Sanjivani Rural Education Society's
Sanjivani College of Engineering, Kopargaon
(An Autonomous Institute, Affiliated to Savitribai Phule Pune University, Pune)
NAAC 'A' Grade Accredited, ISO 9001:2015 Certified
Department of Computer Engineering
(NBA Accredited)

SANJIVANI
GROUP OF INSTITUTES

Introduction to Unix CIA ACTIVITY

[Report writing]

2022-23

Sanjivani Rural Education Society's
Sanjivani College of Engineering, Kopargaon-423 603
(An Autonomous Institute, Affiliated to Savitribai Phule Pune University, Pune)
NACC 'A' Grade Accredited, ISO 9001:2015 Certifie

MANDATORY COURSE

Academic Year: 2022- 2023 / Sem-IV
Second Year/ Computer Engineering / A- Division
Submission of Activity Report

Title :- Introduction to Unix CIA Activity**.**

Submitted by:

Roll No	Name of the Student	Signature
35.	Chothe Suraj Raybhan.	

Subject Incharge
Dr. Prakash N. Kalavadekar

• CERTIFICATE •

This is certify that students of class S.Y. BTech ‘div. A’, ‘Rollno-35’ has successfully completed their CIA-III Report on “ **Introduction to Unix CIA Activity**” under the guidance of **“ Dr. Prakash N. Kalavadekar”.**

Guided by
Dr. Prakash N. Kalavadekar

Introduction to Unix

About Unix

Unix is a multi-user, multi-tasking and open source operating system. Unix acts as an interface between the application software and hardware. The core functionalities of Unix operating system are handled by kernel . Users can instruct the operating system to perform task by issuing a command . Users issue these commands through an interface i.e. shell. Shell program interacts with the kernel by invoking a well-defined set of system calls. You can find Unix operating system mostly in servers, and also in many areas such as smartphones, cars, home appliances etc.

The advantage of Unix is that it is an open source software. Many organizations use Unix as the server operating system due to its stability and reliability.

E-Commerce application developed on Unix platform. Users can register, search for the products of their choice, add them to the cart and order the products. The application provides the following functionalities:

- User registration / user login
- Display product catalog and save preferences
- Select products to add to cart
- Generate reports

An administrator who manages this application in the Unix platform needs to perform the following:

1. Extract the data and process it
2. Generate customized reports
3. Ensure data availability by performing backup and recovery
4. Ensure security of data
5. Automate routine jobs

Commands

Work in Unix, you must be familier with commonly used file and directory related commands some are given

Requirements	Commands used
Work with files	`ls, rm`
Work with directories	`mkdir, rmdir, cd, pwd`
Work with the other tools	`cal, bc, who, uname, echo, printf, clear, tty, which, type, cat`

1)ls commands

The "ls" command is used to list the files and directories.

Syntax**:** ls [OPTION]... [FILE]..

Example:- ls

Output :-file1 file2 file3 New_directory

2)pwd commands

The command "pwd" displays the path to present directory.

Example:- /home/er111/Unix_Course/

3)cd command

The command "cd" is used to change directory to the specified location in the server.

Example:- cd application_directory/pwd

output :-/home/admin/application_directory

4)mkdir command

The command "mkdir" is used to create a new directory as shown below.

Example:- mkdir newly_created_directory

Ls

Output:- file1 file2 file3 newly_created_directory

5) rmdir command

To delete an empty directory "rmdir" command can be used as shown below.

Example:- rmdir newly_created_directory

Ls

Output:- file1 file2 file3

6) rm-rf commands

If the directory is not empty and the files and directories in it are no longer required, then the command "rm -rf" can be used to forcefully remove all the files and sub-directories in it."rm" command can be used to remove files.

Syntax**:** rm [OPTION]... FILE...

Example:- rm file1

Ls

Output:- file2 file3

7) man commands

The command "man" is used to refer to reference manual for detailed information about any command in Unix.

Example:- man ls

The above command will display the reference manual for "ls" command**.**

8) cal commands

The command "cal", if issued without any options, prints the calendar of the current month.

Syntax: cal [options] [[[day] month] year]

Example:- cal 16 01 2019

9) date commands

The command "date", if issued without any options, prints the current date in predefined format.

Example:- date

10)who commands

The command "who", if issued without any options, prints the users who have remotely logged into the UNIX machine. The display follows a pattern as shown below.

<Login name> <Pseudo-terminal ID> <Logged in time> <remote host name>

Synatx**:** who [OPTION]...

