CS232 - FOSS LAB REPORT

12-05-2019

Nikhil M Tomy

CHN17CS080

S4-D 36

Contents

1 Linux Commands

1.1 ls

`ls` is used to list information about the files (the current working directory by default). Sorts entries alphabetically if none of `-cftuvSUX` nor `--sort` is specified.

`ls`	List file in current directory
`ls -l`	Long list files in current directory

1.2 alias

The `alias` utility shall create or redefine `alias` definitions or write the values of existing `alias` definitions to standard output.An `alias` definition provides a string value that shall replace a command name when it is encountered.
The `unalias` utility shall remove the definition for each alias name specified.

`alias ll='ls -al'`	Define `ll` as `ls -al`
`unalias ll`	Remove the definition of `ll`

1.3 cat

`cat` is used to concatenate files and prints on the `stdout`. With no file or when file is `-`, it reads from `stdin`

`cat <filename>`	Prints the content of `filename` into `stdout`

1.4 sort

`sort` is used to write sorted concatenation of all files to `stdout`. With no file or when file is `-`, it reads from `stdin`

`sort <filename>`	Prints the sorted output of `filename` into `stdout`

1.5 cut

`cut` is used to print selected parts of lines from each file to `stdout`.

`cut -f3 -d' '` Prints the third column of space separated line into `stdout`

1.6 paste

`paste` is used to write lines consisting of the sequentially corresponding lines from each file separated by TABs to `stdout`. With no file or when file is `-`, it reads from `stdin`

```
paste <file1> <file2>
```

1.7 join

`join` is used to write a line for each pair of lines with identical join fields `stdout`. The default join field is the first, delimited by blanks. When file1 or file2 (not both) is `-`, it reads from `stdin`

```
join <file1> <file2>
```

1.8 mkdir

`mkdir` is used to create a new directory.

`mkdir <folder>` Creates a new `<folder>`

1.9 ps

`ps` is used for process management.

`ps aux` Display currently running processes

1.10 kill

`kill` is used to terminate process by pid

`kill <pid>` terminates the process with the given pid

1.11 useradd

`useradd` is used to create a new user.

`useradd <username>` Creates a new user

1.12 mount

`mount` is used to mount a device into the filesystem

`mount <devce> <directory>` Mounts the `<device>` on `<directory>`

1.13 umount

`mount` is used to unmount a device

`umount <devce>` Unounts the `<device>`

2 CGPA Computation

Computing CGPA requires to download the PDFs of the results and the file containing register number and names of students in our class. Then clean the PDFs to create a file containing register number and marks of all courses of students, one per line. Then the file in piped through a `C` program to yeild a file containg register number and CGPA of students, one per line. This file is joined with the file containing the register number and name of students to yeild the final file containing register number, name and CGPA of each students.

Shellscript
https://raw.githubusercontent.com/ceccs17d36/cs232/master/2.task_gpa/task_gpa.sh

C Program
https://raw.githubusercontent.com/ceccs17d36/cs232/master/2.task_gpa/compute_gpa/compute_gpa.c

3 Networking

3.1 ifconfig

`ifconfig` is used to configure the kernel-resident network interfaces. It is used at boot time to set up interfaces as necessary. After thet it is usually only needed when debuggin or when system tuning is needed.
Display details about all interfaces

```
ifconfig -a
```

Display details about ¡interface¿

```
ifconfig <interface>
```

Set ip address for ¡interface¿

```
ifconfig <interface> <ip>
```

3.2 route

`route` manipulates the kernel's IP routing tables. Its primary use is to set up static routes to specifi c hosts or networks via an interface after it has been configured with the ifconfig program.
Display details about current routing table

```
route
```

Add a default gateway address

```
route add default gw <ip>
```

3.3 dhclient

`iwconfig` is similar to ifconfig, but is dedicated to the wireless interfaces. It is used to set the parameters of the network interfaces which are specific to the wireless operation. `iwconfig` may also be used to display those parameters, and the wireless statistics.
Display details all wireless interfaces

```
iwconfig
```

4 FTP Usage and Commands

FTP (File Transfer Protocol) is used to transfer files between two remote systems. It is network protocol similar to HTTP, but for file transfer. `ftp` clients are used to connect to `ftp` servers. After connection tools provided by FTP can be used for file transfer.

