To Get The Type Of Shell That Linux Operation System Used:

- **Run Terminal**.
- **Run Command**: *echo $SHELL*

```
jafar-loka@jafar-loka-VirtualBox:~/Desktop$ echo $SHELL
/bin/bash
```

****

Understanding the Shell

- The shell is the part of the operating system that interprets commands and translates them to machine code that is executed by the kernel
- Computer operating systems can use either a command line interface (CLI) or graphical user interface (GUI) as shell
- As GUIs often only offer a limited subset of functionality that is available on the CLI, it makes sense to learn how to work with a shell to unleash the operating system full power
- In Linux, Bash is the default shell

****

Understanding File Descriptors

- In UNIX and Linux, a file descriptor is a unique handle for a file
- As everything in Linux is a file, file descriptors are also used for input and output, (including STDIN, STDERR, STDOUT) and pipes and sockets
- Any file that has been opened by a process is seen as a file descriptor
- File descriptors can be dynamically allocated (common), or assigned by an operator
- On Linux, file descriptors are visible through /proc/PID/fd
- Notice that processes commonly hold files open in an indirect way, using sockets

****

Understanding Redirection

- Redirection allows you to send STDOUT and STDERR to a file, and obtain STDIN from a file
- Using STDIN from a file is uncommon, using redirection to send STDOUT and/or STDERR to a file is very common
- As these are techniques that are relevant for shell scripts as well, you should know how it works

****

To Redirect Errors To Specific Output We Can Use:

- Run Command: ***grep root /etc/* 2>/dev/null***

```
jafar-loka@jafar-loka-VirtualBox:~/Desktop$ grep root /etc/* 2>/dev/null
/etc/anacrontab:HOME=/root
/etc/anacrontab:LOGNAME=root
/etc/bash.bashrc:# set variable identifying the chroot you work in (used in the prompt below)
/etc/bash.bashrc:if [ -z "${debian_chroot:-}" ] && [ -r /etc/debian_chroot ]; then
/etc/bash.bashrc:    debian_chroot=$(cat /etc/debian_chroot)
/etc/bash.bashrc:  PS1='${debian_chroot:+($debian_chroot)}\u@\h:\w\$ '
/etc/bash.bashrc:       To run a command as administrator (user "root"), use "sudo <command>".
/etc/bash.bashrc:       See "man sudo_root" for details.
/etc/brltty.conf:# user (e.g. by root on a Linux/Unix system). The configured defaults are:
/etc/ca-certificates.conf:mozilla/Comodo_AAA_Services_root.crt
/etc/crontab:17 *       * * *   root    cd / && run-parts --report /etc/cron.hourly
/etc/crontab:25 6       * * *   root    test -x /usr/sbin/anacron || { cd / && run-parts --report /etc/cron.daily; }
/etc/crontab:47 6       * * 7   root    test -x /usr/sbin/anacron || { cd / && run-parts --report /etc/cron.weekly; }
/etc/crontab:52 6       1 * *   root    test -x /usr/sbin/anacron || { cd / && run-parts --report /etc/cron.monthly; }
/etc/deluser.conf:# Default: NO_DEL_PATHS="^/bin\$ ^/boot\$ ^/dev\$ ^/etc\$ ^/initrd ^/lib ^/lost+found\$ ^/media\$ ^/mn
t\$ ^/opt\$ ^/proc\$ ^/root\$ ^/run\$ ^/sbin\$ ^/srv\$ ^/sys\$ ^/tmp\$ ^/usr\$ ^/var\$ ^/vmlinu"
/etc/deluser.conf:#NO_DEL_PATHS="^/bin\$ ^/boot\$ ^/dev\$ ^/etc\$ ^/initrd ^/lib ^/lost+found\$ ^/media\$ ^/mnt\$ ^/opt\
$ ^/proc\$ ^/root\$ ^/run\$ ^/sbin\$ ^/srv\$ ^/sys\$ ^/tmp\$ ^/usr\$ ^/var\$ ^/vmlinu"
/etc/e2scrub.conf:# recipient=root
/etc/fuse.conf:# user_allow_other - Using the allow_other mount option works fine as root, in
/etc/group:root:x:0:
/etc/group-:root:x:0:
/etc/login.defs:# (examples: 022 -> 002, 077 -> 007) for non-root users, if the uid is
/etc/logrotate.conf:su root adm
/etc/mime.types:application/vnd.cyan.dean.root+xml
/etc/mime.types:application/vnd.dvb.notif-aggregate-root+xml
```

