 (as might be the case when migration testing is performed).

Additional software requirements for exploiting specific TCP/IP for z/VM functions are documented in the announcement material and in *TCP/IP Planning and Customization* (SC24-6125).

5.2.2.2 Other Program Product Requirements - SSL Server

To use the Secure Socket Layer (SSL) server, a suitably configured Linux kernel and file system must be installed on your z/VM system. Detailed information about Linux requirements and preparing Linux for use with the SSL server is available at the TCP/IP for z/VM home page on the World Wide Web. The URL for this home page is:

```
www.vm.ibm.com/related/tcpip/
```

Also, ensure that the restrictions and requirements that follow (which pertain to the **SSLSERV** user ID, or your selected equivalent) are met:

- Virtual storage defined for the user ID selected to run the z/VM SSL server **must not exceed 2G**. This restriction also applies to any non-contiguous storage extents that might be defined for this user ID.
- The minidisk used as the SSL server **TRANSITION** minidisk (device address **0203**, by default) must be a CMS-formatted minidisk.

SSL and TCP/IP Server Compatibilty Changes

With TCP/IP level 530, the SSL and TCP/IP servers have been modified to accommodate “Dynamic SSL/TLS Support,” which introduces a set of Application Programming Interfaces (APIs) that permit a Pascal or Assembler client or server application to control the acceptance and establishment of TCP/IP sessions encrypted using SSL/TLS.

To provide this capability, the communication interface used by these servers has been modified. Due to the nature of these changes, an SSL server implementation that is based on prior levels of TCP/IP for z/VM **cannot** be used with the updated TCP/IP server. One of the SSL-related RPM packages supplied with with TCP/IP level 530 **must** be used for SSL server setup and configuration.

For a summary of TCP/IP level 530 SSL and TCP/IP server compatibility, refer to Figure 10 on page 36.

5.2.3 Program Installation/Service Considerations

This section describes items that should be considered before you install or service TCP/IP for z/VM

- VMSES/E is required to install and service this product.
- If multiple users install and maintain licensed products on your system, you may encounter problems when you attempt to gain write access to the Software Inventory (MAINT 51D) minidisk. Write access contention to this minidisk can be eliminated by converting the Software Inventory to use CMS Shared File System (SFS) directories instead of minidisks. See the *z/VM: VMSES/E Introduction and Reference*, Chapter 18, “Changing the Software Inventory to an SFS Directory,” for information about how to make this change.
- TCP/IP service procedures now require the TCP/IP for z/VM installation and service user ID (5VMTCP30, by default) to have ***file pool administration authority*** for the **VMSYS** file pool. Such authorization is necessary to accommodate service update processing for LDAP server components that reside in the z/VM Byte File System (BFS).

  For the z/VM version 5 release 3 System Deliverable, the 5VMTCP30 user ID has already been enrolled as a file pool administrator for the **VMSYS** file pool. If you choose to use a different user ID to service TCP/IP for z/VM, or elect to use a file pool other than VMSYS for maintaining the Byte File System, you then will need to enroll the appropriate user ID as an administrator for the applicable file pool.
- To allow for the installation and servicing of TCP/IP LDAP components, the BFS file pools (**VMSYS** and **VMSYSU**, by default) must be available and in operation.

- TCP/IP for z/VM is supplied with two IBM-defined components. When you perform installation and service tasks, you need to make use of the appropriate TCP/IP component, based on whether the installation and service environment is maintained using minidisks (the default) or using Shared File System (SFS) directories. Select and use the appropriate component for your environment from those listed here:

  The IBM-defined components for TCP/IP for z/VM are:

  TCPIP Used if TCP/IP for z/VM is installed and serviced using minidisks

  TCPIPSFS Used if TCP/IP for z/VM is installed and serviced using the Shared File System.

  Note! Minidisk Requirement

  Certain minidisks **must** be defined and used with TCP/IP server machines, even when TCP/IP for z/VM service minidisks are converted to Shared File System directories. This requirement is explained further in item 6 of 5.3, “DASD Storage and User ID Requirements” on page 44.

- On occasion, you may need to perform additional installation or service processing steps to account for problems or errors that cannot be corrected through conventional means. If such steps are required, appropriate explanatory notes and text will be provided in updated levels of this document.

- TCP/IP for z/VM source files are supplied in packed format. Use the CMS COPYFILE command (with its UNPACK option) to unpack TCP/IP source files prior to their use.

5.2.4 Migration Considerations

This section provides general information about changes to TCP/IP for z/VM that you should be aware of prior to its installation and use. The changes described herein are presented on a level-to-level basis, and grouped with respect to these topics:

- VMSES/E Migration Procedures
- Packaging
- General TCP/IP Usage
- FTP Client
- Printing
- General TCP/IP Server Configuration
- DNS Server
- FTP Server
- IMAP Server
- Kerberos Server
- LDAP Server
- NETSTAT Command
- MPRoute Server
- Remote Execution Services
- SMTP Server
- SNMP Server
- SSL Server
- **TCP/IP (Stack) Server**
- Telnet Server and Client

For the most part, these changes have been implemented to accommodate the introduction of new functions and improvements to existing functions. In some cases, existing functions may have been removed or replaced by alternative functions.

Note - Supported Environments

TCP/IP level 530 is supported on corresponding level 530 releases of CP and CMS only. Refer to section 5.2.1, “Operating System Requirements” on page 13 for details about the CP and CMS levels required for using TCP/IP level 530.

If TCP/IP level 530 services and functions are used with other CP or CMS levels (as might be the case for migration testing purposes), certain capabilities may be limited or may not function. In some instances, non-TCP/IP service updates ***may*** be available to facilitate the temporary use of TCP/IP in such a transitory environment.

5.2.4.1 VMSES/E Migration Procedures

5.2.4.1.1 General Information

- If you use the VMSES/E migration procedures (as documented in the *z/VM: Guide for Automated Installation and Service*) to migrate TCP/IP for z/VM from a z/VM version 5 system to z/VM version 5 release 3, then TCP/IP customizable files will be migrated to z/VM version 5 release 3, where possible.

  If customizable files have been changed on the new, serviced level of TCP/IP for z/VM and you have made changes to them on your z/VM version 5 system, then you will be informed about pertinent files for which your changes must be reworked.

  Note that when the VMSES/E migration procedures are used, no attempt is made to migrate data that resides on the TCP/IP for z/VM SSL server 201 and 203 minidisks.

5.2.4.2 Packaging

5.2.4.2.1 General Information About TCP/IP Level 530

- TCP/IP level 530 is included as a pre-installed component of the z/VM product; its use is governed by your license for z/VM.
- TCP/IP level 530 is **not** separately orderable or installable from the z/VM product. However, service that is obtained for TCP/IP for z/VM can be *applied* separately from that for z/VM.
- TCP/IP level 530 RSU service is provided as part of a *stacked* z/VM RSU, and not as a separately orderable RSU. Corrective (COR) service for TCP/IP for z/VM can be obtained and applied separately from other z/VM service.
- This level of TCP/IP relies on the presence of certain functions in the z/VM version 5 release 3 levels of CP and CMS. The converse is also true — using z/VM version 5 release 3 CMS requires that TCP/IP level 530 be present, to accommodate those functions that use TCP/IP (DNS) resolver services.

  Abends and incorrect results are possible if you attempt to use mixed levels of TCP/IP, CP and CMS.
- TCP/IP softcopy publications are provided in the same manner as other z/VM softcopy publications, and are included with these z/VM publications.

5.2.4.2.2 Changes Introduced in TCP/IP Level 530

- TCP/IP service procedures now require the TCP/IP for z/VM installation and service user ID (5VMTCP30, by default) to have ***file pool administration authority*** for the **VMSYS** file pool. Such authorization is necessary to accommodate service update processing for LDAP server components that reside in the z/VM Byte File System (BFS).

  For the z/VM version 5 release 3 System Deliverable, the 5VMTCP30 user ID has already been enrolled as a file pool administrator for the **VMSYS** file pool. If you choose to use a different user ID to service TCP/IP for z/VM, or elect to use a file pool other than VMSYS for maintaining the Byte File System, you then will need to enroll the appropriate user ID as an administrator for the applicable file pool.

- Support for the BOOTPD and RouteD servers has been withdrawn. The server user IDs, command programs, configuration statements and files associated with these servers have likewise been removed from the z/VM version 5 release 3 System Deliverable.

  If appropriate, the DHCPD and MPRoute servers should be configured to provide any support and functionality for your installation that was previously provided by the respective BOOTPD and RouteD servers.

- The default minimum virtual storage size defined for *most* TCP/IP server virtual machines is now **32M**. However, the default minimum for certain virtual machines, such as the IMAP, LDAP and SSL servers is defined higher than this default.

  For detailed information about how specific TCP/IP user IDs have been defined, consult the **5VMTCP30 PLANINFO** file. This file is located on the 5VMTCP30 191 minidisk.

  Due to changes in CMS resolver code, TCP/IP level 530 requires the CMS component of z/VM version 5 release 3 (CMS level 23) when various TCP/IP functions are used. Refer to 5.2.1, "Operating System Requirements" on page 13 for details about additional operating system and other program product requirements.

5.2.4.2.3 Changes Introduced in TCP/IP Level 520

- Definitions and sample configuration files have been added to provide a pair of system-default Virtual Switch (VSWITCH) controller virtual machines, **DTCVSW1** and **DTCVSW2**. Refer to *z/VM: Connectivity* (SC24-6080) for more information about Virtual Switches, and their definition and use in a z/VM environment.
- The various *sample* EXEC files provided with TCP/IP for z/VM are now supplied with a file type of **SAMPEXEC** instead of the previously-used file type of **SEXEC**.

5.2.4.2.4 Changes Introduced in TCP/IP Level 510

- The TCP/IP for z/VM configuration files provided as part of the z/VM version 5 release 1 System Deliverable are now supplied with only *sample* file names and file types, and reside on only the appropriate TCP/IP production minidisks (TCPMAINT 591 or TCPMAINT 592). The TCP2PROD command (with the **TCPCONFIG** section name specified as an operand) can optionally be used to copy these files — using appropriate file naming and file placement — to create an initial (or, "starter") set, of configuration files that then can be customized for your installation.
- The **TCP2PROD** command has been updated to employ improved selectivity and (sample) file change notification when serviced TCP/IP for z/VM files are placed into production. This capability is facilitated in part by the addition of **TCPSAMPLE** and file *exclusion* definition sections within the supplied TCP/IP CATALOG file.

5.2.4.2.5 Changes Introduced in TCP/IP Level 440

TCP/IP Port Restriction Defaults Have Changed

Ensure you review the information presented in 5.2.4.18, "TCP/IP (Stack) Server" on page 38, which discusses the **RESTRICTLOWPORTS** default change introduced with TCP/IP level 440.

- The **Pascal**-based client and server functions supplied with TCP/IP level 440 (for example, FTP MODULE or SMTP MODULE) are **not backward compatible** with prior-level TCP/IP "stack" servers, due to internal control block changes that have been implemented in this level of TCP/IP.

  Conversely, in some cases, it *may* be possible to use older Pascal-based functions in concert with a TCP/IP level 440 "stack" module. However, such use may result in client "hang" situations or other unexpected results.

  To alert TCP/IP administrators of conditions in which a client-stack mismatch may be relevant, the TCP/IP stack will produce **DTCREQ076I** and **DTCREQ077I** console messages on a default basis. In the event the volume of such messages is too great, the **NOLEVELWARNING** operand of the **ASSORTEDPARMS** statement can be used to suppress these messages.

  For a summary of TCP/IP level 440 Pascal component compatibility, refer to Figure 7.

Figure 7. TCP/IP level 440 Pascal Module Compatibility

	TCP/IP level 440 Pascal Functions	**Prior-level TCP/IP Pascal Functions**
TCP/IP level 440 Stack Server	Compatible	**Not Recommended**
Prior-level TCP/IP Stack Server	**Not Compatible**	Compatible

- The **C**-based client and server functions supplied with TCP/IP level 440 rely upon the Language Environment for z/VM run-time library that is provided with z/VM version 4 release 4. Because this library resides on the z/VM Product Code minidisk (typically the MAINT 19E minidisk, or **Y-disk**), changes to local procedures and modifications may be required to ensure this library is used in conjunction with TCP/IP services.

  For example, if the VMLINK command is currently used to acquire Language Environment support minidisks for using TCP/IP services, it may no longer be necessary to use this command for such a purpose. Similarly, the use of `:vmlink.` tags within a customized **DTCPARMS** file may no longer be required.

- TCP/IP CMS Help files are now maintained separately from other TCP/IP for z/VM client files. With this change, these help files are now placed into production on the z/VM Product HELP disk (MAINT 19D, or its equivalent).

- Linux-based components of the SSL server implementation are now provided using RPM-format files, which are maintained on a newly introduced TCP/IP service build disk. When the SSL server is configured for use, the appropriate RPM package file must be retrieved from this disk and then installed on the intended Linux guest by using the Red Hat Package Manager (RPM).

  The DASD requirements for the SSL Server have also been modified, to allow for more flexibility in how the required Linux guest services are provided. As a result of these changes, SSL-specific binary system files are no longer required (or provided) with TCP/IP for z/VM.

- The z/VM version 4 release 4 system directory entries for each TCP/IP for z/VM service virtual machine now include Read-only LINK statements for the 5VMTCP30 491 and 492 minidisks. These minidisk links have been added to better facilitate the testing of newly applied service.
- An IMAPAUTH user ID and a corresponding 191 minidisk have been added to the default installation, for supporting the (optional) IMAP User Authentication Exit introduced with this level of TCP/IP for z/VM.

5.2.4.2.6 Changes Introduced in TCP/IP Level 430

- The **Pascal**-based client and server functions supplied with TCP/IP level 430 (for example, FTP MODULE or SMTP MODULE) are **not backward compatible** with prior-level TCP/IP “stack” servers, due to internal control block changes that have been implemented in this level of TCP/IP. Conversely, it should be possible to use older Pascal-based functions in concert with a TCP/IP level 430 “stack” module in most environments.

  For a summary of TCP/IP level 430 Pascal component compatibility, refer to Figure 8.

Figure 8. TCP/IP level 430 Pascal Module Compatibility

	TCP/IP level 430 Pascal Functions	**Prior-level TCP/IP Pascal Functions**
TCP/IP level 430 Stack Server	Compatible	Compatible
Prior-level TCP/IP Stack Server	**Not Compatible**	Compatible

5.2.4.2.7 Changes Introduced in TCP/IP Level 420

- TCP/IP level 420 is included as a pre-installed component of the z/VM product; its use is governed by your license for z/VM.
- TCP/IP level 420 is **not** separately orderable from the z/VM product, although it is serviced separately from z/VM.

5.2.4.2.8 Changes Introduced in TCP/IP Level 410

- TCP/IP for z/VM is no longer a priced feature of z/VM version 4 release 1. No action is required to activate (enable) this component of z/VM.
- The TCP/IP NFS Server Feature for z/VM is no longer a priced feature of TCP/IP for z/VM. No action is required to activate (enable) the TCP/IP NFS server.
- The TCP/IP Source Feature for z/VM is no longer a priced feature of TCP/IP for z/VM. TCP/IP for z/VM source files are now supplied and installed as part of the z/VM version 4 release 1 System DDR.

5.2.4.2.9 Changes Introduced in TCP/IP Level 3A0

- DES Kerberos functions are now incorporated in the base TCP/IP Feature for z/VM (non-DES Kerberos functions are no longer available). Therefore, customers who have a Kerberos database that was created in a non-DES environment **must** delete and then rebuild that database using the supplied DES Kerberos functions. Refer to the chapter titled “Configuring the Kerberos Server” in *TCP/IP Planning and Customization* for more information about building the Kerberos database.

- The various servers and code that provide support for the Network Computing System (NCS) are no longer provided.
- The installation user ID-owned “Sample” (2C2) minidisk and its corresponding sample files are no longer provided. The sample files provided on this disk in prior levels of TCP/IP can be obtained via the IBM TCP/IP for VM Internet home page, for which the URL is:

  `www.vm.ibm.com/related/tcpip/`

5.2.4.2.10 Changes Introduced in TCP/IP Function Level 320

- No function-specific packaging changes have been made as part of TCP/IP function level 320.

5.2.4.2.11 Changes Introduced in TCP/IP Function Level 310

- TCP/IP-based user e-mail functions are included in CMS. TCP/IP-specific versions of NOTE and SENDFILE are **not** provided with the TCP/IP Feature for z/VM.
- A subset of RSCS version 3 release 2 function is available with the z/VM product to provide enhanced TCP/IP client and server print capabilities. RSCS functions not related to TCP/IP printing enhancements require a separate RSCS license for use. Refer to the *RSCS version 3 release 2 Program Directory* for additional information.
- Non-DES Kerberos functions are included in the base TCP/IP Feature for z/VM. DES Kerberos functions are available as a separate feature that must be separately installed.

5.2.4.3 General TCP/IP Usage

5.2.4.3.1 Changes Introduced in TCP/IP Level 530

- The level of authorization to use the TRACERTE command has changed. OBEY authority is no longer required to use this command.
- The FTP and Telnet clients have been updated to accommodate Dynamic SSL/TLS support. Changes to provide this capability include support for additional command options, which include:
  - **CERTFULLCHECK**
  - **CERTNOCHECK**
  - **NOSECURE**
  - **SECURE**
- A new statement, **SECURETELNETCLIENT**, may now be specified in the TCPIP DATA file. This statement provides the default Telnet client security value to use when neither the SECURE nor NOSECURE option is specified on the Telnet command.
- Due to changes in CMS resolver code, there is a requirement that some TCP/IP modules such as FTP be at the 5.3.0 level when the CMS IPLed is 5.3.0. For this same reason, customer-built modules that include COMMTXT in the GLOBAL TXTLIB command (in order to use routines such as `SayIntAd`) must be re-built using the z/VM 5.3.0 level of COMMTXT. In addition, some TCP/IP modules require Language Environment (LE) to include **APAR VM64055**.

Note: The following message will appear if either a TCP/IP routine or LE is back-level:

```
DMSZER2571E A resolver request failed due to missing LE support or incorrect TCP/IP
module levels.
```

To work around this problem, upgrade TCP/IP and LE and re-build programs that use COMMTXT or IPL an earlier level of CMS.

5.2.4.3.2 Changes Introduced in TCP/IP Level 520

- The **IPFORMAT** diagnostic utility has been added. This utility can be used to analyze LAN traffic captured via the CP **TRSOURCE** command, as well as data captured using the TCP/IP server packet tracing facility (which makes use of existing **PACKETTRACESIZE** and **TRACEONLY** configuration statements). A related sample program, **PKTTRACE**, is also included. This program may be used as an aid in gathering TCP/IP server packet trace data, and preparing this information for use as input to IPFORMAT.
- The content of the **ETC SERVICES** sample file (**ETC SAMPSERV**) has been revised such that port number assignments reflect current assignments as listed by the Internet Assigned Numbers Authority (IANA), at this URL:

  ```
  www.iana.org/assignments/port-numbers
  ```

5.2.4.3.3 Changes Introduced in TCP/IP Level 510

- With this level, **host/domain name resolution** is now performed for various TCP/IP functions through the use of IBM Language Environment (LE) sockets. To ensure that adequate virtual storage is available to allow for such resolution, ***virtual machines that use TCP/IP functions should be defined with a minimum of 16M of virtual storage***.
- The NETSTAT, TRACERTE and PING applications have been enhanced to include IPv6 support. In addition, the Pascal TRACERTE and PING functions have been replaced with C-based equivalents.
- With this level, TCP/IP for z/VM offers two local host files for domain name resolution and reverse name resolution:
  - the introduced and *preferred* **ETC HOSTS** file, which supports both IPv4 and IPv6 address formats; this file is supplied in sample form as the file, ETCHOSTS SAMPLE
  - the older HOSTS LOCAL file, which supports only IPv4 address formats.

  A sample conversion utility (LCL2ETC SEXEC) is also provided, which can be used to create an ETC HOSTS file from an existing HOSTS LOCAL file.

  Note: If **loopback** support is required for your installation, you must explicitly code a LOOPBACK entry for address 127.0.0.1 in the ETC HOSTS file.

5.2.4.3.4 Changes Introduced in TCP/IP Level 440

- The **Pascal**-based client and server functions supplied with TCP/IP level 440 (for example, FTP MODULE or SMTP MODULE) are **not backward compatible** with prior-level TCP/IP “stack” servers, due to internal control block changes that have been implemented in this level of TCP/IP.

Conversely, in some cases, it *may* be possible to use older Pascal-based functions in concert with a TCP/IP level 440 “stack” module. However, such use may result in client “hang” situations or other unexpected results.

To alert TCP/IP administrators of conditions in which a client-stack mismatch may be relevant, the TCP/IP stack will produce **DTCREQ076I** and **DTCREQ077I** console messages on a default basis. In the event the volume of such messages is too great, the **NOLEVELWARNING** operand of the **ASSORTEDPARMS** statement can be used to suppress these messages.

For a summary of TCP/IP level 440 Pascal component compatibility, refer to Figure 7 on page 20.

- The **C**-based client and server functions supplied with TCP/IP level 440 rely upon the Language Environment for z/VM run-time library that is provided with z/VM version 4 release 4. Because this library resides on the z/VM Product Code minidisk (typically the MAINT 19E minidisk, or **Y-disk**), changes to local procedures and modifications may be required to ensure this library is used in conjunction with TCP/IP services.

  For example, if the VMLINK command is currently used to acquire Language Environment support minidisks for using TCP/IP services, it may no longer be necessary to use this command for such a purpose. Similarly, the use of `:vmlink.` tags within a customized **DTCPARMS** file may no longer be required.

- The **NETSTAT** command has been modified to produce nonzero return codes to aid in distinguishing command processing errors when they arise. Thus, local applications that utilize NETSTAT commands may require modification to interrogate and account for return codes other than zero. Details about **NETSTAT** return codes and the reasons for them can be found in the *TCP/IP User's Guide*.

5.2.4.3.5 Changes Introduced in TCP/IP Level 430

- The **Pascal**-based client and server functions supplied with TCP/IP level 430 (for example, FTP MODULE or SMTP MODULE) are **not backward compatible** with prior-level TCP/IP “stack” servers, due to internal control block changes that have been implemented in this level of TCP/IP. Conversely, it should be possible to use older Pascal-based functions in concert with a TCP/IP level 430 “stack” module in most environments.

  For a summary of TCP/IP level 430 Pascal component compatibility, refer to Figure 8 on page 21.

5.2.4.3.6 Changes Introduced in TCP/IP Level 420

- TCPIP has been changed to recognize a TCP/IP loopback address of *only* **127.0.0.1**. TCPIP no longer supports an alternative loopback address of 14.0.0.0.

  Existing TCPIP DATA files should be reviewed for **NSINTERADDR** statements that currently include a 14.0.0.0 loopback address. All such statements should be modified to instead make use of the conventional 127.0.0.1 address that is now supported/in use.

5.2.4.4 FTP Client

5.2.4.4.1 Changes Introduced in TCP/IP Level 530

- The Telnet client has been updated to accommodate Dynamic SSL/TLS support. Changes to provide this capability include support for additional command options, which include:

  - **CERTFULLCHECK**
  - **CERTNOCHECK**
  - **NOSECURE**
  - **SECURE**

  and new or updated FTP subcommands:

  - **CLEAR**
  - **CPROTECT**
  - **PRIVATE**
  - **LOCSITE** (updated to accept and process **CERTFULLCHECK** and **CERTNOCHECK** options)

  Support for **SECUREDATA** and **SECURECONTROL** statements within the **FTP DATA** file has been added as well.

5.2.4.4.2 Changes Introduced in TCP/IP Level 430

- The FTP client includes support for a **WIDTH** command option that allows more user control over how FTP console output is formatted.

5.2.4.4.3 Changes Introduced in TCP/IP Level 3A0

- The FTP client includes support for the **SIZE** subcommand, which allows a client to obtain the size of a file before it is retrieved.

5.2.4.4.4 Changes Introduced in TCP/IP Function Level 320

- The FTP client has been enhanced to make use of login information present in a NETRC DATA file. By default, when a connection is made to a foreign host, user ID and password information in the NETRC DATA file that is specific to that host are used as part of an automatic FTP login process. Automatic FTP login can be suppressed using the new **NONETRC** option, or through use of the new **NETRC** subcommand.

5.2.4.5 Printing

5.2.4.5.1 Changes Introduced in TCP/IP Function Level 310

- The LPR command can now route print files to RSCS for printing, freeing a user's virtual machine for other work. This may introduce the need for users to learn a limited set of RSCS commands in order to determine the status of their print files. Refer to the *TCP/IP User's Guide* for more information.
- RSCS can be used instead of LPSERVE to provide enhanced printer daemon (LPD) capabilities. The operation and administration characteristics of RSCS are very different from LPSERVE. Refer to the chapter titled "Configuring the RSCS Print Server" in *TCP/IP Planning and Customization* for more information.

5.2.4.6 General TCP/IP Server Configuration

5.2.4.6.1 Changes Introduced in TCP/IP Level 520

- Processing of the of the **HOME** and **GATEWAY** statements by the TCP/IP (Stack) server has been updated to allow for the specification of subnet masks using BSD-style and Classless Inter-Domain Routing (CIDR) notation. Corresponding **NETSTAT** command changes have been implemented to accommodate the reporting of such notation in data produced by the NETSTAT HOME and NETSTAT GATE commands.
- The various *sample* EXEC files provided with TCP/IP for z/VM are now supplied with a file type of **SAMPEXEC** instead of the previously-used file type of **SEXEC**.

5.2.4.6.2 Changes Introduced in TCP/IP Level 510

- The TCP/IP for z/VM configuration files provided as part of the z/VM version 5 release 1 System Deliverable are now supplied with only *sample* file names and file types, and reside on only the appropriate TCP/IP production minidisks (TCPMAINT 591 or TCPMAINT 592). The TCP2PROD command (with the **TCPCONFIG** section name specified as an operand) can optionally be used to copy these files — using appropriate file naming and file placement — to create an initial (or, "starter") set, of configuration files that then can be customized for your installation.
- With this level, TCP/IP for z/VM offers two local host files for domain name resolution and reverse name resolution:

  - the introduced and *preferred* **ETC HOSTS** file, which supports both IPv4 and IPv6 address formats; this file is supplied in sample form as the file, ETCHOSTS SAMPLE
  - the older HOSTS LOCAL file, which supports only IPv4 address formats.

  A sample conversion utility (LCL2ETC SEXEC) is also provided, which can be used to create an ETC HOSTS file from an existing HOSTS LOCAL file.

  Note: If **loopback** support is required for your installation, you must explicitly code a LOOPBACK entry for address 127.0.0.1 in the ETC HOSTS file.

5.2.4.6.3 Changes Introduced in TCP/IP Level 440

- The **C**-based client and server functions supplied with TCP/IP level 440 rely upon the Language Environment for z/VM run-time library that is provided with z/VM version 4 release 4. Because this library resides on the z/VM Product Code minidisk (typically the MAINT 19E minidisk, or **Y-disk**), changes to local procedures and modifications may be required to ensure this library is used in conjunction with TCP/IP services.

  For example, if the VMLINK command is currently used to acquire Language Environment support minidisks for using TCP/IP services, it may no longer be necessary to use this command for such a purpose. Similarly, the use of `:vmlink.` tags within a customized **DTCPARMS** file may no longer be required.

5.2.4.6.4 Changes Introduced in TCP/IP Function Level 310

- Separate server initialization execs (TCPIPXIT, FTPDEXIT, etc.) are no longer used. All server parameters and features are controlled by entries contained in a DTCPARMS file. There is support for you to supply an exit which is specific to a particular server, or an exit that is used by all servers, for customization needs that cannot be met using a customized SYSTEM DTCPARMS file.
- Changing server names (when defining a second set of TCP/IP servers, for example) no longer requires changes to IBM-provided execs. All information related to the user ID of a particular server is kept in the TCPIP DATA file, or is part of the server configuration and is maintained in the DTCPARMS file.
- External Security Manager (ESM) interfaces have been standardized across all servers. The RPIUCMS command is used to initialize the RACROUTE environment, and RPIVAL is used to validate user IDs and passwords supplied by clients. These commands can be changed using a customized SYSTEM DTCPARMS file.
- The ESM environment is automatically established for a server when `:ESM_Enable.YES` is specified for that server in a customized SYSTEM DTCPARMS file.

Refer to the chapter titled “TCP/IP Server Configuration” in *TCP/IP Planning and Customization* for more information.

5.2.4.7 DNS Server

5.2.4.7.1 Changes Introduced in TCP/IP Level 3A0

- The use of a cache file (as used with a CACHINGONLY name server configuration) has been expanded to the PRIMARY and SECONDARY configurations. A new **ROOTCACHE** statement allows a cache file to be specified in a manner similar to that which can be specified on the CACHINGONLY statement. A corresponding sample root cache file, NSMAIN SCACHE, supplies several configuration recommendations, a list of root name servers, and the Internet site from which the most current list can be obtained.
- A **FORWARDERS** statement has been added to improve name resolution capability for z/VM hosts which are connected to other hosts that provide network firewall services. This statement can be used to identify other name server hosts which can perform name resolution outside the firewall system.

- The algorithms used in the caching subsystem have been improved to facilitate faster access to cached information and to more quickly determine when cached information does not exist.

5.2.4.8 FTP Server

5.2.4.8.1 Changes Introduced in TCP/IP Level 530

- The FTP server has been updated to accommodate Dynamic SSL/TLS support. Changes to provide this capability include support for additional server configuration statements, which include:
  - **PASSIVEPORTRANGE**
  - **SECURECONTROL**
  - **SECUREDATA**
  - **TLSLABEL**

  along with new or updated SMSG administrative command operands:
  - **SECURE**
  - **TLSLABEL**
  - **QUERY** (Updated to support an added **SECURE** keyword)

  The **CHKIPADR** server exit has been updated as well, to accommodate the use of a **SECUREDATA** keyword that can be used to establish minimum security levels for data connections.
- The FTP server has been modified such that it will no longer create the file FTPSERVE LOG. Messages formerly written to this file now are either directed to the server console (the case for message DTCFTS4015I) or replaced with an equivalent, existing message (message DTCFTS2512I replaces message DTCFTS2513I).

5.2.4.8.2 Changes Introduced in TCP/IP Level 3A0

- The FTP server has been modified such that it can now process more than 256 concurrent connections. The maximum number of connections possible is governed by available virtual storage within the server.

5.2.4.8.3 Changes Introduced in TCP/IP Level 3A0

- The FTP server now makes use of a server-specific configuration file (**SRVRFTP CONFIG** by default) for its startup information, in a manner more consistent with other TCP/IP servers. Therefore, **:parms.** entries within the DTCPARMS file that are associated with the FTP server need be modified to account for this change. Any SRVRFTP operands currently specified using a `:parms.` entry need to be removed, with appropriate modifications made to either the FTP server configuration file or the DTCPARMS file (for example, to account for use of the ANONYMOU or RACF command operands).

  In addition, the FTP server no longer makes use of the client-oriented FTP DATA file.
- The FTP server has been enhanced to better accommodate web browser and graphical FTP clients. Changes for this support include the ability to provide UNIX® -format list information in response to client DIR subcommand requests, through the use of the **LISTFORMAT** configuration statement. In addition, the **AUTOTRANS** statement can be used to configure the server to perform automatic

EBCDIC-ASCII data translation for transferred files, based on file types (or, file extensions), as determined by **VMFILETYPE** statements which are maintained in the TCPIP DATA file.

- Support for **AUTOTRANS** and **LISTFORMAT** operands to client-supplied SITE subcommands has been implemented as well, to provide client control over automatic EBCDIC-ASCII data translation and list information formats.
- **AUTOTRANS** and **LISTFORMAT** characteristics can now be established when specific FTP users login, when the CHKIPADR exit exec is customized for use.
- Support for additional SMSG interface commands has been added to allow for dynamic server configuration changes, console and tracing control, and shutdown/restart capability.

5.2.4.8.4 Changes Introduced in TCP/IP Function Level 320

- The FTP server has been enhanced to exploit new CP and CMS user authorization facilities provided with VM/ESA version 2 release 4. These enhancements allow an FTP user to access minidisks they own without the need for minidisk passwords to be defined or supplied during an FTP session. Thus, the link results achieved when FTP users access minidisk resources may noticeably differ from those using prior levels of TCP/IP for VM.

  For example, if a user establishes a read-only link to a minidisk (through the use of an explicit "**ACCT** *read_password*" command), a subsequent PUT command that initiates a write request to that minidisk may now succeed *if* the login user ID owns the minidisk in question. By comparison, with a prior TCP/IP level, such a request would cause an FTP user to be prompted to supply a Read/Write (or Multiple Read) password through use of the **ACCT** subcommand in order to first gain write access to the minidisk.

  The use of these new user authorization facilities also allows users with LOGONBY authority for a given *base* virtual machine to exercise that same authority during an FTP session. This is accomplished through use of the CP Diagnose X'88' command by the FTP server, which allows access to *base* user ID resources without requiring the password for that *base* user ID to be supplied. This kind of access is achieved through use of a new "/BY/*byuser_id*" parameter of the FTP **USER** subcommand. Additionally, the CP directory entry for the FTP server must include an OPTION DIAG88 statement, to allow use of the Diagnose X'88' command.
- FTP virtual reader support has been added, which allows an FTP client to use the virtual reader of a VM user ID as the current directory. For such users to issue DELETE and DIR or LS commands against a reader directory, the FTP server virtual machine must have class D privileges. To allow user to PUT files to a reader directory, the **RDR** parameter must be included with the FTP server startup command, SRVRFTP.

