Operating System and Process Monitoring Tools

Arik Brooks, awb1@wustl.edu

Abstract:

Monitoring the performance of operating systems and processes is essential to debug processes and systems, effectively manage system resources, making system decisions, and evaluating and examining systems. These tools are primarily divided into two main categories: real time and log-based. Real time monitoring tools are concerned with measuring the current system state and provide up to date information about the system performance. Log-based monitoring tools record system performance information for post-processing and analysis and to find trends in the system performance. This paper presents a survey of the most commonly used tools for monitoring operating system and process performance in Windows- and Unix-based systems and describes the unique challenges of real time and log-based performance monitoring.

See Also:

Table of Contents:

1. Introduction

Computer operating systems monitor their resources constantly. In a system, processes are the main resource owners, and as such, most monitoring is done at the process level. This information is used by operating systems while they are running to perform effective memory management, scheduling, multiprogramming, and many other important decisions. In addition, performance monitoring is useful while developing and refining systems, and it provides user support during everyday operation. Records of operating system and process performance can be used to quantify changes to the system and allow accurate comparisons to other systems. They can also be used to predict the performance of similar systems and what type of performance gains may be expected in the future.

There are a large number of operating system and process monitoring tools available. This paper will present a survey of some of the most common tools used for operating system and process monitoring. These tools are primarily divided into two main categories: real time and log-based. Real time monitoring tools are concerned with measuring the current system state and provide up to date information about the system performance. Log-based monitoring tools record system performance information for post-processing and analysis and to find trends in the system performance. In addition to these categories, this survey will look at Windows-based tools and Unix-based tools, since the tools are very different for these two common operating system types.

Back to Table of Contents

2. Real Time Performance Monitoring Tools

Real Time Monitoring Tools are concerned with monitoring and displaying the current system performance. They sum up the performance for a particular factor with a single number. Typically, these tools rely on system calls that are built into the operating system to extract the performance readings. Because these calls are built into the operating system, they do affect the system performance, sometimes significantly. They are also very difficult to change since the operating system source code is not usually readily available. I will describe a number of the most common operating system and process performance monitoring tools for Windows and Unix systems in the following sections.

Back to Table of Contents

2.1 Windows-Based Tools

In this section, I will examine real time operating system and process monitoring tools for Windows systems.

Back to Table of Contents

2.1.1 Task Manager (taskmgr)

Task Manager is probably the most well-known tool for monitoring processes on the Windows operating system. Task Manager was introduced with Windows NT and provides a fast look into the current system state [Moore]. It shows all applications (one or more processes running within a single application context) and their state, all processes and some of their most frequently used performance measures, and some general system performance measurements. Newer versions also display networking performance measurements. All measurements are made by directly calling functions in the operating system to retrieve system counters [Moore].

In addition, Task Manager gives users the ability to control the system by affecting the running process. This is the function general computer users typically use Task Manager for when an application or process enters a "hung" state due to errors in the code and cannot be exited normally or when a process is hogging the CPU. In addition to ending a process, Task Manager also allows users to end a process tree killing all threads associated with the selected process, set a process's priority to reduce or increase its CPU consumption, and set a processes affinity to certain CPUs to control which CPUs a process will execute on in a multi-CPU system [Aubley01].

Two of Task Manager's tabs present operating system performance data to the user. The first is the Processes tab, shown in Figure 1. The Processes tab shows the current memory and percentage of CPU usage of every process running on the computer as well as the total CPU and memory usage of the system. While this is not a lot of information, it is a very good first indicator when a process is taking too much of the CPU or has a memory leak. It is easy to use and even allows sorting by name, user name, CPU usage, or memory usage.

The second Task Manager tab that presents operating system performance data is, not surprisingly, the Performance tab, shown in Figure 2. The Performance tab in Task Manager provides a top level view of the system state in terms of CPU and memory usage. A short history is shown on a graph, but once again, none of this data is logged for comprehensive analysis.

