

CPSC 6240**Chenxi Hu****Biyang Fu****Due date: 2/12/2020 before 5pm.****Instructions:**

This homework is done in *teams of two*, or *individually*, your choice. If you are working with another student, that student must be from the same level as you are (4240 or 6240). You are allowed to use search engines, textbook/s, lecture notes, and any other sources you wish. But you are *not* allowed to help other teams with their work, give hints or solutions to anyone in the class. The reason it is called a *Try It* is because you will do some hands-on exercises and show the output or the results you got. You will also summarize other important things you have learned. You will take a number of screenshots (no, not with your phone, with your keyboard). All screenshots should be clearly legible and illustrate without a doubt what you are doing. You can open them in an image editor and trim off the parts you do not need to make images smaller. Insert them when answering the question, do not submit them separately as image files. Make space between the questions in this document and type your answers there. Please do not type in red.

What and how to submit

Please submit your .doc, .odt or .pdf file to canvas before the due date. All screenshots illustrating your work should be part of the document, not submitted separately. If working with another student, please include both of your names on top, as well as your course # 4240 or 6240. No name → no credit.

Grading and Points

Every question indicates how many points it is worth. 4000-level and 6000-level are graded differently, with points indicated as (x/y), where x is 4240 and y is 6240.

Exercises

1. We have learned a lot of Linux commands in the class so far. Compile a list of all the commands you have learned, organize them by group, with multiple flags/options, with an explanation of what they do. For example, process management, user management, etc. Indicate which commands need to run with sudo. You can make a table, if you wish. You will be adding to this list throughout the course. If working in a group of 2, this should be one complete list, not two partial ones. (8/8)

1-SYSTEM INFORMATION	COMMAND NAME	EXPLANATION
	uname -a	# Display Linux system information
	uname -r	# Display kernel release information
	cat /etc/redhat-release	# Show which version of redhat installed
	uptime	# Show how long the system has been running + load
	hostname	# Show system host name
	hostname -I	# Display the IP addresses of the host
	last reboot	# Show system reboot history
	date	# Show the current date and time
	cal	# Show this month's calendar
	w	# Display who is online
	whoami	# Who you are logged in as
	efibootmgr -v	Can modify boot menu entries from user space on a running system
	sudo pico /etc/default/grub	/etc/default/grub allows you to change GRUB variables.
2-HARDWARE INFORMATION	COMMAND NAME	EXPLANATION
	dmesg	# Display messages in kernel ring buffer
	cat /proc/cpuinfo	# Display CPU information
	cat /proc/meminfo	# Display memory information
	free -h	# Display free and used memory (- h for human readable, -m for MB, -g for GB.)
	lspci -tv	# Display PCI devices
	lsusb -tv	# Display USB devices
	dmidecode	# Display DMI/SMBIOS (hardware info) from the BIOS
	hdparm -i /dev/sda	# Show info about disk sda
	hdparm -tT /dev/sda	# Perform a read speed test on disk sda
	badblocks -s /dev/sda	# Test for unreadable blocks on disk sda
3 - PERFORMANCE	COMMAND NAME	EXPLANATION
	top	# Display and manage the top processes
	htop	# Interactive process viewer (top alternative)

MONITORING AND STATISTICS	mpstat 1	# Display processor related statistics
	vmstat 1	# Display virtual memory statistics
	iostat 1	# Display I/O statistics
	tail 100 /var/log/messages	# Display the last 100 syslog messages (Use /var/log/syslog for Debian based systems.)
	tcpdump -i eth0	# Capture and display all packets on interface eth0
	tcpdump -i eth0 'port 80'	# Monitor all traffic on port 80 (HTTP)
	lsuf	# List all open files on the system
	lsuf -u user	# List files opened by user
	free -h	# Display free and used memory (-h for human readable, -m for MB, -g for GB.)
	watch fd -h	# Execute "df -h", showing periodic updates
4 - USER INFORMATION AND MANAGEMENT	COMMAND NAME	EXPLANATION
	id	# Display the user and group ids of your current user.
	last	# Display the last users who have logged onto the system.
	who	# Show who is logged into the system.
	w	# Show who is logged in and what they are doing.
	groupadd test	# Create a group named "test".
	useradd -c "John Smith" -m john	# Create an account named john, with a comment of "John Smith" and create the user's home directory.
	userdel john	# Delete the john account.
	usermod -aG sales john	# Add the john account to the sales group
	sudo shutdown	usually scheduled, gives time to save files and shut down.
	halt	logs shutdown, stops services, flushes cached data to disk, halts kernel
	halt -p	powers down the system
	sudo reboot	Reboots system
	sudo visudo	Add user to sudo file
	pwconv	Reconciling differences between passwd and shadow
	sudo adduser username	Add user live in /etc/login.defs, /etc/default/useradd

	sudo useradd username	Add user live in /etc/adduser.conf, /etc/deluser.conf
	sudo useradd -D	To see all current values
	write username	Communicate with other users
	finger username	use finger to get information about one specific user
5 - FILE AND DIRECTORY COMMANDS	COMMAND NAME	EXPLANATION
	ls -al	# List all files in a long listing (detailed) format
	pwd	# Display the present working directory
	mkdir directory	# Create a directory
	rm file	# Remove (delete) file
	rm -r directory	# Remove the directory and its contents recursively
	rm -f file	# Force removal of file without prompting for confirmation
	rm -rf directory	# Forcefully remove directory recursively
	cp file1 file2	# Copy file1 to file2
	cp -r source_directory destination	# Copy source_directory recursively to destination. If destination exists, copy source_directory into destination, otherwise create destination with the contents of source_directory.
	mv file1 file2	# Rename or move file1 to file2. If file2 is an existing directory, move file1 into directory file2
	ln -s /path/to/file linkname	# Create symbolic link to linkname
	touch file	# Create an empty file or update the access
	cat file	and modification times of file.
	less file	# View the contents of file
	head file	# Browse through a text file
	tail file	# Display the first 10 lines of f ile
	tail -f file	# Display the last 10 lines of f ile
	man grep	find manual entry for command grep