5.2.4.9 IMAP Server

5.2.4.9.1 Changes Introduced in TCP/IP Level 440

- Support for an IMAP User Authentication Exit is introduced, which removes the restriction that all IMAP users who are enrolled in the IMAP mail store must have a VM user ID and password. With this support, eight-character limits on user ID and password values are also removed, which allows registered IMAP clients to use IMAP services without knowledge (or restrictions imposed by) the VM IMAP implementation.

  The IMAP User Authentication Exit is activated by including a new **AUTHENTICATEID** statement in the IMAP server configuration file (IMAP CONFIG). A sample authentication exit exec is provided as **IMAPAUTH SEXEC**. The exit itself runs in a server virtual machine that is separate from the IMAP server; this additional server is named **IMAPAUTH**, by default.

  A new IMAP administrative command, **IMAPADM SETTINGS**, is also now supported. This command provides information about current IMAP server attributes and settings.

5.2.4.10 Kerberos Server

5.2.4.10.1 Changes Introduced in TCP/IP Level 3A0

- The non-DES Kerberos functions that were provided with previous levels of IBM TCP/IP for VM are no longer available. Instead, only a DES Kerberos system is supported, with the DES Kerberos functions incorporated as part of the TCP/IP Feature for z/VM.

  Any Kerberos database that was created in a non-DES environment will not work with the DES Kerberos functions supplied as part of TCP/IP Feature for z/VM. The existing non-DES Kerberos database **must** be deleted and then rebuilt using the supplied DES Kerberos functions. Refer to the chapter titled “Configuring the Kerberos Server” in *TCP/IP Planning and Customization* for more information about building the Kerberos database.

5.2.4.11 LDAP Server

5.2.4.11.1 Changes Introduced in TCP/IP Level 530

- A Lightweight Directory Access Protocol server (**LDAPSRV**) is introduced, which provides client access to data maintained in an LDAP directory. An LDAP directory provides an easy way to maintain directory information in a central location for storage, update, retrieval, and exchange.

  For more detailed information about what functions have been implemented in the initial version of the z/VM LDAP server, see *TCP/IP LDAP Administration Guide* (SC24-6140).

5.2.4.12 NETSTAT Command

5.2.4.12.1 Changes Introduced in TCP/IP Level 520

- The **NETSTAT** command has been updated to accommodate the reporting of IP network masks using BSD-style and Classless Inter-Domain Routing (CIDR) notation in results produced by the NETSTAT HOME and NETSTAT GATE commands (as dictated by the notation used for the TCP/IP (Stack) server **HOME** and **GATEWAY** configuration statements).
- A new **CONFIG** option has been added to allow NETSTAT to query stack configuration information.

5.2.4.12.2 Changes Introduced in TCP/IP Level 510

- The MTU size associated with a given link is now reported as part of the NETSTAT **DEVLINKS** output. The MTU size reported is either a defined LINK MTU value, or an intelligent default that has been assigned by TCP/IP.
- The NETSTAT command is now implemented as a C-based application.

5.2.4.12.3 Changes Introduced in TCP/IP Level 440

- The NETSTAT **DEVLINKS** command now reports data traffic information for the majority of devices that are supported by TCP/IP for z/VM. Traffic information is provided by newly introduced **BytesIn** and **BytesOut** fields that are included as part of the NETSTAT information produced for a given device.
- NETSTAT has been modified to produce nonzero return codes to aid in distinguishing command processing errors when they arise. Thus, local applications that utilize NETSTAT commands may require modification to interrogate and account for return codes other than zero. Details about **NETSTAT** return codes and the reasons for them can be found in the *TCP/IP User's Guide*.

5.2.4.12.4 Changes Introduced in TCP/IP Level 430

- An **OBEY** operand is now supported, which can be used to make dynamic changes to the TCP/IP stack server configuration. Any data that is appropriate for use with the OBEYFILE command can be processed using a NETSTAT OBEY command (to the extent that data can be provided via the CMS command line).

5.2.4.13 MPRoute Server

5.2.4.13.1 Changes Introduced in TCP/IP Level 530

- The MPRoute server implementation of TCP/IP for z/VM has been adapted from z/OS V1.8.
- A combination of an *ip_address* and a *subnet_mask* that describes either a subnet address or a broadcast address, is no longer accepted for the **OSPF_INTERFACE**, **RIP_INTERFACE** or **INTERFACE** configuration statements. This change is a result of TCP/IP server enforcement of RFC rules that restrict a subnetwork broadcast or network address from being used as a host address.

  By convention, an address that has all ***ones*** in the host portion is a subnet ***broadcast*** address, whereas an address that has all ***zeros*** in the host portion is a subnet ***network*** address. Therefore, an attempt to use an *ip_address* and *subnet_mask* combination that specifies one of these special addresses will likely lead to difficulty in communicating with other hosts on the network.

- The maximum *mtu* value that may be specified for the **OSPF_INTERFACE**, **RIP_INTERFACE** or **INTERFACE** configuration statements is now the minimum of:
  - the device buffer capacity (as reported by a HiperSockets™ or QDIO device; for other devices, the value is based on the device architecture)
  - the *buffer_size* value associated with the **LARGEENVELOPEPOOLSIZE** statement in the TCP/IP server configuration file.

Note: The RouteD server has been withdrawn from TCP/IP level 530. The server user ID, command program, configuration statements and files associated with this server have likewise been removed from the z/VM version 5 release 3 System Deliverable.

5.2.4.13.2 Changes Introduced in TCP/IP Level 520

- The MPRoute server implementation provided with this level of TCP/IP for z/VM has been revised, and is adapted from z/OS V1.7.

  With this implementation change, the **format of the MPRoute server messages differs from that of other TCP/IP messages**. The following message numbering convention is used for MPRoute messages:

  EZZ*nnnnt*

  where:

  EZZ is the product identifier for MPRoute

  nnnn is a unique 4-digit numeric value assigned to the message

  t is the message type that indicates the action assigned to the message.

  Types and their meanings are:

Figure 9. MPRoute Message Type (Action) Codes

Type Code	Meaning
A	Immediate action required
E	Eventual action required
D	Immediate decision required
I	Informational

- Various MPRoute server configuration statements have been revised, with respect to their use and supported operands, while others have been added in support of IPv6.
- A subnet mask value of **255.255.255.255** is no longer accepted for the **OSPF_INTERFACE**, **RIP_INTERFACE** or **INTERFACE** configuration statements. The most specific subnet mask that now can be specified for these statements is **255.255.255.252**. This change is a result of TCP/IP server enforcement of RFC rules that restrict a subnetwork broadcast or network address from being used as a host address.

By convention, an address that has all ***ones*** in the host portion is a subnet ***broadcast*** address, whereas an address that has all ***zeros*** in the host portion is a subnet ***network*** address. Therefore, the *subnet_mask* value specified for the aforementioned statements must have a sufficient number of zero bits such that no home address in that subnet has all zeros or all ones in the host portion of the address.

Consider a subnet that incorporates these two home addresses — `10.1.1.1` and `10.1.1.2` The subnet mask for this subnet must have zeros in at least two bit positions; for example, `255.255.255.252` (where the binary representation of `252` is `11111100`). However, if a subnet includes four home addresses — `10.1.1.1`, `10.1.1.2`, `10.1.1.3` and `10.1.1.4` — its subnet mask must have zeros in at least *three* bit positions; for example `255.255.255.248` (`248` in binary form is `11111000`). In this instance, up to six home addresses can be included in this subnet (`10.1.1.1` through `10.1.1.6`).

In general, if a subnet mask has ***n*** zero bits, then there can be up to $\mathbf{2^n - 2}$ home addresses in that subnet. This limit applies even if the subject home addresses are configured on different TCP/IP stacks.

- The name of the MPRoute server configuration file is no longer restricted to MPROUTE CONFIG. A newly introduced DTCPARMS **:config.** tag can be used to specify the file ID of the configuration file to be used when an MPRoute server is initialized.
- The existing support limit of only four equal-cost paths is removed — up to 16 equal-cost routes may now be generated for a given destination, to provide improved load-balancing support.
- The TCP/IP server now automatically generates static routes for IP addresses for which a subnet mask has been specified. When a subnet mask is specified for IPv4 home addresses, the TCP/IP server automatically generates a direct static route to the subnet described by the IP address and mask. For IPv6 addresses, the TCP/IP server automatically generates a direct static route to the network described by the first 64 bits of the address. Unlike static routes added through the GATEWAY statement, these routes may be replaced by dynamic routing protocols if MPRoute is running.

5.2.4.13.3 Changes Introduced in TCP/IP Level 510

- **LINK MTU** values, specified as part of LINK statements within the TCP/IP server configuration file, now override any MTU values that are specified for interface statements in the MPRoute server configuration file (MPROUTE CONFIG).

5.2.4.14 Remote Execution Services

5.2.4.14.1 Changes Introduced in TCP/IP Level 440

- The **REXEC** server has been updated to allow users with LOGONBY authority for a given *base* virtual machine to exercise that same authority when z/VM commands are remotely executed. This is accomplished through use of the CP Diagnose X'88' command within the REXEC server, which allows access to *base* user ID resources without requiring the password for that *base* user ID to be supplied.

  This kind of access is achieved by qualifying a remote login user ID with a new “`/BY/`*`byuser_id`*” parameter, when that user ID is specified using the **-l** operand of the **rexec** command. In addition, the system directory entry for the REXEC server must include an `OPTION DIAG88` statement, to allow use of the Diagnose `X'88'` command.

5.2.4.15 SMTP Server

5.2.4.15.1 Changes Introduced in TCP/IP Level 530

- The SMTP server has been updated to accommodate Dynamic SSL/TLS support. Changes to provide this capability include support for additional server configuration statements, which include:
  - **TLS**
  - **TLSLABEL**
- The SMTP server has been enhanced to honor the **DOMAINLOOKUP** statement in the TCPIP DATA file. In addition, the SMTP server now uses the ETC HOSTS file for the local site table, if present. If the ETC HOSTS file is not present, SMTP uses the HOSTS SITEINFO file.
- An SMTP NAMES file restriction against using a nickname that is the same as a user ID in the list defined by that nickname is removed. With this level, you can specify a nickname that matches a user ID in that nickname list.

5.2.4.15.2 Changes Introduced in TCP/IP Level 510

- Support for the **SUPPRESSNOTIFICATION** configuration statement has been modified such that “Received From” messages can be suppressed in addition to “Mail Delivered” messages. With this change, **RECEIVED**, **DELIVERED** and **ALL** operands are added for this statement. **ALL** is the default, meaning that both the mail received and mail delivered messages are to be suppressed.

5.2.4.15.3 Changes Introduced in TCP/IP Level 440

- Support for a new **VERIFYBATCHSMTPSENDER** configuration statement is added. When this statement is used, SMTP will reject batch mail that contains a MAIL FROM: address that differs from available RSCS information. Specific users can be exempted from this verification through the use of additional VERIFYBATCHSMTPSENDER operands.

5.2.4.15.4 Changes Introduced in TCP/IP Level 420

- Support for new **FILESPERCONN** and **MAXCONNPERSITE** statements has been added. These statements allow more control over the establishment of connections with remote SMTP servers, and may aid with adjusting mail delivery throughput.
- An **SMSG REFRESH** command can now be used by authorized users to dynamically update SMTP security or nickname table information, depending on the configuration of the SMTP server.

5.2.4.15.5 Changes Introduced in TCP/IP Level 3A0

- Timing values for the **RETRYAGE** and **WARNINGAGE** statements can now be specified in hours or minutes (in addition to a number of *days*), through the use of additional **H** or **M** statement operands. Existing defaults (specified in days, for which a **D** statement operand can now be specified) remain unchanged.

5.2.4.15.6 Changes Introduced in TCP/IP Function Level 320

- Support for the SMTP **EHLO** command has been added, as has support for the Message Size Declaration and 8-bit MIME transport service extensions. With this support, SMTP now accepts and handles the SIZE and BODY parameters on MAIL FROM: commands.

5.2.4.15.7 Changes Introduced in TCP/IP Function Level 310

- The **DEBUG** configuration file statement is no longer supported. This statement has been replaced by the **TRACE** statement which, when specified with the DEBUG parameter, provides identical function as the DEBUG statement, except that output goes to only the console; there is no debug file support. Refer to the chapter titled “Configuring the SMTP Server” in *TCP/IP Planning and Customization* for more information about the TRACE statement and additional parameters that can be specified.
- The default for the **WARNINGAGE** configuration file statement has been reduced from 3 days to 1 day.
- Due to the introduction of new and changed trace options, the **TRACE** SMSG command now requires additional options to be specified. In TCP/IP for VM releases prior to TCP/IP function level 310, the **TRACE** SMSG command was used to trace host name resolution via name severs. This SMSG command has been replaced with the **TRACE RESOLVER** SMSG command. Similar results can be achieved by specifying the TRACE RESOLVER statement in the SMTP configuration file. Refer to the chapter titled “Configuring the SMTP Server” in *TCP/IP Planning and Customization* for more information about configuration file changes, and to the *TCP/IP User's Guide* for SMTP SMSG command interface changes.
- The SMTP DATA file is no longer needed. The **ATSIGN** statement previously supported using this file has been incorporated within the TCPIP DATA file.
- The sample SMTP configuration file (SMTP SCONFIG) now specifies “NETDATA” as the default for both the **LOCALFORMAT** and **RSCSFORMAT** statements, to reflect the operational default value associated with these statements.
- Host names for mail items can now be re-resolved (when required) by using the **REPROCESS** SMSG command. Refer to the chapter titled “Configuring the SMTP Server” in *TCP/IP Planning and Customization* for more information. The formerly documented SMTP-EXP EXEC is no longer needed for this purpose and **should not be used**.
- SMTP work files (NOTE and ADDRBLOK files) have new formats and names; the file types for these files have been changed to NOTEFILE and ADDRFILE, respectively. When migrating from IBM TCP/IP for VM version 2 release 4, any NOTE or ADDRBLOK files present on the SMTP server A-disk are converted to the new format and renamed; such converted files cannot be processed by previous versions of TCP/IP for VM.

5.2.4.16 SNMP Server

5.2.4.16.1 Changes Introduced in TCP/IP Level 530

- An SNMP Subagent server is introduced which can be customized to supply a specific set of MIB variables (such as BRIDGE-MIB variables that are associated with a virtual switch). With this server, support for the use of various SNMP MIB exit routines, identified within a MIB_EXIT DATA file is added as well.

5.2.4.17 SSL Server

5.2.4.17.1 Changes Introduced in TCP/IP Level 530

- The SSL sever has been modified to accommodate "Dynamic SSL/TLS Support," which introduces a set of Application Programming Interfaces (APIs) that permit a Pascal or Assembler client or server application to control the acceptance and establishment of TCP/IP sessions encrypted using SSL/TLS.

  To provide this capability, the interface used by the SSL server to communicate with the TCP/IP stack server has been modified. Due to the nature of these changes, one of the SSL-related RPM packages supplied with TCP/IP level 530 **must** be used for SSL server setup and configuration. Note also that the TCP/IP level 530 SSL server **cannot** be used with prior levels of TCP/IP for z/VM.

  For a summary of TCP/IP level 530 SSL and TCP/IP server compatibility, refer to Figure 10.

Figure 10. TCP/IP level 530 SSL / TCP/IP Server Compatibility

	TCP/IP level 530 SSL Server	**Prior-level SSL Server**
TCP/IP level 530 Stack Server	Compatible	**Not Compatible**
Prior-level TCP/IP Stack Server	**Not Compatible**	Compatible

- The SSL server command (**VMSSL**) has been updated to allow additional security levels to be specified for the **EXEMPT** operand, to more easily eliminate weak ciphers. Support for a **NOHALT** operand has also been added, which allows the SSL server Linux guest to remain active after a critical error is encountered during server operations.
- The SSL server administration command (**SSLADMIN**) has been modified as follows:
  - The **SSLADMIN SELF** command has been updated to support an **EXPIRATION** operand, which may be used to specify number of days that a self-signed certificate is to be valid.
  - The **SSLADMIN LOG** command has been updated to accept file ID operands that allow SSL server log information to be maintained in a file named other than SSLADMIN LOG.
  - the **SSLADMIN LOGSIZE** and **SSLADMIN LOGCLEAR** commands are introduced. These respective commands allow a maximum size to be established for the SSL server log, and for accumulated log information to be purged within the SSL server.

- The RPM packages provided with this level of TCP/IP for z/VM support running the SSL server using these Linux distributions:
  - SUSE SLES9 Service Pack 3 (31-bit)
  - SUSE SLES9 Service Pack 3 (64-bit)
  - Red Hat Enterprise Linux AS (Version 4 U4) (31-bit)
  - Red Hat Enterprise Linux AS (Version 4 U4) (64-bit)

5.2.4.17.2 Changes Introduced in TCP/IP Level 520

- A **FIPS** operand is introduced that signifies the SSL server is to operate in FIPS (Federal Information Processing Standard, FIPS 140-2) mode, which restricts connections to those that employ FIPS-approved cipher suites.
- Server operation has been modified such that a server restart is not required to activate or remove a certificate.
- The RPM packages provided with this level of TCP/IP for z/VM support running the SSL server using these Linux distributions:
  - SUSE SLES8 Service Pack 3 (31-bit)
  - SUSE SLES9 Service Pack 2 (31-bit)
  - SUSE SLES9 Service Pack 2 (64-bit)
  - Red Hat Enterprise Linux AS V3 (31-bit)
  - Red Hat Enterprise Linux AS V3 (64-bit)
  - Red Hat Enterprise Linux AS V4 (31-bit)
  - Red Hat Enterprise Linux AS V4 (64-bit)

5.2.4.17.3 Changes Introduced in TCP/IP Level 440

- Linux-based components of the SSL server implementation are now provided using RPM-format files, which are maintained on a newly introduced TCP/IP service build disk. When the SSL server is configured for use, the appropriate RPM package file must be retrieved from this disk and then installed on the intended Linux guest by using the Red Hat Package Manager (RPM).

  The DASD requirements for the SSL Server have also been modified, to allow for more flexibility in how the required Linux guest services are provided. As a result of these changes, SSL-specific binary system files are no longer required (or provided) with TCP/IP for z/VM.
- The syntax for the **SSLADMIN SELF** command has been modified to be more consistent with that of the SSLADMIN STORE command.

5.2.4.18 TCP/IP (Stack) Server

5.2.4.18.1 Changes Introduced in TCP/IP Level 530

- Several enhancements have been added to help maintain high availability in the event of an ethernet QDIO (IPv4 or IPv6) or LCS (IPv4 only)) device failure. These enhancements include:

  – **IPv4 ARP Takeover Support**

    This support provides redundant ethernet (LCS or QDIO) adapters (two or more adapters connected to the same LAN segment) the ability to take over for each other in the event of a device failure.

    Provided all TCP/IP stack interfaces are properly configured, no TCP/IP configuration changes should be necessary for this function to be exploited. The TCP/IP stack uses the configured IP address/subnet mask pairs for each interface to make a determination about which interfaces are connected to the same LAN segment and are therefore eligible to take over for each other.

  – **IPv6 Neighbor Discovery Takeover Support**

    This support provides redundant QDIO ethernet adapters (two or more adapters connected to the same LAN segment) the ability to take over for each other in the event of a device failure.

    Configuration changes should not be necessary for this function to be used, since the TCP/IP stack perform “same LAN determination” for IPv6 interfaces at initialization, to identify which adapters are connected to the same LAN segment and are therefore eligible to take over for each other.

  – **Enhanced OSA Address Table (OAT) Management**

    With this enhancement, the IP addresses reported to an OSA adapter are restricted to:

    - those that are defined as adapters HOME addresses
    - IPv4 VIPA addresses (if the device is enabled for IPv4)
    - IPv4 Proxy ARP addresses (if the device is enabled for IPv4)
    - IPv6 VIPA addresses (if the device is enabled for IPv6)
    - any IP addresses for which the device has taken over

- **IPv6 VIPA Support** has been added, which gives hosts the ability to use the VIPA address as a target for IP traffic, allowing such traffic to be routed to a working adapter on the TCP/IP stack, should one of several applicable adapters fail. Configuration statement changes associated with this support include the following additions:

  – **IPV6SOURCEVIPA** operand (added for the **ASSORTEDPARMS** statement)
  – **ENABLEIPV6** option (added for the **LINK** statement)

- TCP/IP has been updated to allow both an IPv4 and IPv6 VLAN to be associated with a **QDIOETHERNET** link. Changes include:

  – acceptance of an IPv6 VLAN ID (in addition to an IPv4 VLAN ID) for the **QDIOETHERNET LINK** statement
  – **IFCONFIG** command changes that allow both an IPv4 VLAN ID and an IPv6 VLAN ID to be specified as part of the **VLAN** interface operand

- The **HOME** statement now accepts a **VSWITCH** operand, which identifies the link that is associated with an IPv4 address as one that can be used for VSWITCH management purposes such as retrieval of SNMP Bridge MIBs for the virtual switch. The named VSWITCH is also returned in **NETSTAT HOME** command results.
- Be aware that with z/VM version 5 release 3, it is no longer necessary to uncouple/recouple to a Guest LAN or VSWITCH for VLAN or promiscuous mode authorization changes to take effect. Thus, such changes now can immediately affect the operation of a TCP/IP stack server that is connected to a VSWITCH or Guest LAN.
- Support for a **-Remove** command option has been added to the **IFCONFIG** command, which allows an interface to be dynamically removed from the TCP/IP server configuration. This support makes use of a new socket API IOCTL subcommand (**SIOCDINTERFACE**) introduced at this level.
- Support for the **BSDROUTINGPARMS** statement has been withdrawn.
- The TCP/IP sever has been modified to accommodate “Dynamic SSL/TLS Support,” which introduces a set of Application Programming Interfaces (APIs) that permit a Pascal or Assembler client or server application to control the acceptance and establishment of TCP/IP sessions encrypted using SSL/TLS. New configuration statements introduced in conjunction with this capability are:

  – SSLSERVERID

  Note that to provide this capability, the interface used by the TCP/IP server to communicate with the SSL server has been modified. Due to the nature of these changes, **an SSL server implementation that is based on prior levels of TCP/IP for z/VM cannot be used with the TCP/IP level 530 TCP/IP server**.
- The **INTERNALCLIENTPARMS** statement, used for configuring the Telnet server, has been updated to accommodate Dynamic SSL/TLS support. Changes include support for **SECURECONNECTION** and **TLSLABEL** operands.

5.2.4.18.2 Changes Introduced in TCP/IP Level 520

- Processing of the of the **HOME** and **GATEWAY** statements by the TCP/IP (Stack) server has been updated to allow for the specification of subnet masks using BSD-style and Classless Inter-Domain Routing (CIDR) notation. Corresponding **NETSTAT** command changes have been implemented to accommodate the reporting of such notation in data produced by the NETSTAT HOME and NETSTAT GATE commands.
- The TCP/IP server now automatically generates static routes for IP addresses for which a subnet mask has been specified. When a subnet mask is specified for IPv4 home addresses, the TCP/IP server automatically generates a direct static route to the subnet described by the IP address and mask. For IPv6 addresses, the TCP/IP server automatically generates a direct static route to the network described by the first 64 bits of the address. Unlike static routes added through the GATEWAY statement, these routes may be replaced by dynamic routing protocols if MPRoute is running.

5.2.4.18.3 Changes Introduced in TCP/IP Level 510

- Internet Protocol Version 6 (IPv6) support has been updated to allow the z/VM TCP/IP stack to be configured for IPv6 networks connected through OSA-Express devices which operate in QDIO mode, as well as those associated with a z/VM QDIO Guest LAN. New configuration statements introduced in conjunction with this capability are:

  – EQUALCOSTIPV6MULTIPATH (for ASSORTEDPARMS)
  – IGNOREIPV6REDIRECT (for ASSORTEDPARMS)
  – ICMPERRORLIMIT
  – NCBPOOLSIZE
  – ROUTERADV
  – ROUTERADVPREFIX

  Various, existing statements have also been modified to accommodate their use for an IPv6 configuration. Refer to *TCP/IP Planning and Customization* for detailed information about the statements affected by these changes.

- Support for a **LINK** statement **MTU** operand has been added. When a value of zero (0) is specified for this operand, TCP/IP uses an intelligent MTU value based on the pertinent LINK definition. Refer to the section titled “Intelligent default MTU Values Based on the Device and Link Type” in *TCP/IP Planning and Customization* for detailed information about the MTU operand and its use.

  Note that with this change:

  – the MTU size reported by the **NETSTAT** command for a given link is either a defined LINK MTU value, or an intelligent default that has been assigned by TCP/IP

  – LINK MTU values override any MTU values that are specified for interface statements in the **MPRoute** server configuration file (MPROUTE CONIFIG).

- Server initialization logic has been updated to additionally support the use of an initial configuration file that is named to match the user ID of a given TCP/IP server virtual machine. With this change, TCP/IP searches for an initial configuration file in the following order:

  1. *userid* TCPIP
  2. *nodename* TCPIP
  3. PROFILE TCPIP

- **AUTORESTART** processing for **HiperSockets** and **OSD** devices has been modified. Upon an OSD device failure for which a restart is appropriate, TCP/IP will attempt to restart that device once, and then every 30 seconds thereafter, until the device becomes “ready” or it is stopped (as by OBEY STOP processing). When restart processing is in effect, a new status is reported for this device by the NETSTAT DEVLINKS command:

  ```
  AUTORESTART retry
  ```

- The **VSWITCH CONTROLLER** statement has been updated to specify whether failover capability is enabled or disabled for a designated TCP/IP stack controller. When failover is enabled, CP checks timestamps to confirm that the z/VM TCP/IP stack controller is responding to requests to service an OSA Express device associated with an active Virtual Switch.

- A **:vnic.** tag can now be specified in a customized SYSTEM DTCPARMS file, to automatically define a virtual Network Interface Card (NIC) and couple this adapter to a designated guest LAN or virtual switch, as part of TCP/IP initialization.

5.2.4.18.4 Changes Introduced in TCP/IP Level 440

- Be certain the following change is noted and properly addressed for your environment:

  Action Required

  - **TCP/IP Port Restriction Defaults Have Changed**
  - **Multiple TCP/IP applications may be affected and some TCP/IP applications may no longer function unless you take action.**

  The security of the z/VM TCP/IP stack has been improved by making the **RESTRICTLOWPORTS** operand of the **ASSORTEDPARMS** statement active ***by default***. Thus, **all TCP/IP applications that listen on “well-known” ports** (ports **1 through 1023**) must be given permission to do so. Such permission can be granted by customizing the TCP/IP server configuration file (PROFILE TCPIP, or its equivalent) in one of three ways:

  1. Use the **PORT** statement to reserve the specific port (or ports) required by each application (virtual machine) used on your system. **This is the preferred method**. Note that with TCP/IP level 440, ports can be reserved within a specific range, in addition to being reserved on an individual basis.
  2. Modify the **OBEY** statement such that affected application virtual machines are included in the TCP/IP obey list.
  3. Include the **FREELOWPORTS** operand as part of an **ASSORTEDPARMS** statement. Note that *this method removes the default protection for all well-known ports*

  Note: When the **RESTRICTLOWPORTS** default is in effect and appropriate port authorizations have not been provided, applications that rely upon well-known ports (for example, VM-based web servers or remote printing functions such as **lpr**) are likely to report “Unable to open port(s)” or “Permission denied” conditions.

- Variable-length subnet masks may be used without specification of the **VARSUBNETTING** operand of the **ASSORTEDPARMS** statement. VARSUBNETTING is now the default behavior of the TCP/IP stack and cannot be disabled.
- For **DEVICE** statements that correspond to *real* interface devices, a **CPU** operand can now be specified to designate a particular virtual processor for running the device drivers associated with such devices. Note that to exploit this capability, the stack must be configured as a virtual multiprocessor (Virtual MP).
- The *send_limit* and *receive_limit* defaults for the **DATABUFFERLIMITS** statement have been increased from **5** to **8**.
- Support for a new **SOMAXCONN** statement is introduced. This statement provides control over the maximum number of pending connection requests that are queued for a listening socket. The default

maximum is **10**. Note that using a different SOMAXCONN value may require corresponding changes to the SOCKET.H file, to allow for effective use of the specified maximum by C socket programs.

- Support for a new **LINK** statement operand, **NOFORWARD** (and its synonym, **NOFWD**) is introduced. This operand specifies that packets received on the designated link are not to be forwarded to another host, and that packets transmitted on this link must originate from the local host. In other words, when the **NOFORWARD** operand is specified for a link, traffic carried on that link will not be intermingled with traffic on other links.
- By default, the TCP/IP stack will produce **DTCREQ076I** and **DTCREQ077I** console messages to alert TCP/IP administrators of conditions in which a (Pascal-based) client-stack mismatch is detected. In the event the volume of such messages is too great, the **NOLEVELWARNING** operand of the **ASSORTEDPARMS** statement can be used to suppress these messages.
- The **PORTNAME** operand for **OSD** and **HiperSockets DEVICE** statements is now optional. However, for some OSA Express adapter levels (whether real or virtual), a PORTNAME operand and value must still be specified. When such an adapter is in use and the PORTNAME operand is omitted, error message DTCOSD310E will be displayed during device initialization.
- Virtual Local Area Network (**VLAN**) identifiers can now be specified on **QDIOETHERNET** and **QDIOIP LINK** statements, as well as with the **IFCONFIG** command.
- The TCP/IP stack can now act as a controller for a z/VM Virtual Switch. The **VSWITCH CONTROLLER** statement, along with the **IUCV *VSWITCH** system directory statement, determines whether a given stack can act as such a controller.
- The **NETSTAT DEVLINKS** command now reports data traffic information for the majority of devices that are supported by TCP/IP for z/VM. Traffic information is provided by newly introduced **BytesIn** and **BytesOut** fields that are included as part of the NETSTAT information produced for a given device.
- The **NETSTAT** command has been modified to produce nonzero return codes to aid in distinguishing command processing errors when they arise. Thus, local applications that utilize NETSTAT commands may require modification to interrogate and account for return codes other than zero. Details about **NETSTAT** return codes and the reasons for them can be found in the *TCP/IP User's Guide*.

5.2.4.18.5 Changes Introduced in TCP/IP Level 430

- The **NETSTAT** command now supports an **OBEY** operand, which can be used to make dynamic changes to the TCP/IP stack server configuration. Any data that is appropriate for use with the OBEYFILE command can be processed using a NETSTAT OBEY command (to the extent that data can be provided via the CMS command line).
- The **IFCONFIG** command introduced with this level of TCP/IP can be used to make dynamic configuration changes to network interfaces for the z/VM TCP/IP stack, or to display the current configuration. Note that network changes made using IFCONFIG are ***not permanent***. However, IFCONFIG can produce data that is compatible with the TCP/IP server configuration file, which then can be used for permanent modifications.

5.2.4.18.6 Changes Introduced in TCP/IP Level 420

- TCPIP has been changed to recognize a TCP/IP loopback address of *only* **127.0.0.1**. TCPIP no longer supports an alternative loopback address of 14.0.0.0.

  Existing TCPIP DATA files should be reviewed for **NSINTERADDR** statements that currently include a 14.0.0.0 loopback address. All such statements should be modified to instead make use of the conventional 127.0.0.1 address that is now supported/in use.

- The number of **DEVICE** statements that can be specified within the TCP/IP server configuration file is no longer limited to 100 such statements. The number of DEVICE statements that can be supported is now determined by the virtual storage size of the TCPIP virtual machine.
- Proxy ARP support (activated through use of the **PROXYARP** operand of the **ASSORTEDPARMS** statement) can now be used with OSA Direct (OSD) devices, as well as for more traditional point-to-point connections.
- The default for the **SCANINTERVAL** operand of the **INTERNALCLIENTPARMS** statement (used for Telnet server configuration) has been changed from 120 to 60 seconds.

5.2.4.18.7 Changes Introduced in TCP/IP Function Level 320

- The **TIMESTAMP** statement default has been changed from TIMESTAMP 0 to TIMESTAMP PREFIX, which specifies that a time stamp is to preface every trace and console message. This change helps in diagnosing problems and isolating error conditions. Therefore, it is recommended that any existing TCP/IP server configuration files be changed to specify TIMESTAMP PREFIX to aid in problem determination.
- The Telnet session connection exit interface has been changed to pass the LU name supplied by a client (if any) as an additional parameter. Existing exits may need to be changed to accommodate this behavior.
- The Telnet printer management exit is called for any printer session, regardless of whether the client LU name and IP address are defined by a TN3270E statement in the TCP/IP configuration file. Existing exits may need to be changed to accommodate this behavior.

5.2.4.19 Telnet Server and CLient

5.2.4.19.1 Changes Introduced in TCP/IP Level 530

- The **INTERNALCLIENTPARMS** statement, used for configuring the Telnet server, has been updated to accommodate Dynamic SSL/TLS support. Changes include support for **SECURECONNECTION** and **TLSLABEL** operands.
- The Telnet client has been updated to accommodate Dynamic SSL/TLS support. Changes to provide this capability include support for additional command options, which include:
  - **CERTFULLCHECK**
  - **CERTNOCHECK**
  - **NOSECURE**
  - **SECURE**

- A new statement, **SECURETELNETCLIENT**, may now be specified in the TCPIP DATA file. This statement provides the default Telnet client security value to use when neither the SECURE nor NOSECURE option is specified on the Telnet command.

5.2.4.19.2 Changes Introduced in TCP/IP Level 420

- The default for the **SCANINTERVAL** operand of the **INTERNALCLIENTPARMS** statement (used for Telnet server configuration) has been changed from 120 to 60 seconds.

