VM/ESA TCP/IP Performance

SHARE 89 - Summer Meeting 1997
Session 9224

Bill Bitner
IBM Corp.
1701 North St.
Endicott, NY 13760
(607) 752-6022
bitner@vnet.ibm.com

Romney White
IBM Corp.
1701 North St.
Endicott, NY 13760
(607) 755-8276
romney@vnet.ibm.com

Disclaimer

The information contained in this document has not been submitted to any formal IBM test and is distributed on an "as is" basis without any warranty either express or implied. The use of this information or the implementation of any of these techniques is a customer responsibility and depends on the customer's ability to evaluate and integrate them into the operational environment. While each item may have been reviewed by IBM for accuracy in a specific situation, there is no guarantee that the same or similar results will be obtained elsewhere. Customers attempting to adapt these techniques to their own environment do so at their own risk.

In this document, any references made to an IBM licensed program are not intended to state or imply that only IBM's licensed program may be used; any functionally equivalent program may be used instead.

Any performance data contained in this document was determined in a controlled environment and, therefore, the results which may be obtained in other operating environments may vary significantly.

Users of this document should verify the applicable data for their specific environments. It is possible that this material may contain references to, or information about, IBM products (machines and programs), programming, or services that are not announced in your country or not yet announced by IBM. Such references or information should not be construed to mean that IBM intends to announce such IBM products, programming, or services.

Should the speaker start getting too silly, IBM will deny any knowledge of his association with the corporation.

Permission is hereby granted to SHARE to publish an exact copy of this paper in the SHARE proceedings. IBM retains the title to the copyright in this paper, as well as the copyright in all underlying works. IBM retains the right to make derivative works and to republish and distribute this paper to whomever it chooses in any way it chooses.

Trademarks

- The following are trademarks of the IBM Corporation:
  - VM/ESA
  - MVS/ESA
  - OS/2
  - VTAM

I'd also like to take this chance to thank everyone who gave me input on this presentation and taught me much along the way. In particular, Wes Ernsberger, Romney White, Angelo Macchiano, Keith Jones, Maryrita Steinhour, Joe Hust, Alan Altmark, John Thornton, and Mark Cibula, and anyone else I missed. I look forward to working more with them and to the continued improvement of TCP/IP.

Introduction

- This presentation has been humbling.
- VM TCP/IP Performance has been neglected for many years.
- That is Changing!
  - Bad news: small team working on this.
  - Good news: small team working on this.
- First challenge: What does TCP/IP Performance mean?

Putting this presentation together and presenting it is a humbling experience. This presentation just scratches the surface of VM TCP/IP performance. I have much to learn. Not to make excuses, but VM TCP/IP performance has been neglected for a number of years. Let me apologize, and let you know that we hope to change that. The bad news is that we only have the resources to have a few people work on improving the performance. The good news is they are good people and work well together.

The first challenge is one of definition. You may have called or sent me a note at one time or another that said you had a performance problem. My immediate response was most likely "What do you mean by performance?". Well now, I find I have to add another qualifier when looking at TCP/IP performance. Does TCP/IP mean the stack? web serving? FTP? TELNET? SMTP? I've had questions in all those areas so far this year.

Sources of Information

- IBM TCP/IP Performance Tuning Guide, SC31-7188
  - Also contains OS/2, DOS, and VM information.
  - Concepts, Tuning Information, and benchmark data.
- VM TCP/IP Home Page
  - http://www.vm.ibm.com/related/tcpip/
- IBM TCP/IP List
  - IBMTCP-L@VM.MARIST.EDU
  - Newsgroup: bit.listserv.ibmtcp-l

There is little VM TCP/IP performance information in the VM libraries, but there is a manual which does cover a lot of good performance related information. That is the IBM TCP/IP Performance Tuning Guide (SC31-7188). The PTG contains good background material on TCP/IP concepts that relate to performance and then show application of them to MVS, VM, OS/2, and other environments.

When looking for TCP/IP performance information, why not check out the internet. The VM TCP/IP home page URL is given in the foil. This presentation will be made available there and updated as it is improved. In addition, performance related questions can be asked on various lists on the net.

