Red Hat System Administration I – Quick Guide

Version 24.05

Ahmed Abdelwahed
ahmed@abdelwahed.me
LinkedIn

Access the command line

Linux Command Syntax

- **Basic Structure** `<cmd> [+- option] [argument].`

Here's a breakdown of each part:

1. **<cmd>**: The command itself. This is the executable program or script you are running. For example, `ls`, `cd`, `echo`, etc.
2. **[+- option]**: Options or flags modify the behavior of the command. They can usually be prefixed with a single dash `-` (for short options) or a double dash `--` (for long options). For example, `-l`, `--help`, `-a`.
3. **[argument]**: These are the inputs or targets for the command. They can be files, directories, or other values the command operates on. For example, in the command `ls -l /home`, `/home` is an argument specifying the directory to list.

Keyboard Shortcuts

- **Interrupting and Navigating** `ctrl+c` to interrupt, `ctrl+d` to logout, `ctrl+u` to remove from the beginning, `ctrl+k` to remove to the end.
- **Cursor Movement** `ctrl+a` for the beginning of the command, `ctrl+e` for the end.
- **Word Deletion** `ctrl+w` to delete one word, `esc+d` to delete the next word.
- **Grouping and Accessing Consoles** `ls; cal; date; pwd` for grouping, `ctrl+alt+f2-f6` for tty2-tty6, `chvt 5` to move to terminal 5, `pkill -9 -t tty4` to send a SIGKILL signal.
- **Clearing and Searching** `ctrl+l` to clear, `ctrl+r` for reverse search.
- **Auto-completion** To enable, you can use `yum install bash-completion` and set `disable-completion off`.
- `cd ~` Redirects to the home directory.

System Utility Commands

- **Date Commands** Display and manipulate the date and time.
  - `date` Shows the current date and time.
  - `date +%d-%m-%Y-%H-%M-%S` Formats the date output.
- **Uptime and Load Monitoring**
  - `uptime` Shows the system running time and active users.
  - `w` Shows the uptime output and working users.
  - `watch -n 3 uptime` Refreshes the uptime every 3 seconds.
- **User and System Information**
  - `tty` Displays the current logged-in shell.
  - `bc` Provides a scientific calculator (binary calculator).
  - `whoami` Prints the username associated with the current effective user ID.
  - `lscpu; cat /proc/cpuinfo` Displays CPU information.
  - `lsmem; free -h; cat /proc/meminfo` Shows memory status information.
  - `hostname; hostname -f` Displays the full hostname.
  - `hostname -I` Shows the IPv4 address of the current machine. While `hostname -i` shows both IPv4 and v6.
  - `uname -a` Prints details about the machine and operating system.
- **Measures how long it takes to execute operation**
  - `time dd if=/dev/zero of=/root/bigfile bs=1G count=3`
  - `time ./my-script.sh`

NTP and Time Zone Commands

- **Date and Hardware Clock**
  - `hwclock` Displays the hardware clock.
  - `hwclock –hctosys` Sets the system clock from the hardware clock.
  - `hwclock --systohc` Sets the hardware clock from the current system time.
- **NTP Configuration and Status**
  - `vim /etc/chrony.conf` Edits the configuration file for Chrony, a replacement for NTPd by add your server to the list `server 192.168.100.100 iburst`
  - `systemctl restart chronyd` Restarts Chronyd, the default NTP daemon in Red Hat.
  - `chronyc sources` Lists the current sources being used by Chrony.
  - `ntpstat` Displays the status of the NTP daemon.
- **Time Zone Management**
  - `timedatectl list-timezones` Lists all available time zones.
  - `timedatectl set-timezone Asia/Muscat` Changes the system time zone.
  - `cal 6 1982` Shows the calendar for June 1982.

Aliases

Aliases are shortcuts that allow you to abbreviate a command or series of commands.

- **Creating and Using Aliases**
  - `alias cls="clear"` Creates an alias named 'cls' for the 'clear' command. To execute the alias, use $cls.
  - `alias docs='cd /home/user/Documents'` Creates a shortcut to navigate to a specific directory.
  - `alias -p` Prints all predefined aliases.
  - `unalias cls` Removes the 'cls' alias.
  - `unalias -a` Removes all aliases, including defaults.
- **Persistence of Aliases**
  - To make aliases permanent, you can save them in `.bashrc` inside the user's home directory.

History Size

- **Viewing and Manipulating History**
  - `cat ~/.bash_history` Displays the history saved file.
  - `history -c` Clears history.
  - `history -d 100` Removes line number 100 from history.
  - `history -w` Saves the current session's command history to the user's history file, typically `~/.bash_history.` Try `cat .bash_history | wc -l` to compare between `history` and and saved in `.bash_history`
- **History Size Configuration (~/.bashrc** or **~/.bash_profile)**
  - `echo $HISTSIZE` Displays the maximum number of commands to remember.
  - `echo $HISTFILESIZE` Displays the maximum number of lines contained in the history file.
  - `export HISTSIZE=2000` Sets a new size for history. (usually **~/.bash_history**),
  - `export HISTFILESIZE=2000` Sets a new file size for history.
  - `source ~/.bash_profile`
- **Utilizing History Shortcuts**
  - `!!` or `ctrl+p` Executes the last command.
  - `!g` Runs the last command that started with `'g'`.
  - `!$` retrieves the last argument from the last command
  - `!20` Executes the command previously executed with the history number 20.

Manage Files from the Command Line

- **Filesystem Structure (Linux File System Layout)**
  - `tree -L 2 / > system_structure.txt` Redirects the Linux hierarchy to a text file.
  - `man hier` Provides manual pages explaining the file system hierarchy.
  - `man file-hierarchy` Another command to view the hierarchy details.
- **File System Paths**
  - **Relative Path** Starts from the current directory.
  - **Absolute Path** Starts from the root directory (/).

Listing Files (Directories)

Linux treats everything as a file, and directories are no exception. Here's a breakdown of commands related to listing files

- **Basic Commands for Listing Files**
  - `pwd` Prints the working directory.
  - `ls` Lists files in the current directory.
  - `ls -a` Lists all files, including hidden ones.
  - `ls -lt /etc` Shows a long list, sorted by time.
  - `ls -ld /etc` Lists information about the specified directory.
- **Example of File Type Display**
  - `ls -aF /etc` Shows files and their types, with directories ending with a /

Creating Files and Directories

- **Creating Directories**
  - `mkdir -p par1/par2/dir` Creates a directory with parent directories as needed.
- **Creating Files**
  - `touch f1 f2 f3` Creates multiple files.
  - `touch "my file.txt"` Creates a file with spaces in its name.

File Display Commands

- **Viewing File Content**
  - `cat -n /etc/passwd` Displays file contents with line numbers.
  - `tac /etc/passwd` Reads files in reverse.
  - `less /etc/passwd` Used to read long files; press 'q' to quit.
- **Monitoring Files**
  - `tail -f /var/log/messages` Monitors a system log file in real-time.

