NETWORK PERFORMANCE & AVAILABILITY REPORTING: SOMEONE HAS TO START IT (PAPER #5141)

By Cathy Liu, AES, and Dr. Leo Lo, Shenlo, 2005.

ABSTRACT

The measurement of network service levels has long been a neglected practice by the performance industry. This is most likely due to a lack of network skills training and standard measurement and reporting definitions, as well as the fast pace of emerging technologies. There are, however, some TCP/IP utilities that present a practical opportunity to measure network service level goals proactively. With some development effort, nominal coding and collaboration with other network tools, these TCP/IP utilities are very useful. This paper shows how such utilities may be used to gather network service level statistics. Sample reports are also provided.

I. TRADITIONAL NETWORK SERVICE LEVEL REPORTING

In legacy host-based networking environments such as those utilizing SNA, network service levels are concerned with response times and availability. Response time is defined as the elapsed time between when a remote user enters a command or transaction and when that same user is able to continue with a subsequent request. The measurable unit can be expressed in seconds per transaction over a period of time, normally hours or days. Availability is defined as the uptime of a network expressed as a percentage over a period of time, normally in days, weeks or months.

Response time reporting for host-based applications has been facilitated by using monitors such as NPM and NetSpy residing on the host system with interfaces to the operating system. The monitors use a round-trip time stamp of a “definite response (DR)” command to calculate the application response time. The response time is broken into host processing time and rest time components, called network time. This works well in a simple response-request sequential PU-LU environment.

When peer-to-peer client-server computing was introduced in the 1980’s, the so-called transaction response time became more complex because one physical transaction (or session) could contain nested logical sessions (e.g., APPC in an LU6.2 environment). Most monitors could not address the response time for the logical sessions. Because of this, LU6.2-based transaction response times were never properly reported. When the TCP/IP protocol was introduced it further complicated network service level reporting, especially for response times, since TCP/IP-based transactions could have different characteristics, making a generalized end-to-end response time hard to obtain. Specific applications may have to use embedded code within the application to collect the end-to-end response time. For enterprise-wide network service level reporting, there is no one tool that can realistically report true end-to-end user response times. It behooves the network administrator to begin exploring network response time for all targets and use Service Level Reporting proactively.

II. WHAT TOOLS OR UTILITIES ARE AVAILABLE?

In current networking environments, the Transmission Control Protocol (TCP) provides a connection-oriented, reliable, flow-controlled, end-to-end communications service between sessions. TCP allows two connected end-points to exchange data simultaneously in a full-duplex mode through a set of interfaces (e.g., SYN, ACK, data, FIN). The Internet Protocol (IP) defines the format of the packets sent across the network. The IPv4 protocol uses a 32-bit address scheme to define the uniqueness of a network (or the nodes) via its prefix bits, and, the number of hosts within the network via its suffix bits. The data unit transported by IP is called a ***datagram*** and the data unit exchanged at the TCP level is called a ***segment***.

SNMP MIBs

What utilities are available to collect network-related data for service level reporting? The Simple Network Management Protocol (SNMP) provides the base of performance data for the network devices (e.g., routers, switches, hubs) through SNMP's manager-agent architecture. A manager component can run on a network device (e.g., a server) and sends requests to managed devices where the agent component resides. The agent sends back the requested information to the manager which maintains a database to keep the historical network data.

SNMP exchanges network information through messages known as PDUs (protocol data units). From a high-level perspective, the PDU is seen as an object containing variables consisting of both titles and values. SNMP uses five types of PDUs to monitor a network: two of them deal with setting device data, two of them deal with reading device data, and one, the trap, is used for monitoring network events such as start-ups and shut-downs. The collected information is logically organized into a tree structure known as the Management Information Base (MIB or MIB-II). Each branch contains a specific type of information for easy retrieval. There are public MIBs and private (vendor specified) MIBs. Most vendors support the public MIB formats and interfaces.

The MIB for remotely connected LAN devices utilizes the Remote Monitoring MIB (RMON) facilities for collecting specific information at the LAN level. For Ethernet-based LANs for example, information such as packets per second, number of data collisions per second, device utilization, and the number of Cyclic Redundant Code (CRC) errors can be collected. There are many popular RMON-based network monitors such as IBM's Tivoli TME, Hewlett Packard's OpenView, and Sun Microsystems' Sun Netra Manager.