4.1 Usage

Connect to ¡server¿

```
ftp <server>
```

Download file from server

```
get <file>
```

Download multiple files from server

```
mget <directory>
```

Upload file to server

```
put <file>
```

Upload multiple files to server

```
mput <server>
```

Change remote working directory

```
cd <directory>
```

Change local working directory

```
lcd <directory>
```

Change file permissions of remote file

```
chmod [options] [files]
```

Delete remote file

```
delete <filename>
```

Delete multiple remote files

```
mdelete <files>
```

Remove directory on remote server

```
rmdir <directory>
```

Exit ftp session

```
exit
```

5 SSH

```
ssh [-46AaCfGgKkMNnqsTtVvXxYy] [-B bind_interface]
     [-b bind_address] [-c cipher_spec] [-D [bind_address:]port]
     [-E log_file] [-e escape_char] [-F configfile] [-I pkcs11]
     [-i identity_file] [-J destination] [-L address] [-l login_name]
     [-m mac_spec] [-O ctl_cmd] [-o option] [-p port] [-Q query_option]
     [-R address] [-S ctl_path] [-W host:port] [-w local_tun[:remote_tun]
     destination [command]
```

ssh (SSH client) is a program for logging into a remote machine and for executing commands on a remote machine. It is intended to provide secure encrypted communications between two untrusted hosts over an insecure network. X11 connection, arbitrary TCP ports and UNIX-domain sockets can also be forwarded over the secure channel.
ssh connects and logs into the specified destination, which may be specified as either [user@]hostname or a URI of the form ssh://[user@]hostname[:por
The user must prove his/her identity to the remote machine using one of the sevaral methods.

Connect to a remote server

```
ssh user@remoteip
```

5.1 Steps

Connect to a 14.139.189.217 as cs17d36

```
$ ssh cs17d36@14.139.189.217
```

List remote directory

```
$ ls
```

Change directory to cs232

```
$ cd cs232
```

List contents of remote directory

```
$ ls
```

Change directory to parent directory

```
cd ..
```

Create empty file named samplefile

```
touch samplefile
```

Remove file named samplefile

```
rm samplefile
```

Create empty directory named sampledirectory

```
mkdir sampledirectory
```

Remove directory named `sampledirectory`

```
rm -r sampledirectory
```

5.2 Result

```
$ ssh cs17d36@14.139.189.217
The authenticity of host '14.139.189.217
   ↪ (14.139.189.217)' can't be established.
ECDSA key fingerprint is SHA256:
   ↪ pkLWQEXAwg4Zdh5CXNYydUTPcU7dwDFxyzJnqJ01m/s
   ↪ .
Are you sure you want to continue connecting (
   ↪ yes/no/[fingerprint])? yes
Warning: Permanently added '14.139.189.217' (
   ↪ ECDSA) to the list of known hosts.
.
                  |
.    .      .  .-.  | .-..-.  .--.--.  .-.
 \  \  /  (.-'  |(   (    )|    |    |(.-'
  `'  `'     `--'`  `-'`-'  '    '   `  `--'
cs17d36@14.139.189.217's password:
Welcome to elementary OS 0.4.1 Loki (GNU/Linux
   ↪ 4.13.0-32-generic x86_64)

 * Website:  http://elementary.io/

99 packages can be updated.
0 updates are security updates.

Last login: Thu Apr 25 06:57:56 2019 from
   ↪ 137.97.99.121
$ ls
cs232  css  examples.desktop  fonts  index.html
   ↪  own_server.png  README.md
$ cd cs232
$ ls
1.task_linux_commands  2.task_gpa  3.
   ↪ task_networking  4.ssh_rsync_scp  5.
   ↪ task_ftp  6.linux_installation  7.
   ↪ http_ftp_server  8.further_tasks  9.
   ↪ own_server  README.md
```

```
$ cd ..
$ touch samplefile
$ rm samplefile
$ mkdir sampledirectory
$ rm -r sampledirectory
```

6 Rsync

`rsync` is a fast, versatile file-copying tool. It can copy locally, to/from another host over any remote shell, or to/from a remote rsync daemon. It offers a large number of options that control every aspect of its behaviour and permit very flexible specification of the set of files to be copied. It is famous for its delta transfer algorithm, which reduces the amount of data sent over the network by sending only differences between the source files and the existing files in the destination. Rsync is widely use dfor backups and mirroring and as an improved copy command for everyday use.

