# Red Hat System Administration I – Quick Guide

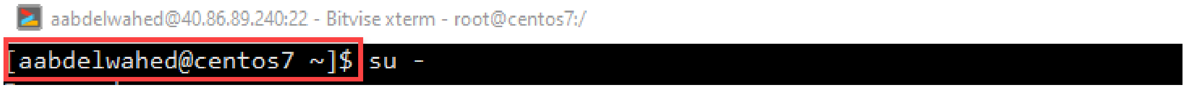
**Version 22.10**

Ahmed Abdelwahed
MCT
ahmed@abdelwahed.me

# Access the command line

## Command Prompts

Command Prompts is a short text at the start of the command line followed by prompt symbol on a command line interface.


aabdelwahed@40.86.89.240:22 - Bitvise xterm - root@centos7:/

```
[aabdelwahed@centos7 ~]$ su -
```

Where **aabdelwahed** is the username, **centos** is the hostname followed by working directory **(~ /)**, **$** is the prompt symbol where **$** for limited user and **#** for root

## Linux Command Syntax

- **<cmd> [+- option] [argument],** command syntax
- Command sometimes comes without option, option just add more features for the command also command sometimes comes without argument, argument is the file name or file path.
- Option can write in long format **(ls -all)** or in short format **(ls -a)**
- Option start with **–** or **+**
- Option change the command behavior and one command can have multiple option
- Option and argument not mandatory in command
- Directory comes in blue color or end wirh / in case you run **ll -F**
- Hidden file start with dot **(.)**
- **.** current directory
- **..** parent directory
- **cd** or **cd** ~ goes to home (~)
- cd – goes to previous directory
- **ctrl+c** intrupt the running command
- **ctrl+d** logout
- **ctrl+u** remove from current place to begin
- **ctrl+k** removes from current place to end
- **ctrl+a** put arrow to the command begins
- **ctrl+e** put arrow to the end of command
- **ctrl+w** to delete one word
- **esc+d** delete next word
- **ls; cal; date; pwd** command grouping
- **ctrl+alt+ f2-f6** access tty2-tty6 Physical Console F1 is the graphical environment
- **ctrl+l** clears the terminal screen

## TAB Completion and Up Arrow Keys

Tabs make wrting easer in Linux for command and file and directories name. also, you can hit double tab to give you the available options. while hitting up arrow key on the keyboard returns the last command ran. You can add this feature in case not work. **yum install bash-completion**

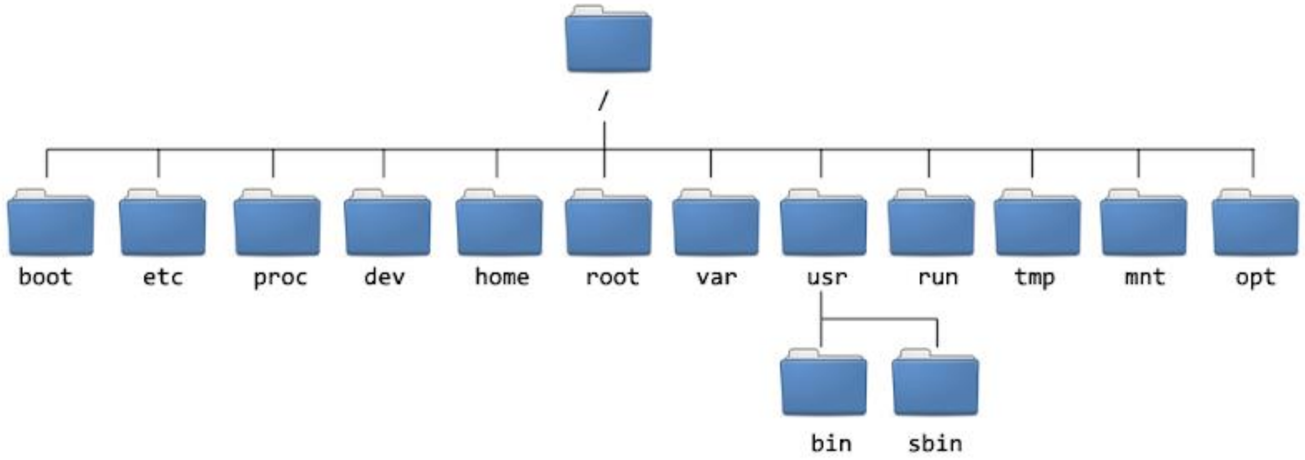
- set disable-completion off

# Manage files from the command line

## Filesystem Structure (Linux File System Layout)

Linux file system has a tree like structure

- **tree -L 2 / > system_stratuctre.txt** to redirect the linux hierarchy tree to system_stratuctre.txt file



- **/ root** topmost directory and is starting point of everything.
- **/boot** contain everything that the system needs to startup (boot loaders). The two files are absolutely essential are: vmlinuz (the compressed Linux kernel) and initramfs (the initial RAM Filesystem) which is mounted before the real root filesystem become available.
- **/etc** contain all system configuration files.
- **/proc** contains information about system resources / processes. **cat /proc/cpuinfo**
- **/dev** system devices files, such as cdrome, keyboard, flashdrive, disk and used to interact with hardware and software devices. Any device attached to your Linux system they created as a file and show up in /dev
- **/home** user home directories placed under /home, there is one exception: the home for the root user on Linux systems is always found under /root.
- **/root** is the home directory for the root user.
- **/var** contains files and directories that expected to grow, these include: - log files – spool directories files for printing, mail queues
- **/usr** contains program files and include the following:
  - **/bin**, executable programs and scripts (Everyday user commands) needed by both administrators and unprivileged users. (cat, chmod, cp, dd, echo, sh, sed, su, ping, tar, .... ) **type ls**, **type cat** or **which cat**, **which su**
  - **/sbin**, this directory contains binaries essential for booting, restoring, recovering, and/or repairing (fdisk, reboot, mkfs, mkswap, swapon, swapoff, update, halt), system and filesystem commands for example if you want to create filesystem or extend filesstem
  - **/lib** changed to **/usr/lib** ib64, these directories contain C programming libraries needed to execute the commands and apps in **/usr/bin** and **/usr/sbin**, these libraries are particularly important for booting the system and executing commands within the root filesystem. For example, if you running pwd command this command has a library attached in **/usr/lib**
- **/run** store system daemons that are start early (e.g. systemd and udev)
- **/ tmp**, this directory is used to store temporary files. If you want to create a file and delete it later you can create it here

- **/mnt**, mounting storage device, in newer OS mounting performing automatically.
- **/opt**, this directory is designed for software packages (third-party applications) that wish to keep all or most of their files in one isolated place, rather than scatter them all over the system in directories shared by other software.
- **/media**, this directory is typically used to mount filesystems on removable media. Including CDs, DVDs, and USB drives. When insert USB for example, the udev creates directories under /media and then mounts the removable filesystems there, upon unmounting and removal, the directories used as mount points under /media will disappear.
- **/sys**, this directory is the mount point for the sysfs pseudo-filesystem where all information resides only in memory, not on disk (du -s /sys/). Like/dev, the /proc directory is empty on a non-running system.

**Note**: on some newer distributions (including SUSE and RHEL7) removable media will pop up under /run/media/username/...

**Note**: some distributions run automated cron jobs, which remove any files older than 10 days like RHEL 6, while other distributions remove the contents of /tmp with every reboot. This has been the Ubuntu policy.

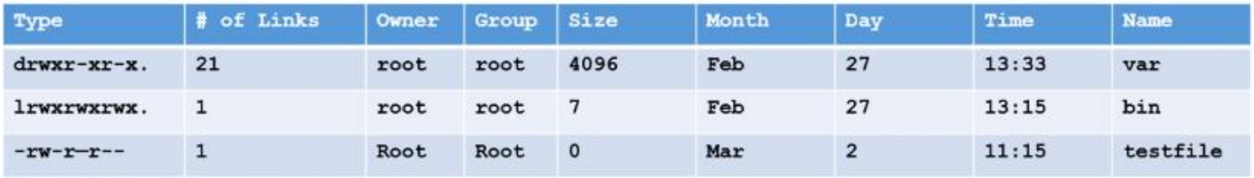
## File System Paths

Relative path starts from current place while Absolute path start from **/** (root directory)

## File System Navigation Commands (cd, ls and pwd)

First, everything starts with – is a file and everything start with d is a directory. There are 3 main command uses to navigate the system:

- **pwd** Print Working Directory
- **cd** Change Directory
- **ls** list directoy contents



| Type | # of Links | Owner | Group | Size | Month | Day | Time | Name |
|---|---|---|---|---|---|---|---|---|
| drwxr-xr-x. | 21 | root | root | 4096 | Feb | 27 | 13:33 | var |
| lrwxrwxrwx. | 1 | root | root | 7 | Feb | 27 | 13:15 | bin |
| -rw-r—r-- | 1 | Root | Root | 0 | Mar | 2 | 11:15 | testfile |

## Listing Files (Directories)

in Linux we use **/** as a path separator, also remember everything case-sensitive.

