
A Practical Approach to Solving Performance Problems with the SAS[®] System

October 17, 2001

Customer Technology Center



Table of Contents

| | |
|---|----|
| Table of Contents | i |
| Introduction | 1 |
| Types of Performance Problems | 1 |
| Performance Solution Process | 1 |
| Defining and Documenting the Performance Complaint | 1 |
| Detecting Performance Issues from within the SAS Session | 2 |
| Detecting Performance Issues at the Host Server System Level | 4 |
| Determining Your Server’s Configuration | 4 |
| Measure System Performance at the Server Level | 5 |
| Measure System Performance at the Network Level | 9 |
| Taking Corrective Action Based on Server/Network Knowledge | 11 |
| Can the Server/Network be Tuned or Modified without Adding Resources to Solve Performance Issues? | 11 |
| Are the Application/Data Management Tunable or Schedulable? | 11 |
| Can Resource or Workload Management Be Employed to Alleviate the Problem? | 11 |
| Are Additional Server/Network Resources Required? | 12 |
| An Ounce of Prevention | 12 |
| Ensure System Sizing is Appropriate for Application/ Software/Solution Demand | 12 |
| When and Where to Get Expert Help | 14 |
| References | 14 |

Introduction

Software performance issues occasionally arise in new implementation testing for SAS[®] software solutions, or in deployed solutions and applications. There is a wide array of contributors to system performance: from application code, data models and process design, to hardware and network architecture and configuration. Given the intricate interaction of these elements, how do you diagnose underlying performance problem causes and determine effective solutions? This paper will suggest a stepwise method that uses measurement tools to diagnose performance problems involving the SAS system, steps to correct those problems, and how to take preventative measures against more problems in new and existing systems.

Types of Performance Problems

How do you know if a performance problem exists? The definitions of a performance problem can be delineated in a number of ways:

- 1) When jobs, job steps, processes, or functions cannot execute within a required time period.
- 2) When jobs, job steps, processes, or functions consume an inappropriate amount of resources to execute or fail within a required time period.
- 3) When a system begins to break down or fail frequently due to overuse.
- 4) When a system cannot sustain expected or unexpected growth (scalability) without resulting in the situations above.

The bottom line is that system performance expectations are not met, and people are not happy. When this happens it is imperative to diagnose and solve the performance issue(s) that will ensure satisfactory system performance and delivery to the customer.

This paper will focus on a process and tools for diagnosing performance problems, and discuss corrective actions. While efficient application/software design and data model

deployment are important to good performance, they will be examined only in the aspects of their setup on the hardware architecture. It is left to the reader to have a requisite understanding of efficient application and data design.

Performance Solution Process

A methodical process can confirm, diagnose, and correct performance problems. The process is iterative, and works in an investigative fashion. The process follows the following basic steps:

- 1) Define and document the performance complaint
- 2) Diagnose the performance problem
 - a. Review SAS logs for information
 - b. Measure system performance from within the SAS session
 - c. Understand your system configuration
 - d. Measure system performance at the server level
 - e. Measure system performance at the system level
- 3) Correct the performance problem based on diagnostic measurement
 - a. Applicable server/network tuning
 - b. Applicable application/process/data management tuning
 - c. Investigate benefits of workload management
 - d. Adding physical system resources

Many of these steps have multiple “how to” subtasks, explained in the following appropriate sections. The conclusion will offer advice on how to recognize the need for and to obtain expert help.

Defining and Documenting the Performance Complaint

The first step in diagnosing and solving performance problems is to understand the complaint. Is there a problem with a specific task?

- Does a task run out of resources causing it to fail?
- Is it too slow?

- Does it consume far more resources than expected?
- Are the problems pervasive across many processes and users?

First clues to the problem are usually apparent by poor test/application response time. Compiling the available information will help direct the diagnostic testing discussed later. Once the complaint is established, a process can be undertaken to find out the root cause(s) and institute correction.

It is helpful to witness the problem firsthand. This will require subsequent setup/run of the problem activity if it is non-recurring task activity. It is best to perform this activity under the same circumstance in which the original problem existed. Those circumstances include:

- 1) Running the exact same procedures and code on the same data. This should be documented into an identified test bed so that the test activity and data will remain constant throughout the diagnostic cycle. This documentation should include the problem complaint definition listed above, the executing code/application, and the data model documentation.
- 2) Running under the same system “load” or like the same physical system (On a multi-user system this means testing with the same number of users with the same level of activity. This can dictate experiencing cyclical activity factors influenced by time of day, day of week, day of month, business cycle activity, etc.).

Once the problem has been established as recurring or persistent, it can be measured and diagnosed. The first step is to measure the behavior from within the SAS Session using the FULLSTIMER option.

Detecting Performance Issues from within the SAS Session

The SAS System provides the FULLSTIMER option to collect performance statistics on each SAS step, and for the job as a whole and place them in the SAS log. It is important to note that the FULLSTIMER measures only give you a snapshot view of performance at the step and job

level. Each SAS port yields different FULLSTIMER statistics based on the host operating system. See the SAS host specific documentation for the exact statistics offered. FULLSTIMER is invoked as a SAS option and takes effect after the option invocation.