Some of the additional features of rsync are

- support for copying links, devices, owners, groups, and permissions
- exclude and exclude-from options similar to GNU tar
- a CVS exclude mode for ignoring the same files that CVS would ignore
- can use any transparent remote shell, including ssh or rsh
- does not require super-user privilates
- pipelining of file transfers to minimize latency costs
- support for anonymous or authenticated rsync daemons (ideal for mirroring)

6.1 Usages

Basic syntax

```
rsync [options] <source> <destination>
```

Copy/Sync a file on a local computer

```
rsync -azvh <sourcefile> <destination>
```

Copy/Sync a directory on a local computer

```
rsync -azvh <sourcedirectory> <destination>
```

Copy/Sync a directory to a remote computer

```
rsync -azvh <sourcedirectory> user@serverip:<destination>
```

7 SCP

scp copies files between hosts on a network. It uses ssh for data transfer and uses the same authentication and provides the same security as ssh. scp will ask for passwords or passphrases if they are needed for authentication.
The source and destination may be specified as a local pathname, a remote host with optional path in the form of [user@]host:[path] or a URI in the form scp://[user@]host[:port][/path]. Local file names can be made explicit using absolute or relative pathnames to avoid scp treating file names containing : as host specifiers.

7.1 Usages

Copy the file ”foobar.txt” from a remote host to the local host

```
$ scp your_username@remotehost.edu:foobar.txt /some/local/director
```

Copy the file ”foobar.txt” from the local host to a remote host

```
$ scp foobar.txt your_username@remotehost.edu:/some/remote/directo
```

Copy the directory ”foo” from the local host to a remote host’s directory ”bar”

```
$ scp -r foo your_username@remotehost.edu:/some/remote/directory/b
```

Copy the file ”foobar.txt” from remote host ”rh1.edu” to remote host ”rh2.edu”

```
$ scp your_username@rh1.edu:/some/remote/directory/foobar.txt \
    your_username@rh2.edu:/some/remote/directory/
```

8 Linux Installation

Installing a linux distribution on system requires the creation of an ext4 (sometimes ext3) formatted partition. Different directories like `/usr /home /boot` can be installed on separate partitions formatted as ext4. These should be mounted during `init` via fstab entries (which will be created by default for most linux distributions).

If the system uses UEFI then a separate partition formatted as `FAT32` is required for `ESP` which contains boot managers and boot information for the UEFI.

The installation of required proprietry drivers may be provided with the installation image. If not it should be manually installed.

After the copying of required OS files localization settings like time, timezone, language, etc.. are configured. The installation interface might provide options for the creation of additional user accounts for the installed operation system. The password for the root user might be configured during installation, or set to a default password.

Images
https://github.com/ceccs17d36/cs232/tree/master/6.linux_installation

9 HTTP Server

Creating HTTP using nginx is simple. Installation of nginx can be done in ubuntu by using apt package manager.

```
# apt install nginx
```

To start nginx server the following command can be used

```
# nginx -s start
```

To stop nginx server

```
# nginx -s stop
```

To restart nginx server

```
# nginx -s restart
```

The most common server hosting directory (server root) is /var/www/html/. This can be changed in nginx configuration file located at /etc/nginx/nginx.conf. It contains configuration settings like ports, hostnames, ect.. By creating an index.html at the server root directory and restart nginx will make nginx load the index.html file when the IP address of the server machine is accessed through the same machine or a machine connected to the same network.

10 FTP Server

FTP server can be created using any FTP server application. This server was created using `vsftpd` (Very Secure FTP daemon). `vsftpd` server can be started using

```
# vsftpd
```

FTP is used to transfer files across devices. Remote file transfer can be done through any FTP clients in conjunction with an FTP server. Linux has a builtin FTP client, `ftp`.
To access FTP server on an IP

```
$ ftp <serverip>
```

Usage of ftp client is demostrated in FTP Usage and Commands

11 Package Management

In linux, programs are provided through package managers. These packages undergo customizations and testings so the the software program is completely compatible with the installed linux distribution. These packages are tested and deployed by the distribution mainters.
Each linux distribution has their own package managers. For example Debian and Ubuntu uses `apt, apt-get, deb`, while Fedora and Redhat uses `rpm, yum, dnf`. Another example is `pacman` which is used by `arch` and other arch based distributions like Manjaro uses.

11.1 Usage of apt

Update package repository list

```
# apt update
```

Upgrade installed packages

```
# apt upgrade
```

Search for a package

```
# apt search <packagename>
```

Install a package

```
# apt install <packagename>
```

Uninstall a package

```
# apt remove <packagename>
```

Uninstall a package and remove configuration files

```
# apt remove --purge <packagename>
```

List upgradable packages

```
# apt list --upgradable
```

List installed packages

```
# apt list --installed
```

Remove unneeded packages

```
# apt autoremove
```

12 Perl

Perl 5 is a high level, general purpose, interpreted, dynamic language. ”Perl” is a family of languages. While Perl generally refers to Perl 5, there is also another language Perl 6, which is a sister programming language to Perl 5, which is not indented as replacement for Perl 5.
It supports object-oriented, procedural and functional programming. It is easily extendable using modules. It provides powerful tools for text manipulation. Perl interfaces with external C/C++ libraries. It can be embedded in web servers and database servers.

