Informational, Navigational & Management Command

Overview of Common Linux Shell Commands

What is a shell?

- User interface for running commands
- Interactive language
- Scripting language



A sea of shells

- Default shell is usually Bash
- Many other shells, including sh, ksh, tcsh, zsh, and fish
- This course uses Bash → strand for bourne again shell

return path to the default shell
$ printenv SHELL
/bin/bash ← this case
$ bash
if not bash, we can switch

$ command prompt
> same purpose with dollar sign

Commands

Shell command applications

- Getting information
- Navigating and working with files and directories
- Printing file and string contents
- Compression and archiving
- Performing network operations
- Monitoring performance and status
- Running batch jobs

Getting information

Some common shell commands for getting information include:

- `whoami` – username
- `id` – user ID and group ID
- `uname` – operating system name
- `ps` – running processes
- `top` – resource usage
- `df` – mounted file systems
- `man` – reference manual
- `date` – today's date

Working with files

Some common shell commands for working with files include:

- `cp` – copy file
- `mv` – change file name or path
- `rm` – remove file
- `touch` – create empty file, update file timestamp
- `chmod` – change/modify file permissions
- `wc` – get count of lines, words, characters in file
- `grep` – return lines in file matching pattern

Navigating and working with directories

Common shell commands for navigating and working with directories include:

- `ls` – list files and directories
- `find` – find files in directory tree
- `pwd` – get present working directory
- `mkdir` – make directory
- `cd` – change directory
- `rmdir` – remove directory

Printing file and string contents

For printing file contents or strings, common commands include:

- `cat` – print file contents
- `more` – print file contents page-by-page
- `head` – print first N lines of file
- `tail` – print last N lines of file
- `echo` – print string or variable value

Compression and archiving

Shell commands related to file compression and archiving applications include:

- `tar` – archive a set of files
- `zip` – compress a set of files
- `unzip` – extract files from a compressed zip archive

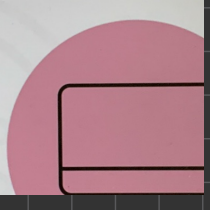
Networking

Networking applications include the following:

- `hostname` – print hostname
- `ping` – send packets to URL and print response
- `ifconfig` – display or configure system network interfaces
- `curl` – display contents of file at a URL
- `wget` – download file from URL

Running Linux on a Windows machine

- Dual boot with a partition
- Install Linux on a virtual machine
- Use a Linux emulator
- Windows Subsystem for Linux (WSL)



Informational Commands

User information

- Display user information
- Verify identity or identify user account

- whoami – returns user name
- id (identity) – user or group ID

```
$ whoami        → display current user's name
johndoe         no argument and option

$ id -u         → number assigned to each
501               user or group
$ id -u -n      → option to get name
johndoe           that corresponding to
                  the numerical user ID
```

System information

uname (Unix name) – returns OS information (like kernel's name or version number)
- Identify system or diagnose issues

```
$ uname
Darwin          ← name of OS system
$ uname -s -r
Darwin 20.6.0   } name and Version
$ uname -v      → detail
Darwin Kernel Version 20.6.0: Mon Aug 30 06:12:21 PDT 2021;
root:xnu-7195.141.6~3/RELEASE_X86_64
```

Displaying your disk usage → to check the available space

df (disk free) – Shows disk usage
- Monitor disk usage or check space

```
$ df -h ~   → Human reable      (~ : specific to home directory)
Filesystem       Size   Used  Avail  Use%  Mounted on
/dev/nvme0n1p2   2.0T   744G  1.2T   40%   /home      directory home
```

Getting disk usage information

df (disk free) – Shows disk usage

```
$ df -h
Filesystem       Size  Used  Avail Use% Mounted on
udev              26G     0    26G   0% /dev
tmpfs            5.1G  2.6M  5.1G    1% /run
/dev/nvme1n1p5   255G   65G  177G   27% /
tmpfs             26G  223M   25G    1% /dev/shm
tmpfs            5.3M  4.1k  5.3M    1% /run/lock
tmpfs             26G     0   26G    0% /sys/fs/cgroup
/dev/loop2       230M  230M     0  100% /snap/gnome-3-34-1804/66
/dev/loop0       132k  132k     0  100% /snap/bare/5
/dev/loop1
```

The output includes the size, capacity used...

Displaying current running processes

ps (process status) – Running processes
- Monitor or manage processes

```
$ ps -e        (process ID)
PID  TTY       TIME  CMD        (time: min / second; name of each running process)
  1  ??     8:15.69  /sbin/launchd
 76  ??     0:13.27  /usr/sbin/syslogd
```

Monitoring system health and status

top (table of processes) – Task manager
- Monitor system performance and resource usage

```
$ top -n 3   → top 3 running tasks     (-n : option)
PID    COMMAND   %CPU  TIME      ...  USER     ...
38702  chrome    10.0  01:00.41  ...  johndoe  ...
38701  top        4.0  00:00.09  ...  johndoe  ...
38699  Spotify    3.0  01:00.07  ...  johndoe  ...
```
(top 3; cpu usage)

Printing strings and variable values

echo – Print string or variable value

```
$ echo

$ echo Hello
Hello                          ← return the word
$ echo "Learning Linux is fun!"    with quote cause there are spaces in this text
Learning Linux is fun!
$ echo $PATH   → dollar sign + PATH       each path of our system's path
/usr/local/bin:/usr/bin:/bin:/usr/sbin:/sbin
                                   ↓ deliminator by colon
```

```
Welcome to the linux lab.
```

These special characters help you better format your output:

Special Character	Effect
\n	Start a new line
\t	Insert a tab

Use the -e (to use) option of the echo command when working with special characters. For example:

```
echo -e "This will be printed \nin two lines"
```

will print:

```
This will be printed
in two lines
```

Getting date information

date – Displays system date and time

```
$ date
Thu 16 Sep 2021 16:50:49 EDT
$ date "+%j day of %Y"
097 day of 2023
$ date "+It's %A, the %j day of %Y!"
It's Friday, the 097 day of 2023!
```

Viewing the manual

man (manual) – Shows manual for any command

```
$ man id
NAME
     id -- return user identity

SYNOPSIS
     id [user]
     id -A
     id -F [user]
...

DESCRIPTION
     The id utility displays the user and group names and numeric
IDs, of the calling process, to the standard output.  If the real...
```

1.8. Display date and time

`date`

The `date` command displays the current date and time.

```
date
```

It has several options which allow you to display the current date and time in different formats.