Monitoring Capabilities

- Regular Monitor Data
  - Note seldom-ending channel program can skew state sampling for TCP/IP machine.
- NETSTAT
  - snapshot of settings and some resource usage
  - not very efficient

Since various parts of TCP/IP run in server virtual machines on VM, the regular monitor data associated with them can be explored. This can give a good indication of resources used or in demand. One area to watch out for it the traditional user state sampling for the TCPIP machine can be skewed because of the use of seldom-ending channel programs. This results in a large value for waiting for active I/O.

The NETSTAT command is included as part of VM TCP/IP and can be useful for getting a snapshot of resources in use and various settings. However, it is not a very efficient interface, therefore not appropriate for long term trending or history data.

Monitoring Capabilities *(continued)*

- SNMP (Simple Network Management Protocol)
  - helps manage internet elements including performance data.
  - used by other applications to present the data
- TCP/IP server machine use of APPLDATA
  - work-in-progress
  - development instrumentation

SNMP (Simple Network Management Protocol) is the architected keep-them-out-of-trouble protocol. It allows for the collection of different types of data including (in some implementations) hardware collected instrumentation. Other applications/products, such as NETVIEW, can be used to retrieve and present the information. All this processing does require resources.

In addition, we are working on having the TCPIP SVM provide performance information through the monitor APPLDATA domain. In short-term, this is for us to better understand where the overhead is and how to lower it. However, we recognize the value of making this generally available.

TCP/IP 2.4.0

- Regression tests compared to 2.3.0 showed equivalence
  - DCE RPC test bed for UDP and IP testing
  - FTP test bed for TCP and IP testing
- Customer testing validated equivalence.
  - Used CP I/O trace to compute bytes moved
  - Performance data for CPU used
  - Equivalent CPU per byte
  - Beware of CPU per I/O comparisons.

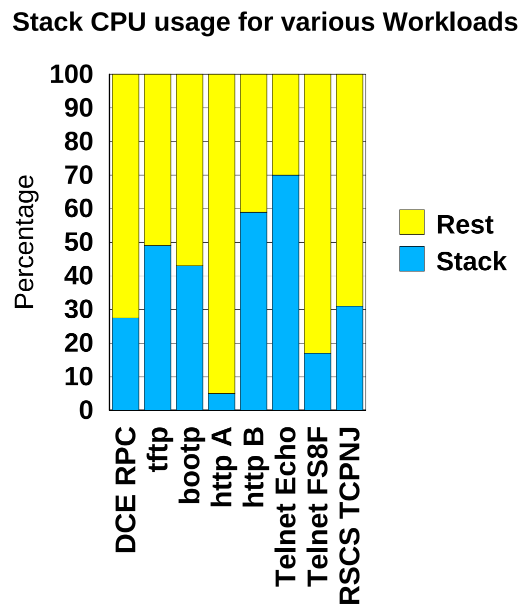
For VM TCP/IP 2.4.0, a set of measurements were made to compare to the 2.3.0. These measurements included workloads that exercised both the UDP and TCP layers. In both cases, equivalent performance was measured. We also validated this in a customer environment. The customer (Mark Wheeler) used a set of CP I/O traces to get detailed metrics such as number of bytes moved over each hardware connection. Resource usage was then normalized on a per KB basis.

Be careful when making comparisons with other changes. For example, a metric of CPU per I/O would be misleading if there was more, or less, data moved with each I/O as a result of more virtual storage being available or changes in buffer settings.

Stack Comments

- Limited by single processor speeds
- Two flavors:
  - UDP
  - TCP

Note: Telnet case includes Telnet running in the TCPIP machine with the stack.



The TCP/IP stack performance can be a limiting factor. The TCPIP virtual machine is not able to run as a virtual MP, so it is limited by the capacity of a single processor. The stack has two basic types of data flows: UDP (User Datagram Protocol) and TCP (Transmission Control Protocol). UDP does not have the overhead associated with managing a connection-driven environment (like TCP or SNA). However, some applications provide the connection management in themselves (so the overhead is still there, just not in the stack); for example DCE RPC. Also note that Telnet is different in that the Telnet code also runs in the TCPIP server machine along with the stack.

