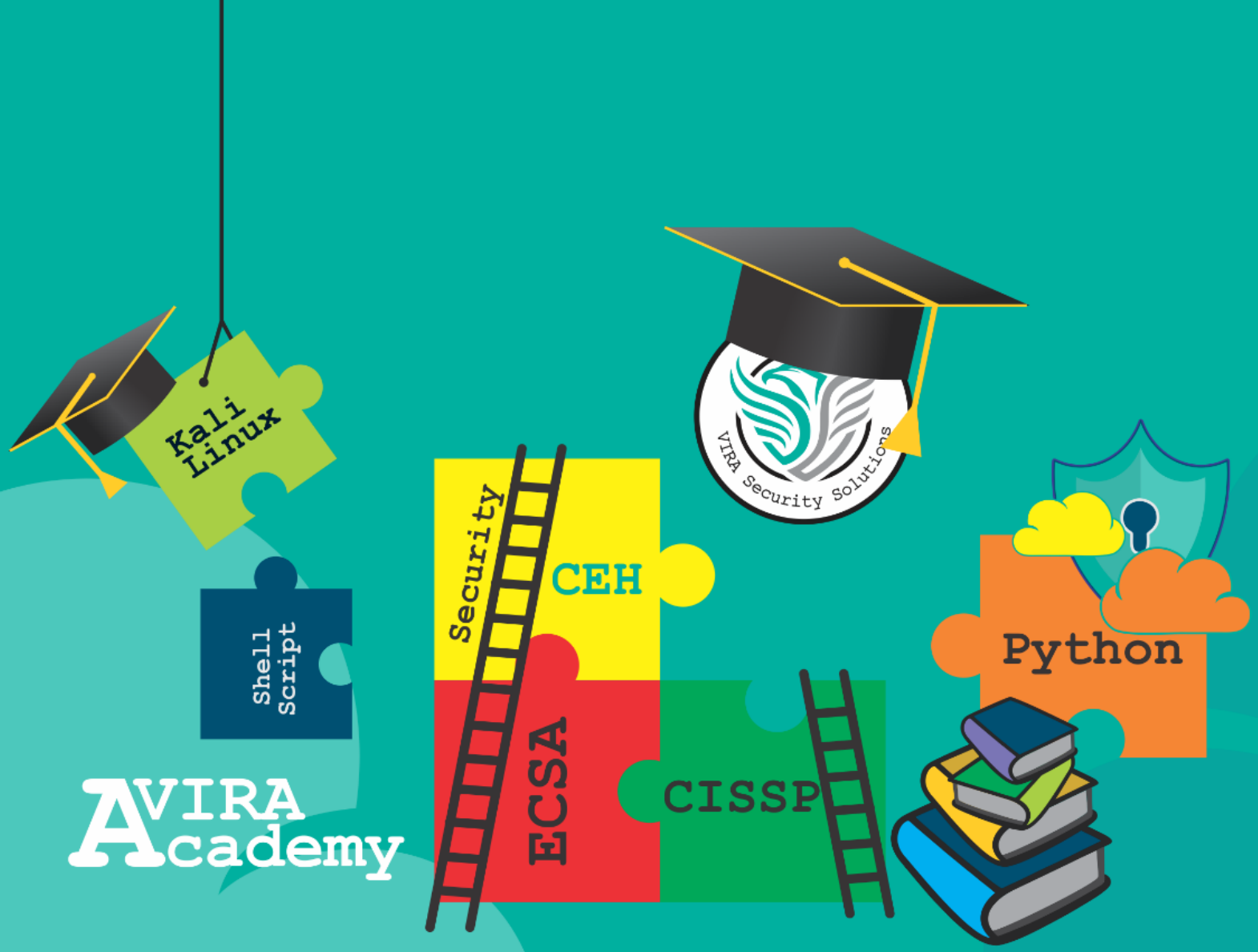

Kali Linux
Shell Script
VIRA Academy
Security
CEH
ECSA
CISSP
VIRA Security Solutions
Python

Linux

Part2

Mohammad Reza Gerami
mrgerami@aut.ac.ir
gerami@virasecsolutions.com
March 01 2020

Manage Files and Directories

- Home Directory

  The directory in which you find yourself when you first login is called your home directory.

  You will be doing much of your work in your home directory and subdirectories that you'll be creating to organize your files.

  You can go in your home directory anytime using the following command –

$cd ~
$
Here ~ indicates the home directory.
Suppose you have to go in any other user's home directory, use the following command –

$cd ~username
$
To go in your last directory, you can use the following command –

$cd -
$

Manage Files and Directories

Absolute/Relative Pathnames

Directories are arranged in a hierarchy with root (/) at the top. The position of any file within the hierarchy is described by its pathname.

Elements of a pathname are separated by a /. A pathname is absolute, if it is described in relation to root, thus absolute pathnames always begin with a /.

Following are some examples of absolute filenames

```
/etc/passwd
/home/gerami/test/file1.txt
```

Manage Files and Directories

A pathname can also be relative to your current working directory. Relative pathnames never begin with /. Relative to user gerami's home directory, some pathnames might look like this –

```
file2.txt
test/file1.txt
```

Manage Files and Directories

To determine where you are within the filesystem hierarchy at any time, enter the command **pwd** to print the current working directory –

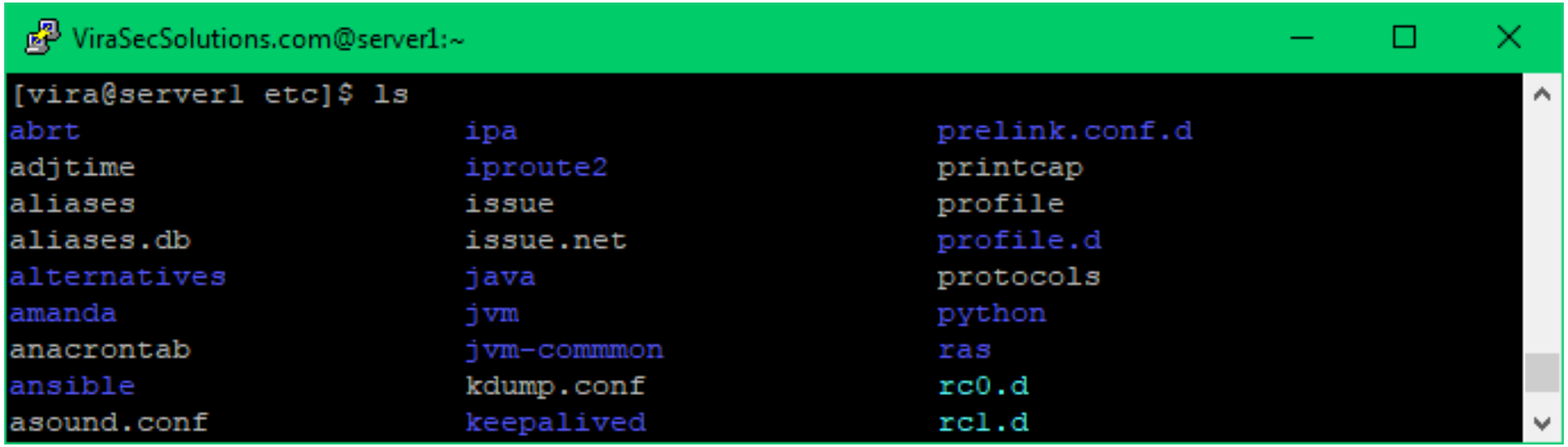
Manage Files and Directories

Listing Directories

To list the files in a directory, you can use the following syntax –

```
ls
```

Following is the example to list all the files contained in /etc directory – (we are in /etc directory)



Manage Files and Directories

Create directory

mkdir directoryname

Move out of directory

cd ..