Why start with the FULLSTIMER option for monitoring? The best reason is that it tells you what is happening with the SAS system specifically. The statistics it provides are at the job step and can help pinpoint performance problems down to the step. This is extremely helpful in narrowing troublesome activity, and relating it to what your code is telling the system to do. (Note: If the test execution is long, expensive, high impact to the environment, and is not easily set up, the SAS session monitoring can be done simultaneously with server and system performance monitoring, which is introduced later in this document.)

FULLSTIMER measures can be used to help determine if more in-depth performance monitoring with host monitoring or third party tools is indicated.

A sample result of a FULLSTIMER option UNIX output for a SAS Data Step is listed below:

```
NOTE: DATA statement used:
real time                0.06 seconds
user cpu time             0.02 seconds
system cpu time          0.00 seconds
Memory                   88k
Page Faults              10
Page Reclaims            0
Page Swaps               0
Voluntary Context Switches 22
Involuntary Context Switches 0
Block Input Operations   10
Block Output Operations  12
```

It is important to know how these numbers are defined and what can be derived from them.

FULLSTIMER Statistics Definition and Interpretation

Real Time – the Real Time represents the elapsed time or “wall clock” time. This is the time spent to execute a job or step. This is the time the user experiences in wait for the job/step to complete. Note: As host system resources are heavily utilized the Real Time can go up significantly – representing a wait for various system resources to become available for the SAS job/step’s usage.

User CPU Time – the time spent by the processor to execute user-written code. This is user-written from the perspective of the operating system and not the customer's language statements. That is all SAS system code that is not operating system code.

System CPU Time – the time spent by the processor to execute operating system tasks that support user-written code (all CPU tasks that were not executing user-written code). The user CPU time and system CPU time are mutually exclusive.

Memory – Memory represents the amount of memory allocated to that job/step. This *does not* represent the entire amount of memory that the SAS session is consuming, as it does not reflect any SAS overhead activities (SAS manager, etc.).

Page Faults – Represents the number of virtual memory page faults that occurred during the job/step. Page Faults are pages that required an I/O to retrieve (a read was done to the I/O subsystem).

Page Reclaims - Represents the number of pages retrieved from the page list awaiting re-allocation (all done in memory). These pages did not require I/O activity to obtain.

Page Swaps – The number of times a process was swapped out of main memory.

Voluntary Context Switches – Represents the number of times a process releases its CPU time-slice voluntarily before it's time-slice allocation is expired. This usually occurs when the process needs an external resource, like making an I/O call for more data.

Involuntary Context Switches – The number of times a process releases its CPU time-slice involuntarily. This usually happens when its CPU time-slice has expired before the task was finished, or a higher priority task takes its time-slice away.

Block Input Operations – The number of "bufsize" reads that occur. These are I/O operations to read the data into memory for usage. Not all reads have to utilize an I/O operation since the page being requested may still be cached in memory from previous reads.

Block Output Operations - This represents the number of "bufsize" writes that occur. These are the same as block input operations except that they pertain to the writes to files. As in the case of block input operations, not all block outputs will cause an I/O operation. Some files may still be cached in memory.

Performance problems usually involve one or more of the following physical areas:

- CPU activity
- Memory activity
- I/O subsystem activity (disk and file systems)
- Network activity (this will be discussed outside the context of the SAS system later).

By examining FULLSTIMER statistics, and interpreting what is happening with and between the factors producing the measures, we can get a quick idea of where the system is having problems. We can then resort to host-level and third-party measuring tools to obtain a very detailed picture of problem issue. If the host-level and third-party tools give such detail why not use them first? Very simply, there are many tools to use, and each is fairly good at one or more specific areas of investigation, such as CPU, Memory, and I/O. Also some require Server Root-Level access to deploy. FULLSTIMER is quick and easy (incorporated in the SAS system), requires no special privileges, you can do it yourself, and it can help quickly narrow the field of things to test next.

The following is a general list of interpretations you can make using FULLSTIMER:

Real Time/CPU Time. The most valuable way to use FULLSTIMER is to compare timing information. By comparing the Real Time (elapsed time), with the total CPU time (system CPU time plus user CPU time) you can quickly determine if the problem is CPU related.

- If the Real time and total CPU time are within 15 percent of each other, this usually indicates that the system is moving data well (at least during the run time of that job/step processing). This means that the ratio of CPU process time is close to that of the total job. This indicates that the system memory, disk system, and file system are getting data to the CPU quickly enough to not be a problem. If you are experiencing bad task

performance, and the real and CPU time are within 15 percent of each other, it most likely means that your task is CPU bound. The only way to improve the performance will be to get a faster CPU, split the process over more CPUs (multi-threading or parallel processing), or reengineer the code to be more efficient.

- If the Real time and total CPU time are routinely very disparate, (for example if there is a 50 percent margin between them), then you very likely have a problem in your system getting information to the CPU fast enough. Make a closer examination of the Memory and I/O subsystems using the host or third-party tools mentioned in the next section.

Other valuable information from FULLSTIMER can be gained by looking at the other statistics:

Memory. If a sizeable quantity of memory is used and your elapsed time differs greatly from your total CPU time, you may also want to take a close look at your memory using host or third-party tools that are mentioned in the next section.

Involuntary Context Switches. If Involuntary Context Switches are consistently high across many steps and jobs over long time-periods, then your CPU system is under a heavy load, and you will want to examine that more closely with the tools mentioned in the next section.