File Maintenance Commands

- **Copying Files**
  - `cp -rvi file1 /media/file4` Copies and renames a file with interactive prompts.
  - `cp -f /etc/*.conf /home/data` Copies all .conf files, overwriting existing ones without prompting.
  - `dd if=/dev/sr0 of=/mnt/dvd.iso bs=1M` Copies DVD content to a specific location.
- **Moving and Renaming Files**
  - `mv file1 file2` Renames a file.
  - `mv file1 file2 file3 dir` Moves multiple files to a specific directory.
- **Removing Files and Directories**
  - `rmdir` or `rm -r` Removes a directory.
  - `shred passwd` Destroys file content.
  - `shred -u passwd` Destroys and removes a file.

GREP and EGREP (Global Regular Expression Print)

Search for Multiple Words

- `grep -e 'root' -e 'aabdelwahed' /etc/passwd`
- `egrep -i 'root|aabdelwahed' /etc/passwd`

Extended Regular Expression Syntax

- `grep -E -i -w -o 'user1|user2|user3' /etc/passwd`

Search Log Messages

- `egrep -i 'error|warning|critical' /var/log/message`

Search Specific Sequence of Words

- `egrep -i 'root\ ahmed' /etc/passwd`

Search Lines Beginning with a Specific Word

- `egrep -i '(^root)' /etc/passwd`

Search Lines Ending with a Specific Word

- `grep -i login$ /etc/passwd`

Display Lines Following a Match

- `cat /etc/passwd | grep -A 5 -i ahmed`

Recursive Search for a Word

- `grep -irl 'error' /home`

Pipes, xargs, awk, cut, sort, uniq, sed

Using xargs

- `ls | xargs rm` Deletes files listed in the directory by passing them as arguments to the rm command.

Using awk

- `ll | awk '{print $NF}'` Prints the name of each file in the directory.
- `ll | awk '{print $1, $NF}'` Displays a list of files along with their permissions.
- `awk -F: 'END {print NR}' /etc/passwd` Counts the number of lines.
- `awk -F: '{print $1}' /etc/passwd` Extracts the first field (usernames) using the ":" as a separator.
- `awk -F: '$3 > 1000 {print $1}' /etc/passwd` Extracts usernames where UID is greater than 1000.

Using cut

- `cut -d: -f1 /etc/passwd` Extracts the first field (usernames) using the ":" as a delimiter.

Using sort and uniq

- `sort file1` Sorts the contents of a file alphabetically.
- `sort -r file1` Sorts the contents of a file alphabetically in reverse order.
- `sort -k2 /etc/passwd` Sorts based on the second field (such as a user's login shell).
- `sort file1 |`uniq Sorts alphabetically and removes duplicates.
- `sort file1 | uniq -c` Sorts alphabetically, removes duplicates, and counts the number of duplicates.
- `sort file1 | uniq -d` Prints only the duplicated data in sorted order.

Using sed

- `sed -i 's/root/admin/g' passwd` Changes all occurrences of "root" to "admin" in the passwd file.
- `sed -i 's/admin//g'` passwd Removes all occurrences of "admin" from the passwd file.
- `sed -i '/^user1/d' /etc/sudoers` Deletes lines that start with user1.
- `sed -i '/^$/d' passwd` Deletes all empty lines from the passwd file.
- `sed -i 'G' passwd` Adds an empty line between all lines in the passwd file.
- `sed -i '1s/root/admin/g' pass` Replaces "root" with "admin" in the first line globally within the file named "pass."
- `sed -i '2,$s/root/admin/g' pass` Replaces "root" with "admin" in all lines starting from the second line through the end of the file named "pass."

Miscellaneous Commands

- `ss -tupln4 | grep LISTEN | awk -F: '{print $2}' | awk '{print $1}'` Prints allowed TCP/UDP ports from the list of active connections.
- `diff file1 file2` Compares the contents of two files and displays the differences.

Wildcards

Wildcards are special characters used to represent one or more characters in a file or directory name.

- `*` Matches any number of characters, including none.
- `?` Matches exactly one character.
- `[]` Matches any one of the characters inside the brackets.
- `{}` Allows you to specify a list of alternative patterns.
- `!` Negates the pattern, matching any file that does not match the pattern.
- `ls *.txt` Lists all files in the current directory with a .txt extension.
- `rm file?.txt` Deletes matching files.
- `cp [ab]*.txt destination` Copies matching files.
- `touch file0{1..9}` Creates 9 files.
- `rm file0*` Removes all files matching the pattern.
- `mkdir dir{1..10}, touch abc0{1..9}-xyz` Creates multiple files and directories.
- `rm *-xyz` Removes everything ending with "xyz."
- `touch dir{1..10}/file{1..100}` Creates 100 files in each directory.
- `touch file01, ll fi??01` Demonstrates the use of the '?' wildcard.
- `cat f[!dfghe]le1` Matches any characters except specified ones.
- `^old^new` Quick substitution in the command line, replacing "old" with "new" in the previous command.

Redirection

Redirecting standard output (stdout)

- `>` Redirects stdout to a file.
- `>>` Appends stdout to a file.
- `cat passwd > passwd_orig` Redirects input to output.
- `df -h > newfile` Overwrites file content.
- `cat /dev/null > ahmedfile` Deletes file content.
- `echo “wooooow” > passwd` Overwrites content.
- `echo “wooow” >> passwd` Appends content.
- `df -hlT > diskfree` Redirects disk free command.

Redirecting standard error (stderr)

- `2>` Redirects stderr to a file.
- `&>` Redirects both stdout and stderr to the same file.
- `(cal 2010; 111) >op.txt 2>err.txt Results` in two files for output and error.
- `(cal 2010; 111) 2> /dev/null` Shows output and hides error.
- `ls /etcw 2> err` Redirects standard error to an error file.

Redirecting standard input (stdin)

- `<` Redirects stdin from a file.
- `|` Uses a command's output as the input for another command.
- `ls /etc | grep ".conf"` Pipes one command into another.
- `cat < /etc/passwd` Redirects standard input from a file.

Get help in Red Hat Enterprise Linux

- **Short Description with whatis** Use the `whatis` command to get a brief description of a command. If it doesn't work, rebuild the manual database with `mandb`.
- **Example whatis** `ls` returns a brief description of `ls`.
- **Locating Files with whereis** Use `whereis ls` for binary, source, and manual pages of `ls`. `whereis -b cat` for binary files of `cat. whereis -m cat` for manual pages of `cat`.
- **Finding Path with which** Run `which ls; which rm` to find the path of `ls` and `rm`.
- **Portfolio using --help** Many commands support `--help`. **Example** `ls --help` provides a summary of `ls`.
- **Manual with man Command (9 Sections)** `man` displays manual pages with various navigation options
  - **`g`** **or p** Start.
  - **`shift+g`** `End`.
  - **`q`** Quit.
  - **`/`** Search forward.
  - **?** Search backward.
  - **n** Next search.
  - **N** Previous search.
  - **`100g`** Go to line 100.
  - **`Space`** Next page.
  - **`man -K "copy files"`** Global search.
  - **`man 5 crontab`** Jump to section 5.
  - `man useradd | grep -i -A 20 ^files` Searches (case-insensitively) for lines starting with "FILES" and displays that line plus the next 20 lines.
  - `man find | grep -i -A 20 ^example`
- **Using `info` for More Information than man** info provides more details. Use space, backspace, and u, s for search.
- **Example** `info vim` for information about `vim`.
- **Accessing System Documentation** `/usr/share/doc` holds system and package documentation.

