RHSA1

Red Hat System Administration I

Information Technology Institute

COURSE MATERIALS

You can access the course materials via this link

https://goo.gl/ezCT7j

DAY 4 CONTENTS

- Processes, priorities and signals Concepts.
- Redirection
- Pipe Line
- Word Count

PROCESSES

- Every program you run creates a process. For example
  - Shell
  - Command
  - An application

PROCESSES

- System starts processes called **daemons** which are processes that run in the background and provide services
- Every processes has a **PID**
- When a process creates another, the first is the **parent** of the new process. The new process is called the **child process**. Parent waits for her child to finish

PROCESS PRIORITY

- Only process at a time may be executed on the CPU.
- Every process which is ready to run has a scheduling priority.
- The Linux process divides CPU time into time slices, in which each process will get a turn to run, higher priority processes first.
- User can affect the priority by setting the niceness value for a process

PROCESS PRIORITY

- Niceness values range from **-20** to **+19**, which indicates how much of a bonus or penalty to assign to the priority of the process.
- Most processes run with a niceness value of 0 (no change).

PROCESS PRIORITY

- Smaller numbers are higher priority. Processes with a higher priority will run first in each time slice, and will run longer before its turn to run ends.
- Users can adjust this value down as far as +19 but can not increase it. Root can increase the priority of a process as high as -20

ADJUSTING PRIORITY

- Adjusting process priority at invocation time

```
nice [-n adjustment] command
nice -n 20 makewhatis
```

- Adjusting the priority of a running process

```
renice priority [[-p] pid ...] [[-g]
group ...] [[-u] user ...]
```

SIGNALS

- A signal is a message sent to a process to perform a certain action
- Signals are identified by a signal number and a signal name, and has an associated action.

–SIGTERM → 15

–SIGKILL → 9

VIEWING PROCESSES

- Process status command

```
ps option(s)
```

- Output
  - PID
  - TTY -> terminal identifier
  - Execution time
  - Command name
- Options
  - -e: all system processes
  - -f: full information
  - -u uid: display processes of that user.
- Viewing processes with `top` utility

SEARCHING FOR A PROCESS

- Using `pgrep` command

```
pgrep option(s) pattern
$pgrep lp
```

– Options

-x: exact match

-u uid: processes for a specific user

-l: display the name with pid

SENDING A SIGNAL

- Using kill command
  - Default signal 15

```
kill [-signal] PIDs
```

- Examples

```
$pgrep -l mail
215 sendmail
12047 dtmail
```

SENDING A SIGNAL

```
$kill 12047
```

```
$pgrep -l mail
215 sendmail
```

- Using `pkill` command

```
pkill [-signal] process_name
```

- Example

```
$pkill -9 dtmail
```

EXAMPLES

```
sleep 500 &
[1]   3028
[1] + Done
jobs
[1] + Running     sleep 500&
fg 1
sleep 500
sleep 500
Ctrl+Z [1] + Stopped (SIGTSTP) sleep 500
jobs
[1] + Stopped (SIGTSTP) sleep 500
bg %1
[1] sleep 500&
```

EXAMPLES

```
jobs
[1] + Running sleep 500&
kill -STOP %1
jobs
[1] + Stopped (SIGSTOP) sleep 500&
kill %1
[1] + Terminated sleep 500&
jobs
```

STANDARD INPUT AND OUTPUT

- **Standard input**
  - Refers to the data source from which data is input to a command
  - Typically the keyboard
- **Standard output**
  - Refer to data destination to which data from the command is written
  - Typically the screen
- **Standard error**
  - Refer to the output destination for the errors and messages generated by the command
  - Typically the screen also

REDIRECTION

- `Command > fname`
- `Command >> fname`
- `Command < fname`

  - Example
    - `$ find /etc -name passwd > findresult`
    - `$ ls -l /etc >> findresult`
    - `Mail < file1`

REDIRECTION

- Standard error is redirected to a file using the regular output redirection operator, but you must place a 2 in front of the operator (2>).
- Example
  - $ find / -name passwd 2> errs
  - $ find / -name passwd 2> errs > results

PIPE LINE

- The pipe
  - A pipe (|) is used to send the output of one command as the input to another
  - The most common use of a pipe is to take a command that’s output might go on for pages (such as cat or ls -l) and feed it through more.
  - Example
    - `$ ls -lR / | more`

THE `tee` COMMAND

- The `tee` command reads from the standard input and writes to the standard output and a file
- Example
    - `$ ls -lR / | tee fname | more`

STRING PROCESSING

- Use the `wc` and the `diff` commands to gather word file statistics and compare two files
- Search strings for patterns using the `grep` command
- Move and delete data using `cut` and `paste` commands
- Organize data using the `sort`, and `paste` command

THE `wc` COMMAND

- The `wc` command displays the number of characters, words, and lines in a specified file.
- The syntax for the wc command is:
  - `wc [option] [filename]`
- The `wc` command is often used when differentiating between two versions of a file.

THE `wc` COMMAND

- Word-count command options

Option	Meanings
-c	Count the number of characters only
-l	Count the number of lines only
-w	Counts the number of words only

THE `wc` COMMAND

- For example,

```
$ wc story.txt
39 237 1901 story.txt
```

THE `diff` COMMAND

- The diff command is also used to compare the contents of two files for differences. If you upgrade a utility and want to see how the new configuration files differ from the old, use the diff command

```
 diff /etc/named.conf.rpm.new /etc/named.conf
will give the output as:
20c20
<
---- file "root.hints";
> file "named.ca"
```

THE `grep` COMMAND

- Displays the lines of its input that match a pattern given as an argument
  - `grep options regular-expression filename(s)`

Option	Description
-I	Ignore case sensitive
-l	List files name
-n	Precedes each line with relative line number in the file
-v	Inverse the search
-c	Counts the line that contains the pattern
-w	Search for the expression as a complete word

THE `tr` COMMAND

- The `tr` command can be used to translate characters from standard input and write to standard output
  - `tr [option] string1 string2`
  - `echo "Hello, world." | tr 'A-Z' 'a-z'`
    HELLO, WORLD

THE `cut` COMMAND

- cut command cuts fields or columns of text from standard input or the named file and displays the result to standard output
  - `cut option[s] [filename]`
    - -f specifies field or column.
    - -d specifies field delimiter (default is TAB).
    - -c specifies characters and cuts by characters.
  - `cut -f3 -d: /etc/passwd`

THE `sort` COMMAND

- The sort command sorts text data after accepting it from either a file or the output of another command.

- The sorted text is sent to the standard output, with the original file remaining unchanged in the process.

- Examples:
  - `sort -t : -k1 /etc/passwd`
  - `sort -t : -k3 /etc/passwd`
  - `sort -t : -n -k3 -o passwd_sorted /etc/passwd`