Page Swaps. If Page Swaps are consistently high then your memory system is being stressed, and needs more examination.

Other statistics like Block Input and Output operations, Page Faults and Reclaims, and Voluntary Context switches can hint at issues, but require more corroboration from the measures previously discussed to make a case for narrowing down investigation. These measures could be high in-and-of themselves without being a symptom of performance problems.

Once FULLSTIMER statistics have been examined, they should help indicate which area(s) should be examined in more detail. It is often the case on overloaded systems that multiple areas present themselves for examination. The FULLSTIMER activity should help point to which tools that could be used first to get a more detailed level picture of any

hardware/file system issues. This comprises our next step, detecting performance issues at the host server system level.

Detecting Performance Issues at the Host Server System Level

The first step in examining our physical systems will involve employing host and third-party tools for continued deeper level measurement and monitoring at the server level. Before that can be done, it is helpful to know something about your server system's configuration.

Determining Your Server's Configuration

Server configuration information is easily obtainable at the command line level for most systems. Your systems administrator can supply any information that you cannot obtain or derive on your own. The information below pertains to the Server only. The I/O and storage management subsystem information (disk, disk arrays, volume arrangement, cabling, etc.) is best acquired through your systems administrator. The setup and arrangement of the volume and physical storage systems can be quite complicated, and depending on which facilities are chosen to set up and manage them (OS level, or third party such as RAID Manager, Veritas, etc.) the information can be difficult to interpret. Utilize the help of your systems administrator to fully understand the I/O and storage management systems.

SUN. For SUN systems the `prtdiag` command yields output that gives detailed system configuration information, including the number of CPUs, their clock speed, e-cache, adaptor cards, memory interleaves, and so on. This executable file is usually available from `/usr/platform/xxx/sbin/prtdiag`, where `xxx` is the output from the `uname -m` command. Another command to examine is the `prtconf` command, which can usually be found in `/etc` or `/usr/bin`. To obtain the desired configuration output, run this command with the `-v` option. Refer to the man pages for additional usage and interpretation.

HP. Server configuration information can be obtained for HP Servers by using a couple of

tools, one is the sam (system administration) tool that ships with HP-UX, and the other is a third party tool, glance+. For details on how to use these please refer to:

<http://www.sas.com/partners/technology/hp/hpux/wp.pdf>

IBM. For IBM AIX servers there are several commands to use to get configuration information. For server configuration information, `lscfg -vp` will yield CPU, memory, I/O adaptor, and disk information. Logical volume information can be derived from the `lsvg -o` command, and specific drive information can be obtained using the `lslv xxx` (where xxx is the logical device name).

NT. There are at least two easy ways to get configuration about NT machines. One is to issue the `set | more` (set pipe more) command at the DOS prompt, which yields some configuration, address and architecture information. Another way to get broader and more detailed information is to use the WNT Diagnostics arrived at through the *Programs, Administrative Tools, Windows NT Diagnostics* menu selections from the *START* menu button. The menu selections can be scripted into a batch file for command level use on networked systems.

Measure System Performance at the Server Level

Once performance statistics have been gathered from the SAS FULLSTIMER facility, additional host-system and third party tools can be employed to get a more detailed look at the process activity and help detect performance issues. You can begin in the area most affected according to your FULLSTIMER statistics, but it is wise to capture initial measures in all the areas CPU, Memory, and I/O. This is necessary because another can generate apparent problems in one system. For example memory shortages or paging problems can overwork the I/O subsystem making it appear to have “bottlenecks”.

This section will introduce a few of the available host system and third party tools and what they can tell us. Much can be written on the detailed usage and interpretation of these tools, the size of this presentation limits the discussion for each

tool to a brief description, and where and how it can be used in performance problem diagnostics. The tools highlighted will be host-level tools that are generally available on most systems, `ps`, `iostat`, `vmstat`, and `sar` for UNIX hosts, `mpstat` for SUN hosts, and Performance Monitor for NT hosts.

Note: When first using these tools it is important to work with the your systems administrator. Some tools require root-level privilege to employ (such as `sar`), and others can be difficult to interpret at first. Also tools may have different options and flags on different host systems – see the “man” or other documentation for your specific host. It takes time and experience to combine the measures across several tools and subsystems to accurately diagnose what is happening to the system.

Detecting UNIX Host Problems

One of the best things to do is get a look at the general system load itself. If you or your administrator occasionally measure the system load, detecting differences in the overall system when problems erupt is much easier. For UNIX hosts, the `ps` tool is a good place to start.

Tools to Assess General Server Process Load

ps. `ps` gives information about active processes and the resources they use. It is good for looking at the processes that are using the most CPU and Memory. The `-e` option lists information about every current running process. There are several run options. Please refer to the man documentation for your system. `Ps` can tell you how many processes are currently active, and what each individual process is currently (at a point in time) requiring of the CPU and memory resources. Its output has columns that can tell:

- How much CPU time a process has used
- The percentage of total available CPU time used since a process started
- The size of a data section in progress
- Percentage of Real Memory being used

The `ps` output can be sorted to quickly detect which processes are utilizing the most CPU time and Memory. For example the syntax `ps -ef | sort +3 -r | head -n 5`, prints the first five lines of the output sorted by the third column (recent used CPU time). If memory problems seem to

occur from other evidence, look for jobs with a large resident memory set (RSS) for potential culprits.