Example:- who

Output:- er111 pts/1 2019-01-10 11:16 (<systemname>)

11)uname commands

The command "uname", if issued without any options, prints the operating system name.

Example:- uname

Output:- Unix

12) echo

To display any information on the terminal use "echo" as shown below.

Example:- echo "You are learning Unix"

Output:- You are learning Unix

13)tty

The command "tty" prints the terminal information.

Example:- tty

Output:- /dev/pts/1

14) which

The command "which" prints the information path in which command/software is installed.

Example:- which mkdir

Output:- /dev/pts/1

Pipelining

With the help of a pipe, output of a command can be given as input to the next command. The symbol " | " between the two commands is called as "pipe".

Example:-Admin would like to know how many records are present in the log before opening it. To do so, use the following command

Example:-cat access.log | wc -l

output:- 300

Text processing

InfCart is an E-Commerce application developed and hosted on Unix platform. The admin of this application needs to generate customized reports such as product search patterns for the business analysis and business promotions.sample access log file to perform text searches and produce the desired reports on the product search patterns of the users.

A typical access log has the following format

<IP Address> <username> <Time> <Requested URL> <status code> <bytes delivered>

The requested URL contains the following:

<Request> <URL of home page>/<Brand>-<Type>-<Model number>/<index page>

Example:-

15)tail

tail command prints the last 'n' number of lines (10 lines by default) of a file. "tail" command is used to retrieve the recently appended 10 lines from a file, if no command option is specified

Syntax**:** tail [OPTION]... [FILE]...

16) head

head command prints the first 10 lines of a file. It might be required to compare the trends extracted from old logs and the recent log records to observe the changes. In this regard, to extract the first n records from the file, the command "head" can be used.

Syntax**:** head [OPTION]... [FILE]...

17)cut command

The cut command

- Extracts parts of each line of a file. It cuts out a set of bytes or characters or fields from each row of the file based on the delimiter
- Processes the file vertically as a result of which it produces a single or multiple columns in the output
- Acts like a filter that processes the file and extracts columns from it

Syntax**:** cut [OPTION]... [FILE]..

18) Comm command

Comm command compares two sorted files line by line and produces three column output when no options were passed to it. Column one contains lines unique to file1, column two contains lines unique to file2, and column three contains lines common to both files.

Syntax**:** comm [option]... FILE1 FILE2

19) diff command

The command "diff" compare files line by line. This command becomes helpful when you wish to know the exact record level differences. It tries to determine the smallest set of deletions and insertions between files. Typically, diff is used to show the changes between two versions of the same file.

Syntax**:** diff [OPTION]... FILES

Example:- diff Brands_old.dat Brands_new.dat

Output:- 1,2c1

< samsung

< samsung

--->> sony

20) sort command

"Sort" rearranges the lines of a file such that lines appear in numerical or alphabetical order. If no options are given, sorting is done as:

- Lines that begin with a number will come before lines that begin with a letter
- Lines that begin with the lowercase letter will come before lines that begin with same letter in uppercase

Syntax**:** sort [options] .. [file]

Example:- cut -d ' ' -f1,3 apache_access.log |uniq -c|sort -r

Output:- 3 12.123.80.60 Admin

2 10.123.80.65 User5

2 10.123.80.65 Admin

1 10.123.80.65 User4

1 10.123.80.65 User4

1 10.123.80.65 User3

1 10.123.80.65 User2

1 10.123.80.65 User2

1 10.123.80.65 User1

1 10.123.80.65 User1

1 10.123.80.65 Admin

21) tr

Translating characters

The tr command manipulates individual characters in a line.Specifically, it translates character by character. tr translates each character of expression1 to its corresponding mapped character in expression2.

Syntax**:** tr [options] expression1 expression2 standard_input

Note**:** tr takes input only from standard input. It will not take any filename as an argument.

22) rev –reversing the given text/line

rev command in Unix is used to reverse the text/lines character wise. This command basically reverses the order of the characters in each line and prints it to the standard output

example:- echo "Unix" | rev

output:- xinU

Pattern matching

Pattern matching is searching for the presence of set of tokens in a file or against a string. A regular expression is a template that either matches or doesn't match with a string. There are several meta characters that aid in matching the pattern in a text.

The content of a file or text input is processed line by line, to search for the pattern and the lines matching the pattern are displayed on the terminal. It can be used to search for lines of text that match one or many regular expressions.

Syntax**:** grep [OPTIONS] PATTERN [FILE...]