Program to print two numbers

```
#!/usr/bin/perl

use strict;
use warnings;

print "Enter two numbers \n";
$a = <>;
$b = <>;
my $sum = $a+$b;
print "Sum = $sum\n";
```

13 LAMP Stack

LAMP - Linux Apache MySQL PHP
LAMP stack is a web development platform which uses Linux as the operating system, Apache as the web server, MySQL as the relational database management system and PHP as the server side scripting language.
Apache is a web server similar to nginx. It hosts files in a specified directory as server root. Apache has modules for executing PHP at the server. This creates dynamic websites.
The data for the webpage like login credentials, user details, etc.. in a MySQL database running on the server system.
Most commonly MySQL is replaced by MariaDB since the latter is an open source fork of the former.

All the packages required for the lamp stack can be installed through

```
# apt install apache2 mariadb-server php \
    libapache2-mod-php php-mysql
```

The default server root is /var/www/html. Deleting all the contents of this directory and adding an index.php file and then restarting apache2 server will result in the serving of the newly created index.php.

14 PHP

PHP is server side, object oriented scripting languages mostly used in conjunction with servers like apache and nginx. It can connect to SQL servers seamlessly which makes it an ideal choice for server side scripting.
HTML can be embedded into PHP file which is server through web servers.

Example PHP script

```
/* index.php */
<?php
echo "Today is " . date("Y/m/d");
?>
```

This script prints the current date when served through a server.

15 Kernel Compilation

Linux kernel is an open source kernal. It's code can be obtained through versioning systems like `git` or directly downloaded as archive from https://kernel.org/.
It is then extracted to a convenient directory. For the configuration and compilation of the kernel certain packages are required. These can be installed with

```
# apt install git fakeroot build-essential ncurses-dev \
    xz-utils libssl-dev bc flex libelf-dev bison
```

It can be then configured manully using

```
$ make menuconfig
```

Configuring the kernel manually can cause system crashes. The configuration can be applied by copying the current system kernel configuration from `/boot` directory.
The kernel can be compiled by

```
$ make
```

The kernel modules can compile by

```
$ make modules_install
```

The kernel can be installed by

```
# make install
```

16 Own web pages on server

The server was configured to use the home directory of the user account as a hosting webpage. So creating an index.html file into the home directory of the student in the server was enough to start a webpage.

17 Linux commands - redirection, pipes, filters

17.1 Redirection

Normally the output of a program is directed to `stdout`. Linux proveides <, >, <<, >> operators

< is used for input redirection(overwrite)
<< is used for input redirection(append)
> is used for output redirection (overwrite)
>> is used for output redirection (append)

```
cat firstfile > secondfile
```

17.2 Pipes

A pipe is a form of redirection (transfer of standard output to some other destination) that is used in Linux and other Unix-like operating systems to send the output of one command/program/process to another command/program/process for further processing. The Unix/Linux systems allow stdout of a command to be connected to stdin of another command. You can make it do so by using the pipe character '—'.

```
ls -l | grep "Aug"
```

17.3 filters

When a program takes its input from another program, it performs some operation on that input, and writes the result to the standard output. It is referred to as a filter.

```
ls -l | grep "Aug"
```

18 Linux commands on ownership and permissions

In linux, each file and directories has an owner and permissions defined for owners, superusers and everyone else. These can be viewed by longlisting the directory using

```
$ ls -l
```

In order to change ownership of a file

```
# chown <username>:<group> <filename>
```

Permissions of file can be modified using `chmod`

```
# chmod 755 <filename>
```

Each digit in the `mode` represensts permissions for superuser, owner and everyone else respectively.