- **ls** listing current directory
- **type ls** show you actual command is 'ls –color=auto' so ls is alias.
- **ls -a**, list all including hidden files that begin with dot. Note that there are some files in blue color and others black, blue color indicates directories while black indicates files.
- **ls -aF** shows file type where directories will end with / (try ls -aF /etc)
- **ls -l /etc** shows long list that including permissions and modified date.
- **ls -lt /etc** shows long list with reverse sorting r according to time t so the result will be most recent.
- ls -lrt /etc shows long list with reverse sorting r according to time t so the result will be most recent last.
- **ls -ld /etc** show etc info not list etc content

## Creating Files and Directories

- **mkdir** to create new directory
- **mkdir -p par1/par2/dir** Create dir with parent
- **touch f1 f2 f3** create multiple files

## File Display Commands

- **cat** essentialy used to concatenate files (merge multiple files), cat file1 file2 > file3 , you can use it also to read from single and multiple files, **cat /etc/hosts /etc/hostname, cat file***

- **cat -n /etc/passwd** display file contents with numbering
- **tac /etc/passwd,** reading file from down o up
- **less /etc/passwd,** less used to read long files, use **q** to quit.
- **more /var/log/messages,** read pages use **s** to search
- **head /etc/passwd,** show first 10 lines in file
- **head -n 15 /etc/passwd:** show first 15 lines in file
- **tail -n 15 /etc/passwd:** show bottom 15 lines in file
- **tail -n +10 passwd | less,** show from line 10 to end
- **wc -l /etc/services** show number of lines, you can use also switch w (word) and c (character).
- **wc -l /etc/services /etc/passwd,** show number of lines in 2 files. Also use option **m** fro characters and **w** for words

## File Maintenance Commands

- **cp** to copy files
- **cp file1 file2 file3 dir** Copy multiple files to dir
- **cp -r file1 file2 file3 /media** use -r with directory
- **cp -rvi file1 /media/file4** copy and rename. Note you can't rename multiple files in one step, also use option -i to ask before process
- **cp /etc/passwd .** copy passwd file to the current directory
- **cp /etc/passwd ..** copy passwd file to the parent directory
- **cp /etc/passwd ./users** to copy and rename passwd file to users in the current directory
- **mv** to move files to another location or to rename files when you use it at the same location
- **mv file1 file2** result is renam
- **mv file1 file2 file3 dir** move multiple files to dir
- **mv -f file1 file2 file3 /media/** to force move files without ask for overwritten
- **rmdir** or **rm -r** to remove directory
- **rm** to remove files
- **rm -rf dir1 dir2 dir3** remove multiple dirs. without ask Note can't undo deleted files

## Wildcards

A wildcard is a character that can be used as a substitute for any of a class of characters in a search:

- **asterisk *** - represents zero or more characters
- **Question mark ?** – represents a one single character
- **Bracketed characters []** – matches any occurrence of character enclosed in the square brackets
- To create multiple files you can use wildcard, in the following example we will create 9 files **touch file0{1..9}**
- and to remove all of them, you can also use wildcard, **rm file0***
- to create multiple files and directories use **mkdir dir{1..10}**, **touch abc0{1..9}-xyz**, now if you want to delete everything ending with xyz, **rm *-xyz**
- **touch dir{1..10}/file{1..100},** create 100 file in each directory from dir1 to dir 10
- create file01, then use ? to search, **touch file01, ll fi??01,** while ?? indicate there are 2 charcters lost
- **cat f[!dfghe]le1**, any characters except **dfghe**

## Redirection

- **Command I/O Redirection** (Standard input, standard output and standard error)
- 0, Keyboard (input) you can use 0< or < to take input from file not from keyboard
- 1, screen terminal (standard output) you can use 1> or > to redirect the output from screen to file

- 2, screen terminal (standard error) you can use **2>** to redirect the error from screen to file

**Note**, Greater than symbol redirect the standard output

- **cat passwd > passwd_orig** take input from passwd and show output to passwd_orig
- **df -h > newfile** overwrite also its content
- **cat /dev/null > ahmedfile** delete file content
- **>> file** append the current file
- **2>/dev/null** ignor error by sending to /dev/null
- **ls /etcw 2> err** redirect standard error to err file
- **(cal 2010; 111) >op.txt 2>err.txt** result 2 files one for output and another for error
- **(cal 2010; 111) &>mix.txt** ouput and error in one file
- **find /etc -type l &> err** redirect standard output and error also
- (cal 2010; 111) 2> /dev/null show output and hide error
- **shred passwd** destroy file content
- **shred -u passwd** destroy and remove passwd file

**Note: shred** is a program that will overwrite your files in a way that makes them very difficult to recover by a third party.

- **echo “wooooow” > passwd,** overwrite passwd content with wooow
- **echo “wooow” >> passwd,** append wooow to passwd
- **df -hlT, disk free** human **l** file system **T** file system type
- **df -hlT > diskfree**

## Using the Command tee and tr (translate)

- **tee** read from standard input and write to both standard output (screen) and files
- **ls | tee f99** redirect output to standard output (monitor) and file
- **echo "Hello World" | tr -d aieou** will delete any characters included in **aieou**, so output will be **Hll Wrld**
- **echo "Hello World" | tr aieou x** will replace any characters included in **aieou** with **x** (or will translated to x), so output will be **Hxllx Wxrld**
- **echo abc | tr ab xy** output **xyc**
- **tr set get < ipconfig.txt** will translate or replace (on fly) set to get in file ipconfig.txt
- **tr set get <file1> file2.txt** take input from tr then translate set to get in file1 then output result in file2.txt
- **tr "[a-z]" "[A-Z]"<passwd>passwd2.txt** translate every small character to capital in file passwd and output result to passwd2.txt file

## Pipes (|)

A pipe is used by the shell to connect the output of one command directly to input of another command.

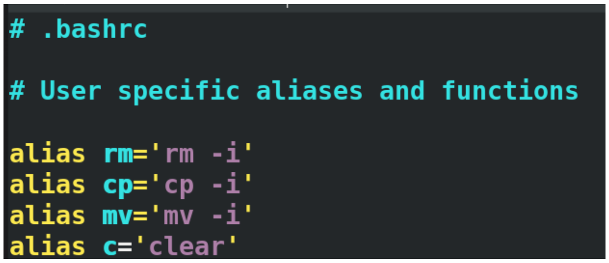
- **ll | wc -l**
- **ll /etc/ | more** to display the result page by page
- **ll /etc/ | tail -1** to display only last line of the result

## System Utility Commands

Here is some basic utility, some of them is used also in windows like calendar and date

- **date** show current date and time
- **date +%d-%m-%Y-%H-%M-%S**
- **uptime**, show system running time and number of working users
- **time cat /etc/passwd** cpu time to open passwd
- **who** see those users
- **w** uptime output and working users
- **lscpu** and **cat /proc/cpuinfo** show CPU info
- **lsmem** and **cat /proc/meminfo** Show online status information about memory blocks

- **cat /proc/uptime** show two columns, right one is CPU running time (by sec) and left one for cpu idle time. If idle time greater than running time it is indicating that you have more than one cpu.
- **watch -n 3 uptime** show the uptime every 3 sec
- **tload** show running CPU utilization (for testing try to copy large file from another terminal and monitor the change in cpu utilization)
- **hostname**
- **hostname -f** display full name
- **hostnamectl set-hostname** change the system hostname
- **uname** prints the name, version and other details about the current machine and the operating system also you can use **uname -a** for more info
- **which** pwd shows command source path because each command is file
- **cal** for calendar of this month and day also you can get specific month **cal 6 1982.** Also you can get calendar of the year by running **cal 1982**
- **whoami**
- **cd,** redirect (Change Directory) to home directory which indicating with tilde ~
- **ip a s,** show IP address
- **tty,** show you current logged in shell
- **passwd** change user password
- **!g** run last command that start with g
- **timedatectl list-timezones**
- **timedatectl set-timezones Aisa/Muscat** change system time zone
- **bc** for scientific calculator (binary calculator)
- **grep -e 'root' -e 'aabdelwahed' /etc/passwd** or **egrep -i 'root|aabdelwahed' /etc/passwd** search for multiple words
- **grep -E -i -w – o 'user1|user2|user3' /etc/passwd**
- **egrep -i 'error|warning| critical' /var/log/message**
- **egrep -i 'root\ ahmed' /etc/passwd** search for sequenced words
- **egrep -i '(^root)' /etc/passwd** search for line starts with root
- **fgrep "root**  
  **> aabdelwahed**  
  **> ssh**  
  **> gnome" /etc/passwd**
- **cls= "clear"** create alias, to read that alias **$cls**
- **alias -p** print all predefined alias
- **unalias cls** remove cls alias
- **unalias -a** remove all alias including defaults
- to persist the alias save it in the following file inside user home directory **.bashrc**
- **!!** or **ctrl+p** to run last command
- **vim !$** to run the command on last argument



```
# .bashrc

# User specific aliases and functions

alias rm='rm -i'
alias cp='cp -i'
alias mv='mv -i'
alias c='clear'
```

# Get help in Red Hat Enterprise Linux

Give you some information or help for other commands.

## Help Levels

- Short description, by using **whatis** in case does not work run **mandb**
- **whereis**
- **which**
- Portfolio, by using **–help**
- Manual, by running **man** and contains 9 sections.

## man command tips:

- **g** page start or **p**
- **shift+g** end of man
- **q** quit
- **/** Search forward
- **?** search backward
- **n** next search result
- **N** previous search result
- **100g** go to line 100
- **Space** next page
- **man -K "copy files"** note you can use install for copying, search for word
- **man 5 crontab** jump to section 5 in man

- More info than man, by using **info** (using space and backspace and u, s, for search) **info vim**
- Documentation, **/usr/share/doc**
- Online documentation

## Create, view, and edit text files

## Linux File Editor (vi)

vi supplies commands for:

- Inserting and deleting text
- Replacing text
- Moving around the file
- Finding and substituting strings
- Cutting and pasting text

### Command Mode

- **1G or gg** go to first line
- **G** go to last line
- **2G** or **2gg** second line
- **20gg**, go to line 20
- **i** insert Mode, write before the current cursor
- **o** inserts new line below current line
- **O** insert new line above current line
- **I** insert at start of line
- **a** appends, write after the cursor
- **A** appends at end of the line
- **k** up one line
- **yl**, copy letter
- **yw** copy word
- **yy** copy line
- **20yy** copy 20 lines
- **p** past below the line
- **P** past above the line
- **dl** delete letter
- **dw** delete word
- **dd** delete one line
- **20dd** delete 20 lines
- **d$** delete to the end of line
- **cl** delete letter and go to change mode
- **cw** delete word and go to change mode
- **cc** delete one line and go to change mode
- **20cc** delete 20 lines and go to change mode
- **u** undo
- **shift u** undo one line
- **^** go to start of the current line
- **$** go to end of the current line
- **Shift+~** change case letter by letter
- **g~~** change whole line case (Upper and small)
- **gUU** change whole line upper case
- **shift J** merge lines
- **shift zz** save and exit from command mode

- **shift zq** exit without save from command mode

## Execute Mode

- **:x** save and exit
- **q!** force exit without save
- **:w**
- **:w!**
- **:wq**
- **:wq!**
- **:set nu** numbering lines
- **:set nonu** remove numbering
- **:set arabic** write Arabic
- **:set noarabic**
- **:set ic** ignore case sensitive
- **:set noic**
- **:se hlsearch** highlight search result
- **:se nohlsearch** remove highlight search
- **:set all** show you all set manual options
- **:%s/install/config/g** substitute for install word and replace it with config in whole file (global)
- **:$d** delete last line
- **:1,9d** delete lines from 1 to 9
- **:%d** delete file content

## Visual mode

For commenting a block of text is almost the same:

1. First, go to the first line you want to comment, press Ctrl+V. This will put the editor in the VISUAL BLOCK mode.
2. Then using the arrow key and select until the last line
3. Now press Shift+I, which will put the editor in INSERT mode and then press #. This will add a hash to the first line.
4. Then press Esc (give it a second), and it will insert a # character on all other selected lines.