6 - PROCESS MANAGEMENT	COMMAND NAME	EXPLANATION
	ps	# Display your currently running processes
	ps -ef	# Display all the currently running processes on the system.
	ps -ef grep processname	# Display process information for processname
	top	# Display and manage the top processes
	htop	# Interactive process viewer (top alternative)
	kill pid	# Kill process with process ID of pid
	sudo killall processname	# Kill all processes named processname
	sudo pkill -u johnny	Can kill user's processes
	program &	# Start program in the background
	bg	# Display stopped or background jobs
	fg	# Brings the most recent background job to foreground
	fg n	#Brings job n to the foreground
	systemctl	without args shows all loaded and active units
	systemctl list-units -type=service	Only shows services
	systemctl list-unit-files -type=service	Shows their files
	systemctl start servicename	Start a service
	systemctl stop servicename	Stop a service
	systemctl restart servicename	Restart a service
	systemctl reload servicename	Reload a service (reloads config files, not restarts the service)
7-NETWORKING	COMMAND NAME	EXPLANATION
	ifconfig -a	# Display all network interfaces and ip address
	ifconfig eth0	# Display eth0 address and details
	ethtool eth0	# Query or control network driver and hardware settings
	ping host	# Send ICMP echo request to host
	whois domain	# Display whois information for domain
	dig domain	# Display DNS information for domain

	dig -x IP_ADDRESS	# Reverse lookup of I P_ADDRESS
	host domain	# Display DNS ip address for domain
	hostname -i	# Display the network address of the host name.
	hostname -l	# Display all local ip addresses
	wget http://domain.com/file	# Download http://domain.com/file
	netstat -nutlp	# Display listening tcp and udp ports and corresponding programs
8 - ARCHIVES (TAR FILES)	COMMAND NAME	EXPLANATION
	tar cf archive.tar directory	# Create tar named archive.tar containing directory.
	tar xf archive.tar	# Extract the contents from archive.tar.
	tar czf archive.tar.gz directory	# Create a gzip compressed tar file name archive.tar.gz.
	tar xzf archive.tar.gz	# Extract a gzip compressed tar file.
	tar cjf archive.tar.bz2 directory	# Create a tar file with bzip2 compression
	tar xjf archive.tar.bz2	# Extract a bzip2 compressed tar file.
9 - INSTALLING PACKAGES	COMMAND NAME	EXPLANATION
	yum search keyword	# Search for a package by keyword. .
	yum install package	# Install package.
	yum info package	# Display description and summary information about package.
	rpm -i package.rpm	# Install package from local file named package.rpm
	yum remove package	# Remove/uninstall package.
	tar zxvf sourcecode.tar.gz cd sourcecode ./configure make make install	# Install software from source code.
	sudo apt list --installed	Check what packages are installed on your machine
	sudo apt list installed gcc / tcpdump	Check if gcc and tcpdump are installed.
	sudo apt-get install gcc	install gcc package
	sudo apt-get install gcc	remove gcc
	sudo apt-get install --reinstall gcc-<version>	reinstall gcc
	Sudo apt-get update	update your system

	Sudo apt-get upgrade	
10 - SEARCH	COMMAND NAME	EXPLANATION
	grep pattern file	# Search for pattern in file
	grep -r pattern directory	# Search recursively for pattern in directory
	locate name	# Find files and directories by name
	find /home/john -name 'prefix*'	# Find files in /home/john that start with "prefix".
	find /home -size +100M	# Find files larger than 100MB in /home
11 - SSH LOGINS	COMMAND NAME	EXPLANATION
	ssh host	# Connect to host as your local username.
	ssh user@host	# Connect to host as user
	ssh -p port user@host	# Connect to host using port
12 - FILE TRANSFERS	COMMAND NAME	EXPLANATION
	scp file.txt server:/tmp	# Secure copy file.txt to the /tmp folder on server
	scp server:/var/www/*.html /tmp	# Copy *.html files from server to the local /tmp folder.
	scp -r server:/var/www /tmp	# Copy all files and directories recursively from server to the current system's /tmp folder.
	rsync -a /home /backups/	#Synchronize/home to /backups/home
	rsync -avz /home	# Synchronize files/directories between the local
	server:/backups/	and remote system with compression enabled
13 - DISK USAGE	COMMAND NAME	EXPLANATION
	df -h	# Show free and used space on mounted filesystems
	df -i	# Show free and used inodes on mounted filesystems
	fdisk -l	# Display disks partitions sizes and types
	du -ah	# Display disk usage for all files and directories in human readable format
	du -sh	# Display total disk usage off the current directory

14 - DIRECTORY NAVIGATION	COMMAND NAME	EXPLANATION
	cd ..	# To go up one level of the directory tree. (Change into the parent directory.)
	cd	# Go to the \$HOME directory
	cd /etc	# Change to the /etc directory
	cd ~	change to your home directory
	cd ../../	navigate two levels up
	cd /	change to root directory

- As a sysadmin, you are responsible for system security. Compile “good practices” that you have learned so far. For example, (do not use this example in your writing) one of the good ideas is to disable the *chfn* that allows users to change their name. Many more good practices can be found throughout the lecture notes. Please organize by topic. And as above, one complete list, not two partial ones, if working in a group of 2. You will be adding to this list throughout the course. (7/6)

1 - ROOT ACCESS	COMMAND NAME	EXPLANATION
	sudo	# execute a command as the superuser
	sudo visudo	# Edit /etc/sudoers file and validate the syntax of the file upon saving