The chart shows how big a part the stack plays in various workloads. It shows the percentage of host resources used by the stack. Remember the single processor limitation. Based on that, this DCE RPC workload would be hard pressed to run on anything above a 4-way. Note, that this is worst case, basically a null RPC.

Telnet Comments

- TCP/IP 2.4.0 Announcement Confusing
  - Lifted the Telnet session limit
  - CP still has limit of 4096 logical devices per system
- What if we ignore the CP limit? Projections:
  - Old data: between 4800 to 7400 on 3090-200J
  - New data: between 5000 to 6000 on 9672-R42
- TCP/IP storage requirements
  - do not grossly over-estimate buffers

It appears the announcement material for TCP/IP 2.4.0 added some confusion. While the TELNET session limit of 2000 was removed, and TELNET is now bounded by virtual storage; TELNET uses logical device support and there is a CP limit of 4096 logical devices per system.

We know we need to remove the CP limit, so watch this space. The question has come up, okay what would the limit be. Again, we do not have current data for that, but based on two old studies, projections were made. Unfortunately, the studies show different results, so we end up with a wide range of between 4800 and 7400 users on a 3090-200J. Remember the single processor limit. The Telnet environment is one that is limited by the capacity of a single processor. These old measurements were basically echoes, so TCPIP is a significant percentage of the system resources. We now have some new measurements with our CMS interactive workload. In this environment, TCPIP is a smaller percentage of the total CPU usage, so we would not be limited by single processor speed until around a 4 to 5 way processor. As the number of users gets into the mid to high thousands of users, it is likely that the network will consist of multiple connections. It would be possible to run multiple TCP/IPs and connect them via virtual CTCs. This would increase the capacity even more.

Web Comments

- As Web servers become more efficient, improving the stack will be critical.
- Some of the challenges of HTTP
  - Connection requirements and startup
  - Round Trip Time is large factor in response time
  - Scaling

Web or HTTP workloads can vary a great deal. Work to date shows that as the Web servers become more efficient (as they have and will continue to), the stack will need to be improved to keep pace. Besides the wide scope of what makes up a typical page being served out, there are some other challenges in the HTTP world in general. The cost of starting up connections and window size management is unfortunately a large part of the HTTP workload. With this effect, the round trip time becomes a large factor in response time. Seldom is it run to the store and get everything you need.

SMTP Comments

- Currently dominated by I/O performance
- Measurements made to better understand:
  - 9021-720 3390 without DASD Fast Write
  - 4 notes/second/SMTP server
  - each note had 3 recipients outbound
- We are committed to improving this.
- We'd like your input to set objectives.

When looking at which application of TCP/IP needed performance attention, it was clear that SMTP was high on the list. Unlike other areas which tend to be more CPU involved and dependent on the stack, the current SMTP is bounded by I/O. This I/O includes not just moving the data, but lots of I/O to keep track of what has been moved. In that past people got around this by replicating the SMTP servers, using I/O caching, or modifying SMTP for special purposes. We are committed to improving this. So watch this space.

Tuning Guidelines

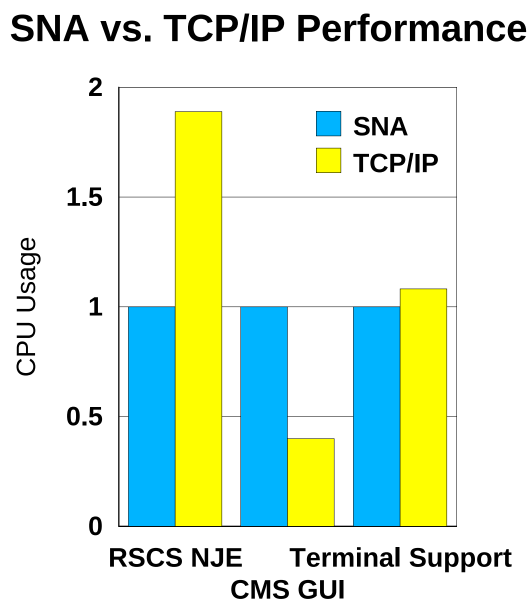
- Typical SVM Tuning
  - QUICKDSP ON
  - Increase Share value
  - Reserve pages if necessary
- Match to use/workload
  - FTP likes bigger buffers
  - TELNET needs more smaller buffers
- Do not forget the other end of the communication if you control it.
- DASD Fast Write for current SMTP