## Set vim defaults with .vimrc (custom vim)

- You can set all defaults you want to apply when using vim by creating hidden file under user home (**.**) named **.vimrc** and write all settings that you want
  - **:set number** show lines number
  - **:set nonumber** remove lines number
  - **:set ignorecase**
  - **:set hlsearch**
  - **:set nohlsearch**

To set vim global setting for all users you have to edit vim **/etc/vimrc**

**vim tips**:

to lock vim screen use **ctrl+s**, to unlock use **ctrl+q**, anything typed will appear after unlock.

- **ctrl+w+n** add new vim screen horizontal
- **ctrl+w+v** add new vim screen vertical
- **ctrl+w** move between screens

- **nano nf2** edit files using nano
- **gedit file** graphic editing tool

## Manage local users and groups

### id Command

- **id** show Linux user identification

**id test1 uid=1003(test1) gid=1003(test1) groups=1003(test1)**

Uid user ID (in Linux root id is 0, and another standard -normal- user account IDs starts from 1000. Users ID less than 1000 is considered special and belong to the system)

Gid primary group (private group) ID almost the same uid, the idea of private groups is it keeps your data more private unless you choose to share it out.

Groups secondary groups for file access permissions

- **id -g** show primary group
- **id -G** show secondary group memberships.
- **id -Gn** show secondary group name
- **id -gn** show primary group name

### Creating User Accounts

- **useradd -m user1** -m to create home directory for created user but by default we don't need this switch because home directory is created automatic

**Note**: when you select only user name all user settings will be the default.

- **useradd -u 5555 user05**, creating user with custom id 5555
- **useradd -M user1**, create user account without home directory, try to login with this user using **su –** and **su**
- **tail -n 1 /etc/passwd**, show only last line in passwd (is the file where our local users are stored), you can the same result using **getent passwd | tail -n -1**

  **ls -a /home/user1** list the home directory its like **ls -a /etc/skel** (..., .bashrc, .bash_profile)
- **useradd -N test03** primary and secondary group will be **users' group, -N**, --no-user-group
- **useradd -N user100 -g user99 -G user10** create user without private group, private group will be users' group and secondary group will be adm.
- **id -Gn user100** show user100 primary and secondary group

### Create Bulk Users

1. Create users.txt
2. Add the users in the following syntax:

LoginName: Passord: UID: GID: Comment: home_dir: ShellName

```
user01:user001:1002:1002:user01:/home/user01:/bin/bash
user02:user002:1003:1003:user02:/home/user02:/bin/bash
user03:user003:1004:1004:user03:/home/user03:/bin/bash
user04:user004:1005:1005:user04:/home/user04:/bin/bash
user05:user005:1006:1006:user05:/home/user05:/bin/bash
```

3. **newusers users.txt**

## Managing User Passwords

- **passwd –help**
- **passwd user1**, create user1 password (with root privileges you can create simple password like 111 but when some users create his own password must be complex)
- **sudo passwd otheruser**
- **grep user1 /etc/shadow** show user hashed password.
- **grep user /etc/shadow** show all users which name start with user and have only one character or write user only to indicate all users they start with user notice that all users without password or has invalid password they have !! sign
  user1:$6$V1rm7Rko$KkCR4zpM3sZBfAj24w4Nb/logKvm85g12iVd8AZ61gGXPLAdDt71gQCS.SkbCWk7/s9cy5fOIdZTlDouQNIl2.:17576:0:99999:7:::
  17576 third one is the number of days after the 1st January 1970
  0 zero saying we don’t need to keep that password for any particular time, sometimes we might set a minimum number of days
  99999 we have to change our password every 99.000 days
  7 get warning 7 days before it expires
  ::: couple of empty fields which looking for the account and password expiration date

- <u>**Implementing Password Policies**</u>

- **less /etc/security/pwquality.conf**
- **pwscore**, to check the quality of password, value between **0** and **100**.

## Password Age

- **chage –help**, change password data
- **chage -l test**, show password age for test user you can change this info by using **chage** command with root privileges, all of this related to shadow data.
- **chage -d 0 username**, to force user to change their password at their first login.
- **grep test /etc/passwd**
  **test:x:1000:1000::/home/test:/bin/bash,** x indicate that the password stored in shadow data, because any user can read this file (passwd) so can take this algorithm and decrypt it, so we use shadow file to keep password private. If you run **pwunconv** the password will be stored at **passwd** file. now **/etc/shadow** not exist to back to it again just run **pwconv**
- **chage -m 10 test1,** minimum password age
- **chage -M 40 test1**, change maximum password age for user1 to 40 days
- **chage -d 0 -m 14 -M 45 -W 3 test10, W** for warning days
- **chage -l user01**, list aging settings for user1
- **grep test /etc/shadow**
  test:$6$21GeUNcu$a.CEGpuc2qHWKqzL72ZMEj/HqsmmCwjxcA..h7yaFT/YcBln68VB9sfsnhk70hRShTczXSdlaJorS6xWbE9Q6.:17577:0:40:7:::
- **passwd -l test1** Locking password for user test01
  test1:!!$6$V1rm7Rko$KkCR4zpM3sZBfAj24w4Nb/logKvm85g12iVd8AZ61gGXPLAdDt71gQCS.SkbCWk7/s9cy5fOIdZTlDouQNIl2.:17577:0:99999:7:::password which start with !! indicate that this password invalid (locked)
- **passwd -u test1**, Unlocking password for user test1
  test1:$6$V1rm7Rko$KkCR4zpM3sZBfAj24w4Nb/logKvm85g12iVd8AZ61gGXPLAdDt71gQCS.SkbCWk7/s9cy5fOIdZTlDouQNIl2.:17577:0:99999:7:::
  !! removed now
  Note: only the root can use **chage**. The one exception to this is that any user can run **chage -l username** to see their aging

------------------------------------------------------------------------------------------------------------------------------------

## Locked Accounts

Linux ships with some **system accounts** that are locked, which means they can run programs but can never login to the system and have no valid password associated with them. For example, **cat /etc/passwd | grep nologin**

- **usermod -L lura** account stay on the system but logging in is impossible, can use su and su – for this user.
- **chage -E 2014-09-11 user** change user expire date, while using usermod -U will unlock user account.
- **chsh -s /sbin/nologin ahmed** block shell for user ahmed

  **Note**: when you access passwd file will notice that some users appear as following **lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin** this means that this user cant access the system.

## Account Defaults

As administrator to make life easier create all options before we create a user.

- **vi /etc/login.defs** user and group id, password options, will notice service account reserve from 201 to 999 IDs, so standard user and group ID start from 1000, also create home directory (-m mentioned) and the default umask
- **useradd -D** show some user defaults (run as root)

  GROUP=100<br>
  HOME=/home<br>
  INACTIVE=-1<br>
  EXPIRE=<br>
  SHELL=/bin/bash<br>
  SKEL=/etc/skel<br>
  CREATE_MAIL_SPOOL=yes
- **vim /etc/default/useradd** also another way to change useradd default settings

## Modify and Delete Accounts

- **usermod -c “user one” user1** add comment to user
- **grep user1 /etc/passwd** notice the comment
- **userdel test1** remove users without removing users home director.
- **userdel -r test1** delete the user's home directory along with the user account itself.

## Managing Local Groups

Identity in Linux system divided to owner, individual user and groups of users that shares permissions and privileges and everybody else often called world

Any Linux user have primary group and up to 15 secondary group,

- **groups**, give you group membership
- **groups username**, group membership for specific user
- **id -Gn**
- **id -Gn test111**
- **getent group,** show all groups on system /etc/group

## Creating Local Groups

In Linux every user is a member of his own group (private group) or primary group

- **grep aabdelwahed /etc/passwd**, show group membership for aabdelwahed user
- **cat /etc/group**, list all groups
- **getent group**, list all groups
- **id**, show more details about user and his group membership (primary and secondary) while secondary is used to give permission
- **groupadd sales**, add new group
- **groupadd -g 555 admins,** creating new group with 555 id
- **grep sales /etc/group**, sales:x:1004:

-------------------------------------------------------------------------------------------------------------------------------

- **grep sales /etc/gshadow**, sales:!:: ! mean your password invalid. (**gshadow** must run as root)
- **getent gshadow**

## Manage Group Membership

- **newgrp**, actually used for switching your group ID, not for creating new group.
  **newgrp wheel**, change your primary group to wheel, so if you go to create new resources while notice that the group owns by **wheel**
  **touch f1, ls -l f1**
- **usermod -G wheel aabdelwahed**, remove all user added secondary groups and add wheel only as secondary group
- **usermod -aG wheel aabdelwahed**, add secondary group (wheel) to an exiting user secondary membership.
- **usermod -aG wheel,admins user02**, add user02 to wheel and admins groups
  Note: usermod is one way, so last command will overwrite any secondary group membership for selected user.
- **gpasswd -a user01 wheel**, adding user01 to wheel group. Also, will save exiting user group membership
- **gpasswd -M user01,user02,user03 sales,** adding multiple user to sales group, will overwrite exiting members
- **gpasswd -d user01 sales**, remove user01 from sales group
- **gpasswd -A user01 sales,** add user01 admin for sales group
- **groupmems -g wheel -l**, list wheel members
- **gpasswd –help**
  **Note**: to see full membership list for specific user who just added to a new group logout or use su – user name (Full Login) which refresh group membership. Now run **id**

## Modify and Delete Groups

- **groupmod -n finance sales,** change sales group name to finance group
- **groupmod -g 1100 hr,** change hr group id to 1100
- **groupdel finance,** delete finance group
  **Note**: usermod used only by root, while standard user can use newgrp to change his group membership

## Group Passwords

if you working as standard user and run **newgrp sales** to login using other group membership, you asked about password this password for group password not user or root password. To assign password for group just run **gpasswd sales** (as root). Now you can add any user to this group through **newgrp** with group assigned password.