For example, the following command displays the current date in `mm/dd/yy` format:

```
date "+%D"
```

Here are some popular format specifiers that you can try out:

Specifier	Explanation
`%d`	Displays the day of the month (01 to 31)
`%h`	Displays the abbreviated month name (Jan to Dec)
`%m`	Displays the month of year (01 to 12)
`%Y`	Displays the four-digit year
`%T`	Displays the time in 24 hour format as HH:MM:SS
`%H`	Displays the hour

File and Directory Navigation Commands

Listing your directory contents

`ls` (list) – list files and directories

```
$ ls → showed the directories that /home contains
Documents Downloads Music Pictures
$ ls Downloads ← list contents in Downloads folder
download1.zip
download2.zip
download3.zip
```

Listing your directory contents

`ls` (list) – list files and directories

```
$ ls -l
permission | owner | | last modified | child files
-rwxr-xr-x me staff 21 Sep 06:45 assignment-1.txt
-rwxr-xr-x me staff 09 Feb 03:27 assignment-2.txt
-rwxr-xr-x me staff  1 Jan 01:23 notes-1.txt
-rwxr-xr-x me staff  3 Aug 10:03 notes-2.txt
-rwxr-xr-x me staff  7 Nov 16:21 notes-3.txt
-rwxr-xr-x re staff 27 Sep 04:56 notes-4.txt
details such as permissions last modified
```

show child files and directories for more detail

Finding your working directory

`pwd` (print working directory) – get current working directory

```
$ pwd
/Users/me
```

Navigating your directories

`cd` (change directory) – change directory

```
$ pwd
/Users/me ← in home directory
$ ls
Documents Downloads Music Pictures
$ cd Documents ← changing
$ pwd
/Users/me/Documents ← changed
```

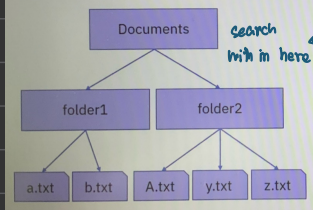
Relative and absolute navigation

`cd` (change directory) – change directory

```
$ pwd ↓ in notes folder
/Users/me/Documents/Academics/Math/Notes
$ cd .. ← to go to parent directory
$ pwd
/Users/me/Documents/Academics/Math
$ cd ~ → navigate to home folder
$ pwd
/Users/me
$ cd /Users/me/Documents/Academics/Math/Notes
$ pwd ↑ roll back to Notes folder
/Users/me/Documents/Academics/Math/Notes
```

Finding files

`find` – find files in directory tree




```
$ pwd ↓ find in Documents folder
/Users/me/Documents
$ find . -name "a.txt" ← put file's name
./folder1/a.txt
$ find . -iname "a.txt"
./folder1/a.txt
./folder2/A.txt ← with another file in the same name, but upper case 'A'
```

File and Directory Management Commands

Creating directories

mkdir (make directory) – Make directory

```
$ pwd
/Users/me/Documents
$ ls
$ mkdir test
$ ls
test
```

new dir. name

when you've entered, then you already got directory

← check

← created

Removing files and directories

rm (remove) – Remove file or directory


```
$ pwd
/Users/me/Documents
$ ls
file1 folder1
$ rm file1
$ ls
folder1
$ rm folder1
rm: folder1: is a directory
$ rm -r folder1
$ ls
$ mkdir empty_folder
$ rmdir empty_folder
$ ls
```

← removed file1 already

↳ removed dir., that folder1 was contained

* not recommend, cause it's easy to accidentally

→ recommend, it's used solely to remove

Creating files

touch – Create empty file, update file date

```
$ pwd
/Users/me/Documents
$ touch a.txt b.txt c.txt d.txt
$ ls
a.txt b.txt c.txt d.txt
$ date -r notes.txt
Mon  8 Nov 2021 16:37:45 EST
$ touch notes.txt
$ date -r notes.txt
Fri 12 Nov 2021 10:46:03 EST
```

← created 4 files

last modified

last modified update

Copying files and directories

cp (copy) – Copy file or directory to destination

↳ เป็นการ copy file ไปอีกที่ โดยไม่ต้องไป copy paste ที่อื่น, เอาไปที่อื่นเลย

To copy files:

```
cp /source/file /dest/filename
cp /source/file /dest/
```

ระบุแค่ แต่ไม่ได้ใส่ชื่อ ไฟล์ที่ folder

To copy directories:

```
cp -r /source/dir/ /dest/dir/
```

```
$ ls
notes.txt Documents
$ cp notes.txt Documents
$ ls Documents
notes.txt
$ cp -r Documents Docs_copy
$ ls
notes.txt Documents Docs_copy
$ ls Docs_copy
notes.txt
```

copy this → to this folder

→ check

copy to

Now it's already duplicated

Moving files and directories

mv (move) - Move a file or directory → move with code

To move files:

```
mv /source/file /dest/dir/
```

To move directories:

```
mv /source/dir/ /dest/dir/
```

```
$ ls
my_script.sh Scripts Notes Documents
$ mv my_script.sh Scripts
$ ls my_script.sh

$ ls Scripts
my_script.sh
$ mv Notes Scripts Documents
$ ls
Documents
$ ls Documents
Scripts Notes
```

file

Destination

move 2 folders to

Managing file permissions

chmod (change mode) – Change file permissions

```
$ ls -l my_script.sh
-rw-r--r-- my_script.sh
$ ./my_script.sh
bash: permission denied: ./my_script.sh
$ chmod +x my_script.sh
$ ls -l
-rwxr-xr-x my_script.sh
$ ./my_script.sh
Learning Linux is fun!
```

→ show permission

read write

can't access/read contents in this file

option

changed access

now can access

Lab

Exercise 1 - Navigating Files and Directories

In these exercises, you will practice using commands for navigating and managing files and directories.

1.1. Get the location of the present working directory

`pwd`

When working in a Linux terminal, you will always be working from a directory. By default, you will start in your home directory. To get the absolute path of your present working directory, enter the following:

```
pwd
```

This will print the name of the directory you are currently working in.

1.2. List the files and directories in a directory

`ls`

To list the files and directories in the current directory, enter the following:

```
ls
```

If your directory happens to be empty, `ls` will not return anything.