Remember how I mentioned this was a humbling presentation? Well this is the most humbling page of the presentation. All I really have to offer at this time are the traditional tuning we do for all SVMs. Now that would not be bad if that was the end of the story because after all it would be nice to ship a product that does not require a lot of tuning. However, there appears to be a huge number of knobs and levers involved here and it is not clear to me which are attached to what (or if they are even attached).

What I have learned so far, is that with TCP/IP it depends. While larger buffers may help an FTP environment where lots of data is being pushed through the stack, it will not significantly help TELNET environments.

Comparison with SNA

- SNA wins in RSCS line driver scenario
- TCP/IP wins in CMS GUI environment
  - Uses UDP
  - SNA not efficient in this connectionless case
- SNA has the edge in terminal support for CPU usage (response time equal).



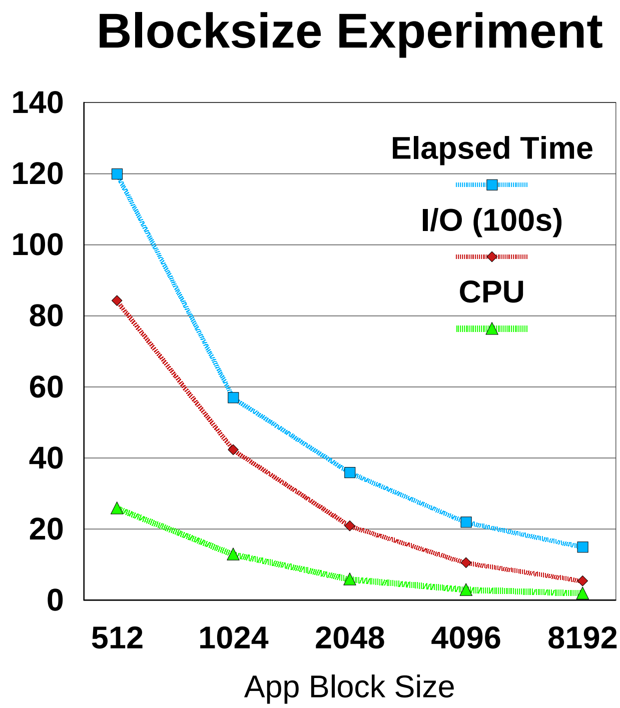
We love comparisons. I have been asked a couple of times about how TCP/IP compares to SNA. I use to reply "I don't know". Now, I stall a bit by asking "what do you mean by TCP/IP?"

I did find some data that allowed rough comparisons to be made. The first was when we did some RSCS testing a while ago where we measured over SNANJE and TCPNJE lines for a RSCS workload of ours. In this case, SNA was the better performer. In the testing of the CMSDESK VM GUI application, we found that better performance was achieved when configured with TCP/IP than in an SNA APPC configuration. This second case showed the use of UDP interface, which avoids some connection overhead. SNA could not shed that part of the processing.

For the TELNET vs. VSCS/VTAM, there are various comparisons we could make. The one shown was measured with TPNS over CTCAs for both with the regular CMS interactive workload. Response time was good in all cases, but SNA required a bit less processor resources.

TFTP with Various Blocksizes

- Early measurements for the Network Station
- Download 2M kernel from 9121-320 with 3172 to 16 Mbit TR
- MTU size = 2000
- Blocksize important even when greater than MTU size



Look at this chart not for details on the network station, but for what we illustrate by using it. (The network station is pretty cool by the way).

In this experiment, there was a net station attached by a 16Mbit IBM token ring to a 9121-320 via a 3172 to a 16 Mbit IBM Token Ring. The TFTP server on VM was used to download a 2 meg kernel file to the net station. The blocksize used for the transfer was varied over time.