2 - BOOT PROCESS	systemctl	# Investigate the status of systemd and change its configurations
	journalctl	# Show all messages from current boot
3 - USER MANAGEMENT	sudo vipw	# Edit /etc/passwd file and validate the syntax of the file upon saving
	sudo vigr	# Edit /etc/group file and validate the syntax of the file upon saving
4 - PROCESS MANAGEMENT	ps	# See a snapshot of the system
	top	# Monitor resources use real time
	uptime	# Find out how long the system is active

3. You have seen a number of commands that display various system information, such as kernel version, available free memory, etc. Explain what these commands do, run them, and place edited screenshots into the same document. Label screenshots. (6/5)

- uname

#The uname command reports basic information about a computer's software and hardware.



```

biyangfu@biyangfu-VirtualBox: ~
File Edit View Search Terminal Help
biyangfu@biyangfu-VirtualBox:~$ uname
Linux

```

- uname -a

#It prints all the system information in the following order: Kernel name, network node hostname, kernel release date, kernel version, machine hardware name, hardware platform,

operating system.

```
biyangfu@biyangfu-VirtualBox:~$ uname -a
Linux biyangfu-VirtualBox 5.3.0-26-generic #28~18.04.1-Ubuntu SMP Wed Dec 18 16:40:14 UTC 2019 x86_64 x86_64 x86_64 GNU/Linux
```

- `uname -r`

#It prints the kernel release date.

```
biyangfu@biyangfu-VirtualBox:~$ uname -r
5.3.0-26-generic
```

- `hostname`

#hostname command in Linux is used to obtain the DNS(Domain Name System) name and set the system's hostname or NIS(Network Information System) domain name.

```
biyangfu@biyangfu-VirtualBox:~$ hostname
biyangfu-VirtualBox
```

- `hostname -I`

#-I : This option is used to get all IP(network) addresses. The option doesn't depend on the resolvability of the hostname.

```
biyangfu@biyangfu-VirtualBox:~$ hostname -I
10.0.2.15
```

- `last reboot`

#Use the 'last reboot' command, which will display all the previous reboot date and time for the system.

```
biyangfu@biyangfu-VirtualBox:~$ last reboot
reboot    system boot  5.3.0-26-generic Mon Feb 10 16:40   still running
reboot    system boot  5.3.0-26-generic Mon Jan 27 17:15 - 17:59   (00:43)
reboot    system boot  5.3.0-26-generic Mon Jan 27 17:08 - 17:59   (00:50)
reboot    system boot  5.3.0-26-generic Mon Jan 27 17:05 - 17:59   (00:53)
reboot    system boot  5.3.0-26-generic Mon Jan 27 16:57 - 17:05   (00:07)
reboot    system boot  5.3.0-26-generic Mon Jan 27 16:47 - 16:57   (00:09)
reboot    system boot  5.3.0-26-generic Mon Jan 27 16:39 - 16:57   (00:17)
reboot    system boot  5.0.0-23-generic Mon Jan 27 16:33 - 16:39   (00:05)

wtmp begins Mon Jan 27 16:33:36 2020
```

- `free`

#It displays the total amount of free space available along with the amount of memory used and swap memory in the system, and also the buffers used by the kernel.

```
biyangfu@biyangfu-VirtualBox:~$ free
              total        used        free      shared  buff/cache   available
Mem:           2035476       1153596        94868        71984       787012       657704
Swap:           969960           780        969180
```

- cal
#cal command shows the current month calendar as output.

```
biyangfu@biyangfu-VirtualBox:~$ cal
    February 2020
Su Mo Tu We Th Fr Sa
                1
 2  3  4  5  6  7  8
 9 10 11 12 13 14 15
16 17 18 19 20 21 22
23 24 25 26 27 28 29
```

- Research aliases. Create aliases that do the following: (7/6)
 - Alias lfiles will list all files/subdirectories in the current directory (should work across directories, so do not hard-code the path), even hidden files that start with a dot, in long format, with a slash indicating if an entry is a directory.

```
chenxi@chenxi-VirtualBox:~$ alias lfiles='ls -alF'
chenxi@chenxi-VirtualBox:~$ lfiles
total 140
drwxr-xr-x 19 chenxi chenxi 4096 Feb 11 18:51 ./
drwxr-xr-x  3 root   root   4096 Jan 30 22:05 ../
drwxr-xr-x  2 chenxi chenxi 4096 Jan 29 10:52 6240/
-rw-r--r--  1 chenxi chenxi 3673 Feb 11 17:18 .bash_history
-rw-r--r--  1 chenxi chenxi  220 Jan 24 13:47 .bash_logout
-rw-r--r--  1 chenxi chenxi 3771 Jan 24 13:47 .bashrc
drwx----- 15 chenxi chenxi 4096 Jan 30 20:41 .cache/
drwx----- 14 chenxi chenxi 4096 Feb  3 15:07 .config/
drwx-----  3 root   root   4096 Jan 30 20:41 .dbus/
drwxr-xr-x  2 chenxi chenxi 4096 Jan 26 13:13 Desktop/
drwx-----  2 chenxi chenxi 4096 Feb 11 16:59 dir1/
drwxr-xr-x  2 chenxi chenxi 4096 Jan 26 13:13 Documents/
drwxr-xr-x  2 chenxi chenxi 4096 Jan 26 13:13 Downloads/
-rw-r--r--  1 chenxi chenxi 8980 Jan 24 13:47 examples.desktop
-rw-r--r--  1 chenxi chenxi  13 Feb 11 16:59 file1.txt
drwx-----  3 chenxi chenxi 4096 Jan 26 14:45 .gnupg/
drwx-----  2 root   root   4096 Jan 30 20:41 .gvfs/
-rw-r--r--  1 chenxi chenxi 5068 Feb 11 18:48 .ICEauthority
drwx-----  3 chenxi chenxi 4096 Jan 26 13:12 .local/
drwxr-xr-x  3 chenxi chenxi 4096 Jan 29 09:53 Music/
drwxr-xr-x  2 chenxi chenxi 4096 Jan 26 13:13 Pictures/
```