The following command will list the many binary and executable files which are present in your `/bin` (binaries) directory.

```
ls /bin
```

The `/bin` directory happens to be where Linux commmands such as `ls` and `pwd` are stored. For example, you can see that `ls` is present by entering the following:

```
ls /bin/ls
```

To list all files starting with `b` in the `/bin` directory, try entering the following:

```
ls /bin/b*
```

> ***Tip:*** *The asterisk `*` is a special character called a wildcard. It is used to represent any string of characters.*

To list all files ending in `r` in the `/bin` directory, enter the following:

```
ls /bin/*r
```

To print a longer list of files with additional information, such as the last-modified date, enter the following:

```
ls -l
```

Here are some common options that you can try with the `ls` command:

Option	Description
`-a`	list all files, including hidden files
`-d`	list directories only, do not include files
`-h`	with `-l` and `-s`, print sizes like 1K, 234M, 2G
`-l`	include attributes like permissions, owner, size, and last-modified date
`-S`	sort by file size, largest first
`-t`	sort by last-modified date, newest first
`-r`	reverse the sort order

To get a long list of all files in `/etc`, including any hidden files, enter the following:

```
ls -la /etc
```

Here we combined the options `-l` and `-a` by using the shorter notation, `-la`.

Exercise 2 - Creating Files and Directories

2.1. Create a directory

`mkdir`

The `mkdir` command is used to create a new directory.

To create a directory named `scripts` in your current directory, run the following command:

```
mkdir scripts
```

Use the `ls` command to verify whether the `scripts` directory was created:

```
ls
```

You should see a directory named `scripts` listed.

2.2. Change your current working directory

`cd`

To change your present working directory to the `scripts` directory, run the following command:

```
cd scripts
```

Now use the `pwd` command to verify whether your current working directory has changed as expected:

```
pwd
```

You can enter `cd` without any directory name to move back to your home directory:

```
cd
```

Then, enter the `pwd` command to verify whether your current working directory has changed:

```
pwd
```

The syntax `..` is a shortcut that refers to the parent directory of your current directory. Run the following command to change dictories up one level:

```
cd ..
```

2.3. Create an empty file

`touch`

First, return to your home directory by entering:

```
cd
```

Next, use the `touch` command to create an empty file named `myfile.txt`:

```
touch myfile.txt
```

Now use the `ls` command to verify the creation of `myfile.txt`:

```
ls
```

If the file already exists, the `touch` command updates the access timestamp, or last-modified date of the file. To see this, enter:

```
touch myfile.txt
```

And use use the `date` command to verify the date change:

```
date -r myfile.txt
```

Exercise 3 - Managing Files and Directories

3.1. Search for and locate files

`find`

The `find` command is used to search for files in a directory. You can search for files based on different attributes, such as the file's name, type, owner, size, or timestamp.

The `find` command conducts a search of the entire directory tree starting from the given directory name.

For example, the following command finds all `.txt` files in the `/etc` directory and all of its subdirectories:

```
find /etc -name \'*.txt\'
```

> ***Note:*** *Along with listing all the `.txt` files, the terminal may return "Permission denied" errors.*
> *These errors are normal, as you have limited access permissions on the lab machine.*

3.2. Remove files

`rm`

The `rm` command is used to delete files, ideally with the `-i` option, which creates a prompt to ask for confirmation before every deletion.

To remove the file `myfile.txt`, enter the following command and press `y` to confirm deletion, or `n` to deny deletion:

```
rm -i myfile.txt
```

Use the `ls` command to verify removal:

```
ls
```

> ***Tip:*** *When you are only removing one file with the `rm` command, the `-i` option is redundant. But if you want to remove multiple files, for example by using a wildcard to find all filenames matching a pattern, it's best practice to confirm or deny each deletion by including the `-i` option.*

> *Be careful when deleting files or directories! There is normally no way to restore a deleted file once it is deleted, as there is no trash folder. This is why you should always back up, or archive, your important files. You will learn more about archiving files soon.*

3.3. Move and rename a file

`mv`

You can use the `mv` command to move files from one directory to another and/or rename them.

Before doing so, let's first create a new file called `users.txt`:

```
touch users.txt
```

You should always use caution when moving a file. If the target file already exists, it will be overwritten, or replaced, by the source file.

Conveniently, however, when the source and target directories are the same, you can use `mv` to rename a file.

To illustrate this, use `mv` to rename `users.txt` to `user-info.txt` by entering the following command:

```
mv users.txt user-info.txt
```

Because the source and target directories are the same (your present working directory), the `mv` command will rename the file.

Now use the `ls` command to verify the name change:

```
ls
```

Now, you can move `user-info.txt` to the `/tmp` directory as follows:

```
mv user-info.txt /tmp
```

Use the `ls` command twice to verify the move:

```
ls
```

```
ls -l /tmp
```

3.4. Copy files

`cp`

You can use the `cp` command to copy `user-info.txt`, which is now in your `/tmp` directory, to your current working directory:

```
cp /tmp/user-info.txt user-info.txt
```

Use the `ls` command to verify that the copy was successful:

```
ls
```

At times, you may want to copy the contents of an existing file into a new one.

The following command copies the content of `/etc/passwd` to a file named `users.txt` within the current directory:

```
cp /etc/passwd users.txt
```

Again, use the `ls` command to verify if the copy was successful:

```
ls
```

Summary

In this lab, you learned that you can use the commands:

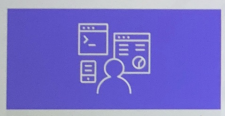
- `pwd` to get the location of your present working directory
- `ls` to list the files and directories within a directory
- `mkdir` to create a new directory
- `cd` to change your present working directory
- `touch` to create a new file
- `find` to search for and locate files
- `rm` to remove a file
- `mv` to rename or move a file
- `cp` to copy a file

Working with text Files, Networking & Archiving Commands

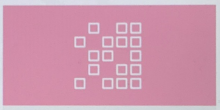
Viewing File Content

What you will learn

In this video, you will learn how to:



View the contents of a file in useful ways



Determine line, word, and character counts

commands

cat : showing entire lines

Viewing your file all at once

`cat` (catenate) – Print entire file contents

```
$ ls
numbers.txt
$ cat numbers.txt
```

print ↑ this

```
89
90
91
92
93
94
95
96
97
98
99
```

all contents have shown

more : showing by page format

Viewing your file page-by-page

`more` – Print file contents page-by-page

```
$ more numbers.txt
```

```
0
1
2
3
4
5
6
7
8
```

we can view more by expand the terminal window it'll increase the page size

Viewing your file page-by-page

`more` – Print file contents page-by-page

```
$ more numbers.txt
```

so to next page

space bar

```
9
10
11
12
13
14
15
16
17
```

Viewing your file page-by-page

`more` – Print file contents page-by-page

```
$ more numbers.txt
```

q + enter

```
$
```

↑ to quit to command prompt

head : start view from begin line

Viewing the first 10 lines

`head` – Print first 10 lines of file

```
$ head numbers.txt
0
1
2
3
4
5
6
7
8
9
```

return first ten lines

Viewing the first N lines

`head` – Print first N lines of file

```
$ head -n 3 numbers.txt
0
1
2
```

number of lines

we can specify number of lines

tail : start to view from last line

Viewing the last 10 lines

`tail` – Print last 10 lines of file

```
$ tail numbers.txt
90
91
92
93
94
95
96
97
98
99
```

Viewing the last N lines

`tail` – Print last N lines of file

```
$ tail -n 3 numbers.txt
97
98
99
```

wc : count number of characters, words and lines

Counting lines, words, and characters

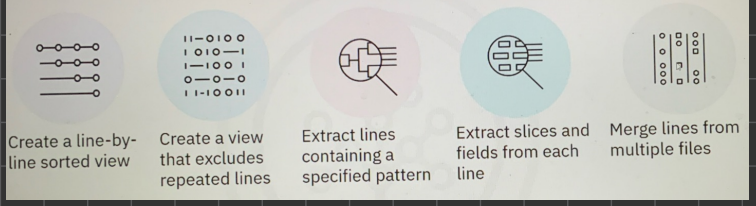
`wc` (word count) – Count characters, words, lines

```
$ cat pets.txt
cat
cat
cat
cat
dog
dog
cat
```

```
$ wc pets.txt
7  7  28 pets.txt   → show all numbers
$ wc -l pets.txt
7  pets.txt         → show number of lines
$ wc -w pets.txt
7  pets.txt         → — n — of words
$ wc -c pets.txt    → ~ n ~ of character
28 pets.txt
```

useful commands for wrangling Text File

What you will learn



Create a line-by-line sorted view

Create a view that excludes repeated lines

Extract lines containing a specified pattern

Extract slices and fields from each line

Merge lines from multiple files

sort

Sorting your views line-by-line

`sort` - Sort lines in a file → sorts the lines of a file alpha-numerically

```
$ sort pets.txt
cat
cat
cat
cat
cat
dog
dog
```

'C' come before 'd'

```
$ sort -r pets.txt
dog
dog
cat
cat
cat
cat
cat
```

reverse sort

uniq

Excluding repeated lines from views

`uniq` ("unique") – Filter out repeated lines

↳ it removes duplicated lines if it consecutive, thus ↴

```
$ cat pets.txt
cat
cat
cat
cat
dog
dog
cat
```

```
$ uniq pets.txt
cat
dog
cat
```

→ interrupt the consecutive sequence so 'cat' appears two times.

I think sort it first to distinct it

grep

Extracting lines matching a pattern

`grep` ("global regular expression print") – Return lines in file matching pattern

```
$ cat people.txt
Alan Turing
Bjarne Stroustrup
Charles Babbage
Dennis Ritchie
Erwin Schrodinger
Fred Hoyle
Guido Rossum
Henri Poincare
Ivan Pavlov
John Neumann
Ken Thompson
```

find 'ch' in

```
$ grep ch people.txt
Dennis Ritchie
Erwin Schrodinger
```

contain 'ch' that matched

return all 'ch' can be

```
$ grep -i ch people.txt
Charles Babbage
Dennis Ritchie
Erwin Schrodinger
```

expands the pattern

search by making it case-insensitive

cut

Extracting slices from lines

`cut` – Extracts a section from each line

start index, last index

```
$ cat people.txt
Alan Turing
Bjarne Stroustrup
Charles Babbage
Dennis Ritchie
Erwin Schrodinger
Fred Hoyle
Guido Rossum
Henri Poincare
Ivan Pavlov
John Neumann
Ken Thompson
```

```
$ cut -c 2-9 people.txt
lan Turi
jarne St
harles B
ennis Ri
rwin Sch
red Hoyl
uido Ros
enri Poi
van Pavl
ohn Neum
en Thomp
```

Extracting fields from lines

`cut` – Extract a field from each line

specify the delimiter this Ex. is space

```
$ cat people.txt
Alan Turing
Bjarne Stroustrup
Charles Babbage
Dennis Ritchie
Erwin Schrodinger
Fred Hoyle
Guido Rossum
Henri Poincare
Ivan Pavlov
John Neumann
Ken Thompson
```

```
$ cut -d ' ' -f2 people.txt
Turing
Stroustrup
Babbage
Ritchie
Schrodinger
Hoyle
Rossum
Poincare
Pavlov
Neumann
Thompson
```

specify to return the second field

Merging by paste

Merging lines from multiple files

`paste` – Merge lines from different files

```
$ cat first.txt
Alan
Bjarne
Charles
Dennis
Erwin
Fred
Guido
Henri
Ivan
John
Ken
```

```
$ cat last.txt
Turing
Stroustrup
Babbage
RSchrodinger
Hoyle
itchie
Rossum
Poincare
Pavlov
Neumann
Thompson
```

```
$ cat yob.txt
1912
1950
1791
1941
1887
1915
1956
1854
1849
1903
1943
```

→ want to merge these three files

```
$ paste first.txt last.txt yob.txt
Alan    Turing       1912
Bjarne  Stroustrup   1950
Charles Babbage      1791
Dennis  Ritchie      1941
Erwin   Schrodinger  1887
Fred    Hoyle        1915
Guido   Rossum       1956
Henri   Poincare     1854
Ivan    Pavlov       1849
John    Neumann      1903
Ken     Thompson     1943
```