You can see the dramatic effect that the blocksize has in the performance seen here. Now in the Net Station scenario, we are using UDP, but acknowledgments are done handled by the TFTP and Net Station client. The number of requests made to move the 2 meg file is directly proportional to the blocksize, and this leads to the number of roundtrips required for handling the download.

The MTU size in this case was 2000 bytes, but even when the blocksize exceeded the MTU size, the steady improvement continued.

Potential Future: Instrumentation

- Use of Appldata domain
- Machines most likely to see it first:
  - TCP/IP Stack Machine
  - SMTP Machine
  - TFTP Machine
- Changes to CP to accommodate the above
  - Appldata event record (supplied by application)
  - Appldata configuration record (in Appldata domain)

Remember Future is funny word, do not take this to be a commitment on IBM that we are definitely doing this or that it will be released.

The improvement, or addition, of performance instrumentation for TCP/IP has been a requirement for a while. We plan on taking some steps to improve this in the short and long term. The majority of this will, at least initially, be done via the appldata interface of the CP monitor system service. This allows an application in a virtual machine to contribute data to the CP monitor data stream in a very efficient manner.
The machines that will be instrumented first include the TCP/IP stack machine, the SMTP machine, and the TFTP machine. We will discuss each briefly in foils that follow.
We also hope to make some changes to CP in order to improve the type of data that can be generated. Currently, the appldata is used to generate sampled data, not event or config. Though, you can mimic event data by starting and stopping sampled appldata. Changes in the works would allow for true event records and configuration records.

Future Stack Performance

- Looking at ways to lower the costs of TCP/IP
- Some items higher on our list:
  - Exploit the CHECKSUM hardware facility
  - RFC 1323 TCP Extensions for high performance
    - aka Long, Fat Network
    - allow more data in the pipe that has not already been acknowledged
    - based on increased network reliability

As discussed earlier, the stack can be an important part of TCP/IP performance and it will become increasingly important over time. That is why we are looking to improve it now. We are still collecting data to better understand where to make the improvements, but we have identified some candidates.

Exploiting the CHECKSUM hardware facility is a nice example of where we want to improve our synergy with the hardware. We haven't quantified the overall impact to TCP/IP, but preliminary tests show there is considerable savings in the checksum processing.

RFC 1323 is another item we are investigating. At one time, there was the concern of managing the amount of data that was "in the pipe" at any one time because of network failures and the need to resend this data. With increases in network speed and reliability, this is not as great a concern. This RFC describes an extension to TCP that would increase the amount of data pushed into the pipe at one time, and therefore improve network throughput in stable conditions.

Future SMTP Performance

- Improvement via incremental improvements mostly in area of synchronous minidisk I/O
- Will better utilize virtual memory and system spool space to accomplish
- In particular:
  - reduce minidisk I/O for Log and Stat files.
  - hardened copy of note in spool, working in virtual storage, only write to minidisk in retry case
  - use of asynchronous spool interface

The current SMTP performance is greatly bounded by the large amount of synchronous I/O that it does. While other approaches were investigated, the team has chosen making several incremental changes to greatly reduce the I/O requirements. The more effective use of system spool and virtual storage will allow this to happen.
In particular, the minidisk I/O for the Log and Stat files will be greatly reduced. The Stat file I/O will be nearly eliminated, while the Log file I/O will be reduced. The notes themselves will only be written to minidisk in the case of 'a retry' being necessary. We are also investigating the use of asynchronous spool to further improve performance or capacity.

Summary

- We have a lot to learn about TCP/IP performance.
- We are committed to improving:
  - the performance
  - the instrumentation
  - the tuning and guidelines
- Please help us set our priorities.

Whew, this presentation is almost over. What? You thought it was over after the Introduction foil? Not funny. The only story I like presenting more than one with great performance is one where performance is improving. And in this case, by performance I mean how fast it is, how well it is instrumented, and how easy is it to tune. If there is a particular area you want us to stress, we want to hear about it. The VM TCP/IP Home Page will have a mailto item on it for feedback.

