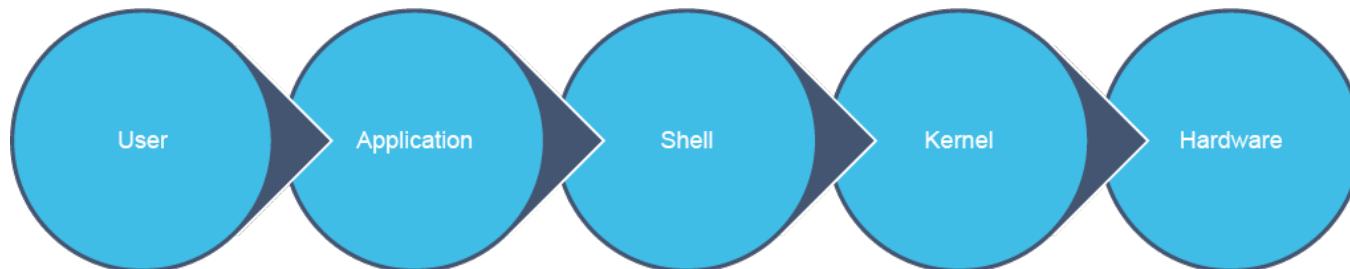


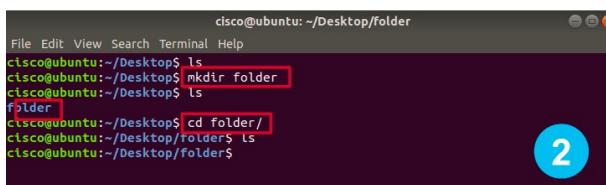
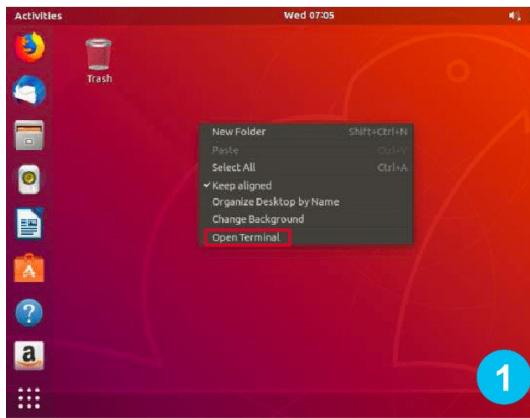
# LINUX BASH

- A Shell is a software layer between you and the operating system (Typically Linux, but now Windows too)
- Bourne Again Shell (Bash) is the default shell for UNIX-like operating systems, such as CentOS and Ubuntu, and conforms to the IEEE Portable Operating System Interface (POSIX) P1003.2/ISO 9954.2 Shell and Tool standard.
- It is command line driven, heavily scriptable, and uses standard commands that are compatible across operating systems



- There are lots of shells that you can use:
  - **bash**: The Bourne again shell is the default for many Linux distribution
  - **zsh**: the Z Shell is a modern variation of BASH type shells. It is fast, has command spellchecks, auto completion, and over 400 plugins
  - **ksh**: the Korn Shell Known for its very strong scripting language
  - **csh**: C Shell written in the 70s and one of the oldest open-source Unix shells
- Want to know what Shells are available on your system?
  - Type **cat /etc/shells** at the command prompt
- To change your shell to Zshell for example **chsh -s /bin/zsh**

# Getting Started with Bash



A screenshot of a terminal window on an Ubuntu system. The user has run the command `man cp`, which displays the manual page for the `cp` command. The terminal shows the following text:

```
cisco@ubuntu:~/Desktop/folder
File Edit View Search Terminal Help
cisco@ubuntu:~/Desktop$ ls
cisco@ubuntu:~/Desktop$ mkdir folder
cisco@ubuntu:~/Desktop$ ls
[redacted]
cisco@ubuntu:~/Desktop$ cd folder/
cisco@ubuntu:~/Desktop/folder$ ls
cisco@ubuntu:~/Desktop/folder$
```

A blue circle with the number '2' is positioned in the bottom right corner of the terminal window.