Merging lines from multiple files

`paste` – Merge lines from different files


```
$ paste -d "," first.txt last.txt yob.txt
Alan,Turing,1912
Bjarne,Stroustrup,1950
Charles,Babbage,1791
Dennis,Ritchie,1941
Erwin,Schrodinger,1887
Fred,Hoyle,1915
Guido,Rossum,1956
Henri,Poincare,1854
Ivan,Pavlov,1849
John,Neumann,1903
Ken,Thompson,1943
```

specify delimiter

LAB

Learning Objectives

Working through the exercises in this lab will enable you to perform some basic but essential text wrangling operations. These operations will allow you to work with text files by:

- Displaying and exploring file contents
- Extracting and displaying first or last N lines of text
- Displaying counts of lines, words, and characters in text
- Sorting lines (rows) of text
- Dropping consecutively duplicated lines of text
- Extracting lines of text containing a pattern match
- Extracting fields from lines of text
- Merging text files as aligned columns of text

These are some of the building blocks of filtering text files. Later in this course, you will learn how to combine such operations. This will empower you to start engineering sophisticated views of your data by creating complex data-processing flows called *data pipelines*.

cat , more , less

1.2. Viewing file content with the `more` command

A better alternative to the `cat` command for viewing file contents is the `more` command. By entering the following command:

```
more entrypoint.sh
```

you will see the top portion of the file first.

> ***Tip:*** *The first line of this particular file,* `#!/bin/bash` *, is called a shebang. Basically, this shebang line makes the file a bash script by invoking the bash shell. You will learn more about shebang lines later in this course.*

When using the `more` command, you can see only as many lines as will fit on your terminal window at once.

To see the next portion of the file, just press your spacebar. You can keep *paging* this way, tapping the spacebar until you reach the end of the file. Once you reach the last *page*, you will exit back to the command prompt.

Another way to exit is simply to type `q`, which quits and returns to the command prompt.

1.3. Scrolling through file content with the `less` command

What if you want to move up and down through the file, not just downward? In this case, you can use the `less` command:

```
less entrypoint.sh
```

Just like `more`, the `less` command displays the first page of the file. What's useful about `less` is that you can use it to move around the file, page by page, using the `Page Up` and `Page Down` keys.

You can also scroll up and down through the file line-by-line, using the `Up Arrow` and `Down Arrow` keys, ↑ and ↓.

Unlike `more`, `less` does not automatically exit when you reach the end of a file, allowing you the option to continue scrolling around. You can quit at any time by typing `q`.

'less' command shows first page, if we would like to continue to see the next contents we can scroll up/down to see line by line in file

head , tail

2.1. Display the first N lines of a file

`head`

By default, `head` will print the first 10 lines of a file. To use it with `usdoi.txt`, enter the following:

```
head usdoi.txt
```

this file comes from the previous instructor

You can also specify the number of lines to be printed. Print only the first *3* lines of text from the file `usdoi.txt` by entering:

```
head -3 usdoi.txt
```

↳ no need -n just put minus before numerical

2.2. Display the last N lines of a file

`tail`

By default, `tail` will print the last 10 lines of the file `usdoi.txt`:

```
tail usdoi.txt
```

Just like with `head`, you can specify the number of lines to be printed. Print the last *2* lines of the file `usdoi.txt` by entering the following:

```
tail -2 usdoi.txt
```

```
theia@theia-tanaphatsa2:/home/project$ head -3 usdoi.txt
The unanimous Declaration of the thirteen united States of America, When in the
Course of human events, it becomes necessary for one people to dissolve the
political bands which have connected them with another, and to assume among the
theia@theia-tanaphatsa2:/home/project$ head -n 3 usdoi.txt
The unanimous Declaration of the thirteen united States of America, When in the
Course of human events, it becomes necessary for one people to dissolve the
political bands which have connected them with another, and to assume among the
```

Equal

wc

3.1. Count lines, words, or characters from a text file

`wc`

If you want to find the number of lines, words, and characters in a file like `usdoi.txt`, enter the following command:

```
wc usdoi.txt
```

The output contains the number of lines, followed by the number of words, followed by the number of characters in the file.

To get just the count of lines in `usdoi.txt`, use the `-l` option:

```
wc -l usdoi.txt
```

Similarly, for the count of words in `usdoi.txt`, use the `-w` option:

```
wc -w usdoi.txt
```

To print the number of characters in `usdoi.txt`, use the `-c` option:

```
wc -c usdoi.txt
```

sort , uniq

Exercise 4 - Basic text wrangling: sorting lines and dropping duplicates

4.1. Sort and display lines of file alphanumerically

`sort`

You can use the `sort` command to display the lines of a file sorted alphanumerically.

To view the lines of `usdoi.txt` sorted alphanumerically, enter:

```
sort usdoi.txt
```

To view those lines sorted in reverse order, enter:

```
sort -r usdoi.txt
```

4.2. Drop consecutive duplicated lines and display result

`uniq`

First download the following file:

```
wget https://cf-courses-data.s3.us.cloud-object-stor
```

View the raw contents of `zoo.txt` with the `cat` command:

```
cat zoo.txt
```

View the contents of `zoo.txt` with identical, consecutive lines dropped using the `uniq` command:

```
uniq zoo.txt
```

The `uniq` line will drop any lines in the file that are identical *and* consecutive. This is similar to what is known as "dropping duplicates". As you can see from this example, however, there can still be duplicated lines left over if these lines are not repeated right after the other.

grep, cut

5.1. Extract lines matching a specified criterion

`grep`

The `grep` command allows you to specify a pattern and search for lines within a file that match that pattern.

For example, the following command prints all lines in the file `usdoi.txt` which contain the word `people`:

```
grep people usdoi.txt
```

Some frequently used options for `grep` include:

Option	Description
`-n`	Along with the matching lines, also print the line numbers
`-c`	Get the count of matching lines
`-i`	Ignore the case of the text while matching
`-v`	Print all lines which do not contain the pattern
`-w`	Match only if the pattern matches whole words

You can use these options to print all the lines from the `/etc/passwd` file which do not contain the pattern `login`:

```
grep -v login /etc/passwd
```

```
theia@theia-tanaphatsa2:/home/project$ grep people usdoi.txt
Course of human events, it becomes necessary for one people to dissolve the
people, unless those people would relinquish the right of Representation in the
firmness his invasions on the rights of the people.
to harrass our people, and eat out their substance.
the lives of our people.
Tyrant, is unfit to be the ruler of a free people.
```

5.2. Extract fields from lines of text

`cut`

The `cut` command allows you to view only specific fields from each line of text in a file.