5.2.4.19.3 Changes Introduced in TCP/IP Function Level 320

- The Telnet session connection exit interface has been changed to pass the LU name supplied by a client (if any) as an additional parameter. Existing exits may need to be changed to accommodate this behavior.
- The Telnet printer management exit is called for any printer session, regardless of whether the client LU name and IP address are defined by a TN3270E statement in the TCP/IP configuration file. Existing exits may need to be changed to accommodate this behavior.

5.3 DASD Storage and User ID Requirements

Figure 13 on page 48 lists the user IDs and minidisks used to install and service TCP/IP for z/VM.

Important Installation Notes:

1. The user IDs necessary for installing and using TCP/IP for z/VM have been defined as part of the installed z/VM version 5 release 3 System Deliverable. Likewise, all required minidisks have been defined. These resources have been listed in Figure 13 so you are aware of the resources that have been allocated on your behalf.

   For information about specific user ID directory entry requirements, consult the **5VMTCP30 PLANINFO** file. This file is located on the 5VMTCP30 191 minidisk.

   Note — z/VM Automated Service Procedure

   If you modify any of the IBM-supplied user IDs, minidisk addresses, or SFS directory names that are associated with TCP/IP for z/VM and you plan on using the z/VM automated service procedure (the **SERVICE** and **PUT2PROD** commands) to service your z/VM system, then you must create a PPF override for the **SERVP2P $PPF** file.

   You must also use the **VMFUPDAT** command to update the VM SYSSUF software inventory file, so that your PPF override of SERVP2P PPF is used for automated service processing. For more information about PPF file overrides, see the *z/VM: VMSES/E Introduction and Reference*

2. The **5VMTCP30** user ID is the IBM-supplied user ID for installing and servicing TCP/IP for z/VM. If you choose to use a different user ID or you elect to use different minidisks and/or SFS directories for TCP/IP for z/VM maintenance purposes, review the information presented in Appendix C, “Modifying the TCP/IP for z/VM Default Installation” on page 131 prior to making any changes.

3. The **5VMTCP30** user ID must have ***file pool administration authority*** for the **VMSYS** file pool. Such authorization is necessary to accommodate service update processing for LDAP server components that reside in the z/VM Byte File System (BFS).

   For the z/VM version 5 release 3 System Deliverable, the 5VMTCP30 user ID has already been enrolled as a file pool administrator for the **VMSYS** file pool. If you choose to use a different user ID to service TCP/IP for z/VM, or elect to use a file pool other than VMSYS for maintaining the Byte File System, you then will need to enroll the appropriate user ID as an administrator for the applicable file pool.

4. The minidisks that are associated with the **5VMTCP30**, **TCPMAINT**, and **TCPIP** user IDs (or your chosen equivalents) **must** be maintained in order to provide TCP/IP services for your installation. The remaining user IDs listed in Figure 13 are associated with servers and clients that provide optional TCP/IP for z/VM services. If you choose to not use a particular optional service, you need not maintain the user IDs and production minidisks associated with that service. However, be sure to review the information presented in Appendix C, “Modifying the TCP/IP for z/VM Default Installation” on page 131 prior to making any changes.

5. If you choose to use user IDs for TCP/IP server virtual machines that differ from the IBM-supplied user IDs shown in Figure 13, review the section titled “Implications of Assigning Different Server Virtual Machine Names” in Chapter 1 of *TCP/IP Planning and Customization*. Also, review the information presented in Appendix C, “Modifying the TCP/IP for z/VM Default Installation” on page 131.

6. Note the following, with regard to the user ID and minidisk information provided in Figure 13:

Specific Minidisk Requirements

Certain minidisks **must** be defined for the TCP/IP server machines you choose to use, as well as for maintaining TCP/IP for z/VM for your installation. Such minidisks **cannot** be replaced with an equivalent SFS directory.

Minidisks to which this requirement applies are listed in Figure 13 with **boldface** virtual device numbers. In addition, dashes are present in place of numeric SFS 4K block values, and no default SFS directory names are provided as part of these entries.

Also, the minidisks defined using the aforementioned set of (boldface) virtual device numbers must be available to their respective user IDs with **Read/Write** (**R/W**) status, when those user IDs are in use. Read/Write access to a 191 minidisk (or another, supplementary minidisk, such as the 203 minidisk defined for the SSL server) is necessary so that writeable “work space” and other data critical to the operation of a given server are available.

Note that for the TCPMAINT user ID, R/W access to its 198, 591, and 592 disks is necessary only when these disks (or the files that reside upon them) are updated for customization purposes. Otherwise, Read-Only (R/O) access to these minidisks is sufficient for this user ID (as is the usual case for the various TCP/IP server user IDs). For TCP/IP client users, Read-Only (R/O) access to only the TCPMAINT 592 minidisk is necessary.

7. **All** 5VMTCP30 ***test build*** minidisks **must** be maintained. If the 5VMTCP30 minidisks for optional services are not maintained, problems will be encountered during installation and service.

   If you choose to eliminate any of the resources which correspond to TCP/IP services that are not required for your installation, review the considerations presented in Appendix C, “Modifying the TCP/IP for z/VM Default Installation” on page 131 prior to making any changes.

8. Additional storage may need to be allocated for a given user ID or server minidisk, depending on your installation. Some examples of minidisks that may need to increased, and possible reasons for so doing, are listed in Figure 11. Note that certain minidisks (not cited here) may also need to be increased to accommodate the logging of tracing or other activities.

Figure 11. Alternate Minidisk Storage Requirements

User ID / Minidisk	Rationale for Storage Revision
SMTP 191	Allow for SMTP processing of a high volume of e-mail
VMNFS 191	Provide support for a large number of NFS clients
DHCPD 191	Provide support a large number of DHCP clients
LPSERVE 191	Allow for LPD processing of sizeable print jobs
SSLSERV 201	Facilitate installation of a Linux kernel and file system
5VMTCP30 2D2	Facilitate maintenance of multiple service levels of (binary) RPM files used by the SSL server

9. The storage requirements for various TCP/IP minidisks may be revised over time to account for TCP/IP for z/VM content changes. In some cases, this may require the size of existing minidisks to be increased.

   When applicable, storage requirement changes for specific minidisks are identified in Figure 13, in any updated levels of this document. Such changes will be noted through the use of revision characters (usually a vertical bar — "**|**" in the left margin of a page).

   For other minidisks, storage requirement changes will need to be assessed locally, for your specific environment. For example, the capacity of the **DELTA** minidisk (**5VMTCP30 2D2**, by default), periodically may need to be increased, based on the specific preventive and corrective service applied to your system.

10. If you choose to provide Network Database services, you may find the need to define multiple virtual machines, named NDBSRV01, NDBSRV02, etc. Each NDBSRV*nn* virtual machine you create should be defined similar to NDBSRV01, which has been defined as part of the installed z/VM version 5 release 3 System Deliverable.

11. If you choose to provide remote execution services through use of the rexec daemon (REXECD), you may find the need to define multiple agent virtual machines, named RXAGENT1, RXAGENT2, etc. Each RXAGENT*n* virtual machine you create should be defined similar to RXAGENT1, which has been defined as part of the installed z/VM version 5 release 3 System Deliverable. However, note that the RXAGENT*n* virtual machines do not "own" any minidisks.

12. To use the Secure Socket Layer (SSL) server, a suitably configured Linux kernel and file system must be installed on your z/VM system. Information about Linux requirements and preparing Linux for use with the SSL server is available at the TCP/IP for z/VM home page on the World Wide Web. The URL for this home page is:

    ```
    www.vm.ibm.com/related/tcpip/
    ```

13. Source files are supplied in packed format. If you intend to unpack source files after installation, ensure sufficient space is allocated for the unpacked files. Alternate storage requirements for storing unpacked files on the TCP/IP **SOURCE** minidisk (**5VMTCP30 2B3**, by default) are listed in Figure 12:

Figure 12. 5VMTCP30 2B3 Minidisk Storage Requirements — Unpacked Source Files

Type of Storage	Alternate Storage Requirement
3390 DASD	116 cylinders
FBA Device	167040 FB-512 blocks
SFS Directory	20867 SFS 4K blocks

14. For information about copying client code to the Product Code minidisk, see Appendix H, "Copying TCP/IP for z/VM Client Code to the Y-Disk" on page 152.

5.3.1 DASD Requirements for TCP/IP for z/VM

Figure 13 (Page 1 of 4). DASD Storage Requirements for Target Minidisks - TCP/IP for z/VM

Minidisk owner (User ID)	Default Device Number	Storage in Cylinders		FB-512 Blocks	SFS 4K Blocks	Usage
		DASD	CYLS			Default SFS Directory Name
5VMTCP30	191	3390	20	28800	3600	5VMTCP30 user ID 191 minidisk **VMSYS:5VMTCP30**
5VMTCP30	2B2	3390	115	165600	20590	Contains all base code shipped with TCP/IP for z/VM **VMSYS:5VMTCP30.TCPIP.OBJECT**
5VMTCP30	2B3	3390	59	84960	10534	Source files disk. (3*) **VMSYS:5VMTCP30.TCPIP.SOURCE**
5VMTCP30	29D	3390	5	7200	750	Contains TCP/IP CMS Help files **VMSYS:5VMTCP30.TCPIP.HELP**
5VMTCP30	2C4	3390	5	7200	750	Contains local modifications **VMSYS:5VMTCP30.TCPIP.LOCAL**
5VMTCP30	2D2	3390	117	168480	20900	Contains serviced files (3*) **VMSYS:5VMTCP30.TCPIP.DELTA**
5VMTCP30	2A6	3390	5	7200	750	Contains AUX files and software inventory tables that represent the test service level of TCP/IP for z/VM **VMSYS:5VMTCP30.TCPIP.APPLYALT**

Notes:

1. Cylinder values defined in this table are based on a 4K block size. FB-512 block and SFS values are derived from the 3390 cylinder values in this table. FBA minidisk sizes are shown in 512-byte blocks; these minidisks should be CMS formatted at 1K size.
2. For installation to SFS directories, a total of **65704** 4K blocks are required.
3. Additional storage may need to be allocated for some minidisks, depending on your environment. For more information, see the accompanying notes on page 44.
4. See Appendix H, “Copying TCP/IP for z/VM Client Code to the Y-Disk” on page 152 for information about copying client code to the Product Code minidisk

Figure 13 (Page 2 of 4). DASD Storage Requirements for Target Minidisks - TCP/IP for z/VM

Minidisk owner (User ID)	**Default Device Number**	**Storage in Cylinders**		**FB-512 Blocks**	**SFS 4K Blocks**	**Usage**
		DASD	**CYLS**			**Default SFS Directory Name**
5VMTCP30	2A2	3390	5	7200	750	Contains AUX files and software inventory tables that represent the service level of TCP/IP for z/VM that is currently in production **VMSYS:5VMTCP30.TCPIP.APPLYPROD**
5VMTCP30	493	3390	40	57600	7080	***Test*** build disk for various binary files; only Linux RPM files are maintained on this disk at this time **VMSYS:5VMTCP30.TCPIP.BINARY**
5VMTCP30	**491**	3390	61	87840	_____	***Test*** build disk for server code; files from this disk are copied to a production disk (TCPMAINT 591) which also requires this amount of free space
5VMTCP30	**492**	3390	70	100800	_____	***Test*** build disk for client code; files from this disk are copied to a production disk (TCPMAINT 592) which also requires this amount of free space
TCPMAINT	**191**	3390	7	10080	_____	TCPMAINT user ID 191 minidisk
TCPMAINT	**198**	3390	9	12960	_____	Contains configuration files for clients and servers.
TCPMAINT	**591**	3390	61	87840	_____	***Production*** build disk for server code
TCPMAINT	**592**	3390	70	100800	_____	***Production*** build disk for client code
ADMSERV	**191**	3390	5	7200	_____	ADMSERV user ID 191 minidisk
DHCPD	**191**	3390	2	2880	_____	DHCPD user ID 191 minidisk (3*)
DTCVSW1	**191**	3390	5	7200	_____	DTCVSW1 user ID 191 minidisk
DTCVSW2	**191**	3390	5	7200	_____	DTCVSW2 user ID 191 minidisk

Notes:

1. Cylinder values defined in this table are based on a 4K block size. FB-512 block and SFS values are derived from the 3390 cylinder values in this table. FBA minidisk sizes are shown in 512-byte blocks; these minidisks should be CMS formatted at 1K size.
2. For installation to SFS directories, a total of **65704** 4K blocks are required.
3. Additional storage may need to be allocated for some minidisks, depending on your environment. For more information, see the accompanying notes on page 44.
4. See Appendix H, “Copying TCP/IP for z/VM Client Code to the Y-Disk” on page 152 for information about copying client code to the Product Code minidisk

Figure 13 (Page 3 of 4). DASD Storage Requirements for Target Minidisks - TCP/IP for z/VM

Minidisk owner (User ID)	Default Device Number	Storage in Cylinders		FB-512 Blocks	SFS 4K Blocks	Usage
		DASD	**CYLS**			**Default SFS Directory Name**
FTPSERVE	**191**	3390	9	12960	_____	FTPSERVE user ID 191 minidisk
IMAP	**191**	3390	1	1440	_____	IMAP user ID 191 minidisk
IMAPAUTH	**191**	3390	6	8640	_____	IMAPAUTH user ID 191 minidisk
LDAPSRV	**191**	3390	5	7200	_____	LDAPSRV user ID 191 minidisk
LPSERVE	**191**	3390	2	2880	_____	LPSERVE user ID 191 minidisk (3*)
MPROUTE	**191**	3390	2	2880	_____	MPROUTE user ID 191 minidisk
NAMESRV	**191**	3390	2	2880	_____	NAMESRV user ID 191 minidisk
NDBPMGR	**191**	3390	1	1440	_____	NDBPMGR user ID 191 minidisk
NDBSRV01	**191**	3390	1	1440	_____	NDBSRV01 user ID 191 minidisk
PORTMAP	**191**	3390	2	2880	_____	PORTMAP user ID 191 minidisk
REXECD	**191**	3390	2	2880	_____	REXECD user ID 191 minidisk
RXAGENT1	_____	3390	__	_____	_____	REXEC agent (a 191 minidisk is **not** required; REXEC agents utilize the REXECD 191 minidisk)
SMTP	**191**	3390	25	36000	_____	SMTP user ID 191 minidisk (3*)
SNALNKA	**191**	3390	3	4320	_____	SNALNKA user ID 191 minidisk
SNMPD	**191**	3390	2	2880	_____	SNMPD user ID 191 minidisk
SNMPQE	**191**	3390	2	2880	_____	SNMPQE user ID 191 minidisk
SNMPSUBA	**191**	3390	2	2880	_____	SNMPSUBA user ID 191 minidisk
SSLSERV	**191**	3390	1	1440	_____	SSLSERV user ID 191 minidisk
SSLSERV	**201**	3390	1	1440	_____	SSLSERV user ID 201 minidisk (3*)
SSLSERV	**203**	3390	1	1440	_____	SSLSERV user ID 203 minidisk
TCPIP	**191**	3390	5	7200	_____	TCPIP user ID 191 minidisk

Notes:

1. Cylinder values defined in this table are based on a 4K block size. FB-512 block and SFS values are derived from the 3390 cylinder values in this table. FBA minidisk sizes are shown in 512-byte blocks; these minidisks should be CMS formatted at 1K size.
2. For installation to SFS directories, a total of **65704** 4K blocks are required.
3. Additional storage may need to be allocated for some minidisks, depending on your environment. For more information, see the accompanying notes on page 44.
4. See Appendix H, “Copying TCP/IP for z/VM Client Code to the Y-Disk” on page 152 for information about copying client code to the Product Code minidisk

Figure 13 (Page 4 of 4). DASD Storage Requirements for Target Minidisks - TCP/IP for z/VM

Minidisk owner (User ID)	**Default Device Number**	**Storage in Cylinders**		**FB-512 Blocks**	**SFS 4K Blocks**	**Usage**
		DASD	**CYLS**			**Default SFS Directory Name**
TFTPD	**191**	3390	2	2880	_____	TFTPD user ID 191 minidisk
UFTD	**191**	3390	2	2880	_____	UFTD user ID 191 minidisk
VMKERB	**191**	3390	6	8640	_____	VMKERB user ID 191 minidisk
VMNFS	**191**	3390	9	12960	_____	VMNFS user ID 191 minidisk (3*)
X25IPI	**191**	3390	2	2880	_____	X25IPI user ID 191 minidisk

Notes:

1. Cylinder values defined in this table are based on a 4K block size. FB-512 block and SFS values are derived from the 3390 cylinder values in this table. FBA minidisk sizes are shown in 512-byte blocks; these minidisks should be CMS formatted at 1K size.
2. For installation to SFS directories, a total of **65704** 4K blocks are required.
3. Additional storage may need to be allocated for some minidisks, depending on your environment. For more information, see the accompanying notes on page 44.
4. See Appendix H, “Copying TCP/IP for z/VM Client Code to the Y-Disk” on page 152 for information about copying client code to the Product Code minidisk

6.0 Installation Instructions

This section describes the method by which TCP/IP for z/VM is installed and provides step-by-step procedures to complete the installation process.

The procedures that follow are presented in two-column format, where the steps to be performed are identified using numbered, **boldface** headings. Any sub-steps that correspond to a given procedure are presented on the right side of each page and are ordered using bold numerals, while the commands associated with these steps are presented on the left side of a page. Pertinent command information may exist to the right of a given command.

Each step of these installation instructions must be followed. Do not skip any step unless directed otherwise.

Throughout these instructions, the use of IBM-supplied default minidisk device numbers and user IDs is assumed. If different user IDs, device numbers, or SFS directories are used to install TCP/IP for z/VM in your environment, adapt these instructions as needed.

Note!

Any sample console output presented throughout these instructions is based on a z/VM version 5 release 3 system; this output reflects an installation environment in which default values (PPF and component names, user IDs, and minidisks) are in use.

6.1 TCP/IP for z/VM Installation Process Overview

A brief description of the steps necessary to complete the installation of TCP/IP for z/VM follows:

- **Review the Default Installation** — Various resources have been defined and allocated for TCP/IP for z/VM, as part of the installed z/VM version 5 release 3 System Deliverable. This default environment should be reviewed and, if necessary, modified for your installation.
- **Review TCP/IP for z/VM Content and Changes** — Review the topics presented in 5.2.4, “Migration Considerations” on page 17, so you are aware of changes that may affect your customization and use of TCP/IP level 530.
- **Configure TCP/IP for z/VM** — The configuration files associated with various TCP/IP services must be customized to effectively use TCP/IP for z/VM.

For a complete description of all VMSES/E installation commands, operands and options, refer to:

- *z/VM: VMSES/E Introduction and Reference* (GC24-6130)

6.2 Installing TCP/IP for z/VM

Note — All z/VM Customers

The material presented in the next few sections is provided mostly for informational and reference purposes. To complete the installation of TCP/IP for z/VM, continue with the instructions in section 6.2.3, “Configure TCP/IP for z/VM for Your Installation” on page 57.

6.2.1 Review the TCP/IP for z/VM Default Installation Environment

Because TCP/IP for z/VM has been installed as part of the z/VM version 5 release 3 System Deliverable, several installation steps have already been performed on your behalf. Among these are the:

- addition of TCP/IP-specific user ID entries and PROFILES to the z/VM version 5 release 3 system directory
- creation of a simplified PROFILE EXEC for the 5VMTCP30 user ID
- allocation of TCP/IP-required minidisks
- loading of TCP/IP for z/VM product files (run-time and sample configuration files) to test build *and* production minidisks, using VMSES/E commands.

6.2.1.1 PPF Override and Other Modification Considerations

The file name (or, *ppfname*) of IBM-supplied Product Parameter File (PPF) for TCP/IP for z/VM is **5VMTCP30**. This file has been installed (and used) as part of the z/VM version 5 release 3 System Deliverable installation. The **5VMTCP30** *ppfname* is also assumed and referenced throughout section 7.0, “Service Instructions.”

If you create your own TCP/IP for z/VM PPF override file, use the *ppfname* of your override file (instead of 5VMTCP30) throughout any procedures that require this file to be identified, unless noted otherwise.

Note — z/VM Automated Service Procedure

If you modify any of the IBM-supplied user IDs, minidisk addresses, or SFS directory names that are associated with TCP/IP for z/VM and you plan on using the z/VM automated service procedure (the **SERVICE** and **PUT2PROD** commands) to service your z/VM system, then you must create a PPF override for the **SERVP2P $PPF** file.

You must also use the **VMFUPDAT** command to update the VM SYSSUF software inventory file, so that your PPF override of SERVP2P PPF is used for automated service processing. For more information about PPF file overrides, see the *z/VM: VMSES/E Introduction and Reference*

As *TCP/IP Planning and Customization* is reviewed and used to configure TCP/IP, you may also identify TCP/IP services that are not required for your installation. If you choose to eliminate the resources that correspond to these services, review the considerations presented in Appendix C, “Modifying the TCP/IP for z/VM Default Installation” on page 131 prior to making any changes.

6.2.1.2 TCP/IP for z/VM Directory PROFILES and User IDs

Two system directory PROFILE entries (PROFILE TCPCMSU and PROFILE TCPGCSU) have been added to the z/VM version 5 release 3 system directory for TCP/IP for z/VM; these entries are shown in Figure 14. Each directory entry supplied for a TCP/IP for z/VM service virtual machine includes one of these profiles.

Figure 14. TCP/IP for z/VM System Directory Profiles

```
PROFILE TCPCMSU
  IPL CMS
  MACHINE XA
  SPOOL 00C 2540 READER *
  SPOOL 00D 2540 PUNCH  A
  SPOOL 00E 1403 A
  CONSOLE 009 3215 T
  LINK MAINT 190 190 RR
  LINK MAINT 19D 19D RR
  LINK MAINT 19E 19E RR
  LINK MAINT 0401 0401 RR
  LINK MAINT 0402 0402 RR
  LINK MAINT 0405 0405 RR
```

```
PROFILE TCPGCSU
  IPL GCSXA PARM AUTOLOG
  MACHINE XA
  NAMESAVE GCS
  SPOOL 00C 2540 READER *
  SPOOL 00D 2540 PUNCH  A
  SPOOL 00E 1403 A
  CONSOLE 009 3215 T
  LINK MAINT 190 190 RR
  LINK MAINT 19D 19D RR
  LINK MAINT 19E 19E RR
  LINK MAINT 0401 0401 RR
  LINK MAINT 0402 0402 RR
  LINK MAINT 0405 0405 RR
```

Notes:

1. Links to the MAINT 401, 402 and 405 minidisks are established to facilitate the use of CMS Kanji, German and Upper Case American English HELP files, for those environments in which these may be required.
2. The `NAMESAVE GCS` statement can be removed from PROFILE TCPGCSU if the GCS saved segment is not restricted.

The user IDs that have been defined for TCP/IP for z/VM are listed in Figure 15 on page 55.

Note: When installation of the z/VM version 5 release 3 System Deliverable has been completed, the login password for a given TCP/IP user ID is identical to that same user ID. If you have not already done so, change these passwords to valid passwords, in accordance with your security guidelines.

Additional User ID Notes:

1. For information about specific user ID directory entry requirements, consult the **5VMTCP30 PLANINFO** file. This file is located on the 5VMTCP30 191 minidisk.
2. The directory entries supplied for each TCP/IP for z/VM service virtual machine now include LINK statements for the 5VMTCP30 491 and 492 minidisks. These minidisk links have been added to better facilitate the testing of newly applied service.
3. The directory entry for the TCPIP virtual machine includes the statement: `SHARE RELATIVE 3000`

   For most installations, the relative CPU share allocation of 3000 should be suitable. However, you are free to change this value to conform to local guidelines established for defining server and guest virtual machine share settings.
4. If you create additional RXAGENT*n* machines, duplicate the RXAGENT1 directory entry for each server you add.
5. If you create additional NDBSRV*nn* machines, duplicate the NDBSRV01 directory entry for each server you add, and include an appropriate LINK statement in the 5VMTCP30 directory entry for each new server 191 minidisk that is created.

Formatting Reminder

Any additional minidisks you create must be formatted before you continue with the installation of TCP/IP for z/VM.

Figure 15 (Page 1 of 2). Default User IDs - TCP/IP for z/VM

TCP/IP User ID	Associated TCP/IP Function
5VMTCP30	Manages the TCP/IP system (component code and service updates).
TCPMAINT	TCP/IP system administration and configuration.
ADMSERV	Runs the Kerberos database remote administration server.
DHCPD	Responds to client requests for boot information using data defined in a DHCPD machine file.
DTCVSW1	System-default VSWITCH controller virtual machine.
DTCVSW2	System-default VSWITCH controller virtual machine (alternate, for provision of failover capability)
FTPSERVE	Implements the File Transfer Protocol (FTP) daemon, which controls access to files on the local host.
IMAP	Implements the Internet Message Access Protocol (IMAP) daemon, which allows a client to access and manipulate electronic mail messages on a server.
IMAPAUTH	Performs IMAP user authentication, when the IMAP server has been configured to make use of the IMAP Authentication Exit.

Notes:

1. Additional changes may need to be made for some user IDs, depending on your environment. For more information, see the accompanying notes on page 55.

Figure 15 (Page 2 of 2). Default User IDs - TCP/IP for z/VM

TCP/IP User ID	**Associated TCP/IP Function**
LDAPSRV	Implements the Lightweight Directory Access Protocol (LDAP) server.
LPSERVE	Implements the Line Printer Daemon (LPD), which handles client requests to print a file.
MPROUTE	Implements the Multiple Protocol Routing (MPRoute) server, which uses OSPF and/or RIP protocols to manage network routing information.
NAMESRV	Implements the Domain Name System (DNS) server.
NDBPMGR	Provides Network Database (NDB) Port Manager support, and is used in conjunction with the NDBSRV01 server.
NDBSRV01 (1*)	Provides Network Database (NDB) System support, in conjunction with the NDB Port Manager (NDBPMGR) server.
PORTMAP	Runs the Portmapper function for RPC systems that support the Network File System protocol.
REXECD	Provides remote execution services for TCP/IP hosts that support the REXEC client.
RXAGENT1 (1*)	Agent virtual machine used by REXECD to process anonymous rexec client requests.
SMTP	Implements the Simple Mail Transfer Protocol (SMTP) server, which provides TCP/IP electronic mail support.
SNALNKA	Provides SNA LU 0 connections between multiple hosts.
SNMPD	Virtual machine for the SNMP Agent.
SNMPQE	Virtual machine for the SNMP Query Engine.
SNMPSUBA	Subagent virtual machine for the SNMP Query Engine.
SSLSERV	Provides Secure Sockets Layer (SSL) protocol support for TCP/IP servers.
TCPIP (1*)	Primary virtual machine that provides TCP/IP and Telnet services.
TFTPD	Transfers files between the Byte File System (BFS) and TFTP clients.
UFTD	Implements the Unsolicited File Transfer (UFT) server.
VMKERB	Runs the Kerberos authentication server.
VMNFS	Implements the Network File System (NFS) server.
X25IPI	Provides an interface which allows the TCPIP virtual machine to communicate with hosts that use the X.25 protocol.

Notes:

1. Additional changes may need to be made for some user IDs, depending on your environment. For more information, see the accompanying notes on page 55.

6.2.2 Move TCP/IP for z/VM to SFS Directories (Optional)

If TCP/IP for z/VM was installed to minidisks during installation of the z/VM version 5 release 3 System Deliverable, you still can move TCP/IP for z/VM service minidisks to Shared File System (SFS) directories, at a time of your choosing. See Appendix G, “Moving TCP/IP for z/VM to SFS Directories” on page 147 for instructions on how this can be done.

6.2.3 Configure TCP/IP for z/VM for Your Installation

As previously mentioned, upon installation of the z/VM version 5 release 3 System Deliverable, the various program files that comprise TCP/IP for z/VM reside on appropriate production minidisks. In addition, representative client and server *sample* configuration files are also present. See 6.2.3.5, “TCP/IP for z/VM Product and Sample Configuration Files” on page 65 for more information about these files and their default location.

Before any TCP/IP services can be used, certain configuration files **must** be created and customized for your installation. See *TCP/IP Planning and Customization* (SC24-6125) for detailed information about the various TCP/IP services that can be established, and the configuration files that are associated with each service. For convenience, the TCP2PROD command can *optionally* be used to create an initial set of configuration files, as described in the next section; such may serve as a starting point for customizing TCP/IP services for your installation. For reference, the sample configuration files supplied by IBM are summarized in Figure 19 on page 69.

IPWIZARD Considerations

If the IPWIZARD command has been used to create an initial TCP/IP configuration, the following files have been created *and* customized:

- PROFILE TCPIP
- SYSTEM DTCPARMS
- TCPIP DATA

These files enable basic network connectivity for your z/VM system, with their content based on information supplied via the IPWIZARD panels. If you intend to provide more comprehensive TCP/IP services for your installation, further customization of the previously listed files is required. Additional TCP/IP configuration files will also require customization, dependent upon the specific services that are to be established.

Note: If the IPWIZARD command has **not** been used, the previously listed files are not present on your system.

6.2.3.1 Create System-default VSWITCH Controller Profiles

Note

If you received an RSU with your z/VM version 5 release 3 System Deliverable and have installed this RSU, the VSWITCH controller configuration files described in this section will have been created on your behalf. In this instance, do not use the steps that follow. Instead, continue the configuration process as described in the next section (6.2.3.2, “Create a Starter Set of TCP/IP Configuration Files (Optional)” on page 61).

The z/VM version 5 release 3 System Deliverable incorporates definitions and automatic start-up processing for a pair of system-default Virtual Switch (VSWITCH) controller virtual machines — **DTCVSW1** and **DTCVSW2** — which require appropriate configuration files to be in place to allow for their correct operation when the system is IPLed. Such configuration files are created automatically as part of the z/VM version 5 release 3 System Deliverable installation process, when an installation-supplied RSU is loaded. However, **if an RSU is not provided with your z/VM version 5 release 3 System Deliverable**, you then must create these files on your own accord.

Notes:

1. If the necessary configuration files are *not* present, the DTCVSW1 and DTCVSW2 VSWITCH controller virtual machines will not operate in the correct manner and will not serve their intended function. If necessary, refer to *z/VM: Connectivity* for more information about Virtual Switches, and their definition and use in a z/VM environment.
2. For step 5 below, it is assumed that the content of the TCPPRECONFIG section of the 5VMTCP30 CATALOG does not require modification for your installation. If this is not the case, you should make any necessary changes to this section of the 5VMTCP30 CATALOG file before you continue with the steps that follow. See Appendix A, “TCP/IP Installation, Service and Migration Utilities” on page 104 for detailed information about the TCP2PROD command and TCP/IP for z/VM catalog files.

Use the steps that follow to create appropriate server configuration files for the **DTCVSW1** and **DTCVSW2** VSWITCH controller virtual machines.

1 Log on the TCP/IP for z/VM service user ID, **5VMTCP30**.

The PROFILE EXEC provided (as part of the z/VM version 5 release 3 System Deliverable) for this user ID contains ACCESS commands for VMSES/E minidisks that are necessary to run the commands cited in later steps. The minidisks required are the VMSES/E code minidisk (MAINT 5E5, by default) and the VMSES/E Software Inventory minidisk (MAINT 51D, by default).

2 Issue the CMS QUERY DISK command to verify the VMSES/E code and Software Inventory minidisks are correctly linked and accessed.

query disk

Verify the MAINT 5E5 minidisk is accessed as file mode **B**, and is linked **R/O**.

Verify the MAINT 51D minidisk is accessed as file mode **D**, and is linked **R/W**.

Note: If another user has the MAINT 51D minidisk linked in write (R/W) mode, you'll obtain only read (R/O) access to this minidisk. If this occurs, have that user re-link the 51D disk in read-only (RR) mode, after which you need issue the appropriate LINK and ACCESS commands for the 51D minidisk. Do not continue with these procedures until a R/W link is established to the 51D minidisk.

3 If necessary, establish the appropriate access to the VMSES/E minidisks.

a Establish read access to the VMSES/E code minidisk.

link maint 5e5 5e5 rr
access 5e5 b

b Establish write access to the Software Inventory minidisk.

link maint 51d 51d mr
access 51d d

4 Access the 5VMTCP30 491 and 492 minidisks.

access 491 i
access 492 j

The 491 minidisk is where the TCP2PROD EXEC and 5VMTCP30 CATALOG files reside. The DTCUME message repository (required for running TCP2PROD) resides on the 492 minidisk.

5 Review the 5VMTCP30 CATALOG file to verify its correctness, as previously suggested.

6 (*Optional*) Establish write links to any TCP/IP for z/VM production minidisks which are not yet linked in this mode.

LINK statements for the various TCP/IP for z/VM minidisks are present in the 5VMTCP30 directory entry (supplied as part of the installed z/VM version 5 release 3 System Deliverable).

If you have changed the default installation user ID or use different minidisk device numbers in your environment, you may need to manually link the necessary TCP/IP production minidisks. See Figure 16 on page 65, Figure 18 on page 67, and Figure 19 on page 69 for device link information. If you created a PPF override that has changed any of these device numbers, use your values.

link *tcpipid vdev1 vdev2* **mr**

Note: If another user has the *vdev1* minidisk linked in write (R/W) mode, you'll obtain only read (R/O) access to this minidisk. If this occurs, have that user re-link the *vdev1* disk in read-only (RR) mode, after which you need to re-issue the above LINK command. Do not continue with these procedures until a R/W link is established to the *vdev1* minidisk.

7 Create the necessary VSWITCH controller server configuration files by using the TCP2PROD command. For reference, files that can be processed using the TCPPRECONFIG section are listed in Figure 20 on page 71.