Create, view, and edit text files

Linux File Editor (Vim)

Command Mode

Moving and Navigating

- **Go to First Line** `1G` or `gg`.
- **Go to Last Line** `G.`
- **Go to Specific Line** `2G` or `20gg` for second or 20th line.
- **End and Start of Line** `^` and `$.`

Inserting and Appending

- **Before Cursor** `i`.
- **New Line Below** `o.`
- **New Line Above** `O.`
- **Start of Line** `I`.
- **After Cursor** `a.`
- **End of Line** `A.`

Copying and Pasting

- **Copy Letter/Word/Line** `yl, yw, yy.`
- **Copy Multiple Lines** `20yy.`
- **Paste Below/Above** `p, P.`

Deleting

- **Delete Letter/Word/Line** `dl, dw, dd.`
- **Delete Multiple Lines** `20dd.`
- **Delete to End of Line/File** `d$, dG.`

Changing Text

- **Delete and Change** `cl, cw, cc.`
- **Change Case** `Shift+~, g~~, gUU.`
- **Merge Lines** `shift J.`

Other Commands

- **Undo** `u, shift u.`
- **Save and Exit** `shift zz.`
- **Exit Without Save** `shift zq.`

Execute Mode

Saving and Exiting

- **Save and Exit** `x, wq.`
- **Force Exit** `q!.`

Numbering and Language Settings

- **Numbering Lines** `set nu, set nonu.`
- **Arabic Writing** `set arabic, set noarabic.`

Search and Replace

- **Highlight Search** `se hlsearch, se nohlsearch.`
- **Substitute Words** `%s/install/config/g.`
- **Delete Lines** `$d, 1,9d, %d.`
- **Add Empty Lines** `%s/$/\r/g.`

Visual Mode

Commenting Blocks

- Use `Ctrl+V, Shift+I, #,` and `Esc` for commenting multiple lines.

Set Vim Defaults with .vimrc (Custom Vim)

Defaults

- `set number, set ignorecase, set hlsearch.`

Vim Tips

Screen Management and Locking

- **Lock and Unlock** `ctrl+s, ctrl+q.`
- **Add New Screens** `ctrl+w+n, ctrl+w+v.`
- **Move Between Screens** `ctrl+w.`
- **Global Settings** Edit `/etc/vimrc.`
- **Open Specific File** `cat /etc/passwd | vim.`

Other Editors

Editing Tools

- **Nano** `nano nf2.`
- **Gedit** `gedit file.`

Manage local users and groups

ID Command for Managing Users and Groups in Linux

- **Showing User Identification** Utilizing the `id` command displays key information about Linux users. For instance, `id test1` would yield the user ID, group ID, and any associated groups.
    - **UID** The User ID, with the root ID being 0 and normal user IDs starting from 1000. Those under 1000 are special and system-related.
    - **GID** Primary (private) Group ID, which usually mirrors the UID. Private groups help maintain data privacy.
    - **Groups** These are secondary groups associated with file access permissions.
    - **Identifying a Specific User** `id test1 uid=1003(test1) gid=1003(test1) groups=1003(test1)` shows user identification for 'test1'.

Creating User Accounts in Linux

- **With Default Settings** `useradd user1` will create 'user1' with all default settings.
- **With Custom Options** `useradd -c "test account" -u 5002 -M -N -g user1 -G sales,hr -s /bin/sh ahmed`
    - `-c "test account"` This sets the comment (or full name) for the user to "test account".
    - `-u 5002` This sets the user ID (UID) for the new account to 5002.
    - `-M` This instructs the command not to create a home directory for the user.
    - `-N` This means do not create a group with the same name as the username. Without this, useradd would create a group named "ahmed" by default when you add the "ahmed" user.
    - `-g user1` This sets the primary group of the new user to "user1".
    - `-G sales,hr` This adds "ahmed" to the supplementary (or additional) groups "sales" and "hr".
    - `-s /bin/sh` This sets the default shell for the user to /bin/sh.
    - `ahmed` This is the name of the user being added.

Account Defaults

- **Configuring User Defaults** edit `/etc/login.defs` for password options, user and group IDs, home directory creation, and default umask.
- **Viewing and Editing User Defaults** Utilize `useradd -D` or edit `/etc/default/useradd` to view or change default settings.
- **Understanding SKEL Variable** It stands for "skeleton directory" and facilitates providing default files and directories to new users.

Creating Bulk Users

- **Create a Text File** 'users.txt' containing details of users you wish to create.
- **Syntax** The file must follow the syntax `LoginName:Password:UID:GID:Comment:home_dir:Shell:Name.`
- **Applying the File** Utilize the `newusers` command with the file as an argument.
    - **Creating Multiple Users** Including the entries in 'users.txt' like `user001:user001:1002:1002:user01/home/user01:/bin/bash` and executing `newusers users.txt` will create the specified users.
    - **Creating a Password** `passwd user1` to set the password for 'user1'.
    - **Checking Status** `passwd -S ahmed` to view 'ahmed's password settings.

Implementing Password Policies

- **Configuring Policies** Files like `'/etc/security/pwquality.conf'` and `'/etc/login.defs'` store configuration for password quality and rules.
- **Checking Quality** `pwscore` checks password quality, with values ranging from `0` to `100`.

Managing User and Password Age

- **Viewing and Changing Password Age** `chage` commands provide detailed control over password aging policies, allowing you to set specific conditions and requirements.
- **Handling Shadow Data** Commands like `'pwunconv'` and `'pwconv'` control how password information is stored, either in `'/etc/passwd'` or `'/etc/shadow'`.
- **Account Locking and Expiration** Various `chage`, `usermod`, and `passwd` commands manage account locking, expiration, and access control.

```
chage -d 0 -M 42 -m 2 -W 4 -E 2024-12-31 user1
```

  - `-d 0` Sets the last password change date to the epoch (January 1, 1970). Setting it to 0 will typically force the user to change their password the next time they log in.
  - `-M 42` Sets the maximum number of days the password is valid (before it expires) to 42 days.
  - `-m 2` Sets the minimum number of days before which the password cannot be changed (once changed) to 2 days.
  - `-W 4` Sets the number of days to warn the user before the password expires. In this case, the user will receive a warning 4 days before their password is set to expire.
  - `-E 2024-12-31` Sets the account expiration date to December 31, 2024. After this date, the user will not be able to log in.
  - `user1` the target user account for which these changes are being made.