For example, you can use `cut` with the `-c` option to view only the *first two* characters of each line:

```
cut -c -2 zoo.txt
```

Or to view each line *starting from* the second character:

```
cut -c 2- zoo.txt
```

The `cut` command can also be used to extract a field from a delimited file.
To demonstrate this, start by downloading and taking a look at the following comma-separated file:

```
wget https://cf-courses-data.s3.us.cloud-object-storage.a
cat names_and_numbers.csv
```

Now you can extract just the phone numbers for each person listed in the file using the `-d` (delimiter) and `f` (field) options as follows:

```
cut -d "," -f2 names_and_numbers.csv
```

`-d ","` tells the command that the delimiter is a comma, and `-f2` tells it to extract the second field.

```
theia@theia-tanaphatsa2:/home/project$ cat names_and_numbers.csv
John Fogerty, 555-1212
Jane Doe, 555-1312
```

all lines

```
theia@theia-tanaphatsa2:/home/project$ cut -d ',' -f2 names_and_numbers.csv
 555-1212
 555-1312
```

after cut

paste

Exercise 6 - Basic text wrangling: merging lines as fields

6.1. Merge text files line-by-line, aligned as columns

`paste`

Use the `paste` command to merge lines of multiple files together.

Download the following file:

```
wget https://cf-courses-data.s3.us.cloud-object-stor
```

Then use the `paste` command to view the two files merged together, line-by-line, as columns delimited by a `Tab` character:

```
paste zoo.txt zoo_ages.txt
```

Try changing the delimiter. Instead of the default `Tab` delimiter, you can specify a comma `,` as follows:

```
paste -d "," zoo.txt zoo_ages.txt
```

```
anaconda        3
zebra   4
zebra   1
lion    0
zebra   1
theia@theia-tanaphatsa2:/home/project$ paste -d ',' zoo.txt  zoo_ages.txt
zebra,17
zebra,12
lion,7
lion,4
anaconda,3
zebra,4
```

Introduction to Networking

Computer Network

- Computer Network: is a set of computers that are able to communicate by Network nodes
- resource: any object, such as a file or document
- network node: device that participates in a network

URLs and IP Address

- IP: • Defines the format of data transmitted over the internet
- IP Address: • Identify any host on internet
- URL (Uniform Resource Locator): • Web address

  Ex: https://en.wikipedia.org/wiki/URL

  • https: protocols

  • en.wikipedia.org: host name

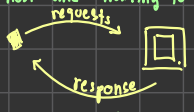
  • /wiki/URL: file name

Packets and Ping

Packets: Header | payload → message

It is the 'control information' that is data about 'how' and 'where' (IP) to deliver payload

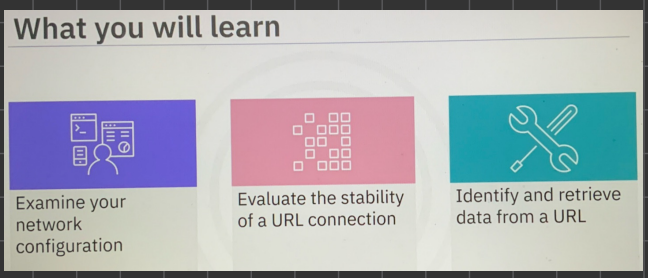
Ping: • It is a command work by sending special 'echo requests' packet to host and waiting to response



• used to test and trouble shoot connectivity to other network

Networking Commands

What you will learn



Getting your machine's host name

- `hostname` – Print host name (get or set; and other information)

```
$ hostname                  ← return hostname of this machine
my-linux-machine.local      ← local domains
$ hostname -s               → to drop
my-linux-machine
$ hostname -i
127.0.1.1                   ← show IP
```

Getting network information

`ifconfig` (Interface configuration) – Display or configure the system network interfaces → show all information in device

```
$ ifconfig with no options → show all
lo0: flags=8049<UP,LOOPBACK,RUNNING,MULTICAST> mtu 16384
  options=1203<RXCSUM,TXCSUM,TXSTATUS,SW_TIMESTAMP>
  inet 127.0.0.1 netmask 0xff000000
  inet6 ::1 prefixlen 128
  inet6 fe80::1%lo0 prefixlen 64 scopeid 0x1
  nd6 options=201<PERFORMNUD,DAD>
gif0: flags=8010<POINTOPOINT,MULTICAST> mtu 1280
…
```

Getting ethernet adapter info

`ifconfig` (Interface configuration) – Display or configure system network interfaces

```
$ ifconfig eth0                 ← specify device to inspect
eth0  Link encap:Ethernet  HWaddr 00:0B:CD:1C:18:5A   ← it is a ethernet adapter
  inet addr:172.16.25.126  Bcast:172.16.25.63  Mask:255.255.255.224
  inet6 addr: fe80::20b:cdff:fe1c:185a/64 Scope:Link
  UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
  RX packets:234558 errors:0 dropped:0 overruns:0 frame:0
  TX packets:2221421 errors:0 dropped:0 overruns:0 carrier:0
  collisions:0 txqueuelen:1000
  RX bytes:293912265 (280.2 MiB)  TX bytes:1044100408 (995.7 MiB)
  Interrupt:185 Memory:f7fe0000-f7ff0000
```

internet address

no. of packets received

total amount of data received and transmitted

Testing server connections

`ping` – Send ICMP packets to URL and print response (test connectivity)

```
$ ping www.google.com
PING www.google.com (142.251.41.68): 56 data bytes
64 bytes from 142.251.41.68: icmp_seq=0 ttl=119 time=21.750 ms
64 bytes from 142.251.41.68: icmp_seq=1 ttl=119 time=20.712 ms
64 bytes from 142.251.41.68: icmp_seq=2 ttl=119 time=24.065 ms   ← total round trip time
64 bytes from 142.251.41.68: icmp_seq=3 ttl=119 time=36.751 ms
64 bytes from 142.251.41.68: icmp_seq=4 ttl=119 time=41.774 ms
^C                                                  ← finish until control+C
--- www.google.com ping statistics ---
5 packets transmitted, 5 packets received, 0.0% packet loss   ← % of packet dropped; static info.
round-trip min/avg/max/stddev = 20.712/229.010/41.774/9.603 ms   ← minimum, avg, maximum, standard deviation in ms
```