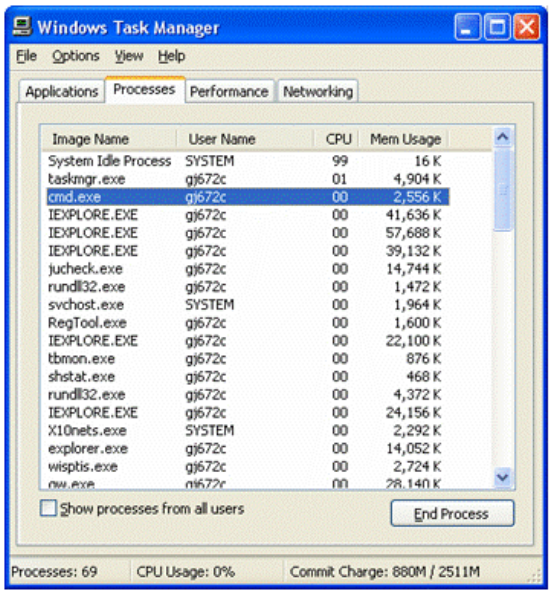

Figure 1: Task Manager - Processes Tab

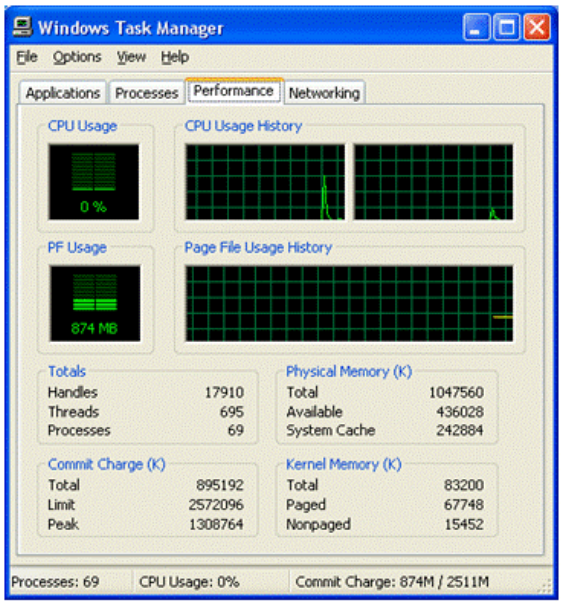

Figure 2: Task Manager - Performance Tab

Task Manager's data is updated every one second by default, but it can be changed to one of three preset values or set to only update manually. It is highly integrated into the operating system, and it is not designed to log any of its performance measures for performance analysis or system evaluation [Moore]. However, it is an invaluable tool in monitoring and adjusting the processes running on a computer. Task Manager is also the standard for real time operating system and process monitoring tools for Windows systems that all other tools must be evaluated against, because another tool must provide some features not available in Task Manager for it to be considered useful.

Back to Table of Contents

2.1.2 Performance Monitor (perfmon)

Performance Monitor is the second most common operating system performance monitoring tool for Windows. Performance Monitor acts as both a real time and log-based performance monitoring tool for operating systems, so only the real time portion of the tool will be discussed in detail in this section, and the logging portion will be discussed later.

Like Task Manager, Performance Monitor measures performance by making system calls to retrieve system counters, but Performance Monitor makes these calls via a performance library that also supports logging of the counters. Unlike Task Manager, Performance Monitor provides an interface to monitor any selection of a huge set of system counters on a graph in real time, rather than just the limited set Task Manager uses. Counters include things like percentage of processor time, thread count, page fault rate, memory size, and elapsed time for processes. Similarly, there are counters that provide state for threads, the processor, the system, network interfaces, memory, physical disks, and many others. This level of detailed information available for monitoring from Performance Monitor is very extensive and makes Performance Monitor ideal for monitoring resource usage and performance of almost all pieces of a Windows system.

The main window for Performance Monitor, shown in Figure 3, is a graph of all selected system counters updated in real time at a specified rate. Almost everything on this display is customizable from whether it is a graph or a histogram to the colors assigned to certain counters. The current reading and statistical information is displayed for the selected counter as the graph updates. Only one counter can be selected at a time for displaying numerical data, but any number of counters can be included on the graph at once.