  - **Setting Last Password Change** `chage -d 2024-6-13 ahmed`.
  - **Locking and Unlocking Account** `usermod -L ahmed;usermod -U ahmed`
  - **Blocking Shell (non-interactive shell)** `chsh -s /sbin/nologin ahmed`.
  - **Deleting User** `userdel -r ahmed` to remove user and their home directory.

Managing Local Groups

- **Linux Identity System** Linux divides identity into owner, individual user, groups of users, and the world.
- **Viewing Group Membership** Commands like `groups`, `groups username`, and `getent group` show group membership.

Creating Local Groups

- **Creating and Viewing Groups** Utilize `groupadd` to create groups and `cat /etc/group`, `getent group`, or `grep` to view groups.
- **Adding New Group** `groupadd sales` to create a 'sales' group.
- **Creating Group with Specific ID** `groupadd -g 555 admins`.

Group Membership Management

- **Primary Group Changes**
  - `newgrp wheel` Switch the current session's primary group to `wheel`.
- **Secondary Group Modifications**
  - `usermod -G wheel aabdelwahed` Set `wheel` as the sole secondary group for `aabdelwahed`, removing all others.
  - `usermod -aG wheel aabdelwahed` Add `wheel` as a secondary group for `aabdelwahed` without removing current secondary groups.
  - `usermod -aG wheel,admins ahmed` Add `wheel` and `admins` as secondary groups for `ahmed`.

- **Group Membership Administration with gpasswd**
  - `gpasswd -a user01 wheel` Add `user01` to the `wheel` group.
  - `gpasswd -M user01,user02,user03 sales` Explicitly set members of `sales` group, overwriting existing ones.
  - `gpasswd -d user01 sales` Remove `user01` from `sales` group.
  - `gpasswd -A user01 sales` Set `user01` as an administrator for `sales` group.
- **Querying Group Memberships**
  - `groupmems -lg wheel` List members of the `wheel` group.
  - `lid ahmed` List groups associated with `ahmed`.
  - `lid -g wheel` Display members of `wheel` group.

Group Management

- **Modifications**
  - `groupmod -n finance sales` Rename `sales` group to `finance`.
  - `groupmod -g 1100 hr` Set GID of `hr` group to 1100.
- **Deletions**
  - `groupdel finance` Remove the `finance` group.

Note `usermod` requires root privileges, while `newgrp` can be used by regular users to change their session's primary group.

Group Passwords

- **Setting Group Password** Use `gpasswd` to assign a group password, allowing users to change group membership with `newgrp`.

User and Group Configuration Files

- **Viewing Configuration** Utilize `cat` on files like `/etc/passwd`, `/etc/shadow, /etc/group`, `/etc/gshadow` to view user and group properties.
- **Editing Configuration** Edit files like `/etc/logon.defs` and `/etc/default/useradd` for generic configurations.

Visudo (Enable Sudo)

- **Enabling Sudo** Utilize `visudo` to specify who can run which command without needing the root password.
- **Enable Specific Access** Enable `%wheel ALL=(root) ALL`, and add specific users.
- **Enabling sudo Access** Edit specific lines or add users to the 'wheel' group. `usermod -G wheel aabdelwahed`.
- **Disabling 5-Minute Timeout** Set `timestamp_timeout=0`.
- **Restart SSH Service** Use `systemctl restart sshd` to apply changes.

Control access to files with Linux file system permissions

Understanding Files and Directory Permissions

- **Command to View Permissions** `ls -l` is used to display file and directory permissions.
- **Permissions Structure** Permissions are depicted in 9 characters, representing permissions for the owner, group, and others.
- **Order of Checking Permissions** The system checks permissions in the order of user, group, and others, stopping once a match is found.
- **File Access Rights** Defined by Read (R), Write (W), and Execute (X).

Examples of Right Access and Commands

- **Read (R)**
  - **Files** View contents with `cat`, `less`, `more`, `tac`.
  - **Directories** List contents with `ls`.
- **Write (W)**
  - **Files** Modify contents with `echo`, `cat`, `vim`.
  - **Directories** Create or remove with `mkdir`, `rm`.
- **Execute (X)**
  - **Files** Allows execution if it's a script or program.
  - **Directories** Change to the directory with `cd`.

Alphabet vs Numerical Syntax for Permissions

Numerical Representation

- `0` for No permissions; `1` for Execute only; `2` for Write only (used with command redirection), `3` for Write and Execute; `4` for Read only; `5` for Read and Execute, `6` for Read and Write, defining various access levels for users on.

Listing Permissions

- **Viewing Specific File Permissions** `ls -l file01`.
- **Viewing Full Metadata** `stat file01`.
- **Displaying Symbolic Permissions** `stat -c %A file01`.
- **Displaying Numeric Permissions** `stat -c %a file01`.
- **Showing Numeric Permissions for Specific Patterns** `stat -c %a test_*.`

Modifying Permissions with chmod

- **Changing Permissions** The `chmod` command is used to change file and directory permissions.
- **Symbolic Mode** Use characters like `u` for owner, `g` for group, `o` for others, and `a` for all, combined with `+`, `-,` or `=` to add, remove, or set specific permissions.
  - `chmod uo+x, g-w file01` to add execute permission to the owner and others and write only for group.
  - `chmod u=r,g=rw,o=rwx file01` sets specific permissions for user, group, and others.
- **Numeric Mode** Use numerical values to define permissions.
  - `chmod 755 file01` to set read, write, and execute for owner, and read and execute for group and others.

Managing Default Permissions (Umask)

- **Default Values** Files (0666), Directories (0777).
- **Use umask Command** Adjust the default permissions.
  - **Examples** `umask 0` to set 0000, `umask 27` for specific settings.
- **Persistent Change** Edit `/etc/bashrc` and write, e.g., `umask=555`.

Managing File Ownership

- **Use chown Command** Change user and group ownership.
    - **Change User Owner** `chown user1 file01` (only root can do this).
    - **Change Group Owner** `chgrp sales file01` or `chown :Sales file01`.
    - **Change Both** `chown user1:Sales file01`.
    - **Recursive Change** `chown -R user1:Sales ./` for current directory and subdirectories.
- **View Numeric IDs** `ls -ldn dir1` to view numeric user and group IDs.