Example:- Admin would like to know the volume of search requests for the product "Inverter". Use the following grep command to do so

grep -i 'inverter' apache_access.log

output:- 10.123.80.65 - Admin [11:10:15] "GET /sony-Inverter-IN8009/index.html HTTP/1.0" 200 2048

10.123.80.65 - Admin [11:10:15] "GET /samsung-Inverter-sam98009/index.html HTTP/1.0" 206

10.123.80.65 - User3 [11:10:15] "GET /sony-Inverter-IN8009/index.html HTTP/1.0" 304

10.123.80.65 - User4 [11:10:15] "GET /sony-Inverter-IN8009/index.html HTTP/1.0" 404

10.123.80.65 - User5 [11:10:15] "GET /sony-Inverter-IN8009/index.html HTTP/1.0" 200 512

10.123.80.65 - User5 [11:10:15] "GET /sony-Inverter-IN8009/index.html HTTP/1.0" 200 4096

10.123.80.65 - User4 [11:10:15] "GET /sony-Inverter-IN8009/index.html HTTP/1.0" 403

10.123.80.65 - Admin [11:10:15] "GET /sony-Inverter-IN8009/index.html HTTP/1.0" 304

12.123.80.60 - Admin [11:10:15] "GET /sony-Inverter-IN8009/index.html HTTP/1.0" 200 2048

12.123.80.60 - Admin [11:10:15] "GET /sony-Inverter-IN8009/index.html HTTP/1.0" 200 1024

Pattern matching with extended grep

The command "grep" substitutes meta characters as strings and does not interpret their special meaning unless they are escaped with a '\'.To interpret the special meaning of meta characters the command "grep" with option "-E" is to be used.

Extended grep utility

- Print lines matching a pattern
- Searches the argument (can be a file or input from standard input) for lines matching a given pattern
- By default, searches for matching lines, unless otherwise specified
- Can be classified into two types, basic and extended

Syntax**:** egrep or grep -E

I/O redirection

It is done in 2 steps:

1. Store the records that contain either successful or unsuccessful product searches in a file
2. Apply text processing techniques on the file content, to extract the report containing searches made for the products mobile or laptop
3. To store the result of a command execution in a file, the concept of "I/O redirection" is used.

In Unix, inputs can be taken from any source other than the default input source and output can be redirected to any destination other than the default output destination.

- Output Redirection: This can be achieved by using the > operator
- Input Redirection: This can be achieved by using the < operator

Access Permission

The application should ensure that should not be accessible to others. In this regard, the following requirements has to be taken care of by the application admin.

- A user should not be able to make changes to the files created by another user
- A user should not be able to delete the files created by another user

The application admin can make use of Unix file access permissions. Unix users are grouped into the following categories based on the file and directory access privileges they have.

- User/Owner
- Group
- Others
- All

Each of these users has access permissions of three types i.e.

- Read
- Write
- Execut

In Unix, 12 bits are used for defining file access permissions. Of these, 9 are permission bits. They are again divided into three categories, namely, **user/owner**, **group** and **others**. The 9 permission bits, the first triad is assigned to the owner, the second is assigned to the group and last to others. In the below representation, 'd' represents a directory, the next 3 bits represent permissions to owner, followed by group followed by others.Let's execute the command "ls -l" to check the file permissions.

Example:- ls -l testfile

Sample output:- -rw-rw-r--. 1 testuser testuser 0 Sep 20 10:23 testfile

Numerical	r	w	x	Symbolic
0	0	0	0	---
1	0	0	1	--x
2	0	1	0	-w-
3	0	1	1	-wx
4	1	0	0	r--
5	1	0	1	r-x
6	1	1	0	rw-
7	1	1	1	rwx

A file with read, write and execute permission to all the users will have the following permissions.- rwxrwxrwx.

Umask

The command "umask" is used for computing the file permissions for a newly created file. A mask is a group of bits that restrict how the corresponding permission bit is set for a newly created file.

Syntax**:** umask <value>

Without the suffix value, the command "umask" displays the current value of the mask. By mentioning the value following the command, the current value of the mask gets changed.

Chmod

Changing the access permissions on file/directory - chmod command. To change permissions of a file or directory, the command "chmod" is used.

Syntax**:** chmod [OPTION]... MODE[,MODE]... FILE

mv command

The "mv" command is used to move or rename files. Unlike the "cp" command, with the "mv" command file will be moved to the destination location. The source copy is deleted.

Syntax**:** mv[option]...SOURCE...DIRECTORY.

S No	Option	Usage
1	-b or --backup	Take the backup of the destination file
2	-i or --interactive	Prompt before overwriting
3	-u or update	Move only when the SOURCE file is newer than the destination file or when the destination file is missing
4	-n or –no-clobber	Does not overwrite the existing file

Archiving files

Old data from the application may need archival, as it may not be in frequent use.

