10.1.4 Process Display

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

Process Display 0:00-0:17

In this lesson, we're going to look at the various tools that you can use from the command line of a Linux system to view information about all the various processes that are running on that system. One of my favorite utilities to do this with is the top utility. Let's take a look at how it works.

---

top Command 0:18-4:07

Now, you run top simply by entering the top command--all lowercase--at the shell prompt. When you do, an interface similar to the one you see here will be displayed. As you can see, top displays a list of some of the running processes on your system--one process on each line.

As you can see here, several different columns are used to display information about each of these running processes.

The PID column right here displays the process ID number of that process. The USER column displays the name of the user that owns that process. As you can see here, most of these are owned by root. There's one process that's owned by the rtracy user.

The PR column identifies the priority assigned to that process. As you can see, most of these are running with the same priority level of 20, except for this one right here that's running with a lower priority number of 0, which actually means it has a higher priority on the system.

The NI column displays the nice value of the process, which influences the priority level. We'll look at how that works later on. The VIRT column displays the amount of virtual memory that's being used by the process.

The RES column displays the amount of physical RAM the process is using in kilobytes. That's its resident size and memory. The SHR column lists the amount of shared memory that's being used by the process. The S column is actually a very important one because it displays the status of the process.

There are actually several different values that could possibly be displayed in this column. A value of D indicates that the process is uninterruptedly sleeping. A value of R in this column indicates the process is running. A value of S indicates that it's sleeping. A value of T indicates that it is currently stopped.

We call that being traced. Z indicates that the process is zombied. You're probably asking, "What on earth is a zombied process?" A zombied process is one where the process itself has finished executing and has exited, but somehow that process's parent process never got notified that the child process was done.

Therefore, the parent process hasn't released the child process's process ID number. Now, a zombied process may eventually clear up by itself but if it doesn't, you may have to go in and manually kill and then restart the parent process that just won't let go of its child.

The percent CPU column identifies the percentage of CPU time that's being consumed by that process. Percent MEM specifies the percentage of available physical RAM that's being used by this process. The TIME column displays the total amount of CPU time the process has consumed since it was started.

Finally, the COMMAND column indicates the name of the command that was originally entered at the shell prompt in order to start this process. For example, the first process in the output of top is the process being used by the top utility itself. We know this because we see that the command that was used to start this process was top.

The main thing I like about the top utility is the fact that it's dynamic in nature. It's continually updating the display with the latest information about the processes running on the system.

You'll see that when you run top, the information changes constantly because one process might start using more of the CPU time, then it uses less. Another process might request extra memory, and so on.

The thing that I don't like about top is the fact that it shows only a very limited number of processes. They'll be many times when I need to see everything that's running on my system. In this situation, top just doesn't cut it. Instead, you may need to use the ps utility.

---

ps Command 4:08-10:52

The ps utility is also used to display information about running processes on your system.

It's different than top. Instead of using a dynamic display that's always updated with the latest information, the ps utility instead displays a snap shot of the current state of the processes running at that time. Basically, you get more information with ps, but the information is not dynamically updated like it is with top.

If you entered just the ps command at the shell prompt, it will view the processes that are associated with the current shell session. In this example, I ran the ps command and we see that there's a process for the su command. This is displayed because the su utility was used inside the shell to switch to the root user account.

There's also a process for the bash session itself right here. There's also a process named ps. That's because we use the ps process to list the current processes. Therefore, its process is listed in the output as well.

Notice that four columns of information are displayed by default with ps. The first one lists the process ID number of the process. The next one is labeled TTY, and this contains the name of the shell session that the process is running within.

The TIME column displays the amount of CPU time used by the process. Then finally, the CMD column displays the name of the command itself that was entered at the shell prompt to create the given process.

Notice here that only three processes were actually listed in the output of the ps command. There are actually many, many processes running on this system.

Why didn't they show up in the list? Well, this happens because by default ps shows only the processes associated with the current shell session. Therefore, only the su and ps processes are displayed, along with the bash shell process itself.

If you want to see all processes on the system, then you have to use some options with the ps command. The best one is the -e option. The -e option results in many, many more processes being displayed in the output of the ps command.