If the system is heavily loaded, refer to the recommendations in “An Ounce of Prevention” below.

Tools to Assess CPU Usage

Once a general load picture is obtained, and a general idea of how much CPU time is being utilized by the active processes, more specific CPU measurement can be gathered with other tools.

iostat. *iostat* works natively in most of the UNIX host platforms. The first line of output is for summary statistics since the last machine reboot, and should be ignored. All the subsequent lines are produced based on the parameters of how often you commanded it to generate statistics, and for how long (number of harvests). The command follows the basic syntax of {*iostat* <-parms> <-c count> <-w wait> <drives>}.
`iostat -c 10 20 - run the CPU statistics at 10 second intervals for 20 intervals).`

Use the `-c` option with interval and count for CPU measurement (e.g., `iostat -c 10 20` – run the CPU statistics at 10 second intervals for 20 intervals).

If the *iowait* and CPU idle times are constantly zero, and the system is busy, you could have a CPU bound situation. Run *ps* to see what processes are using CPU time, and *vmstat* on any host (also *mpstat* on SUN) to get a read on the CPUs.

mpstat. This tool is available on SUN Solaris only. Specify interval and count { *mpstat* 20 10 } – run at 20 second intervals for 10 intervals.

If the “*idl*” time column is always close to or at zero, take a closer look at CPU utilization. If the “*usr*” and “*sys*” times combined are always over 80 percent (for many hours at a time), your system could be CPU bound. Keep in mind these are aggregated views of all the jobs running on that CPU during the test intervals.

To better test the effect of single jobs, first identify your CPU hogs in *ps*, and then run them singularly with *mpstat* (nothing else running – a quiet test). This is the only accurate way to obtain a good measure of how a resource is being

used by a task that has no other competing resources. If these jobs that seem to spend a lot of time on the CPU have combined “*usr*” and “*sys*” times close to the job wall clock time (elapsed time), the task is probably CPU bound. If it is not performing as you expect, or if it is helping to clog up the system in busy times, you will have to re-engineer the task to use less cycles, schedule it in less busy periods, or get a faster CPU.

Another good hint at CPU overload are the involuntary context switches. If SAS FULLSTIMER for your host version supports that metric, corroborate it with the “*icsw*” field in *mpstat*. Even on its own *icsw* field in *mpstat* can give you a good idea of how often time slices are expiring without task completion, or getting stolen by other higher priority processes. If the involuntary context switches are constantly high for long periods of time (hours) you have a heavy CPU load situation.

vmstat. *vmstat* produces statistics regarding process, virtual memory, disk, trap, and CPU activity. It is usually run with flags to request how long the report run periods should be (interval), and how many periods should be run (count). Flags are not necessary for CPU statistics only, but it is wise to turn them on if you are going to look at CPU, I/O and Memory from the same set of runs.

CPU statistics from *vmstat* consist of user and system time, CPU idle time, and CPU idle time waiting for pending disk I/O. The first two are most useful for CPU indications. If the user plus the system CPU time consistently stays over 80 percent in a multi-user environment, the system *could* be CPU constrained, and further checks are warranted.

Tools to Assess Memory Usage

ps. The *ps* facility can give you a fair idea about how many jobs are running, and by sorting them by *%mem* or *RSS* you can get a quick picture of who your top memory users are. Examine the jobs involved to see if the Data Steps or Procedures can be altered or rearranged to utilize less memory.

vmstat. Several memory and paging statistics are tracked, most notably the active virtual pages (allocated), the size of the free list, and page-ins and page-outs. The free list should not

consistently get low (e.g., below 120 on AIX, check with your systems administrator for what is normal for your system). If it does the system is constantly paging in and out resulting in a need to obtain more memory or invoke utilization management of the memory resource. The page-ins and page-outs can give you clues to memory problems. If page-ins get high then you should investigate, and consistent page-outs point to memory shortages. In addition, consistent swap outs point to extreme memory shortages. If your flavor of *vmstat* supports the “w” (number of run able processes that have been swapped out) and it is non-zero, your system is suffering from a serious memory shortage. Also if your flavor of *vmstat* supports the “de” column, anticipated short-term memory shortfall, and it is non-zero, your system is performing desperation swapping and is extremely short of memory.

Additional memory monitoring tools include *svmon*, and *sar*, and other third party tools listed later in this document.

Tools to Assess I/O Usage

iostat. *iostat* is probably the single best “free” tool to use in measuring general I/O activity. It does a fairly good job of reporting terminal, disk and tape I/O activity. The first line of output is for summary statistics since the last machine reboot, and is not that helpful. All the subsequent lines are produced based on the parameters of how often you commanded it to generate statistics, and for how long (number of harvests). The command follows the basic syntax of { *iostat* <-parms> <-c count> <-w wait> <drives> }.

We recommend using the following parameters for I/O reporting:

- *-c* Report the percentage of time the system has spent in user mode, in system mode, waiting for I/O, and idling.
- *-C* When the *-n* and *-x* options are also selected, report extended disk statistics aggregated by controller id.
- *-d* For each disk, report the number of kilobytes transferred per second, the number of transfers per second, and the average service time in milliseconds.
- *-m* Report file system mount points. This option is most useful if the *-P* or *-p* option is also specified.
- *-M* Display data throughput in MB/sec instead of KB/sec.
- *-n* Display names in descriptive format (for example, *cXtYdZ*, *rmt/N*, *server:/export/path*).
- *-t* Report the number of characters read and written to terminals per second.
- *-x* For each disk, report extended disk statistics. The output is in tabular form.
- *-z* Don't print lines whose underlying data values are all zeroes.