Testing server connections

`ping` – Send ICMP packets to URL and print response

```
$ ping -c 5 www.google.com
PING www.google.com (142.251.41.68): 56 data bytes
64 bytes from 142.251.41.68: icmp_seq=0 ttl=119 time=17.491 ms
64 bytes from 142.251.41.68: icmp_seq=1 ttl=119 time=19.784 ms
64 bytes from 142.251.41.68: icmp_seq=2 ttl=119 time=24.279 ms
64 bytes from 142.251.41.68: icmp_seq=3 ttl=119 time=24.964 ms
64 bytes from 142.251.41.68: icmp_seq=4 ttl=119 time=26.106 ms

--- www.google.com ping statistics ---
5 packets transmitted, 5 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 17.491/22.525/26.106/3.308 ms
```

Web scraping with curl

`curl` (Client URL) – Transfer data to and from URL(s)

```
$ curl www.google.com
<!doctype html><html itemscope="" itemtype="http://schema.org/WebPage"
lang="en-CA"><head><meta content="text/html; charset=UTF-8" http-
equiv="Content-Type"><meta
content="/images/branding/googleg/1x/googleg_standard_color_128dp.png"
itemprop="image"><title>Google</title><script
nonce="gPa6M7RHuxLHFwYnP5CH4A==">(function(){window.google={kEI:'FdCKYdj-
LrOt0PEPuoqLIA',kEXPI:'0,18168,1284368,56873,1709,4350,206,4804,2316,383,
246,5,1354,5250,1122516,1197719,329548,51224,16114,17444,11240,17572,4859
,1361,9291,3027,2816,1931,12834,4020,978,13228,516,3331,4192,6430,7432,14
390,919,5081,887,706,1279,2212,530,149,1103,840,1983,213,4101,3514,606,20
...
```

Scraping a web page's HTML to file

`curl` (Client URL) – Transfer data to and from URL(s)

```
$ curl www.google.com -o google.txt
$ head -n 1 google.txt
<!doctype html><html itemscope=""
itemtype="http://schema.org/WebPage" lang="en-CA"><head><meta
content="text/html; charset=UTF-8" http-equiv="Content-
Type"><meta content="/images/branding/googleg/1x/googleg_standard_color_1
28dp.png" itemprop="image"><title>Google</title><script nonce="gPa6M7RHux
LHFwYnP5CH4A==">(function(){window.google={kEI:'FdCKYdj-
LrOt0PEPuoqLIA',kEXPI:'0,18168,1284368,56873,1709,4350,206,4804,2316,383,
246,5,1354,5250,1122516,1197719,329548,51224,16114,17444,11240,17572,4859
...
```

Downloading files from a URL

`wget` (Web get) – Download file(s) from a URL

- more focused than `curl`, supports recursive file downloads

```
$ wget https://www.w3.org/TR/PNG/iso_8859-1.txt
Resolving www.w3.org (www.w3.org)... 128.30.52.100
Connecting to www.w3.org (www.w3.org)|128.30.52.100|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 6121 (6.0K) [text/plain]
Saving to: 'iso_8859-1.txt'

iso_8859-
1.txt                   100%[===================================>]
  5.98K  --.-KB/s    in 0s
```

Downloading files from a URL

https://www.w3.org/TR/PNG/iso_8859-1.txt

The following are the graphical (non-control) characters defined by ISO 8859-1 (1987). Descriptions in words aren't all that helpful, but they're the best we can do in text. A graphics file illustrating the character set should be available from the same archive as this file.

Hex	Description	Hex	Description
20	SPACE		
21	EXCLAMATION MARK	A1	INVERTED EXCLAMATION MARK
22	QUOTATION MARK	A2	CENT SIGN
23	NUMBER SIGN	A3	POUND SIGN
24	DOLLAR SIGN	A4	CURRENCY SIGN
25	PERCENT SIGN	A5	YEN SIGN
26	AMPERSAND	A6	BROKEN BAR
27	APOSTROPHE	A7	SECTION SIGN
28	LEFT PARENTHESIS	A8	DIAERESIS
29	RIGHT PARENTHESIS	A9	COPYRIGHT SIGN
2A	ASTERISK	AA	FEMININE ORDINAL INDICATOR

Downloading files from a URL

```
$ head -n 12 iso_8859-1.txt
The following are the graphical (non-control) characters defined by
ISO 8859-1 (1987).  Descriptions in words aren't all that helpful,
but they're the best we can do in text.  A graphics file illustrating
the character set should be available from the same archive as this
file.

Hex Description                 Hex Description

20  SPACE
21  EXCLAMATION MARK            A1  INVERTED EXCLAMATION MARK
22  QUOTATION MARK              A2  CENT SIGN
23  NUMBER SIGN                 A3  POUND SIGN
```

LAB

Exercise 1 - View configuration info about your network

1.1. Display your system's hostname and IP address

`hostname`

A **hostname** is a name that is assigned to a computer or device on a network, and it is used to identify and communicate with that device.

To view the current hostname, run the command below:

```
1  hostname
```

An **IP address** (Internet Protocol address) is a numerical label assigned to each device connected to a computer network that uses the Internet Protocol for communication.

You can use the `-i` option to view the IP address of the host:

```
1  hostname -i
```

1.2. Display network interface configuration

`ifconfig`

The `ifconfig` command is used to configure or display network interface parameters for a network.

To display the configuration of all network interfaces of your system, enter:

```
1  ifconfig
```

To display the configuration of a particular device, such as the ethernet adapter `eth0`, enter:

```
1  ifconfig eth0
```

`eth0` is usually the primary network interface that connects your server to the network.

You can see your server's IP address in line 2 after the word `inet`.

Exercise 2 - Test network connectivity

2.1. Test connectivity to a host

`ping`

Use the `ping` command to check if `www.google.com` is reachable. The command keeps pinging data packets to server at `www.google.com` and prints the response it gets back. (Press `Ctrl` + `c` to stop pinging.)

```
1  ping www.google.com
```

If you want to ping only a limited number of times, use `-c` option.

```
1  ping -c 5 www.google.com
```

Exercise 3 - View or download data from a server

3.1. Transfer data from a server

`curl`

You can use `curl` to access the file at the following URL and display the file's contents on your screen:

```
1  curl https://cf-courses-data.s3.us.cloud-object-stor
```

To access the file at the given URL and also save it in your current working directory, use the `-O` option:

```
1  curl -O https://cf-courses-data.s3.us.cloud-object-s
```

You can also use `curl` to view the HTML code for any web page if you know its URL.