Audit Permission Changes

- **Install Audit** `yum install audit`.
- **Enable Audit Service** `systemctl enable auditd.service; systemctl start auditd.service.`
- **Set Audit Rule** `auditctl -w /day2 -p a -k day2_permission_change` to monitor specific changes.
- **Search Audit Logs** `ausearch -k day2_permission_change` to search for specific tagged events.
- **Make the Rule Persistent** `echo "-w /day2 -p a -k day2_permission_change" | tee -a /etc/audit/rules.d/audit.rules`

Making Links Between Files (Soft and Hard Links)

1. **Hard Links**
    - **Definition** Hard links are essentially multiple directory entries for a single file on the file system.
    - **Creation** `ln source_file hard_link_name`
    - **Characteristics**
        - Shares the same inode as the original file.
        - Acts as a regular file and doesn't indicate if it's a link.
        - Modifications are reflected across all linked files because they point to the same data blocks.
        - If the original file is deleted, the hard link will still access the data.
        - Can't be created for directories to prevent cyclic references and loops.
2. **Symbolic (Soft) Links**
    - **Definition** Symbolic links are special files that point to another file or directory path.
    - **Creation** `ln -s source_file_or_directory symlink_name`
    - **Characteristics**
        - Contains a path to the target file, rather than pointing to the data blocks directly.
        - Has a different inode number.
        - Acts as a pointer or shortcut.
        - If the original file or directory is deleted or moved, the symlink becomes a "dangling link" (pointing to a non-existent file).
        - Can be easily identified with the `ls -l` command because they show the path to the original file.

Using find Command

1. **Search by Name Pattern**
   - **Example** `find / -name file*`
   - **Explanation** Searches for all files starting with "file" in the root directory.
2. **Search by Size**
   - **Example** `find / -size +1G`
   - **Explanation** Searches for all files larger than 1 GB in the root directory.
3. **Search by Permissions**
   - **Example** `find -type f -perm 644`
   - **Explanation** Searches for all files with permission 644.
4. **Search by Type (e.g., Empty Directories)**
   - **Example** `find /tmp -type d -empty`
   - **Explanation** Searches for all empty directories in the /tmp directory.
5. **Search by Owner or Group**
   - **Example** `find /tmp/ -user root`
   - **Explanation** Searches for all files owned by the "root" user in /tmp.
6. **Copy Files Based on Search**
   - **Example** `find /usr/share/doc -name *.html -exec cp {} . \;`
   - **Explanation** Finds all html files in /usr/share/doc and copies them to the current directory.
7. **Change Permissions of Specific Files**
   - **Example** `find /root -type f -perm 0777 -exec chmod 500 {} \;`
   - **Explanation** Searches for all regular files under "/root" with permission 0777 and changes their permissions to 500.
8. **Find Files Without Specific Permissions**
   - **Example** `find / -type f ! -perm 777`
   - **Explanation** Finds files without 777 permissions in the root directory.
9. **Find and Delete Specific Files**
   - **Example** `find / -type f -name *.mp3 -size +10M -exec rm {} \;` or `find / -type f -name *.mp3 -size +10M -delete`
   - **Explanation** Finds and deletes all MP3 files larger than 10 MB in the root directory.
10. **Identifying Recently Accessed and Modified Files**

- `find / -mtime 1` Searches the entire filesystem for files modified exactly 24 hours (1 day) ago.
- `find / -mtime -1` Finds files modified within the last 24 hours from the entire filesystem.
- `find / -atime 5` Searches for files last accessed exactly 5 days ago throughout the entire filesystem.
- `find / -atime -5 2>/dev/null` Finds files accessed within the last 5 days, suppressing error messages.
- `find / -amin 10 2>/dev/null` Searches for files last accessed exactly 10 minutes ago, hiding errors.
- `find / -mmin -10` Finds files modified within the last 10 minutes across the filesystem.
- `find / -mmin +10` Searches the entire filesystem for files that were last modified more than 10 minutes ago.

Using locate Command

1. **Basic Search**
   - **Example** `locate filename`
   - **Explanation** Searches for files with "filename" in their names using the previously built database.
2. **Update Database**
   - **Example** `updatedb`
   - **Explanation** Updates the database used by `locate` to ensure that the search results are current.

Monitor and manage Linux Processes

Process Inspection Using ps Command

- **HTTP Process Inspection**
  - `ps aux | grep http` Searches for processes related to HTTP from the list of all processes.
- **General Process Inspection**
  - `ps aux` Provides detailed information about most processes currently running on the system. It displays owner, CPU usage, memory usage, and the command itself.
  - `ps -elf` Displays a long-format listing of all processes, including their parent process IDs (PPID).
  - `ps -eo pid,ppid,uid,cputime,pmem,cmd` Offers a customized output for the `ps` command, showing process ID, parent process ID, user ID, CPU time, percentage of memory used, and the command itself.
  - `ps -fU ahmed` Displays full-format listing of all processes specifically owned by the user "ahmed".
- **Tree View & Paging**
  - `ps fax | less` Displays a hierarchical view (similar to a tree structure) of processes and their child processes. The `less` command is used for paging through the list.
- **Quick Analysis**
  - `ps aux | head` Outputs the first ten lines from the `ps aux` command. This usually includes the column headers and the first nine processes.
  - `ps aux | wc` Counts the number of lines, words, and characters in the output of `ps aux`. This essentially gives an idea of the number of processes running.

Backgrounding Tasks

- `sleep 1000` Pauses the shell or script's execution for 1000 seconds.
- `sleep 1000&` Starts the `sleep 1000` command in the background.
- `jobs` Displays the status of jobs in the current session.
- `bg` Resumes the last job that was stopped and runs it in the background.
- `fg` Brings the most recent background job to the foreground.
- `fg 1` Brings the job with job number 1 to the foreground.
- `ps -p 13732` or `ps -F 13732` Displays information about the process with the process ID (PID) 13732.

Stress Testing

- `dnf install -y https://dl.fedoraproject.org/pub/epel/epel-release-latest-9.noarch.rpm` Installs the Extra Packages for Enterprise Linux (EPEL) repository.
- `yum install stress` Installs the `stress` tool.
- `stress --cpu 2 --timeout 600` Applies load on 2 CPU cores for 10 minutes.
- `stress --vm 2 --vm-bytes 256M --timeout 60` Consumes 256MB of RAM for 60 seconds using 2 workers.
- `stress --hdd 2 --timeout 60` Stresses the hard disk for 60 seconds with 2 workers.

Query Processes

  - `pgrep sleep` Searches for processes named "sleep" and prints their process IDs.
  - `ps p $(pgrep sleep)` Uses the process IDs found by `pgrep sleep` to display detailed information about each "sleep" process.
  - `pgrep --count sleep` Returns the number of "sleep" processes currently running.

Quick Guide I RH124

List all internet and network files in use by all processes: `lsof -i`
List processes listening on specific ports: `lsof -i :22`

Terminate Processes

- Soft Kill `kill 8537, kill -15 8537, kill -term 8537, kill -sigterm 8537`.
- Hard Kill `kill -9 8537, kill -kill 8537, kill -sigkill 8537`.
- `pkill sleep` Sends the `TERM` signal to all "sleep" processes, asking them to terminate gracefully.
- `pkill -KILL -u user` Sends the `KILL` signal to all processes owned by the user named "user", forcefully terminating them.
- `killall sshd` Sends the `TERM` signal to all processes named "sshd", asking them to terminate gracefully.