Commands (MVS, SNA, TCP/IP)

The TCP/IP protocol suite also provides a number of facilities that can be used to collect network service level information. For example, the "ping", TRACERT, and RTT (Round Trip Time) commands can provide an opportunity to measure the network response time proactively and make the network service level reporting feasible. Some ICMP-based "pings" may be rejected by firewalls at the Intranet level but there are ways to overcome this, for example: avoid the use of broadcast messages, or add secondary check rules for further examination by the firewall. The Trace-route (TRACERT) command can measure the component time within the total network time. The RTT command can be used to report the round trip time associated with a "ported" application. *A ported application may contain specific micro-code used by - but not created by - the application).*

SMF data (both session based and interval based SMF records)

Type 118 and 119 SMF records (from IBM) contain TCP/IP session statistics. With some programming effort, session-based historical reports can be produced. SMF interval records could be used for interval-based measurement for service level reporting. These SMF records providing additional diagnostic information are being utilized by network performance analyzers (e.g., NetView, CleverView).

Component trace (IP packet trace)

IP packet traces, or component traces, provide true end-to-end response times for IP packets, and they contain invaluable information. "Socket" programmers are using this information for tuning, testing and planning purposes. They are useful tools to help network system programmers diagnose problems. Running traces continuously for monitoring purposes is very expensive on resources however, and slows the overall TCP/IP stack CPU usage. Since traces record what has already been processed, this approach is not pro-active. However, using traces to establish profiles of critical transactions, and monitoring these transactions on a routine basis is still an excellent way to ensure adequate service levels are being provided.

III. HOW TO TAKE ADVANTAGE OF AVAILABLE TCP/IP COMMANDS

Since there are TCP/IP commands available for producing network service reports, it is important to review what types of results can be generated, and how realistic they are. In order to collect TCP/IP data, a data collection mechanism with a two-layer architecture is used: a master module and a number of agent modules. The master module consists of a set of programs* that issue various data collection commands, dependent upon how a user configures the module, to initiate the various data collection procedures *(*the programs are written in Unix or Linux scripts).* The master module can reside in a workstation with the ability to send, receive, process, store and report historical data. It requires network

access to interface with the agent modules. The agent modules can be remotely installed on desktop computers and their main purpose is to collect specified route information on network devices and send the resulting data back to the master module. The agent modules are also capable of auto-discovering local IP addresses, executing run-time scripts, launching the route collection scheduler, and maintaining logs. The agent modules do not require any local maintenance. Control of the agent modules, including maintenance, is performed remotely from the master module. Agent modules can collect information from numerous IP-based devices. The number of devices to be monitored depends upon the ability of the desktop computer where a given agent module resides.

As an example, the TCP/IP traceroute command can be initiated through the master module to a number of agent modules instructing them to collect TCP route information. Optional parameters could include: maximum time to live (or number of hops to traverse), timeout (how long to wait for a response in seconds), interval (how frequently to collect data in minutes), number of samples to be collected on each device or node, and the destination data set for storing the data.

The following diagram shows the architecture of the data collection system. For a large network, the 3-layer architecture should be used; for smaller networks, the 2-layer architecture without the "regional Collectors" should be used.