Copy and move directories

cp -r /path_of_current_folder /path_of_destination

mv -r /path_of_current_folder /path_of_destination

How to delete a directory

rm -r /path_of_current_folder

Working Directory

pwd

-f or --force

causes cp to attempt to remove an existing target file even if it is not writable.

-i or --interactive

asks for confirmation before attempting to replace an existing file.

File Management

Listing Files

ls

ls –l

drwxrwxr-x 2 gerami gerami 4096 Apr 16 09:59 vira

First Column – Represents the file type and the permission given on the file. Below is the description of all type of files.

Second Column – Represents the number of memory blocks taken by the file or directory.

Third Column – Represents the owner of the file. This is the Unix user who created this file.

Fourth Column – Represents the group of the owner. Every Unix user will have an associated group.

Fifth Column – Represents the file size in bytes.

Sixth Column – Represents the date and the time when this file was created or modified for the last time.

Seventh Column – Represents the file or the directory name.

File Management

Metacharacters

Metacharacters have a special meaning in Unix. For example, * and ? are metacharacters. We use * to match 0 or more characters, a question mark (?) matches with a single character.

For Example –

$ls ch*.doc

Displays all the files, the names of which start with ch and end with .doc –

ch01-1.doc ch010.doc ch02.doc ch03-2.doc

Here, * works as meta character which matches with any character. If you want to display all the files ending with just .doc, then you can use the following command –

ls *.doc

File Management

Hidden Files

An invisible file is one, the first character of which is the dot or the period character (.). Unix programs (including the shell) use most of these files to store configuration information.

To list the invisible files, specify the -a option to ls –

$ ls -a

. .. bin hw1 .privatefile work ch07 .bashrc .bash_profile

Single dot (.) – This represents the current directory.

Double dot (..) – This represents the parent directory.

File Management

Creating/Edit Files

You can use the vi or nano editor to create ordinary files on any Unix system. You simply need to give the following command –

$vi filename

$nano filename

The above commands will open a file with the given filename.

You will now have a file created with filename in the current directory.

You can find that with:

$ ls

Also you can use touch command for create an empty file: **touch filename1**

File Management

Display Content of a File

You can use the cat command to see the content of a file. Following is a simple example to see the content of the above created file –

```
$ cat filename
This is unix file....I created it for the first time.....
I'm going to save this content in this file.
```

You can display the line numbers by using the -b option along with the cat command as follows –

```
$ cat -b filename
1  This is unix file....I created it for the first time.....
2  I'm going to save this content in this file.
```

File Management

Counting Words in a File

You can use the wc command to get a count of the total number of lines, words, and characters contained in a file. Following is a simple example to see the information about the file created above –

$ wc filename

2 19 103 filename

Here is the detail of all the four columns –

First Column – Represents the total number of lines in the file.

Second Column – Represents the total number of words in the file.

Third Column – Represents the total number of bytes in the file. This is the actual size of the file.

Fourth Column – Represents the file name.

File Management

Copying Files

```
cp    source_file    destination_file
cp    filename       copyfile
```

Renaming Files

```
mv    old_file    new_file
mv    filename    newfile
```

Deleting Files

```
rm    filename
rm    -rf    filename
```

Permissions

There are three permissions for any file, directory or application program.

The following lists the symbols used to denote each, along with a brief description:

r **— Indicates that a given category of user can read a file.**

w **— Indicates that a given category of user can write to a file.**

x **— Indicates that a given category of user can execute the file.**

Permissions

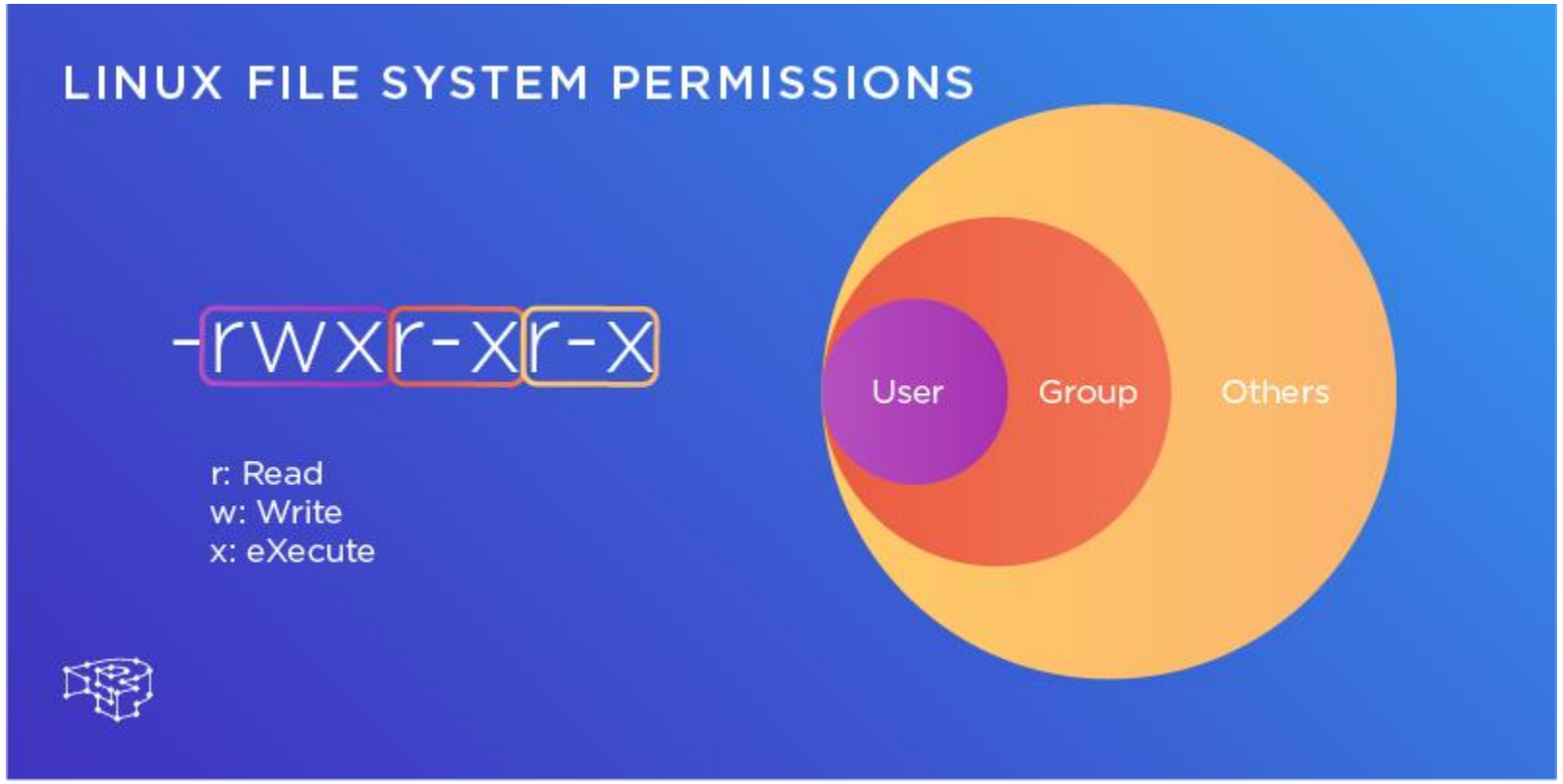
Each of the three permissions are assigned to three defined categories of users.

The categories are:

owner — **The owner of the file or application.**

group — **The group that owns the file or application.**

others — **All users with access to the system.**


LINUX FILE SYSTEM PERMISSIONS
-rwxr-xr-x
r: Read
w: Write
x: eXecute
User
Group
Others

```
# ls -l file
-rw-r--r-- 1 root root 0 Nov 19 23:49 file
```

Other (r - -)

Group (r- -)

Owner (rw-)

File type

r = Readable

w = Writeable

x = Executable

- = Denied

Permissions

One can easily view the permissions for a file by invoking a long format listing using the command ls -l.