Disk load can be viewed in terms of number of kilobytes and transfers per second for reads and writes, average number of commands being processed by the drive, the I/O service time, percentages of time that commands were waiting in the queue for drive access, and number of commands active on the drive at the measurement time. This information can tell you how busy an individual disk is. Some things to watch for are:

- If your *iowait* is consistently very high you may have a memory shortage due to paging, or a bottleneck in your I/O subsystem.
- If your CPU utilization is consistently low, and the CPU idle times and *iowait* times are greater than 25 percent, you could have an I/O or disk bottleneck. Constant high *iowait* times generally point to excessive paging, or unbalanced disk loads, fragmented data, or unbalanced usage patterns against file systems or disks.
- If the CPU spends over 50 percent of time in the system state, an I/O problem can exist, look at other measures (*iowait*) and *sar -d*.
- If the physical disk percentage active time is consistently high and the transfer rate is low, then you could have physical fragmentation of the volume, file system or file. (Note: long sequential reads/writes may exhibit this to a degree and not be an unusual event). If the physical disk percentage active time is consistently high by itself (> 25 percent) you could have an I/O bottleneck.
- If you combine the transfer rates for all the physical disks on the same channel (controller) and compare to the channel throughput rate (use 50 percent of the stated throughput of the hardware channel by the

Hardware Vendor), then you could be experiencing a flooded adapter and may need to add more adapter bandwidth.

mpstat. As a SUN only tool, *mpstat* is not the best tool for I/O assessment on SUN, but it can provide corroboration in a general way for spotting symptoms. A high percent CPU wait and idle times when the system is symptomatically heavily loaded are the indicators. This tool is not discussed in detail here, but is worth checking out if you have a SUN system.

vmstat. *vmstat* is excellent for memory diagnostics, but it can also tell something about general I/O health. The thread information and CU information can indicate if the I/O system appears to be backed up or have a bottleneck. If so, using *iostat* can verify and further pinpoint the problem.

Thread Information – *vmstat* yields information about how many kernel threads are currently in the run queue and how many of those are in a wait-state (i.e., waiting for I/O, etc.) The wait-state threads should be close to zero in a healthy system.

CPU Information – If the CPU idle time waiting for I/O stays consistently high (check with your Systems administrator to determine what high is for your system) your I/O subsystem may not be balanced appropriately or a disk is too intensively used (I/O bottleneck).

In addition to the UNIX standard host utilities above, there are additional easy-to-use utilities and tools worth investigating. One host utility referenced above but not discussed is *sar*. Unfortunately, *sar* requires root-level privilege to operate. It can produce a lot of output based on the options used, but is very comprehensive. It is valuable if you can obtain root privilege to run it, because it can be used as a stand-alone to diagnose CPU, memory and I/O issues, in addition to corroborating other tools findings.

Third-Party Measurement Tools

Most of the third-party tools (tools that are purchased separately, and do not come as part of the host system) offer easy-to-use graphical interfaces, with very understandable output. Some of the third-party tools worth learning and

using if you are involved with performance monitoring on a regular basis are:

- Top (not an OS utility, multiple vendor hosts)
- Proctool (Solaris only)
- Glance+ (HP-UX, Solaris, AIX)
- Openview (HP-UX)

Detecting NT Host Problems

On the Microsoft Windows/NT platform the Microsoft Performance Monitor (PerfMon) is one of the most widely used tools for detecting and analyzing performance bottlenecks. Perfmon has counters of various types (instantaneous, averaging, and difference) for the system “objects.” A minimum baseline of objects (as defined by Microsoft) to monitor when doing performance monitoring include; Memory, Processor, Physical Disk, and Network objects. As an example each processor in the system would be an instance of the Processor object. Other objects include cache, logical and physical (if using a RAID system) disk, server, and system.

The tool is fairly comprehensive and can be used to detect problems in the object areas listed above. What it can be used to discern concerning some of the objects is listed below (the network object is discussed in “Measure System Performance at the System/Network Level” in a subsequent section).

CPU – The two most common causes of processor bottlenecks are CPU-bound applications and drivers, and excessive interrupts generated by inadequate disk or network components. An assessment of the following counters gives some numbers to watch for:

- %Processor Time – processor busy time = %Privileged Time + % User Time. When it is consistently above 75 to 80 percent it is a bottleneck
- %Privileged Time – Processor Time spent servicing operating system services. Should average below 75 percent or it can be a bottleneck
- %User Time – Processor Time spent running user services (e.g., SAS applications). Should be below 75 percent.
- Interrupts/sec – Number of interrupts from applications and hardware devices. If this

number gets very high and stays there, it can indicate a hardware problem. This is harder to diagnose because this number does not delineate voluntary versus involuntary context switches.

- System: Processor – Number of requests the Queue Length processor has in the queue. Each request is a waiting thread. This number should normally be zero. If it is consistently over two a problem exists. Look at the processes to see which one is causing the bottleneck.
- Server Work Queues: Queue Length – Number of requests in the queue for a particular processor. Greater than two is a problem. For multi-processor systems the queue length should not be more than $n + 1$, n =number of processors.