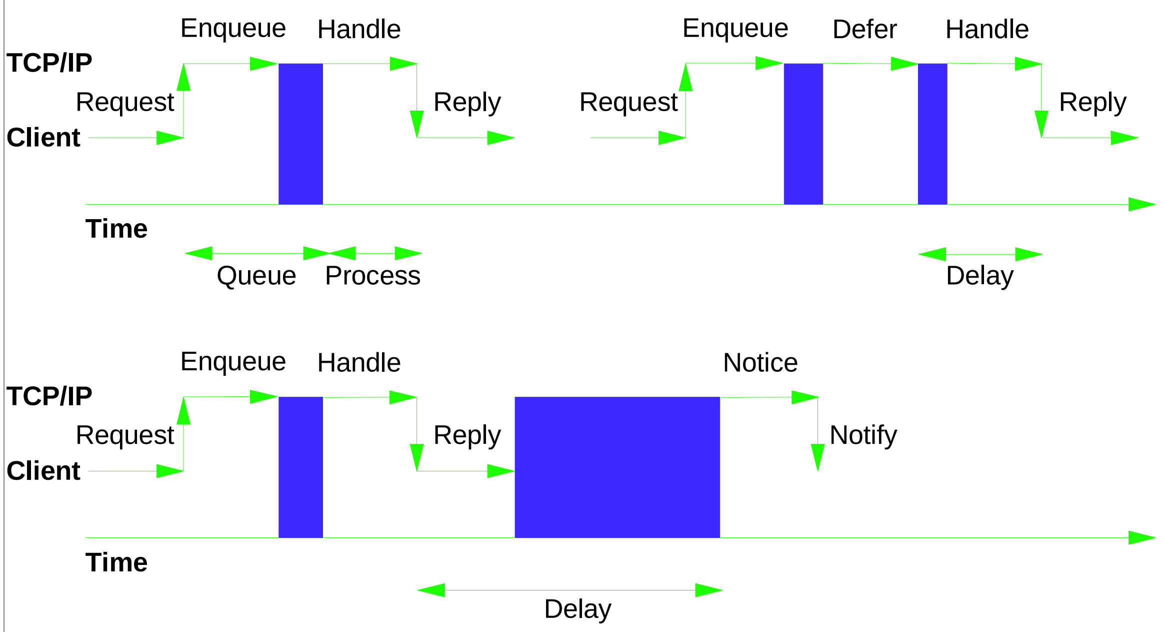
I look forward to coming back and to this presentation growing. And I'm counting on you to keep me humble.

Thanks.

Questions and Comments

- Now is your time to let us know:
  - If you think we are heading in the right direction
  - what questions you have about TCP/IP performance that you need instrumentation data to answer
  - what objectives we should set in terms of how fast TCP/IP on VM needs to be to meet your business objectives.
- The floor is yours.

TCP/IP Stack Instrumentation
Client Interaction Models



The diagram above shows models three potential interactions of the TCP/IP stack. As far as the stack is concerned servers could be clients and clients could be servers. So for this discussion, consider everything that communicates with TCP/IP to be a client. The first model shows the case where a request is presented to the stack, is queued briefly, processed, and then the stack replies. The second model adds another component of defer time, perhaps where the stack needs to do I/O or is actually waiting to handle a receive request for the client where nothing has been received yet. The third model is similar to the first, except that so time after the reply to the client, the stack receives some sort of notice from the client's other end (perhaps over the network) and then turns around and presents that to the client. This could give an indication of network delay.

TCP/IP Stack Instrumentation Buffer Pool Configuration

```
          |---------ACB---------| |---------CCB---------|
Time      Count Limit  Warn  Size Count Limit  Warn  Size
-------- ----- ----- ----- ----- ----- ----- ----- -----
21:51:01   800    40    80 75.0K   150     0    10 31.1K


          ... |---------UCB---------|    Total
              Count Limit  Warn  Size  Storage
              ----- ----- ----- ----- --------
                100     0     6 23.9K    11.7M
```

This and several of the following reports are examples of data generated from the prototype stack instrumentation we have running. The reports are simple ones generated with REXX programs looking just at this new data. Eventually, we hope to have formal reports. These are for illustration purposes.
This first report shows how the buffer pools are configured for TCP/IP. The "count" is number of defined, "limit" is the number we hold for emergencies, "warn" is the number at which warnings are issued, and the "size" is the total amount of bytes allocated.