**command –flag argument**

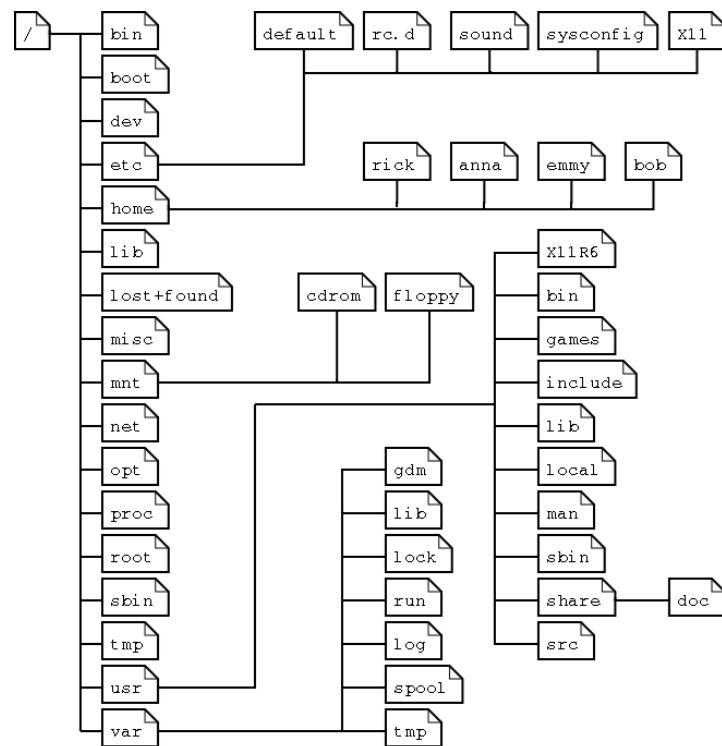
```
student@student-vm:~$ man cp
<... output omitted ...>
DESCRIPTION
Copy SOURCE to DEST, or multiple SOURCE(s)
-a, --archive    same as -dR --preserve=all
<... output omitted ...>
-b      like --backup but does not accept an argument
<... output omitted ...>
-f, --force
Manual page cp(1) line 1 (press h for help or q to quit)
```

# Common Directories

Dir	Description
/	The directory called "root." It is the starting point for the file system hierarchy. Note that this is not related to the root, or superuser, account.
/bin	Binaries and other executable programs.
/etc	System configuration files.
/home	Home directories.
/opt	Optional or third party software.
/tmp	Temporary space, typically cleared on reboot.
/usr	User related programs.
/var	Variable data, most notably log files.

# Directory Navigation with Bash

- **Pwd**
- **Cd**
- **Ls**



# Pwd command

If you ever get lost while navigating around the file system, you can use the **pwd** command to print out your current working directory path. You can use it as follows:

```
$ pwd Print your current working directory
```

**\$ cd /** Changes directory to the root directory

**\$ cd /home/username** Changes directory to the /home/username directory

**\$ cd test** Changes directory to the test folder

**\$ cd ..** Moves up one directory

# Directory Shortcuts

- This directory
  - .. The parent directory
  - cd – Change to the previous directory
- ~ home directory
- / root directory

# ls command

**\$ ls**

Lists files and directories in the current working directory

**\$ ls -a**

Lists everything in the current directory, including hidden files

**\$ ls /home/username**

Lists everything in the /home/username directory

**\$ ls -l**

Lists permissions and user and group ownership

**\$ ls -F**

Displays files and directories and denotes which are which

**\$ mkdir test**

Makes a new directory called test in the current working directory if you have permission

**\$ mkdir  
/home/username/test**

Makes a new directory called test at /home/username/test

**Touch:** create/make files

**Rm:** remove files

**Cp:** Copy files

# Touch command

**\$ touch emptyfile.txt** Creates an empty file named emptyfile.txt

**\$ touch file{1..20}.txt** Bulk creates files from file1.txt to file20.txt

**\$ cp sydney.txt sydney2.txt**

Copies a file called sydney.txt from the current directory and names the copy sydney2.txt

**\$ cp /home/username/sydney.txt ~/sydney2.txt**

Copies a file as described above but using the full path and the home directory path

**\$ cp -r folder folder.old**

Copies a folder

```
$ mv caleb.txt calebfinal.txt
```

Renames a file called caleb.txt to calebfinal.txt

```
$ mv /home/username/caleb.txt  
~/calebfinal.txt
```

Renames a file as described above but using full paths

```
$ mv -i * /home/username/new/
```

Moves all files and directories in the current folder to a directory called new

**\$ rm test.txt** Deletes the file test.txt in the current working directory

**\$ rm -rf test** Forces the deletion of the folder test and everything in it

- First there was **cat**. Output was streamed in an uncontrollable way.
- Then there was **pg**, which may still be found on older UNIXes. This command puts text to the output one page at the time.
- The **more** program was a revised version of **pg**. This command is still available on every Linux system.
- **less** is the GNU version of more and has extra features allowing highlighting of search strings, scrolling back etc. The syntax is very simple:

**less name\_of\_file**

**\$cat file1.txt**      Displays the contents of file1.txt

**\$cat file1.txt | more**      Displays the contents of file1.txt and pipes the output to **more** to add page breaks

**\$cat >file2.txt**      Sends a user's typed or copied content from the command line to file2.txt

`vi [file]` : Edit file.