****

Note: We Can Also, Use Multiple Redirection For One Command(For Errors And Output Data):

```
jafar-loka@jafar-loka-VirtualBox:~/Desktop$ grep root /etc/* 2>/dev/null > grepout.txt
jafar-loka@jafar-loka-VirtualBox:~/Desktop$ open grepout.txt
```

****

To Run The Administrator Session:

- **Run Command**: ***sudo -i***
- **Set The Password**

```
jafar-loka@jafar-loka-VirtualBox:~/Desktop$ sudo -i
root@jafar-loka-VirtualBox:~#
```

****

Using Pipes Make The Output Of First Command Is The STDIN For The Second Command:

- Ex: ps aux | less
    - This Will Make The Output Is More Readable In The Current Screen

```
USER         PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root           1  0.5  0.3  23144 13932 ?        Ss   11:27   0:07 /sbin/init splash
root           2  0.0  0.0      0     0 ?        S    11:27   0:00 [kthreadd]
root           3  0.0  0.0      0     0 ?        S    11:27   0:00 [pool_workqueue_release]
root           4  0.0  0.0      0     0 ?        I<   11:27   0:00 [kworker/R-rcu_g]
root           5  0.0  0.0      0     0 ?        I<   11:27   0:00 [kworker/R-rcu_p]
root           6  0.0  0.0      0     0 ?        I<   11:27   0:00 [kworker/R-slub_]
root           7  0.0  0.0      0     0 ?        I<   11:27   0:00 [kworker/R-netns]
root           8  0.2  0.0      0     0 ?        I    11:27   0:03 [kworker/0:0-ata_sff]
root          12  0.0  0.0      0     0 ?        I<   11:27   0:00 [kworker/R-mm_pe]
root          13  0.0  0.0      0     0 ?        I    11:27   0:00 [rcu_tasks_kthread]
root          14  0.0  0.0      0     0 ?        I    11:27   0:00 [rcu_tasks_rude_kthread]
root          15  0.0  0.0      0     0 ?        I    11:27   0:00 [rcu_tasks_trace_kthread]
root          16  1.2  0.0      0     0 ?        S    11:27   0:17 [ksoftirqd/0]
root          17  0.1  0.0      0     0 ?        I    11:27   0:02 [rcu_preempt]
root          18  0.0  0.0      0     0 ?        S    11:27   0:00 [migration/0]
root          19  0.0  0.0      0     0 ?        S    11:27   0:00 [idle_inject/0]
root          20  0.0  0.0      0     0 ?        S    11:27   0:00 [cpuhp/0]
root          21  0.0  0.0      0     0 ?        S    11:27   0:00 [cpuhp/1]
root          22  0.0  0.0      0     0 ?        S    11:27   0:00 [idle_inject/1]
root          23  0.0  0.0      0     0 ?        S    11:27   0:00 [migration/1]
root          24  0.1  0.0      0     0 ?        S    11:27   0:01 [ksoftirqd/1]
root          29  0.0  0.0      0     0 ?        S    11:27   0:00 [kdevtmpfs]
root          30  0.0  0.0      0     0 ?        I<   11:27   0:00 [kworker/R-inet_]
root          31  0.1  0.0      0     0 ?        I    11:27   0:01 [kworker/u5:1-events_unbound]
root          32  0.0  0.0      0     0 ?        S    11:27   0:00 [kauditd]
root          33  0.0  0.0      0     0 ?        S    11:27   0:00 [khungtaskd]
root          34  0.0  0.0      0     0 ?        S    11:27   0:00 [oom_reaper]
root          36  0.0  0.0      0     0 ?        I<   11:27   0:00 [kworker/R-write]
root          37  0.0  0.0      0     0 ?        S    11:27   0:00 [kcompactd0]
root          38  0.0  0.0      0     0 ?        SN   11:27   0:00 [ksmd]
:
```

****

To Find The Command That Are Related With Administrator User:

Note 1: Here 8 is The Administrator User.