For instance, if the user gerami creates an executable file named test, the output of the command ls -l test would look like this:

-rwxrwxr-x 1 gerami student 20 Apr 17 12:25 test

Permissions

r =4 w=2 x=1

rwx	rw-	r-x
7	6	5

chmod 765 test

Permissions

The permissions for this file are listed are listed at the start of the line, starting with rwx.

This first set of symbols define owner access.

The next set of rwx symbols define group access

The last set of symbols defining access permitted for all other users.

Changing Permissions

To change the file or the directory permissions, you use the chmod (change mode) command. There are two ways to use chmod — the symbolic mode and the absolute mode.

Using chmod in Symbolic Mode

The easiest way for a beginner to modify file or directory permissions is to use the symbolic mode.

With symbolic permissions you can add, delete, or specify the permission set you want by using the operators in the following table.

S.No.	Chmod operator	Description
1	+	Adds the designated permission(s) to a file or directory.
2	-	Removes the designated permission(s) from a file or directory.
3	=	Sets the designated permission(s).

Changing Permissions

chmod o+wx testfile

$ls -l testfile

-rwxrwxrwx 1 gerami users 1024 Apr 16 00:10 testfile

$chmod u-x testfile

$ls -l testfile

-rw-rwxrwx 1 gerami users 1024 Apr 16 00:10 testfile

$chmod g = rx testfile

$ls -l testfile

-rw-r-xrwx 1 gerami users 1024 Apr 16 00:10 testfile

Changing Permissions

Here's how you can combine these commands on a single line –

$chmod o+wx,u-x,g = rx testfile

$ls -l testfile

-rw-r-xrwx 1 gerami users 1024 Apr 16 00:10 testfile

Changing Permissions

Using chmod with Absolute Permissions

The second way to modify permissions with the chmod command is to use a number to specify each set of permissions for the file.
Each permission is assigned a value, as the following table shows, and the total of each set of permissions provides a number for that set.

Number	Octal Permission Representation	Ref
0	No permission	---
1	Execute permission	--x
2	Write permission	-w-
3	Execute and write permission: 1 (execute) + 2 (write) = 3	-wx
4	Read permission	r--
5	Read and execute permission: 4 (read) + 1 (execute) = 5	r-x
6	Read and write permission: 4 (read) + 2 (write) = 6	rw-
7	All permissions: 4 (read) + 2 (write) + 1 (execute) = 7	rwx

Mount and Unmount File Systems

On Linux and UNIX operating systems, you can use the mount command to attach (mount) file systems and removable devices such as USB flash drives at a particular mount point in the directory tree.

The umount command detaches (unmounts) the mounted file system from the directory tree.

In this tutorial, we will go over the basics of attaching and detaching various file systems using the mount and umount commands.

Mount and Unmount File Systems

How to List Mounted File Systems

When used without any argument, the mount command will display all currently attached file systems:

$ mount

By default, the output will include all file systems including the virtual ones such as cgroup, sysfs, and others. Each line contains information about the device name, the directory to which the device is mounted to, the type of the filesystem and the mount options in the following form:

device_name on directory type filesystem_type (options)

To display only certain file systems use the -t option.

For example, to print only the ext4 partitions you would use:

$ mount –t ext4

$ mount –t xfs

Mount and Unmount File Systems

Mounting a File System

To mount a file system in a given location (mount point), use the mount command in the following form:

mount [OPTION...] DEVICE_NAME DIRECTORY

Once the file system is attached, the mount point becomes the root directory of the mounted file system.

For example, to mount the /dev/sdb1 file system to the /mnt/media directory you would use:

sudo mount /dev/sdb1 /mnt/media

Mount and Unmount File Systems

Mounting a File System

Usually when mounting a device with a common file system such as ext4 or xfs the mount command will auto-detect the file system type. However, some file systems are not recognized and need to be explicitly specified.

Use the -t option to specify the file system type:

```
mount -t TYPE DEVICE_NAME DIRECTORY
```

To specify additional mount options, use the -o option:

```
mount -o OPTIONS DEVICE_NAME DIRECTORY
```

Multiple options can be provided as a comma-separated list (do not insert a space after a comma).

You can get a list of all mount options by typing man mount in your terminal.

Mount and Unmount File Systems

Mounting a File System using /etc/fstab

When providing just one parameter (either directory or device) to the mount command, it will read the content of the /etc/fstab configuration file to check whether the specified file system is listed or not.

If the /etc/fstab contains information about the given file system, the mount command uses the value for the other parameter and the mount options specified in the fstab file.

The /etc/fstab file contains a list of entries in the following form:

/etc/fstab

[File System]	[Mount Point]	[File System Type]	[Options]	[Dump]	[Pass]
/dev/sdb1	/mnt/test	default		0	0

Use the mount command in one of the following forms to attach a file system specified in the /etc/fstab file:

mount [OPTION...] DIRECTORY

mount [OPTION...] DEVICE_NAME

Mounting a File System

Now that we have created a new file system on the Linux partition of our new disk drive we need to mount it so that it is accessible. In order to do this we need to create a mount point.

#mkdir /mountpoint

The file system may then be manually mounted using the mount command:

#mount /dev/sdb1 /mountpoint

#mount -t ext4 /dev/sdb1 /mountpoint

Running the mount command with no arguments shows us all currently mounted file systems (including our new file system):

#mount

```
/dev/mapper/vg_CentOS6-lv_root on / type ext4 (rw)
proc on /proc type proc (rw)
sysfs on /sys type sysfs (rw)
....
/dev/sdb1 on /mountpoint type ext4 (rw)
```

Archive File (Compressing)

Create tar Archive File

```
tar -cvf test-970127.tar /home/gerami/test
```

Create tar.gz Archive File

```
tar cvzf test-970127.tar.gz /home/gerami/test
OR
# tar cvzf test-970127.tgz /home/gerami/test
```

Create tar.bz2 Archive File

```
# tar cvfj test-org.tar.bz2 /home/gerami/test
OR
# tar cvfj test-org.tar.tbz /home/gerami/test
OR
# tar cvfj test-org.tar.tb2 /home/gerami/test
```

Archive File (Uncompressing)

Untar tar Archive File

Untar files in Current Directory

```
# tar -xvf test-970127.tar
```

Untar files in specified Directory

```
# tar -xvf test-970127.tar -C /home/gerami/test2
```

Uncompress tar.gz Archive File

```
tar -xvf test-970127.tar.gz
```

Uncompress tar.bz2 Archive File

```
tar -xvf test-org.tar.bz2
```

History Command

HISTSIZE Variable

```
HISTSIZE=10000
```

List Last/All Executed Commands in Linux

```
history
```

List All Commands with Date and Timestamp

```
export HISTTIMEFORMAT='%F %T  '
history
```

Ignore Duplicate Commands in History

```
export HISTCONTROL=ignoredups
history
```

Unset export Command

```
unset export HISTCONTROL
```

History Command

Save export Command Permanently

#vi ~/.bash_profile

```
# .bash_profile
# Get the aliases and functions
if [ -f ~/.bashrc ]; then
. ~/.bashrc
fi
```

User specific environment and startup programs

```
export HISTCONTROL= ignoredups
```

History Command

Delete or Clear History of Commands

history -c

Search Commands in History Using Grep Command

history | grep pwd

Search Lastly Executed Command

Search previously executed command with ‘Ctrl+r’ command. Once you’ve found the command you’re looking for, press ‘Enter‘ to execute the same else press ‘esc‘ to cancel it.