## Vi line mode

`:w` Writes (saves) the file.

`:w!` Forces the file to be saved. `:q` Quit.

`:q!` Quit without saving changes. `:wq!` Write and quit.

`:x` Same as `:wq`

## Vi insert mode

`i` Insert at the cursor position.

`I` Insert at the beginning of the line.

`a` Append after the cursor position.

`A` Append at the end of the line

- **Vi - Deleting Text**

x Delete a character.

dw Delete a word.

dd Delete a line.

D Delete from the current position

:[from],[to]d Delete a Range of Lines

:%d Delete All Lines

- **Vi Navigation Keys**

^ Go to the beginning of the line

\$ Go to the end of the line

# Editing file with Nano

Shortcut	Description
nano filename	Open file for editing in Nano
Arrow keys	Move cursor up, down, left and right
Ctrl+A, Ctrl+E	Move cursor to start and end of the line
Ctrl+Y/Ctrl+V	Move page up and down
Ctrl+_	Move cursor to a certain location
Alt+A and then use arrow key	Set a marker and select text
Alt+6	Copy the selected text
Ctrl+K	Cut the selected text
Ctrl+U	Paste the selected text
Ctrl+6	Cancel the selection
Ctrl+K	Cut/delete entire line
Alt+U	Undo last action
Alt+E	Redo last action
Ctrl+W, Alt+W	Search for text, move to next match
Ctrl+\	Search and replace
Ctrl+O	Save the modification
Ctrl+X	Exit the editor

# **USER AND GROUP MANAGEMENT**

# Linux user/group CLI



useradd  
passwd  
usermod  
Groupadd  
groups [username]  
Userdel  
groupdel  
cat /etc/group  
Cat /etc/passwd

```
root:x:0:0:root:/root:/bin/bash
```

The format of the /etc/passwd:

```
username:password:UID:GID:comments:  
home_dir:shell
```

`useradd [options] username`

- `-c "COMMENT"`      Comments for the account.
- `-m`                      Create the home directory.
- `-s /shell/path`        The path to the user's shell.

# Create a password using passwd

```
# passwd grant
```

```
Enter new UNIX password:
```

```
Retype new UNIX password:
```

```
passwd: password updated  
successfully
```

```
#
```

# Account information for "grant"



```
# tail -1 /etc/passwd
grant:x:1000:1000:Grant Stewart:/home/grant
:/bin/bash
# tail -1 /etc/shadow
grant:$6$IDDsgsPYtR8c2Uc.:16507:0:99999:7:::
```

# More useradd options



**-g GROUP**

Specify the default group.

**-G GROUP1,GRUPN** Additional groups.

`usermod [ options ] username`

- |                         |                            |
|-------------------------|----------------------------|
| <b>-c "COMMENT"</b>     | Comments account.          |
| <b>-g GROUP</b>         | Specify the default group. |
| <b>-G GROUP1,GROUPN</b> | Additional groups.         |
| <b>-s /shell/path</b>   | Path to the user's shell.  |

root:x:0:

sales:x:1001:john,mary

The format of the /etc/group file:

group\_name:password:GID:account1, accountN

# groups [username]



```
osboxes@osboxes:~$ groups osboxes
osboxes : osboxes adm cdrom sudo dip plugdev lpadmin
lxd sambashare
```

# groupadd [ options ] group\_name

```
# groupadd web
# tail -1 /etc/group
web:x:1003:
# groupadd -g 2500 db
# tail -1 /etc/group
db:x:2500:
#
```

# groupdel group\_name



```
# groupdel db  
#
```

# groupmod

groupmod [options] group\_name

- g GID        Change the group ID to GID.
- n GROUP      Rename the group to GROUP.

```
# grep web /etc/group
web:x:1003:
# groupmod -g 1234 web
# grep web /etc/group
web:x:1234:
# groupmod -n http web
# grep http /etc/group
http:x:1234:
```