## User and Group Configuration Files

- **cat /etc/passwd,** give you all users with properties
- **cat /etc/shadow,** give password properties (hash represent the password)
- **cat etc/groups,** give you group name and id and users
- **vim etc/logon.defs,** generic configuration files for users and groups
- **vim /etc/default/useradd**

## Switch Users and Sudo Access (su, sudo)

### Using su (Substitute User or Switch User)

- **$** you access as standard user while **#** you access as root
- **su username** switch to specific user, if use only **su** will switch to root
- **su - = su -l** provide full login, run **pwd** after connect through **su** only and after su –
- recommended to use **sudo** not **su**

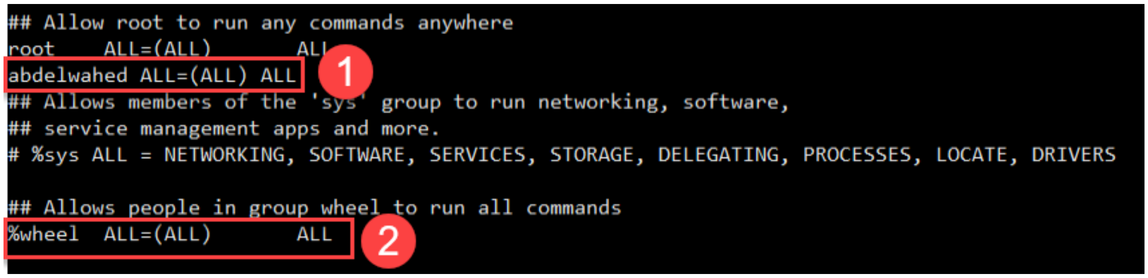
### Implementing sudo

- With **sudo** you can specify who's allowed to run which command and we don't need to root password. When use <u>**sudo** your password saved for 5 minutes.</u>

## Visudo (Enable Sudo)

1- Enable **%wheel ALL=(root) ALL**, and add **aabdelwahed ALL=(root) ALL**
**Defaults timestamp_timeout=0** will disable 5 minutes and prompt user
2- Enable wheel as mentioned in first way then you can add the user direct to wheel group by running **usermod -G wheel aabdelwahed**

- **systemctl restart sshd**



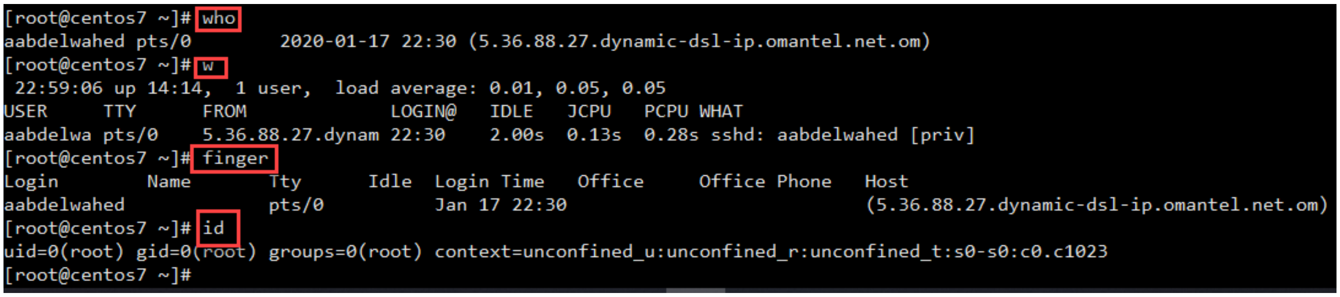
```
## Allow root to run any commands anywhere
root    ALL=(ALL)       ALL
abdelwahed ALL=(ALL) ALL
## Allows members of the 'sys' group to run networking, software,
## service management apps and more.
# %sys ALL = NETWORKING, SOFTWARE, SERVICES, STORAGE, DELEGATING, PROCESSES, LOCATE, DRIVERS

## Allows people in group wheel to run all commands
%wheel  ALL=(ALL)       ALL
```

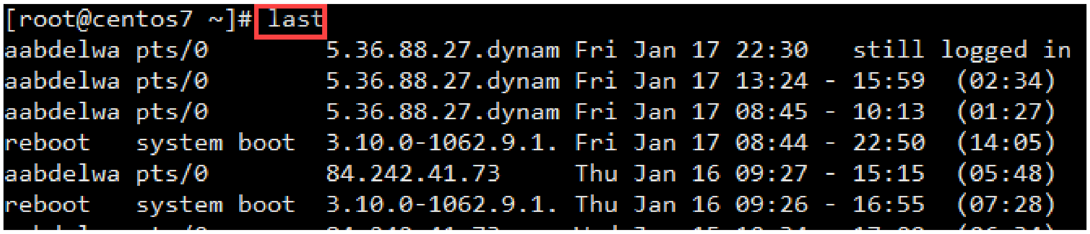
## Monitor Users

To monitor what users, do at our system.

- **who** tell you some details about the current logged in the system like user name and public IP and which ISP used.
- **last** tell you about logged in history
- **w** more details about current logged in users
- **finger** first you have to install it (**yum install finger -y**)
- **id** give you information about yourself



```
[root@centos7 ~]# who
aabdelwahed pts/0        2020-01-17 22:30 (5.36.88.27.dynamic-dsl-ip.omantel.net.om)
[root@centos7 ~]# w
 22:59:06 up 14:14,  1 user,  load average: 0.01, 0.05, 0.05
USER     TTY      FROM             LOGIN@   IDLE   JCPU   PCPU WHAT
aabdelwa pts/0    5.36.88.27.dynam 22:30    2.00s  0.13s  0.28s sshd: aabdelwahed [priv]
[root@centos7 ~]# finger
Login       Name       Tty      Idle  Login Time   Office     Office Phone   Host
aabdelwahed            pts/0          Jan 17 22:30                           (5.36.88.27.dynamic-dsl-ip.omantel.net.om)
[root@centos7 ~]# id
uid=0(root) gid=0(root) groups=0(root) context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
[root@centos7 ~]#
```



```
[root@centos7 ~]# last
aabdelwa pts/0        5.36.88.27.dynam Fri Jan 17 22:30   still logged in
aabdelwa pts/0        5.36.88.27.dynam Fri Jan 17 13:24 - 15:59  (02:34)
aabdelwa pts/0        5.36.88.27.dynam Fri Jan 17 08:45 - 10:13  (01:27)
reboot   system boot  3.10.0-1062.9.1. Fri Jan 17 08:44 - 22:50  (14:05)
aabdelwa pts/0        84.242.41.73     Thu Jan 16 09:27 - 15:15  (05:48)
reboot   system boot  3.10.0-1062.9.1. Thu Jan 16 09:26 - 16:55  (07:28)
```

# Control access to files with Linux file system permissions

## Files and Directory Permissions (chmod)

To protect your environment (files and directories) from being deleted or access by another user.

It is very important on a Linux system to control exactly who has access to any given file or directory on the system, and how they might make use of them.

When you do **ls -l**, list directory and files in this directory with information. there are nine more which indicate the access right granted to file divided into 3 parts first one permission for **owner**, second one permission for **groups** and last one permission for **others** (reset of the **world**)

## File Access Rights

- **R:** read access is allowed
- **W:** write access is allowed
- **X:** execute access is allowed

If the permission is not allowed, a – appears instead of these characters.

| Right Access | File | Directory |
|---|---|---|
| R | cat, less, more, tac | ls |
| W | echo, cat, vim | mkdir, rm |
| X | Execute | cd |

## File Permissions and Security and Authentication

File authentication is granted depending on one of these three sets of permissions, in the following order:

1. If the requester is the file owner, the file **owner** permissions are used.
2. Otherwise, if the requester is the member of **group** that owns the files, the group permissions are examined.
3. If that doesn't succeed, the **world** permissions are examined.

## Alphabet vs Numerical syntax for permissions

4 = read
2 = write
1 = execute
so, 7 for rwx, 6 for read/write, 5 for read/execute

## Listing Permissions

- **ls -l file01**
- **stat file01** get full metadata, notice will show you permission as numeric.
- **stat -c %A file01** permissions symbolic
- **stat -c %a file01** permission numeric
- **stat -c %a test_*** show numeric permissions for all dir and files that start with test_

## Managing Default Permissions (Umask)

We have a default permission which affected by **umask** so umask can adjust defaults**, (Default file 666, Default directory 777)** The default permissions given when creating a file are read/write for owner, group and world (0666). And for a directory it is read/write/execute for everyone (0777). However, if you create a new file and a new directory, the applied permissions changes to 664 for the file and 755 for the directory, they have been modified by the current **umask**, just use umask command, 0002 so this value is combined with the file creation permissions to get the actual result 0666 & 0002 = 0664, you can change umsk as in umask 0022

- **umask 0** change **umask** to **0000**

-------------------------------------------------------------------------------------------------------------------------------------

- **mkdir d1**
- **touch f1**
- **ls – l** check change in permission
- **stat -c %a file01** check numeric permission for **file01**
- **stat -c %a d1 d2** check numeric permission for **d1** and **d2**
- **umask 27** change **umask** to **27** then try to create again **f3** and **d3** and check permission changes

  **Note:** to change umask permanently **vim /etc/bashrc** and write **umask=555** for example

## Changing Permissions

Changing file permissions is done with **chmod** with **+ - =**. You can change permissions on files you own, unless you are the superuser.