19 Wget, cURL, grep

19.1 Wget

GNU `wget` is program that retrieves content from web servers. It is part of the GNU Project. Its name derives from World Wide Web and get. It supports downloading via HTTP, HTTPS, and FTP. Its features include recursive download, conversion of links for offline viewing of local HTML, and support for proxies.

```
$ wget -o <localfilename> <remoteurl>
```

19.2 cURL

cURL is a computer software project providing a library and command-line tool for transferring data using various protocols. It was first released in 1997. The name stands for ”Client URL”. The original author and lead developer is the Swedish developer Daniel Stenberg.

```
$ curl -fLO <remoteurl>
```

19.3 grep

grep is a command-line utility for searching plain-text data sets for lines that match a regular expression. Its name comes from the ed command g/re/p, which has the same effect: doing a global search with the regular expression and printing all matching lines.

```
$ grep "<searchstring>" <filename>
```

20 Experiment 4

20.1 Currently logged user and name

```
$ whoami
```

20.2 Current shell

```
$ echo $SHELL
```

20.3 Home directory

```
$ echo ~
```

20.4 Operating system type

```
$ uname -o
```

20.5 Path settngs

```
$ echo $PATH
```

20.6 Working directory

```
$ pwd
```

20.7 Number of logged in users

```
$ who
```

21 Experiment 5

21.1 OS, version, release number, kernel version

```
$ uname -v #kernel version
$ cat /etc/os-release #os version
```

21.2 Available shells

```
$ chsh -l
```

21.3 CPU information

```
$ cat /proc/cpuinfo
```

21.4 Memory information

```
$ cat /proc/meminfo
$ free -m
```

21.5 Harddisk information

```
# hdparm -I /dev/sda
```

21.6 File System (mounted)

```
$ mount
```

22 Experiment 6

The program reads two nubers and display a menu. The choice is again read and is used in a bash `case` statement and the result in printed on the screen

```
#!/usr/bin/env bash

printf "Enter two numbers\n"
read a
read b
printf "1.Add\n"
printf "2.Subtract\n"
printf "3.Multiply\n"
printf "4.Divide\n"
printf "5.Modulus\n"
printf "Enter choice : "
read c
case $c in
    1)
        printf "%d + %d = %d\n" $a $b $(($a+$b
           ↪ ));;
    2)
        printf "%d - %d = %d\n" $a $b $(($a-$b
           ↪ ));;
    3)
        printf "%d * %d = %d\n" $a $b $(($a*$b
           ↪ ));;
    4)
        printf "%d / %d = %d\n" $a $b $(($a/$b
           ↪ ));;
    5)
        printf "%d %% %d = %d\n" $a $b $(($a%
           ↪ $b));;
esac
```

23 Experiment 7

The programs check if the number of arguments is 2. Then it will check if the file exists. Then it will check if the username exist. Else it will add the username. If file is not found it will display an error. If the argument is not equal to 2 then it will display an error message.

```
#!/usr/bin/env bash

if [[ "$#" -ne 2 ]]
then
    echo "Illegal number of parameters"
else
    if [[ -f "$1" ]]
    then
        if grep -q "$2" "$1"; then
            echo "name already exist"
        else
            echo "$2" >> "$1"
        fi
    else
        echo "File does not exist"
    fi
fi
```

24 Experiment 8

24.1 Creating a repository

```
$ git init
```

24.2 Checking out a repository

```
$ git clone <repositoryurl> #clone the repository
$ git checkout <branchname> #checkout the branch
```

24.3 Adding contetnt to repository

```
$ git add <directory_or_file>
```

24.4 Committing the data to the repository

```
$ git commit -m <commitmessage>
```

24.5 Updating the local copy

```
# git pull
```

24.6 Comparing different versions

```
$ git diff <commit> <commit> [path]
```

24.7 Revert

```
$ git revert <commit>
```

24.8 Conflict and conflict resolution

```
$ git reset
```

25 Experiment 9

The program reads all the process sorted by percentage cpu usage and if the cpu usage is greater than 10 percentage. Then it starts terminating those processes. This file is named `terminate.sh` and the `~/.bash_profile` file is edited so that `termiante.sh` is executed at each system start.

```
#!/usr/bin/env bash
# terminate.sh
processes=$(ps -Ao pid,pcpu --sort=-pcpu | awk
   ↪  '$2>10')
pid=$(echo $processes | awk '{print $1'})

while read -r line; do
    echo "terminating $line"
    kill "$line"
done <<< "$pid"
```

```
# ~/.bash_profile
...
./terminate.sh
```

26 awk

Awk is a scripting language used for manipulating data and generating reports.The awk command programming language requires no compiling, and allows the user to use variables, numeric functions, string functions, and logical operators.

Awk is a utility that enables a programmer to write tiny but effective programs in the form of statements that define text patterns that are to be searched for in each line of a document and the action that is to be taken when a match is found within a line. Awk is mostly used for pattern scanning and processing. It searches one or more files to see if they contain lines that matches with the specified patterns and then performs the associated actions.

Awk is abbreviated from the names of the developers – Aho, Weinberger, and Kernighan.

```
awk options ’selection _criteria {action }’
   ↪ input-file > output-file
```