- Alias fhid lists only hidden files in your home directory in long format


```

chenxi@chenxi-VirtualBox:/home$ alias fhid='ls -ald /home/chenxi/.*'
chenxi@chenxi-VirtualBox:/home$ pwd
/home
chenxi@chenxi-VirtualBox:/home$ fhid
drwxr-xr-x 19 chenxi chenxi 4096 Feb 11 18:51 /home/chenxi/.
drwxr-xr-x  3 root  root  4096 Jan 30 22:05 /home/chenxi/..
-rw-r----- 1 chenxi chenxi 3673 Feb 11 17:18 /home/chenxi/.bash_history
-rw-r----- 1 chenxi chenxi  220 Jan 24 13:47 /home/chenxi/.bash_logout
-rw-r----- 1 chenxi chenxi 3771 Jan 24 13:47 /home/chenxi/.bashrc
drwx----- 15 chenxi chenxi 4096 Jan 30 20:41 /home/chenxi/.cache
drwx----- 14 chenxi chenxi 4096 Feb  3 15:07 /home/chenxi/.config
drwx-----  3 root  root  4096 Jan 30 20:41 /home/chenxi/.dbus
drwx-----  3 chenxi chenxi 4096 Jan 26 14:45 /home/chenxi/.gnupg
drwx-----  2 root  root  4096 Jan 30 20:41 /home/chenxi/.gvfs
-rw-r----- 1 chenxi chenxi 5068 Feb 11 18:48 /home/chenxi/.ICEauthority
drwx-----  3 chenxi chenxi 4096 Jan 26 13:12 /home/chenxi/.local
-rw-r----- 1 chenxi chenxi  807 Jan 24 13:47 /home/chenxi/.profile
-rw-r----- 1 root  root    0 Feb 11 16:09 /home/chenxi/.selected_editor
drwx-----  2 chenxi chenxi 4096 Jan 26 14:45 /home/chenxi/.ssh
-rw-r----- 1 chenxi chenxi    0 Jan 26 14:46 /home/chenxi/.sudo_as_admin_successful
-rw-r----- 1 chenxi chenxi    5 Feb 11 18:48 /home/chenxi/.vboxclient-clipboard.pid
-rw-r----- 1 chenxi chenxi    5 Feb 11 18:48 /home/chenxi/.vboxclient-display.pid
-rw-r----- 1 chenxi chenxi    5 Feb 11 18:48 /home/chenxi/.vboxclient-draganddrop.pid
-rw-r----- 1 chenxi chenxi    5 Feb 11 18:48 /home/chenxi/.vboxclient-seamless.pid
-rw-r----- 1 chenxi chenxi 7678 Feb 11 18:51 /home/chenxi/.viminfo

```

- Place these aliases into the appropriate file, so that they work every time you boot the system. What file is this? Open that file in an editor, take a screenshot and insert below.

The file is the .bashrc in the home directory.

```

chenxi@chenxi-VirtualBox: ~
File Edit View Search Terminal Help

# Add an "alert" alias for long running commands.  Use like so:
# sleep 10; alert
alias alert='notify-send --urgency=low -i "[ $? = 0 ] && echo terminal || echo error" "$(
history|tail -n1|sed -e '\''s/^\s*[0-9]\+\s*//;s/[:;&]\s*alert$//'\''"'

# Alias definitions.
# You may want to put all your additions into a separate file like
# ~/.bash_aliases, instead of adding them here directly.
# See /usr/share/doc/bash-doc/examples in the bash-doc package.

if [ -f ~/.bash_aliases ]; then
. ~/.bash_aliases
fi

# enable programmable completion features (you don't need to enable
# this, if it's already enabled in /etc/bash.bashrc and /etc/profile
# sources /etc/bash.bashrc).
if ! shopt -oq posix; then
  if [ -f /usr/share/bash-completion/bash_completion ]; then
. /usr/share/bash-completion/bash_completion
  elif [ -f /etc/bash_completion ]; then
. /etc/bash_completion
  fi
fi

alias lfiles='ls -alF'
alias fhid='ls -ald /home/chenxi/.*'

```

5. Investigate these files: *.profile*, *.bashrc*, and *.bash_profile*. Why do they start with a dot? What is the role of each of these files, and when do they get executed? What are the contents of those files? Screenshot please. (8/6)

1) In Unix-like operating systems, any file or folder that starts with a dot character (for example, `/home/user/.config`), commonly called a dot file or dotfile, is to be treated as hidden – that is, the `ls` command does not display them unless the `-a` flag (`ls -a`) is used. In most command-line shells, wildcards will not match files whose names start with `dot(.)` unless the wildcard itself starts with an explicit `dot(.)`.