You might notice here that most of these processes, in fact all of them that you can see here--the actual list was way longer than this--most of the one's you can see here have a question mark in the TTY column.

This indicates that these processes are system processes. They're daemons. Remember that a daemon is loaded by the init or systemd process that start up, and therefore are not associated with any particular shell session. This is indicated by the question mark at the output of the TTY column, because that process isn't associated with any shell session.

Now, you might notice in the output here of the ps command, that the amount of detailed displayed is pretty minimal compared to what we saw with top. This also can be fixed.

One way to do that is to use the -f option with the ps command. The -f option causes the ps command to display many more details about the processes running on the system. In fact, it's frequently used in conjunction with the -e option.

In this example, I ran ps -ef at the shell prompt. This causes the ps command to display all of the processes running on the system and to display additional detail about these processes. Notice with the -f option, that several new columns have been added to the output.

At first we have the UID column, which displays the user name of the processes owner. This is the user that ran the command that created the process. Next, we have the PPID column, which displays the process ID number of that process's parent process.

We have the C column, which displays the amount of processor time utilized by the process. We've also added the STIME column, which indicates the time that the process started.

If you really want to crank things up, you can also add the -l option to the ps command. The -l option displays the long format of the ps output.

Notice in this example that I ran the ps command with all three options that we've talked about thus far: -e, -l, and -f (-elf). It's really easy to remember because just remember ps -elf. With the -l option, we have additional columns added to the output of the ps command to display additional information about processes running on the system.

We have the F column right here. These are the flags associated with the process. This column uses the code shown here. A value of one indicates that the process forked but didn't execute. A process of four indicates that it used root privileges.

We also have the S column right here. The S column indicates the state of the process. It uses the same code we saw with the top command. We've also added the PRI column. This is the priority of the process.

Notice we're looking at a lot of the same information we saw on top now. We have the NI column, which displays the nice value of the process. We have the ADDR column, which displays the memory address of the process.

We have the SZ column, which indicates the size of the process and we have the WCHAN column, which is the name of the kernel process in which the process is sleeping. If the process isn't sleeping but currently running, you'll see a dash (-) in this column.

Part of managing processes on a Linux system is knowing how much memory has been used by the processes running on the system and how much is still available. As we looked at earlier, you can use the output of top or ps to view this type of information, but you can also do this but using the free command.

The free command displays three critical pieces of information. It displays how much memory is free on the system. It displays how much memory has been allocated on the system. It also displays the usage of swap memory on the system. Notice here I've used the -m and -t (-mt) options with free.

The -m option displays memory statistics in terms of megabytes so it's human readable. The -t option displays totals for each category of information. The ps command is really useful for viewing information about processes running on your system.

But as we just saw, sometimes the output of ps can be overwhelming, especially if you're just looking for information about one or two specific processes. There is another way to do it, however, and that's to use the pgrep command instead.

---

pgrep Command 10:53-12:11

As its name implies, pgrep combines the functionality of ps and the grep commands together into a single utility. When you run pgrep at the shell prompt, you specify certain selection criteria that you want to look for. Then the command will search through all the currently running processes and output a list that matches just the criteria that you specified.

Now, there's several different options listed here that you can run with pgrep in order to create your selection criteria. One option is to use -P followed by a particular parent process ID number. This will cause pgrep to match on the specified parent process ID.

You can used -f to tell pgrep to match on a specific process name. So you enter -f followed by a name. You can also use the -u option to find processes owned by a particular user.

You'd enter -u followed by a particular user name, which is what we did up here. We went pgrep -u rtracy to display just the processes owned by the rtracy user. We can also use the -l option to display the process name.

By default, pgrep only displays the processes ID number. In this example, I ran pgrep -l so we see the second column here of process names, and I only want to see those that are owned by the rtracy user.

Summary 12:12-12:25

That's it for this lesson. In this lesson, we reviewed the various tools you can use from the shell prompt to view the information about the processes running on a Linux system. We first looked at the top utility. We then looked at the ps utility. We looked at free, and we ended this lesson by talking about pgrep.