Verifying Your Environment

When you perform this step, it is suggested that you first invoke TCP2PROD as illustrated, but with the **TEST** option also specified. This will verify that all resources can be accessed and that the appropriate files will be processed.

With the **TEST** option in effect, **no files are copied**.

Resolve any reported problems, then invoke TCP2PROD (without the TEST option) as illustrated.

tcp2prod 5vmtcp30 {tcpip | tcpipsfs} 5vmtcp30 tcppreconfig (setup

Use **tcpip** if the TCP/IP for z/VM default minidisk environment has been maintained; use **tcpipsfs** if the service minidisks were moved to Shared File System directories.

8 Review the TCP2PROD message log (TCP2PROD $MSGLOG). If necessary, correct any problems before you proceed with the next step.

vmfview tcp2prod

6.2.3.2 Create a Starter Set of TCP/IP Configuration Files (Optional)

This section provides ***optional*** steps for using the TCP2PROD command to create an initial (or, “starter”) set of TCP/IP configuration files then can be customized for the TCP/IP services that are provided and used by your installation. The files created by this procedure are listed in Figure 19 on page 69.

Note that all, or a subset, of the configuration files listed in Figure 19 can be (manually) created on and individual or as-needed basis, if you choose to not use the TCP2PROD command and the steps that follow.

Note

When the TCP2PROD command is used as described here, a configuration file is created *only if the intended file does not already exist*. Existing (and presumably customized) configuration files are *not* replaced.

Notes

- The configuration files created by TCP2PROD have the same content as the *sample* files on which they are based.
- For step 8 below, it is assumed that the content of the TCPCONFIG section of the 5VMTCP30 CATALOG does not require modification for your installation. If this is not the case, you should make any necessary changes to this section of the 5VMTCP30 CATALOG file before you continue with the steps that follow. See Appendix A, “TCP/IP Installation, Service and Migration Utilities” on page 104 for detailed information about the TCP2PROD command and TCP/IP for z/VM catalog files.

1 Shutdown TCP/IP services. For most installations, this step will likely be required only if the IPWIZARD command has been used to create an initial TCP/IP configuration *and* the TCP/IP server virtual machine (TCPIP, by default) is active.

Note - TCP/IP Server Shutdown Considerations

Before you shutdown any TCP/IP servers, ensure any applicable conditions or guidelines for your installation have been followed.

For information on shutting down TCP/IP servers, see the section that discusses “Starting and Stopping TCP/IP Servers” in the chapter titled “General TCP/IP Server Configuration,” of *TCP/IP Planning and Customization*.

Note that the TCPMSMGR command can be used to manage the shutdown and initialization of the TCP/IP servers that are used by your installation. For more information about the TCPMSMGR command, see Appendix A, “TCP/IP Installation, Service and Migration Utilities” on page 104.

2 Log on the TCP/IP for z/VM service user ID, **5VMTCP30**.

The PROFILE EXEC provided (as part of the z/VM version 5 release 3 System Deliverable) for this user ID contains ACCESS commands for VMSES/E minidisks that are necessary to run the commands cited in later steps. The minidisks required are the VMSES/E code minidisk (MAINT 5E5, by default) and the VMSES/E Software Inventory minidisk (MAINT 51D, by default).

3 Issue the CMS QUERY DISK command to verify the VMSES/E code and Software Inventory minidisks are correctly linked and accessed.

```
query disk
```

Verify the MAINT 5E5 minidisk is accessed as file mode **B**, and is linked **R/O**.

Verify the MAINT 51D minidisk is accessed as file mode **D**, and is linked **R/W**.

Note: If another user has the MAINT 51D minidisk linked in write (R/W) mode, you'll obtain only read (R/O) access to this minidisk. If this occurs, have that user re-link the 51D disk in read-only (RR) mode, after which you need issue the appropriate LINK and ACCESS commands for the 51D minidisk. Do not continue with these procedures until a R/W link is established to the 51D minidisk.

4 If necessary, establish the appropriate access to the VMSES/E minidisks.

a Establish read access to the VMSES/E code minidisk.

```
link maint 5e5 5e5 rr
access 5e5 b
```

b Establish write access to the Software Inventory minidisk.

```
link maint 51d 51d mr
access 51d d
```

5 Access the 5VMTCP30 491 and 492 minidisks.

```
access 491 i
access 492 j
```

The 491 minidisk is where the TCP2PROD EXEC and 5VMTCP30 CATALOG files reside. The DTCUME message repository (required for running TCP2PROD) resides on the 492 minidisk.

6 Review the 5VMTCP30 CATALOG file to verify its correctness, as previously suggested.

7 (*Optional*) Establish write links to any TCP/IP for z/VM production or server minidisks which are not yet linked in this mode.

LINK statements for the various TCP/IP for z/VM minidisks are present in the 5VMTCP30 directory entry (supplied as part of the installed z/VM version 5 release 3 System Deliverable).

If you have changed the default installation user ID or use different minidisk device numbers in your environment, you may need to manually link the necessary TCP/IP production and server minidisks. See Figure 16 on page 65, Figure 18 on page 67, and Figure 19 on page 69 for device link information. If you created a PPF override that has changed any of these device numbers, use your values.

link *tcpipid vdev1 vdev2* **mr**

Note: If another user has the *vdev1* minidisk linked in write (R/W) mode, you'll obtain only read (R/O) access to this minidisk. If this occurs, have that user re-link the *vdev1* disk in read-only (RR) mode, after which you need to re-issue the above LINK command. Do not continue with these procedures until a R/W link is established to the *vdev1* minidisk.

8 Create initial TCP/IP for z/VM configuration files by using the TCP2PROD command. For reference, files that can be processed using the TCPCONFIG section are listed in Figure 19 on page 69.

> **Verifying Your Environment**
>
> When you perform this step, it is suggested that you first invoke TCP2PROD as illustrated, but with the **TEST** option also specified. This will verify that all resources can be accessed and that the appropriate files will be processed.
>
> With the **TEST** option in effect, **no files are copied**.
>
> Resolve any reported problems, then invoke TCP2PROD (without the TEST option) as illustrated.

```
tcp2prod 5vmtcp30 {tcpip | tcpipsfs} 5vmtcp30 tcpconfig (setup
```

Use **tcpip** if the TCP/IP for z/VM default minidisk environment has been maintained; use **tcpipsfs** if the service minidisks were moved to Shared File System directories.

9 Review the TCP2PROD message log (TCP2PROD $MSGLOG). If necessary, correct any problems before you proceed with the next step.

```
vmfview tcp2prod
```

6.2.3.3 Initialize TCP/IP Services

Once TCP/IP for z/VM has been (fully) configured for your installation, the appropriate TCP/IP servers must be initialized. For more information, see the section that discusses “Starting and Stopping TCP/IP Servers” in the chapter titled “General TCP/IP Server Configuration,” of *TCP/IP Planning and Customization*.

If the TCPMSMGR command was previously used to manage the shutdown of the TCP/IP servers used by your installation, it now can be used to initialize those servers. For more information about the TCPMSMGR command, see Appendix A, “TCP/IP Installation, Service and Migration Utilities” on page 104.

6.2.3.4 Copy TCP/IP Client Code to the z/VM Product Code Disk (Optional)

After TCP/IP for z/VM has been configured for your installation, you may want to consider copying TCP/IP client code (or a subset of this) to the z/VM Product Code minidisk. See Appendix H, “Copying TCP/IP for z/VM Client Code to the Y-Disk” on page 152 for additional information and instructions concerning this process.

6.2.3.5 TCP/IP for z/VM Product and Sample Configuration Files

Figure 16 lists the TCP/IP for z/VM product files that must reside on individual TCP/IP server virtual machine (SVM) minidisks. The production locations shown are those established with installation of the z/VM version 5 release 3 System Deliverable. For reference, source and production file naming information is provided as well.

Notes:

1. The *Link Device* numbers cited in Figure 16 correspond to LINK statement defaults for minidisks listed in the *Production Location* column. These LINK defaults are defined for the 5VMTCP30 user ID, in the system (CP) directory entry that is supplied as part of the z/VM version 5 release 3 System Deliverable.
2. Because of their use and composition, the files listed in Figure 16 and Figure 17 on page 66 ***usually*** are not processed or updated when TCP/IP for z/VM service is applied to your system.

   However, should the need arise to process these files — such as to restore them to their base-level, unmodified state for unique or extenuating circumstances, or if notification of a service change to these parts is received — this can be accomplished by using the TCP/IP for z/VM **TCP2PROD** command, with the appropriate catalog section name (**tcpsvmcms** or **tcpsvmgcs**) specified as an operand. For more information, see Appendix I, “Managing TCP/IP Files with Unique Service Requirements” on page 155.

Figure 16. TCP/IP for z/VM Production Run-Time Files (CMS SVM-Specific)

Source Location (1*)	LINK Device Number (2*)	Source File Name / Type	Production File Name / Type	Production Location
491	262	TCPROFIL EXEC (3*)	PROFILE EXEC	TCPIP 191
491	263	TCPROFIL EXEC	PROFILE EXEC	ADMSERV 191
491	265	TCPROFIL EXEC	PROFILE EXEC	DHCPD 191
491	266	TCPROFIL EXEC	PROFILE EXEC	FTPSERVE 191
491	267	TCPROFIL EXEC	PROFILE EXEC	LPSERVE 191
491	268	TCPROFIL EXEC	PROFILE EXEC	MPROUTE 191
491	269	TCPROFIL EXEC	PROFILE EXEC	NAMESRV 191
491	26A	TCPROFIL EXEC	PROFILE EXEC	NDBPMGR 191
491	26B	TCPROFIL EXEC	PROFILE EXEC	NDBSRV01 191
491	26C	TCPROFIL EXEC	PROFILE EXEC	PORTMAP 191
491	26D	TCPROFIL EXEC	PROFILE EXEC	REXECD 191
491	26F	TCPROFIL EXEC	PROFILE EXEC	SMTP 191
491	271	TCPROFIL EXEC	PROFILE EXEC	SNMPD 191
491	272	TCPROFIL EXEC	PROFILE EXEC	SNMPQE 191
491	273	TCPROFIL EXEC	PROFILE EXEC	SSLSERV 191
491	279	TCPROFIL EXEC	PROFILE EXEC	TFTPD 191
491	27A	TCPROFIL EXEC	PROFILE EXEC	UFTD 191
491	27B	TCPROFIL EXEC	PROFILE EXEC	VMKERB 191
491	27C	TCPROFIL EXEC	PROFILE EXEC	VMNFS 191
491	27E	TCPROFIL EXEC	PROFILE EXEC	IMAP 191
491	27F	TCPROFIL EXEC	PROFILE EXEC	IMAPAUTH 191
491	280	TCPROFIL EXEC	PROFILE EXEC	DTCVSW1 191
491	281	TCPROFIL EXEC	PROFILE EXEC	DTCVSW2 191
491	282	TCPROFIL EXEC	PROFILE EXEC	SNMPSUBA 191
491	283	TCPROFIL EXEC	PROFILE EXEC	LDAPSRV 191

Notes:

1. Source minidisks owned by the 5VMTCP30 user ID.
2. LINK defaults for the 5VMTCP30 user ID.
3. The TCPROFIL EXEC file should be copied to the 191 disk of any additional CMS-based TCP/IP for z/VM servers that are installed (or added at a later time).
4. TCPROFIL EXEC profiles are not interchangeable with the TCPROFIL GCS files used for TCP/IP for z/VM GCS-based servers.

Figure 17. TCP/IP for z/VM Production Run-Time Files (GCS SVM-Specific)

Source Location (1*)	**LINK Device Number** (2*)	**Source File Name / Type**	**Production File Name / Type**	**Production Location**
491	270	TCPROFIL GCS (3*)	PROFILE GCS	SNALNKA 191
491	27D	TCPROFIL GCS	PROFILE GCS	X25IPI 191

Notes:

1. Source minidisks owned by the 5VMTCP30 user ID.
2. LINK defaults for the 5VMTCP30 user ID.
3. The TCPROFIL GCS file should be copied to the 191 disk of any additional GCS-based TCP/IP for z/VM servers that are installed (or added at a later time).
4. TCPROFIL GCS profiles are not interchangeable with the TCPROFIL EXEC files used for TCP/IP for z/VM CMS-based servers.

Figure 18 lists the TCP/IP for z/VM product *run-time* files that must reside on TCP/IP production disks (or other z/VM minidisks) to provide and use TCP/IP services. The production locations shown are those established by the TCP/IP for z/VM **TCP2PROD** command, with **tcprun** specified as the catalog section operand. For reference, source and production file naming information is provided as well.

Note: The *Link Device* numbers cited in Figure 18 correspond to LINK statement defaults for minidisks listed in the *Production Location* column. These LINK defaults are defined for the 5VMTCP30 user ID, in the system (CP) directory entry that is supplied as part of the z/VM version 5 release 3 System Deliverable.

Figure 18. TCP/IP for z/VM Production Run-Time Files

Source Location (1*)	**LINK Device Number** (2*)	**Source File Name / Type**	**Production File Name / Type**	**Production Location**
491	591	— **All Files** —	(no change)	TCPMAINT 591
492	592	— **All Files** —	(no change)	TCPMAINT 592
492	493C	VMRPC TXTLIB	VMRPC TXTLIB	MAINT 493 (3*)
492	193C	VMRPC TXTLIB	VMRPC TXTLIB	MAINT 193 (3*)

Notes:

1. Source minidisks owned by the 5VMTCP30 user ID.
2. LINK defaults for the 5VMTCP30 user ID.
3. Required for building the DMSVSMAS MODULE

Figure 19 on page 69 lists the various TCP/IP for z/VM ***sample*** configuration files that have been provided to assist you with customization of TCP/IP services for your installation. The *sample* file locations shown are those established by the TCP/IP for z/VM **TCP2PROD** command, (with **tcpsample** specified as the catalog section operand), where as *configured* locations are those established through *optional* use of the **TCP2PROD** command, with **tcpconfig** specified as the catalog section operand. For reference, source and production file naming information provided as well.

Note: Unless noted otherwise, the minidisks listed in the *Sample Location* and *Configured Location* columns in Figure 19 on page 69 are TCP/IP for z/VM *production* minidisks owned by the TCPMAINT user ID.

Figure 19 (Page 1 of 3). TCP/IP for z/VM Sample and Configuration Files

Sample Location (1*)	Configured Location (1*, 2*)	Sample File Name / Type	Configured File Name / Type	Usage
591	198	TCPRUNXT SAMPEXEC	TCPRUNXT EXEC	TCP/IP Servers
591	198	PROFILE STCPIP	PROFILE TCPIP	TCPIP
591	198	SCEXIT SAMPEXEC	SCEXIT EXEC	TCPIP
591	198	SCEXIT SAMPASM	SCEXIT ASSEMBLE	TCPIP
591	198	PMEXIT SAMPEXEC	PMEXIT EXEC	TCPIP
591	198	PMEXIT SAMPASM	PMEXIT ASSEMBLE	TCPIP
591	191 (3*)	ADM@AADD SAMPAUTH	ADM@ACL ADD	ADMSERV
591	191 (3*)	ADM@AGET SAMPAUTH	ADM@ACL GET	ADMSERV
591	191 (3*)	ADM@AMOD SAMPAUTH	ADM@ACL MOD	ADMSERV
591	198	CHKIPADR SAMPEXEC	CHKIPADR EXEC	FTPSERVE
591	198	FTPEXIT SAMPEXEC	FTPEXIT EXEC	FTPSERVE
591	198	FTPEXIT SAMPASM	FTPEXIT ASSEMBLE	FTPSERVE
591	198	SRVRFTP SCONFIG	SRVRFTP CONFIG	FTPSERVE
591	198	IMAP SCONFIG	IMAP CONFIG	IMAP
591	198	TCPVMIPC SAMPNAME	$SERVER$ NAMES	IMAP
591	198	IMAPAUTH SAMPEXEC	IMAPAUTH EXEC	IMAPAUTH
591	198	LDAP-DS SCONFIG	DS CONF	LDAPSRV
591	198	LDAP-DS SAMPENVR	DS ENVVVAR	LDAPSRV
591	198	LPD SCONFIG	LPD CONFIG	LPSERVE
591	198	MPROUTE SCONFIG	MPROUTE CONFIG	MPRoute
591	198	NSMAIN SCACHE	NSMAIN CACHE	NAMESRV
591	198	NSMAIN SDATA	NSMAIN DATA	NAMESRV
591	198	VALIDUSR SAMPEXEC	VALIDUSR EXEC	NAMESRV
591	198	RSCSTCP SCONFIG	RSCSTCP CONFIG	RSCS
591	198	RSCSLPD SCONFIG	RSCSLPD CONFIG	RSCS (LPD)
591	198	RSCSLPR SCONFIG	RSCSLPR CONFIG	RSCS (LPD)
591	198	RSCSLPRP SCONFIG	RSCSLPRP CONFIG	RSCS (LPD)
591	198	RSCSUFT SCONFIG	RSCSUFT CONFIG	RSCS (UFT)

Notes:

1. Minidisks owned by the TCPMAINT user ID.
2. Location as ***optionally*** placed into production by the **TCP2PROD** command.
3. The ADMSERV 191 minidisk.

Figure 19 (Page 2 of 3). TCP/IP for z/VM Sample and Configuration Files

Sample Location (1*)	**Configured Location** (1*, 2*)	**Sample File Name / Type**	**Configured File Name / Type**	**Usage**
591	198	SMTP SCONFIG	SMTP CONFIG	SMTP
591	198	SMTPCMDX SAMPEXEC	SMTPCMDX EXEC	SMTP
591	198	SMTPCMDX SAMPASM	SMTPCMDX ASSEMBLE	SMTP
591	198	SMTPVERX SAMPEXEC	SMTPVERX EXEC	SMTP
591	198	SMTPVERX SAMPASM	SMTPVERX ASSEMBLE	SMTP
591	198	SMTPFWDX SAMPEXEC	SMTPFWDX EXEC	SMTP
591	198	SMTPFWDX SAMPASM	SMTPFWDX ASSEMBLE	SMTP
591	198	SMTPMEMO SAMPLE	SECURITY MEMO	SMTP
591	198	SMTPSECT SAMPTABL	SMTP SECTABLE	SMTP
591	198	SNALNKA SAMPGCS	SNALNKA GCS	SNALNKA
591	198	MIB_DESC SDATA	MIB_DESC DATA	SNMPQE
591	198	MIB_EXIT SDATA	MIB_EXIT DATA	SNMPSUBA
591	198	SNMPMIBX SAMPASM	SNMPMIBX ASSEMBLE	SNMPSUBA
591	198	UFTD SCONFIG	UFTD CONFIG	UFTD
591	198	UFTCMDX SAMPEXEC	UFTCMDX EXEC	UFTD
591	198	UFTNSLKX SAMPEXEC	UFTNSLKX EXEC	UFTD
591	198	VMNFS SCONFIG	VMNFS CONFIG	VMNFS
591	198	VMNFSCMS SAMPEXEC	VMNFSCMS EXEC	VMNFS
591	198	VMNFSSMG SAMPEXEC	VMNFSSMG EXEC	VMNFS
591	198	VMNFSMON SAMPEXEC	VMNFSMON EXEC	VMNFS
591	198	X25IPI SCONFIG	X25IPI CONFIG	X25IPI
591	198	X25IPI SAMPGCS	X25IPI GCS	X25IPI
592	592	TCPIP SDATA	TCPIP DATA	All Services
592	592	ETCHOSTS SAMPLE	ETC HOSTS	All Services
592	592	ETC SAMPSERV	ETC SERVICES	All Services
592	198	HOSTS SLOCAL	HOSTS LOCAL	All Services
592	592	LCL2ETC SAMPEXEC	LCL2ETC EXEC	TCP/IP Admin.
592	592	RTD2MPR SAMPEXEC	RTD2MPR EXEC	TCP/IP Admin.
592	592	MIBX2DSC SAMPEXEC	MIBX2DSC EXEC	TCP/IP Admin.
592	592	IPFORMAT SCONFIG	IPFORMAT CONFIG	TCP/IP Admin.
592	592	PKTTRACE SAMPEXEC	PKTTRACE EXEC	TCP/IP Admin.

Notes:

1. Minidisks owned by the TCPMAINT user ID.
2. Location as ***optionally*** placed into production by the **TCP2PROD** command.
3. The ADMSERV 191 minidisk.

Figure 19 (Page 3 of 3). TCP/IP for z/VM Sample and Configuration Files

Sample Location (1*)	**Configured Location** (1*, 2*)	**Sample File Name / Type**	**Configured File Name / Type**	**Usage**
592	592	FTP SDATA	FTP DATA	FTP Client
592	592	KRB SCONFIG	KRB CONF	Kerberos Clients
592	592	GDXAPLCS SAMPMAP	GDXAPLCS MAP	GDDMXD/VM

Notes:

1. Minidisks owned by the TCPMAINT user ID.
2. Location as ***optionally*** placed into production by the **TCP2PROD** command.
3. The ADMSERV 191 minidisk.

Figure 20 lists TCP/IP for z/VM ***preconfigured*** files that are provided (or, created) to facilitate the use of certain z/VM services. The *sample* file locations shown are those established by the TCP/IP for z/VM **TCP2PROD** command, (with **tcprun** specified as the catalog section operand), where as *configured* locations are those established through use of the **TCP2PROD** command, with **tcppreconfig** specified as the catalog section operand. For reference, source and production file naming information provided as well.

Note: Unless noted otherwise, the minidisks listed in the *Sample Location* and *Configured Location* columns in Figure 20 are TCP/IP for z/VM *production* minidisks owned by the TCPMAINT user ID.

Figure 20. TCP/IP for z/VM Sample and Preconfigured Files

Sample Location (1*)	**Configured Location** (1*, 2*)	**Sample File Name / Type**	**Configured File Name / Type**	**Usage**
591	591	DTCVSW1 STCPIP	DTCVSW1 TCPIP (3*)	VSWITCH
591	591	DTCVSW2 STCPIP	DTCVSW2 TCPIP (3*)	VSWITCH

Notes:

1. Minidisks owned by the TCPMAINT user ID.
2. Location as placed into production by the **TCP2PROD** command.
3. This file contains pre-configured default values. If modifications are necessary, these should be implemented in a locally-created, 198-resident copy of this file.

Figure 21 on page 72 lists TCP/IP for z/VM ***BFS installation control*** files that are used to facilitate the installation of select TCP/IP program files into the z/VM Byte File System. The control file locations shown are those established by the TCP/IP for z/VM **TCP2PROD** command, (with **tcprun** specified as the catalog section operand), where as *configured* locations are those established through use of the **TCP2PROD** command, with **tcpbfs** specified as the catalog section operand.

Note: Unless noted otherwise, the minidisks listed in the *Control Location* and *Production Location* columns in Figure 21 are TCP/IP for z/VM *production* minidisks owned by the TCPMAINT user ID.

Figure 21. TCP/IP for z/VM Byte File System Resident Files

Control Location (1*)	**Configured Location** (1*, 2*)	**Control File Name / Type**	**Production File Name / Type**	**Usage**
591	BFS	LDAPSRV LOADBFS	- - (3*)	LDAPSRV

Notes:

1. Minidisks owned by the TCPMAINT user ID.
2. Location as placed into production by the **TCP2PROD** command.
3. For detailed information about relevant files and directories, review the listed LOADBFS file.

TCP/IP for z/VM is now installed and built on your system.

7.0 Service Instructions

Note — z/VM Automated Service Procedure

The z/VM automated service procedure (use of the z/VM **SERVICE** and **PUT2PROD** commands) is the ***preferred*** method for applying service to TCP/IP for z/VM.

If you have chosen to use the automated service procedure for applying (RSU) and CORrective service to your z/VM version 5 release 3 system, use the service instructions documented in *z/VM: Guide for Automated Installation and Service* for applying service to TCP/IP for z/VM, instead of those presented here.

This section describes the method by which TCP/IP for z/VM is serviced; it provides step-by-step procedures to install corrective (COR) and preventive service for TCP/IP for z/VM, using VMSES/E. Preventive service for TCP/IP for z/VM is delivered via a Recommended Service Upgrade (RSU).

To become more familiar with service using VMSES/E, read the introductory chapters in:

- *z/VM: VMSES/E Introduction and Reference* (GC24-6130)

This publication also contains command syntax for the VMSES/E commands cited throughout these instructions.

Each step of these service instructions must be followed. Do not skip any step unless directed otherwise.

Throughout these instructions, the use of IBM-supplied default minidisk device numbers and user IDs is assumed. If different user IDs, device numbers, or SFS directories are used to install TCP/IP for z/VM in your environment, adapt these instructions as needed.

Note!

Any sample console output presented throughout these instructions is based on a z/VM version 5 release 3 system; this output reflects an installation environment in which default values (PPF and component names, user IDs, and minidisks) are in use.

7.1 VMSES/E Service Process Overview

A brief description of the steps required to service TCP/IP for z/VM, using VMSES/E, follows:

- **Merge Existing Service** — The VMFMRDSK command is used to clear the alternate apply disk before receiving new service. This allows you to easily remove new service if a serious problem is encountered during its testing or use.

- **Receive New Service** — The VMFREC command is used to receive service from the delivery media and place it on the DELTA disk.
- **Apply the Service** — The VMFAPPLY command is used to update the version vector table (VVT), which identifies the service level of all serviced parts. In addition, AUX files are generated for parts that require them, based on the content of the VVT.
- **Reapply Local Service** (if applicable) — All local service (*user modifications*, or “mods”) must be entered into the software inventory to allow VMSES/E to track these changes and build them into the system. Refer to Chapter 7 of the *z/VM: Service Guide* for this procedure.
- **Build New Levels** — The build task generates the serviced level of an object and places the new object on a test BUILD disk.
- **Place the New Service into Production** — Once all service has been satisfactorily tested, it is placed into production by copying the new service to the production disks. Note that customized files that have been serviced may require additional customization.

7.2 Servicing TCP/IP for z/VM

7.2.1 Important Service Notes

Before you service TCP/IP for z/VM, you should review the information that follows and take appropriate action to ensure that your service environment is correct.

7.2.1.1 PPF Override Considerations

A *ppfname* of **5VMTCP30** is cited throughout these service instructions, which assumes the PPF supplied by IBM for TCP/IP for z/VM is in use. If you have created your own TCP/IP for z/VM PPF override file, use the *ppfname* of your override file (instead of 5VMTCP30) **throughout** this procedure, unless noted otherwise.

7.2.1.2 Language Environment Run-time Library Considerations

If you service TCP/IP for z/VM C components, the Language Environment for z/VM must be available when you build serviced objects using the VMFBLD command. If the Language Environment for z/VM does not reside on a system minidisk automatically accessed by VMSES/E (such as the MAINT 19E minidisk), you need to ensure the appropriate minidisk is available (perhaps through the use of a PPF override).

7.2.1.3 Byte File System (BFS) Considerations

To allow for the installation of service for TCP/IP LDAP components, the BFS file pools (**VMSYS** and **VMSYSU**, by default) must be available and in operation.

7.2.1.4 Installing RSU and COR Service — Where to Begin

Quick Index for Service Instructions

Select the service instructions you should use, based on the type of service you are installing:

- TCP/IP for z/VM RSU Service — Begin with 7.2.2, “Preventive (RSU) Service for TCP/IP for z/VM” on page 76
- TCP/IP for z/VM COR Service — Begin with 7.2.3, “Corrective (COR) Service for TCP/IP for z/VM” on page 86

7.2.2 Preventive (RSU) Service for TCP/IP for z/VM

Preventive service is available periodically on the Recommended Service Upgrade (RSU). Each RSU is cumulative and contains selected, important PTFs. The service on the RSU is in pre-applied, pre-built format and includes serviced files, the objects that were rebuilt using these files, and an updated software inventory.

The RSU content allows for installing new service more quickly than an equivalent group of Corrective (COR) PTFs. However, the installation of an RSU requires that you reapply any reach-ahead service that is already installed (service you have applied to your system that is not on the RSU).

RSU deliverables are in *install* format; thus, the VMFINS command is used to load an RSU.

7.2.2.1 Prepare to Receive Service

Electronic Service (Envelope File)

If you have received the RSU electronically or on CD-ROM, follow the appropriate instructions to retrieve and decompress the envelope file(s) to your A-disk. Decompression is currently accomplished by using the DETERSE module, which is provided as part of the VMSES/E component of z/VM.

The service (PTF) envelope files that result from the decompression process must have a file type of **SERVLINK**. Make a note of the file names that you use when envelope files are decompressed, because you will need to supply these in place of the *envfilename* in the VMFINS commands that follow.

Also, the documentation envelope file produced after having run DETERSE will be a readable plain-text file. This file will not be used as part of the RSU application instructions that follow.

1 Log on the TCP/IP for z/VM service user ID, **5VMTCP30**.

The PROFILE EXEC provided (as part of the z/VM version 5 release 3 System Deliverable) for this user ID contains ACCESS commands for VMSES/E minidisks that are necessary to run the commands cited in later steps. The minidisks required are the VMSES/E code minidisk (MAINT 5E5, by default) and the VMSES/E Software Inventory minidisk (MAINT 51D, by default).

2 Issue the CMS QUERY DISK command to verify the VMSES/E code and Software Inventory minidisks are correctly linked and accessed.

query disk

Verify the MAINT 5E5 minidisk is accessed as file mode **B**, and is linked **R/O**.

Verify the MAINT 51D minidisk is accessed as file mode **D**, and is linked **R/W**.

Note: If another user has the MAINT 51D minidisk linked in write (R/W) mode, you'll obtain only read (R/O) access to this minidisk. If this occurs, have that user re-link the 51D disk in read-only (RR) mode, after which you need issue the appropriate LINK and ACCESS commands for the 51D minidisk. Do not continue with these procedures until a R/W link is established to the 51D minidisk.

3 If necessary, establish the appropriate access to the VMSES/E minidisks.

a Establish read access to the VMSES/E code minidisk.

```
link maint 5e5 5e5 rr
access 5e5 b
```

b Establish write access to the Software Inventory minidisk.

```
link maint 51d 51d mr
access 51d d
```

4 Establish access to the RSU deliverable.

a If receiving the RSU from **tape**:

Mount the TCP/IP for z/VM RSU tape on an appropriate device and ensure this device has been attached to the **5VMTCP30** user ID using virtual device number 181.

b If receiving the RSU from an **envelope** file:

Ensure the envelope file resides on either the A-disk of the service user ID, or on a minidisk or SFS directory accessed at file mode C.

5 Receive the product documentation (5VMTCP30 MEMO) to the 51D minidisk, then identify the products and components for which service is included on the RSU. The MEMO documentation identifies the amount of storage necessary to receive the service present on the RSU. Use this information to ensure that your service disks or directories have adequate storage for this purpose.

a If receiving the RSU from **tape**, issue:

```
vmfins install info (nomemo
```

nomemo will load but not print the memo.

b If receiving the RSU from an **envelope** file, issue:

vmfins install info (nomemo env *envfilename*

nomemo will load but not print the memo.

6 Clear the alternate APPLY disk to ensure that a clean minidisk exists for receipt of the new service.

vmfmrdsk 5vmtcp30 {tcpip | tcpipsfs} apply (setup

Use **tcpip** if the TCP/IP for z/VM default minidisk environment has been maintained; use **tcpipsfs** if the service minidisks were moved to Shared File System directories.

This command copies the alternate APPLY disk to the production APPLY disk and then clears the alternate APPLY disk.

7 Review the merge message log ($VMFMRD $MSGLOG). If necessary, correct any problems before you proceed with the next step. For information about handling specific error messages, see the appropriate *z/VM: Messages and Codes* publication or use on-line HELP.

vmfview mrd

8 Invoke the VMFPSU command to obtain additional information about the service contained on the RSU and how it will affect your local modifications. This command creates an output file (*appid* **PSUPLAN**) that you should review. See the *z/VM: Service Guide* for an explanation of this file and its content.

vmfpsu 5vmtcp30 {tcpip | tcpipsfs}

Use **tcpip** if the TCP/IP for z/VM default minidisk environment has been maintained; use **tcpipsfs** if the service minidisks were moved to Shared File System directories.

This command produces an output file which compares service present on the RSU to the service on your system. The file name is *appid* **PSUPLAN**, where *appid* is as specified in the PPF file.

7.2.2.2 Receive the Service

1 Receive the service on the RSU.

Because the RSU contains pre-applied, pre-built service in *install* format, the VMFINS command is used to load:

- new service to the DELTA disk,
- an updated apply service inventory to the APPLY disk, and
- pre-built objects to the appropriate test build disks.

a If receiving the RSU from **tape**, issue:

```
vmfins install ppf 5vmtcp30 {tcpip | tcpipsfs} (nomemo nolink
```

Use **tcpip** if the TCP/IP for z/VM default minidisk environment has been maintained; use **tcpipsfs** if the service minidisks were moved to Shared File System directories.

nolink prevents VMFINS from linking required minidisks; it will only access these minidisks if they are not accessed.

b If receiving the RSU from an **envelope** file, issue:

```
vmfins install ppf 5vmtcp30 {tcpip | tcpipsfs} (nomemo nolink env envfilename override no
```

Use **tcpip** if the TCP/IP for z/VM default minidisk environment has been maintained; use **tcpipsfs** if the service minidisks were moved to Shared File System directories.

nolink prevents VMFINS from linking required minidisks; it will only access these minidisks if they are not accessed.

2 Review the install message log ($VMFINS $MSGLOG). If necessary, correct any problems before you proceed with the next step. For information about handling specific error messages, see the appropriate *z/VM: Messages and Codes* publication or use on-line HELP.

```
vmfview install
```

7.2.2.3 Apply the Service

Because service on the RSU is pre-applied, this step reapplies applicable reach-ahead service (service you have applied to your system that is not on the RSU).