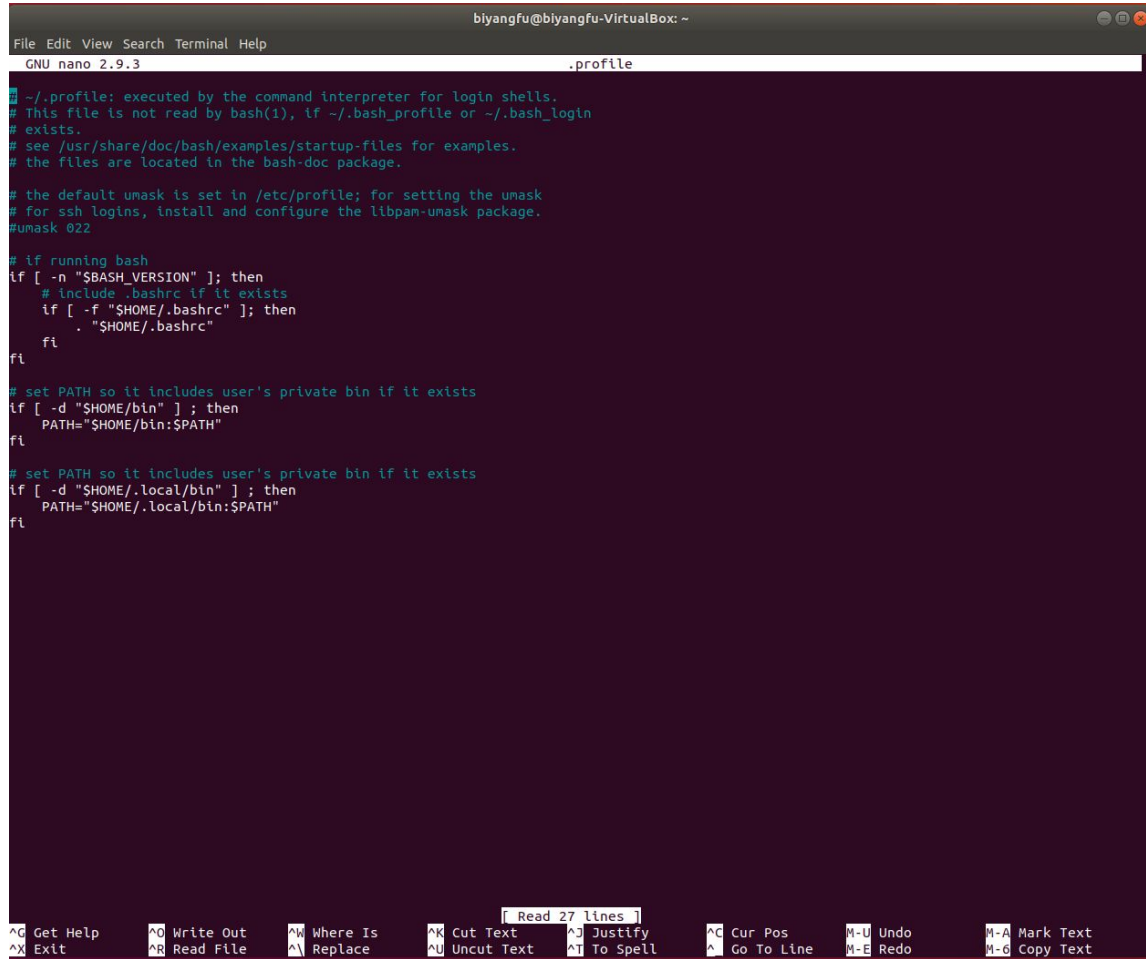
A convention arose of using dotfiles in the user's home directory to store per-user configuration or informational text. Early uses of this were the well-known dotfiles *.profile*, *.login*, and *.cshrc*, which are configuration files for the Bourne shell and C shell and shells compatible with them, and *.plan* and *.project*, both used by the `finger` and `name` commands.

2) `~/.profile` is the place to put stuff that applies to your whole session, such as programs that you want to start when you log in (but not graphical programs, they go into a different file), and environment variable definitions.

`~/.bashrc` is the place to put stuff that applies only to `bash` itself, such as alias and function definitions, shell options, and prompt settings. (You could also put key bindings there, but for `bash` they normally go into `~/.inputrc`.)

`~/.bash_profile` can be used instead of `~/.profile`, but it is read by `bash` only, not by any other shell. (This is mostly a concern if you want your initialization files to work on multiple machines and your login shell isn't `bash` on all of them.) This is a logical place to include `~/.bashrc` if the shell is interactive.

3)



The screenshot shows a terminal window titled "biyangfu@biyangfu-VirtualBox: ~". The terminal is running the GNU nano 2.9.3 text editor, editing the file ".profile". The file content is as follows:

```
#!/usr/bin/env bash
# ~/.profile: executed by the command interpreter for login shells.
# This file is not read by bash(1), if ~/.bash_profile or ~/.bash_login
# exists.
# see /usr/share/doc/bash/examples/startup-files for examples.
# the files are located in the bash-doc package.

# the default umask is set in /etc/profile; for setting the umask
# for ssh logins, install and configure the libpam-umask package.
umask 022

# if running bash
if [ -n "$BASH_VERSION" ]; then
    # include .bashrc if it exists
    if [ -f "$HOME/.bashrc" ]; then
        . "$HOME/.bashrc"
    fi
fi

# set PATH so it includes user's private bin if it exists
if [ -d "$HOME/bin" ] ; then
    PATH="$HOME/bin:$PATH"
fi

# set PATH so it includes user's private bin if it exists
if [ -d "$HOME/.local/bin" ] ; then
    PATH="$HOME/.local/bin:$PATH"
fi
```