Memory – Using the Perfmon tool to look at the memory statistics can give good indicators of memory bottlenecks. A high, sustained rate of hard page faults is the best indicator of memory problems. The following counters for the memory object are listed with Microsoft's indications of when they represent memory bottlenecks, and are indications for adding RAM:

- Pages/sec - Extended periods over five indicate a problem (range values are 0 to 20).
- Available Bytes – Amount of available physical memory. This is normally low because of the NT Disk Cache Manager using extra memory for caching until requests for memory occur. If Pages/sec are consistently below four MB, excessive paging is most likely occurring.
- Committed Bytes – Virtual memory allocated to physical RAM or the page-file. If the committed bytes are greater than RAM, then you should add RAM.
- Pool Non-paged Bytes – Amount of RAM in the pool non-paged memory area (space where OS components perform their tasks. If the value steadily increases without an upsurge in user activity, something might have a memory leak. This needs to be investigated.

I/O – Disk bottlenecks can occur fairly easily because disks tend to have slower response times than other resources. In addition, poorly performing disks can actually hide memory

problems. Some of the disk counters and their values to watch for are listed below:

- %Disk Time – This is the percentage of time the disk is busy with reads and writes. If the value is always near 100 percent the disk is very heavily used. A better target to shoot for is a high around 50 percent consistently.
- Disk Queue Length – This represents the number of waiting disk I/O requests. If this is consistently two or higher, upgrade the disk.
- Avg. Disk Bytes/Transfer – Average number of bytes transferred to or from the system during read and write operations. This number should be large to indicate system efficiency. This number is hard to gauge for acceptability due to the different types of processes being run, and you will have to judge it by experience.
- Disk Bytes/Sec. – This is the transfer rate from and to disk during read/write operations. Same advice as Avg. Disk Bytes/Transfer.

Measure System Performance at the Network Level

If your system performance works well on the host (no network access involved to get the job done), but very slow when network access is involved in the application, you might want to measure the network performance.

If you suspect that a network problem may be contributing to your performance trouble there are some quick monitoring tools that can help. These tools can give you enough information to work with your network administrator to perform more in-depth testing if it is indicated. Keep in mind that all networks get heavily loaded at fairly regular cycles by nature of their usage. It is when the network stays heavily loaded on a continual basis, and throughput performance is constantly unacceptable that you want to spend a lot of time here.

UNIX Host Network Monitoring

netstat

One of the easier UNIX host network performance indicators to use is *netstat*. *netstat*

can help indicate whether network problems exist, and whether they exist network side of the system or on the host side. The use and interpretation of this information is best left in the hands of or interpreted with the involvement of the LAN and Systems administrators. But it can be used for preliminary looks if those resources are stressed or unavailable.

The *-i* option of *netstat* will give a network interface device level report that will tell among other things:

- Device name
- Network to which the Interface is connected
- Internet address of the interface
- Number of input packets received by the interface since last boot
- Number of input errors occurred at this interface since last boot
- Number of output packets sent by this interface since last boot
- Number of output errors occurring at this interface since last boot
- Number of collisions detected at this interface since last boot

netstat -i statistics indicating potential problems are:

- If the input errors are high (larger than 0.025 of all input packets), it may be an indication of faulty hardware on the network itself. Have your network administrator investigate.
- If output errors are high as well (same percentage as input errors), it could indicate a problem in your system's network controller, your network cable drop, or on your server itself.
- If the number of collisions is consistently greater than ten percent of the total output packets for this interface, your network could be overloaded. In addition, if the interface is an NFS drive, poor performance can be manifested as a consistently very high number of collisions.

Without any options invoked, *netstat* produces a very usable activity connections report. Refer to the documentation on this tool to understand the range of detailed information it can yield.

NT Host Network Performance Monitoring

The Perfmon tool discussed earlier for the NT environment also produces some network related statistics that are useful. They are explained below:

Server:

- Bytes Total/Sec. – Number of bytes the server has sent and received over the network. Indicates how busy the server is on sending and receiving data.
- Logon/Sec. – Number of logon attempts for local, across the network, and service account authentication that occurred in the last second. This indicates how domain controllers are handling load.
- Logon Total – Number of logon attempts for local, across the network, and service account authentication that occurred since the server was last started up.

These are all good server measures to evaluate domain controllers. If you are expecting a much higher server login rate than what is being experienced (e.g., people are not getting into the server), your domain controllers could be overloaded, and your LAN Administrator will determine if the situation necessitates adding more. This is typically only seen in very heavily trafficked servers.

Network Segment: %Network utilization – Percentage of bandwidth in use for the LAN segment. This is used to monitor the effects of different network operations like account synchronization and logon validation, etc. Your LAN Administrator can determine appropriate action to take if the segment bandwidth is suffering.

Network Interface:

- Bytes Sent/Sec – The number of bytes sent using the selected adapter.
- Bytes total/Sec – Number of bytes sent and received using the selected adapter. Upgrade the adapter if there is a problem.

The network interface statistics are of interest to the LAN and Systems administrators to understand if a particular adapter is overloaded.