(reverse-i-search)`source ': source .bash_profile

Recall Last Executed Command

!NumberofCommand

!9

!ls

Date and Time

Linux Display Current Date and Time

```
date
```

Linux Set Date Command

```
date --set="STRING"
# date -s "18 Apr 2018 10:00:00"
OR
# date --set="18 Apr 2018 10:00:00"
```

To set time use the following syntax:

```
# date +%T -s "10:13:13"
```

Where,

10: Hour (hh) **13: Minute (mm)** **13: Second (ss)**

Use %p locale's equivalent of either AM or PM, enter:

```
# date +%T%p -s “9:10:30AM"
# date +%T%p -s "12:10:30PM"
```

Package Management In Linux
(Redhat Based)

-Package Management In Linux

yum install packagename
yum install httpd

-Check For Available Updates

yum check-update

-Update New Package

yum update
yum update -y

-Uninstall Package

yum remove httpd

-Reinstall Package

yum reinstall httpd -y

-View Repository Information

yum repolist
yum repolist all

The cat (short for “concatenate“)

General Syntax

cat [OPTION] [FILE]...

Display Contents of File

In the below example, it will show contents of /etc/passwd file.

```
# cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
narad:x:500:500::/home/narad:/bin/bash
```

The cat (short for “concatenate“)

View Contents of Multiple Files in terminal

```
# cat test test1
```

Use Cat Command with More & Less Options

```
# cat test.txt | more
# cat test.txt | less
```

Display Line Numbers in File

```
# cat -n test.txt
```

Use Standard Output with Redirection Operator

We can redirect standard output of a file into a new file else existing file with ‘>‘ (greater than) symbol. Careful, existing contents of test1 will be overwritten by contents of test file.

```
# cat test > test1
```

Appending Standard Output with Redirection Operator

Appends in existing file with ‘>>‘ (double greater than) symbol. Here, contents of test file will be appended at the end of test1 file.

```
# cat test >> test1
```

Find

Find Files Using Name in Current Directory

```
# find . -name test1.txt
```

Find Files Under Home Directory

```
# find /home -name vira.txt
```

Find Files Using Name and Ignoring Case

```
# find /home -iname test.txt
./test.txt
./Test.txt
```

Find Directories Using Name

```
# find / -type d -name Test1
```

Find VCF Files Using Name

```
# find . -type f -name test.vcf
```

Find all VCF Files in Directory

```
# find . -type f -name "*.vcf"
```

Find

Find Files With 777 Permissions

```
# find . -type f -perm 0777 -print
```

Find Files Without 777 Permissions

```
# find / -type f ! -perm 777
```

Find and remove single File

```
# find . -type f -name "test.txt" -exec rm -f {} \;
```

Find and remove Multiple File

```
# find . -type f -name "*.txt" -exec rm -f {} \;
OR
# find . -type f -name "*.mp3" -exec rm -f {} \;
```

Find all Empty Files

```
# find /tmp -type f -empty
```

Find all Empty Directories

```
# find /tmp -type d -empty
```

Find

Find Single File Based on User

```
# find / -user root -name test.txt
```

Find all Files Based on User

```
# find /home -user gerami
```

Find all Files Based on Group

```
# find /home -group tcgs
```

Find Last 50-100 Days Modified Files

```
# find / -mtime +50 –mtime -100
```

Find Changed Files in Last 1 Hour

```
# find / -cmin -60
```

Find Size between 50MB – 100MB

```
# find / -size +50M -size -100M
```

Find and Delete 100MB Files

```
# find / -size +100M -exec rm -rf {} \;
```

Processes in Linux

A process refers to a program in execution; it's a running instance of a program.

Types of Processes

There are fundamentally two types of processes in Linux:

- Foreground processes (also referred to as interactive processes) – these are initialized and controlled through a terminal session. In other words, there has to be a user connected to the system to start such processes; they haven't started automatically as part of the system functions/services.
- Background processes (also referred to as non-interactive/automatic processes) – are processes not connected to a terminal; they don't expect any user input.

Processes in Linux

Because Linux is a multi-user system, meaning different users can be running various programs on the system, each running instance of a program must be identified uniquely by the kernel.

A program is identified by its process ID (PID) as well as it's parent processes ID (PPID), therefore processes can further be categorized into:

Parent processes – these are processes that create other processes during run-time.

Child processes – these processes are created by other processes during run-time.

Processes in Linux

- Starting a Process in Linux
  - Once you run a command or program (for example top), it will start a process in the system. You can start a foreground (interactive) process as follows, it will be connected to the terminal and a user can send input it:
    - $ sleep 10

- Linux Background Jobs
  - To start a process in the background (non-interactive), use the & symbol, here, the process doesn't read input from a user until it's moved to the foreground.
    - $ sleep 10 &
    - $ jobs

Processes in Linux

- You can also send a process to the background by suspending it using [Ctrl + Z], this will send the SIGSTOP signal to the process, thus stopping its operations; it becomes idle:
  - # tar -cf backup.tar /backups/* #press Ctrl+Z
  - # jobs
- To continue running the above-suspended command in the background, use the bg command:
  - # bg
- To send a background process to the foreground, use the fg command together with the job ID like so:
  - # jobs
  - # fg %1

Processes in Linux

States of a Process in Linux

During execution, a process changes from one state to another depending on its environment/circumstances. In Linux, a process has the following possible states:

- Running – here it's either running (it is the current process in the system) or it's ready to run (it's waiting to be assigned to one of the CPUs).
- Waiting – in this state, a process is waiting for an event to occur or for a system resource. Additionally, the kernel also differentiates between two types of waiting processes; interruptible waiting processes – can be interrupted by signals and uninterruptible waiting processes – are waiting directly on hardware conditions and cannot be interrupted by any event/signal.
- Stopped – in this state, a process has been stopped, usually by receiving a signal. For instance, a process that is being debugged.
- Zombie – here, a process is dead, it has been halted but it's still has an entry in the process table.

Processes in Linux

How to View Active Processes in Linux

There are several Linux tools for viewing/listing running processes on the system, the two traditional and well known are ps and top commands:

$ ps

$ top

$ glances

Processes in Linux

How to Control Processes in Linux

Linux also has some commands for controlling processes such as kill, pkill, pgrep and killall, below are a few basic examples of how to use them:

```
$ pgrep -u vira top
$ kill 2308
$ pgrep -u vira top
$ pgrep -u vira glances
$ pkill glances
$ pgrep -u vira glances
```

Processes in Linux

Sending Signals To Processes

The fundamental way of controlling processes in Linux is by sending signals to them. There are multiple signals that you can send to a process, to view all the signals run:

$ kill -l

ViraSecSolutions@server1:~

```
[root@server1 ~]# kill -l
 1) SIGHUP       2) SIGINT       3) SIGQUIT      4) SIGILL       5) SIGTRAP
 6) SIGABRT      7) SIGBUS       8) SIGFPE       9) SIGKILL     10) SIGUSR1
11) SIGSEGV     12) SIGUSR2     13) SIGPIPE     14) SIGALRM     15) SIGTERM
16) SIGSTKFLT   17) SIGCHLD     18) SIGCONT     19) SIGSTOP     20) SIGTSTP
21) SIGTTIN     22) SIGTTOU     23) SIGURG      24) SIGXCPU     25) SIGXFSZ
26) SIGVTALRM   27) SIGPROF     28) SIGWINCH    29) SIGIO       30) SIGPWR
31) SIGSYS      34) SIGRTMIN    35) SIGRTMIN+1  36) SIGRTMIN+2  37) SIGRTMIN+3
38) SIGRTMIN+4  39) SIGRTMIN+5  40) SIGRTMIN+6  41) SIGRTMIN+7  42) SIGRTMIN+8
43) SIGRTMIN+9  44) SIGRTMIN+10 45) SIGRTMIN+11 46) SIGRTMIN+12 47) SIGRTMIN+13
48) SIGRTMIN+14 49) SIGRTMIN+15 50) SIGRTMAX-14 51) SIGRTMAX-13 52) SIGRTMAX-12
53) SIGRTMAX-11 54) SIGRTMAX-10 55) SIGRTMAX-9  56) SIGRTMAX-8  57) SIGRTMAX-7
58) SIGRTMAX-6  59) SIGRTMAX-5  60) SIGRTMAX-4  61) SIGRTMAX-3  62) SIGRTMAX-2
63) SIGRTMAX-1  64) SIGRTMAX
[root@server1 ~]#
```

Processes in Linux

How to Control Processes in Linux

To send a signal to a process, use the kill, pkill or pgrep commands we mentioned earlier on. But programs can only respond to signals if they are programmed to recognize those signals.

