10.1 Process States

Process State	Process	Description
Running	**R**	The process is either running (it is the current process in the system) or waiting for the CPU to process it.
Stopped	**T**	The process has been stopped or suspended.
Waiting - Sleep	**S**	Interruptible sleep (waiting for an event to complete).
Waiting - Uninterruptible sleep	**D**	Uninterruptible sleep. This is a process that cannot be killed or interrupted with a signal.
Zombie	**Z**	The process has finished executing and exited, but the process' parent didn't get notified that the child process was finished and hasn't released the child process' PID number.

10.1 Viewing Processes

Command	Description
top	Returns Process ID (PID), uptime, load, CPU status, memory, and priority information for processes. Is useful in situations where you need to monitor processes continuously. **h** to display the help screen. **f** to add or remove columns from the chart. **F** to show a list of sortable columns, then press the key of the letter next to the column to be sorted. **u** to specify processes for a specific user. *You can use the -u option with top to display only those processes owned by a particular user.*
ps	Displays a snapshot of currently running processes in ascending order based on the PID. By default, the ps command displays the following information: • PID • Name of the shell session where the process is running (TTY) • CPU time the process has used (TIME) • The command used to invoke the process (CMD) Be aware of the following ps options: **-A** shows all processes. **-e** " " **-a** shows processes owned by other users and attached to a terminal (e.g., foreground processes). **-f** shows detailed information for processes. **-u** shows processes by user ID. **-l** shows the processes in long format and the process state (under the STAT column). The process states include: ◦ sleeping (S) ◦ running (r) ◦ traced (t) by another process ◦ zombie (Z) **-x** shows processes that are not attached to A terminal. Use this option to view daemon processes that begin during system boot.
pgrep	The pgrep command includes many options. The following are a few of the more useful options: **-f** searches for A specific process name. **-p** only match processes whose parent process ID is listed. **-u** only match processes whose effective user ID is listed (the user that owns the process).

10.2 Process Management

Command	Function
command &	Starts a process in the background, leaving the shell available for other commands. When running a process in the background, the shell displays the following information: • The job ID in brackets • The process ID (PID)
jobs	Views background jobs and their associated job ID numbers. The job ID number is specific to the terminal session. Each opened terminal has its own set of jobs and job ID numbers. Jobs from one terminal cannot be managed from a second terminal using job ID numbers. jobs' options function as follows: **-c** prints the command name for each process in jobs. **-g** only print the group id of each job. **-l** only the last job to be started is printed. **-p** print the process id for each process in all jobs.
bg	Sends a job to the background.
fg	Brings a job to the foreground.
Ctrl+z	Pauses a running process and sends it to the background. A job ID number is assigned to the process.
Ctrl+D	Sends the end-of-file (EOF) marker to bash, which closes bash. Frequently used to indicate no more input should be read.

10.2 Adjusting Process Priorities

Command	Function
nice	Starts a process and customizes its priority on the system by manipulating the process' nice value. Be aware of the following: • Each process running on a Linux system has a priority (PR) and nice (NI) value associated with it. ◦ The PR value represents the process' kernel priority. The higher the PR value, the lower the priority of the process. The lower the PR value, the higher the process' priority. ◦ The NI value is factored into the kernel calculations that determine the PR value of a process. Changing the NI value associated with a process also changes its PR value. • NI values range from 19 (lowest priority) to -20 (highest priority). The higher the NI value, the lower the process' priority. The lower the NI value, the higher the process' priority. • Use -n to specify the priority value. If no value is specified, the process starts with a nice value of 10 by default. • Zero (0) is the default nice value for processes not executed with the nice command. To keep users from abusing the nice command, standard users are not allowed to specify a nice value less than 0. Only root can assign a nice value that is a negative number.
renice	Assigns a new nice value to a running process using the PID of the process. The command can contain multiple PIDs separated by a space to give them all the same priority number. The nice values assigned are identical to those used with the nice command. You can use several options with renice: **-n** specifies a priority. **-u** specifies a user. **-g** specifies a group.