TCP/IP Stack Instrumentation
Buffer Pool Activity

```
                                   |---Data--| |--Small--|
          |---ACB---| |---CCB---| |--Buffer-| |-Envelope|
Time       Size   Min  Size   Min  Size   Min  Size   Min
-------- ----- ----- ----- ----- ----- ----- ----- -----
21:52:39   798   792   140   140   198   198   749   747
21:55:40   798   792   140   140   198   198   749   747
21:58:39   797   792   139   139   197   197   749   747
22:01:39   796   791   138   138   196   194   749   747
22:04:34   796   789   138   138   196   194   749   747
22:07:38   795   789   138   138   195   194   749   747
22:10:36   796   789   138   138   195   192   749   747
22:13:39   794   788   138   138   194   190   749   728
...
-------- ----- ----- ----- ----- ----- ----- ----- -----
Summary:   787   749   138   138   183   132   747   671
```

This report in combination with the previous report can be used to determine if the appropriate number of buffer pools have been defined.

TCP/IP Stack Instrumentation Link Configuration

```
         |-------------Device--------------|
Time     Name             Type     Addresses
-------- ---------------- -------- ---------
21:50:57 LCS2             LCS      0D42-0D43
         LCS3             LCS      0722-0723
         LCS4             LCS      0200-0201


                                                    Max
         |----Link---|                            Trans
         Name      No. Network Type  Description   Unit Speed
         -------- ---- ------------- ------------ ----- -----
         ETH2        1 Ethernet      IBM LCS       1500  10Mb
         ETH3        1 Ethernet      IBM LCS       1500  10Mb
         FDDI2       1 FDDI          IBM LCS      32768 100Mb
```

This report provides configuration information on the various links. Note that the "speed" field is not a measured speed, but the typical speed associated with the link type.

TCP/IP Stack Instrumentation Link Activity

```
         |------------Link------------|
                                  Status
Time     Name                No.  Change
-------- ---------------- ---- --------
21:55:40 ETH2                1 21:51:06
21:58:39

         |----------------Incoming per second----------------|
                    Ucast   NUcast                      Unknown
           Octets  Packets  Packets Discards   Errors   Protos
         -------- -------- -------- -------- -------- --------
           437781      298     0.02     0.00     0.00     0.00
           324154      218     0.02     0.00     0.00     0.00

         |-----------Outgoing per second------------|
                    Ucast   NUcast
           Octets  Packets  Packets Discards   Errors
         -------- -------- -------- -------- --------
            18060    66.96     0.00     0.00     0.00
             2822    45.52     0.01     0.00     0.02
```

Along with the link configuration, we can also tell how much data is being moved across each link. Octets is TCP/IP for bytes. Indications of problems can also be seen in high counts for Errors or Unknown protos.

TCP/IP Stack Instrumentation CPU Activity

	Percent			Time (seconds)		
Time	Virt CPU	Wait	Other	Virtual CPU	Wait	Other
21:55:40	0.0	100.0	0.0	0.0	180.2	0.0
21:58:39	0.0	100.0	0.0	0.0	179.9	0.0
...						
05:46:39	39.4	33.5	27.0	73.0	62.1	50.0
05:49:34	34.3	43.8	22.0	60.0	76.6	38.5
05:52:39	37.1	39.5	23.4	68.5	73.1	43.2
05:55:35	33.4	47.8	18.8	58.5	83.9	32.9
05:58:40	36.5	43.1	20.4	67.4	79.8	37.8
Summary:	23.3	62.3	14.4	6780.8	18179.0	4200.4

By instrumenting the stack itself, we can also get a measure of the utilization of the server.

TCP/IP Stack Instrumentation Management Information Base

- Internet Protocol
- Internet Control Message Protocol
  - PING
- Transmission Control Protocol
  - FTP, SMTP, Telnet
- User Datagram Protocol
  - TFTP, BOOTP
- Address Resolution Protocol
- Input/Output
- Inter-UserCommunication Vehicle
  - Sockets
- Virtual Machine Communication Facility
  - TCP/UDP/IP APIs

The standard information available through SNMP will also be available through the stack instrumentation. This includes the MIB or Management Information Base.