- **chmod 467 file01** change f1 permission
- **chmod u=r,g=rw,o=rwx file01,** where u stands for user (Owner), **o** stands for other (world), and **g** stands for group.
- **chmod a=rwx file01**
- **chmod 777 f1**
- **chmod a=rwx f2**
- **chmod go=rwx f3**
- **chmod uo+x, g-w file01**
- **chmod g+s /data/profs** Getting new files to inherit group permissions on Linux

**Note**: permission in Linux **not cumulative** (didn't added together), so if I match on the user ID, group ID doesn't check, if I don't match on the user ID, the group ID will be checked. But if I match on the group ID, I don't check for others.

## Managing File Ownership

Changing file ownership is done with **chown** and changing the group is done with **chgrp**. Only the superuser can change ownership on file. Likewise, you can change group ownership to groups that you are a member of.
Every file has an owner, group owner and user owner,

- **ls -l f1** show file owner (user and group)
- **id -un** show user name while **id -u** show user ID
- **id -gn** show primary group name
- **id -Gn** show secondary groups

**Note**, primary group is used to creating resources, while secondary group is used to accessing these resources.

- **chgrp wheel file01** change file01 group to wheel
- **chown :Sales file01** change file01 group to sales
- **chown user1 file01** change user owner to user1 (only root can do it)
- **chown user1: Sales file01** change user and group owner for file1
- **chown -R user1: Sales./** will change the owner and group of all files in the current directory and all its subdirectories.
- **usermod -aG sales user1,** add user1 to sales secondary group

now if you copy this file to another directory the user and group ownership will change to copy file and maintained user and group ownership use the following command (need root privilege)

- **cp -a f1 /root/file1**

# Finding files

find searches in the real system. Is slower but always up-to-date and has more options (size, modification time, user, ...) also you can run excute command on the result.

locate uses a previously built database (command **updatedb**). Is much faster, but uses an 'older' database and searches only names or parts of them

- **find / -name file*** search in / for all file that start with file
- **find /tmp -type f -empty** find all empty files
- **find /tmp -type d -empty** find all empty directories
- **find /usr/share/doc -name '*.pdf'** find all pdf files in **doc**
- **find /etc -name "*.conf"**
- **find /etc -maxdepth 1 -type l for** symbolic links
- **df -h /boot**
- **find /boot -size +10000k -type f**
- **find / -size +1G -exec cp {} /data/sdc2/ \;** search for files with more than 1 GB and copy result to /data/sdc2 dir
- **find -type f -perm 644** find file that have permission 644
- **find -type f -perm 777 -delete** Find Files With 777 Permissions then delet it
- **find / -type f ! -perm 777** Find Files Without 777 Permissions
- **find / -type f -perm 0777 -print -exec chmod 644 {} \;** Find Files with 777 Permissions and Chmod to 644
- **find -type d -exec chmod 777 {} \;** find all dirs. And change permissions to 777
- **find /tmp/ -user root** find all filles owned by root in /tmp
- **find -type d -name dir*** find directory that name start with dir
- **find /usr/share/doc -name '*.pdf' -exec cp {} . \;**
- **find /boot -size +10000k -type f -exec du -h {} \;**
- **find / -size +50M -size -100M, Find** Size between 50MB – 100MB
- **find -name '*.pdf' -delete**
- **find dir* -delete** find and delete all start with dir
- **find . -type f -name "*.txt" -exec rm -f {} \;** Find and remove Multiple File
- **find / -mtime 50** Find Last 50 Days Modified Files
- **find / -atime 50** Find Last 50 Days Accessed Files
- **find / -cmin -60** Find Changed Files in Last 1 Hour
- **find / -amin -60** Find Accessed Files in Last 1 Hour
- **find / -type f -name *.mp3 -size +10M -exec rm {} \;** Find Specific Files and Delete
- **find / -name "*.img" -size +999M -size -1025M -amin 8 -exec chmod 777 {} \;**
- **locate –regex readme**
- **grep -i "tom" /etc/passwd i** to ignore case sensitive
- **whereis ls** locate the binary, source, and manual page files for a command **ls**
- **whereis -b cat** locate cat binzary files
- **whereis -m cat** locate cat man pages
- **which** shows the full path of (shell) commands.
- **Which ls**
- **Which rm**

# Monitor and manage Linux processes

## System Admin Tasks

1- List Process using **CLI** through **ps** and **pstree**
2- Monitor Process using **GUI** and **TUI** through **top**
3- Control Process using **CLI** (**kill**, **pkill** and **killall**) and **TUI** (**top**), you can monitor and control process through **top**

## Listing Processes with ps

- **man ps**
- **ps aux** show all process (long list)
- **ps aux | wc** how many numbers of process
- **ps -le | grep bash** check bash PID
- **dd if=/dev/zero of=/dev/null** & job that copy nothing to nowhere (use **&** to run process in background)
- **sleep 600 &,** run job in background direct
- **ps -le | grep dd** check parent process for dd (will be back PID)
- **top** then press k to kill dd process
- **ps -elf** long and full list with PPID
- **ps -o pid,uid,cputime,pmem,command** customizing the ps output
- **ps aux**, most process that running with system
- **ps aux | head**

  **PID** process unique number
  **%CPU** amount of cpu usage
  **%MEM** amount of memory usage
  **VSZ** virtual memory size
  **RSS** resident set size (reserved memory)
  **TTY** terminal that process run on it, if process run in background tty appear as **?**
  **STAT** current process state, most of them in sleep mode s, R (running process)
  **START** start time
  **TIME**
  **COMMAND** associated command
- **ps aux | grep http** gives http process
- **ps fax** give process with its child
- **firefox, ctrl + z** will stop firefox (freeze not closed) start again type, **bg**

## Sending signal to processes

signal are instructions that send to processes.

- **man signal**
- open **top** and hit **k** and follow the step to kill the process
- **kill 2907** kill process ID 2907
- **killall dd** kill all processes that have name **dd**
- **SIGINT** interrupt signal using **ctrl-c** the default behavior is to terminate the process, but it can be caught or ignored.
- **SIGTERM** termination signal the default behavior is to terminate the process, but it also can be caught or ignored. The intention is to kill the process, gracefully or not, but to first allow it a chance to cleanup.

---

## Kill means terminate

- **kill -l**, list all signals that I can send and we can send signals by number or long word.
  All of the following signals are equivalent that asking nicely to close down but **didn't force just askin**
- **kill 8537**
- **kill -15 8537**
- **kill -term 8537**
- **kill -sigterm 8537**

to force kill process use one of the following signals:

- **kill -9 8537**
- **kill -kill 8537**
- **kill -sigkill 8537**

**Note:** standard user can see through ps aux all running process from all users, but only can kill his own process.

## Shortcuts with pgrep and pkill

Before we using grep to indicate specific process so we can kill it easily, using pgrep and pkill make this easier.

- **pgrep sshd** show all sshd process **PID**
- **ps -F -p $(pgrep sshd)** with this way we can get an extra full listing of
- **sleep 100** create sleep process running in the background for 100 sec.
- **pkill sleep**
- **pkill -KILL -u user** kill user process (logout user)
- **kill pgrep sshd**
- **killall sshd** kill all process sshd process
- **kill -q PID** force kill

## Process Priority

first come first served, every process has the same priority but i can adjust priority (lower) to process to serve first.

- **top** is a monitoring tool, which Monitoring system active processes on your system at real time. current running processes sorting based on processor utilization, if you want to change this press f and select another factor like memory.
  - use f to add other fields like PPID
  - use z to color results
  - use h for more help
  - use k to kill process
  - use i to show only show only active processes
  - use r for renice
  - use **q** to quit
- **top** notice that there is column called PR this priority number
- open **top** and press **r** will ask you about **PID** for process that you want to change priority to it then ask you about priority that you want to assign (20 is highest) write -5 to change priority to 15
- **nice --help**
- **nice -n 10 dd if=/dev/zero of=/dev/null** create job with custom nice value 10
- **renice -n 19 --pid 128359** modify nice value to 19
  **Note**, top refresh itself every 3 sec

---

## Backgrounding Tasks

- **sleep 1000** create and running processes for 1000 sec (in foreground) we can use **ctrl+z** to suspend these processes but still using the memory but doesn't use CPU time.
- **sleep 1000&** create and running processes for 1000 sec (in Background)
- **jobs** show all running and suspend processes.
- **bg** using this command will resume the jobs in the background.
- **fg** bring the job to the foreground.
- **fg 1** bring processes number 1 to foreground
- **ps -p 13732** show more info about process ID 13732 processes
- **ps -F 13732**

  **Note**: we only using sleep to get a long running process

## Using Nice to Control Process Priority

**Nice Value**

Lowest nice value, highest priority

-20 >>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>> -1 0 1 <<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<< +19

**Highest periorty** **Normal value** **Lowest periorty**

Note: minus value reserved for root while positive value reserved for regular user