Note 2: Here -k user → Search For user in the Man Pages.

```
root@jafar-loka-VirtualBox:/# man -k user | grep 8
adduser.conf (5)      - configuration file for adduser (8) and addgroup (8)
deluser.conf (5)      - configuration file fordeluser(8) and delgroup(8).
addgroup (8)          - add or manipulate users or groups
adduser (8)           - add or manipulate users or groups
adduser.local (8)     - hook for local actions in adduser and deluser
applygnupgdefaults (8) - Run gpgconf --apply-defaults for all users.
arpd (8)              - userspace arp daemon.
delgroup (8)          - remove a user or group from the system
deluser (8)           - remove a user or group from the system
deluser.local (8)     - hook for local actions in adduser and deluser
funcslower-bpfcc (8) - Trace slow kernel or user function calls.
groupmems (8)         - administer members of a user's primary group
lastlog (8)           - reports the most recent login of all users or of a given user
libnss_systemd.so.2 (8) - UNIX user and group name resolution for user/group lookup via Varlink
NetworkManager-dispatcher (8) - Dispatch user scripts for NetworkManager
newusers (8)          - update and create new users in batch
nss-systemd (8)       - UNIX user and group name resolution for user/group lookup via Varlink
ntfsusermap (8)       - NTFS Building a User Mapping File
pam_extrausers (8)    - Module for libnss-extrausers authentication
pam_issue (8)         - PAM module to add issue file to user prompt
pam_localuser (8)     - require users to be listed in /etc/passwd
pam_loginuid (8)      - Record user's login uid to the process attribute
pam_mkhomedir (8)     - PAM module to create users home directory
pam_nologin (8)       - Prevent non-root users from login
pam_systemd (8)       - Register user sessions in the systemd login manager
pam_tty_audit (8)     - Enable or disable TTY auditing for specified users
pam_userdb (8)        - PAM module to authenticate against a db database
pam_usertype (8)      - check if the authenticated user is a system or regular account
pam_xauth (8)         - PAM module to forward xauth keys between users
rdmaucma-bpfcc (8)    - Trace RDMA Userspace Connection Manager Access Event. For Linux, uses BCC, eBPF.
```

****

Understanding Internal Commands

- When working with Linux, external commands are commands that need to be fetched from disk
- To provide core functionality, Bash includes internal commands
- Internal commands are a part of the Bash binary and don't have to be loaded from disk
- Using internal commands is much faster than using external commands
- Type **help** for a list of all internal commands
- Best practice: Use internal commands whenever possible

****

By Typing *help* in Terminal, We List All Internal Commands:

```
root@jafar-loka-VirtualBox:/# help
GNU bash, version 5.2.21(1)-release (x86_64-pc-linux-gnu)
These shell commands are defined internally.  Type `help' to see this list.
Type `help name' to find out more about the function `name'.
Use `info bash' to find out more about the shell in general.
Use `man -k' or `info' to find out more about commands not in this list.

A star (*) next to a name means that the command is disabled.

 job_spec [&]                                                    history [-c] [-d offset] [n] or history -anrw [filename]>
 (( expression ))                                                if COMMANDS; then COMMANDS; [ elif COMMANDS; then COMMAN>
 . filename [arguments]                                          jobs [-lnprs] [jobspec ...] or jobs -x command [args]
 :                                                               kill [-s sigspec | -n signum | -sigspec] pid | jobspec .>
 [ arg... ]                                                      let arg [arg ...]
 [[ expression ]]                                                local [option] name[=value] ...
 alias [-p] [name[=value] ... ]                                  logout [n]
 bg [job_spec ...]                                               mapfile [-d delim] [-n count] [-O origin] [-s count] [-t>
 bind [-lpsvPSVX] [-m keymap] [-f filename] [-q name] [-u >      popd [-n] [+N | -N]
 break [n]                                                       printf [-v var] format [arguments]
 builtin [shell-builtin [arg ...]]                               pushd [-n] [+N | -N | dir]
 caller [expr]                                                   pwd [-LP]
 case WORD in [PATTERN [| PATTERN]...) COMMANDS ;;]... esa>      read [-ers] [-a array] [-d delim] [-i text] [-n nchars] >
 cd [-L|[-P [-e]] [-@]] [dir]                                    readarray [-d delim] [-n count] [-O origin] [-s count] [>
 command [-pVv] command [arg ...]                                readonly [-aAf] [name[=value] ...] or readonly -p
 compgen [-abcdefgjksuv] [-o option] [-A action] [-G globp>      return [n]
 complete [-abcdefgjksuv] [-pr] [-DEI] [-o option] [-A act>      select NAME [in WORDS ... ;] do COMMANDS; done
 compopt [-o|+o option] [-DEI] [name ...]                        set [-abefhkmnptuvxBCEHPT] [-o option-name] [--] [-] [ar>
 continue [n]                                                    shift [n]
 coproc [NAME] command [redirections]                            shopt [-pqsu] [-o] [optname ...]
 declare [-aAfFgiIlnrtux] [name[=value] ...] or declare -p>      source filename [arguments]
 dirs [-clpv] [+N] [-N]                                          suspend [-f]
 disown [-h] [-ar] [jobspec ... | pid ...]                       test [expr]
```