The bottom of the terminal window shows the nano editor's status bar with various keyboard shortcuts and a message "[Read 27 lines]".

```
GNU nano 2.9.3 .bashrc

# ~/.bashrc: executed by bash(1) for non-login shells.
# see /usr/share/doc/bash/examples/startup-files (in the package bash-doc)
# for examples

# If not running interactively, don't do anything
case $- in
  *(*) ;;
  *) return;;
esac

# don't put duplicate lines or lines starting with space in the history.
# See bash(1) for more options
HISTCONTROL=ignoreboth

# append to the history file, don't overwrite it
shopt -s histappend

# for setting history length see HISTSIZE and HISTFILESIZE in bash(1)
HISTSIZE=1000
HISTFILESIZE=2000

# check the window size after each command and, if necessary,
# update the values of LINES and COLUMNS.
shopt -s checkwinsize

# If set, the pattern "*" used in a pathname expansion context will
# match all files and zero or more directories and subdirectories.
shopt -s globstar

# make less more friendly for non-text input files, see lesspipe(1)
[ -x /usr/bin/lesspipe ] && eval "$(SHELL=/bin/sh lesspipe)"

# set variable identifying the chroot you work in (used in the prompt below)
if [ -z "${debian_chroot:-}" ] && [ -r /etc/debian_chroot ]; then
    debian_chroot=$(cat /etc/debian_chroot)
fi

# set a fancy prompt (non-color, unless we know we "want" color)
case "$TERM" in
  xterm-color|*-256color) color_prompt=yes;;
esac

# uncomment for a colored prompt, if the terminal has the capability; turned
# off by default to not distract the user: the focus in a terminal window
# should be on the output of commands, not on the prompt
#force_color_prompt=yes

if [ -n "$force_color_prompt" ]; then
    if [ -x /usr/bin/tput ] && tput setaf 1 >&/dev/null; then
        # We have color support; assume it's compliant with Ecma-48
        # (ISO/IEC-6429). (Lack of such support is extremely rare, and such
        # a case would tend to support setf rather than setaf.)
        color_prompt=yes
    else
        color_prompt=
    fi
fi

if [ "$color_prompt" = yes ]; then
    PS1='${debian_chroot:+($debian_chroot)}\[\033[01;32m\]\u@\h\[\033[00m\]:\[\033[01;34m\]\w\[\033[00m\]\$ '
else
    PS1='${debian_chroot:+($debian_chroot)}\u@\h:\w\$ '
fi
unset color_prompt force_color_prompt

# If this is an xterm set the title to user@host:dir
case "$TERM" in
  xterm*|rxvt*)
    PS1="\[\e]0;${debian_chroot:+($debian_chroot)}\u@\h: \w\a\]$PS1"
  *)
    ;;
esac

^G Get Help    ^O Write Out  ^W Where Is   ^K Cut Text   ^J Justify    ^C Cur Pos    M-U Undo      M-A Mark Text
^X Exit        ^R Read File  ^\ Replace    ^U Uncut Text ^T To Spell   ^_ Go To Line  M-E Redo      M-G Copy Text
```



```
;;
*)
;;
esac

# enable color support of ls and also add handy aliases
if [ -x /usr/bin/dircolors ]; then
    test -r ~/.dircolors && eval "$(dircolors -b ~/.dircolors)" || eval "$(dircolors -b)"
    alias ls='ls --color=auto'
    #alias dir='dir --color=auto'
    #alias vdir='vdir --color=auto'

    alias grep='grep --color=auto'
    alias fgrep='fgrep --color=auto'
    alias egrep='egrep --color=auto'
fi

# colored GCC warnings and errors
#export GCC_COLORS='error=01;31:warning=01;35:note=01;36:caret=01;32:locus=01:quote=01'

# some more ls aliases
alias ll='ls -lF'
alias la='ls -A'
alias l='ls -CF'

# Add an "alert" alias for long running commands. Use like so:
# sleep 10; alert
alias alert='notify-send --urgency=low -i "${[ $? = 0 ]} && echo terminal || echo error)" "${history|tail -n1|sed -e '\''s/^$

# Alias definitions.
# You may want to put all your additions into a separate file like
# ~/.bash_aliases, instead of adding them here directly.
# See /usr/share/doc/bash-doc/examples in the bash-doc package.

if [ -f ~/.bash_aliases ]; then
    . ~/.bash_aliases
fi

# enable programmable completion features (you don't need to enable
# this, if it's already enabled in /etc/bash.bashrc and /etc/profile
# sources /etc/bash.bashrc).
if ! shopt -oq posix; then
    if [ -f /usr/share/bash-completion/bash_completion ]; then
        . /usr/share/bash-completion/bash_completion
    elif [ -f /etc/bash_completion ]; then
        . /etc/bash_completion
    fi
fi
```

⌘ Get Help

⌘ Write Out

⌘ Where Is

⌘ Cut Text

⌘ Justify

⌘ Cur Pos

⌘ Undo

⌘ Mark Text

⌘ Exit

⌘ Read File

⌘ Replace

⌘ Uncut Text

⌘ To Spell

⌘ Go To Line

⌘ Redo

⌘ Copy Text

6. Find out what umask is. Run the umask command and find out what your default umask is. Change it to another number with command umask 076. Now create a file. What permissions does it have? Create a directory. What permissions does the directory have? Are they the same or different? In which file is the default umask for directory creation stored? (7/6)


```

chenxi@chenxi-VirtualBox:~$ umask
0022
chenxi@chenxi-VirtualBox:~$ umask 076
chenxi@chenxi-VirtualBox:~$ cat > file1.txt
Hello world!
chenxi@chenxi-VirtualBox:~$ ls -l file1.txt
-rw----- 1 chenxi chenxi 13 Feb 11 16:59 file1.txt
chenxi@chenxi-VirtualBox:~$ mkdir dir1
chenxi@chenxi-VirtualBox:~$ ls -ld dir1
drwx-----x 2 chenxi chenxi 4096 Feb 11 16:59 dir1
chenxi@chenxi-VirtualBox:~$ █

```

- 1) umask is a command that determines the settings of a mask that controls how file permissions are set for newly created files.
- 2) My default umask is 0022.
- 3) The permissions of the file and the directory are different. The permissions of the created file are read and write or edit the file, however the permissions of the created directory are read, write or edit, and execute the directory.
- 4) The default umask for directory creation is stored in /etc/profile.

7. Change the login prompt that will last across boots, so that instead of the default `user@user-VirtualBox:~$` it shows: `LinuxRules:~$` (7/6)

```

biyangfu@biyangfu-VirtualBox:~$ vi ~/.bashrc
biyangfu@biyangfu-VirtualBox:~$ source ~/.bashrc
LinuxRules:~$ █

```

```

export PS1="LinuxRules:\W\$\n"
"~/.bashrc" 119L, 3802C

```

Graduate students have one additional exercise:

8. How does SHA-512 algorithm work? Please include a brief explanation below, half a page to a page should suffice (0/7)

SHA-512 is a hashing encryption algorithm which works in four phases.