1 Reapply reach-ahead service.

vmfapply ppf 5vmtcp30 {tcpip | tcpipsfs}

Use **tcpip** if the TCP/IP for z/VM default minidisk environment has been maintained; use **tcpipsfs** if the service minidisks were moved to Shared File System directories.

This command reapplies the reach-ahead service on your system. The version vector table (VVT) is updated with serviced part information and all necessary AUX files are generated on the alternate apply disk.

2 Review the apply message log ($VMFAPP $MSGLOG). If necessary, correct any problems before you proceed with the next step. For information about handling specific error messages, see the appropriate *z/VM: Messages and Codes* publication or use on-line HELP.

vmfview apply

3 Re-work and reapply local service, if it has been affected by RSU-provided service.

a. The output file created by the VMFPSU command (invoked in step 8 on page 78) identifies local modifications that are affected by RSU service.

b. For information on re-working local modifications, refer to Chapter 7 in the *z/VM: Service Guide* and follow the steps that are applicable to your local modification(s).

   Note that when using this information, the following substitutions may need to be made:

   - **zvm** should be: **5vmtcp30**
   - *compname* should be: **tcpip** or **tcpipsfs**
   - *appid* should be: **5vmtcp30**
   - *fm-local* should be the file mode of the 2C4 minidisk
   - *fm-applyalt* should be the file mode of the 2A6 minidisk

   Keep in mind that when you reach this step in the *z/VM: Service Guide*:

   - “Rebuilding Objects”

you should return to using this program directory and continue with step 7.2.2.4, “Update the Build Status Table” on page 81.

7.2.2.4 Update the Build Status Table

1 Update the Build Status Table for serviced parts.

vmfbld ppf 5vmtcp30 {tcpip | tcpipsfs} (status

Use **tcpip** if the TCP/IP for z/VM default minidisk environment has been maintained; use **tcpipsfs** if the service minidisks were moved to Shared File System directories.

This command updates the Build Status Table to determine what objects (if any) remain to be built.

Note - $PPF Service

If a $PPF file has been serviced you will receive this prompt:

```
VMFBLD2185R The following source product parameter files have been
            serviced:
VMFBLD2185R 5VMTCP30 $PPF
VMFBLD2185R When source product parameter files are serviced, all
            product parameter files built from them must be recompiled
            using VMFPPF before VMFBLD can be run.
VMFBLD2185R Enter zero (0) to have the serviced source product
            parameter files built to your A-disk and exit VMFBLD so
            you can recompile your product parameter files with VMFPPF.
VMFBLD2185R Enter one (1) to continue only if you have already
            recompiled your product parameter files with VMFPPF.
```

0

Enter a zero (**0**) and complete the steps given here before you continue.

```
VMFBLD2188I Building 5VMTCP30 $PPF
            on 191 (A) from level $PFnnnnn
```

Note - $PPF Service (Continued)

vmfppf 5vmtcp30 *	**Note:** If you've created your own PPF override, use your PPF name for this command instead of 5VMTCP30. **Note:** **Do not** use your own PPF name in place of 5VMTCP30 for the COPYFILE and ERASE commands that follow:

copyfile 5vmtcp30 $ppf a = = d (olddate replace

erase 5vmtcp30 $ppf a

vmfbld ppf 5vmtcp30 {tcpip | tcpipsfs} (status

1	Re-issue VMFBLD to complete updates to the build status table. If you've created your own PPF override, use your PPF name instead of 5VMTCP30. Use **tcpip** if the TCP/IP for z/VM default minidisk environment has been maintained; use **tcpipsfs** if the service minidisks were moved to Shared File System directories. When you receive the previously displayed prompt, enter a one (**1**) and continue to the next step.

2 Use VMFVIEW to review the build status messages, and see what objects need to be built.

vmfview build

7.2.2.5 Build Serviced Objects

Because service on the RSU is pre-built, this step builds only objects which have been affected by any reach-ahead and local service that has been reapplied.

1 Ensure Language Environment for z/VM support is available if you are applying service to functions that require this support.

2 Rebuild TCP/IP for z/VM serviced parts.

vmfbld ppf 5vmtcp30 {tcpip | tcpipsfs} (serviced

Use **tcpip** if the TCP/IP for z/VM default minidisk environment has been maintained; use **tcpipsfs** if the service minidisks were moved to Shared File System directories.

3 Review the build message log ($VMFBLD $MSGLOG). If necessary, correct any problems before you proceed with the next step. For information about handling specific error messages, see the appropriate *z/VM: Messages and Codes* publication or use on-line HELP.

vmfview build

7.2.2.6 Test the New Service

All new service should be thoroughly tested before it is placed into production. A suggested method for doing this is to temporarily link and then access the appropriate *test* build minidisks ahead of their *production* counterparts. Representative LINK and ACCESS statements for this purpose are illustrated here:

```
LINK TCPMAINT 198 198 RR
LINK 5VMTCP30 491 491 RR
LINK 5VMTCP30 492 492 RR
LINK TCPMAINT 591 591 RR
LINK TCPMAINT 592 592 RR
...
ACCESS 198 fm198                  /* fm198 is 'D' (perhaps) */
ACCESS 491 fm491                  /* fm491 is 'E' (perhaps) */
ACCESS 492 fm492                  /* fm492 is 'F' (perhaps) */
ACCESS 591 fm591                  /* fm591 is 'G' (perhaps) */
ACCESS 592 fm592                  /* fm592 is 'H' (perhaps) */
```

- To facilitate the testing of new service that affects TCP/IP **server virtual machines**, a TCP/IP *server profile exit* (or the supplied sample *global profile exit*, TCPRUNXT SEXEC) could be used to establish a suitable environment, as part of server initialization (**SETUP**) processing. For information on these exits, see the chapter titled “General TCP/IP Server Configuration” in *TCP/IP Planning and Customization*.

  When TCP/IP server and administrative functions are tested, ensure that the TCPMAINT 198, 5VMTCP30 491, and TCPMAINT 591 minidisks are accessed (in addition to the 5VMTCP30 492 and TCPMAINT 592 minidisks) by the TCP/IP service virtual machines and administrative user IDs involved in this activity.

- To facilitate the testing of new service that affects TCP/IP **client functions** (for example, the NETSTAT or FTP commands), only the 5VMTCP30 492 and

TCPMAINT 592 minidisks need to be accessed (with respect to TCP/IP-specific minidisk requirements).

When new service is tested, consult any applicable documentation (for example, that provided with the APARs which comprise the service) to account for changes specific to new or changed function. Your testing may also require TCP/IP services affected by service to be shutdown and restarted, possibly more than once.

Note: Service to certain TCP/IP components may be relevent to z/VM virtual switch (VSWITCH) support, and may at times necessitate a shutdown and restart of any VSWITCH controller servers that are used by your installation.

Note - TCP/IP and VSWITCH Controller Shutdown Considerations

Before you shutdown any TCP/IP or VSWITCH controller servers, ensure that any applicable conditions or guidelines for your installation have been followed.

For information on shutting down TCP/IP servers, see the section that discusses “Starting and Stopping TCP/IP Servers” in the chapter titled “General TCP/IP Server Configuration,” of *TCP/IP Planning and Customization*.

Note that the TCPMSMGR command can be used to manage the shutdown and initialization of the TCP/IP servers and VSWITCH controllers that are used by your installation. For more information about the TCPMSMGR command, see Appendix A, “TCP/IP Installation, Service and Migration Utilities” on page 104.

7.2.2.7 Place the Service into Production

Once the new service has been thoroughly tested, it needs to be placed into production; complete the appropriate TCP/IP service instructions that remain to accomplish this.

RSU Service Instructions — Where To Next...

See 7.2.4, “Place the New TCP/IP for z/VM Service Into Production” on page 94 for instructions for completing the installation of RSU service.

7.2.3 Corrective (COR) Service for TCP/IP for z/VM

Corrective service for TCP/IP for z/VM is provided in COR format via tape or electronic envelope. It is installed using the VMSES/E VMFREC, VMFAPPLY, and VMFBLD commands.

Electronic Service (Envelope File)

If you have received service electronically or on CD-ROM, follow the appropriate instructions to retrieve and decompress the envelope file(s) to your A-disk. Decompression is currently accomplished by using the DETERSE module, which is provided as part of the VMSES/E component of z/VM.

The documentation envelope and the service (PTF) envelope files that result from the decompression process must have a file type of **SERVLINK**. Make a note of the file names that you use when envelope files are decompressed, because you will need to supply these in place of the *envfilename* in the VMFREC commands that follow.

7.2.3.1 Prepare to Receive Service

1 Log on the TCP/IP for z/VM service user ID, **5VMTCP30**.

The PROFILE EXEC provided (as part of the z/VM version 5 release 3 System Deliverable) for this user ID contains ACCESS commands for VMSES/E minidisks that are necessary to run the commands cited in later steps. The minidisks required are the VMSES/E code minidisk (MAINT 5E5, by default) and the VMSES/E Software Inventory minidisk (MAINT 51D, by default).

2 Issue the CMS QUERY DISK command to verify the VMSES/E code and Software Inventory minidisks are correctly linked and accessed.

query disk

Verify the MAINT 5E5 minidisk is accessed as file mode **B**, and is linked **R/O**.

Verify the MAINT 51D minidisk is accessed as file mode **D**, and is linked **R/W**.

Note: If another user has the MAINT 51D minidisk linked in write (R/W) mode, you'll obtain only read (R/O) access to this minidisk. If this occurs, have that user re-link the 51D disk in read-only (RR) mode, after which you need issue the appropriate LINK and ACCESS commands for the 51D minidisk. Do not continue with these procedures until a R/W link is established to the 51D minidisk.

3 If necessary, establish the appropriate access to the VMSES/E minidisks.

a Establish read access to the VMSES/E code minidisk.

```
link maint 5e5 5e5 rr
access 5e5 b
```

b Establish write access to the Software Inventory minidisk.

```
link maint 51d 51d mr
access 51d d
```

4 Establish access to the COR service deliverable.

a If receiving service from **tape**:

Mount the TCP/IP for z/VM corrective (COR) service tape on an appropriate device and ensure this device has been attached to the **5VMTCP30** user ID using virtual device number 181.

b If receiving service from an **envelope** file:

Ensure the envelope file resides on either the A-disk of the service user ID, or on a minidisk or SFS directory accessed at file mode C.

5 Establish the correct minidisk access order.

vmfsetup 5vmtcp30 {tcpip | tcpipsfs}	Use **tcpip** if the TCP/IP for z/VM default minidisk environment has been maintained; use **tcpipsfs** if the service minidisks were moved to Shared File System directories.

6 Receive the service documentation. The VMFREC command (with its INFO option) loads the service documentation and displays a list of all products for which service is present on the media.

a If receiving service from **tape**, issue:

vmfrec info	This command loads the service memo to the 5VMTCP30 191 minidisk.

b If receiving service from an **envelope** file, issue:

vmfrec info (env *envfilename*

This command loads the service memo to the 5VMTCP30 191 minidisk.

7 Review the receive message log ($VMFREC $MSGLOG). If necessary, correct any problems before you proceed with the next step. For information about handling specific error messages, see the appropriate *z/VM: Messages and Codes* publication or use on-line HELP.

vmfview receive

Also, note the products and components for which service has been supplied. To do this, use the PF5 key to show all “status” messages, which will identify these products and components.

8 Clear the alternate APPLY disk to ensure that a clean minidisk exists for receipt of the new service. for new service.

vmfmrdsk 5vmtcp30 {tcpip | tcpipsfs} apply

Use **tcpip** if the TCP/IP for z/VM default minidisk environment has been maintained; use **tcpipsfs** if the service minidisks were moved to Shared File System directories.

This command copies the alternate APPLY disk to the production APPLY disk and then clears the alternate APPLY disk.

9 Review the merge message log ($VMFMRD $MSGLOG). If necessary, correct any problems before you proceed with the next step. For information about handling specific error messages, see the appropriate *z/VM: Messages and Codes* publication or use on-line HELP.

vmfview mrd

7.2.3.2 Receive the Service

1 Receive the service.

a If receiving service from **tape**, issue:

```
vmfrec ppf 5vmtcp30 {tcpip | tcpipsfs}
```

Use **tcpip** if the TCP/IP for z/VM default minidisk environment has been maintained; use **tcpipsfs** if the service minidisks were moved to Shared File System directories.

This command receives service from the service tape. All new service is loaded to the DELTA disk.

b If receiving service from an **envelope** file, issue:

```
vmfrec ppf 5vmtcp30 {tcpip | tcpipsfs} (env envfilename
```

Use **tcpip** if the TCP/IP for z/VM default minidisk environment has been maintained; use **tcpipsfs** if the service minidisks were moved to Shared File System directories.

This command receives service from the service envelope. All new service is loaded to the DELTA disk.

2 Review the receive message log ($VMFREC $MSGLOG). If necessary, correct any problems before you proceed with the next step. For information about handling specific error messages, see the appropriate *z/VM: Messages and Codes* publication or use on-line HELP.

```
vmfview receive
```

7.2.3.3 Apply the Service

1 Apply the new service.

vmfapply ppf 5vmtcp30 {tcpip | tcpipsfs}

Use **tcpip** if the TCP/IP for z/VM default minidisk environment has been maintained; use **tcpipsfs** if the service minidisks were moved to Shared File System directories.

This command applies the just-received service. The version vector table (VVT) is updated with serviced part information and all necessary AUX files are generated on the alternate apply disk.

2 Review the apply message log ($VMFAPP $MSGLOG). If necessary, correct any problems before you proceed with the next step. For information about handling specific error messages, see the appropriate *z/VM: Messages and Codes* publication or use on-line HELP.

vmfview apply

Note - Local Modifications

If you receive message VMFAPP2120W, you need to reapply any local modifications before building an updated level of TCP/IP for z/VM. For information on re-working local modifications, refer to Chapter 7 in the *z/VM: Service Guide* and follow the steps that are applicable to your local modification(s).

Note that when using this information, the following substitutions may need to be made:

- **zvm** should be: **5vmtcp30**
- *compname* should be: **tcpip** or **tcpipsfs**
- *appid* should be: **5vmtcp30**
- *fm-local* should be the file mode of the 2C4 minidisk
- *fm-applyalt* should be the file mode of the 2A6 minidisk

Keep in mind that when you reach this step in the *z/VM: Service Guide*:

- “Rebuilding Objects”

you should return to using this program directory and continue with step 7.2.3.4, “Update the Build Status Table” on page 91.

7.2.3.4 Update the Build Status Table

1 Update the Build Status Table for serviced parts.

vmfbld ppf 5vmtcp30 {tcpip | tcpipsfs} (status

Use **tcpip** if the TCP/IP for z/VM default minidisk environment has been maintained; use **tcpipsfs** if the service minidisks were moved to Shared File System directories.

This command updates the Build Status Table to identify objects that need to be built as a result of applying new service.

Note - $PPF Service

If a $PPF file has been serviced you will receive this prompt:

```
VMFBLD2185R The following source product parameter files have been
            serviced:
VMFBLD2185R 5VMTCP30 $PPF
VMFBLD2185R When source product parameter files are serviced, all
            product parameter files built from them must be recompiled
            using VMFPPF before VMFBLD can be run.
VMFBLD2185R Enter zero (0) to have the serviced source product
            parameter files built to your A-disk and exit VMFBLD so
            you can recompile your product parameter files with VMFPPF.
VMFBLD2185R Enter one (1) to continue only if you have already
            recompiled your product parameter files with VMFPPF.
```

0

Enter a zero (**0**) and complete the steps given here before you continue.

```
VMFBLD2188I Building 5VMTCP30 $PPF
            on 191 (A) from level $PFnnnnn
```

Note - $PPF Service (Continued)

vmfppf 5vmtcp30 *

Note: If you've created your own PPF override, use your PPF name (for this command *only*) instead of 5VMTCP30.

Note: **Do not** use your own PPF name in place of 5VMTCP30 for these COPYFILE and ERASE commands:

copyfile 5vmtcp30 $ppf a = = d (olddate replace

erase 5vmtcp30 $ppf a

vmfbld ppf 5vmtcp30 {tcpip | tcpipsfs} (status setup

1

Re-issue VMFBLD to complete updates to the build status table.

If you've created your own PPF override, use your PPF name instead of 5VMTCP30.

Use **tcpip** if the TCP/IP for z/VM default minidisk environment has been maintained; use **tcpipsfs** if the service minidisks were moved to Shared File System directories.

When you receive the previously displayed prompt, enter a one (**1**) and continue to the next step.

2 Use VMFVIEW to review the build status messages, and see what objects need to be built.

vmfview build

7.2.3.5 Build Serviced Objects

1 Ensure Language Environment for z/VM support is available if you are applying service to functions that require this support.

2 Rebuild TCP/IP for z/VM serviced parts.

vmfbld ppf 5vmtcp30 {tcpip | tcpipsfs} (serviced

Use **tcpip** if the TCP/IP for z/VM default minidisk environment has been maintained; use **tcpipsfs** if the service minidisks were moved to Shared File System directories.

3 Review the build message log ($VMFBLD $MSGLOG). If necessary, correct any problems before you proceed with the next step. For information about handling specific error messages, see the appropriate *z/VM: Messages and Codes* publication or use on-line HELP.

vmfview build

7.2.3.6 Test the New Service

All new service should be thoroughly tested before it is placed into production. A suggested method for doing this is to temporarily link and then access the appropriate *test* build minidisks ahead of their *production* counterparts. Representative LINK and ACCESS statements for this purpose are illustrated here:

```
LINK TCPMAINT 198 198 RR
LINK 5VMTCP30 491 491 RR
LINK 5VMTCP30 492 492 RR
LINK TCPMAINT 591 591 RR
LINK TCPMAINT 592 592 RR
...
ACCESS 198 fm198                /* fm198 is 'D' (perhaps) */
ACCESS 491 fm491                /* fm491 is 'E' (perhaps) */
ACCESS 492 fm492                /* fm492 is 'F' (perhaps) */
ACCESS 591 fm591                /* fm591 is 'G' (perhaps) */
ACCESS 592 fm592                /* fm592 is 'H' (perhaps) */
```

- To facilitate the testing of new service that affects TCP/IP **server virtual machines**, a TCP/IP *server profile exit* (or the supplied sample *global profile exit*, TCPRUNXT SEXEC) could be used to establish a suitable environment, as part of server initialization (**SETUP**) processing. For information on these exits, see the chapter titled “General TCP/IP Server Configuration” in *TCP/IP Planning and Customization*.

  When TCP/IP server and administrative functions are tested, ensure that the TCPMAINT 198, 5VMTCP30 491, and TCPMAINT 591 minidisks are accessed (in addition to the 5VMTCP30 492 and TCPMAINT 592 minidisks) by the TCP/IP service virtual machines and administrative user IDs involved in this activity.

- To facilitate the testing of new service that affects TCP/IP **client functions** (for example, the NETSTAT or FTP commands), only the 5VMTCP30 492 and TCPMAINT 592 minidisks need to be accessed (with respect to TCP/IP-specific minidisk requirements).

When new service is tested, consult any applicable documentation (for example, that provided with the APARs which comprise the service) to account for changes specific to new or changed function. Your testing may also require TCP/IP services affected by service to be shutdown and restarted, possibly more than once.

Note: Service to certain TCP/IP components may be relevent to z/VM virtual switch (VSWITCH) support, and may at times necessitate a shutdown and restart of any VSWITCH controller servers that are used by your installation.

Note - TCP/IP and VSWITCH Controller Shutdown Considerations

Before you shutdown any TCP/IP or VSWITCH controller servers, ensure that any applicable conditions or guidelines for your installation have been followed.

For information on shutting down TCP/IP servers, see the section that discusses "Starting and Stopping TCP/IP Servers" in the chapter titled "General TCP/IP Server Configuration," of *TCP/IP Planning and Customization*.

Note that the TCPMSMGR command can be used to manage the shutdown and initialization of the TCP/IP servers and VSWITCH controllers that are used by your installation. For more information about the TCPMSMGR command, see Appendix A, "TCP/IP Installation, Service and Migration Utilities" on page 104.

7.2.4 Place the New TCP/IP for z/VM Service Into Production

7.2.4.1 5VMTCP30 CATALOG File Modification Notes

Before you continue with the steps in this section, it is advised that you verify the correctness of any 5VMTCP30 CATALOG file modifications that may have been made for your installation. The 5VMTCP30 CATALOG file is used by the TCP2PROD command to copy files to TCP/IP for z/VM minidisks. See Appendix A, "TCP/IP Installation, Service and Migration Utilities" on page 104 for detailed information about the TCP2PROD command and TCP/IP for z/VM catalog files.

Note: If any new sample and configuration files are supplied with TCP/IP for z/VM service, the 5VMTCP30 CATALOG file will be updated to reflect this.

The various definition sections of the 5VMTCP30 CATALOG file, and the files associated with each section, are briefly described here:

Section **Description**

TCPBFS Used to place various TCP/IP for z/VM files into the z/VM Byte File System (BFS). The files listed in this section are those used by the z/VM LOADBFS command to identify specific files for installation into the BFS, their designated location, and how each should be processed.

TCPCONFIG **Optionally** used to create suitably named configuration files for customizing TCP/IP and TCP/IP services for your installation. For reference, the files processed using the this section are listed in Figure 19 on page 69.

TCPHELP Used to process TCP/IP CMS Help files. The files processed using this section are listed in the TCP/IP for z/VM CMS Help file VMSES/E build list (TCPBLHLP EXEC).

TCPPRECONFIG Used to create suitably named configuration files that incorporate *preconfigured* content for using certain z/VM or TCP/IP services (for example, for running z/VM system-default VSWITCH controllers). For reference, the files processed using the this section are listed in Figure 20 on page 71.

TCPRUN Used to process non-customizable TCP/IP for z/VM run-time files. For reference, the files processed using this section are listed in Figure 18 on page 67.

TCPSAMPLE Used to process customizable TCP/IP sample files. For reference, the files processed using this section are listed in Figure 19 on page 69.

TCPSVMCMS Used to process non-customizable TCP/IP for z/VM run-time files that must reside on individual **CMS-based** server virtual machine (SVM) minidisks. For reference, the files processed using the TCPSVMCMS section are listed in Figure 16 on page 65.

TCPSVMGCS Used to process non-customizable TCP/IP for z/VM run-time files that must reside on individual **GCS-based** server virtual machine (SVM) minidisks. For reference, the files processed using the TCPSVMGCS section are listed in Figure 17 on page 66.

7.2.4.2 Copy Serviced TCP/IP for z/VM Run-time Files Into Production

Once the new service has been thoroughly tested, it needs to be placed into production (that is, copied to TCP/IP for z/VM production build minidisks).

Note: Service to certain TCP/IP components may be relevent to z/VM virtual switch (VSWITCH) support, and may at times necessitate a shutdown and restart of any VSWITCH controller servers that are used by your installation.

1 Shutdown TCP/IP services.

Note - TCP/IP and VSWITCH Controller Shutdown Considerations

Before you shutdown any TCP/IP or VSWITCH controller servers, ensure that any applicable conditions or guidelines for your installation have been followed.

For information on shutting down TCP/IP servers, see the section that discusses “Starting and Stopping TCP/IP Servers” in the chapter titled

"General TCP/IP Server Configuration," of *TCP/IP Planning and Customization*.

Note that the TCPMSMGR command can be used to manage the shutdown and initialization of the TCP/IP servers and VSWITCH controllers that are used by your installation. For more information about the TCPMSMGR command, see Appendix A, "TCP/IP Installation, Service and Migration Utilities" on page 104.

2 Log on the TCP/IP for z/VM service user ID, **5VMTCP30**.

The PROFILE EXEC provided (as part of the z/VM version 5 release 3 System Deliverable) for this user ID contains ACCESS commands for VMSES/E minidisks that are necessary to run the commands cited in later steps. The minidisks required are the VMSES/E code minidisk (MAINT 5E5, by default) and the VMSES/E Software Inventory minidisk (MAINT 51D, by default).

3 Issue the CMS QUERY DISK command to verify the VMSES/E code and Software Inventory minidisks are correctly linked and accessed.

query disk

Verify the MAINT 5E5 minidisk is accessed as file mode **B**, and is linked **R/O**.

Verify the MAINT 51D minidisk is accessed as file mode **D**, and is linked **R/W**.

Note: If another user has the MAINT 51D minidisk linked in write (R/W) mode, you'll obtain only read (R/O) access to this minidisk. If this occurs, have that user re-link the 51D disk in read-only (RR) mode, after which you need issue the appropriate LINK and ACCESS commands for the 51D minidisk. Do not continue with these procedures until a R/W link is established to the 51D minidisk.

4 If necessary, establish the appropriate access to the VMSES/E minidisks.

a Establish read access to the VMSES/E code minidisk.

```
link maint 5e5 5e5 rr
access 5e5 b
```

b Establish write access to the Software Inventory minidisk.

```
link maint 51d 51d mr
access 51d d
```

5 Access the 5VMTCP30 491 and 492 minidisks.

```
access 491 i
access 492 j
```

The 491 minidisk is where the TCP2PROD EXEC and 5VMTCP30 CATALOG files reside. The DTCUME message repository (required for running TCP2PROD) resides on the 492 minidisk.

6 Review the 5VMTCP30 CATALOG file to verify its correctness, as suggested in 7.2.4.1, “5VMTCP30 CATALOG File Modification Notes” on page 94. Ensure that any changes you may have made to this file remain in effect, and that any local service used to customize this file has been properly re-worked and applied.

7 (*Optional*) Establish write links to any TCP/IP for z/VM production or server minidisks which are not yet linked in this mode.

LINK statements for the various TCP/IP for z/VM minidisks are present in the 5VMTCP30 directory entry (supplied as part of the installed z/VM version 5 release 3 System Deliverable).

If you have changed the default installation user ID or use different minidisk device numbers in your environment, you may need to manually link the necessary TCP/IP production and server minidisks. See Figure 16 on page 65, Figure 18 on page 67, and Figure 19 on page 69 for device link information. If you created a PPF override that has changed any of these device numbers, use your values.

link *tcpipid vdev1 vdev2* **mr**

Note: If another user has the *vdev1* minidisk linked in write (R/W) mode, you'll obtain only read (R/O) access to this minidisk. If this occurs, have that user re-link the *vdev1* disk in read-only (RR) mode, after which you need to re-issue the above LINK command. Do not continue with these procedures until a R/W link is established to the *vdev1* minidisk.

8 Copy serviced TCP/IP for z/VM files into production using the TCP2PROD command.

The command cited below processes files that are identified in the TCPRUN section of the 5VMTCP30 CATALOG file. See Appendix A, “TCP/IP Installation, Service and Migration Utilities” on page 104 for information about this command and TCP/IP for z/VM catalog files.

Verifying Your Environment

When you perform this step, it is suggested that you first invoke TCP2PROD as illustrated, but with the **TEST** option also specified. This will verify that all resources can be accessed and that the appropriate files will be processed.

With the **TEST** option in effect, **no files are copied into production**.

Resolve any reported problems, then invoke TCP2PROD (without the TEST option) as illustrated.

tcp2prod 5vmtcp30 {tcpip | tcpipsfs} 5vmtcp30 tcprun (setup

Use **tcpip** if the TCP/IP for z/VM default minidisk environment has been maintained; use **tcpipsfs** if the service minidisks were moved to Shared File System directories.

9 Review the TCP2PROD message log (TCP2PROD $MSGLOG). If necessary, correct any problems before you proceed with the next command.

vmfview tcp2prod

If Message DTCPRD3061W is Reported...

If message **DTCPRD3061W** is reported for one or more of the files processed in step 8, you will need to take further action to ensure that the subject file is properly placed into production. For more information, see Appendix I, “Managing TCP/IP Files with Unique Service Requirements” on page 155.

10 Copy selected TCP/IP for z/VM files into the z/VM Byte File System (BFS).

The command cited below processes files that are identified in the TCPBFS section of the 5VMTCP30 CATALOG file.

```
tcp2prod 5vmtcp30 {tcpip | tcpipsfs} 5vmtcp30 tcpbfs (setup
```

Use **tcpip** if the TCP/IP for z/VM default minidisk environment has been maintained; use **tcpipsfs** if the service minidisks were moved to Shared File System directories.

11 Review the TCP2PROD message log (TCP2PROD $MSGLOG). If necessary, correct any problems before you proceed with the next step.

```
vmfview tcp2prod
```

7.2.4.3 Copy Serviced TCP/IP for z/VM Sample Files Into Production

Use the TCP2PROD command, as described in this section, to place any new or updated IBM-supplied **sample** configuration files into production.

> **Note**
>
> When you perform this step, *only* new or updated sample files are copied into production. Files that have been customized for your installation are *not affected or replaced*.

Note: For step 2 below, it is assumed that the TCPSAMPLE section of the 5VMTCP30 CATALOG has been verified, as suggested in 7.2.4.1, “5VMTCP30 CATALOG File Modification Notes” on page 94. If this is not the case, you should make any necessary changes to this section of the 5VMTCP30 CATALOG file before you continue with the steps that follow.

1 If necessary, establish the appropriate environment, as described by steps 1 through 5, in 7.2.4.2, “Copy Serviced TCP/IP for z/VM Run-time Files Into Production” on page 95.

2 Copy serviced TCP/IP for z/VM configuration files into production using the TCP2PROD command. For reference, files that can be processed using the TCPSAMPLE section are listed in Figure 19 on page 69.

Verifying Your Environment

When you perform this step, it is suggested that you first invoke TCP2PROD as illustrated, but with the **TEST** option also specified. This will verify that all resources can be accessed and that the appropriate files will be processed.

With the **TEST** option in effect, **no files are copied into production**.

Resolve any reported problems, then invoke TCP2PROD (without the TEST option) as illustrated.

tcp2prod 5vmtcp30 {tcpip | tcpipsfs} 5vmtcp30 tcpsample (setup

Use **tcpip** if the TCP/IP for z/VM default minidisk environment has been maintained; use **tcpipsfs** if the service minidisks were moved to Shared File System directories.

3 Review the TCP2PROD message log (TCP2PROD $MSGLOG). If necessary, correct any problems before you proceed with the next step.

vmfview tcp2prod

If Message DTCPRD3043W is Reported...

If message **DTCPRD3043W** is reported for any of the sample files processed in step 2, you should review the updated content of these files and determine whether changes are required to any customized, production-use counterparts that are used for your installation.

When necessary, an updated sample file can be compared with its base-level counterpart (on the **5VMTCP30 2B2** Base Code minidisk) to identify specific changes that may not be apparent in a customized, production-use file.

7.2.4.4 Update your TCP/IP for z/VM Configuration

If any TCP/IP for z/VM *sample* configuration files have been updated through service (as reported by message **DTCPRD3043W** in step 2 on page 99), you should review the updated content of these files and determine whether changes are required to any customized, production-use counterparts that are used for your installation.

See *TCP/IP Planning and Customization* (SC24-6125) for detailed information about the content and use of these files, and how to configure specific TCP/IP servers for your environment. To accommodate service-related changes, you may at times need to consult APAR-specific documentation.

7.2.4.5 Re-Initialize TCP/IP Services

Once you have completed any necessary configuration changes, the appropriate TCP/IP servers must be initialized. For more information, see the section that discusses “Starting and Stopping TCP/IP Servers” in the chapter titled “General TCP/IP Server Configuration,” of *TCP/IP Planning and Customization*.

If the TCPMSMGR command was previously used to manage the shutdown of the TCP/IP servers used by your installation, it now can be used to initialize those servers. For more information about the TCPMSMGR command, see Appendix A, “TCP/IP Installation, Service and Migration Utilities” on page 104.

7.2.4.6 Copy Serviced TCP/IP Client Code to the z/VM Product Code Disk (Optional)

If you previously copied TCP/IP for z/VM client code to the z/VM product code disk, you should replace the appropriate files with their serviced counterparts. See Appendix H, “Copying TCP/IP for z/VM Client Code to the Y-Disk” on page 152 for additional information and instructions concerning this process.

7.2.4.7 Verify the RSU Service Level (Optional)

The procedures in this section describe how to determine the current TCP/IP for z/VM RSU service level by using either of these commands:

- TCP/IP **NETSTAT LEVEL**
- VMSES/E **VMFSIM QUERY**
- VMSES/E **SERVICE**

The service contained on each RSU constitutes a new service level; this service level is updated in the system inventory when RSU service is installed.

Note: Corrective (COR) service does not affect the TCP/IP for z/VM RSU service level. However, you should record and use this service level when corrective (COR) service is ordered.

7.2.4.7.1 RSU Level Information - Using the TCP/IP NETSTAT LEVEL Command

To use the TCP/IP **NETSTAT LEVEL** command to obtain the TCP/IP for z/VM RSU service level, the TCPIP server (or, the TCP/IP *stack*) must be operating and the NETSTAT command must be available.

access 592 *fm* where *fm* is an available file mode.
netstat level

```
VM TCP/IP Netstat Level 530
IBM 2094; z/VM Version 5 Release 3.0, service level 0701 (64-BIT), VM TCP/IP Level
530; RSU 0701 running TCPIP MODULE E2 dated 06/12/07 at 12:34
TCP/IP Module Load Address: xxxxxxxx
```

Both the z/VM and TCP/IP RSU service levels are reported (in order) in the returned output. For this example, the RSU level for each is: **0701**

Note that any user ID that has the TCPMAINT 592 minidisk accessed can use the NETSTAT LEVEL command.