Taking Corrective Action Based on Server/Network Knowledge

Once you have identified where the “slowdowns” can be found in your system, determine what to change for improvement. Performance bottlenecks manifest themselves in the I/O, CPU, memory or network subsystems and can be alleviated or eliminated with one or more of the following:

- System tuning,
- Application tuning,
- Application architecture,
- Data architecture,
- Workload management,
- Increasing system resources

Each of these areas needs to be considered when a performance problem exists. Making changes in one, or more of these areas may address the problem readily, but they should all be examined for possible contribution to bring efficiency.

Can the Server/Network be Tuned or Modified without Adding Resources to Solve Performance Issues?

Tuning or modifying the system kernel settings, setup, and behavior can help alleviate performance problems depending on the cause. This activity can cover setting system parameters (cache, buffers, paging, protocol limits, etc.), re-arranging or tuning the I/O subsystem (disk volumes, file systems and architectures, etc.), to gain better throughput and performance. Your systems or network administrator is the best resource to assess and perform this tuning work.

The important thing is that he or she clearly understand your application behavior and setup, the performance demands it will place on the system, and how to optimize system setup for decision support activity. Work with your administrator to ensure there is a good understanding of the process and performance characteristics associated with the process. Mixing heavily transactional systems and decision support activity on the same server can

be very tricky in terms of system tuning paradigms.

Decision support activities (reporting, analysis, data mining, HOLAP, etc.) behave significantly different from transaction-based systems at physical layer. Refer to “An Ounce of Prevention” further in this document.

Are the Application/Data Management Tunable or Schedulable?

Application Coding and Behavior. Examine the application(s) in light of what types of performance problems you are experiencing. Walk through your application process design and coding efficiency and make sure it’s efficient. This review can find and stem unnecessary use of CPU, memory, and I/O resources. Also look at your process in terms of local and remote work. Are you shipping large amounts of data over a network? If so, can you work on remote servers and ship only result sets back to the application server? Studying the locality of processing (whether processing is taking place on the client or server) and how data is moved from client to server to process can aid in reducing network traffic. Applications that are improperly coded, can inadvertently cause a remote SAS dataset to be pulled to the local machine or engine for processing.

Data management. If you are experiencing I/O bottlenecks, review your data models against the exploitation patterns and demand load placed on them. Examine your physical data model and file placement within the volume/file system setup to determine which file or files may be heavily utilized. Based on that information you may wish to alter your physical data model, or consider moving the affected file(s) to a less busy file/volume set, or replicating the file(s) and tuning the application so that the demand load is partitioned across files in different physical disk/volume/file system areas.

Can Resource or Workload Management Be Employed to Alleviate the Problem?

After examining system tuning, and application and data management design and efficiency, it is

wise to examine whether imposing scheduling, or managing, controlling, and optimizing resource consumption can improve system performance. This can be done in a variety of methods from the simple to the complex, just a few are:

- Placing kernel limits on the users themselves (number of active processes allowed, memory consumption allowed per process, user priority – *nice*, etc.)
- Utilizing a system scheduler (third party scheduling tools such as Platform Computing's Load Sharing Facility, CA's Unicenter, Control-M, and others) to force resource intensive jobs to certain execution time windows, servers, etc. at chosen priority levels. Some applications may need to have scheduling constraints built within them to control the number of concurrent processes submitted to the system.
- Utilizing a software management facility that controls host or network resource usage by user or process classes. Good examples are SUN's Resource and Bandwidth Managers and HP's Workload Manager and Process Resource Manager. These types of tools have differing methods and approaches, but typically enforce user or application class or work group constraints on resource utilization, manage resource utilization for maximum workload, etc. These management facilities are especially valuable if you are working on a large system with a high number of users, and experience a large and variable job mix demands.

Resource and workload management tools are essential for getting the most out of large complex systems. On smaller systems with less concurrent processes, the system tuning topics discussed earlier may alleviate a lot of problems without the need to add system hardware resources.

Are Additional Server/Network Resources Required?

Another approach to solving performance problems is adding server or network resources. You should have worked with the server and /or Network administrators through this process. Together you should have developed a good definition of the problem, what has been done to

correct it, and if additional system resources are needed to provide the capacity for acceptable future performance. Your administrator and hardware/software vendor and/or partner can work with you to build a best path expansion plan.

In concert with the previously discussed corrective action items, hardware resource addition can often solve many problems, and adding hardware tends to be the first thing many people do if they have the budget for it. Even if a simple hardware resource addition looks like it will solve a problem, it is still wise to cover the preceding "Taking Corrective Action" topics. This will ensure that no grave inefficiencies are left uncorrected. If not, adding hardware can work for a while, but at some point those grave inefficiencies likely will catch up with you as the system grows and scales. The next section, "An Ounce of Prevention," will give planning system sizing and capacity for SAS applications and solutions.

An Ounce of Prevention

Ensure System Sizing is Appropriate for Application/Software/Solution Demand

One of the best ways to avoid performance problems is to evaluate the capability of the physical system you are working within to meet the demanded activity. This evaluation is a two-fold process: first to predict level of usage and demand for new systems, or measuring demand on existing systems; second, to determine if the system throughput capacity is adequate to meet continuing demand.

Many white papers have been written on the process of measuring and determining system sizing appropriateness. The subject calls for a fair amount of knowledge of the hardware/software systems, the application software, the general I/O, memory, and CPU demands of the application mix, and the setup and interaction of all of these things. This section will give a basic outline of the process, and point the reader to appropriate resources for additional direction and help.