10.2 Keep Running a Process After Logging Out

Command	Function
nohup &	Allows a command or shell script to continue running in the background after logging out from a shell. nohup does not automatically put the command it runs in the background; use the ampersand (&) symbol to start a process in the background.
screen	Uses multiple shell windows from within a single SSH session. Using screen, you can: • Keep processes running while you access the shell prompt through an SSH connection to enter additional commands. • Keep an SSH shell active even if the network connection is closed or goes down. • Disconnect and reconnect to a shell session from multiple locations without having to stop and then restart whatever processes were running. The screen package must be installed on the system before it can be used. After screen is installed, launch it by entering screen at the shell prompt. A shell prompt is displayed within a window inside the screen. Each window functions like a normal shell session. Within screen, pressing Ctrl+a causes whatever is typed by the user to be sent to the screen process instead of to the shell: **Ctrl+a ?** causes the screen help to be displayed. **Ctrl+a c** causes a new screen window to be created. The old window remains active along with any processes that were running within it. **Ctrl+a n** toggles between open windows in the screen. **Ctrl+a d** detaches the screen window and returns the user to the original shell prompt. Whatever was running in the window remains running. In fact, the user can completely log out, and everything will keep running within the detached window. **-r** reattaches a detached screen window. If multiple detached screen windows exit, the user will be prompted to specify which one to reattach to.

10.3 at Daemon Commands

Command	Used To
at *time date*	Schedules the command to run at a specific time and date. Options and syntax include: • today • tomorrow • month # • MMDDYY • MM/DD/YY • DD.MM.YY
at *time_of_day*	Uses time-of-day keywords to run the command. Options are: • Midnight (12:00 AM) • Noon (12:00 PM) • Teatime (4:00 PM)
at now	Runs the command immediately.
at now + ***number time_period***	Schedules the command to run at the designated time in the future. Use: • minutes • hours • days • months
at -f *filename time*	Schedules the tasks listed in the specified file to run at the designated time.
at -l **atq**	Lists the tasks in the at queue for the current user. • When run as root, atq or at -l lists all the jobs in the at daemon's queue. • When run as a user other than root, at lists only the jobs for the current user.
at -d *jobnumber* **atrm *jobnumber***	Removes jobs from the at queue. Uses spaces to separate multiple jobs.

10.3 cron Configuration Files

File	Description
/etc/crontab	The /etc/crontab (cron table) file holds entries that direct commands to execute at a specific time. The /etc/crontab file: • Is used to schedule custom tasks that run system-wide. • Can only be edited by the root user. crond runs tasks scheduled in the /etc/crontab file as the root user.
/etc/cron.directory	The cron daemon executes the scripts found in each of the following directories at the specified interval for the whole system: **/etc/cron.hourly** **/etc/cron.daily** **/etc/cron.weekly** **/etc/cron.monthly**
/var/spool/cron/username	If permitted, each user can create a personal crontab file located at /var/spool/cron/username.
/etc/cron.allow	The /etc/cron.allow file identifies users who are allowed to create their own cron jobs. If /etc/cron.allow file exists, then only users listed within it are allowed to create a crontab file in /var/spool/cron/username. All other users are denied, and the /etc/cron.deny file is ignored.
/etc/cron.deny	The /etc/cron.deny file identifies users who are not allowed to create cron jobs. If the /etc/cron.deny file exists, only the users listed within it are not allowed to edit /var/spool/cron/username. Everyone else is allowed. This file is only processed if the /etc/cron.allow file does not exist.

10.3 cron Commands

Command	Function
crontab	Manages the /var/spool/cron/username crontab file. Be aware of the following options: **-e** edits the crontab file for the current user in vi. **-l** displays the contents of the current user's crontab file. **-r** removes the current user's crontab file. **-u** username specifies a different user for the -e, -l, and -r options.
crontab file	Loads a crontab job from a file. Write the file using the crontab syntax. This command overwrites the current crontab.

10.3 Differences Between cron and anacron

The differences between cron and anacron are:	
	The cron daemon assumes that the Linux operating system will remain up and running 24 hours a day, seven days a week. If a system is not powered on when a scheduled cron job should run, it is skipped. The anacron service compensates for times when the system is powered off. If a job is scheduled in anacron while the system is powered off, the missed job will automatically run when the system comes back up.

10.3 anacron Fields

Field	Description
Period	The period field specifies the recurrence interval in days. For example: 1 means the task recurs daily. 7 means the task recurs weekly. 30 means the task recurs every 30 days. @monthly means the task recurs once per calendar month.
Delay	The delay field specifies the time (in minutes) that the anacron daemon should wait before executing a missed job after the system starts back up.
Job-identifier	The job-identifier field specifies a name that will be used for the job's timestamp file. The identifier must be unique for each anacron job. The timestamp file is created in the /var/spool/anacron directory and contains a single line with a timestamp that indicates the last time the particular job was run.
Command	The command field specifies the command or script that should be run.