And most signals are for internal use by the system, or for programmers when they write code. The following are signals which are useful to a system user:

SIGHUP 1 – sent to a process when its controlling terminal is closed.

SIGINT 2 – sent to a process by its controlling terminal when a user interrupts the process by pressing [Ctrl+C].

SIGQUIT 3 – sent to a process if the user sends a quit signal [Ctrl+D].

SIGKILL 9 – this signal immediately terminates (kills) a process and the process will not perform any clean-up operations.

SIGTERM 15 – this a program termination signal (kill will send this by default).

SIGTSTP 20 – sent to a process by its controlling terminal to request it to stop (terminal stop); initiated by the user pressing [Ctrl+Z].

Processes in Linux

How to Control Processes in Linux

The following are kill commands examples to kill the top application using its PID once it freezes:

```
ViraSecSolutions@server1:~
[vira@server1 ~]$ pidof top
2044
[vira@server1 ~]$ kill -9 2044
```

To kill an application using its name, use pkill or killall like so:

```
ViraSecSolutions@server1:~
[vira@server1 ~]$ pkill top
[vira@server1 ~]$ killall top
[vira@server1 ~]$
```

Processes in Linux

Changing Linux Process Priority

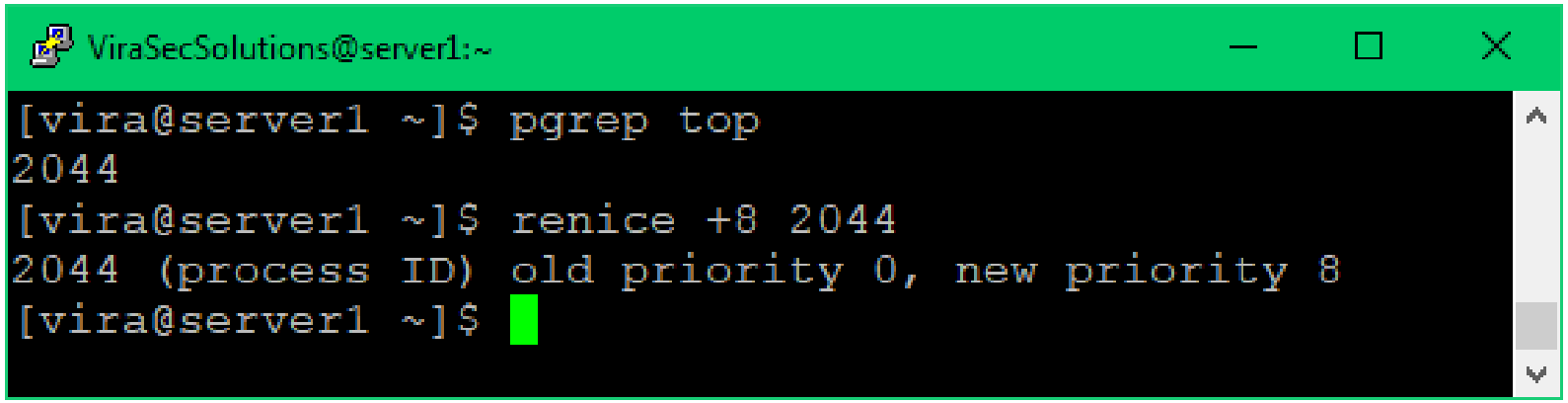
On the Linux system, all active processes have a priority and certain nice value. Processes with higher priority will normally get more CPU time than lower priority processes.

However, a system user with root privileges can influence this with the nice and renice commands.

Use the nice command to set a nice value for a process. Keep in mind that normal users can attribute a nice value from zero to 20 to processes they own.

Only the root user can use negative nice values.

To renice the priority of a process, use the renice command as follows:



Command Line Tools to Monitor Linux Performance

- Top – Linux Process Monitoring
- VmStat – Virtual Memory Statistics
- Lsof – List Open Files
- Tcpdump – Network Packet Analyzer
- Netstat – Network Statistics
- Htop – Linux Process Monitoring
- Iotop – Monitor Linux Disk I/O
- Iostat – Input/Output Statistics
- IPTraf – Real Time IP LAN Monitoring
- Psacct or Acct – Monitor User Activity

Command Line Tools to Monitor Linux Performance

- Monit – Linux Process and Services Monitoring
- NetHogs – Monitor Per Process Network Bandwidth
- iftop – Network Bandwidth Monitoring
- Monitorix – System and Network Monitoring
- Arpwatch – Ethernet Activity Monitor
- Suricata – Network Security Monitoring
- VnStat PHP – Monitoring Network Bandwidth
- Nagios – Network/Server Monitoring
- Nmon: Monitor Linux Performance
- Collectl: All-in-One Performance Monitoring Tool

TOP command examples on Linux to monitor processes

Be aware that the top command comes in various variants and each has a slightly different set of options and method of usage.

- To check your top command version and variant use the -v option

```
ViraSecSolutions.com@server1:~
[vira@server1 ~]$ top -v
  procps-ng version 3.3.10
```

Top – Linux Process Monitoring

Display processes

$ top

ViraSecSolutions.com@server1:~

```
top - 04:42:05 up 3 min,  1 user,  load average: 0.10, 0.18, 0.09
Tasks: 123 total,   3 running, 120 sleeping,   0 stopped,   0 zombie
%Cpu(s):  0.3 us,  0.3 sy,  0.0 ni, 99.3 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
KiB Mem :   999696 total,   689704 free,   148932 used,   161060 buff/cache
KiB Swap:  2097148 total,  2097148 free,        0 used.   676588 avail Mem

   PID USER      PR  NI    VIRT    RES    SHR S %CPU %MEM     TIME+ COMMAND
   902 root      20   0       0      0      0 S  0.3  0.0   0:00.04 xfsaild/dm+
  1123 root      20   0  305096   6144   4780 S  0.3  0.6   0:00.35 vmtoolsd
     1 root      20   0  128460   7244   4148 S  0.0  0.7   0:02.46 systemd
     2 root      20   0       0      0      0 S  0.0  0.0   0:00.00 kthreadd
     3 root      20   0       0      0      0 S  0.0  0.0   0:00.09 ksoftirqd/0
     4 root      20   0       0      0      0 S  0.0  0.0   0:00.00 kworker/0:0
     5 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 kworker/0:+
     6 root      20   0       0      0      0 S  0.0  0.0   0:00.12 kworker/u2+
     7 root      rt   0       0      0      0 S  0.0  0.0   0:00.00 migration/0
     8 root      20   0       0      0      0 S  0.0  0.0   0:00.00 rcu_bh
     9 root      20   0       0      0      0 R  0.0  0.0   0:00.76 rcu_sched
    10 root      rt   0       0      0      0 S  0.0  0.0   0:00.00 watchdog/0
    12 root      20   0       0      0      0 S  0.0  0.0   0:00.00 kdevtmpfs
    13 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 netns
    14 root      20   0       0      0      0 S  0.0  0.0   0:00.00 khungtaskd
    15 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 writeback
    16 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 kintegrityd
```

Top – Linux Process Monitoring

PID - Process ID
USER - The system user account running the process.
%CPU - CPU usage by the process.
%MEM - Memory usage by the process
COMMAND - The command (executable file) of the process

Top – Linux Process Monitoring

Sort by Memory/Cpu/Process ID/Running Time

To find the process consuming the most CPU or memory, simply sort the list.