Monitoring using top and htop

- `top` A monitoring tool that monitors active processes in real time, sorting them based on processor utilization. Can be customized with various keys
- `f` Add fields like PPID; z Color results; h More help.
- `k` Kill process; i Show only active processes; r Renice; q Quit.
- `1` Displays detailed information for each CPU.
- `dnf install htop` Installs htop (epel must be installed first).

Setting Process Priority with nice

- **Commands**
  - `nice -n 5 dd if=/dev/zero of=/dev/null &` This command starts a `dd` process with a `nice` value of 5, meaning it has a lower priority than the default processes.
  - `nice -n -5 dd if=/dev/zero of=/dev/null &` This starts the same `dd` process, but this time with a `nice` value of -5, giving it a higher priority.
  - `renice -n 10 -p 14721` This command is used to change the `nice` value of a running process. In this case, it changes the `nice` value of the process with the PID (Process ID) 14721 to 10.
- **Important Notes**
  - Only the root user can set negative nice values since they give processes a higher priority. Such changes can be resource-intensive and potentially disruptive, hence the need for root privileges.
  - When multiple dd commands are run with different nice values, the one with the higher priority (lower nice value) is likely to be given more CPU time.

Manipulating CPU Cores

- The command `echo 0 > /sys/bus/cpu/devices/cpu1/online` disables a CPU core (in this case, cpu1), effectively turning it off. This can be helpful when trying to observe the CPU resource consumption of processes with different nice values on a single core.

Setting Default nice Values

- As the root user, you can set default `nice` values for specific users or groups. This is achieved by adding appropriate entries in the `/etc/security/limits.conf` file. For example
- `user hard priority 7` This sets the default `nice` value for the user named "user" to `7`.

Using ulimit

- `ulimit` is a shell builtin that can set user limits. The example command `ulimit -u 19` sets the maximum number of processes a user can run simultaneously to 19. However, this change is temporary and applies to the current shell session.

Monitor and Terminate Other Users' and Sessions

- `chvt 3` Changes the virtual terminal to 3, allowing login with another user.
- `loginctl list-sessions` Lists all sessions.
- `loginctl terminate-session 4` Shuts down session 4.
- `loginctl list-users` Lists all users.
- `loginctl terminate-user ahmed` Terminates all sessions for the user "ahmed."
- `loginctl user-status 1000` Checks the status of user with ID 1000.

Control services and daemons

View and Query Services

- `systemctl -t help` Shows the available units.
- `systemctl list-units` Shows all loaded units.
- `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 processes.
- `systemctl is-active sshd` Check if sshd is active.
- `systemctl is-enabled sshd` Check if sshd is enabled.
- `systemctl --failed --type=service` List failed services.

Start, Stop, Reload Services

- `systemctl stop sshd.service` Stop sshd service.
- `systemctl start sshd.service` Start sshd service.
- `systemctl restart sshd.service` Restart sshd service.
- `systemctl reload sshd.service` Reload sshd service.
- `systemctl reload-or-restart sshd.service` Reload if available or restart sshd service.

View Dependencies

- `systemctl list-dependencies sshd.service` Display dependencies hierarchy.

Masking Services

- `systemctl mask name.service` Mask service to prevent it from being started.
- `systemctl unmask name.service` Unmask a service.
- `systemctl list-unit-files --state=masked` Get all masked services.

Edit and Customize Services

- `/usr/lib/systemd/system/` Directory containing systemd service files.
- `vim /usr/lib/systemd/system/httpd.service` Edit the httpd service file.
- `systemctl cat httpd` Shows the configuration file for the httpd service.
- `systemctl show httpd` Display properties of httpd.
- `systemctl edit httpd` Edit the httpd service in the default editor.
- `systemctl daemon-reload` Reload system manager configuration.
- `systemctl restart httpd` Restart the httpd service.
- `ps aux | grep http` Find and kill the httpd process by PID, used in conjunction with killing the process to observe its state.

Change Default Editor (optional)

- In `~/.bash_profile` for the current user or `/etc/environment` for all users, add the `line export EDITOR=/usr/bin/vim` to set vim as the default editor.

Configure and secure SSH

1. Identify the Package Providing sshd

- `dnf whatprovides */sshd`

2. Install OpenSSH

- `dnf install openssh`

3. Edit SSH Configuration to use port 1414

- `vi /etc/ssh/sshd_config` Modify it to work through port 1414.

4. Update Firewall Rules

- `firewall-cmd --zone=public --add-port=1414/tcp --permanent` Allow port 1414.
- `firewall-cmd --reload` Reload firewall rules.

5. Configure SELinux for Port 1414 (if enabled)

- `semanage port -a -t ssh_port_t -p tcp 1414` Allow 1414 port from SELinux.

6. Enable and Start SSHD

- `systemctl enable sshd; systemctl start sshd`

7. Verify SELinux Configuration

- `semanage port -l | grep sshd`

Secure SSH

1. Backup Current Configuration

- `cp /etc/ssh/sshd_config /etc/ssh/sshd_config.bak`

2. Edit SSH Configuration for Security

- `vi /etc/ssh/sshd_config`
- Set `LoginGraceTime 30` Restrict time for authentication.
- Set `PermitRootLogin no` Disable root login.
- Set `Port 1414` Change default port.

Configure SSH Keys

1. Generate and secure SSH Key Pair on Remote Server

- `ssh-keygen`
- `chmod 700 ~/.ssh`
- `chmod 600 ~/.ssh/authorized_keys`

2. View Keys

- `cat /home/ahmed/.ssh/id_rsa` Contains private key.
- `cat /home/ahmed/.ssh/id_rsa.pub` Contains public key.

3. Copy Public Key to Target Host

- `ssh-copy-id -i ~/.ssh/id_rsa.pub user@host`

4. Restart SSHD

- `sudo systemctl restart sshd`
- `ssh user@host` Test the connection.

Disable Root Login and Password-Based Login

1. Edit SSH Configuration

- `vi /etc/ssh/sshd_config` and change the following
  - `ChallengeResponseAuthentication no`
  - `PasswordAuthentication no`
  - `UsePAM no`

2. Reload SSHD

- `systemctl reload sshd`

Linux System Logs Monitor Guide

Overview Log files on Linux servers provide a detailed record of system activities. These logs aid in troubleshooting, monitoring, and security evaluations. To get the most out of these logs, they need to be managed and analyzed effectively.

Common Linux Log File Locations and Descriptions

System Logs

1. `/var/log/messages` Contains general system messages, including startup messages. It's one of the primary logs administrators check when troubleshooting.
2. `/var/log/boot.log` Logs related to the system booting process.
3. `/var/log/kern.log` Logs from the Linux kernel. Useful for troubleshooting hardware and kernel-specific issues.
4. `/var/log/secure` (or `/var/log/auth.log`) Authentication-related logs, recording user authentications, attempted logins, and other security-related events.
5. `/var/log/utmp` or `/var/log/wtmp` These are not really log files. They store information about who is currently logged in and login history. The who, w, last and lastb commands use these files.
6. `/var/log/cron.log` Logs from the cron daemon, showing the execution of scheduled tasks.