7.2.4.7.2 RSU Level Information - Using the VMFSIM QUERY Command

To use the **VMFSIM QUERY** command, the VMSES/E code (MAINT 5E5) and Software Inventory (MAINT 51D) minidisks must be accessed. Thus, this command is most readily issued while logged on a service maintenance user ID such as **MAINT** or the TCP/IP installation user ID (**5VMTCP30**).

vmfsim query vm sysrecs tdata :ppf 5vmtcp30 :stat

```
VMFSIP2408I RESULTS FOR
            TDATA :PPF 5VMTCP30 :STAT
:PPF 5VMTCP30 TCPIP
  :STAT RECEIVED.06/12/07.11:23:45.5VMTCP30.RSU-0701
```

The last part of the status line indicates the TCP/IP RSU service level; for this example, this is: **0701**

7.2.4.7.3 RSU Level Information - Using the VMSES/E SERVICE Command

If you are using the automated service procedure to service TCP/IP for z/VM, then you can use the **SERVICE** command (with the **status** operand specified) to determine the current RSU level for TCP/IP

To use the **SERVICE** command, the VMSES/E code (MAINT 5E5) and Software Inventory (MAINT 51D) minidisks must be accessed. Thus, this command is most readily issued while logged on a service maintenance user ID such as **MAINT** or the TCP/IP installation user ID (**5VMTCP30**).

service tcpip status

```
VMFSRV2760I SERVICE processing started
VMFSRV1225I TCPIP (5VMTCP30%TCPIP) status:
VMFSRV1225I    Service Level     RSU-0701
VMFSRV1225I    Production Level  000-0000
VMFSRV2760I SERVICE processing completed successfully
```

You have finished servicing TCP/IP for z/VM.

Appendix A. TCP/IP Installation, Service and Migration Utilities

A.1 TCP2PROD Command

A.1.1 Purpose

Use the TCP2PROD command to copy one or more groups of TCP/IP for z/VM files into production. TCP2PROD uses the VMSES/E VMFCOPY command to copy designated files from one resource (a minidisk or SFS directory) to another. A TCP/IP for z/VM ***catalog*** file, identified by a command operand, is used to identify which product files are to be placed into production, as well as the minidisks and SFS directories that are to be used for this process. See A.2, “TCP/IP for z/VM CATALOG Files” on page 108 for more information about the catalog file and its structure and content.

Note: The TCP2PROD command is intended for use by the 5VMTCP30 user ID (or an equivalent maintenance user ID), and should be used only when TCP/IP for z/VM is installed or when TCP/IP service has been applied to your system.

```
►►──TCP2PROD──ppfname──compname──ctlg_name──ctlg_section──────────►

►──┬──────────────────────┬──────────────────────────────────────►◄
   └─(──┤ Options ├──┬───┬┘
                     └─)─┘
```

Options:

```
├──┬───────┬──┬───────────┬──┬───────┬──┬──────┬─────────────────────┤
   ├─DEBug─┤  ├─FORCECopy─┤  └─SETUP─┘  └─TEST─┘
   └─TRAce─┘  └─REPLace───┘
```

A.1.2 Operands

ppfname The name of the usable form product parameter file that is used for the installation and maintenance of TCP/IP for z/VM. The file type must be **PPF**.

compname The name of a component, as specified for a `:COMPNAME.` tag in the TCP/IP for z/VM product parameter file; *compname* is a 1- to 16-character alphanumeric identifier.

The PPF Variable Declarations (:DCL.) section defined for *compname* determines the *source* minidisks and SFS directories from which product files are copied; likewise for the *target* production minidisks to which these files are copied.

ctlg_name The name of the product catalog file to be processed. The file type must be **CATALOG**.

ctlg_section The definition section of the catalog file to be processed. The value specified as *ctlg_section* is used as a *root* for the “begin” and “end” tags that define each section of grouped entries (records) within a catalog file.

A.1.3 Options

DEBUG
TRACE

Causes supplementary messages to be issued to provide information for diagnostic purposes. The DEBUG and TRACE options are synonymous.

FORCECopy
REPLace

Causes files identified within a *configuration* definition section to be copied to their configured name and type, regardless of whether such a configuration file already exists. By default, a file identified within such a definition section is copied to its configured name and type *only* if the production instance of that file does not already exist. See A.2.7, “Conventional Catalog Definitions” on page 112 for details about how entries within a *configuration* section are processed. The FORCECOPY and REPLACE options are synonymous.

SETUP

Causes a VMSES/E **VMFSETUP (LINK** command to be issued as part of TCP2PROD processing, to establish a correct operational environment. The *ppfname* and *compname* operands supplied for the TCP2PROD command are also used as operands for the VMFSETUP command.

TEST

Causes processing for the current invocation to be performed such that no files are placed into production. The TEST option allows you to verify that required minidisks and SFS directories can be accessed without error, and that the appropriate catalog file entries will be processed. Additional messages are issued to clarify what actions *would* occur if this option were not specified.

A.1.4 Usage Notes

1. For the most part, TCP2PROD **does not issue minidisk LINK commands** as part of its processing (an exception to this is when the **TCPHELP** catalog section is processed).

   To ensure that required minidisks are accessible, appropriate LINK statements should be added to the 5VMTCP30 user ID entry in the system (CP) directory. Alternatively, the necessary LINK commands can be added to the PROFILE EXEC of the 5VMTCP30 user ID.

2. TCP2PROD uses the first CATALOG file found in the CMS search order that matches that specified by the *ctlg_name* operand.
3. Catalog file entries that are found to be unusable are bypassed, with appropriate warning or error messages issued.
4. TCP2PROD requires a minidisk or SFS directory to be accessed at file mode A with read/write (R/W) status, for use as temporary work space and for message logging.

A.1.5 TCP2PROD File Exclusion Support

To ensure that TCP/IP for z/VM production minidisks contain only those files necessary to provide or use TCP/IP services, the TCP2PROD command incorporates support that allows various files present on a *source* minidisk or directory resource to be excluded as certain “wildcard” catalog entries are processed. When this “file exclusion” support is applied, files can be excluded based on specific file names, file types, or by using conventional CMS file pattern matching techniques that employ the wildcard (*) and pattern matching (%) characters.

Files that are to be excluded in this manner must be identified within one or more TCP/IP for z/VM catalog file *exclude* sections that are separately defined for this purpose. Within such a section, one or more *exclusion* entries are defined that identify the specific files or file groups that are to be excluded. For more information about these entries, see A.2.8, “File Exclusion Definitions” on page 113.

TCP2PROD file exclusion processing is activated by the **XCLUDE** entry processing option, which is specified as part of a TCP/IP for z/VM wildcard file entry. See A.2.5, “Entry Processing Options” on page 111 for details about the XCLUDE option.

Notes:

1. When exclusion processing is activated for a given entry (and thus, a specific source and target resource pairing), TCP2PROD automatically excludes *unchanged* files from a copy operation in addition to those files identified within an exclude section. This avoids unnecessarily processing source files that have not been modified since such files were last placed into production.

   In this context, a file is considered to be unchanged when file attributes — other than file mode number and CMS data block count — for a source file and its production counterpart are identical.
2. Automatic exclusion of unchanged files is performed only when a valid XCLUDE option has been specified for a wildcard entry.

A.1.6 The TCP2PROD $MSGLOG File

Pertinent informational, warning and error messages that are issued to the console by TCP2PROD are also recorded in a message log file, TCP2PROD $MSGLOG. This log file is written to the minidisk or directory accessed at file mode A, and can be viewed using the VMSES/E **VMFVIEW** command.

The TCP2PROD $MSGLOG is cumulative, with the most recent entries appended at the **top** of the file. Separator headers that include date and time stamps are inserted in the log with each TCP2PROD invocation so newer log entries can be distinguished from older ones.

Notes:

1. Messages are not logged until TCP2PROD has completed an initial validation of supplied operands.
2. Diagnostic and other incidental messages are not recorded in the TCP2PROD $MSGLOG file.

A.1.7 Return Codes

Return Code	Description
0	Successful execution; no processing errors were encountered.
1	Incorrect invocation. TCP2PROD was invoked with an incorrect number of operands. A message that identifies the missing operand is displayed, in addition to the command syntax.
2	Internal error. If this return code is produced, processing status is indeterminate. Contact the TCP/IP for z/VM support group for problem determination and assistance in addressing this type of error.
4	Errors encountered, with warnings issued. The errors encountered may have caused processing to complete with only partial success. Review the TCP2PROD $MSGLOG for warning messages that identify any problems that were encountered.
8	Errors encountered; processing has not completed successfully. Review the TCP2PROD $MSGLOG for messages regarding the problems encountered.

A.2 TCP/IP for z/VM CATALOG Files

A.2.1 Purpose

The TCP/IP for z/VM **catalog** file is used by the TCP2PROD command to identify which product files are to be placed into production, as well as the minidisks and SFS directories that are to be used for this process. Structure, content and customization requirements and considerations associated with the TCP/IP for z/VM catalog file are described in the sections that follow.

A.2.2 Catalog File Structure

Within the TCP/IP for z/VM catalog file, distinct *sections* are defined which identify groups of related files that are to be copied from a given (*source*) minidisk or SFS directory resource to one or more (*target*) production resources. For example, product *run-time* files (such as the TCPIP and FTP MODULE files) are processed as one group, whereas TCP/IP *sample* configuration files (such as the TCPIP SDATA file) are processed as another, separate group.

Similarly named, paired *begin* and *end* tags are used to define a given section of grouped catalog *entries*. For example, in the supplied 5VMTCP30 CATALOG file, the section defined for sample configuration files is delimited by the `:TCPCSAMPLE.` and `:ETCPSAMPLE.` tags.

In general, the various entries (or, records) in the catalog file provide information sufficient for TCP2PROD to process and copy the files for a given group. See A.2.3, “Catalog Entry Syntax” on page 109 and A.2.6, “Catalog Entry Types” on page 112 for more specific information.

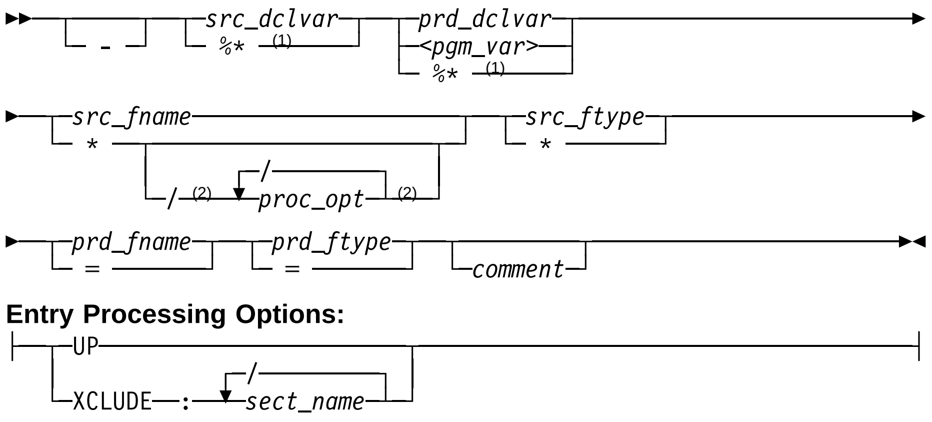
A unique, *exclusion* entry can be also be defined (within a separate catalog section) that identifies certain files or groups of files to be excluded as TCP2PROD processes the *conventional* entries just described. For more information about these entries and their use, see A.2.8, “File Exclusion Definitions” on page 113 and A.1.5, “TCP2PROD File Exclusion Support” on page 106.

Notes:

1. The file type of the catalog files used by the TCP2PROD command must be **CATALOG**.
2. Section definition tags must begin with a colon (:), end with a period (.), and must be comprised of a non-blank string (intervening blanks are not permitted). Case is not significant.
3. Section tags must be present on unique lines within a catalog file — they cannot be combined with file data entries. Tags must also be properly paired (that is, no attempt is made to detect a missing *end* tag for a given *begin* tag).

4. File exclusion entries must be defined separately from conventional catalog file entries, in sections defined specifically for this purpose.

A.2.3 Catalog Entry Syntax



Notes:

[1] Valid only for entries and section definitions used for file exclusion purposes.

[2] A period (.) is also accepted, for compatibility purposes.

A.2.4 Operands

- The entry bypass character, a hyphen (-). The presence of this character at the beginning of a file entry signifies that TCP2PROD should *not* process such an entry.

Note: It is suggested that *all* entries within a given section be maintained within the IBM-supplied catalog files. Doing so allows TCP2PROD to report such bypassed entries when files are processed, and also allows file entries to more readily be distinguished from comments as a catalog file is modified over time.

If a given TCP/IP for z/VM file is not required for use by your installation, its corresponding entry should be bypassed as just described, rather than deleted or changed to a comment.

src_dclvar A PPF :DCL. variable name for the (*source*) minidisk or SFS directory resource where a source file resides. For file exclusion entries, a wildcard value (%*) may be specified that allows an entry to be matched to a given source resource when files are processed.

prd_dclvar A PPF :DCL. variable name for the (*target*) minidisk or SFS directory resource where a copied production file is to reside. For file exclusion entries, a wildcard value (%*) is accepted, although this operand serves only as a positional place holder (that is, the specification of a wildcard for this operand has no effect on file exclusion processing).

For entries within the TCPBFS section, a *program variable name*, of the form <*name*> is used in the place of a conventional PPF :DCL. variable name, to signify that program-specific actions should be taken to process the designated source file. The only variable recognized at this time is **BFS**.

src_fname The file name for a given *source* file. An asterisk (*) can be specified as wildcard value to signify that all files of the type specified by *src_ftype* are to be processed. When a wildcard (*) is used, the production file name remains unchanged from the source file name.

proc_opt An entry-specific processing option. Such options affect how the TCP2PROD command processes (copies) files that are associated with the designated entry. Entry processing options are valid only for wildcard (*) *source* file names, and are delimited by a slash (/), with no intervening spaces. Processing options recognized by TCP2PROD are further explained in A.2.5, “Entry Processing Options” on page 111.

src_ftype The file type for a given *source* file. An asterisk (*) can be specified as wildcard value to signify that all files of the type specified by *src_fname* are to be processed. When a wildcard (*) is used, the production file type remains unchanged from the source file type.

prd_fname The file name for a given *target* production file. If the *source* file name is specified as a wildcard (*), *prd_fname* must still be specified to maintain correct entry format. For clarity, it is suggested that an equal sign (=) be specified for *prd_fname* in such a case (though any specified value is processed as if `'='` had been specified).

prd_ftype The file type for a given *target* production file. If the *source* file type is specified as a wildcard (*), *prd_ftype* must still be specified to maintain correct entry format. For clarity, it is suggested that an equal sign (=) be specified for *prd_ftype* in such a case (though any specified value is processed as if `'='` had been specified).

comment Commentary text that is ignored by TCP2PROD during processing.

Notes:

1. All operands must be separated by at least one space.
2. Comment lines within a catalog file must begin with an asterisk (*). Such lines are ignored during TCP2PROD processing.
3. Literal resource values (such as a minidisk device address or an SFS directory name) are not accepted in place of the *src_dclvar* or *prd_dclvar* operands.
4. :DCL. wildcard values (%*) are unique to the TCP/IP catalog file and are *not* supported (or present) within a VMSES/E PPF file. These values should be used only to define a file exclusion entry that can be referenced during a wildcard file copy operation.
5. For file exclusion entries, the *prd_fname* and *prd_ftype* operands (if specified) are treated as commentary information.

A.2.5 Entry Processing Options

Options that can be specified as part of a catalog file entry to affect TCP2PROD copy processing are:

UP Instructs TCP2PROD to use the CMS COPYFILE **UPCASE** option when files are copied to production resources.

XCLUDE:*sect_name*
Instructs TCP2PROD to exclude one or more files when processing is performed for a wildcard catalog entry. The files to be excluded must be listed in a separate catalog file section (identified by *sect_name*) that is specifically defined for this purpose.

Multiple section names may be specified for a given XCLUDE option (with each name separated by a colon), from which a cumulative exclusion list is generated. See A.1.5, “TCP2PROD File Exclusion Support” on page 106 for more information about TCP2PROD file exclusion support.

Notes:

1. The **XCLUDE** option can be specified for only a ***wildcard*** entry within a *general* catalog section, and for only the ***source file name*** of such an entry. This restriction stems from the presumption that any files that are to be excluded from TCP2PROD operations are a subset of a substantially larger file group — that is, a group that is more readily processed through file name and file type wildcard (*) pattern matching, than on a file-by-file basis.
2. When multiple entry options are specified, do not include intervening blanks between operands or delimiters.

3. To maintain compatibility with prior option processing support, a period (.) is also accepted as an option delimiter. However, mixed use of this alternate value and the preferred delimiter (/) is not supported.

A.2.6 Catalog Entry Types

The different types of catalog entries that can be defined within a TCP/IP for z/VM catalog file are described here:

Entry Type **Description**

Conventional A conventional (or, general) entry is one that identifies an individual file that is to be processed without special consideration. Such an entry can be used in any catalog definition section. An example of such an entry is:

```
&BLD1Z   &BLD0Z   PROFILE  STCPIP   =  =    *A comment...
```

Wildcard A *wildcard* catalog entry is similar to a conventional entry, but is one in which the source file name or source file type (or both) is specified as an asterisk (*). For such an entry, all of the files to which this wildcard pattern is matched are processed. An example of such an entry is:

```
&BLD1Z   &BLD0Z   *        MODULE   =  =   Commentary text
```

Exclusion An *exclusion* catalog entry is somewhat different from the previous types and is used to exclude one or more files from being processed. This type of entry is further described in A.2.8, “File Exclusion Definitions” on page 113.

A.2.7 Conventional Catalog Definitions

While the entries described in the previous section identify specific files or file groups to be processed by TCP2PROD, they do not convey how those files are to be processed. The manner in which files are processed is controlled to a large extent by the *name* associated with a catalog definition (specifically, by its delimiting *begin* tag). The connotation of a section name, and its effect on TCP2PROD processing is further explained here.

A.2.7.1 Sample and Configuration Definitions

A section defined by a *begin* tag that contains either of the strings ***sample*** or ***config*** (meaning *configuration*) must contain only conventional, non-wildcard entries. This requirement exists because TCP2PROD expects to process the entries within such sections (and the files they identify) on an individual basis.

The processing of sample and configuration files on an individual basis is done to allow for a proper comparison of a given source file and its production counterpart. When differences between such files are detected, TCP2PROD then can provide appropriate notification and process the subject source file accordingly.

However, it is important to note that when differences are detected for a ***sample*** file, the *production* instance is **always** replaced by its *source* equivalent. This is because such a file is assumed to be strictly a product sample file — that is, one that has not been customized. When a sample file is replaced, TCP2PROD provides notification that the subject file has been updated (presumably from the application of TCP/IP service).

By contrast, when file differences are detected as entries in a ***configuration*** definition section are processed, the default action is to ***not replace*** the production instance of a given file. This is done to avoid the overlay of a (presumably) customized TCP/IP configuration file. When differences are detected, TCP2PROD provides notification to this affect, which includes readily available attribute information for the subject files.

A.2.7.2 General Catalog Definitions

A catalog section that contains conventional entries, but is neither a sample or configuration section, is considered to be a *general* catalog section. The `TCPRUN` section, defined in the supplied 5VMTCP30 CATALOG file, is an example of such a section. General catalog definitions may include a mixture of conventional or wildcard entries, for which (by default), no special actions are performed as files are copied from their respective source locations to designated production resources.

However, it is with such general definitions that TCP2PROD file exclusion support can be used. The definitions and entries required to exploit this support are described in the next section.

A.2.8 File Exclusion Definitions

The entries described in this section are used to exclude one or more source files from being copied to a production resource. Such entries are referred to as file *exclusion* entries, with the definition section that incorporates them known as a file *exclude* section.

Because file exclusion support is *source file* oriented, a *production* file name and file type are not required for, or applicable to, exclusion entries. This stems from the intended use of these entries, for which identification of only one or a group of source files is required. For this reason, production file name and file type operands are treated as commentary information when they're included as part of an exclusion entry.

A second attribute that distinguishes exclusion entries from their conventional counterparts is the accommodation of a ":DCL. wildcard" value that is unique to the TCP/IP for z/VM catalog file. This wildcard (`%*`) can be specified in the place of a PPF :DCL. variable name so an exclusion entry can be matched to the *source* :DCL. variable name of a conventional catalog entry. When such an entry is processed (and, an **XCLUDE** option is present), the files identified by the matched exclusion entry are omitted from the set of files that are copied to the designated production location. Thus, the “:DCL. wildcard” allows exclusion entries to be defined such that select files can be excluded from TCP2PROD operations, regardless of the (source) resource on which they may reside.

Note that the use of a :DCL. wildcard is not required for an exclusion entry. A PPF :DCL. variable name may still be specified, to identify a specific source resource from which designated files are to be excluded.

The files that are to be excluded by an exclusion entry may be identified by a literal file name and file type, or by using conventional CMS file pattern matching techniques that employ the wildcard (*) and pattern matching (%). Combinations of a literal file name and a file type “pattern” (and vice versa) may also be used.

In the example that follows, a wildcard entry and a separately defined exclude section are illustrated which, in combination, specify that all files for the given &BLD1Z source resource — except those identified in the *XTEST* exclude section — are to be copied to the &BLD0Z production resource.

```
  ...

  &BLD1Z   &BLD0Z   */XCLUDE:xtest  *   =  =   *No XTEST files

  ...

:XTEST.
  %*     %*   *    SAMP*     ** Do not copy SAMP* variations
  %*     %*   *    SEXEC     ** Do not copy any SEXEC files
:EXTEST.

  ...
```

File exclusion is performed by TCP2PROD when the **XCLUDE** processing option is specified for a “conventional” catalog entry. This option also identifies the catalog section which defines the exclusion entries that are to be applied during file processing.

Notes:

1. The **XCLUDE** option can be specified for only a ***wildcard*** entry within a *general* catalog section, and for only the ***source file name*** of such an entry. This restriction stems from the presumption that any files that are to be excluded from TCP2PROD operations are a subset of a substantially larger file group — that is, a group that is more readily processed through file name and file type wildcard (*) pattern matching, than on a file-by-file basis.
2. To be effective, file exclusion entries must be defined using catalog definitions that are separate from conventional file processing entries. Exclusion entries that are encountered outside of a file *exclude* section are ignored by

TCP2PROD, if they can be discerned as such, as would be the case for those that employ :DCL. wildcard values. An exclusion entry that does not incorporate such values may be identified as being not valid for other reasons, or in some cases, may be construed as a conventional catalog entry.

A.2.9 Customization Notes

1. It is advised that any TCP/IP for z/VM CATALOG file changes that are required for your environment be made via a VMSES/E local modification, to allow for the reporting of service-related changes during VMSES/E processing. See Appendix B, “Modifying TCP/IP for z/VM CATALOG Files” on page 127 for more information about how to change a TCP/IP for z/VM CATALOG file in this manner.
2. The source and target minidisk/directory variable names used within this file correspond to those used within the TCP/IP for z/VM ($)PPF file (or an override variation of that file). If any changes are made to the Variable Declarations (:DCL.) section of the TCP/IP for z/VM PPF file via a PPF override, you may need to incorporate similar changes within TCP/IP for z/VM CATALOG files (through separate VMSES/E local modifications) to allow for the correct resolution of PPF :DCL. variable names.
3. :DCL. wildcard values (%*) are unique to the TCP/IP CATALOG file and are *not* supported (or present) within a VMSES/E PPF file. These values should be used only to define a file exclusion entry that is to be referenced during a wildcard file copy operation.

A.2.10 Catalog Files Supplied with TCP/IP for z/VM

The catalog files provided for with TCP/IP for z/VM are listed in Figure 22.

Figure 22. TCP/IP for z/VM Catalog Files

Catalog File Name / Type	Associated Files
5VMTCP30 CATALOG	All TCP/IP for z/VM files

A.2.11 IBM-Supplied Catalog Definition Sections

The catalog sections listed and described here are defined in the catalog file supplied with TCP/IP for z/VM:

Section	Description
TCPBFS	Used to place various TCP/IP for z/VM files into the z/VM Byte File System (BFS). The files listed in this section are those used by the z/VM LOADBFS command to identify specific files for installation into the BFS, their designated location, and how each should be processed.

TCPCONFIG **Optionally** used to create suitably named configuration files for customizing TCP/IP and TCP/IP services for your installation. For reference, the files processed using the this section are listed in Figure 19 on page 69.

TCPHELP Used to process TCP/IP CMS Help files. The files processed using this section are listed in the TCP/IP for z/VM CMS Help file VMSES/E build list (TCPBLHLP EXEC).

TCPPRECONFIG Used to create suitably named configuration files that incorporate *preconfigured* content for using certain z/VM or TCP/IP services (for example, for running z/VM system-default VSWITCH controllers). For reference, the files processed using the this section are listed in Figure 20 on page 71.

TCPRUN Used to process non-customizable TCP/IP for z/VM run-time files. For reference, the files processed using this section are listed in Figure 18 on page 67.

TCPSAMPLE Used to process customizable TCP/IP sample files. For reference, the files processed using this section are listed in Figure 19 on page 69.

TCPSVMCMS Used to process non-customizable TCP/IP for z/VM run-time files that must reside on individual **CMS-based** server virtual machine (SVM) minidisks. For reference, the files processed using the TCPSVMCMS section are listed in Figure 16 on page 65.

TCPSVMGCS Used to process non-customizable TCP/IP for z/VM run-time files that must reside on individual **GCS-based** server virtual machine (SVM) minidisks. For reference, the files processed using the TCPSVMGCS section are listed in Figure 17 on page 66.

The IBM-supplied 5VMTCP30 CATALOG file also includes the (file) *exclude* definition sections that follow. These sections are used in conjunction with those previously described, to exclude the indicated files for file groups from TCP/IP for z/VM production minidisks:

Section **Description**

XTCDMMY Used to active *automatic* exclusion of unchanged files, for processing wildcard entries to which other, specific exclusion is not applicable. This avoids unnecessarily processing source files that have not been modified since such files were last placed into production.

XTCPMAP Used to exclude module load MAP (and similar) files that result when serviced objects are rebuilt.

XTCPMISC Used to exclude various TCP/IP for z/VM files that are not pertinent to the use or provision of TCP/IP services (such as the TCP2PROD command and the 5VMTCP30 CATALOG file).

XTCPSAMP Used to exclude TCP/IP for z/VM *sample* configuration files from copy operations performed for non-customizable TCP/IP files (those processed for the **TCPRUN** section). Such exclusion allows for separate processing of these files, during which file updates can be reported.

Notes:

1. To ensure you are notified of any service-related changes to the 5VMTCP30 CATALOG file, make changes to this file using a **VMSES/E local modification**. See Appendix B, “Modifying TCP/IP for z/VM CATALOG Files” on page 127 for information about how to change the 5VMTCP30 CATALOG file in this manner.
2. When changes are made, ensure the only files identified for TCP2PROD processing are those associated with the servers defined for your environment.

A.3 TCPCMLST Command

A.3.1 Purpose

Use the TCPCMLST command to generate a file that lists PTF-numbered parts for which VMSES/E COMMIT processing may be applicable. The generated file (*ppfname* $REMLIST) can be used as input to the VMSES/E **VMFREM** command, which commits specific service levels for your maintenance environment.

Note: The TCPCMLST command is intended for use by the 5VMTCP30 user ID, and should only be used when you commit service levels for TCP/IP for z/VM files.

```
►►──TCPCMLST──ppfname──ftype_abbrv──fm──────────────────────────────►◄
```

A.3.2 Operands

ppfname The name of the usable form product parameter file used for installing and maintaining TCP/IP for z/VM; the file type must be **PPF**.

ftype_abbrv The 3-character abbreviation used for PTF-numbered files that correspond to the “real” (or, *base*) CMS file types used for TCP/IP for z/VM files. For example, **RPM** is the part-type abbreviation used for TCP/IP parts that have a base file type of **RPMBIN**. The mapping of file type abbreviations and their corresponding base file types can be found in the VM SYSABRVT file.

fm The file mode of the minidisk or directory on which PTF-numbered parts of concern are maintained. By convention, this is the TCP/IP for z/VM **DELTA** minidisk (**5VMTCP30 2D2**, by default) or an equivalent SFS directory.

A.3.3 Usage Notes

1. A minidisk or directory must be accessed at file mode A with Read/Write (R/W) status, to allow for the creation of files by TCPCMLST.
2. TCPCMLST creates the files listed in Figure 23 (dependent upon current maintenance circumstances):

Figure 23. TCPCMLST - Generated Files

File Name / File Type	Content
ppfname $REMLIST	Lists PTFs that are candidates for commit processing. This file is created when PTF-numbered parts exist that correspond to the selected *ftype_abbrv* abbreviation).
ppfname $CMLSTLG	Lists PTFs identified for commit processing through prior TCPCMLST invocations. This file is produced (or updated) when a *ppfname* $REMLIST file already exists and PTF commit candidates are identified by TCPCMLST.
ppfname $BASLIST	Lists base-level parts that can be removed *after* commit processing has been completed for PTFs listed in the *ppfname* $REMLIST file. The base-level parts listed correspond to one (or more) of the listed PTF-numbered parts.
ppfname $CMBASLG	Lists base-level parts identified for removal through prior TCPCMLST invocations. This file is produced (or updated) when a *ppfname* $BASLIST file already exists and TCPCMLST is invoked and base-level removal candidates are identified along with PTF commit candidates.

3. If TCPCMLST is invoked with *ppfname* specified as a question mark (**?**), the command syntax is displayed.

A.3.4 Return Codes

Return Code	Description
0	Successful execution; no processing errors were encountered.
1	Incorrect invocation. TCPCMLST was invoked with an incorrect number of operands. A message that identifies the missing operand is displayed, in addition to the command syntax.

2 Internal error. If return code 2 is returned, processing status is unknown. Contact the TCP/IP for z/VM support group for problem determination and assistance in addressing this type of error.

8 Errors encountered; processing has not completed successfully.

A.4 TCPSLVL Command

A.4.1 Purpose

Use the TCPSLVL command to display service information that is intrinsic to a TCP/IP executable MODULE file. The information presented is obtained from data that is embedded within the various TEXT decks (files) that comprise a given MODULE.

Note: The TCPSLVL command is intended for use as a diagnostic aid, in consultation with the IBM TCP/IP support group.

```
►►──TCPSLVL──part_fn──part_ft──part_fm──────────────────────────────►

►──┬──────────────────────────────────────┬──────────────────────────►◄
   └─(──┬───────┬───┬──────────────────┬──┘
        └─DEBUG─┘   └─SELect──filter───┘
```

A.4.2 Operands

part_fn The file name of the TCP/IP executable file from which service information is to be obtained.

part_ft The file type of the TCP/IP file from which service information is to be obtained. Internal service information is available for only TCP/IP MODULE files.

part_fm The file mode of the minidisk or directory on which the file of interest resides. When specified as an asterisk (*), the first file present in the current search order, which matches the provided *part_fn* and *part_fn*, is evaluated.

A.4.3 Options

DEBUG
Causes supplementary messages and data to be reported for diagnostic purposes.

SELect *filter*
Specifies a character string that is used to limit response information to entries that match the value of *filter*.

A.4.4 Usage Notes

1. When TCPSLVL examines the MODULE you specify, it produces an output line for each TEXT deck in which maintenance data is present. Each line begins with the keyword **SLVL**, followed by the name of a TEXT deck, and its corresponding service indicator. This indicator may reflect either an Authorized Program Analysis Report (APAR) number or an IBM development tracking number. This information, taken as a whole, then can provide an overall (or perhaps, “rule of thumb”) indication of the service that is incorporated within a given module.

2. The TCPSLVL command and the information it provides are intended to supplement the information and files that are maintained or used by VMSES/E and its various utilities. TCPSLVL data should be used, at most, only to form generalizations about the service content of TCP/IP modules.

A.4.5 Examples

- This TCPSLVL command that follows checks the service content of the LPR MODULE that resides at file mode “P”:

  tcpslvl lpr module p

  For this command, the results produced might be:

  ```
  SLVL  LPRP      MT03713
  SLVL  CMERUPT   PQ65653
  SLVL  CMHOSTN   MT03200
  SLVL  CMRESOL   PQ28862
  SLVL  CMXTRPT   PQ68463
  ```

  In this example, APAR level numbers for the CMERUPT, CMRESOL, and CMXTRPT TEXT files are listed. whereas for the LPRP and CMHOSTN TEXT files, internal IBM development tracking numbers are shown.

- This next example adds the **SELECT** option to the previous command, to limit results to entries associated with APAR updates — specifically those that begin with the string PQ6:

  tcpslvl lpr module p (sel pq6

  Based on the results shown for the previous example, the result now produced would be:

  ```
  SLVL  CMERUPT   PQ65653
  SLVL  CMXTRPT   PQ68463
  ```

A.4.6 Return Codes

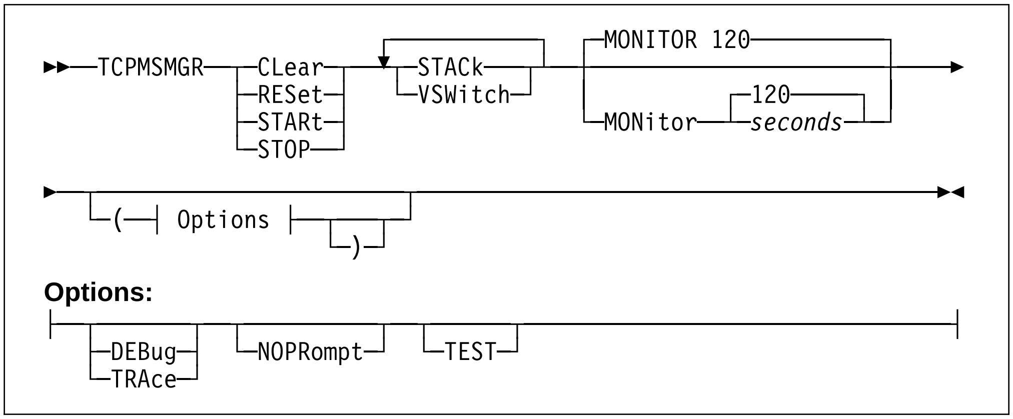
Return Code	Description
0	Successful execution; no processing errors were encountered.
1	Incorrect command invocation, or “help” was requested. TCPSLVL was invoked with an incorrect number of operands, or was invoked with a question mark (?) as the first (or only) operand. In response, the command syntax is displayed.
nn	Processing error. A nonzero return code (other than 1) indicates an problem was encountered when the file was evaluated. Such a return code will be presented when the specified file cannot be located, or when an error occurs when file contents are examined.