Press M key (yes, in capital, not small) to sort the process list by memory usage.
Processes using the most memory are shown first and rest in order.
Here are other options to sort by CPU usage, Process ID and Running Time -
Press 'P' - to sort the process list by CPU usage.

Press 'N' - to sort the list by process id

Press 'T' - to sort by the running time.

Top – Linux Process Monitoring

Reverse the sorting order - 'R'

By default the sorting is done in descending order. Pressing 'R' shall reverse the sorting order of the currently sorted column.

Here is the output sorted in ascending order of CPU usage. Processes consuming the least amount of CPU are shown first.

```
ViraSecSolutions.com@server1:~
top - 04:52:35 up 13 min,  1 user,  load average: 0.08, 0.04, 0.05
Tasks: 118 total,   2 running, 116 sleeping,   0 stopped,   0 zombie
%Cpu(s):  0.0 us,  0.3 sy,  0.0 ni, 99.7 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
KiB Mem :   999696 total,   691724 free,   146860 used,   161112 buff/cache
KiB Swap:  2097148 total,  2097148 free,        0 used.   678616 avail Mem

   PID USER      PR  NI    VIRT    RES    SHR S %CPU %MEM     TIME+ COMMAND
     2 root      20   0       0      0      0 S  0.0  0.0   0:00.00 kthreadd
     5 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 kworker/0:0H
     7 root      rt   0       0      0      0 S  0.0  0.0   0:00.00 migration/0
     8 root      20   0       0      0      0 S  0.0  0.0   0:00.00 rcu_bh
    10 root      rt   0       0      0      0 S  0.0  0.0   0:00.00 watchdog/0
    12 root      20   0       0      0      0 S  0.0  0.0   0:00.00 kdevtmpfs
    13 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 netns
    14 root      20   0       0      0      0 S  0.0  0.0   0:00.00 khungtaskd
    15 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 writeback
    16 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 kintegrityd
    17 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 bioset
    18 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 kblockd
    19 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 md
    25 root      20   0       0      0      0 S  0.0  0.0   0:00.00 kswapd0
    26 root      25   5       0      0      0 S  0.0  0.0   0:00.00 ksmd
    27 root      39  19       0      0      0 S  0.0  0.0   0:00.00 khugepaged
    28 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 crypto
    36 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 kthrotld
    38 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 kmpath_rdacd
    39 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 kpsmoused
    41 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 ipv6_addrconf
    60 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 deferwq
    93 root      20   0       0      0      0 S  0.0  0.0   0:00.00 kauditd
   270 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 ata_sff
   271 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 mpt_poll_0
```

Top – Linux Process Monitoring

Highlight the sorted column with bold text - 'x'

Press x, to highlight the values in the sort column with bold text. Here is a screenshot, with the memory column in bold text.

ViraSecSolutions.com@server1:~

```
top - 04:54:53 up 15 min,  1 user,  load average: 0.03, 0.05, 0.05
Tasks: 118 total,   2 running, 116 sleeping,   0 stopped,   0 zombie
%Cpu(s):  0.0 us,  0.3 sy,  0.0 ni, 99.7 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
KiB Mem :   999696 total,   691724 free,   146860 used,   161112 buff/cache
KiB Swap:  2097148 total,  2097148 free,        0 used.   678616 avail Mem

   PID USER      PR  NI    VIRT    RES    SHR S %CPU %MEM     TIME+ COMMAND
     2 root      20   0       0      0      0 S  0.0  0.0   0:00.00 kthreadd
     3 root      20   0       0      0      0 S  0.0  0.0   0:00.11 ksoftirqd/0
     5 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 kworker/0:0H
     6 root      20   0       0      0      0 S  0.0  0.0   0:00.15 kworker/u256:0
     7 root      rt   0       0      0      0 S  0.0  0.0   0:00.00 migration/0
     8 root      20   0       0      0      0 S  0.0  0.0   0:00.00 rcu_bh
     9 root      20   0       0      0      0 R  0.0  0.0   0:00.89 rcu_sched
    10 root      rt   0       0      0      0 S  0.0  0.0   0:00.00 watchdog/0
    12 root      20   0       0      0      0 S  0.0  0.0   0:00.00 kdevtmpfs
    13 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 netns
    14 root      20   0       0      0      0 S  0.0  0.0   0:00.00 khungtaskd
    15 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 writeback
    16 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 kintegrityd
    17 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 bioset
    18 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 kblockd
    19 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 md
    25 root      20   0       0      0      0 S  0.0  0.0   0:00.00 kswapd0
    26 root      25   5       0      0      0 S  0.0  0.0   0:00.00 ksmd
    27 root      39  19       0      0      0 S  0.0  0.0   0:00.00 khugepaged
    28 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 crypto
    36 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 kthrotld
    38 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 kmpath_rdacd
    39 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 kpsmoused
    41 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 ipv6_addrconf
    60 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 deferwq
```

Top – Linux Process Monitoring

Highlight sorted column background color 'b'

After highlighting the sorted column with bold font, its further possible to highlight with a different background color as well. This is how it looks.

```
ViraSecSolutions.com@server1:~
top - 04:57:26 up 18 min,  1 user,  load average: 0.00, 0.03, 0.05
Tasks: 119 total,   2 running, 117 sleeping,   0 stopped,   0 zombie
%Cpu(s):  0.3 us,  0.7 sy,  0.0 ni, 99.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
KiB Mem :   999696 total,   686916 free,   147228 used,   165552 buff/cache
KiB Swap:  2097148 total,  2097148 free,        0 used.   675980 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU %MEM     TIME+ COMMAND
    2 root      20   0       0      0      0 S   0.0  0.0   0:00.00 kthreadd
    3 root      20   0       0      0      0 S   0.0  0.0   0:00.12 ksoftirqd/0
    5 root       0 -20       0      0      0 S   0.0  0.0   0:00.00 kworker/0:0H
    6 root      20   0       0      0      0 S   0.0  0.0   0:00.15 kworker/u256:0
    7 root      rt   0       0      0      0 S   0.0  0.0   0:00.00 migration/0
    8 root      20   0       0      0      0 S   0.0  0.0   0:00.00 rcu_bh
    9 root      20   0       0      0      0 R   0.0  0.0   0:00.89 rcu_sched
   10 root      rt   0       0      0      0 S   0.0  0.0   0:00.01 watchdog/0
   12 root      20   0       0      0      0 S   0.0  0.0   0:00.00 kdevtmpfs
   13 root       0 -20       0      0      0 S   0.0  0.0   0:00.00 netns
   14 root      20   0       0      0      0 S   0.0  0.0   0:00.00 khungtaskd
   15 root       0 -20       0      0      0 S   0.0  0.0   0:00.00 writeback
   16 root       0 -20       0      0      0 S   0.0  0.0   0:00.00 kintegrityd
   17 root       0 -20       0      0      0 S   0.0  0.0   0:00.00 bioset
   18 root       0 -20       0      0      0 S   0.0  0.0   0:00.00 kblockd
   19 root       0 -20       0      0      0 S   0.0  0.0   0:00.00 md
   25 root      20   0       0      0      0 S   0.0  0.0   0:00.00 kswapd0
   26 root      25   5       0      0      0 S   0.0  0.0   0:00.00 ksmd
   27 root      39  19       0      0      0 S   0.0  0.0   0:00.00 khugepaged
   28 root       0 -20       0      0      0 S   0.0  0.0   0:00.00 crypto
   36 root       0 -20       0      0      0 S   0.0  0.0   0:00.00 kthrotld
   38 root       0 -20       0      0      0 S   0.0  0.0   0:00.00 kmpath_rdacd
   39 root       0 -20       0      0      0 S   0.0  0.0   0:00.00 kpsmoused
   41 root       0 -20       0      0      0 S   0.0  0.0   0:00.00 ipv6_addrconf
   60 root       0 -20       0      0      0 S   0.0  0.0   0:00.00 deferwq
```