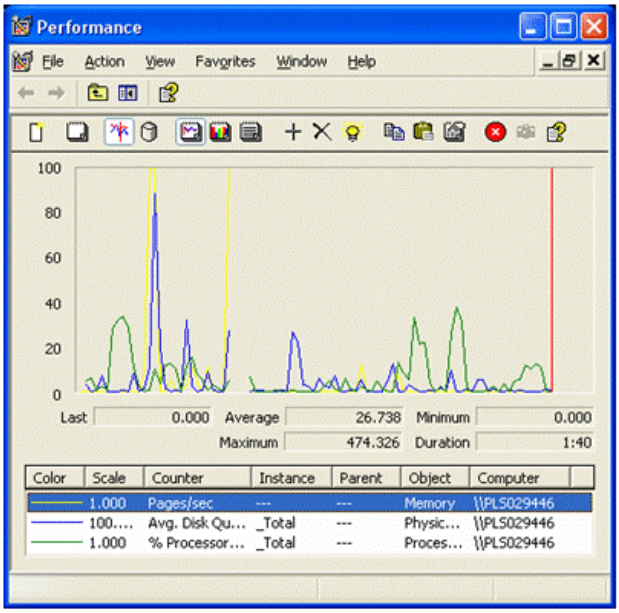

Figure 3: Performance Monitor - Main Window

The Add Counters window of Performance Monitor is shown in Figure 4. This example shows the counters and options available for a process. The list on the left lists all of the available counters for a process that can be selected for monitoring. The list on the right shows all instances of a process object that can be selected for monitoring. Performance objects that can be monitored, detailed in Figure 5, include many objects other than processes.

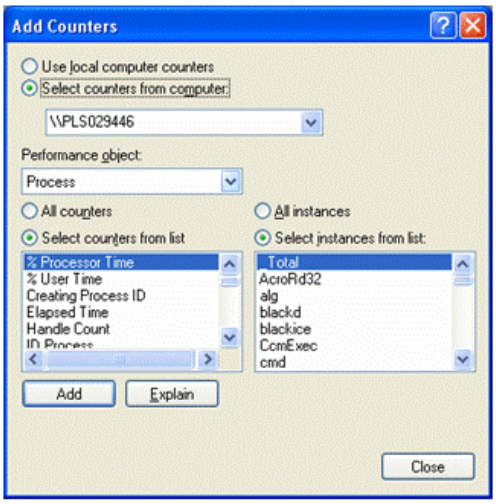

Figure 4: Performance Monitor - Counters

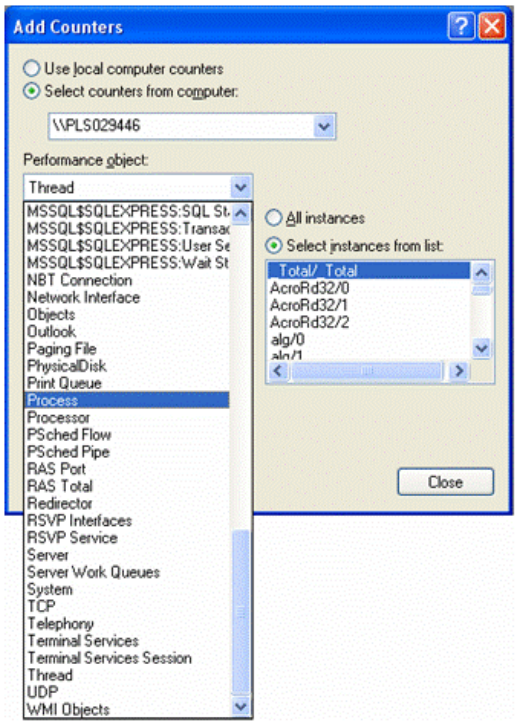

Figure 5: Performance Monitor - Performance Objects

In addition to detailed monitors of a large set of system counters, Performance Monitor also allows the user to set up alerts that will cause an action to occur when a specified counter exceeds a defined threshold [Aubley01]. Alerts can be set up to log an entry, send a message, or run any command or program whenever a counter exceeds a threshold set by the user.

Back to Table of Contents

2.1.3 Process Monitor (pmon)

Process Monitor is another tool for Windows process performance monitoring that is primarily useful for real-time monitoring of process performance. It provides functionality similar to that of Task Manager but without any control over processes. It simply outputs process performance measurements to the command prompt windows. Process Monitor is not very configurable and only has one update rate: every five seconds. Like Task Manager and Performance Monitor, Process Monitor gets its performance measures via the system counters.

The advantage of Process Monitor over Task Manager is more in user preference than anything else. Some users who are more comfortable with command line interfaces or who prefer the simplest view into the state of the processes running on a machine may prefer Process Monitor over Task Manager. However, there is not any data provided by Process Monitor that is not also displayed in Task Manager.

Back to Table of Contents

2.1.4 Process Explode (pview)

Process Explode is another operating system monitoring tools for Windows that is very similar to Performance Monitor. Like Performance Monitor, Process Explode provides a vast amount of performance data of processes, threads, memory, and the system in general. Most of the measurements are the same, but Process Explode provides access to a few additional measurements of these four system objects [Moore]. Like all of the performance measurement tools discussed thus far, measurements are made by getting counts for various system objects. Process Monitor shows details of a selected process's address space, its base priority, its elapsed time and how much time was spent in user and kernel modes, and thread and other system data.