- **yum install psmisc** required package

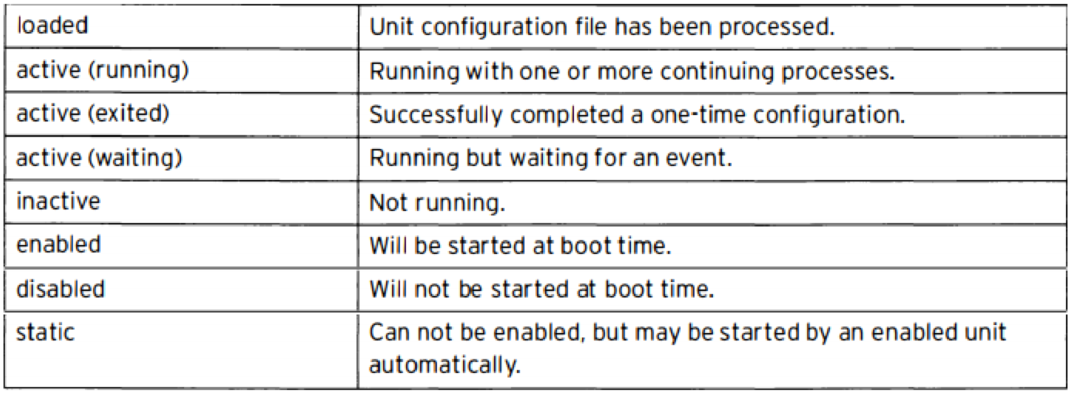
First, create background processes to use in test, **sleep 1000&** then try to run **ps -l** to get long list processes that contain priority and nice value for a given application. The priority column PRI indicates the CPU priority that process has. This is controlled by the nice value, we can see the default nice value of **zero** which give the priority of **80**, we only can configure nice value from **Nice value -20 to +19** while +19 will gives lower priority

- **nice -n 5 firefox&**
- **renice -n 10 -p 14721**

**Note**, only root user can set - (minus) nice value and also only root can set lower nice value. And root can set default nice value by user or group by configure the following:

**vi /etc/security/limits.conf** then add this line at the end of file **user hard priority 7** now test it by using user full logging shell **su -user**

## Control services and daemons



| | |
|---|---|
| loaded | Unit configuration file has been processed. |
| active (running) | Running with one or more continuing processes. |
| active (exited) | Successfully completed a one-time configuration. |
| active (waiting) | Running but waiting for an event. |
| inactive | Not running. |
| enabled | Will be started at boot time. |
| disabled | Will not be started at boot time. |
| static | Can not be enabled, but may be started by an enabled unit automatically. |

- **systemctl status sshd.service,** check service state
- **systemctl --type=service,** check all services
- **systemctl --type=service | grep active | wc -l** show total number of active process
- **systemctl is-active sshd**
- **systemctl is-enabled sshd**
- **systemctl --failed --type=service**
- **systemctl stop sshd. service**
- **systemctl start sshd. service**
- **systemctl restart sshd. service**
- **systemctl reload sshd.service**
- **systemctl reload-or-restart sshd.service,** the command reloads the configuration changes if the reloading functionality is available. Otherwise, the command restarts the service to implements the new configuration changes
- **systemctl list-dependencies sshd.service**, displays a hierarchy mapping of dependencies to start the service unit

### Masking services

you can mask any service unit to prevent it from being started manually or by another service. To do so, run the following command as root:

- **systemctl mask *name*.service**

This command replaces the /etc/systemd/system/name.service file with a symbolic link to /dev/null, rendering the actual unit file inaccessible to systemd. To revert this action and unmask a service unit as root or using sudo

- **systemctl unmask *name*.service**

# Configure and secure SSH

## Configure SSH

- **yum whatprovides */sshd**
- **yum install openssh**
- **vi /etc/ssh/sshd_config**, configure it to work through port 1414
- **firewall-cmd –zone=public –add-port=1414/tcp --permenant**
- **semanage port -a -t ssh_port_t -p tcp 1414** allow 1414 port from selinux in case selinux enabled.
- **firewall-cmd –reload**
- **systemctl enable sshd**
- **systemctl start sshd**
- **semanage port -l | grep sshd**

## Secure SSH

For security consideration its better to restrict root to access through SSH to do this follow the instructions below:

- **cp /etc/ssh/sshd_config /etc/ssh/sshd_config.bak**
- **vi /etc/ssh/sshd_config**
- **LoginGraceTime 30**, to restrict time if hacker access the server first thing to do is trying to get root password
- **PermitRootLogin no**
- **Port 1414**, change default port

## Configure SSH Keys

On remote server (Connect from)

- **ssh-keygen**
- **cat /home/ahmed/.ssh/id_rsa** contains private key
- **cat /home/ahmed/.ssh/id_rsa.pub** contains public key
- **ssh-copy-id -i ~/.ssh/ id_rsa.pub user@host** copy public key from remote server to the target host.
- **sudo systemctl restart sshd**
- **ssh user@host**

## Disable root login and password-based login

- **vi /etc/ssh/sshd_config** and change the following

```
ChallengeResponseAuthentication no
PasswordAuthentication no
UsePAM no
```

- **systemctl reload sshd**

## Analyze and store logs

### System Logs Monitor -logging- (/var/log)

Most services used on a Linux server write information to log files. Tell us the story of what’s happening on your servers and whether it’s good or bad.

### Common Linux log files names and usage

- **/var/log/messages** General message and system related stuff
- **/var/log/auth.log** Authenication logs
- **/var/log/kern.log** Kernel logs
- **/var/log/cron.log** Crond logs (cron job)
- **/var/log/maillog** Mail server logs
- **/var/log/qmail/** qmail log directory (more files inside this directory)
- **/var/log/httpd/** Apache access and error logs directory
- **/var/log/lighttpd/** Lighttpd access and error logs directory
- **/var/log/boot.log** System boot log
- **/var/log/mysqld.log** MySQL database server log file
- **/var/log/secure** or **/var/log/auth.log** authentication log
- **/var/log/utmp** or /**var/log/wtmp** login records file
- **/var/log/yum.log** yum command log file.

### Configuration file

- **etc/rsyslog.conf**

### Understanding Facilities, Priorities, and Log Destinations

To specify what information should be logged to which destination, rsyslogd uses facilities, priorities, and destinations:

■ A *facility* specifies a category of information that is logged.

■ A *priority* is used to define the severity of the message that needs to be logged.

■ *A destination defines where the message should be written to.*

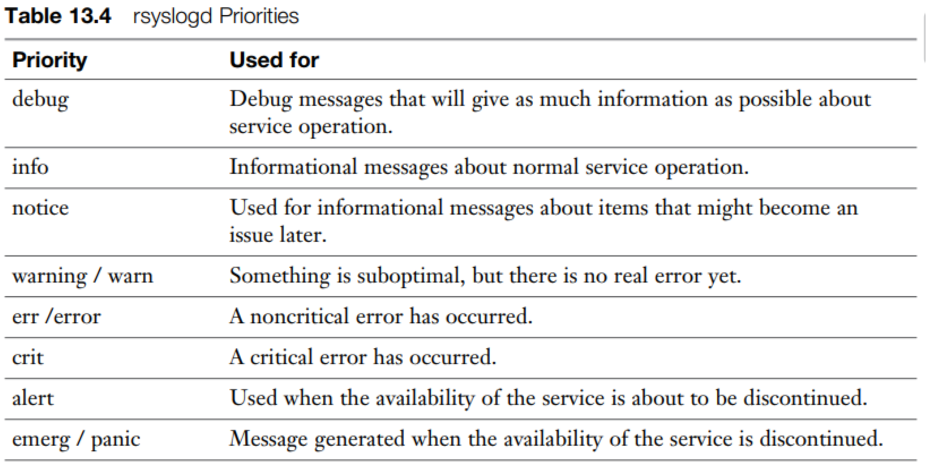


**Table 13.4** rsyslogd Priorities

| Priority | Used for |
|---|---|
| debug | Debug messages that will give as much information as possible about service operation. |
| info | Informational messages about normal service operation. |
| notice | Used for informational messages about items that might become an issue later. |
| warning / warn | Something is suboptimal, but there is no real error yet. |
| err /error | A noncritical error has occurred. |
| crit | A critical error has occurred. |
| alert | Used when the availability of the service is about to be discontinued. |
| emerg / panic | Message generated when the availability of the service is discontinued. |

For example, log all warning message to /var/log/warnings file and all errors to /var/log/errors

- vi /etc/rsyslog.conf

```
#log all warning messages to warning file
*.warn                              /var/log/warnings
*.err                                /var/log/errors
```

- **systemctl restart rsyslog**

these files will automatically create once first logs comes

## Remote logs

### Server configuration (Receive logs)

**1- allow 514/tcp 514/udp on firewall**

**2- vim /etc/rsyslog.conf  in module section:**

**uncomment tcp and udp syslog reception**

- add the following lines to create separate directory for each server

**$template DynamicFile,"/var/log/%HOSTNAME%/forwarded-logs.log"**

***.* -?DynamicFile**

**3- systemctl restart rsyslog**

-----------------------------

### Client configuration (send logs)

**1- vim /etc/rsyslog.conf**

```
*.warn              @logserver
*.*                 @logserver
```

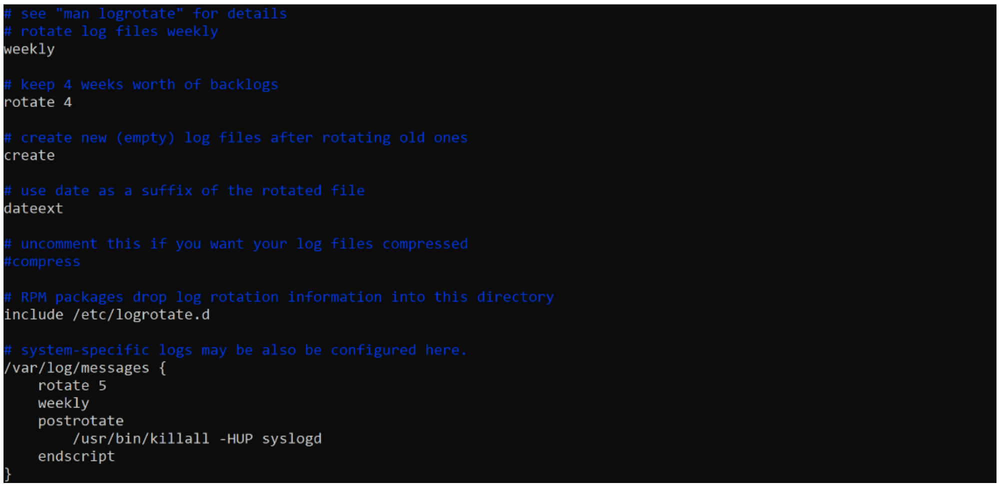
**2- systemctl restart rsyslog.service**

## logrotate

**logrotate** is designed to ease administration of systems that generate large numbers of log files. It allows automatic rotation, compression, removal, and mailing of log files. Each log file may be handled daily, weekly, monthly, or when it grows too large. **Normally**, logrotate is run as a daily cron job. It will not modify a log multiple times in one day unless the criterion for that log is based on the log's size and logrotate is being run multiple times each day, or unless the -f or --force option is used.
The following configuration applied to all logs (global options), you can un-comment compress so all rotated logs will be compressed.