10.4 Linux Clocks

Clock	Description
Hardware clock	The hardware clock keeps time using a chip on the system's motherboard. • The hardware clock runs independently of any software. The clock does not require access to the CPU or system RAM to run. • Soft power from the power supply and the CMOS battery ensures that the clock continues to run when the computer is turned off or even unplugged. • The hardware clock is sometimes called the real-time clock (RTC), BIOS clock, CMOS clock, or time of year (TOY) clock. • The current hardware clock time is stored in the /proc/driver/rtc file.
System time	System time keeps time using software running within the operating system. • Time on the system clock is measured as the number of seconds that have elapsed since 12:00 AM on Jan 1, 1970. • By default, when the computer boots, it initially sets the system time using the current time from the hardware clock. • After the operating system is running, the system time is the only clock used by applications and services. The hardware clock is ignored. • Changing the system time does not automatically change the hardware clock, although you can change the system time and change the hardware clock to match.

10.4 UTC Offset

Time	Description
Coordinated Universal Time (UTC)	Coordinated Universal Time (UTC), formerly known as Greenwich Mean Time (GMT), is a method of identifying a common time between devices regardless of their physical location in the world. • UTC is adjusted periodically to match the rotation of the earth by adding leap seconds. Leap seconds are required because the official duration of a second does not exactly match the earth's rotation (but it is very close). • UTC matches time to the rotation of the earth using a single fixed point in Greenwich, England. A line drawn from the North Pole to the South Pole that passes through Greenwich is called the prime meridian. • It is preferable for Linux systems to use UTC (not local time). This ensures that a single method of keeping time is used, regardless of the physical location of the computer. It also ensures that timestamps on files, within logs, and on database records remain consistent, especially on networks with hosts residing in multiple time zones. • Time expressed using UTC is identified by adding UTC or Z to the time. For example, 09:30 UTC is the same as 09:30Z or 0930Z. UTC is also called Zulu time.
UTC offset	The UTC offset identifies the amount of time that local time is ahead of or behind Coordinated Universal Time (UTC). • **Local time in each time zone is identified by the UTC offset**. For example: ◦ Time zones used in the United States are UTC-05 (Eastern), UTC-06 (Central), UTC-07 (Mountain), and UTC-08 (Pacific), with time being behind UTC. ◦ Time zones used in Europe and Asia are ahead of UTC. For example, time in Germany is UTC+01, and time in Japan is UTC+09. • **To convert UTC to local time, add time based on the UTC offset (UTC + offset)**. For example, if UTC is 06:00: ◦ Local time in New York (UTC-05) would be 01:00 (06:00 + - 5:00 = 1:00 AM). ◦ Local time in Los Angeles (UTC-08) would be 22:00 the previous day (10:00 PM). ◦ Local time in Japan (UTC+09) would be 15:00 (3:00 PM). • **To convert local time to UTC, subtract time based on the UTC offset (UTC - offset)**. For example: ◦ If the local time in New York (UTC-05) is 14:00, UTC is 19:00 (14:00 - - 5:00 = 14:00 + 5:00). ◦ If the local time in Japan (UTC+09) is 14:00, UTC is 5:00 (14:00 - + 9:00). • **UTC does not change for daylight saving time; however, the offset used by a time zone will change. During daylight saving time, add one hour to the UTC offset**. For example: ◦ Standard time in New York is UTC-05; daylight time in New York is UTC-04. ◦ Standard time in Germany is UTC+01; daylight time in Germany is UTC+02.
Local time	Local time is the current time in a local time zone. It is designated using the number of hours ahead or behind UTC time. For example, the local time for the Mountain Time Zone in the United States is UTC -7. The default setting for several hardware clocks in a system BIOS is often local time.