The main differences of Process Explode compared to Performance Monitor are:

- Process Explode requires no setup to see all of the performance measurement information all on one window. However, this can be a little overwhelming upon initial use of this tool, since there is so much data displayed in a single window.
- Process Explode does not update automatically, but instead only updates when the user refreshes it manually.
- Process Explode does not include any alert or logging capabilities.
- Process Explode can change process priorities and permissions and can stop running processes like Task Manager.

The main differences of Process Explode compared to Task Manager are:

- Process Explode shows many more performance measurements than Task Manager.
- Process Explode cannot be used to get a quick summary of the CPU and memory usage of all running processes

Back to Table of Contents

2.1.5 Process Viewer (pviewer)

Process Viewer is another tool that was introduced with the Windows NT Resource Kit 4.0. Process Viewer is very similar to Process Explode, but it shows a subset of the process performance measurements shown by Process Explode [Moore]. This is a much easier to read interface than Process Explode's window, and still usually shows all of the details that a user would need. Like Process Explode, Process Viewer does not update its information automatically but instead only updates manually when a user presses the Refresh button. Also like Process Explode and Task Manager, Process Viewer can kill processes and view and change process priorities.

Process Viewer is most useful for examining process memory usage. It has a separate Memory Details window that shows all of the user address space counts for a process along with the virtual memory counts.

Back to Table of Contents

2.2 Unix-Based Tools

In this section, I will examine real time operating system and process monitoring tools for Unix systems.

Back to Table of Contents

2.2.1 Process Status (ps)

While Task Manager and Performance Monitor are probably the most well-known real-time performance monitoring tools for Windows systems, Process Status (ps) is almost certainly the most widely used performance monitoring tool for Unix based systems. The ps command provides detailed information and performance statistics about running processes such as the names and process IDs as well as the current CPU usage [Ezolt05]. The ps command is one of the oldest performance measurement tools for Unix systems that is still used widely today, and it has a large number of options and parameters that it can display.

Some of the parameters for the ps command that are most useful for process performance monitoring include:

- pcpu - Percentage of the CPU that the process is currently using
- etime - Elapsed time since the process started
- time - CPU time used by the process

Back to Table of Contents

2.2.2 Top

The other Unix tool that is almost as widely used as ps is its continuously updating counterpart called top. Top has been one of the most used tool for monitoring real time performance of processes in Unix systems ever since its introduction 22 years ago (1984) in BSD Unix 4.1.

Top produces a list of all the currently running processes listed in order of CPU usage. The top CPU users appear at the top of the list, leading to the name of this command. The list is continuously updated at five second intervals by default, and there are options to shorten or lengthen the update period [Fink02].

Top also provides some general system information and performance measures including:

- how long the system has been running
- what the average load has been
- a summary of the running processes and their states
- average percentages of CPU time in each state
- memory statistics
- swap statistics

The process information Top provides is presented in a table with a large number of possible fields including:

- PID - Process ID
- USER - User that started the process
- PRI - Process priority
- NI - Process nice value
- SIZE - Memory used by process (including code size)
- STATE - Current process state
- TIME - Total CPU time used by process
- %CPU - Percent CPU used by process
- %MEM - Percent memory used by process
- COMMAND - Process's name

All of these fields can be configured to show up in Top or not, and there are several other fields that can be shown as well such as the page fault count and the physical memory usage. In addition to displaying all of this information, Top also allows manipulation of processes in a limited interactive mode. In interactive mode processes can be terminated, their priorities can be changed, and the display can be configured for filtering and desired formatting.

Back to Table of Contents

2.2.3 Xosview

Xosview OS Monitor (xosview) is a graphical real time system performance monitoring tool for Unix systems that runs under the X Window System. It is very simplistic compared to the Windows tools described thus far because it does not monitor individual process, but instead only monitors the system as a whole. The display for xosview shows histograms of a number of useful system parameters including CPU usage split up by process type and the load average of processes [Fink02].

The complete list of system parameters that can be monitored includes load average, CPU usage, memory usage, swap space usage, page swapping, disk usage, and interrupt rate. Each histogram plot is color coded to represent different regions. For example, CPU usage is split into four regions: user, nice, system, and idle. Each of these shows up as a different colored bar representing what percentage of the total belongs to that region. Each measurement bar is called a meter, so the bar that represents CPU usage is referred to as the CPU meter.