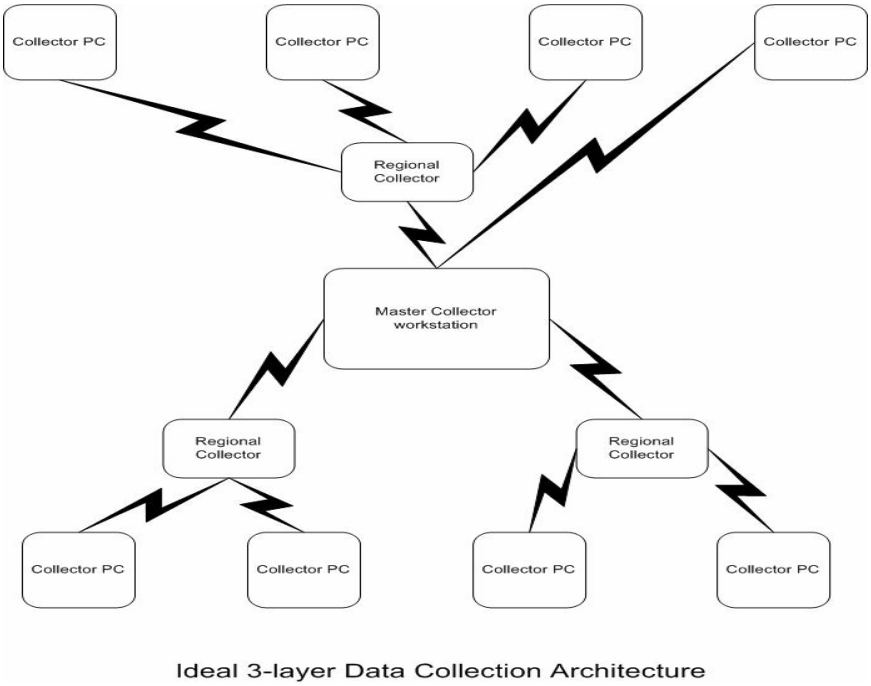

Figure 1 – Data Collection architecture

For network availability measurements, the simplest approach is to utilize the TCP/IP "***ping"*** command to collect timing information from either the "master Collector" or "Regional Collectors". In order to ensure a non-responsive device is not truly "down," there are ways to double-check the validity of the "timeout" response of a ***ping*** command. One way is to issue a second ***ping*** command to the same non-responsive device after a pre-determined time interval, say 5 minutes. If the timeout response is received, one can declare that device is not operational. If a device is powered off however, the ***ping*** command cannot distinguish its true operational status. Other approaches can be taken to avoid powered-off devices being declared failures. One option is to establish a table containing the IP addresses of auto-discovered devices when the network is started and to periodically update the table. Only those addresses will be queried. The validity of this approach depends upon the choice of the sampling size to ensure that the report is statistically significant. There are many good sampling techniques that can be used but such a discussion is beyond the scope of this paper. As mentioned previously, if the ***ping*** is an Internet Control Message Protocol (ICMP) ping, it may not be able reach beyond corporate Internet firewalls.

To initiate the TCP/IP commands for collecting service level related statistics, one needs to define the network IP addresses (e.g., subnets, starting and ending IP addresses, etc.), output data sets (names

and locations), and parameters to be collected. The IP addresses can be gathered using auto-discovery and stored in the output data set. For example, one can define the subnet address as 137.72.43 with starting and ending addresses of 137.72.43.1 and 137.72.43.255 respectively. Once the IP addresses within the defined ranges are discovered, they can be stored in a data set for future usage. When the IP routes have been defined, the route parameters can be selected. Below are the parameters used for this study:

- Data set name – to inform the data collection agent program where to get the IP routes.
- Total Collection Time - time to be spent on the route collection process. Range can be 1-10 minutes with a default of 3 minutes.
- # of Trace routes Per Host - the number of traceroute actions to be performed on each host during the route collection process. Range is between 3 and 99 with a default of 3.
- Maximum Hop Threshold - the maximum number of hops allowable within the route collection process. Range is 1-30 with a default of 10.
- Trace Route Timeout Threshold - the maximum number of milliseconds for the Agent to be idle during the route collection process. Range can be in 100-10,000 milliseconds, with a default of 1000.
- Trace Route 'Pause' Interval - the number of seconds to pause between Trace Route invocations when the # of Trace Route per Host option has been selected. The default is 0.
- Remote Output File Name - the name to be assigned to the file resulting from the route collection (for further analysis and reporting).

Several files can be combined to generate the profile for analysis. The parameters define the location and contents of the files. Below are the parameters that can be used for studying TCP/IP route profiles:

- File(s)
- New Hosts
- Routes
- Hops
- Failing Routes
- Packet Losses
- Looping Hops
- Response Times

The table below shows information based on the specified routes and times:

Session #	IP address	Collection date	Combined date	File_names
1	137.72.43.10	3/27/2004 4:31:19 pm	4/2/2004 9:05:15 am	Rte-data_QA6_137.72.43.10
2	137.72.43.34	3/16/2004 9:16:08 am	4/2/2004 9:05:42 am	Rte-data_QA8_137.72.43.34
3	137.72.43.34	3/26/2004 11:05:00 am	4/2/2004 9:05:50 am	Rte-data_QA8_137.72.43.34
4	137.72.43.10	3/27/2004 4:31;19 pm	4/2/2004 4:00:31 pm	Rte-data_QA6_137.72.43.10
5	137.72.43.34	3/16/2004 9:16:08 am	4/2/2004 4:01:05 pm	Rte-data_QA8_137.72.43.34
6	137.72.43.34	3/26/2004 11:05:00 am	4/2/2004 4:01:16 pm	Rte-data_QA8_137.72.43.34