# **FILE AND DIRECTORY PERMISSION**

# Permision

\$ ls -

```
drwxr-xr-x 2 devnet devnet 4096 Mar 29 23:54 Desktop
drwxr-xr-x 2 devnet devnet 4096 Mar 29 23:54 Documents
drwxr-xr-x 2 devnet devnet 4096 Mar 29 23:54 Downloads
drwxr-xr-x 2 devnet devnet 4096 Mar 29 23:54 Music
-rw-r--r-- 1 root  root  4122 Mar  7 11:23 README.signature
```

Symbol	Meaning
-	Regular file
d	Directory
l	Symbolic link
r	Read
w	Write
x	Execute

# Permissions - Files vs Directories



Permission	File	Directory
Read (r)	Allows a file to be read.	Allows file names in the directory to be read.
Write (w)	Allows a file to be modified.	Allows entries to be modified within the directory.
Execute (x)	Allows the execution of a file.	Allows access to contents and metadata for entries.

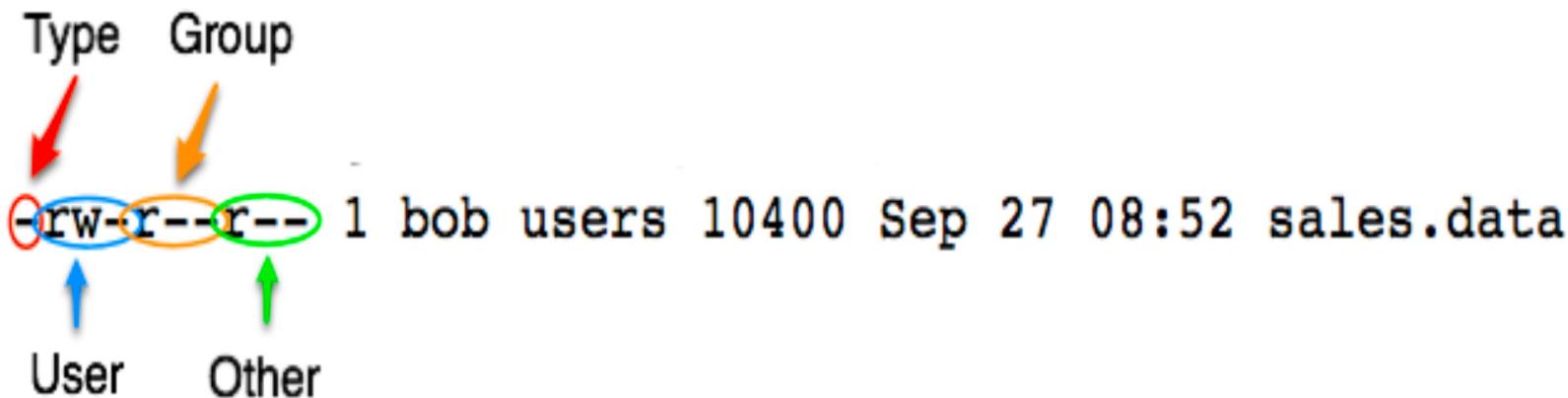
# Permission Categories

Symbol	Category
u	User
g	Group
o	Other
a	All

# Secret Decoder Ring

Type Group  
User Other

-rw-r--r-- 1 bob users 10400 Sep 27 08:52 sales.data



# Changing Permissions

Item	Meaning
chmod	change mode command
ugoa	User category user, group, other, all
+--	Add, subtract, or set permissions
rwx	Read, Write, Execute

# Numeric Based Permissions

r	w	x	
0	0	0	Value for off
1	1	1	Binary value for on
4	2	1	Base 10 value for on

# Numeric Based Permissions

Octal	Binary	String	Description
0	0	---	No permissions
1	1	--x	Execute only
2	10	-w-	Write only
3	11	-wx	Write and execute (2+1)
4	100	r--	Read only
5	101	r-x	Read and execute (4+1)
6	110	rw-	Read and write (4 + 2)
7	111	rwx	Read, write and execute (4+2+1)

# Order Has Meaning

	U	G	O
Symbolic	rwx	r-x	r--
Binary	111	101	100
Decimal	7	5	4

# Commonly Used Permissions

Symbolic	Octal
-rwx-----	700
-rwxr-xr-x	755
-rw-rw-r--	664
-rw-rw----	660
-rw-r--r--	644

```
$ ls -l test.sh
-rw-rw-r-- 1 devnet devnet 11 Mar 30 04:12 test.sh
$ chmod u+x test.sh
$ ls -l test.sh
-rwxrw-r-- 1 devnet devnet 11 Mar 30 04:12 test.sh
$ chmod g+x test.sh
$ ls -l test.sh
-rwxrwxr-- 1 devnet devnet 11 Mar 30 04:12 test.sh
$ chmod o+wx test.sh
$ ls -l test.sh
-rwxrwxrwx 1 devnet devnet 11 Mar 30 04:12 test.sh
$ ./test.sh
hello
$ chmod 0 test.sh
$ ls -l test.sh
----- 1 devnet devnet 11 Mar 30 04:12 test.sh
```