Xosview is highly configurable, with options to turn on or off every meter and change other detailed settings like separately controlling the sample rate for each parameter. It is one of the easiest ways to quickly visualize the current system state in a graphical format.

Back to Table of Contents

2.2.4 Treeps

Treeps is a graphical real time process monitoring tool for Unix systems running the X Window System. The treeps application dynamically updates it measurements at an adaptive rate based on process activity, sampling data for inactive processes less frequently than active ones. Treeps displays all the fields from the procinfo structure including CPU time, process status, current memory used, and a number of static properties like the process name and Id. In addition to showing process status, treeps also displays process hierarchy, so it is easy to see process parents and children. All of the detailed information can be displayed by selecting a process from the hierarchy tree [MacDonald].

Treeps does allow limited process control similar to Task Manager for Windows. From treeps, processes can be killed or signaled. Treeps is for real time monitoring only, and it does not include any logging capabilities.

Back to Table of Contents

2.3 Summary of Real Time Monitoring Tools

Table 1 summarizes the real time performance monitoring tools described above including a summary of the features, pros, and cons of each tool. The five features listed in the table indicate whether or not each tool exhibits five common features of real time monitoring tools. The graphical user interface (GUI) column indicates whether the tool has a GUI or command line interface. The second column indicates whether the tool automatically updates its measurements at some sample rate or if it must be updated manually. The third column indicates whether the tool allows control over the processes it is monitoring, typically including the ability to exit the processes and change their priorities. The fourth column indicates whether or not the tool shows process performance measurements, and the fifth column indicates whether or not the tool shows system performance measurements.

	Tool	Features					Pros	Cons
		Graphical User Interface	Automatically Updates Measurements	Allows Process Control	Shows Process Performance	Shows System Performance		
Windows Tools	Task Manager (taskmgr)	X	X	X	X	X	• No setup required • Highly integrated into OS - low overhead	• Small set of measurements available
	Performance Monitor (perfmon)	X	X		X	X	• Huge set of available measurements that can be monitored • Supports alerts when user specified thresholds are exceeded	• Hard to sift through the many available counters • All measurements displayed on a single graph, regarles of how many are selected
	Process Monitor (pmon)		X		X		• Extremely simple to use • Appeals to those who like command line interfaces best	• Not very configurable
	Process Explode (pview)	X		X	X	X	• No setup required • All performance measurements for process and system displayed in a single window	• Display is crowded • No graphs
	Process Viewer (pviewer)	X		X	X	X	• No setup required • Has very useful Memory details window for processes	• Display is crowded (similar to pview)
UNIX Tools	Process Status (ps)				X		• Extremely simple to use • Supported by almost every Unix based system	• Very limitied set of performance measurements
	top		X	X	X	X	• Process info is sorted in a useful way (order of CPU usage) • Highly configurable (fields, sample rate can be adjusted)	• Difficult to change sort order
	xosview	X	X			X	• Histogram graphs provide a lot of information in a small graph • Very configurable	• Somewhat limitied set of performance measurements (no process measurements)
	treeps	X	X	X	X		• Process hierarchy easily viewed in tree form • Adapts sample rates based on process state (higher sample rate for active processes)	• No system performance information

Table 1: Summary of Real Time Performance Monitoring Tools

Back to Table of Contents

3. Log-Based Performance Monitoring Tools

There are another set of tools for monitoring operating systems and processes that follow a completely different model than that of the real time

performance monitoring tools described in the previous section. I will refer to this second set of tools as log-based performance monitoring tools for operating systems and processes. Log-based monitoring tools are used primarily for the same purposes as real-time monitors but to a different level of detail. Where real-time monitoring tools are used to quickly evaluate the systems status and identify resource utilization of processes, log-based monitoring tools are used to thoroughly evaluate these same things. With log-based monitoring tools, trends in the performance measures can be observed and identified.

In addition to supporting evaluation of system performance for optimization and debugging purposes, log-based performance monitoring tools can be very useful because of the detailed records of system performance measurements they produce. These records can be used for a number of purposes other than simple data analysis. Performance logs can be used to evaluate operating system improvements against a realistic and known performance measurement. They can also be used to predict performance improvement rates as better and better systems are measured and logged.