If you want to set different settings for specific logs add the following lines to logrotate configuration file
**vim /etc/logrotate.conf**

The next section of the config files defined how to handle the log file **/var/log/messages**. The log will go through five weekly rotations before being removed. After the log file has been rotated (but before the old version of the log has been compressed), the command **/sbin/killall -HUP syslogd** will be executed to **restarts** and **re-reads** all syslogd configuration.



```
# see "man logrotate" for details
# rotate log files weekly
weekly

# keep 4 weeks worth of backlogs
rotate 4

# create new (empty) log files after rotating old ones
create

# use date as a suffix of the rotated file
dateext

# uncomment this if you want your log files compressed
#compress

# RPM packages drop log rotation information into this directory
include /etc/logrotate.d

# system-specific logs may be also be configured here.
/var/log/messages {
    rotate 5
    weekly
    postrotate
        /usr/bin/killall -HUP syslogd
    endscript
}
```

# Managing Red Hat Enterprise Linux Networking

## IP vs. IFCONFIG

- **ip** is the preferred command line utility as compared to **ifconfig** because it uses **netlink** sockets than **ioctl** system calls. It can be used to configure, control and query devices and interface parameters, as well as manipulate routing, policy-based routing, and tunneling.
  - **ip link** show information about all network interfaces
  - **ip -s link show eth0** s for statistics show information for the eth0 network interface
  - **ip a show eth0** or **ip a s enos3**
  - **ip link set eth0 down**
  - **ip addr** = **ip a** display information about interfaces
  - **ip** show ip help
  - **ip addr add 192.168.1.100/24 dev eth0** add temporary IP
  - **ip r** show route
  - **ip route add 172.16.1.0/24 via 192.168.1.1** add temporary default gateway
  - **ip n** show neighbors
  - **ip -4 a s em1**, show em1 interface only
  - **ip -6 a** note fe80 refer to local address
  - **ip r s** show routing table

## Network Configuration file

**/etc/sysconfig/network**
**/etc/sysconfig/network-scripts/ifcfg-ethx**

## Name Resoluation

To show your DNS server use **cat /etc/resolve.conf**

## The Local Hosts File

here we will go to add our system name to the local resolution file (hosts)

- **cat /etc/hosts** list all records in name resolution file
- **vim** !$ to run the command on last argument use !$
  at the same file (hosts) you can add alias or multiple alias for the same server
  **172.16.0.249 abdelwahed.me d1**

## Storing Network Configuration Persistently (Fixed)

**vim /etc/sysconfig/network-scripts/ifcfg-ens33**

configure network setting from file
NAME=ens33, network card interface
ONBOOT=yes, network card is enabled at booting
IPADDR0=x.x.x.x, 0 is refer to first ip you can use IPADDR1 to select second ip
BOOTPROTO=none, dhcp disabled or BOOTPROTO="dhcp"
NETMASK=255.255.255.0
GATEWAY=y.y.y.y
DNS1=y.y.y.y
DNS2=y.y.y.y
USERCTL=yes
HWADDR='your mac address'

- **vim /etc/hostname**, machine Name
- **vim /etc/ressolve.conf**
- **systemctl restart NetworkManager**

## Network Files and Commands

- -**tcpdump** monitor all coming and outgoing traffic, for testing run the command then open brswer and see changes.
- **ifup, ifdown** to drop and up your NIC
- **netstat** (network statistics) is a command line tool for monitoring network connections both incoming and outgoing as well as viewing routing tables, interface statistics etc.
- **netstat -a | more** Listing all the LISTENING Ports of TCP and UDP connections
- **netstat -at** Listing TCP Ports connections
- **netstat -au** Listing UDP Ports connections
- **netstat -s** Showing Statistics by Protocol
- **netstat -tp** Displaying Service name with PID
- **netstat -r** Displaying Kernel IP routing
- **netstat -g** Displaying IPv4 and IPv6 Information

## Downloading Files or Apps (wget)

**wget + fileurl Path**

## curl and ping commands

these tools used to test communication

- **curl** www.google.com browse google site if replay with content so google is up and you can reach.
- **curl -O** www.site/filename used for download the same function of wget in case wget not supported at your system.
- **ping** www.google.com
- **ping -c 5** www.google.com to controlling the number of pings

# Archive and transfer files

## Using **tar** to Archive (Backup) and restore files

- du -h /etc/
- **c** create
- **x** extract
- **v** for verbose printing out what we are backing up
- **f** name of archived or compressed file
- **t** for list (--list) archive contents without extracting
- **C** extract archive in another place
- **tar cvf home.tar /home/ c** for –create, **f** for –file
- **tar cvf etc.tar /etc/**
- **du -h home.tar** gives file size
- **file home.tar** gives file type
- **tar tf home.tar**
- **tar -tf etc.tar | grep rsyslog**
- **tar xvf home.tar** unarchive in the same place
- **tar xvf home.tar -C /mnt/** exctract to different location
- **tar xpvf home.tar -C /mnt/** extract with saving the same permission
- **tar –extract –same-permissions –verbose –file home.tar -C /mnt/** same previous command
- **tar zcvf /lab01/home.gz /home/** compress in a different location using gz
- tar zxvf home.gz -C /save exctract gz file to different location
- **tar jcvf /lab01/home.bz2 /home/** compress in a different location using bzip2
- tar jxvf home.bz2 -C /save exctract bz2 file to different location
- **tar Jcvf /lab01/home.xz /home/** compress in a different location using xz
- tar Jxvf home.xz -C /save exctract xz file to different location

## Backup and restore /home

- **tar zcvf home.tar.gz /home** backup home
- login with root
- **tar zxvf home.tar.gz -C /**

## Using Compression Using GUN Zip

- **gzip arch.tar** compress tar file
- **gunzip arch*** uncompress gz files
- **zcat** to read compressed file
- **zless** to read compressed file

## Compressed using tar ball

- **yum install bzip***
- **bzip2 file,** compress
- **file file1.bz2**
- **bunzip2 file*** uncompressed bz2 files

---

## Compressed Using xz

- **xz passd** compress passwd file
- **xz -k passwd,** compress and keep source file
- **xz f1.txt f2.txt f3.txt** compress multiple files
- **xz -d passwd.xz = unxz passwd.xz** decompressed **xz** files
- **unxz -k passwd.xz** decompress and keep the orginal .xz
- **xz -l passwd.xz** List compression information
- **xzcat passwd.xz** list **xz** file
- **xz -k6ev centos7.iso** k to keep source, **v** for verbose, **e (Extrem Mode)** to use more CPU when encoding to increase the compression ratio, however this will take more time. **-6** compress ratio (0-9) while -9 take long time and increase the compression ratio.

## Copying Files using rsync

**Rsync (Remote Sync)** is a most commonly used command for **copying** and **synchronizing** files and directories **remotely (across a network)** as well as **locally** in **Linux/Unix** systems, with a special delta transfer algorithm.

- **rsync options source destination** basic rsync syntax
- **yum install rsync -y**
- **rsync -zvh backup.tar /tmp/backups/** Copy/Sync Files and Directory Locally, **z** for compress **h** for human readable
- **rsync -avzh /root/rpmpkgs /tmp/backups/** Copy/Sync a Directory on Local Computer, **a** for archive
- **rsync -avz rpmpkgs/ root@192.168.0.101:/home/** Copy a Directory from Local Server to a Remote Server
- **rsync -avzh root@192.168.0.100:/home/source/rpmpkgs /tmp/myrpms** Copy/Sync a Remote Directory to a Local Machine
- **rsync -avzhe ssh --progress /home/rpmpkgs root@192.168.0.100:/root/rpmpkgs** Show Progress While Transferring Data with rsync
- **rsync -avzhe ssh --max-size='200k' /var/lib/rpm/ root@192.168.0.100:/root/tmprpm** this command will transfer only those files which are equal or smaller than 200k with ssh
- **rsync --remove-source-files -zvh backup.tar /tmp/backups/,** Automatically Delete source Files after successful Transfer
- **rsync --dry-run --remove-source-files -zvh backup.tar /tmp/backups/ --dry-run** this option will not make any changes only do a dry run of the command and shows the output of the command
- **rsync --bwlimit=100 -avzhe ssh /var/lib/rpm/ root@192.168.0.100:/root/tmprpm/ '–bwlimit'** option. This option helps us to limit I/O bandwidth while transferring data from one machine to another machine.