Top – Linux Process Monitoring

Change the update delay - 'd'

The top command updates the information on the screen every 3.0 seconds by default. This refresh interval can be changed.
Press the 'd' key, and top will ask you to enter the time interval between each refresh. You can enter numbers smaller than 1 second as well, like 0.5. Enter the desired interval and hit Enter.

```
ViraSecSolutions.com@server1:~
top - 05:48:25 up  1:09,  1 user,  load average: 0.09, 0.11, 0.08
Tasks: 119 total,   2 running, 117 sleeping,   0 stopped,   0 zombie
%Cpu(s):  0.3 us,  0.0 sy,  0.0 ni, 99.7 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
KiB Mem :   999696 total,   561332 free,   204536 used,   233828 buff/cache
KiB Swap:  2097148 total,  2097148 free,        0 used.   583596 avail Mem
Change delay from 3.0 to
   PID USER      PR  NI    VIRT    RES    SHR S %CPU %MEM     TIME+ COMMAND
  1123 root      20   0  305096   6148   4780 S  0.3  0.6   0:06.00 vmtoolsd
  1988 root      20   0       0      0      0 S  0.3  0.0   0:00.82 kworker/0:0
  2051 vira      20   0  157708   2168   1508 R  0.3  0.2   0:00.03 top
     1 root      20   0  128460   7252   4156 S  0.0  0.7   0:02.73 systemd
```

Top – Linux Process Monitoring

Display full command path and arguments of process - 'c'

Press 'c' to display the full command path along with the command line arguments in the COMMAND column.

```
ViraSecSolutions.com@server1:~
top - 06:07:22 up  1:28,  2 users,  load average: 0.00, 0.01, 0.05
Tasks: 123 total,   3 running, 120 sleeping,   0 stopped,   0 zombie
%Cpu(s):  0.3 us,  0.7 sy,  0.0 ni, 99.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
KiB Mem :   999696 total,   554176 free,   210492 used,   235028 buff/cache
KiB Swap:  2097148 total,  2097148 free,        0 used.   577416 avail Mem

   PID USER      PR  NI    VIRT    RES    SHR S %CPU %MEM     TIME+ COMMAND
  1123 root      20   0  305096   6148   4780 S  1.0  0.6   0:07.62 /usr/bin/vmtoolsd
  2051 vira      20   0  157736   2396   1652 R  0.7  0.2   0:00.69 top
     1 root      20   0  128460   7252   4156 S  0.0  0.7   0:03.14 /usr/lib/systemd/systemd --switc+
     2 root      20   0       0      0      0 S  0.0  0.0   0:00.00 [kthreadd]
     3 root      20   0       0      0      0 S  0.0  0.0   0:00.43 [ksoftirqd/0]
     5 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 [kworker/0:0H]
     6 root      20   0       0      0      0 S  0.0  0.0   0:00.35 [kworker/u256:0]
     7 root      rt   0       0      0      0 S  0.0  0.0   0:00.00 [migration/0]
     8 root      20   0       0      0      0 S  0.0  0.0   0:00.00 [rcu_bh]
     9 root      20   0       0      0      0 R  0.0  0.0   0:01.08 [rcu_sched]
    10 root      rt   0       0      0      0 S  0.0  0.0   0:00.05 [watchdog/0]
    12 root      20   0       0      0      0 S  0.0  0.0   0:00.00 [kdevtmpfs]
```

Top – Linux Process Monitoring

Display full command path and arguments of process - 'c'

To view the processes of a specific user only, press 'u' and then top will ask you to enter the username.

Which user (blank for all)
Enter the desired username and hit Enter.
This example shows vira user processes.

```
ViraSecSolutions.com@server1:~
top - 06:09:37 up  1:30,  2 users,  load average: 0.00, 0.01, 0.05
Tasks: 125 total,   2 running, 123 sleeping,   0 stopped,   0 zombie
%Cpu(s):  0.0 us,  0.3 sy,  0.0 ni, 99.7 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
KiB Mem :   999696 total,   551084 free,   213160 used,   235452 buff/cache
KiB Swap:  2097148 total,  2097148 free,        0 used.   574720 avail Mem

   PID USER      PR  NI    VIRT    RES    SHR S %CPU %MEM     TIME+ COMMAND
  2285 vira      20   0  157688   2220   1576 R  0.7  0.2   0:00.20 top
  1707 vira      20   0  116448   3188   1712 S  0.0  0.3   0:00.05 bash
  2287 vira      20   0  116556   3360   1808 S  0.0  0.3   0:00.30 bash
  2380 vira      20   0  107944    608    516 S  0.0  0.1   0:00.00 tail
```

Top – Linux Process Monitoring

Batch mode

Top also supports batch mode output, where it would keep printing information sequentially instead of a single screen. This is useful when you need to log the top output for later analysis of some kind.

Here is a simple example that shows the CPU usage at intervals of 1 second.

ViraSecSolutions.com@server1:~

```
[vira@server1 ~]$ top -d 1.0 -b | grep Cpu
%Cpu(s):  0.0 us, 21.1 sy,  0.0 ni, 78.9 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu(s):  1.0 us,  1.0 sy,  0.0 ni, 98.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu(s):  0.0 us,  1.0 sy,  0.0 ni, 99.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu(s):  1.0 us,  0.0 sy,  0.0 ni, 99.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu(s):  0.0 us,  2.0 sy,  0.0 ni, 98.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu(s):  1.0 us,  1.0 sy,  0.0 ni, 98.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu(s):  0.0 us,  1.0 sy,  0.0 ni, 99.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu(s):  1.0 us,  1.0 sy,  0.0 ni, 98.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu(s):  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu(s):  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
```

Top – Linux Process Monitoring

Batch mode

Top also supports batch mode output, where it would keep printing information sequentially instead of a single screen. This is useful when you need to log the top output for later analysis of some kind.
Here is a simple example that shows the CPU usage at intervals of 1 second.

vira@server1:~

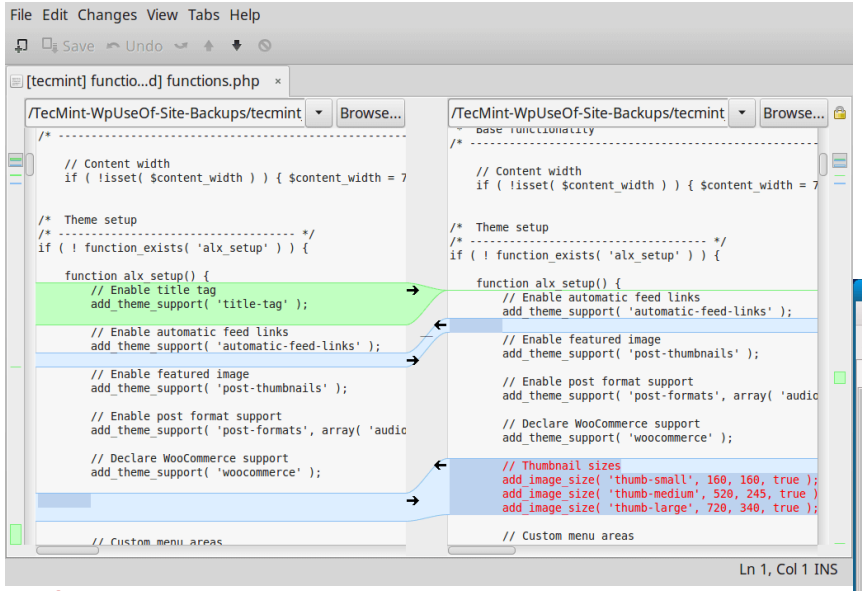
```
[vira@server1 ~]$ top -d 1.0 -b | grep Mem
KiB Mem :   999696 total,   545652 free,   216592 used,   237452 buff/cache
KiB Swap:  2097148 total,  2097148 free,        0 used.   570800 avail Mem
KiB Mem :   999696 total,   545652 free,   216592 used,   237452 buff/cache
KiB Swap:  2097148 total,  2097148 free,        0 used.   570800 avail Mem
KiB Mem :   999696 total,   545652 free,   216592 used,   237452 buff/cache
KiB Swap:  2097148 total,  2097148 free,        0 used.   570800 avail Mem
KiB Mem :   999696 total,   545528 free,   216716 used,   237452 buff/cache
KiB Swap:  2097148 total,  2097148 free,        0 used.   570676 avail Mem
KiB Mem :   999696 total,   545528 free,   216716 used,   237452 buff/cache
KiB Swap:  2097148 total,  2097148 free,        0 used.   570676 avail Mem
```