****

To Get Information About Specific Internal Command:

- Run Command: help internal-command-here
  - Ex: *help shift*
  - Ex: *help test*

```
root@jafar-loka-VirtualBox:/# help shift
shift: shift [n]
    Shift positional parameters.

    Rename the positional parameters $N+1,$N+2 ... to $1,$2 ...  If N is
    not given, it is assumed to be 1.

    Exit Status:
    Returns success unless N is negative or greater than $#.
```

```
root@jafar-loka-VirtualBox:/# help test
test: test [expr]
    Evaluate conditional expression.

    Exits with a status of 0 (true) or 1 (false) depending on
    the evaluation of EXPR.  Expressions may be unary or binary.  Unary
    expressions are often used to examine the status of a file.  There
    are string operators and numeric comparison operators as well.

    The behavior of test depends on the number of arguments.  Read the
    bash manual page for the complete specification.

    File operators:

      -a FILE        True if file exists.
      -b FILE        True if file is block special.
      -c FILE        True if file is character special.
      -d FILE        True if file is a directory.
      -e FILE        True if file exists.
      -f FILE        True if file exists and is a regular file.
      -g FILE        True if file is set-group-id.
      -h FILE        True if file is a symbolic link.
      -L FILE        True if file is a symbolic link.
      -k FILE        True if file has its `sticky' bit set.
      -p FILE        True if file is a named pipe.
      -r FILE        True if file is readable by you.
      -s FILE        True if file exists and is not empty.
      -S FILE        True if file is a socket.
      -t FD          True if FD is opened on a terminal.
      -u FILE        True if the file is set-user-id.
      -w FILE        True if the file is writable by you.
      -x FILE        True if the file is executable by you.
```

****

Note (To Remember): To Show The File OR Tool Location:

- Run Command: which command
  - Ex: which test

```
root@jafar-loka-VirtualBox:/# which test
/usr/bin/test
```

****

Understanding Variables

- To have programs (including shell scripts) work with site-specific data, variables are used
- Using variables allows an operating system to keep program code generic and separated from site-specific information
- The Linux operating system itself comes with variables
- Use **env** to print a list of these environment variables
- Best practice: To write efficient shell scripts, use variables a lot

****

The Environment Variables:

```
jafar-loka@jafar-loka-VirtualBox:~/Desktop$ env
SHELL=/bin/bash
SESSION_MANAGER=local/jafar-loka-VirtualBox:@/tmp/.ICE-unix/1989,unix/jafar-loka-VirtualBox:/tmp/.ICE-unix/1989
QT_ACCESSIBILITY=1
COLORTERM=truecolor
XDG_CONFIG_DIRS=/etc/xdg/xdg-ubuntu:/etc/xdg
XDG_MENU_PREFIX=gnome-
GNOME_DESKTOP_SESSION_ID=this-is-deprecated
GNOME_SHELL_SESSION_MODE=ubuntu
SSH_AUTH_SOCK=/run/user/1000/keyring/ssh
MEMORY_PRESSURE_WRITE=c29tZSAyMDAwMDAgMjAwMDAwMAA=
XMODIFIERS=@im=ibus
DESKTOP_SESSION=ubuntu
GTK_MODULES=gail:atk-bridge
PWD=/home/jafar-loka/Desktop
LOGNAME=jafar-loka
XDG_SESSION_DESKTOP=ubuntu
XDG_SESSION_TYPE=wayland
SYSTEMD_EXEC_PID=2028
XAUTHORITY=/run/user/1000/.mutter-Xwaylandauth.L5Y6X2
HOME=/home/jafar-loka
USERNAME=jafar-loka
IM_CONFIG_PHASE=1
LANG=en_US.UTF-8
LS_COLORS=rs=0:di=01;34:ln=01;36:mh=00:pi=40;33:so=01;35:do=01;35:bd=40;33;01:cd=40;33;01:or=40;31;01:mi=00:su=37;41:sg=
30;43:ca=00:tw=30;42:ow=34;42:st=37;44:ex=01;32:*.tar=01;31:*.tgz=01;31:*.arc=01;31:*.arj=01;31:*.taz=01;31:*.lha=01;31:
*.lz4=01;31:*.lzh=01;31:*.lzma=01;31:*.tlz=01;31:*.txz=01;31:*.tzo=01;31:*.t7z=01;31:*.zip=01;31:*.z=01;31:*.dz=01;31:*.
gz=01;31:*.lrz=01;31:*.lz=01;31:*.lzo=01;31:*.xz=01;31:*.zst=01;31:*.tzst=01;31:*.bz2=01;31:*.bz=01;31:*.tbz=01;31:*.tbz
2=01;31:*.tz=01;31:*.deb=01;31:*.rpm=01;31:*.jar=01;31:*.war=01;31:*.ear=01;31:*.sar=01;31:*.rar=01;31:*.alz=01;31:*.ace
=01;31:*.zoo=01;31:*.cpio=01;31:*.7z=01;31:*.rz=01;31:*.cab=01;31:*.wim=01;31:*.swm=01;31:*.dwm=01;31:*.esd=01;31:*.avif
=01;35:*.jpg=01;35:*.jpeg=01;35:*.mjpg=01;35:*.mjpeg=01;35:*.gif=01;35:*.bmp=01;35:*.pbm=01;35:*.pgm=01;35:*.ppm=01;35:*
.tga=01;35:*.xbm=01;35:*.xpm=01;35:*.tif=01;35:*.tiff=01;35:*.png=01;35:*.svg=01;35:*.svgz=01;35:*.mng=01;35:*.pcx=01;35
:*.mov=01;35:*.mpg=01;35:*.mpeg=01;35:*.m2v=01;35:*.mkv=01;35:*.webm=01;35:*.webp=01;35:*.ogm=01;35:*.mp4=01;35:*.m4v=01
```