TCP/IP Stack Instrumentation Scheduler Data

```
         |--------------------Queue sizes--------------------|
         |---Right Now---| |----Priority---| |-----Normal----|
Time         Size  Maximum     Size  Maximum     Size  Maximum
-------- -------- -------- -------- -------- -------- --------
21:55:40        0        3        0        0        0        4


21:58:39        0        3        0        0        0        4



|----Times per second----| |----------Process Activity-------- ...
  Queues  Process Moved to              Con- Internal   Socket
   Empty    Dying    Timer    Total sistency   Client  Request
-------- -------- -------- -------- -------- -------- --------
    0.17                       0.24     0.02     0.01       98
                                215       65      287      106
                                138       66      287      347
    0.22                       0.27     0.02     0.01      104
                                361      515      265      113
                                214      448      266      377
```

The TCP/IP Stack Scheduler Data is detailed information that is of more interest for ourselves in making design decisions.

TCP/IP Stack Instrumentation
Client Activity

Start Time	Client	Requests Received (rate per second)	Replies Sent (rate per second)	Receives Delayed (rate per second)	Notices Sent (rate per second)
23:17:18	CLIENT1	7.0	0.9	6.1	6.3
23:17:32	CLIENT2	9.5	0.4	9.1	9.2
23:18:18	CLIENT3	7.0	0.9	6.1	6.3

Process (per request, microseconds)	Queue (per request, microseconds)	Network Delay (ms)	Receive Delay (ms)	Busy (Percent)	Request Pending (Percent)	Time Active
135.4	169.7	15.4	15.6	0.22	9.63	0:13.904
38.9	207.1	9.9	3.6	0.23	9.14	0:41.853
130.1	147.9	12.5	14.5	0.19	7.87	0:14.622

The Client Activity data can help determine if performance problems are system-wide or related to a particular client. And also if they are related to network delays.

TCP/IP Stack Instrumentation
TCP Connection Activity

Start Time	Client	Application	Foreign IP Address	Foreign Port	Local Port
17:47:00	INTCLIEN	Telnet 0004	10.0.0.2	1027	23
17:47:12	CLIENT3	FTP	10.0.0.1	21	1024
17:47:31		FTP Transfer	10.0.0.1	20	1025

Max Seg Size	Maximum Send Window	Receive Window Limits Minimum	Receive Window Limits Maximum	Bytes per second Sent	Bytes per second Received
1452	28.0K	1024	2048	869.2	11.7
1452	28.0K	30.6K	32.0K	5.1	16.9
1452	28.0K	30.6K	32.0K	225.6K	0.8

Maximum Segments UnACKed	ACKs Received /sec	Mean ACK Time	Round Trip Time (ms) Mean	Round Trip Time (ms) Var	Effectiveness	Session Duration
2	1.38	142	135	78	5.1	00:01:14
1	0.37	202	218	95	13.5	0:35.322
15	160.35	49	52	13	0.1	0:01.316

This report has a wealth of information on the various client connection and the performance seen by each. We can get some idea of who we are communicating, the data rate, round trip time, window size, and session duration. "Effectiveness" is a computed number to give a relative idea of performance; the smaller the number, the better.

TCP/IP Stack Instrumentation
Hash Tree Use

```
          |------IP-------| |------TCP------|
          |----Routing----| |--Connections--|
Time       Free  Used   Min  Free  Used   Min
-------- ----- ----- ----- ----- ----- -----
15:15:25   296     4   296   253     3   253
...
-------- ----- ----- ----- ----- ----- -----
Summary:   296     4   296   253     3   251

          |------UDP------| |----Address----|
          |-----Ports-----| |--Translation--|
           Free  Used   Min  Free  Used   Min
          ----- ----- ----- ----- ----- -----
            100     0   100  1497     3  1497
          ...
          ----- ----- ----- ----- ----- -----
            100     0   100  1497     3  1497
```

This last report is low level information, but could be important. The stack uses a series of hash tables of a fixed size. If the stack runs out of entries in any of these tables, things stop working. That would be bad.