Redirection, Pipes &Processes

Redirection
&
Standard Input and Output

Redirection is a feature in Linux such that when executing a command, you can change/redirect the standard input/output devices.

Linux provides three I/O channels to Programs

- Standard input (STDIN) - keyboard by default
- Standard output (STDOUT) - terminal window by default
- Standard error (STDERR) - terminal window by default

Redirection

Redirecting Output to a File

STDOUT and STDERR can be redirected to files:

command output_redirection_operator filename

> is the output_redirection_operator

Supported operators include:

- > Redirect STDOUT to file
- 2> Redirect STDERR to file
- &> Redirect all output to file

Eg: ls –al > file1

cat > f3.txt

find /etc -name passwd > userpass

find /etc -name passwd 2> /home/user1/Desktop/error1

File	File Descriptor
Standard Input STDIN	0
Standard Output STDOUT	1
Standard Error STDERR	2

- **2>&1**: Redirects STDERR to STDOUT

find /etc -name passwd > /home/user1/Desktop/alldata 2>&1

File contents are overwritten by default.

>> appends.

Eg: cat f1.txt >> f2.txt

Redirection cont..

Redirecting STDIN from a File

- Redirect standard input with **<**

command output_redirection_operator filename

< is the input_redirection_operator

- Some commands can accept data redirected

to STDIN from a file:

Eg: sort < /home/user1/Desktop/*alldata*

Piping/Pipeline/ Pipes

- It is possible to connect multiple commands together to form a *pipeline*. With pipelines, the standard output of one command is fed into the standard input of another.

  Pipes (the **|** character) can connect commands:

  command1 | *command2*
- Sends STDOUT of command1 to STDIN of command2 instead of the screen.
- STDERR is not forwarded across pipes
- Used to combine the functionality of multiple tools

  command1 | *command2* | *command3*... etc

Examples of Piping

- ls -lt | head
- ls –al | sort
- ls -l /etc | less
- grep linux f1 f2 | sort
- grep linux f1 f2 | sort –r
- cat f1 |less
- ls | wc -l
- ls –lt |cut –d “ “ –f 1

Investigating and Managing Linux Processes

What is Process

- An instance of a program is called Process
- Each process in the system has a unique **pid or process ID**
- Types of Processes
  - Foreground

    By default, every process that you start runs in the foreground

    Also known as Interactive processes
  - Background

    Also known as non-Interactive processes

    To run a process in the background, use ampersand (**&**) at the end of the command

Process Control Commands

- ps –*To list the running processes*
- kill –*To terminate processes manually*
- top -*To get the list of all the running processes on your Linux machine.*
- bg -*To run all the pending and force stopped jobs in the background.*
- fg-*To run all the pending and force stopped jobs in the foreground.*
- jobs -*To get the list of jobs that are either running or stopped*

ps

Command	Description
ps	Display the processes from the current terminal by default
ps –a	Display the processes from all the terminals
ps –e ps -A	Display all running process on a Linux system
ps –ef	Full-format listing all running process on a Linux system
ps –u user1	Display the processes associated with a specific user, user1
ps -ely	Long Format of all running process on a Linux system
ps –ef |grep user1	Full-format listing all processes owned by user1
pidof httpd	Display the process ID/s of a running program by name

top

- Display all the processes executing in the machine
- Shift+M - *Sort by memory usage*
- Shift+p - *Sort processes as per CPU utilization*
- top –u user - *Display Specific User Process*

Kill

- Kill command is used to terminate the process
- kill pid
- kill 1230
- Kill -9 3279
- pkill <process name>

Signals

- Signals are software interrupts sent to a program to indicate that an important event has occurred

Signal Name	Signal Number	Description
SIGHUP	1	Hang up detected on controlling terminal or death of controlling process
SIGINT	2	Issued if the user sends an interrupt signal (Ctrl + C)
SIGQUIT	3	Issued if the user sends a quit signal (Ctrl + D)
SIGKILL	9	If a process gets this signal it must quit immediately and will not perform any clean-up operations

Scheduling a process- Crontab

- crontab [-u user] file

Options

- *file* *Load the crontab data from the specified file. If file is a dash ("-"), the crontab data is read from standard input.*
- *-u user* *Specifies the user whose crontab is to be viewed or modified. If this option is not given, crontab opens the crontab of the user who ran crontab.*
- -l *Display the current crontab.*
- -r *Remove the current crontab.*
- -e *Edit the current crontab, using the editor specified in the environment variable VISUAL or EDITOR.*
- -i *Same as -r, but gives the user a yes/no confirmation prompt before removing the crontab.*

Scheduling a process- Crontab

- **Linux Crontab Format**

  MIN HOUR DOM MON DOW CMD

Field	Description	Allowed Value
MIN	Minute field	0 to 59
HOUR	Hour field	0 to 23
DOM	Day of Month	1-31
MON	Month field	1-12
DOW	Day Of Week	0-6
CMD	Command	Any command to be executed.