Application and Service Logs

7. `/var/log/maillog` Logs from mail servers like Sendmail or Postfix. Useful for troubleshooting email delivery issues.
8. `/var/log/qmail/` Directory containing logs specifically for the qmail mail server.
9. `/var/log/httpd/` Contains log files for the Apache HTTP server, including access.log (recording all requests processed by the server) and error.log (recording errors).
10. `/var/log/lighttpd/` Directory with logs for the Lighttpd web server, structured similarly to Apache's logs.
11. `/var/log/mysqld.log` Log file for the MySQL database server. Contains database server-related messages, including errors, warnings, and other diagnostic info.
12. `/var/log/yum.log` Log for the yum package manager, recording package installations, updates, and removals.

Understanding rsyslogd Configurations

`rsyslogd` provides flexible logging configurations that can be tailored based on

- **Facilities** Categories of information, e.g., system, security, mail.
- **Priorities** Severity levels such as emergency, error, warning, etc.
- **Destinations** Where the logs will be written to.

Example Configuration Edit `/etc/rsyslog.conf` to

#log all warning messages to warning file

```
*.warn   /var/log/warnings
*.err    /var/log/errors
systemctl restart rsyslog
```

Managing Logs with logrotate

logrotate is an essential tool on Linux systems for rotating, compressing, and managing log files.

Sample Configuration for Log Files

1. **Create Two Test Log Files:**
```
mkdir /tmp/log
vim /tmp/log/test.log   # Add data to this log file
vim /tmp/log/test1.log  # Add data to this log file
```
2. **Logrotate Configuration:** Edit the logrotate configuration file for your test logs:
```
vim /etc/logrotate.d/test
```
   Sample configuration:
```
/tmp/log/*.log {
    size 100M
    rotate 4
    weekly
    compress
}
```
   - **size 100M**: Rotate logs when they reach 100MB.
   - **rotate 4**: Keep four rotated logs.
   - **weekly**: Rotate logs weekly.
   - **compress**: Compress the rotated logs.
3. **Manually Running Logrotate:** To manually trigger log rotation:
```
logrotate -f /etc/logrotate.conf
```

SOS Report

The SOS report is crucial for system administrators and support professionals. Here's more about its usage

1. **Generating the Report**
   - The command to generate the report `sosreport`
   - The utility might require superuser privileges, so you might need to use sudo.
2. **What's Included**
   - Configuration details System and application configurations.
   - Hardware information List of devices, memory usage, CPU details, etc.
   - System logs Logs from /var/log, journal logs, etc.
   - Status of system services and processes.
3. **Usage**
   - After generating the report, you'll get a compressed file, typically in .tar.xz format.
   - This file can be shared with support or analyzed locally.
   - It's wise to understand the content of the report, especially if sharing externally, to avoid unintentionally sharing sensitive information.

Managing Red Hat Enterprise Linux Networking

Basic IP Commands

- **Display Interface Details** `ip a show eth0` or `ip a s enos3; ip -6 a`
- **Turn Interface Off** `ip link set eth0 down` Turns the `eth0` interface off.
- **Add Temporary IP Address** `ip addr add 192.168.1.100/24 dev eth0` Assign a temporary IP address to `eth0`.
- **Show Routing Table** `ip r show`
- **Add Temporary Default Gateway** `ip route add 172.16.1.0/24 via 192.168.1.1`
- **ARP Table / Neighbors** `ip n show`
- **Display Specific Interface Information** `ip -4 a s em1` Displays information about the `em1` interface for IPv4.
- **Routing Table for IPv4/IPv6** `ip r s` and `ip -6 r`
- **Trace Network Paths** `tracepath www.google.com` and `tracepath6 www.google.com`

Network Diagnostics

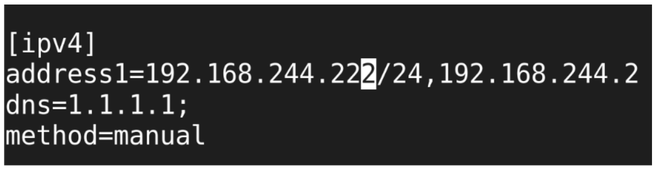
- **Packet Capture and Analysis** `tcpdump` .
- **Network Statistics - Netstat** `netstat -a | more` Listens to all TCP and UDP ports.
  `netstat -tupln, ss -tunap ,netstat -s, netstat -r`
  - `dnf install bind-utils`
  - `dig yahoo.com; dig mx yahoo.com; dig ns yahoo.com; dig txt yahoo.com`
  - `host -t MX p yahoo.com`
  - `dig @1.1.1.1 yahoo.com`
  - `tcpdump -i eth0 port 22`
  - `tracepath www.google.com`
  - `ping -I eth0 www.yahoo.com`
  - `nmap -p 80 192.168.1.1 ; nmap -p 22,80,443 192.168.1.1; nmap 192.68.244.1-100`

Configuration Files

1. **Global Network Configuration** `/etc/sysconfig/network`

- **The network configuration files used to be in** `/etc/sysconfig/network-scripts/ifcfg-ethx` **but have moved to** `/etc/NetworkManager/system-connections/.`

2. **Hostname** `/etc/hostname`
3. **Name Server Configuration** `/etc/resolve.conf`
4. **Static Table Lookup for Hostnames** `/etc/hosts`



```
[ipv4]
address1=192.168.244.222/24,192.168.244.2
dns=1.1.1.1;
method=manual
```

Persistently Storing Network Configurations

Edit the interface configuration using a text editor like `vim`

- `vim /etc/NetworkManager/system-connections/`

After making changes, restart the Network Manager `systemctl restart NetworkManager`

curl and ping Commands

- **Download Files** `wget [fileurl]`
- **Check Website Accessibility** `curl www.google.com` If it returns the website's content, the site is accessible.
- **Download Using** `curl curl -O www.site/filename` This can replace `wget` if it's not supported in your system.
- **Ping a Website** `ping www.google.com` For a limited number of pings, use
  `ping -c 5 www.google.com; Ping ping6 www.google.com`

Network Management with nmcli and nmtui

- **Show Devices** `nmcli d s` or `ip link`
- **Show Connections** `nmcli connection show`
- **Show Active Connections** `nmcli connection show --active`
- **Device Status** `nmcli device status`
- **Add a Connection to NIC (Static)**
  ```
  nmcli connection add con-name newcon1 ifname ens224 ipv4.addresses 192.168.219.140/24 ipv4.gateway 192.168.219.2 ipv4.dns 1.1.1.1 ipv4.method manual type ethernet
  ```
- **Modify an Existing Connection**
  ```
  nmcli connection modify "ens160" ipv4.addresses "192.168.219.111/24" ipv4.gateway "192.168.219.2" ipv4.dns "1.1.1.1" ipv4.dns-search "search.abdelwahed.me" ipv4.method manual autoconnect yes
  ```
- **Add extra connection to the same NIC (DHCP)**
  ```
  nmcli con add type ethernet con-name newcon2 ifname ens224 ipv4.method auto
  ```
- **Reload Connection** `nmcli connection reload`
- **Activate a Connection** `nmcli connection up "newcon1"`
- **Deactivate a Connection** `nmcli connection down newcon1`
- **Disable NIC** `nmcli device disconnect eth0`
- **Enable NIC** `nmcli device connect eth0`
- **Graphical Network Configuration Tool** `nmtui`