Most of the issues with log-based performance monitoring tools are similar to those of real time performance monitoring tools. It is still very difficult to create any performance monitoring tool that does not affect the system in some way while it is measuring the system. In addition, it is very difficult to select the best sampling rate for performing measurements because of the tradeoff between measurement accuracy and measurement overhead. One extra issue the log-based performance monitoring tools have to deal with that was not present in real time performance monitoring tool is file management. With log-based performance monitoring tools, measurements have to be written permanently to the disk. This is a high overhead operation, plus measurements that occur very frequently can quickly fill up the available disk space. As with real time performance monitoring tools, I will describe a number of the most common log-based tools for Windows and Unix systems in the following sections.

Back to Table of Contents

3.1 Windows-Based Tools

In this section, I will examine log-based operating system and process monitoring tools for Windows systems.

Back to Table of Contents

3.1.1 Event Log Service and Event Viewer

The most commonly used log-based monitoring tool in Windows is the Event Log Service. The Event Log Service in Windows records all application, security, and system events into logs. Event Viewer can be used to view the contents of these logs to help identify the cause of system problems or evaluate system performance to predict future problems. This is not limited to performance measurements, and most events logged in the Event Viewer logs have little to do with performance.

The application log contains events logged by applications. Therefore, the types of events that are logged in this log are purely up to the authors of the applications that are running. The security log is not of interest in this discussion. The system log contains events logged by Windows components, usually when there is a failure or warning to report. This information is not typically very useful for measuring or monitoring performance of the operating system or processes, but it is a good place to turn first when the operating system or a process is misbehaving and you want to see if it logged any events that might help determine why it failed.

Back to Table of Contents

3.1.2 Performance Logs and Alerts

The most commonly used log-based tool in Windows specifically for performance monitoring is the Performance Logs and Alerts segment of the Performance Monitor tool described above in section 2.1.2. To allow the user to manage the required disk space and data resolution, Performance Logs and Alerts allows detailed management of the time to log data during, the sampling rate to log data at, and the selection of data to log.

When a log is set up, the first thing a user must select is what data they want to record. This can include any selection of system counters available in Performance Monitor as well as a selection of other system parameters or from sources external to Performance Logs and Alerts. Next, a sampling rate is selected based on the data resolution that is needed and the disk space that is available. For debugging a problem or for detailed analysis, a high sampling rate may be required, while a very low sampling rate may be perfectly adequate for long-term data analysis. According to [Aubley01], logging nine key counters at a rate of one sample per second will consume about 720,000 kb per hour, one sample every five seconds will consume about 144,000 kb per hour, and one sample every ten minutes will consume 1,200 kb per hour.

After that, a logging schedule is selected. A start and stop time are selected and a command can be scheduled to run occur when the log is stopped. Some useful selection techniques are included such as selecting a logging duration instead of a stop time and the option to keep logging until there is no more space. A common selection is to log data during peak usage times such as 8:00 am to 6:00 pm [Aubley01]. Finally, a log name and type are selected. Logs can be saved to comma or tab separated text files that can be viewed in Microsoft Excel, or they can be saved to an SQL Database, or they can be saved as binary files viewable in Performance Monitor. Binary files can also be circular, recording data until a predefined size is reached and then starting over. The result of all this is a log file of performance data that can be analyzed and post-processed to examine system performance.

There are two types of logs that can be recorded and viewed using Performance Logs and Alerts: counter logs and trace logs. Counter logs are simply logs of a selection of system counters measured and recorded at a specific sampling rate. There are hundreds of counters to choose from, and they can measure everything from physical disk transfers per second to processor time consumed by a specific thread. Trace logs collect traces of events into a log. There is no sampling rate for logging events, so trace logs log the events from start to finish. Examples of events that might be logged include process creations or deletions, disk I/O, or page faults. Other data providers can be selected as sources of events for logging in data logs, making Performance Logs and Alerts somewhat extensible.

Back to Table of Contents

3.1.3 Performance Data Log Service (PerfLog)

Performance Data Log Service is another new tool that logs performance data from system counters to a log file. Performance Data Log Service produces comma or tab separated log files. Like Performance Logs and Alerts, it lets the user select which performance counters to log and the rate to log them at.

The features that are different with Performance Data Log Service have to do with its scheduling interface and automation and its log format. Performance Data Log Service automatically logs directly to comma or tab separated for easy import into almost any program used to view logs. The latest version of Performance Logs and Alerts can log directly to TSV or CSV files as well, but Performance Data Log Service could do it first. Performance Data Log Service also allows for a number of useful scheduling and automation options. It can be set to start monitoring on system boot up and can start prior to a user logging on. It can also be set up to use a base file name and append a date or automatically incremented number on the end to avoid user interaction or produce logs at a set frequency. Performance Data Log Service can also be set to start a new log file based on a set frequency in hours, days, or months or based on a file size in kilobytes or megabytes. In addition, Performance Data Log Service has a number of preset counters that can be selected by choosing a desired level of detail for monitoring. This allows novices at performance logging to select a reasonable data set for logging and provides a good starting point for all users when selecting which counters should be recorded.