Table 1 – An Example of the TCP Route Information

The combined information can be summarized to show the aggregated route profiles:

Session number(s)	1, 4, 5
From	03/16/2004 09:00:00 am
To	03/27/2004 05:00:00 pm
Total samples	195
Total Routes	106
Loops in complete routes	6
Packet Losses in complete route	5
Loops in partial routes	21
Packet losses in partial routes	53
Total (Loops & Losses) in routes	85
Total segments	1736
Failed (not reachable)	34
Packet losses	58
Total (Loops & Losses) in segments	92
Best Response Time	10 ms
Average Response Time	10 ms
Worst Response Time	1754 ms

Table 2 – An Example of the Route Performance Statistics

The segment response times can be reported based on the following parameters:

Field Name	**Definition**	
Session	The identifier assigned to the session during import.	
Times	Min RT (ms)	The Minimum Response Time in milliseconds for the segment, based upon the total number of scans done for the route.
	Avg RT (ms)	The Average Response Time in milliseconds for that segment. Avg RT is calculated by averaging the total response times of all hops preceding the current one and subtracting that from the current hop's response time.
	Max RT (ms)	The Maximum Response Time in milliseconds for the segment, based upon the total number of scans done for the route.

Table 3 – The segment Response Time Definition

Note that partial routes may occur when a Trace Route ends before reaching the last hop because the Time-To-Live (TTL) parameter has been exceeded, or when the Maximum Hop Threshold value in the Route is exceeded.

The following table shows an example to further examine the segment performance within a route.

Route ID	Segment #	From	To	Segment status	Avg. RT (ms)	Min. RT (ms)	Max. RT (ms)
1	1	137.72.43.13	unknown	failed	0	0	0
1	2	unknown	64.139.27.1	success	0	0	0
1	3	64.139.27.1	66.80.128.1	success	13	3	23
1	4	66.80.128.1	66.80.133.73	success	27	20	30
1	5	66.80.133.73	66.80.128.1	success	26	20	31
1	5	66.80.128.1	64.139.27.1	success	14	3	23
1	6	64.139.27.1	unknown	success	0	0	0

Table 4 – An Example of Segment Performance Statistics

The same calculations listed in Table 3 can be applied to the web application transaction timing calculation. Various timing components are discussed below:

Let

N_c = web navigate for document completion time

D_c = document download completion time

W_i = individual idle/wait time (i = 1, 2, 3,….k)

Therefore, we have:

Response time (RT) = $D_c - N_S$

Download time (DL) = $D_c – N_c$

Wait Time (W) = $S W_i$

Below is an example of a simple web transaction timing component report which consists of date, time, day, count, average RT, DL and W times, minimum RT, DL and W times and Maximum RT, DL and W times. For illustration purpose, we only display the averaged timing values. All timing units are in milliseconds.

Date	Hour	Day	Count	Avg. RT	Avg. DL	Avg. W
1/12/2003	1300	SUN	6	5741	5732	5
1/12/2003	1400	SUN	16	5483	5477	4
1/12/2003	1500	SUN	10	5536	5525	6
1/12/2003	1600	SUN	21	5842	5834	4
1/13/2003	1100	MON	2	6733	6682	7
1/13/2003	1200	MON	1	6158	6080	7
1/19/2003	0700	SUN	3	13059	13010	6
1/19/2003	1000	SUN	3	5427	5386	2
1/19/2003	1200	SUN	4	5886	5818	3
1/20/2003	1000	MON	1	9678	9616	5
1/20/2003	1100	MON	2	6335	6310	4
1/20/2003	1700	MON	1	5806	5772	4
1/22/2003	1200	WED	5	6304	6251	2
1/22/2003	1300	WED	1	6192	6112	2

Table 5 – An Example of Web Transaction Timing Statistics

The SNMP MIB data can be used to report the utilization profile using the parameters in the following table. (Note: there are more parameters in the MIB but only those related to utilization calculation are used.)

Host Data	Optional (name, location, uptime, contact name, etc)
ICMP Packets in/out	Data transferred in/out for this MIB type.
IP Packets in/out	Data transferred in/out for this MIB type.
UDP Packets in/out	Data transferred in/out for this MIB type.
TCP Packets in/out	Data transferred in/out for this MIB type.