A.5 TCPMSMGR Command

A.5.1 Purpose

Use the TCPMSMGR command to shutdown (stop) or initialize (start) the set of **TCP/IP stack** servers, **VSWITCH controller** virtual machines, or both, that are defined for your installation. The virtual machines that are to be stopped or started using this command are identified based on `:stack` class definitions that are present within available DTCPARMS files.

Note: The TCPMSMGR command has been provided as an aid for stopping and starting the indicated groups of servers as part of the z/VM service procedures. However, it can be used in a stand-alone manner (provided the appropriate operational environment is established).



A.5.2 Operands

CLear
RESet

Causes saved GLOBALV values used by the program to be cleared. Variables for STACK and VSWITCH processing are reset independent of one another using this operand. Thus, a STACK or a VSWITCH operand **must also be specified** when the CLEAR operand is used. RESET is synonymous with CLEAR.

If *test mode* values are to be cleared, include the TEST option as part of the command.

STARt

Initiates the start-up of TCP/IP stack or VSWITCH controller servers that were previously stopped via this program. Such servers are identified by saved GLOBALV values, as set through use of the STOP command function.

STOP

Initiates a shutdown of active TCP/IP stack or VSWITCH controller servers, as defined by applicable DTCPARMS files.

STACk

Directs TCPMSMGR START or STOP operations to affect the set of TCP/IP stack servers that are defined for the system, or signifies that GLOBALV variables which identify such servers should be cleared (for a CLEAR operation).

VSWitch

Directs TCPMSMGR START or STOP operations to affect the set of VSWITCH controllers that are defined for the system, or signifies that GLOBALV variables which identify such servers should be cleared (for a CLEAR operation).

MONitor *seconds*

Specifies the time (duration) for which a server should be monitored for reaching a logoff state, once it has successfully received a shutdown command. The default is 120 seconds, with minimum and maximum values of 10 and 360 seconds, respectively.

If the specified value is not a multiple of the internally defined monitoring interval of 10 seconds, the supplied value is rounded to the nearest such value. This operand is ignored for START and CLEAR operations.

A.5.3 Options

DEBUG
TRACE

Causes supplementary messages to be issued, to provide information for diagnostic purposes. Some supplementary messages (prefaced with a header of the form: `DTCMSM---->`) are also issued when this option is used. The DEBUG and TRACE options are synonymous.

NOPRompt

Prevents the issuance of confirmation prompts. An ***affirmative response*** (**1**) is assumed for prompts that are bypassed through use of this option.

TEST

Instructs TCPMSMGR to operate in *test* mode. Test mode allows one to see how the various servers identified for a START or STOP operation will be dealt with by TCPMSMGR, without taking direct action against those servers.

Note that because no such action is taken, successful command operations are assumed. Thus, any error handling that might be required for a non-test operation is likely not be evident.

A.5.4 Usage Notes

1. This command is intended for use by an appropriate TCP/IP or system administrative user ID (such as **TCPMAINT** or **MAINT**) that is authorized to use privileged TCP/IP functions. (That is, a user ID that is included in appropriate **OBEY** statement lists, as defined within the TCP/IP server configuration files that pertain to your installation). In lieu of such authorization, a privilege class sufficient to use the CP FORCE command is necessary.

2. A privilege class sufficient to use the CP QUERY CONTROLLER ALL command is necessary to use the TCPMSMGR command.

3. When the **TEST** option is used, the server monitoring time is forced to a period of **three** seconds, with a one second interval applied. This is done to portray the fact that such delays would occur during normal operations, even though no actions are taken under test mode to stop or start a given server virtual machine.

A.5.5 Return Codes

Return Code	Description
0	Successful execution; no processing errors were encountered.
1	Incorrect invocation. TCPMSMGR was invoked with an incorrect number of operands, or one or more operands that are not recognized.

2 Internal error. If this return code is produced, processing status is indeterminate. Contact the TCP/IP for z/VM support group for problem determination and assistance in addressing this type of error.

3 TCP/IP for z/VM configuration error encountered; processing is cancelled upon the identification and reporting of such a problem.

4 Errors encountered, with warnings issued. The errors encountered may have caused processing to complete with only partial success. Review the messages produced by the command for information about any problems that were encountered.

8 Errors encountered; processing has not completed successfully. Review the messages produced by the command for information regarding the problems encountered.

A.6 MIGVMTCP Command

A.6.1 Purpose

The MIGVMTCP command is a VMSES/E MIGRATE command exit that performs a collective evaluation of customized TCP/IP for z/VM files that are in use on a production z/VM system. This evaluation attempts to identify configuration files that are associated with IBM-supplied sample file counterparts, so that the VMSES/E migration procedures can properly manage such customized files as they are migrated to a new z/VM system (which may also include the reporting of customization changes that might be required, due content changes in IBM sample counterparts).

Note: The MIGVMTCP command has been provided for use by VMSES/E MIGRATE command, and is designed to operate on a z/VM system that is the intended target of a migration operation. The MIGVMTCP command **is not intended for use in a stand-alone manner**.

```
►►──MIGVMTCP──┬─PRE──┬──┬──────────────────────────┬──────────────►◄
              └─POST─┘  └─(──┤ Options ├──┬─────┬──┘
                                          └─)───┘
Options:
├──┬─────────┬──┬───────────┬─────────────────────────────────────────┤
   ├─DEBug───┤  └─VERBose───┘
   └─TRAce───┘
```

A.6.2 Operands

PRE Instructs the MIGVMTCP command to perform preparation operations that pertain to the migration of TCP/IP for z/VM. When this operand is used, the MIGVMTCP command analyzes the various configured files and attempts to associate these files with IBM-supplied sample file counterparts. In addition, selected TCP/IP server minidisks (those for server virtual machines defined in the relevant system TCP/IP PPF file) are evaluated, again to discern server-specific configuration files from those that are not pertinent to the migration procedure. The results of this evaluation then are used to update various VMSES/E tables that are referenced by the VMSES/E MIGRATE command.

POST Instructs the MIGVMTCP command to perform follow-on operations that pertain to the migration of TCP/IP for z/VM. At present, no specific actions are performed when this operand is used.

A.6.3 Options

DEBUG
TRACE

Instructs the MIGVMTCP command to log internal data and logic information in a file (migvmtcp $DEBUG), for diagnostic purposes. Some supplementary messages (prefaced with a header of the form: `DTCMIG---->`) are also issued when this option is used. The DEBUG and TRACE options are synonymous.

VERBose

Causes supplementary messages to be issued to provide information for diagnostic purposes. Messages produced from using this option are prefaced with a header of the form:

- `DTCMIG....:`
- `DTCMIG....*>`

A.6.4 Usage Notes

1. The DEBUG, TRACE, and VERBOSE command options are intended for diagnostic use, in consultation with the IBM TCP/IP support group

A.6.5 The MIGVMTCP $MSGLOG File

Pertinent informational, warning and error messages that are issued to the console by MIGVMTCP are also recorded in a message log file, MIGVMTCP $MSGLOG. This log file is written to the minidisk or directory accessed at file mode A, and can be viewed using the VMSES/E **VMFVIEW** command.

The MIGVMTCP $MSGLOG is cumulative, with the most recent entries appended at the **top** of the file. Separator headers that include date and time stamps are inserted in the log with each MIGVMTCP invocation so newer log entries can be distinguished from older ones.

Notes:

1. Messages are not logged until MIGVMTCP has completed an initial validation of supplied operands.
2. Diagnostic and other incidental messages are not recorded in the MIGVMTCP $MSGLOG file.

A.6.6 Return Codes

Return Code	Description
0	Successful execution; no processing errors were encountered.
4	Errors encountered, with warnings issued. The errors encountered may have caused processing to complete with only partial success. Review the messages produced by the command for information about any problems that were encountered.
8	Errors encountered; processing has not completed successfully. Review the messages produced by the command for information regarding the problems encountered.
9	TCP/IP for z/VM configuration error encountered; processing is cancelled upon the identification and reporting of such a problem.
10	Incorrect invocation. MIGVMTCP was invoked with an incorrect number of operands, or one or more operands that are not recognized.
11	Internal error. If this return code is produced, processing status is indeterminate. Contact the TCP/IP for z/VM support group for problem determination and assistance in addressing this type of error.

Appendix B. Modifying TCP/IP for z/VM CATALOG Files

This appendix describes how to create a VMSES/E local modification for TCP/IP for z/VM CATALOG files, which are provided as replacement-maintained (or, “full-source”) objects. This procedure can also be used to modify other replacement-maintained TCP/IP for z/VM files, such as sample files.

The example provided in this appendix describes how to modify the 5VMTCP30 CATALOG file to prevent files associated with the **LPD** server from being processed by the TCP2PROD EXEC.

For more information about installing and maintaining local modifications, see Chapters 5, 6, and 7 of the *z/VM: Service Guide* (GC24-6117).

Note: Throughout this procedure, references are made to several TCP/IP for z/VM installation minidisks. Default device numbers (and equivalent SFS directories, when applicable) for these minidisks are listed in the tables in 5.3, “DASD Storage and User ID Requirements” on page 44.

1 Log on the installation user ID, **5VMTCP30**.

The PROFILE EXEC provided for this user ID (as part of the z/VM version 5 release 3 System Deliverable) contains ACCESS commands for the required VMSES/E minidisks — the VMSES/E code minidisk (MAINT 5E5, by default) and the VMSES/E Software Inventory minidisk (MAINT 51D, by default).

2 Issue the CMS QUERY DISK command to verify the VMSES/E code and Software Inventory minidisks are correctly linked and accessed.

query disk

Verify the MAINT 5E5 minidisk is accessed as file mode **B**, and is linked **R/O**.

Verify the MAINT 51D minidisk is accessed as file mode **D**, and is linked **R/W**.

Note: If another user has the MAINT 51D minidisk linked in write (R/W) mode, you'll obtain only read (R/O) access to this minidisk. If this occurs, have that user re-link the 51D disk in read-only (RR) mode, after which you need issue the appropriate LINK and ACCESS commands for the 51D minidisk. Do not continue with these procedures until a R/W link is established to the 51D minidisk.

3 If necessary, establish the appropriate access to the VMSES/E minidisks.

a Establish read access to the VMSES/E code minidisk.

```
link maint 5e5 5e5 rr
access 5e5 b
```

b Establish write access to the Software Inventory minidisk.

```
link maint 51d 51d mr
access 51d d
```

4 Create and apply the local modification.

a Run the LOCALMOD command to create the local modification for the catalog file.

localmod *compname partfn partft*

where:

- *compname* is the installed component name with which the configuration file is associated. For example, for TCP/IP for z/VM (5VMTCP30), you would use either: **tcpip** or **tcpipsfs**
- *partfn* and *partft* are the real file name and file type of the file being modified. For example: 5VMTCP30 CATALOG

b Reply to any prompt messages.

c Make your changes to the displayed file.

For example, assume you've chosen to not use the LPD server provided with TCP/IP for z/VM. To avoid TCP2PROD processing of files associated with the LPD server, you need to bypass the 5VMTCP30 CATALOG file shown here:

```
:TCPSVMCMS.
    ...
- &BLD1Z   &DISK7     TCPROFIL EXEC     PROFILE  =          LPSERVE
    ...
:ETCPSVMCMS.
    ...
    ...
:TCPSAMPLE.
    ...
- &BLD1Z   &BLD0Z     LPD      SCONFIG  =        =          LPSERVE
    ...
:ETCPSAMPLE.
    ...
    ...
:TCPCONFIG.
    ...
- &BLD0Z   &BLD4Z     LPD      SCONFIG  =        CONFIG     LPSERVE
    ...
:ETCPCONFIG.
```

d File your changes.

====> file

5 Run the SERVICE command to build the local modifications.

Note

When you have completed steps 4 through 4d for all of the local modifications necessary for this component, then rebuild these objects.

Build the modified (or, serviced) configuration file(s).

service *compname* **build**

where:

compname is the same as that described in step 4 on page 128.

Appendix C. Modifying the TCP/IP for z/VM Default Installation

As the installation and configuration of TCP/IP for z/VM is completed, you may identify one or more TCP/IP services that are not required for your installation. Alternatively, it may be desirable (or necessary) to modify certain aspects of the TCP/IP default installation, to meet requirements that are specific to your organization.

If you choose to modify or eliminate any of the IBM-supplied resources for TCP/IP for z/VM, the changes outlined here should be considered prior to making any changes.

Note — z/VM Automated Service Procedure

If you modify any of the IBM-supplied user IDs, minidisk addresses, or SFS directory names that are associated with TCP/IP for z/VM and you plan on using the z/VM automated service procedure (the **SERVICE** and **PUT2PROD** commands) to service your z/VM system, then you must create a PPF override for the **SERVP2P $PPF** file.

You must also use the **VMFUPDAT** command to update the VM SYSSUF software inventory file, so that your PPF override of SERVP2P PPF is used for automated service processing. For more information about PPF file overrides, see the *z/VM: VMSES/E Introduction and Reference*

- The **5VMTCP30** user ID is the IBM-supplied user ID for installing and servicing TCP/IP for z/VM. If you choose to use a different user ID or you elect to use different minidisks and/or SFS directories for TCP/IP for z/VM maintenance purposes, suitable PPF overrides must be implemented to reflect your changes.

  Note: If you choose to use an existing or "common-use" user ID (such as the MAINT user ID) to install and maintain TCP/IP for z/VM, minidisks may already be defined which have device numbers that conflict with TCP/IP for z/VM minidisk defaults. If such minidisks exist, you will need to create a PPF override to change the TCP/IP default minidisk device numbers so they are unique within your environment.

- If you choose to use user IDs for TCP/IP server virtual machines that differ from the IBM-supplied user IDs shown in Figure 13, review the section titled "Implications of Assigning Different Server Virtual Machine Names" in Chapter 1 of *TCP/IP Planning and Customization*.
- Note the following, with regard to the user ID and minidisk information provided in Figure 13 on page 48 of 5.3.1, "DASD Requirements for TCP/IP for z/VM":

Specific Minidisk Requirements

Certain minidisks **must** be defined for the TCP/IP server machines you choose to use, as well as for maintaining TCP/IP for z/VM in your environment. Such minidisks **cannot** be replaced with an equivalent SFS directory.

Minidisks to which this requirement applies are listed in Figure 13 with **boldface** virtual device numbers. In addition, no default SFS directory names or SFS 4K block values are provided for these minidisks.

- The minidisks that are associated with the **5VMTCP30**, **TCPMAINT**, and **TCPIP** user IDs (or your chosen equivalents) **must** be maintained in order to provide TCP/IP services within your environment. The remaining user IDs listed in Figure 13 on page 48 of 5.3.1, “DASD Requirements for TCP/IP for z/VM” are associated with servers and clients that provide optional TCP/IP for z/VM services.
- To identify TCP/IP services that are not required for your installation, review the service descriptions provided in *TCP/IP Planning and Customization* (SC24-6125). The server-specific configuration chapters of this publication may also help you with determining whether a service or its resources can (or should) be removed.
- Suitable **5VMTCP30 $PPF** file overrides are necessary to avoid references to any TCP/IP user ID that is removed, or to account for any user IDs that are changed. This is also applies to the removal of any minidisks that are uniquely associated with such user IDs.

  If such overrides are created, the **VMFPPF** command must be used to generate a compiled PPF filed that corresponds to your override file name, as well as to create an updated **SERVP2P** PPF file, for use by the VMSES/E SERVICE command.
- The z/VM version 5 release 3 system directory entry for the **5VMTCP30** user ID (or its equivalent) should be modified such that LINK statements to any removed minidisks are no longer present.
- The z/VM version 5 release 3 system directory entry for the **MAINT** user ID (or its equivalent) should be modified such that LINK statements to any removed minidisks are no longer present.
- The content of the 5VMTCP30 CATALOG file (used by the TCP2PROD command to copy files into production) should be reviewed for correctness. It may be necessary to modify this file (via a VMSES/E local modification) to bypass references to files that correspond to any TCP/IP user ID that is removed, as well as to any minidisks that are uniquely associated with that user ID. See Appendix A, “TCP/IP Installation, Service and Migration Utilities” on page 104 for information about the TCP2PROD command and TCP/IP for z/VM catalog files.

Appendix D. Making Local Modifications to TCP/IP for z/VM Modules

This appendix provides information to assist you in making local modifications to TCP/IP for z/VM modules, and is intended to supplement the local modification process described in Chapter 6, “Procedures for Local Service” of the *z/VM: Service Guide* (GC24-6117). The process described herein is oriented toward local service that is implemented through:

- TEXT file ***replacement*** — when an existing TCP/IP TEXT file is fully replaced and the file name of the existing TEXT file remains unchanged
- ***local updates*** to the source for an existing TEXT file (so that a “local replacement” is effectively created)
- TEXT file ***substitution*** or ***addition***, in which an existing TCP/IP TEXT file is fully replaced by one or more new, differently-named TEXT files.

With regard to the latter two types of modification, additional steps to accommodate changes to TCP/IP C and Pascal-based modules or added TEXT files are provided.

When to Use This Information

In general, the VMSES/E local modification process as documented in Chapter 6, “Procedures for Local Service” of the *z/VM: Service Guide* should be used when local modifications are made. *This is especially true* for ***common*** part types — such as EXEC and ASSEMBLE files — for which VMSES/E itself provides appropriate part handlers.

However, C or Pascal-specific VMSES/E part handlers are not available for TCP/IP **C** and **Pascal** source parts, so local modifications to TCP/IP for z/VM modules based on these languages cannot be processed completely within the VMSES/E environment. Instead, the VMFC and VMFPAS execs provided with TCP/IP for z/VM can be used to facilitate local modifications to its C and Pascal-based modules. Additional steps which make use of these execs, are included in this appendix to help you modify TCP/IP C or Pascal-based objects.

Information about the VMFC and VMFPAS execs can be found in the chapter titled “Using Source Code Libraries,” of *TCP/IP Planning and Customization* (SC24-6125).

Note that VMFC and VMFPAS are ***not*** “VMSES/E-compliant” part handlers. They do not process source files in the same manner as VMSES/E part handlers (for example, by using VMFEXUPD); nor do they produce object (TEXT) files that comply with VMSES/E maintenance naming conventions (such as file abbreviations and part numbering). Therefore, several manual steps are required to ensure the TEXT files produced by VMFC and VMFPAS can be used as part of the VMSES/E maintenance process.

For detailed information about installing and maintaining local modifications, see Chapters 5, 6, and 7 of the *z/VM: Service Guide*.

Notes:

1. Throughout this procedure, references are made to several TCP/IP for z/VM installation minidisks. Default device numbers, and equivalent SFS directories (when applicable), for these minidisks are listed in the tables in 5.3, “DASD Storage and User ID Requirements” on page 44.
2. VMSES/E local modifications require a modification identifier, or *modid*, to be associated with the parts affected by a change; *modid* is a locally-determined value. It should begin with **L**, and is followed by up to 4 alphanumeric characters that identify a specific local modification. For example: `L0002`

D.1.1 Prepare for Local Service

1 Log on the installation user ID, **5VMTCP30**.

The PROFILE EXEC provided for this user ID (as part of the z/VM version 5 release 3 System Deliverable) contains ACCESS commands for the required VMSES/E minidisks — the VMSES/E code minidisk (MAINT 5E5, by default) and the VMSES/E Software Inventory minidisk (MAINT 51D, by default).

2 Issue the CMS QUERY DISK command to verify the VMSES/E code and Software Inventory minidisks are correctly linked and accessed.

query disk

Verify the MAINT 5E5 minidisk is accessed as file mode **B**, and is linked **R/O**.

Verify the MAINT 51D minidisk is accessed as file mode **D**, and is linked **R/W**.

Note: If another user has the MAINT 51D minidisk linked in write (R/W) mode, you'll obtain only read (R/O) access to this minidisk. If this occurs, have that user re-link the 51D disk in read-only (RR) mode, after which you need issue the appropriate LINK and ACCESS commands for the 51D minidisk. Do not continue with these procedures until a R/W link is established to the 51D minidisk.

3 If necessary, establish the appropriate access to the VMSES/E minidisks.

a Establish read access to the VMSES/E code minidisk.

link maint 5e5 5e5 rr
access 5e5 b

b Establish write access to the Software Inventory minidisk.

link maint 51d 51d mr
access 51d d

4 Create a usable PPF file.

Note: This VMFPPF step is necessary **only** if you have made changes to the PPF file since it was last compiled.

vmfppf *ppfname compname*

where:

- *ppfname* is the name of a PPF file that corresponds to the source file to be modified. For example, **5vmtcp30**. If you have created an override for the PPF file in question, use your override file name.
- *compname* is the installed component name with which the source file is associated. For example, for TCP/IP for z/VM (5VMTCP30), you would use either: **tcpip** or **tcpipsfs**

5 Establish the correct minidisk access order.

vmfsetup *ppfname compname*

where *ppfname* and *compname* are the same as described in step 4.

6 Access the TCP/IP for z/VM **SOURCE** minidisk (**5VMTCP30 2B3**, by default) or an equivalent SFS directory.

access 2b3 *fm*

where *fm* is an available file mode, perhaps “C.”

D.1.2 Receive the Local Service

D.1.2.1 Create the Replacement TEXT File

Create the replacement TEXT file (or, “text deck”). How you perform this step depends on the nature of the TEXT file itself.

- If your modification is comprised of a ***replacement*** TEXT file, you need only to copy this file to an appropriately named file which VMSES/E can then use.
- If your modification is comprised of a ***new*** (additional) TEXT file, you need to copy this file to an appropriately named file and update the build list that corresponds to the module affected by your changes.

If either of the above cases are applicable, skip the steps provided for “C and Pascal Source Modifications,” and create the required TXT*modid* file, as described in D.1.2.5, “Copy the TEXT File for Use by VMSES/E” on page 138.

- If your modification requires updates to a C or Pascal source file, continue with the steps provided for “C and Pascal Source Modifications.”

C and Pascal Source Modifications

D.1.2.2 Add an Update Record to the AUX File

1 Update or create the AUX file for the part being modified (*fn* AUXLCL) and add an entry for the update file; the default AUX file type is “AUXLCL.”

Note: Perform this step for each source file you need to modify.

```
xedit fn auxlcl fm-local (noprof
===> input update-ft svclvl lcmodid comment
===> file
```

where:

- *fn* is the file name of the source-maintained part being modified.
- *fm-local* is the file mode of the TCP/IP for z/VM local modification (LOCALMOD) disk.
- *update-ft* is the file type of the update file that contains your local modifications.
- *svclvl* is a service level indicator; by convention, the string “LCL” is most often used for local modifications.
- *modid* is the local modification number (as described in Note 2 on page 134). For example: `L0002`
- ***lc****modid* is the string “LC” concatenated with *modid*; for example: `LCL0002`

D.1.2.3 Create the Update File for the Part

1 Xedit the source file (*fn* C **or** *fn* PASCAL) with the **CTL** option.

Note: Perform this step for each source file you need to modify.

```
xedit fn ft-src fm-src (ctl tcpip
```

where:

- *fn ft-src fm-src* are the file name, file type and file mode of the source file you are modifying.
- **tcpip** is the name of the TCP/IP control file.

2 Make your changes to the displayed source file. The original source file is *not* changed.

3 When you have completed your changes, save them on the TCP/IP for z/VM local modification (LOCALMOD) disk. When you enter the FILE command in XEDIT, all of your changes are placed in the update file (*fn update-ft*).

```
===> file = = fm-local
```

where *fm-local* is the file mode of the TCP/IP for z/VM local modification (LOCALMOD) disk.

D.1.2.4 Create an Updated Replacement TEXT File

1 Compile the source to include your updates. Use the VMFC EXEC to compile C source files; use the VMFPAS EXEC to compile PASCAL source files. For more information about the VMFC and VMFPAS execs, see the chapter titled “Using Source Code Libraries,” of *TCP/IP Planning and Customization* (SC24-6125).

Note: Perform this step for each source file you have modified.

a To compile **C**-based source files, issue:

```
vmfc fn tcpip ( options
```

OR

b To compile **Pascal**-based source files, issue:

```
vmfpas fn tcpip ( options
```

where:

- *fn* is the name of the source file to be compiled.
- **tcpip** is the name of the TCP/IP control file.
- *options* are C or PASCAL compiler options required for your environment.

After successful completion of the VMFC (or VMFPAS) EXEC, a text file (*fn* TXTLCL A) incorporating your local modifications will exist.

Note: If you choose to use a different compilation method instead of using the VMFC or VMFPAS execs provided with TCP/IP for z/VM, the resulting TEXT file may be named differently than "*fn* TXTLCL."

End of C and Pascal Source Modifications

D.1.2.5 Copy the TEXT File for Use by VMSES/E

Copy the replacement or new TEXT file (or, "text deck") to the TCP/IP for z/VM local modification (LOCALMOD) disk, with the correct file type for the replacement part.

Note: Perform this step for each TEXT file affected by your modifications.

vmfrepl *fn* **text** *ppfname compname fn ft-txt fm-txt* **($select logmod** *modid* **outmode localmod ftabbr txt**

where:

- *fn* is the file name of the TEXT file in question.
- *ppfname* and *compname* are the same as described in step 4 on page 135.
- *fn ft-txt fm-txt* are the file name, file type and file mode of the TEXT file in question.
- *modid* is the local modification number (as described in Note 2 on page 134). For example: `L0002`
- ***localmod*** is the symbolic name for the local modification disk in the :MDA. section of the 5VMTCP30 PPF file.

D.1.3 (Optional) Modify the Build List for Added TEXT Files

If your local modifications do not require the addition of a new TEXT file, skip the steps provided in "Additional Steps for Adding a New TEXT File," and continue with the steps provided in D.1.4, "Rebuild the Modified Objects" on page 141.

If your local modifications are implemented within a unique TEXT file (that is, one not supplied by IBM as part of TCP/IP for z/VM) you need to perform additional steps for this TEXT file to be incorporated by the affected TCP/IP module. These steps, provided in "Additional Steps for Adding a New TEXT File," implement build list changes that will accommodate the new TEXT file; they would also be required if you find it necessary to replace an IBM-supplied TEXT file with one of a different name.

Additional Steps for Adding a New TEXT File

D.1.3.1 Add an Update Record to the Build List AUX File

1 Update or create the build list AUX file (*fn-blst* AUXLCL) and add an entry for the update file; the default file AUX file type is "AUXLCL."

A complete list of TCP/IP for z/VM build lists is provided in Appendix F, "TCP/IP for z/VM Build Lists" on page 145.

```
xedit fn-blst auxlcl fm-local (noprof
===> input update-ft svclvl lcmodid comment
===> file
```

where:

- *fn-blst* is the file name of the build list used to build the affected module (that is, the build list to which the new TEXT file must be added).
- *fm-local* is the file mode of the TCP/IP for z/VM local modification (LOCALMOD) disk.
- *update-ft* is the file type of the update file that contains your local modifications.
- *svclvl* is a service level indicator; by convention, the string "LCL" is most often used for local modifications.
- *modid* is the local modification number (as described in Note 2 on page 134). For example: `L0002`
- ***lc**modid* is the string "LC" concatenated with *modid*; for example: `LCL0002`

D.1.3.2 Create the Update File for the Build List

1 Xedit the build list source file with the **CTL** option.

xedit *fn-blst* **$exec** *fm-src* **(ctl tcpip**

where:

- *fn-blst* and *fm-src* are the file name and file mode of the build list file you are modifying.
- **$exec** is the file type of the build list.
- **tcpip** is the name of the TCP/IP control file.

2 Make your changes to the displayed source file. The original source file is *not* changed.

Locate the `:OBJNAME.` tag associated with the name of the module that will include your text file. For example, `TCPIP.MODULE`. Add a new `:PARTID.` tag record after the last `:PARTID.` tag for this object (`TCPIP.MODULE`), and before its `:EOBJNAME.` tag.

The new :PARTID. tag should define the file name of the text file you're adding, followed its file type abbreviation, TXT. Use only a single space to separate the tag, the text file name, and the TXT abbreviation, as follows

```
:PARTID. newtxtfn TXT
```

In the example that follows, the TCPBLM91 build list is to be updated to add TEXT files to the TCPIP module. To do this, line 176 is copied (or duplicated); then, in the newly created line(s), the string “TCOFFPR” is changed to the name of the added TEXT file(s).

```
 TCPBLM91 EXCnnnnn I2  V 80  Trunc=80 Size=265 Line=178 Col=1 Alt=nn
====>
00087 :EOBJNAME.
00088 *
00089 :OBJNAME. TCPIP.MODULE AMODE 31 FROM TCPIP NOMAP
00090 :BLDREQ. TCPBLCOM.BLDLIST TCPBL492.TCPASCAL.TXTLIB TCPBL492.TCPLAN
00091          TCPBL492.TCPXXA.TXTLIB
00092 :GLOBAL. TXTLIB COMMTXT TCPXXA TCPASCAL TCPLANG
00093 :OPTIONS. CLEAR NOAUTO RLDSAVE NOLIBE NOUNDEF RMODE 24
00094 :PARTID. TCPIP TXT
00095 ---------------------  80  line(s) not displayed ------------------
00175 :OPTIONS. LIBE UNDEF RESET VSPASCAL
00176 :PARTID. TCOFFPR TXT
00177 :EOBJNAME.
00178 *
```

3 When you have completed your changes, save them on the TCP/IP for z/VM local modification (LOCALMOD) disk. When you enter the FILE command in XEDIT, all of your changes are placed in the update file (*fn-blst update-ft*).

===> file = = *fm-local*

where *fm-local* is the file mode of the TCP/IP for z/VM local modification (LOCALMOD) disk.

D.1.3.3 Create the Updated Replacement Build List

1 Create a replacement part from the build list $EXEC (source) file. To do this, update the build list file with the VMFEXUPD command.

vmfexupd *fn-blst* **exec** *ppfname compname* **(outmode localmod $select logmod**

where:

- *fn-blst* is the file name of the build list being updated. For example, `TCPBLM91`
- *ppfname* and *compname* are the same as described in step 4 on page 135.

The VMFEXUPD command records the update for the build list in the local version vector table (VVTLCL), adds the update to the $SELECT file, and creates the replacement part (*fn-blst* EXEC).

End of Additional Steps for Adding a New TEXT File

D.1.4 Rebuild the Modified Objects

Note

When you have completed the steps described in D.1.2, "Receive the Local Service" through D.1.2.5, "Copy the TEXT File for Use by VMSES/E" (or D.1.3.3, "Create the Updated Replacement Build List") for all of the local modifications necessary for this component, then rebuild the modified objects.

In general, your modification is likely to require other steps associated with the service process to be completed (such as updating the build status table, re-building serviced objects, testing service, and copying the service into production). To complete the service process, continue with one of the steps listed, as appropriate:

- 7.2.2.4, "Update the Build Status Table" on page 81, to complete TCP/IP for z/VM RSU service
- 7.2.3.4, "Update the Build Status Table" on page 91, to complete TCP/IP for z/VM COR service

For reference, a sample VMFBLD command is shown below that can be used to rebuild specific objects within a given build list:

```
vmfbld ppf ppfname compname fn-blst fn-mod.module (serviced
```

where:

- *ppfname* and *compname* are the same as described in step 4 on page 135.

- *fn-blst* is the appropriate TCP/IP for z/VM build list file name. A complete list of TCP/IP for z/VM build lists is provided in Appendix F, "TCP/IP for z/VM Build Lists" on page 145.
- *fn-mod* is the file name of module which incorporates the TEXT file that has been modified.

Also, if you have modified only one module, you may want to manually copy it into production, instead of using the TCP2PROD EXEC. If this is the case, use the VMFCOPY command that follows:

```
vmfcopy fn module fm-bld = = fm-prd (oldd repl sprodid 5vmtcp30%tcpip prodid 5vmtcp30%tcpip
```

where:

- *fn* is the name of the modified module.
- *fm-bld* is the file mode of the TCP/IP for z/VM build disk on which the module was built.
- *fm-prd* is the file mode of the TCP/IP for z/VM production disk where the module should reside.

The VMFCOPY command will update the VMSES/E PARTCAT file on the appropriate TCP/IP for z/VM production minidisk.

Appendix E. Modifying TCP/IP for z/VM VMNFS Code

This appendix provides information to assist you in making local modifications to the TCP/IP for z/VM NFS server module (VMNFS). Modifications would be required for the NFS server to:

- use of a file handle encryption subroutine different from that in NFSFHCIP ASSEMBLE
- validate SMSG requests in a manner different from its current implementation (affects NFSSMSG C)
- report failed minidisk link attempts in a manner different from its current implementation (affects NFSBADPW C).

Certain modifications may also require changes to the TCPBLC91 EXEC, which is the build list used to build the VMNFS module.

Note: TCP/IP source files are distributed as part of the z/VM version 5 release 3 System Deliverable. These files reside on the TCP/IP for z/VM **SOURCE** minidisk (**5VMTCP30 2B3**, by default), or an equivalent SFS directory.