## Copy Files Securely (Secure copy using ssh)

- **scp -r aabdelwahed@104.214.49.143:/etc/passwd .** Copy from remote to local, The -r option can be used to copy directories recursively.
- **scp username@from_host:/remote/directory/file.txt username@to_host:/remote/directory/** copy from remote to remote
- **scp file01 file02 file03 /repo,** copy multiple files locally
- **scp file1 file2 … fileN user@host:/destination/directory/** copy multiple files remotely
- **scp:22 172.16.0.228:/etc/hostname .**
- **scp /etc/hosts server1:/tmp**
- **scp -r some_dir server1/tmp/some_dir** where **r** option to transfer files recursively

---

## Install and update software

In red hat to install new package you have to link your lab with one red hat subscription like developer subscription or create local repo

### Create local repo

1. Check version Linux server **cat /etc/redhat-release**
2. Create folder repository **mkdir /repo**
3. Copy RedHat Iso contents to **/repo**
4. Edit repo config file. **vim /etc/yum.repos.d/public.repo**
5. add new line

```
[redhat1_repo]
name=redhat2_repo
baseurl=file:///repo/BaseOS
enabled=1
gpgcheck=1
gpgkey=file:///repo/RPM-GPG-KEY-redhat-release
```

### find rpm online

Rpmfind mirror

### RPM Queries

even you still not use rpm to install packages rpm still important as shown:

- **rpm -qi bash** give everything in package including Docs
- **rpm -qc bash** give configuration file
- **rpm -qf /bin/bash give all related files to bash**
- **rpm -qd bash** give all documentation
- **rpm -ql bash** give location
- **rpm -qa** or **rpm –query –all** gives all installed packages
- **rpm -qa | grep http** give http version installed
- **rpm -qi httpd-2.4.6-67.el7.centos.6.x86_64** (note must type the full name of rpm)
- **rpm -q --whatprovides vim** return with full package name
- **rpm -Va** allows you to verify whether the files from a particular package are consistent with the system's **RPM** database
- **rpm -V bash** no output when everything is OK (verifying packages)

### installing and uninstall packages using RPM

- **rpm -ivh http** i is for install, v for verbose, h for print out hash marks while doing show progress.
- **rpm -ivh --nodeps** https://www.rpmfind.net/linux/opensuse/ports/aarch64/tumbleweed/repo/oss/aarch64/finger-1.3-169.2.aarch64.rpm online install with ignoring dependencies
- **rpm -e http** e for erase
- **rpm -e httpd-2.4.6-67.el7.centos.6.x86_64 –test** --test to determine whether the uninstall would succeed or fail

## Upgrading and freshen Packages

**Note** that -**I** option is not designed for upgrades, instead use -**U** for upgrade. When upgrading, the already installed package is removed after newer version is installed. Also, if the package is not installed will installed by running -U oprion.

- **rpm -Uvh bash-4.2.45-5.el7_0.4.x86_64.rpm**
- **rpm -Fvh *.rpm**, to freshen all packages

## Using yum

**yum** provides a frontend to **rpm**. Its primary task is to fetch packages from multiple remote repositories and resolve dependencies among packages. yum caches information and databases to speed up performance. To remove some or all cached information run **yum clean all**

## Using yum queries

- **yum info httpd**
- **yum search httpd**
- **yum install httpd**
- **yum install screen**
- **yum list all** list all packages that available
- **yum list installed** give all installed packages
- **yum whatprovides */httpd** or yum provides */httpd
- **yum remove bash**
- **yum check-update** check available update
- **yum update** update system
- **yum remove – y** yes not ask you (disable user interaction)
- **yum -y install** also yes not ask you
- **yum install –downloadonly dhcp --downloaddir=/yum** download only without installation in /yum directory
- **yum grouplist**
- **yum groupupdate**
- **yum groupremove**
- **yum repolist** show all enabled repo
- **yum repolist all** list all repo list
- **sudo yum --enablerepo=epel install Nagios** install from specific repo
- **yum --disablerepo="*" --enablerepo="epel" list available**
- **yum localinstall /yum/tree-1.6.0-10.el7.x86_64.rpm** install local downloaded rpm package
- **yum localupadte /yum/tree-1.6.0-10.el7.x86_64.rpm**
- **yum downgrade /yum/tree-1.6.0-10.el7.x86_64.rpm**
- **yum reinstall /yum/tree-1.6.0-10.el7.x86_64.rpm**
- **yum --disablerepo Centos_Repo install libgtk* --skip-broken**
- **yum history**
- **yum history undo 8** undo used for Rollback

# Access Linux files systems

## Examining file systems

- **df -h** get an overview about the file system mount points and the amount of free space available
- **du /root,** disk used
- **du -h /root** h for human readable

## Mounting and Unmounting File Systems (on fly)

- **blkid**, overview o f existing partitions with a file system o n them and the UUID o f the file system
- **mount /dev/vdb1 /mnt/mydata**, Mount by device file of the partition
- **mount UUID="46f543fd - 7Bc9 - 4526 - a857 · 244811be2dBB" /mnt /mydata**, Mount the fi l e system by universal unique id
- **umount /mnt/mydata**
- **iosf /mnt/mydata,** if you can't unmount because target busy use this command to check which process running on target then kill it so you can unmount.

## Making Links Between Files (Soft and Hard Links)

First, we need to talk about inode, inode is a pointer or number of a file on the hard disk. File name only for me but system doesn't understand names only understand numbers. So everytime you create a file computer assign number for this file this number called inode and everytime we try to retrieve the file we retrieve the number

## Inodes (index node)

every Linux file and directory have a unique number called Inode

- **ls -li** to see inode
- **stat f1** show metadata including inode
    1. hard link act as secondary name for the file so no different between hard link and original file with the same inode number.
    2. symbolic link (soft) act as shortcut so if you remove or rename source you can't access symbolic link, and take different inode number

Deleting renaming or moving the original file will not affect the hard link.

- **ln /etc/hosts computers** create hard link for hosts with new name computers
- **ls -il /etc/hosts computers** display properties of hosts and computers
- **ln -s computers newcomputers** ls -s sourcefile DestinationFile, create symbolic link
- **ls -il /etc/hosts computers newcomputers** check colors
- **rm -f computers** computers as last configuration act as hardlink for newcomputers so you can't access newcoomputers (Symblic link) and red color mark it. must be have permission to create hard link
- **ln -s /etc/hosts computers** create soft link for hosts with new name computers

Why the result of linux commands "df" and "du" are different after deleting file?
**du** is used to estimate file space usage—space used under a particular directory or files on a file system.
**df** is used to display the amount of available disk space for file systems on which the invoking user has appropriate read access.
After deleting a file, the disk space occupied by this file will be slowly released.
The result of the command du doesn't include the size of the deleting file.
But the result of the command df includes the size of the deleting file due to its disk space is not released immediately.
So after deleting the file, the results of df and du are different until the disk space is released.

-------------------------------------------------------------------------------------------------------------------------

# RedHat Admin 1 Lab

Task 1: reset root password and protect the root password change.

Task 2: switch to root account and run the following tasks

1- Create **adam** and **sara** users and set both password to **333.**

2- Join **adam** to **sara** group as a secondary group without removing **adam's** currunt group membership.

3- Join **sara** to **sudoers**.

4- Set **sara** user expire date to 01-01-2023 and let her to change her password with next login.

5- set **sara** shell from **bash** to **sh.**

Task 3: continue as root

6- Create **dir1**, **dir2**, **dir3**, **dir4** and **dir5** under **adam's Desktop**.

7- Create under each of last directories 100 file named **file01** to **file100.**

8- make **dir1** and all including subdirectories and files written by the owner, group and only readable by the others.

9- Change the group owner for **dir1** and all including subdirectories and files to user named **adam.**

10- Copy **dir1** and all including subdirectories and files to the **/** and preserve the ownership and permissions as it.

Task 4: Switch to adam user and complete the following tasks

11- Copy the contents of **/etc/passwd** and save it in **adam's documents** with file name **users**.

12- Extract from **users** file all line includes **adam** and **ROOT** and save the result in **adam's home** directory in file named **adam file.**

13- Create hard link for **adam file** and name it **adam_link.**

14- Using **xz,** compress **dir2** and save the compressed data in **adam's Downloads** area as **dir2.xz** then extract the compressed data to **dir3** in **adam's Desktop.**

Task 5: switch to sara user and complete the following tasks:

15- Set the server time zone to **Muscat.**

16- Kill all process that running with **adam** user.

17- Start the following process in background with **nice value -10: sleep 100.**

18- Set numbering in **vim** for all users.

19- Clear bah history

20- Set all users history size to **5000.**

21- Set **cls** as alias for **clear** command for **adam** user only.

22- Check www.google.com is working or not using bash terminal.

23- Get and save all information about system **CPU**, **Memory**, **Hard Disk** to one file named **system info.**

24- Run the following commands together **cal**, **date**, **ll**, **pwd**, **asdf** and save both standard out and errors to **system info** file which you create in previous task with keeping the current **system info** file contents.

Task 6: switch to root user and complete the following tasks:

25- Using **yum**, only download all packages required for httpd and save it to **/packages**

26- Using RedHat iso file create **lrepo** as a local repo

27- Disable all repos and install **DHCP-Sever** using **lrepo**

28- Clean up all downloaded packages during installation.

-------------------------------------------------------------------------------------------------------------------------

Task 7: continue as root

29- Get the total number of all running services.
30- Change sshd port to 1234
31- Disable root user to access through sshd
32- Stop sshd service permanently.
33- Get the total number of installed packages.

Task 8: continue as root

34- set maximum size for all logs 50MB
35- configure your server to save all errors to /var/log/errors
36- configure your server to send all warning to another server named redlab

Task 9: continue as root

37- Disable the network card in server startup.
38- Configure the server to use 1.1.1.1 as a DNS server.
39- Print all firewall allowed ports to /tmp/ports.txt
40- Print the routing table to /tmp/rtable.txt

Task 10: continue as root

41- Find all **pdf** files included in **/usr/share/doc** and copy all to **/home/adam/Destop/dir2.**
42- Change permissions to 666 for all files included in /**home/adam/Destop/dir2**.
43- Find all files with permission **666** in adam's home and change the permission to **555**
44- Find and delete all files in **/home/adam/Destop/dir2** with size more than **100kb**