****

Using Variables

- To define a variable, use **key=value**
- By default, variables will be available in the current shell only
- To define a variable for the current shell and all subshells, use **export key=value**
- To make sure a variable is automatically set, put it in one of the Bash startup files
- Refer to a variable using **echo $key**
- To avoid ambiguity, use **echo ${key}**; compare **echo ${key}1** with **echo $key1**

****

```
jafar-loka@jafar-loka-VirtualBox:~/Desktop$ echo $SHELL
/bin/bash
jafar-loka@jafar-loka-VirtualBox:~/Desktop$ echo $PATH
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games:/snap/bin:/snap/bin
jafar-loka@jafar-loka-VirtualBox:~/Desktop$ echo $LANG
en_US.UTF-8
```

****

```
jafar-loka@jafar-loka-VirtualBox:~/Desktop$ myvar=green
jafar-loka@jafar-loka-VirtualBox:~/Desktop$ echo $myvar
green
jafar-loka@jafar-loka-VirtualBox:~/Desktop$ echo ${myvar}1
green1
```

****

```
jafar-loka@jafar-loka-VirtualBox:~/Desktop$ myvar=green
jafar-loka@jafar-loka-VirtualBox:~/Desktop$ echo $myvar
green
jafar-loka@jafar-loka-VirtualBox:~/Desktop$ echo ${myvar}1
green1
jafar-loka@jafar-loka-VirtualBox:~/Desktop$ bash
jafar-loka@jafar-loka-VirtualBox:~/Desktop$ echo $myvar

jafar-loka@jafar-loka-VirtualBox:~/Desktop$ exit
exit
jafar-loka@jafar-loka-VirtualBox:~/Desktop$ export $myvar
jafar-loka@jafar-loka-VirtualBox:~/Desktop$ bash
jafar-loka@jafar-loka-VirtualBox:~/Desktop$ echo $myvar

jafar-loka@jafar-loka-VirtualBox:~/Desktop$ exit
exit
jafar-loka@jafar-loka-VirtualBox:~/Desktop$ export myvar=green
jafar-loka@jafar-loka-VirtualBox:~/Desktop$ echo $myvar
green
jafar-loka@jafar-loka-VirtualBox:~/Desktop$ bash
jafar-loka@jafar-loka-VirtualBox:~/Desktop$ echo $myvar
green
```

****

Understanding **alias**

- **alias** is a Bash internal command that allows you to define your own commands
- It is convenient to use **alias** to replace long commands with something shorter
- Type **alias** in a shell to get a list of current aliases
- Best practice: Do NOT use alias in shell scripts, as alias settings are not universal and might not exist on other computers where the shell script is used

****

```
jafar-loka@jafar-loka-VirtualBox:~/Desktop$ alias sander='man useradd'
jafar-loka@jafar-loka-VirtualBox:~/Desktop$ sander
jafar-loka@jafar-loka-VirtualBox:~/Desktop$ unalias sander
jafar-loka@jafar-loka-VirtualBox:~/Desktop$ sander
sander: command not found
```

****