- 1) Input formatting

The formatted input of the SHA-512 has three parts: the original message, padding bits, size of original message. And the size of the input is also limited to 1024 bits. In this phase, the input message will be formatted into the determined format. The padding bits are appended to the original message in order to get it to the desired length.

- 2) Hash buffer initialization

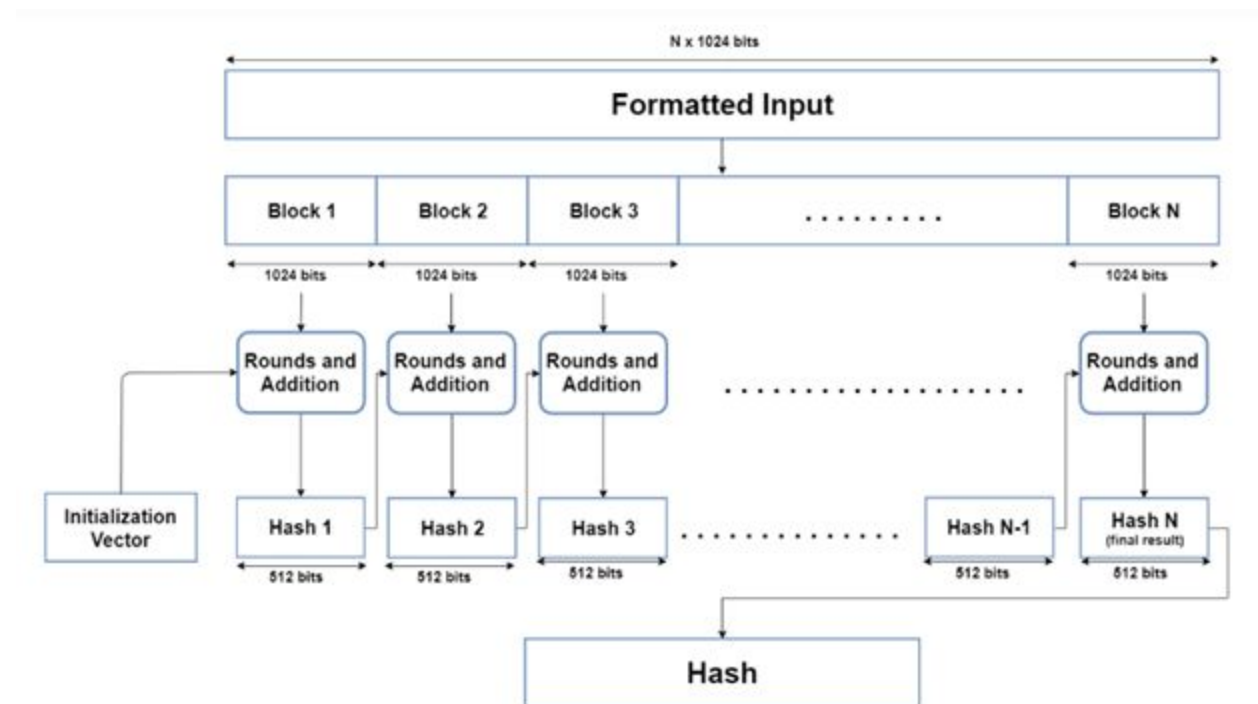
The procedure of the SHA-512 algorithm is composed of continuous block process on 1024 bits based on the previous block result. Specially, the first block uses the default value to start off the process. Furthermore, all intermediate results are stored in the hash buffer which consists of eight subparts.



(Image Credit: William Stallings, Cryptography and Network Security — Principles and Practice)

