11.2.1 CPU Resource Monitoring

Click one of the buttons to take you to that part of the video.

Resource Monitoring 0:00-0:17

In this video, we're going to show you a few methods and tools that can be used to help troubleshoot and diagnose Linux CPU, memory, and hardware issues.

Since we could devote an entire course to troubleshooting Linux issues, this video will only cover a few key topics.

System Uptime and Load Average 0:18-2:21

To begin, let's first look at a command used to determine how long a system has been running and what the system's load averages are.

This command is uptime.

Uptime is a very simple command with only a few options.

When run without any options, uptime displays several pieces of default information.

The first piece of information is the current time of day.

This is followed by the word up and the amount of time the system has been running. This value is shown in both days and hours.

Next, you see the number of users currently logged in.

And the last bit of information is the load average for the last 1 minute, 5 minutes, and 15 minutes.

A system's load average is the average number of processes that are in either a runnable or uninterruptable state.

In other words, the number represents a percentage of the full CPU load.

For example, if the number is zero, there was no load on the CPU for the specified time period. And if the number is 1, then there was a full load on the CPU.

Any number greater than 1 indicates that the CPU has been asked to do more than it can in real time, and some tasks will have to wait for CPU time.

For example, as we look at the results from the uptime command shown here, we see the values of 1, .40, and 1.70.

This means that for the last 1 minute, the average CPU load was 100%.

For the last 5 minutes, the average utilization was 40%, which means that that CPU was idle for 60% of the time.

And finally, with the value of 1.70 found in the last field, we know that the average CPU load for the last 15 minutes was 170%, meaning that that 70% of the current processes had to be forced to wait for CPU time.

You can also change the output of uptime using switches.

For example, if you run uptime with -p, the output is simplified to only show how long the system has been running.

The only other switch you'll normally use is the -s switch.

With this switch, you can easily see the date and time the server was started.

The first value is the year, month, and date, and the last value is in hours, minutes, and seconds.

CPU Monitoring – /proc/cpuinfo 2:22-4:19

Now that you know how to determine how long your system has been running and find metrics about its load average, let's turn our attention to a few methods you can use to monitor the CPU.

One of the simplest methods for viewing the details about the individual CPU cores is to look at the contents of the /proc/cpuinfo file.

By viewing this file, you can gain information about things like the processor ID number. Keep in mind that each CPU will have a unique number and that the first CPU will have an ID of zero.

You'll also see the CPU family, which tells you the type of processor you have in the system.

For example, if your computer is an Intel-based system, simply place the number in front of "86" to determine the value. This is particularly helpful if you're attempting to identify the architecture of an older system, such as a 586, 486, or 386 system, because some RPM packages are compiled for each of these particular architectures. This value also helps users determine which packages to install.

The model name gives you the common name of the processor, including the project name.

The CPU MHz shows the processor's precise speed (in megahertz) to the thousandth decimal point.

The cache size displays the amount of level 2 memory cache available to the processor.

And the flags define a number of different processor attributes, such as the presence of a floating-point unit (FPU) and the ability to process MMX instructions.

Since this file can contain a lot of information, it's helpful to display the information one page at a times using less or a similar command.

You can also filter out or find specific information about things like the number of processors or cores the system has with the aid of the grep command.

For example, this command (cat /proc/cpuinfo | grep processor) lists all the lines containing the word processor.

To make this even simpler, you can pipe this result to the wc command with the dash lowercase-L switch (-l), which will add up the number of lines and give you a single number.

CPU Monitoring – sar 4:20-7:23

Another useful tool is the sar command. SAR is an acronym for System Activity Report.

It's an important tool that helps you get an overview of the computer and gives you important status information or metrics at different points of time for the CPU, memory, input, and output.

The cron job scheduler gathers the information sar displays.

For example, many Linux distributions have a sar cron file named sysstat in the /etc/cron directory.

If you look at this file, you see that every 10 minutes, the sa1 script is run, and then a daily summary of the accounting process is collected at 23:53 military time.

Now, when you run sar, you are shown the information collected by these cron jobs.

As we look at this output, we see some basic information about the date and the number of CPUs, and we also see specific information for several other categories.

For example, in the %user column, we see the percentage of CPU utilization that occurred while executing at the user level.

In Linux, a user can change the priority of a process by changing the nice value. This column shows the percentage of CPU utilization that occurred while executing at the user level with nice priority.

The %system column shows the percentage of CPU utilization that occurred while executing at the system level (or kernel). Note that this field does not include time spent servicing hardware or software interruptions.

The %iowait column shows the time the processor spent waiting for the input or output devices. This is idle time, when nothing could be scheduled.

The %steal column shows the amount of time a virtualized CPU spent while the hypervisor was servicing another virtual processor.

And finally, the %idle column shows the percentage of time that the CPU or CPUs were idle and the system didn't have an outstanding disk I/O request.

By default, sar stores all of its binary data in the /var/log/sa/ directory.

If you prefer to extend or shorten how long this history is saved, you can edit the /etc/sysconfig/sysstat file. By default, the history is typically stored for 28 days.

Keep in mind that if your history goes beyond 28 days, a new directory will be created as needed, one for each month.

If you want to view the sar information for a specific day, you can enter ‘sar', the desired metric flag, ‘-f /var/log/sa/', and then the filename.

In this command, the -f flag denotes that we'll be reading input from a file followed by the filename. The metric flag specifies the metric whose information we'd like to access.

For example, sar -u -f /var/log/sa/sa04 will display the recorded CPU utilization information for the fourth day of the month.

If you view your man pages for sar, you will find there are many other ways to use sar to find just the information you're looking for. For example, -R reports memory statistics, and -s sets the starting time of the data.

CPU Monitoring – sysctl 7:24-8:20

The last utility I want to show you is named sysctl.

This powerful Linux command lets you configure the kernel parameters at runtime.

To view all of the available parameters, run ‘sysctl -a' as root.

You can change these parameters, and the change will take effect immediately. You can write the change permanently using the -w switch.

Keep in mind that there may not be any parameters that directly correlate to CPU usage. But changing some parameters may have a direct impact on the CPU's performance.

For example, tuning the network adapter by disabling the TCP timestamps option can result in better CPU utilization.

sysctl is really a powerful tool, so use it with care. There are a lot of kernel parameters that can really mess your system up if you make poor changes.

Be sure to look at the man pages for sysctl before you start working with it, and make sure you understand all the effects of changing a variable before you use sysctl to modify it.

Summary 8:21-8:42

That's it for this lesson.

In this lesson, we showed you how to monitor and analyze your CPU using several utilities.

First, we discussed how to view your system's up times using the uptime command.

Then we talked about how to monitor CPU statistics by viewing the /proc/cpuinfo file.

We also reviewed how to monitor CPU statistics using the sar command.

And finally, we showed you how to configure Linux kernel parameters at runtime using the sysctl command.