# BASH SCRIPTING

# Environmental Variables with Bash

- **env**: To get to know the current environment, with all its variables

```
$env | more Shows all environment variables with page breaks
```

- **Export**: Set export attribute for shell variables.  
export: export [-fn] [name[=value] ...] or export -p
- **unset** - delete an environment variable.
- **printenv**  
Print all or part of environment
- **echo \$ENV\_VAR**  
Print the ENV\_VAR variable

# Common Environment Variables



Variable	Description
EDITOR	Program to run to perform edits
HOME	Home directory of the user.
LOGNAME	The login name of the user.
MAIL	The location of the user's local inbox.

## Variable

OLDPWD

PATH

PAGER

## Description

The previous working directory.

A list of directions to search for commands.

Program used to paginate through files.

Variable	Description
PS1	The primary prompt string.
PWD	Present working directory.
USER	The username of the current user.

# Variable Assignment

- `VARIABLE_NAME="Value"` (*no space before and after*)
- Variables are case sensitive
- By convention variables are uppercase

```
osboxes@ubuntu:~/Documents/devnet/linux_bash$  
osboxes@ubuntu:~/Documents/devnet/linux_bash$ export a=5  
osboxes@ubuntu:~/Documents/devnet/linux_bash$ echo $a  
5  
osboxes@ubuntu:~/Documents/devnet/linux_bash$  
osboxes@ubuntu:~/Documents/devnet/linux_bash$ export b=a  
osboxes@ubuntu:~/Documents/devnet/linux_bash$ echo $b  
a  
osboxes@ubuntu:~/Documents/devnet/linux_bash$ █
```

- Contain a series of commands.
- An interpreter executes commands in the script.
- Anything you can type at the command line, you can put in a script.
- Great for automating tasks.

# Writing the script

```
osboxes@ubuntu:~/Documents/devnet/linux_bash$ cat script1.sh
echo "The script starts now"
echo "Hi $USER !"
echo
echo "I will now fetch you a list of connected users"
echo
w
echo "I am setting two variables"
COLOUR="BLACK"
VALUE=9

echo "This is a string: $COLOUR"
echo "And this is a number: $VALUE"
echo
echo "i'm giving you back your prompt"
echo
```

```
clear
echo "The script starts now"
echo "Hi $USER !"
echo
echo "I will now fetch you a list of connected
users"
echo
w
echo "I am setting two variables"
COLOUR="BLACK"
VALUE=9

echo "This is a string: $COLOUR"
echo "And this is a number: $VALUE"
echo
echo "i'm giving you back your prompt"
echo
```

# Executing the script

```
osboxes@ubuntu:~/Documents/devnet/linux_bash$ ./script1.sh
The script starts now
Hi osboxes !

I will now fetch you a list of connected users

10:53:10 up 56 min, 2 users, load average: 0.00, 0.00, 0.00
USER     TTY      FROM           LOGIN@   IDLE    JCPU   PCPU WHAT
osboxes :0      :0            09:48 ?xdm?  29.30s  0.00s /usr/libexec/gdm-x-session --run-scr
osboxes pts/1    192.168.1.2  09:50    2.00s  0.26s  0.00s w
I am setting two variables
This is a string: BLACK
And this is a number: 9

i'm giving you back your prompt

osboxes@ubuntu:~/Documents/devnet/linux_bash$ /home/osboxes/Documents/devnet/linux_bash/script1.sh
The script starts now
Hi osboxes !

I will now fetch you a list of connected users

10:53:37 up 57 min, 2 users, load average: 0.14, 0.03, 0.01
USER     TTY      FROM           LOGIN@   IDLE    JCPU   PCPU WHAT
osboxes :0      :0            09:48 ?xdm?  29.39s  0.00s /usr/libexec/gdm-x-session --run-scr
osboxes pts/1    192.168.1.2  09:50    1.00s  0.26s  0.00s -bash
I am setting two variables
This is a string: BLACK
And this is a number: 9

i'm giving you back your prompt
```