For detailed information about installing and maintaining local modifications, see Chapters 5, 6, and 7 of the *z/VM: Service Guide* (GC24-6117).

E.1 Modifying the NFSFHCIP ASSEMBLE and TCPBLC91 EXEC Files

If you need to modify the NFSFHCIP ASSEMBLE or TCPBLC91 EXEC files, you should follow the steps provided in Chapter 6, “Procedures for Local Service” of the z/VM: Service Guide (GC24-6117). In so doing, the following substitutions may need to be made:

- **zvm** should be: **5vmtcp30**
- *compname* should be: **tcpip** or **tcpipsfs**
- *appid* should be: **5vmtcp30**
- *fm-local* should be the file mode of the 2C4 minidisk
- *fm-applyalt* should be the file mode of the 2A6 minidisk

You may also find some of the information provided in Appendix D, “Making Local Modifications to TCP/IP for z/VM Modules” on page 133 to be useful, such as the steps provided in D.1.1, “Prepare for Local Service.”

Keep in mind that when you get to the following step in the *z/VM: Service Guide*:

- “Rebuilding Objects”

you should return to using this program directory and continue with 7.2.2.4, “Update the Build Status Table” on page 81.

E.2 Modifying VMNFS C Source Files

If you need to modify the source for the NFSSMSG or NFSBADPW C files, use the information provided in Appendix D, “Making Local Modifications to TCP/IP for z/VM Modules” on page 133, especially that which is specific to “C and Pascal Source Modifications.”

Appendix F. TCP/IP for z/VM Build Lists

This appendix provides a complete list of the VMSES/E build lists used to maintain TCP/IP for z/VM. This information has been provided to help you determine which build list to use with VMSES/E commands when you need to build or service specific TCP/IP objects, and to assist you with making local modifications. For more information about build list content and formats, see the *z/VM: VMSES/E Introduction and Reference* (GC24-6130).

The build lists identified in the tables that follow can be found on the 5VMTCP30 2B2 (BASE1) minidisk. However, before using the information within a given build list, the 5VMTCP30 2D2 (DELTA) minidisk should be checked for a more current, serviced counterpart; this will ensure the most current build list file is referenced.

Also, note that the minidisks shown under the “Build String” headings are 5VMTCP30 minidisk defaults. If a PPF override has been used in your environment to change Build String minidisks or SFS directories, use your values when you determine which files are affected by a build list.

F.1 TCP/IP for z/VM Build Lists

Figure 24 lists the VMSES/E build lists used for TCP/IP for z/VM, and provides general information about the objects (files) managed by each:

Figure 24 (Page 1 of 2). VMSES/E Build Lists - TCP/IP for z/VM

Build List Name	VMSES/E Part Handler	Build String (Minidisk)	Build List Description / Affected Objects
TCPBL491	VMFBDCOM	BUILD1 (491)	Full-replacement objects built to the 491 minidisk
TCPBL492	VMFBDCOM	BUILD3 (492)	Full-replacement objects built to the 492 minidisk
TCPBLM91	VMFBDMOD	BUILD1 (491)	MODULE objects built to the 491 minidisk
TCPBLM92	VMFBDMOD	BUILD3 (492)	MODULE objects built to the 492 minidisk
TCPBLC91 (1*)	VMFBDMOD	BUILD1 (491)	C-based MODULE objects built to the 491 minidisk
TCPBLC92 (1*)	VMFBDMOD	BUILD3 (492)	C-based MODULE objects built to the 492 minidisk
TCPBLP91 (1*)	VMFBDTLB	BUILD1 (491)	VMFBDPMD-dependent MODULE objects built to the 491 minidisk
TCPBLP92 (1*)	VMFBDTLB	BUILD3 (492)	VMFBDPMD-dependent MODULE objects built to the 492 minidisk
TCPBLHLP	VMFBDCOM	BUILD8 (29D)	TCP/IP CMS Help Files for z/VM 19D Help minidisk

Notes:

1. Language Environment for z/VM support must be available when building objects identified in this build list.

Figure 24 (Page 2 of 2). VMSES/E Build Lists - TCP/IP for z/VM

Build List Name	VMSES/E Part Handler	Build String (Minidisk)	Build List Description / Affected Objects
TCPBLRPM	VMFBDCOM	BUILD9 (493)	RPM binary files for SSL server Linux guest
TCPBLLDC	VMFBDCOM	BUILD1 (491)	LDAP server message catalog build list
TCPBLLBF	VMFBDBFS	None (BFS)	Facilitates processing of BFS-resident files
TCPBLALL	VMFBDMLB	BUILD3 (492)	ALLMACRO MACLIB build list
TCPBLTRP	VMFBDMLB	BUILD3 (492)	TFTPRP MACLIB build list
TCPBLCSL	VMFBDCLB	BUILD1 (491)	TCPCSLIB CSLIB build list
TCPBLCOM	VMFBDTLB	BUILD3 (492)	COMMTXT TXTLIB build list
TCPBLGDD	VMFBDTLB	BUILD3 (492)	GDDMXD TXTLIB build list
TCPBLXAW	VMFBDTLB	BUILD3 (492)	XAWLIB TXTLIB build list
TCPBLDPI	VMFBDTLB	BUILD3 (492)	DPILIB TXTLIB build list
TCPBLRPC	VMFBDTLB	BUILD3 (492)	RPCLIB TXTLIB build list
TCPBLRPT	VMFBDPMD	BUILD3 (492)	VMRPC TXTLIB build list
TCPBLOLD	VMFBDTLB	BUILD3 (492)	OLDXLIB TXTLIB build list
TCPBLXTL	VMFBDTLB	BUILD3 (492)	XTLIB TXTLIB build list
TCPBLX11	VMFBDTLB	BUILD3 (492)	X11LIB TXTLIB build list
TCPBLSNA	VMFBDLLB	BUILD1 (491)	SNALINK LOADLIB build list
TCPBLXNX	VMFBDLLB	BUILD1 (491)	XNX25 LOADLIB build list
TCPBLSNM	VMFBDLLB	BUILD1 (491)	SNMPLIB LOADLIB build list
TCKBLC91 (1*)	VMFBDMOD	BUILD1 (491)	Kerberos C-based MODULE objects built to the 491 minidisk
TCKBLC92 (1*)	VMFBDMOD	BUILD3 (492)	Kerberos C-based MODULE objects built to the 492 minidisk
TCKBLDES	VMFBDTLB	BUILD3 (492)	DES TXTLIB build list
TCKBLKRB	VMFBDTLB	BUILD3 (492)	KRB TXTLIB build list
TCKBLKDB	VMFBDTLB	BUILD3 (492)	KDB TXTLIB build list
TCKBLBPL	VMFBDTLB	BUILD3 (492)	BPLDBM TXTLIB build list

Notes:

1. Language Environment for z/VM support must be available when building objects identified in this build list.

Appendix G. Moving TCP/IP for z/VM to SFS Directories

By default, TCP/IP for z/VM is installed to minidisks as part of the z/VM version 5 release 3 System Deliverable, during the initial install of z/VM version 5 release 3 itself. However, you can move certain TCP/IP for z/VM minidisks — the **service** minidisks — to Shared File System (SFS) directories. Refer to Figure 13 in 5.3.1, "DASD Requirements for TCP/IP for z/VM" on page 48 to see which minidisks can reside in SFS space. When service disks are moved to SFS directories, you can use either the default file pool (VMSYS) or a file pool of your choosing.

A summary of the steps necessary to move TCP/IP for z/VM service minidisks to SFS space are:

- Allocate space in the user-defined (or default) file pool
- Provide the installation user ID, 5VMTCP30, access to the file pool
- Create the necessary TCP/IP for z/VM subdirectories
- Copy files from minidisks to the new SFS subdirectories
- Create a PPF override, if not using default file pool or subdirectory names.

Where to Next

You need to do one of the following:

- To place TCP/IP for z/VM into SFS directories using ***5VMTCP30 PPF file defaults***, you need to continue with instructions in the *z/VM: CP Commands and Utilities Reference* that provide information about using the **MOVE2SFS** command to move z/VM components to SFS Directories.
- To place TCP/IP for z/VM into your own ***user-defined file pool or SFS directories***, continue with the instructions in G.1.1, "Setup the SFS File Pool and Required Directories" on page 148.

After you have chosen one of the previous options and completed the steps required (whether provided in this appendix or in the *z/VM: CP Commands and Utilities Reference*) you need to return to 6.2.3, "Configure TCP/IP for z/VM for Your Installation" on page 57 and complete the installation of TCP/IP for z/VM.

G.1.1 Setup the SFS File Pool and Required Directories

Notes:

1. The steps that follow help you determine TCP/IP for z/VM file pool space requirements, enroll the 5VMTCP30 user ID in a file pool, and define the required SFS directories. If not all of these steps are required — for example, you are changing only SFS directory names — adapt these instructions as needed for your environment.
2. The steps that follow assume the use of a user-defined SFS file pool name; if you are using the z/VM default file pool name (VMSYS), you should substitute “VMSYS” in place of the text “user-defined file pool” or the variable *userfilepool*, when appropriate.
3. For information about planning for, generating, and managing a file pool and server, see *z/VM: CMS File Pool Planning, Administration, and Operation* (SC24-6074).

1 Determine the number of 4K blocks required for your SFS directories by adding up the 4K block requirements for each SFS directory you plan to use. If you intend to use *all* of the TCP/IP for z/VM SFS directory defaults, the 4K block requirements are summarized in Figure 13 on page 48.

This information will be used when the 5VMTCP30 user ID is enrolled in the user-defined file pool.

2 Enroll user 5VMTCP30 in the user-defined file pool, using the ENROLL USER command:

```
ENROLL USER 5VMTCP30 userfilepool: (BLOCKS blocks
```

where *blocks* is the number of 4K blocks you calculated in the previous step.

Note: This must be done from a user ID that is an administrator for the user-defined file pool.

3 Determine if there are enough blocks available in the file pool to accommodate TCP/IP for z/VM. This information can be obtained via the QUERY FILEPOOL STATUS command. Near the end of the output from this command is a list of minidisks in the file pool and the number of blocks free. If the number of blocks free is smaller than the total number needed to install TCP/IP for z/VM, you need to add space to the file pool before you continue with this process. See *z/VM: CMS File Pool Planning, Administration, and Operation* for information about adding space to a file pool.

4 Use the CREATE DIRECTORY command to create the required subdirectories; the default subdirectories are listed in Figure 13 on page 48. If necessary, refer to the *z/VM: CMS Commands and Utilities Reference* for more information about the CREATE DIRECTORY command.

set filepool *userfilepool:*
create directory *dirid*
create directory *dirid*

dirid is the name of the SFS directory you're creating, such as the default names:

```
create directory userfilepool:5vmtcp30.tcpip
create directory userfilepool:5vmtcp30.tcpip.local
create directory userfilepool:5vmtcp30.tcpip.delta
create directory userfilepool:5vmtcp30.tcpip.applyalt
create directory userfilepool:5vmtcp30.tcpip.applyprod
create directory userfilepool:5vmtcp30.tcpip.object
```

5 If you intend to use an SFS directory as the work space for the 5VMTCP30 user ID, include the following IPL control statement in the 5VMTCP30 directory entry (after the `INCLUDE TCPCMSU` statement):

```
IPL CMS PARM FILEPOOL userfilepool
```

This will cause CMS to automatically access the 5VMTCP30's top directory as file mode A.

G.1.2 Copy Minidisk Files to SFS Directories

1 Copy the files from the TCP/IP for z/VM minidisks (or from the VMSYS file pool, if TCP/IP for z/VM is already installed there) to your new user-defined SFS file pool and directories, using the VMFCOPY command.

Note: Repeat the ACCESS and VMFCOPY commands that follow for each minidisk you need to copy. If necessary, see Figure 13 on page 48 for TCP/IP for z/VM default minidisk device numbers and SFS directory names.

access *vdev* **e**
access *dirid* **f**
vmfcopy * * e = = f (prodid 5vmtcp30%tcpip olddate replace

where:

- *vdev* is the minidisk from which you are copying files.
- *dirid* is the name of the (target) SFS directory to which you are copying files.
- ***5vmtcp30%tcpip*** is the PRODID defined within the 5VMTCP30 PPF file, for both the minidisk *and* SFS components of TCP/IP for z/VM.

The VMFCOPY command will update the VMSES PARTCAT file on the target directory.

G.1.3 Create a Product Parameter File (PPF) Override

This section provides information to help you create a product parameter file (PPF) override. The example used in this section changes the name of the SFS file pool where TCP/IP for z/VM files reside. See the *z/VM: VMSES/E Introduction and Reference* for more information about PPF file overrides.

Note: Do **not** directly modify the product-supplied 5VMTCP30 $PPF or 5VMTCP30 PPF files to change the VMSYS file pool name or any other installation parameters. If the 5VMTCP30 $PPF file is serviced, the existing $PPF file will be replaced, and any changes to that file will be lost. By creating your own $PPF override, your updates will be preserved.

The following process describes changing the TCP/IP for z/VM default file pool name from “VMSYS” to “MYPOOL1”:

1 Create a new $PPF override file or edit an existing override file.

xedit *overname* **$ppf** *fm***2**

overname is the PPF override file name (such as "mytcpip") that you want to use.

fm is an appropriate file mode. If you create this file yourself, specify a file mode of A.

If you modify an existing override file, specify a file mode of A or D, based on where the file currently resides (A being the file mode of a R/W 191 minidisk, or equivalent; D, that of the MAINT 51D minidisk).

2 Create (or modify as required) the Variable Declarations (**:DCL.**) section for the **TCPIPSFS** override area so it resembles the :DCL. section that follows:

```
:OVERLIST. TCPIPSFS
:TCPIPSFS. TCPIP 5VMTCP30
*
*===================================================================*
* Override for TCPIPSFS SFS Component -- File Pool Name Change      *
*===================================================================*
:DCL. UPDATE
 &191    DIR   MYPOOL1:5VMTCP30                  * A-disk
 &LMODZ  DIR   MYPOOL1:5VMTCP30.TCPIP.LOCAL      * Local modifications
 &DELTZ  DIR   MYPOOL1:5VMTCP30.TCPIP.DELTA      * Product service
 &APPLX  DIR   MYPOOL1:5VMTCP30.TCPIP.APPLYALT   * AUX and Inv files (ALT)
 &APPLZ  DIR   MYPOOL1:5VMTCP30.TCPIP.APPLYPROD  * AUX and Inventory file
 &BAS1Z  DIR   MYPOOL1:5VMTCP30.TCPIP.OBJECT     * Base disk
 &BAS2Z  DIR   MYPOOL1:5VMTCP30.TCPIP.SOURCE     * Source code (Optional)
:EDCL.
:END.
```

This override will update the TCPIPSFS override area :DCL. section of the 5VMTCP30 $PPF file.

3 Save your changes when they are complete.

===> file = = *fm*

where *fm* is the file mode where your override file resides (or A if you are creating one).

4 If your $PPF override file was created at file mode A, copy it to file mode D — the Software Inventory minidisk (MAINT 51D).

copyfile *overname* **$ppf** *fm* **= = d (olddate**

5 Compile your changes to create the usable PPF file, ***overname*** **PPF**.

vmfppf *overname* **tcpipsfs**

where *overname* is the file name of your $PPF override file.

Now that the ***overname*** **PPF** file has been created, specify “*overname*” instead of “5VMTCP30” as the PPF name to be used for any VMSES/E commands that require a PPF name.

Appendix H. Copying TCP/IP for z/VM Client Code to the Y-Disk

To simplify access to TCP/IP client functions for your user community, you may find it desirable to copy all, or a subset of, TCP/IP for z/VM client code to the z/VM Product Code minidisk (typically the MAINT 19E minidisk, or the **Y-disk**). Doing so will avoid the need for users to additionally link and access the TCPMAINT 592 minidisk.

As well, applications that use certain programming interfaces may require TCP/IP-specific information to be available for proper operation. For example, information defined in the TCPIP DATA file is referenced by:

- the C run-time library sockets support to correctly identify the TCP/IP virtual machine. See the *C/C++ for z/VM Run-Time Library Reference* (SC09-7624) for more information.
- the VMTCPDT routine, which resides in the VMMTLIB TXTLIB that is associated with the VMLIB Callable Services Library (CSL). See the *z/VM: CMS Callable Services Reference* (SC24-6072) for more information about the VMTCPDT CSL routine.
- various functions provided as part of the CMS REXX Socket library. See the *z/VM: REXX/VM Reference* (SC24-6113) for more information.

To copy TCP/IP for z/VM client files to the Product Code minidisk, use the following procedure **after** you have installed TCP/IP for z/VM.

Warning - Cross-Component File Overlap Considerations

Before you copy *any* TCP/IP for z/VM client files to the Y-disk (or a similar *common use* minidisk), you should first determine whether any conflicts exist between the TCP/IP client files you choose to copy, and those present on the target (Y-disk) minidisk. If any file conflicts are found, these should be addressed and resolved with respect to your installation environment before you continue with the procedure that follows.

Notes:

1. You will need to repeat this procedure each time you apply service to TCP/IP for z/VM.
2. Use discretion when wildcards (*) are used for both the *fn* (file name) and *ft* (file type) parameters of the VMFCOPY commands shown in this section, since files that exist on the Y-disk can be replaced with similarly-named TCP/IP counterpart files. The overlay of certain files may be warranted in some cases, and may be undesirable for others.

An example of this latter case is cited here. Both TCP/IP for z/VM and the Language Environment for z/VM have several **H** files that are identically named, but differ in content. These files are:

```
FCNTL    H    IF       H    IN       H    INET     H
IOCTL    H    NETDB    H    RESOURCE H    SOCKET   H
STRINGS  H    TTYDEV   H    TYPES    H    UIO      H
```

An overlay of Language Environment for z/VM **H** files (those already present on the Y-disk) by their TCP/IP counterparts may create problems when applications are developed or rebuilt that expect (and rely upon) the content of Language Environment for z/VM files.

3. Before copying TCP/IP for z/VM files to another minidisk, ensure adequate storage space is available to maintain the files you have selected.

1 Log on the **MAINT** (or equivalent) user ID.

2 Process TCP/IP for z/VM files used by or available to TCP/IP clients.

link tcpmaint 592 592 rr
access 592 e
access 19e f

Note: If the Y-disk is not defined as the 19E minidisk in your environment, substitute the appropriate device number for this minidisk.

vmfcopy *fn ft* **e = = f2 (olddate replace sprodid 5vmtcp30%tcpip prodid 5vmtcp30%tcpip**

The VMFCOPY command will update the VMSES/E PARTCAT file on the Y-disk.

Wildcards (*) can be substituted for *fn* (file name) and *ft* (file type), but should be used with discretion.

3 (Optional) Erase any TCP/IP for z/VM files that you do not want to remain on the Y-disk — for example, any MAP files that correspond to TCP/IP for z/VM modules re-built during service. Refer to the VMSES/E PARTCAT file on Y-disk to determine which files are associated with TCP/IP for z/VM.

Note: Additional information about managing TCP/IP for z/VM client files, as well as their association with specific TCP/IP functions, is available on-line via the TCP/IP for z/VM home page on the World Wide Web. The URL for this home page is:

```
www.vm.ibm.com/related/tcpip/
```

vmferase file *filename filetype* **f**

See the *z/VM: VMSES/E Introduction and Reference* for more information about the VMFERASE command and options that may help you remove specific files.

4 Re-save the CMS saved system, to return the Y-disk to shared status. See the “Placing (Serviced) Components into Production” section of the *z/VM: Service Guide* for detailed information about how to save the CMS saved system.

Appendix I. Managing TCP/IP Files with Unique Service Requirements

When to Use This Procedure

The steps outlined in this appendix must be completed if message **DTCPRD3061W** is reported by the TCP2PROD command — through its direct use, or as part of the z/VM automated service procedure — when TCP/IP for z/VM *run-time* files are processed.

This appendix provides information to assist you with managing certain TCP/IP files that require some manner of unique processing to fully place those files into production on your system.

Files that warrant such action are the:

- VMRPC TXTLIB
- TCPROFIL EXEC
- TCPROFIL GCS

I.1.1 VMRPC TXTLIB Considerations and Requirements

The **VMRPC TXTLIB** provides Remote Procedure Call (RPC) library functions. Certain z/VM Systems Management Application Programming Interfaces (APIs), provided as part of the CMS component of z/VM, utilize RPC services and do so, in part, through the RPC server program, **DMSVSMAS MODULE**.

Because the DMSVSMAS MODULE is a CMS object, and not part of TCP/IP for z/VM, this module must be rebuilt when the VMRPC TXTLIB is updated by TCP/IP for z/VM service, to ensure that the content of the DMSVSMAS MODULE is correct.

Refer to the appendix titled "Using VMFBLD Outside of the Service Process" in the *z/VM: Service Guide* for specific information about how to rebuild the DMSVSMAS MODULE by processing the CMSALOAD build list.

I.1.2 TCP/IP Server Profile Processing Requirements

Two distinct server profiles are provided with TCP/IP for z/VM:

- the **CMS** server profile (**PROFILE EXEC**), which is common to all TCP/IP *CMS-based* servers — inclusive of VSWITCH controller virtual machines. This file is supplied (and serviced) as the file: TCPROFIL EXEC
- the **GCS** server profile (**PROFILE GCS**), which is common to a small number of TCP/IP *GCS-based* servers. This file is supplied (and serviced) as the file: TCPROFIL GCS

If either of these server profiles is updated by service, the subject file must be copied to the 191 minidisks of the pertinent TCP/IP servers used by your installation. To accomplish this, write access to each such minidisk is necessary; however, this type of access is not possible while the servers are in operation. Thus, each affected server used by your installation must be stopped, with the server profile then copied to the appropriate minidisks, and the servers restarted.

Because write access to the various TCP/IP server 191 minidisks is generally not possible when z/VM service is installed, the z/VM automated service procedure does not attempt to place any updated TCP/IP server profiles into production.

For the rare occasion when processing of this nature is required, the procedure that follows can be used to affect the necessary updates.

I.1.2.1 Copy Server Profile Files Into Production

1 Log on the **MAINT** user ID, or its equivalent.

The MAINT user ID is used in this instance to ensure that sufficient privilege class and authorizations are in effect to use the commands required to complete this procedure.

2 Issue the CMS QUERY DISK command to verify the VMSES/E code and Software Inventory minidisks are correctly linked and accessed.

query disk	Verify the MAINT 5E5 minidisk is accessed as file mode **B**, and is linked **R/O**. Verify the MAINT 51D minidisk is accessed as file mode **D**, and is linked **R/W**.

Note: If another user has the MAINT 51D minidisk linked in write (R/W) mode, you'll obtain only read (R/O) access to this minidisk. If this occurs, have that user re-link the 51D disk in read-only (RR) mode, after which you need issue the appropriate LINK and ACCESS commands for the 51D minidisk. Do not continue with these procedures until a R/W link is established to the 51D minidisk.

3 If necessary, establish appropriate access to the VMSES/E minidisks.

a Establish read access to the VMSES/E code minidisk.

```
link maint 5e5 5e5 rr
access 5e5 b
```

b Establish write access to the Software Inventory minidisk.

```
link maint 51d 51d mr
access 51d d
```

4 Establish access to required TCP/IP minidisks.

vmfsetup servp2p {tcpipp2p | tcpipsfsp2p} (link

Use **tcpipp2p** if the TCP/IP for z/VM default minidisk environment has been maintained; use **tcpipsfsp2p** if the service minidisks were moved to Shared File System directories.

5 Access the TCP/IP for z/VM Server Configuration minidisk.

access 198 *fm*

where *fm* is an available file mode.

The 198 minidisk contains TCP/IP server configuration files (such as the SYSTEM DTCPARMS file).

6 Access the TCP/IP for z/VM Client Code minidisk.

access 592 *fm*

where *fm* is an available file mode.

The 592 minidisk contains TCP/IP client code and related configuration files (such as the TCPIP DATA file).

7 Shutdown the appropriate set of servers, based on the server profile that has been serviced (as reported by message DTCPRD3061W).

a Shutdown the TCP/IP and VSWITCH controller servers (if the TCPROFIL EXEC server profile has been serviced).

Note - Server Shutdown Considerations

Before you shutdown any TCP/IP or VSWITCH controller servers, ensure any applicable conditions or guidelines for your installation have been followed. The shutdown of such servers can impact TCP/IP connectivity for:

- traditional CMS users and applications
- remote users and applications
- virtual machines (including Linux guests) that rely upon CP virtual switch connectivity support.

The TCPMSMGR command is used in the next step to manage the shutdown (and later, the re-initialization) of the TCP/IP protocol *stack* servers and VSWITCH controller virtual machines that are used by your installation.

If other procedures are required by your installation for such operations, use those procedures instead of the TCPMSMGR command.

For additional information about shutting down TCP/IP servers, see the section that discusses “Starting and Stopping TCP/IP Servers” in the chapter titled “General TCP/IP Server Configuration,” of *TCP/IP Planning and Customization*.

For more information about the TCPMSMGR command, and its operands and capabilities, see Appendix A, “TCP/IP Installation, Service and Migration Utilities” on page 104.

Verifying Your Environment

When you perform this step, it is suggested that you first invoke TCPMSMGR as illustrated, but with the **TEST** option also specified. This will help verify that certain command authorization requirements have been met, and that the appropriate set of servers will be affected by TCPMSMGR command operations.

With the **TEST** option in effect, **no servers are shutdown**.

Resolve any reported problems, then invoke TCPMSMGR (without the TEST option) as illustrated.

tcpmsmgr stop stack vswitch

where the **stack** and **vswitch** operands signify that respective shutdown operations are to be performed for TCP/IP protocol *stack* servers and VSWITCH controller virtual machines.

Notes:

1) If servers in either of the listed **stack** or **vswitch** groups need to remain operational at this time, **do not continue with this procedure**, since write access to minidisks associated with any operational severs will not be possible.
2) For most TCP/IP configurations, the shutdown of a TCP/IP protocol *stack* server causes similar actions to be performed for subordinate protocol servers (such as an FTP server). However, there are instances when a subordinate protocol server may need to be stopped through some overt action.

At this time, stop any such servers used by your installation to which such considerations apply.

b Shutdown the GCS-based TCP/IP servers (if the TCPROFIL GCS server profile has been serviced).

cp force *userid*

where *userid* is the user ID of a GCS-based server used by your installation. For example, **SNALNKA** for **X25IPI**, which are the default GCS-based servers supplied with TCP/IP for z/VM. Repeat this command, as needed, to stop all applicable TCP/IP GCS-based servers used by your installation.

8 Detach previously acquired TCP/IP minidisks.

vmfsetup detach

This step is necessary to allow the various TCP/IP server minidisks to be acquired with Read/Write status, in the next step.

9 Re-establish access to required TCP/IP minidisks, in R/W mode.

vmfsetup servp2p {tcpipp2p | tcpipsfsp2p} (link

Use **tcpipp2p** if the TCP/IP for z/VM default minidisk environment has been maintained; use **tcpipsfsp2p** if the service minidisks were moved to Shared File System directories.

10 Copy the serviced TCP/IP server profile into production using the TCP2PROD command. The command cited below processes files that are listed in the TCPSVMCMS or TCPSVMGCS sections of the 5VMTCP30 CATALOG file. See Appendix A, “TCP/IP Installation, Service and Migration Utilities” on page 104 for information about this command and TCP/IP for z/VM catalog files.

Verifying Your Environment

When you perform this step, it is suggested that you first invoke TCP2PROD as illustrated, but with the **TEST** option also specified. This will verify that all resources can be accessed and that the appropriate files will be processed.

With the **TEST** option in effect, **no files are copied into production**.

Resolve any reported problems, then invoke TCP2PROD (without the TEST option) as illustrated.

tcp2prod servp2p {tcpipp2p | tcpipsfsp2p} 5vmtcp30 *ctlg_section*

Use **tcpipp2p** if the TCP/IP for z/VM default minidisk environment has been maintained; use **tcpipsfsp2p** if the service minidisks were moved to Shared File System directories.

If the **TCPROFIL EXEC** file has been serviced, specify *ctlg_section* as **tcpsvmcms**. If the **TCPROFIL GCS** file has been serviced, specify *ctlg_section* as **tcpsvmgcs**.

11 Review the TCP2PROD message log (TCP2PROD $MSGLOG). If necessary, correct any problems before you proceed with the next step.

vmfview tcp2prod

12 Detach previously acquired TCP/IP minidisks.

vmfsetup detach

This step is necessary to allow the various TCP/IP servers to obtain their respective A-disks with Read/Write status, when they are re-initialized in step 14.

13 Acquire the TCP/IP minidisks necessary to run the TCPMSMGR command.

link 5vmtcp30 491 491 rr
link 5vmtcp30 492 492 rr

access 491 *fm1*
access 492 *fm2*

where *fm1* and *fm2* are available file modes.

14 (Re)Initialize TCP/IP and VSWITCH controller servers.

Note - TCP/IP and VSWITCH Controller Startup Considerations

Before you initialize any TCP/IP or VSWITCH controller servers, ensure any applicable conditions or guidelines for your installation have been followed. The shutdown of such servers can impact TCP/IP connectivity for:

- traditional CMS users and applications
- remote users and applications
- virtual machines (including Linux guests) that rely upon CP virtual switch connectivity support.

The TCPMSMGR command is used in the next step to manage the (re)initialization of the TCP/IP protocol *stack* servers and VSWITCH controller virtual machines that are used by your installation.

If other procedures are required by your installation for such operations, use those procedures instead of the TCPMSMGR command.

For additional information about starting up TCP/IP servers, see the section that discusses “Starting and Stopping TCP/IP Servers” in the chapter titled “General TCP/IP Server Configuration,” of *TCP/IP Planning and Customization*.

For more information about the TCPMSMGR command, and its operands and capabilities, see Appendix A, “TCP/IP Installation, Service and Migration Utilities” on page 104.

Verifying Your Environment

When you perform this step, it is suggested that you first invoke TCPMSMGR as illustrated, but with the **TEST** option also specified. This will help verify that certain command authorization requirements have been met, and that the appropriate set of servers will be affected by TCPMSMGR command operations.

With the **TEST** option in effect, **no servers are initialized**.

Resolve any reported problems, then invoke TCPMSMGR (without the TEST option) as illustrated.

tcpmsmgr start stack vswitch

where the **stack** and **vswitch** operands signify that respective startup operations are to be performed for TCP/IP protocol *stack* servers and VSWITCH controller virtual machines.

Note: For most TCP/IP configurations, the initialization of a TCP/IP protocol *stack* server causes similar actions to be performed for subordinate protocol servers (such as an FTP server). However, there are instances when a subordinate protocol server may need to be started through some overt action.

At this time, start any such servers used by your installation to which such considerations apply.

15 Log off the **MAINT** user ID once server initialization operations are complete.

Notices

This information was developed for products and services offered in the U.S.A. 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 give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106-0032, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: 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 states 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 to 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 Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply 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 Corporation
TCP/IP for VM Development
Dept. G79G
1701 North Street
Endicott, NY 13760
```

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.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurement may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

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 on non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information contains 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 the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrates 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 IBM programming interfaces. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

Trademarks and Service Marks

The following terms are trademarks of International Business Machines Corporation in the United States, other countries, or both:

BookManager
HiperSockets
IBM
IBMLink
Language Environment
RACF
RETAIN
VM/ESA
z/VM

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Other company, product, and service names may be trademarks or service marks of others.

Reader's Comments

TCP/IP for z/VM

You may use this form to comment about this document, its organization, or subject matter. Please understand that your feedback is of importance to IBM, but IBM makes no promises to always provide a response to your feedback. If you prefer to provide feedback electronically, please e-mail your comments to: **vmtcpip@vnet.ibm.com**

For each of the topics below please indicate your satisfaction level by circling your choice from the rating scale. If a statement does not apply, please circle N.

RATING SCALE

very satisfied	◄------------------------------►			**very dissatisfied**	**not applicable**
1	**2**	**3**	**4**	**5**	**N**

	Satisfaction					
Ease of product installation	1	2	3	4	5	N
Time required to install the product	1	2	3	4	5	N
Contents of program directory	1	2	3	4	5	N
Readability and organization of program directory tasks	1	2	3	4	5	N
Necessity of all installation tasks	1	2	3	4	5	N
Accuracy of the definition of the installation tasks	1	2	3	4	5	N
Technical level of the installation tasks	1	2	3	4	5	N
Installation verification procedure	1	2	3	4	5	N
Ease of customizing the product	1	2	3	4	5	N
Ease of migrating the product from a previous release	1	2	3	4	5	N
Ease of putting the system into production after installation	1	2	3	4	5	N
Ease of installing service	1	2	3	4	5	N

- Did you order this product as an independent product or as part of a package?

  □ Independent
  □ Package

  What type of package was ordered?

  □ CustomPac
  □ System Delivery Offering (SDO)
  □ Other - Please specify type: ____

- Is this the first time your organization has installed this product?

  □ Yes
  □ No

- Were the people who did the installation experienced with the installation of VM products using VMSES/E?

  □ Yes

  How many years of experience do they have? ____________________

  □ No

- How long did it take to install this product? ______________________________

- If you have any comments to make about your ratings above, or any other aspect of the product installation, please list them below:

  ____

  ____

  ____

  ____

  ____

  ____

  ____

  ____

Please provide the following contact information:

____
Name and Job Title

____
Organization

____

____
Address

____
Telephone

Thank you for your participation.

Please send the completed form to the following address, or give to your IBM representative who will forward it to the TCP/IP for z/VM Development group:

IBM Corporation
TCP/IP for VM Development
Dept. G79G
1701 North Street
Endicott, NY 13760

Program Number: 5741-A05

Printed in U.S.A.

GI10-0784-00