10.4 Time Zone Tools and Files

Item	Description
/usr/share/zoneinfo	Contains time zone configuration files and directories, with each file identifying a specific time zone. • Files are typically organized in subfolders based on the continent (such as Australia) or major country (such as the US). • Individual files identify a major city in the time zone (such as Perth) or a specific region (either a division of the country or a country within the continent). • Information in the file identifies the UTC offset and any rules for daylight saving time. Depending on the distribution, time zone files might be located at /usr/lib/zoneinfo.
/etc/localtime	Identifies the current time zone file used on the system. This file is a symbolic link to the appropriate time zone file in the /usr/share/zoneinfo directory. Relinking this file to a different time zone file changes the system's time zone.
/etc/timezone	Configures the time zone on Debian-based distributions. /etc/timezone identifies the current time zone by region and zone.
/etc/sysconfig/clock	Configures the time zone on some distributions, such as openSUSE. The syntax is: **TIMEZONE="timezone"**
date	Used to view and manually set the system time.
tzselect	Changes the value of the time zone (TZ) environment variable. When executed, the utility prompts you to select a region, then a country, and so on until it has enough information to determine the time zone. Only the root user can invoke the tzselect utility. To use tzselect: **1.** Enter tzselect. **2.** Enter the number from the list that corresponds to the correct continent or ocean. **3.** Enter the number from the list that corresponds to the correct region. **4.** Enter the number from the list that corresponds to the correct timezone. **5.** Press 1 to confirm the setting. Use the tzconfig command on Debian-based Linux distributions instead of tzselect.
TZ=time_zone **export TZ**	Changes the time zone environment variable. Use the file names in the /usr/share/zoneinfo directory to select the appropriate names for time zones.
timedatectl	Can be used to query and change the system clock and its settings. With no options, this command lists information on the system's current date and time, as well as time-related settings. A few options for this command include: • **set-time** ***yyyy/mm/dd hh/mm/ss*** changes the date and time according to the parameters entered. • **list-timezones** lists the available time zones recognized on your system. • **set-timezone** ***region/area*** sets the system time zone to the specified value. • **set-local-rtc** ***x*** where x is the value of 0 or 1. Configures the real-time clock (RTC) to either maintain the RTC in universal time or to maintain the RTC in local time instead. • **set-ntp** ***true/false*** Controls whether NTP-based network time synchronization is enabled.

10.4 Clock and Time Files and Utilities

Command	Function
/proc/driver/rtc	Views the hardware clock time.
hwclock	Views and sets the hardware clock time and synchronizes the hardware clock and the system time. Options include: **-a**, **--adjust** adds or subtracts time from the hardware clock to account for systematic drift since the last time the clock was set or adjusted. **-r**, **--show** displays the current hardware clock time. hwclock assumes -r if no options are used. **--set --date=** sets the hardware clock time and date. **-s**, **--hctosys** sets the system time to the current hardware clock time. **-w**, **--systohc** sets the hardware clock based on the system time. **--localtime** sets the hardware clock to local time. **-u**, **--utc** sets the hardware clock to UTC time.
netdate	Sets the system time to match the time on a server running the time daemon on the network. The time provider must be running the time service on UPD port 37.
date	Views and manually sets the system time. Options include: **-d** ***[date]*** displays the date specified by date. Use now to display the current date. **-s** sets the date and time. **-u**, **--utc** specifies UTC time.

10.4 NTP Files and Utilities

Command or File	Function
/etc/ntp.conf	Configures the time providers where the ntpd daemon should get its time from. • Each entry in the file begins with server, followed by the address of the time provider. • The server 127.127.1.0 entry identifies the local host address and sets the system time to the hardware clock if no other time providers are available.
ntpdate	Updates the current time on a computer. ntpdate must be run as root; it will not function if the ntpd daemon is currently running. ***ntpdate is deprecated; use ntpd in its place.***
ntpd	Manages the NTP daemon from the command line. Options include: **-q** does a one-time synchronization with a time provider. It is similar to ntpdate. **-g** allows the NTP daemon to ignore insane time restrictions for the first synchronization. **-c** specifies the name and path of the configuration file. The default is /etc/ntp.conf.
rcntp start (init) or **systemctl start ntpd (systemd)**	Starts the NTP daemon.
insserv ntp (init) or **systemctl enable ntpd (systemd)**	Configures the NTP daemon to start at boot time (BSD systems only.)
ntpq -p	Queries the status of the NTP daemon. The output displays the following information for each time provider: • remote specifies the IP address of the current time provider. • refid specifies the type of the time provider. • st shows the stratum of the time provider. • when shows the last synchronization time. • poll shows the synchronization interval. • reach lists the last time NTP queried the time provider. • delay displays the network lag time between the time provider and the client (in milliseconds). • offset specifies the time difference between the local system clock and the time provider (in milliseconds). • jitter specifies the size of time discrepancies (in milliseconds).
ntptrace	Traces how the time consumer receives time from the provider. It lists the time provider's name, its stratum, and its time offset from the system clock on the local system.