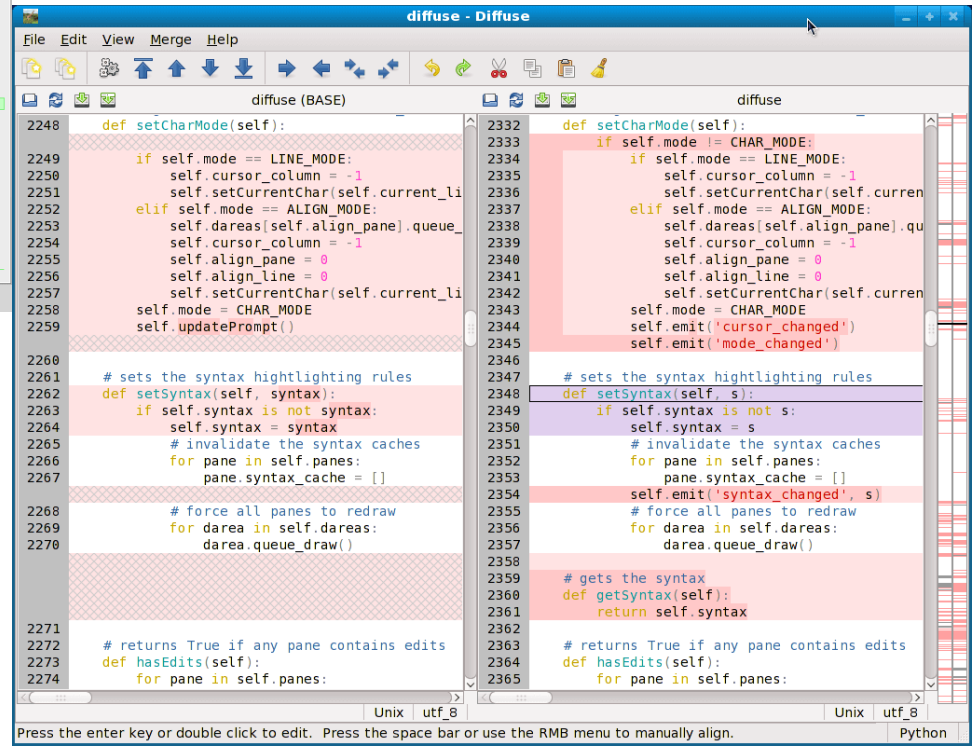
File Comparison and Difference

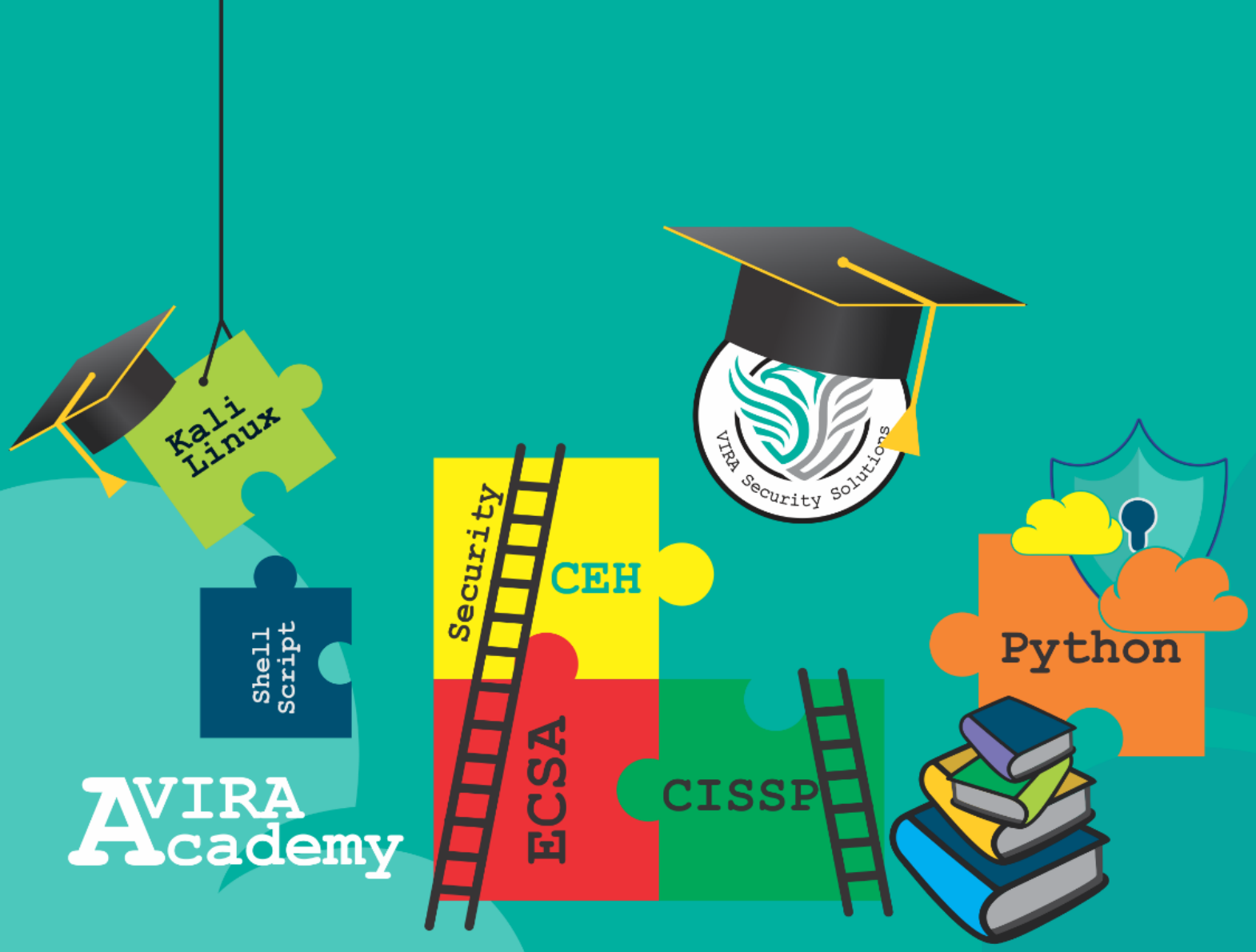
Tools

- diff Command
- Vimdiff Command
- Kompare
- DiffMerge
- Meld – Diff Tool
- Diffuse – GUI Diff Tool
- XXdiff – Diff and Merge Tool

diff file1 file2






Kali Linux
Shell Script
AVIRA Academy
Security
CEH
ECSA
CISSP
VIRA Security Solutions
Python