Table 6 – SNMP MIB Parameters

Note that SNMP has multiple versions. Version 1 (SNMPv1) is the initial implementation of the SNMP protocol which is a simple protocol designed to allow networked entities (hosts, routers, and so on) to exchange monitoring information. It is described in Request For Comments (RFC) 1157 and functions within the specifications of the Structure of Management Information (SMI).

Based on Version 1, SNMP Version 2 improved the protocol by adding secure management (via authentication, encryption and access control per object), transporting management information in a more efficient way (with the 'GetBulk' operation), and building a hierarchy of managers as well as a large number of small improvements. The traffic volume can be used to calculate the device or port utilization using the following formula:

Inbound utilization = (IP Packet-In)/device packet capacity in %

Outbound utilization = (IP Packet-Out)/device packet capacity in %

The utilization can be grouped into average, peak/ maximum and percentile (e.g., 90% or 95%). The table below shows the Ethernet port utilization exception report in which the pre-established performance threshold (top 10% with 30% or greater either inbound or outbound in this case) was exceeded.

Date	Device Name	Top 10% Util.	Top 30% Util.	Avg. Util.	Max. Util %
10/10/2003	SC65AR401A1	44.17	22.08	9.22	44.81
10/10/2003	sc270e4d10c1	42.41	38.66	18.37	48.22
10/10/2003	SC64L2N5A3	38.90	31.75	16.01	47.51
10/10/2003	sc3073c9a2	31.90	24.18	13.20	52.69
10/10/2003	SC307L3B10A	31.79	22.91	11.86	46.49
10/10/2003	DPRO301W1B6A3	31.70	24.36	12.18	39.86
10/10/2003	DPRO301W1B6A2	31.67	24.32	12.18	39.47
10/10/2003	DPRO301W1B6A1	31.65	24.28	12.18	39.34
10/10/2003	sc270e3d10c1	31.15	25.22	11.26	42.46

Table 7 – A Sample of Ethernet Port Utilization Exception Report

The device utilization can also be displayed over a long period of time to show the trend. The figure below shows the trend for the Ethernet port over a five-month period.

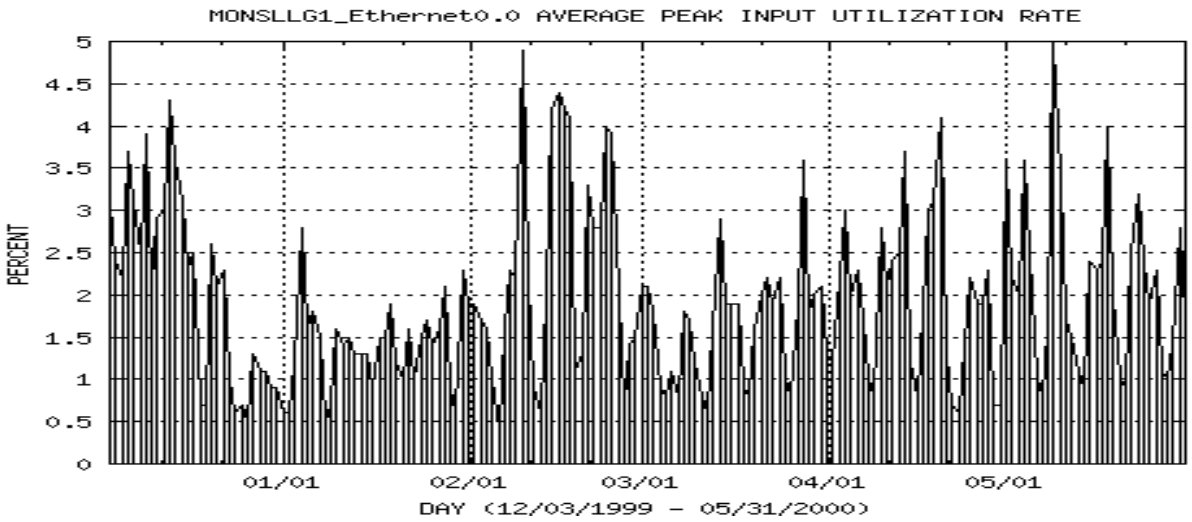

Figure 2 – A Sample Utilization Report

The device (or port interface) availability is measured by its response to the ***ping*** command. If it returns with a positive response, the device is considered "up." If the device does not return a positive response, the second ***ping*** is issued to ensure no transmission error occurred. The device availability can be calculated by the standard formula:

Device up-time/Total advertised Up-time in %

Where the advertised Up-Time is the total time a device is in service minus the scheduled maintenance time. All the device availability statistics can be pooled to calculate the overall network availability. The first example below is the daily availability report for selected device or port interface.

```
03/10/02 COLLCRT00_EOBC0.0     285 100.00
03/10/02 COLLCRT00_Loopback0   285 100.00
03/10/02 COLLCRT00_Null0       285 100.00
03/10/02 COLLCRT00_Vlan100     285 100.00
03/10/02 COLLCRT00_Vlan104     285 100.00
03/10/02 COLLCRT00_Vlan12      285 100.00
03/10/02 COLLCRT00_Vlan16      285 100.00
03/10/02 COLLCRT00_Vlan20      285 100.00
```

Table 8 – A Sample of Daily Availability Report

The **first** item in each line is the date. The **second** item in each line is the resource, which may represent either a specific device or a device interface. The **third** item in each line represents the number of poll samples issued against this resource. The **last** item in each line represents the availability of the resource for the day specified at the beginning of the line.

Table 8 is a sample hosted-based SLR report utilizing TCP/IP commands and modification to handle the firewall restriction, and is provided here as an illustration of the types of network Service Level Reporting that can be done.

Batch/PR (Batch/Performance Reporter) Date: 12/16/2002
REPORT: BatchPRSLS

A1 Special Company
DATE: 2002.350
SHIFT: 1
FROM Hour: 9
TO Hour: 15

NETWORK AVAILABILITY RATING
Total # of Critical Resources Monitored: 16
Total # of Observed Events: 4012
Total # of Resources Unavailable Events 292
Percentage of Network Availability: 92.722%

Top 3 Critical Resources Reporting Unavailable Events

Resource IP Address	# Events	# Unavailable	Percentage
137.72.43.244	991	85	8.577%
137.72.43.21	260	81	31.154%
137.72.43.141	371	65	17.520%

Table 9 – An Example of Device Unavailability Report using batch Report Facility

The same batch report facility can also show the selected critical application's response time statistics as shown in Table 9.

Batch/PR (Batch/Performance Reporter) Date: 12/16/2002
REPORT: BatchPRSLS

A1 Special Company
DATE: 2002.350
SHIFT: 1
FROM Hour: 9
TO Hour: 15

NETWORK RESPONSE TIME RATING
Total # of Critical Resources Monitored: 16
Total # of Observed Events: 4012
Total # of Threshold Exceeding Events: 123
Percentage of Network RT within Threshold: 96.934%

Top 3 Critical Resources Exceeding RT Threshold:

Resource IP Address	# Events	# Over Threshold	Percentage
216.32.74.53	282	44	15.603%
137.72.43.241	319	36	11.285%
137.72.43.242	284	33	11.620%

Table 10 – An Example of Response Time Report using Batch Reporting Facility

CONCLUDING REMARKS

As discussed above, there are several ways to produce network service level reports using available TCP/IP commands to satisfy most service level agreements. The ***auto-discovery*** command can discover active IP-enabled devices which can be used for further study. The ***Trace Route*** command can be used to collect data on selected TCP/IP routes and segments within a route. The ***ping*** command can be used to detect a device's up/down status, which is the basis for availability calculations. Much of the public SNMP MIB information can be used to gather network traffic statistics, which can then be used to calculate utilization for performance and capacity planning studies.

Several examples were given to illustrate using the following processes:

- Data collection using a 2-layer architecture
- Extraction of target IP addresses (or applications)
- Extraction of SNMP MIB information of selected IP addresses
- Selection of SLA parameters to gather reports with date, time and exception range (using various TCP/IP and MIB parameters such as Time-to-Live to calculate the response time)
- Processing of collected data vs. the selected IP addresses and SLA parameters
- Production of SLA reports

Although there are always disagreements* about the accuracy or validity of using just TCP/IP commands for the gathering of network statistics *(*some believe TCP/IP cannot distinguish between a powered-down and a failed network device; others believe the elapsed time between two consecutive "pings" may be too long to detect a troubled network device)*, it has been shown that it is feasible as a methodology for getting started, and for the generation of network service reports that are both satisfactory and cost effective.