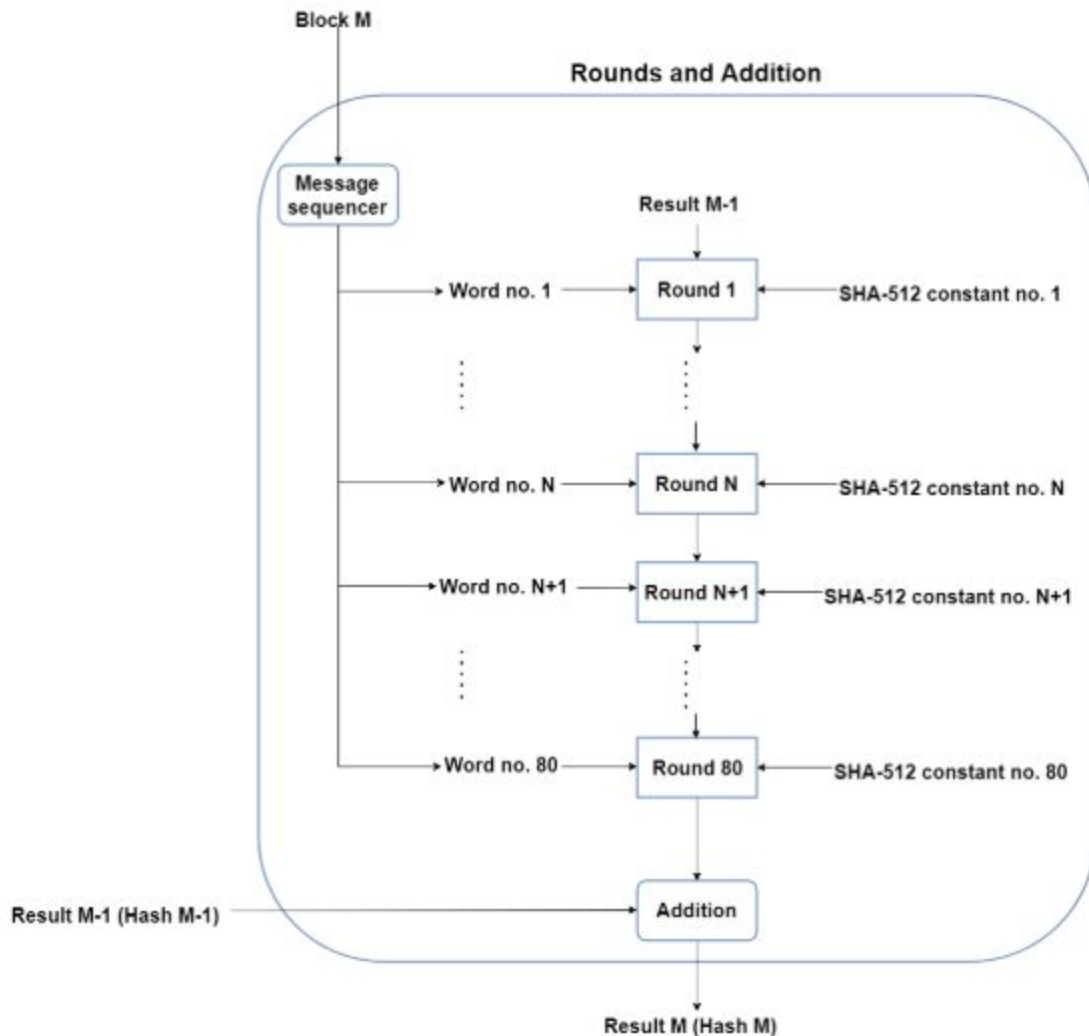
3) Message processing

The detailed procedures are shown in the following figure. In this phase, the formatted input data is processed by block. And the process consists of rounds and addition operations. The input data of each process unit includes formatted input data and the result from the previous process or the default value.



(Image Credit: William Stallings, Cryptography and Network Security — Principles and Practice)

The detailed 'Rounds and Addition' operations are shown in the following figure. In each round, one block, the output of the previous Round, and a SHA-512 constant are required. The first two components are mentioned above and SHA-512 constants refer to predetermined values.



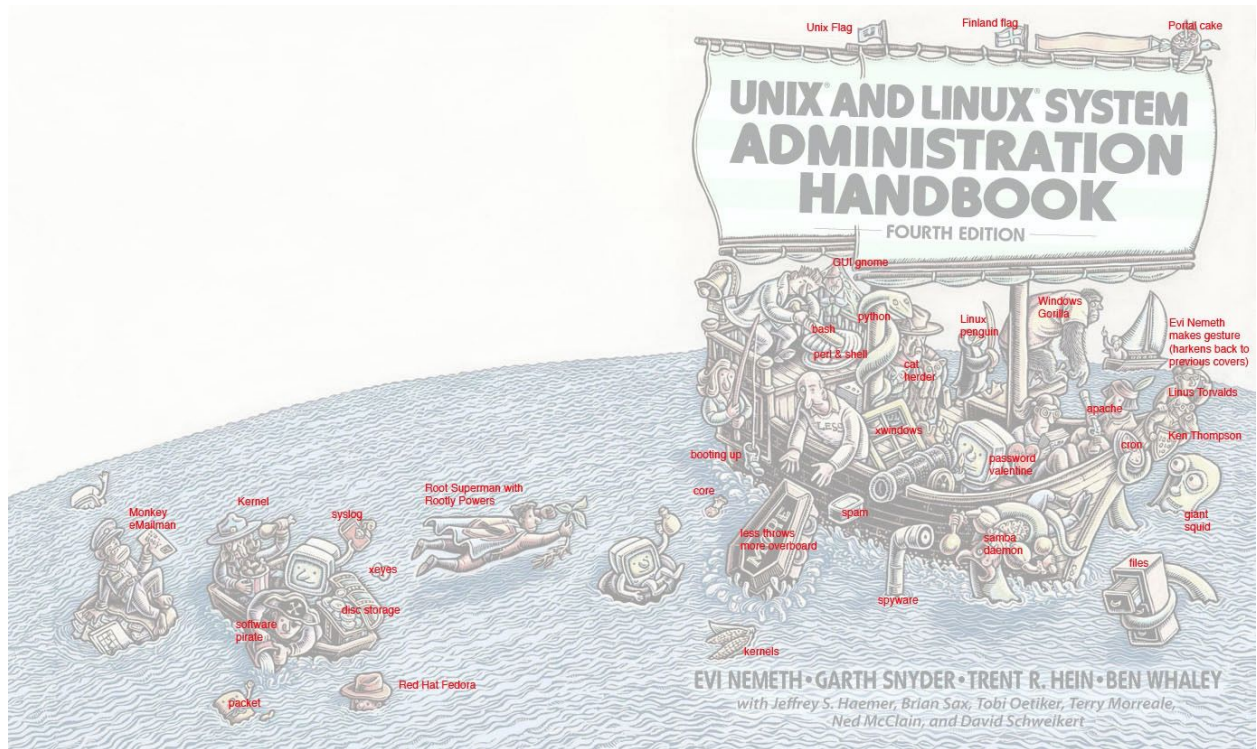
(Image Credit: William Stallings, Cryptography and Network Security — Principles and Practice)

4) Output

After all blocks in 1024 bits finish the above process, the final result of the SHA-512 algorithm is regarded as the ciphertext.

Extra credit question for everyone (2/2)

What does the book cover represent? Do you know why this may be? (2)



I think that the Linux system, like this sailboat, is composed of various things. Each thing plays a different role, and together forms a complete Linux system. This picture abstracts the various components of the Linux system into concrete objects according to their functions, properties, and names. This makes it easier for us to understand the meaning of each component in the entire Linux system.