Back to Table of Contents

3.2 Unix-Based Tools

In this section, I will examine log-based operating system and process monitoring tools for Unix systems.

Back to Table of Contents

3.2.1 System Activity Reporter (sar)

System Activity Reporter, usually referred to as sar, is another one of the most commonly used performance monitoring tools for Unix. Sar can be run as either a real time tool requiring immediate user interaction or as a log-based tool for detailed performance analysis [Musumeci02]. I classify it as a log-based tool because its real time usage is no different than most of the other tools described earlier in that it simply reports various system activity counters at the command prompt. In its log-based form, sar samples the same counters at a specified rate and stores the measurements to a binary file. There are a number of options that can be used to select which counters are sampled to reduce the amount of data that is stored including one for CPU utilization, system calls, memory allocation, buffer activity, paging activity, and many more.

The man pages for sar recommend using a sampling period of five seconds or longer to avoid having sar interfere noticeably with its own measurements. Sar has no built in support for scheduling logging or stopping based on available memory. Also, like most of the tools described so far, sar does not include any analysis capabilities. It simply logs huge amounts of data to disk, and users are responsible for making sense of interpreting the data to determine the system performance.

The sar command is also used to view the log files it generates. It can read and display the contents of the binary log files it generates in the same way that it displays real time measurements when run interactively. To avoid displaying too much information, flags can be used to specify what time period from the log the user would like to view.

Back to Table of Contents

3.2.2 Cpustat

Another log-based performance monitoring tool for Unix is cpustat. Cpustat monitors system performance by recording the system's performance counters. Like sar, cpustat can be used in either real time mode or log-based mode. Also like sar, the sampling rate can be specified depending on the resolution of counter measurements needed and the amount of disk space available. Unlike sar, cpustat does not allow the user to specify which counters to record. However, cpustat is already limited compared to sar because it is only concerned with the CPU performance counters while sar is able to record many other system counters.

The output of cpustat is not logged to a file by default, but it can be redirected to a file, and it is formatted so that it is easily parseable by a scripting language such as Perl. Nominally, cpustat samples the CPU performance counters every 5 seconds and runs indefinitely once the user starts it. The only modifications that can be made are adjustment of the sampling rate and limiting cpustat's lifetime to a certain number of samples.

Back to Table of Contents

3.3 Summary of Log-Based Monitoring Tools

Table 2 summarizes the log-based performance monitoring tools described above including a summary of the features, pros, and cons of each tool. The five features listed in the table indicate whether or not each tool has five common features of real time monitoring tools. The graphical user interface column indicates whether the tool has a GUI or command line interface. The second column indicates whether the tool allows adjustment of the sample rate to log data at. The third column indicates whether the tool allows filtering of what data is logged or whether the data that is logged is predetermined. The fourth column indicates whether or not the tool allows scheduling of when to start and stop logging, and the fifth column indicates whether or not the tool produces logs that are formatted in a way that is compatible with other standard tools for viewing data.

	Tool	Features					Pros	Cons
		Graphical User Interface	Allows Sample Rate Selection	Allows Data Filtering	Allows Log Scheduling	Produces Compatible Logs		
Windows Tools	Event Log Service and Event Viewer	X					▪ No setup required ▪ Always running	▪ No performance data is logged by default ▪ Very limited data, and no high rate data
	Performance Logs and Alerts (part of Performance Monitor)	X	X	X	X	X	▪ Huge set of available measurements that can be monitored ▪ Very configurable	▪ Hard to sift through the many available counters ▪ Easy to produce very large files
	Performance Data Log Service	X	X	X	X	X	▪ Huge set of available measurements that can be monitored ▪ Very sophisticated scheduling features ▪ Has easy to use preset data filtering selections	▪ Hard to sift through the many available counters ▪ Easy to produce very large files
UNIX Tools	System Activity Reporter (sar)		X	X			▪ Extremely simple to use ▪ Supported by almost every Unix based system ▪ Has easy to use preset data filtering selections	▪ Binary logs not compatible with other programs (must be decoded using sar) ▪ Hard to view large logs
	cpustat		X			X	▪ Very simple to use ▪ Output easily parseable by a scripting language like Perl ▪ Can specify the number of samples to make	▪ Output not automatically logged to a file ▪ Does not log process performance measurements

Table 2: Summary of Log-Based Performance Monitoring Tools

Back to Table of Contents

4. Summary

Performance monitoring of operating systems and processes is essential for debugging processes and systems, effectively manage system resources, making system decisions, and evaluating and examining systems. There are two primary types of performance monitoring tools: those for real time performance monitoring and those for logging performance measurements for future perusal.