Fundamental application activity information that is needed to determine system sizing and tuning:

- Number of SAS users and concurrent jobs on the system
- How many jobs performing what activities and what SAS products are used: simple queries, complex joins, HOLAP, STATS, data mining, etc.
- Data sizes and data model the processes are exploiting
- System growth and scalability requirements – data, users, processes
- Other process loads currently on the machine(s) and/or network besides SAS
- Expected response time for the various activities

The above activities list can be used to generate a general “demand load” guiding the system sizing and setup. Using some general factors and working with your systems administrator, you can arrive at an appropriate configuration for your system. Some factors to consider in system sizing and setup include:

CPU Requirements – General experience has shown that 8-10 concurrent SAS sessions per CPU for nominal reporting, querying, HOLAP, generally delivers adequate performance. The number of sessions per CPU drops to 4-5 concurrent SAS sessions for data mining and heavy statistical or computational processes.

System Memory – Generally recommended memory amounts are 32 MB per SAS session or 512 MB per CPU for most work. Exceptions are:

- MDDDB tables larger than 32MB/512MB – use size of largest MDDDB + a small safety factor.
- Data Mining/Large Statistical Processes – use at least 128 MB per session/512 MB per CPU.

Required Disk Space – Allow enough disk space to hold all the data, estimated SAS WORK Spaces (see file system configuration below), and a safety/growth factor. All figures should allow for file system management overhead (e.g. Veritas Vxfs can use up to 15 percent depending on how the file systems are set up), and any disk redundancy plans (e.g., RAID 0+1 (mirroring – multiple copies), RAID 5 (parity,

can use up to 20 percent of the disk in overhead). Note on SAS work spaces: PROC SORT can require three copies of the data in work, other PROCs can create utility files in WORK (FREQ, TABULATE, LOG, etc.).

I/O Throughput Required – By using the application demand information, and knowing how many CPUs are required, you can create a general picture of I/O throughput demand based on how many concurrent processes are running against how much data read/write activity. Work with your systems administrator to set up the I/O storage subsystem for maximum delivery. Attention should be paid to striping across enough disks for performance, ensuring volume and file system setup and management that can handle the expected I/O traffic, and having enough controller and channel bandwidth.

A good example is setting up SAS workspace. For a large system with many users, or even a small system with users that have very heavy constant workloads, more than one SAS workspace should be allocated across different file system/volume/disk/controller paths. Then assign users across these multiple workspaces (even to the point of dynamically collecting I/O stats to assist in the assignment in very heavily used systems). SAS workspaces should never be set up in RAID 5 areas, to avoid the write overhead of the parity write. This type of activity assures that the disks, volumes, and file systems the workspaces reside on don't easily become overwhelmed.

SAS Tuning Parameters – These are the tunable parameters within SAS to govern how SAS allocates resources for I/O, memory, etc. These include parameters such as BUFNO, BUFSIZE, CATCACHE and COMPRESS for data access optimization, and MEMSIZE for regulating memory usage. Proper setting of these parameters can significantly boost performance. The discussion of the proper use of tuning parameters is platform specific, therefore view the white papers for each hardware partner by visiting the platform partner website at <http://www.sas.com/partners/directory/>, and select the platform partner of your choice, then chose white papers from that partners page.

System Configuration – This covers a broad range of topics including file system choice and setup, logical and physical volume management and setup, and setting up the file systems that

house the directory structures on the logical and physical systems. It is wise to pay attention to:

- File system behavior with memory usage, caching, buffering, and data transfer, and how to tune it
- Locating heavily used file systems on different disk controllers, ranks, or arrays to avoid I/O contention on disks, file systems, channels, etc.
- Locating SAS Work areas in RAID 0 space (not paying a write penalty for Work files)
- Placement of the journal logs for journaled file systems

There are many more things to consider in sizing and setting up systems for performance success. Help your systems and network administrators understand your processing needs, loads, and mixes so they can offer their best help.

When and Where to Get Expert Help

If you are confused by the diagnostic information you have gathered, cannot seem to pinpoint causes of problems, or solving one problem seems to cause another, it may be time to get some expert help. Earlier we talked about the necessity of using your Systems and Network Administrators to help interpret monitoring results, use host specific third-party tools, etc.

In addition, hardware vendor systems and support engineers are excellent resources for expert systems help. Check your support level agreement to see what level of service you have contracted for, and be aware that you may encounter fees depending on the situation.

References

SAS Customer Technology Center and SAS Partner White Papers:

<http://www.sas.com/partners/technology/>

Training Guide MCSE Covers Exam 70-068
Windows NT Server 4 Enterprise, Second Edition, Jason Sirockman, copyright 1998 New Riders Publishing

Mike Loukides, *System Performance Tuning*, O'Reilly and Associates, 1990

Adrian Cockcroft, Richard Pettit, *Sun Performance and Tuning, Java and the Internet*, Second Edition, Sun Microsystems Press/Prentice Hall, 1998.

For additional information:

Tony Brown
SAS Institute Inc., Dallas, TX
(214) 977-3916; <<mailto:tony.brown@sas.com>>

Copyright © 2001 SAS Institute Inc., Cary, NC, USA. All rights reserved. SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration. Other brand and product names are trademarks of their respective companies. SDM Rev. 2.4, Dec. 2000.