3.2. Download file(s) from a URL

`wget`

The `wget` command is similar to `curl`, however its primary use is for file downloading. One unique feature of `wget` is that it can recursively download files at a URL.

To see `wget` in action, first remove `usdoi.txt` from your current directory:

```
1  rm usdoi.txt
```

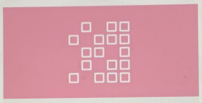
then download it again using `wget` as follows:

```
1  wget https://cf-courses-data.s3.us.cloud-object-stor
```

File Archiving and Compression Commands

What you will learn

Distinguish file archiving from file compression



Create archived files and unpack them

Apply commands to compress, decompress, and extract files from archives

Archiving and compression

Archives:

- store rarely used information and preserve it
- are a collection of data files and directories stored as a single file
- make the collection more portable and serve as a backup in case of loss or corruption

Archiving and compression

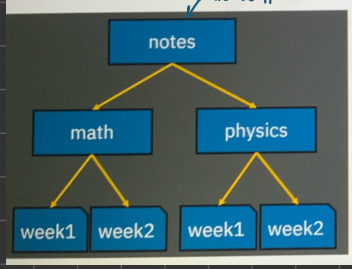
File compression:

- reduces file size by reducing information redundancy
- preserves storage space, speeds up data transfer, and reduces bandwidth load

Directory tree archiving

Notes directory tree example

archive it

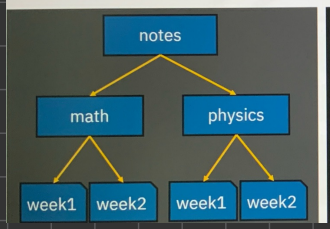



```
$ ls -r
.:
notes
./notes:
math  physics
./notes/math:
week1  week2
./notes/physics:
week1  week2
```

File archiving and compression

tar (tape archiver) – Archive and extract files

↗ to archive or dearchive files




```
$ tar -cf notes.tar notes
$ ls
notes notes.tar
$ tar -czf notes.tar.gz notes
$ ls
notes notes.tar notes.tar.gz
```

file or directory that we need to archive

name to archive

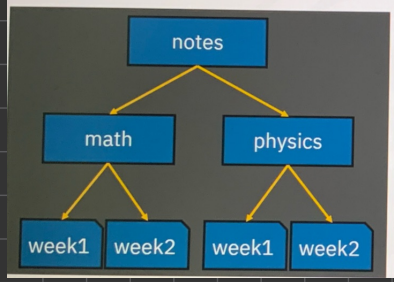
to compress the archived file

↳ this minus z is filter archive

file though compression programe call g-zip (.gz)

Checking your archive contents

tar – List archive contents

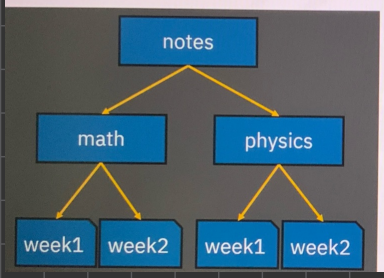



```
$ tar -tf notes.tar
notes/
notes/math/
notes/physics/
notes/physics/week1
notes/physics/week2
notes/math/week1
notes/math/week2
```

Extracting archived files

tar – Extract files and folders

to de archive or unpack



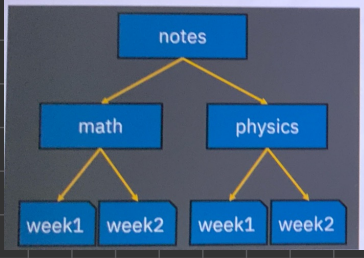

```
$ tar -xf notes.tar notes
$ ls -R
.:
notes  notes.tar
./notes:
math  physics
./notes/math:
week1  week2
./notes/physics:
week1  week
```

tell tar to extract

has already de archived

Decompressing and extracting archives

tar – Decompress and extract




```
$ tar -xzf notes.tar.gz notes
$ ls -R
.:
notes  notes.tar
./notes:
math  physics
./notes/math:
week1  week2
./notes/physics:
week1  week2
```

File compression and archiving

zip – Compress files and directories and package into a single archive to an archive

zip:

compress → bundle

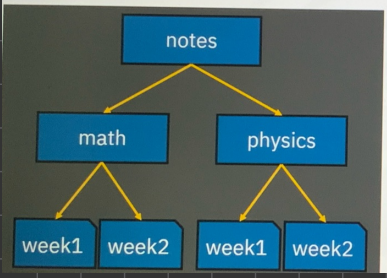
tar:

bundle → compress

```
$ zip -r notes.zip notes
$ ls
notes notes.zip
```

Extracting and decompressing archives

`unzip` – Extract and decompress zipped archive




```
$ unzip notes.zip
$ ls -R
.:
notes  notes.tar
./notes:
math  physics
./notes/math:
week1  week2
./notes/physics:
week1  week2
```

LAB

Exercise 1 - File and folder archiving and compression

1.1. Create and manage file archives

`tar`

The `tar` command allows you to pack multiple files and directories into a single archive file.

The following command creates an archive of the entire `/bin` directory and writes the archive to a single file named `bin.tar`.

The options used are as follows:

Option	Description
-c	Create new archive file
-v	Verbosely list files processed
-f	Archive file name

```
1  tar -cvf bin.tar /bin
```

To see the list of files in the archive, use the `-t` option:

```
1  tar -tvf bin.tar
```

1.2. Package and compress archive files

`zip`

The `zip` command allows you to compress files.

The following command creates a zip file named `config.zip` consisting of all the files with extension `.conf` in the `/etc` directory.

```
1  zip config.zip /etc/*.conf
```

The `-r` option can be used to zip an entire directory.

The following command creates an archive of the `/bin` directory.

```
1  zip -r bin.zip /bin
```

1.3. Extract, list, or test compressed files in a ZIP archive

`unzip`

The `unzip` command allows you to extract files.

To list the files of the archive `config.zip`, enter the following:

```
1  unzip -l config.zip
```

The following command extracts all the files in the archive `bin.zip`.

```
1  unzip -o bin.zip
```

We added the `-o` option to force overwrite in case you run the command more than once.

You should see a folder named `bin` created in your directory.