Real time performance monitoring tools for operating systems and processes are typically concerned with presenting the performance measurements to the user in a way that is well organized and easy to use. The main issue with real time performance monitoring tools is in data resolution and presentation. The tools try to select all data that might be useful without presenting so much data that it is difficult to find what is needed. The most common tools used to perform real time performance monitoring include Task Manager and Performance Monitor for Windows-based systems and ps and top for Unix-based systems.

Log-based performance monitoring tools for operating systems and processes are most concerned with balancing the conflicting goals of storing high resolution measurements with consuming the least amount of disk space and making the logs the easiest to view. The tools try to address

these goals by allowing the user to filter the measurements to only record the information they are concerned with and to specify the sampling rate to use based on the resolution needs and disk space available. Also, the tools try to produce simple outputs that are highly compatible with various post-processing tools so that the logs are easy to use. Additional features like easy automation separate the best tools from others. Some of the most common tool for performing log-based performance monitoring of operating systems and processes include Performance Logs and Alerts (part of Performance Monitor) and Performance Data Log Service for Windows systems and sar for Unix-base systems.

Back to Table of Contents

References

The following references are arranged in order of usefulness and relevance to this paper.

1. [Moore] Moore, Sonia Marie (Ed.). "MS Windows NT Workstation 4.0 Resource Guide." 1995. http://www.microsoft.com/technet/archive/ntwrkstn/reskit/default.mspx.

   This is a comprehensive resource kit for Windows NT 4.0 for Workstations. Part III of the kit talks about optimizing the operating system, including chapters on the art of performance monitoring, performance monitoring tools, and general information about performance monitoring on Windows NT systems.

2. [Aubley01] Aubley, Curt. "Windows 2000 Performance Tools: Leverage Native Tools for Performance Monitoring and Tuning." Prentice Hall PTR, 2001. http://www.informit.com/articles/article.asp?p=20478&rl=1.

   The author describes the latest updates to Task Manager and Performance Monitor, the two most used performance monitoring tools developed for Windows NT. The article includes a comparison of the two tools, detailed information about the tools, and tips for using them in the real world.

3. [Fink02] Fink, Jason R. and Matthew D. Sherer. "Linux Performance Tuning and Capacity Planning." Sams, 2002.

   The authors of this book present a theoretical overview of performance tuning, and they examine a number of Unix/Linux performance tuning tools.

4. [Ezolt05] Ezolt, Phillip G. "Optimizing Linux(R) Performance: A Hands-On Guide to Linux(R) Performance Tools." Prentice Hall, 2005.

   The author presents a comprehensive guide to Linux performance improvement. Actual examples are included, and the book teaches how to determine performance bottlenecks and select performance tools.

5. [Musumeci02] Musumeci, Gian-Paolo D. and Mike Loukides (Ed). "System Performance Tuning." O'Reilly Media, 2002 (2nd ed.)

   This book is written as an introduction to performance tuning and discusses capacity planning and, performance monitoring. The authors focus on how to perform effective performance tuning with the standard software tools available for Unix-based systems.

6. [MacDonald] MacDonald, George. "Unix Process Monitor Programs" http://www.orbit2orbit.com/gmd/tps/other/monitors.html

   This is a web page where the author lists many process monitoring tools for Unix systems along with a brief description of each.

Back to Table of Contents

List of Acronyms/Abbreviations

CSV	Comma-Separated Variable
GUI	Graphical User Interface
kb	kilobytes
man	Manual
perfmon	Performance Monitor
pmon	Process Monitor
ps	Process Status
pview	Process Explode
pviewer	Process Viewer
sar	System Activity Reporter
SQL	Structure Query Language

taskmgr	Task Manager
TSV	Tab-Separated Variable

Back to Table of Contents

This report is available on-line at http://www.cse.wustl.edu/~jain/cse567-06/.htm
List of other reports in this series
Back to Raj Jain's home page