Archiving and Transferring Files Using `tar` and Compression Tools

Using tar for Archiving and Extraction

- **Archiving**
  - To archive `/home` directory into a file named home.tar `tar cvf home.tar /home/`
  - For the `/etc` directory into `etc.tar` `tar cvf etc.tar /etc/`
- **Checking Archive Type** `file home.tar`
- **Listing Archive Contents**
  - For `home.tar` `tar tf home.tar`
  - To find specific files (e.g., rsyslog in etc.tar) `tar -tf etc.tar | grep rsyslog`
- **Extraction**
  - To extract in the current location `tar xvf home.tar`
  - To extract to a different location like `/mnt/` `tar xvf home.tar -C /mnt/`
  - For extracting with original permissions `tar xpvf home.tar -C /mnt/`
- **Compression with tar**
  - Using **gzip** `tar zcvf /lab01/home.gz /home/`
    - Extraction `tar zxvf home.gz -C /save`
  - Using **bzip2** `tar jcvf /lab01/home.tar.bz2 /home/`
    - Extraction `tar jxvf home.bz2 -C /save`
  - Using **xz** `tar Jcvf /lab01/home.xz /home/`
    - Extraction `tar Jxvf home.xz -C /save`
- **Backup and Restore**
  - For backup `tar zcvf home.tar.gz /home`
  - For restoration `tar zxvf home.tar.gz -C /`

Compression Using gzip

- **Compression** `gzip arch.tar`
- **Decompression** `gunzip arch*`
- **Reading Compressed Files** Use `zcat` or `zless`

Compression Using bzip2

- First, ensure the necessary software is installed `yum install bzip*`
- **Compression** `bzip2 file`
- **Decompression** `bunzip2 file*`

Compression Using xz

- **Compression**
  - For a single file `xz passwd`
  - While retaining the original `xz -k passwd`
  - For multiple files `xz f1.txt f2.txt f3.txt`
- **Decompression**
  - Basic decompression `xz -d passwd.xz`
- While retaining the .xz original `unxz -k passwd.xz`
- **Inspection and Listing**
  - Compression information `xz -l passwd.xz`
  - View contents without decompression `xzcat passwd.xz`
- **Advanced Compression with xz**
  - Using multiple options like keeping the source, verbose output, extreme mode, and compression ratio `xz -k6ev centos7.iso`

Installing and Updating Software in Red Hat

Using Online Repository

1. Register the system with the Red Hat Subscription Manager
   ```
   subscription-manager register --username user01 --password pass12345 --auto-attach --force
   ```
2. Remove all subscriptions
   ```
   subscription-manager remove --all
   ```
3. Unregister from the Red Hat Subscription Manager
   ```
   subscription-manager unregister
   ```
4. Clean up subscription data
   ```
   subscription-manager clean
   ```

Installing EPEL Repository

- Install the EPEL repository
  ```
  dnf install epel*
  ```

Creating a Local Repository

1. Create a directory for repository data and copy DVD content to it
   ```
   mkdir /mnt/repos
   cp -r /mnt/dvd/AppStream/ /mnt/repos/
   cp -r /mnt/dvd/BaseOS/ /mnt/repos/
   ```
2. Set up local, HTTP, and FTP repositories by appending content to the `ahmed.repo` file
   `cat /etc/yum.repos.d/redhat.repo >> /etc/yum.repos.d/ahmed.repo` Then, edit the `ahmed.repo` to include the repositories.
   ```
   [Ahmed_Repo]
   name=Ahmed Repo (Source DVD)
   baseurl=file///repo/BaseOS
   enabled=1
   gpgcheck=1
   gpgkey=file///repo/RPM-GPG-KEY-redhat-release

   [Ahmed_Repo2]
   name=Ahmed Repo2 (Source DVD)
   baseurl=file///mnt/repos/AppStream
   enabled=1
   gpgcheck=0

   [rhel9base]
   name=RHEL 9 BaseOS
   baseurl=http//mirror.centos.org/centos/9/os/x86_64/
   enabled=1
   gpgcheck=0

   [myftp]
   name=My FTP Repository
   baseurl=ftp//ftp.example.com/pub/rhel/9/x86_64/
   enabled=1
   gpgcheck=0
   ```

Checking and Managing Repositories

- Removes cached packages and metadata `dnf clean all`
- To see all enabled repositories `dnf repolist`
- To list all repositories `dnf repolist all`
- Enable a Repository `dnf config-manager --set-enabled epel`
- Disable a Repository `dnf config-manager --set-disabled epel`
- To add a repository using `dnf dnf config-manager –add-repo="file///repo/BaseOS"`
- To install nginx from the Ahmed_Repo `yum install nginx --disablerepo=* --enablerepo=Ahmed_Repo`

Using dnf

dnf has mostly replaced yum in recent Red Hat-based distributions. It offers similar functionality but with improved dependency resolution and other features.

1. **Queries**
   - **Count all packages** (nstalled, available, and available updates) `dnf list | wc -l`
   - **List All Installed Packages** `dnf list installed`
   - **List Available Packages** `dnf list available`
   - **List Package Information** `dnf info httpd; dnf list httpd`
   - **Search for Packages by Description** `dnf search all web server`
   - **Find package that provides a file or binary** `dnf provides bash`
   - `dnf repoquery --whatprovides webserver`
   - `dnf repoquery --list httpd | grep '^/etc'`
   - **Show Package Dependencies** `dnf deplist httpd`
2. **Package Management**
   - **Install** `dnf install nmap -y`
   - **Install Multiple Packages** `dnf install httpd`
   - **Remove** `dnf remove httpd`
   - **Check for package updates** `dnf check-update`
   - **Update system** `dnf update`
   - **Update a specific package (e.g., kernel)** `dnf update kernel`
   - **Download a Package Without Installing** `dnf download httpd`
   - **download a package along with its dependencies** `dnf download --resolve httpd`
3. **History**
   - View package management history `dnf history`
   - Undo a specific transaction `dnf history undo 21`
   - View a specific transaction `dnf history info 21`
   - Reverts the system to the state it was in just before the transaction 21 `dnf history rollback 21`
4. **Group Management**
   - List groups `dnf group list`
   - Get group details `dnf group info "Virtualization Host"`
   - Install group with optional packages `dnf group install --with-optional "Minimal Install"`
   - Remove group with optional packages `dnf group remove --with-optional "Server with GUI"`