‘tar’ command saves many files together into a single archive, and can restore individual files from the archive.

Syntax**:** tar [option]...[file]...

cpio command

cpio command is used for creating archive and extracting files that are archieved. It can also be used to copy archived files from one location to another. The modes available are:

Copy-out mode**:** Here, the files given in standard input are copied into an archive.

Copy-in mode**:** Here, the files from the given archive are copied out and listed on standard output.

Copy-pass mode**:** This is used to move files from one location to another.

Working with editor

Shell scripts can be written in any of the editors and then interpreted by the shell. One such editor is "vi", which stands for visual editor. It is an editor based on UNIX created by Bill Joy in 1976.

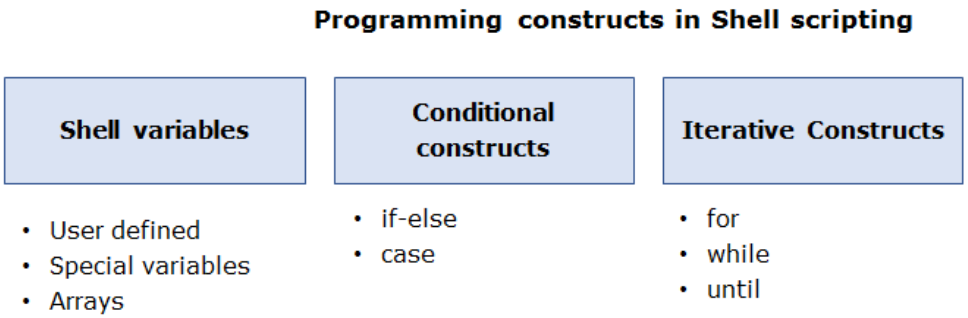
Different modes of operation

There are three modes of operation in vi editor.

- Command Mode (keys are interpreted as commands): When you enter the "vi" editor, you will be in "vi" command mode. You can issue commands to perform insert, append or delete. To insert text into the file, you need to switch to "vi" insert mode
- Insert Mode (keys are interpreted as raw text): In insert mode, you can type text into the file and use arrow keys to navigate
- Escape Mode (keys are interpreted for saving / exiting purposes): From insert mode, if you press ESC key, you will enter into command mode. On typing a ":", you can enter **into escape mode and give commands to save and quit the editor**

Shell scripts

The admin receives a requirement from the vendor to share the daily access records for analysis. The job of the admin is to back up the daily access log and truncate the log file. Such requirements need more than one command to be executed to perform the task. Moreover, such tasks need to be scheduled on a daily basis to meet the requirement of the vendor. simple shell scripts comprising of multiple commands to perform a specific task. learn shell scripting in three stages as shown below.



We learn about the each part of the shell

Shell variables

As in any programming language, to store data, variables are used. learn about three types of shell variables namely

- User defined variables
- Arrays
- Special variables

Shell variables can be accessed using a '$' in front of the variable name. The name of a variable can contain only letters (upper or lower case), numbers (0 through 9) or the underscore character. Quoted variables preserve white spaces whereas non-quoted variables don't. Non-quoted variables will split the variable into parts at white spaces. An argument enclosed in double quotes presents itself as a single word.

Arrays in shell

Arrays are a collection of elements. In shell scripting, data type declaration is not required for arrays. The values of the variables can be accessed using '$' as prefix to these elements. Arrays provide a method to group a set of data. Create and initialize array in shell script in the following way.

Example:- ARRAY=(10 20 30 40 50 60)

ARRAY_2=(10 20 30 “Infosys” “Amrit”)

Special variables

Variable	Description
$0	Current script name
$#	Number of arguments
$@	Arguments sent in command line
$?	Exit status of last command
$$	Process number of shell
$!	Process number of last background command

Input and output statements in shell Shell scripting allows user to provide input either through the command line or in an interactive way. The command "read" is used to take the input from the user.

Syntax**:** read [-p prompt] [-a array] [-d delim] [-ers] [-n nchars]

Conditional constructs

Shell script supports conditional constructs such as if-else and case statements. The following is the syntax for if-else construct.

Syntax

if [condition];

then

<statements>

elif [condition];

then

<statements>

else

<statements>

```
Fi
Example:- #!/bin/bash
if [ $# -eq 0 ] ; then
        echo -e " No arguments supplied"
        exit 1
fi
filename="$1"
if [ -e "$filename" ] ; then
echo -e "$filename exists"
        if [ -f "$filename" ] ; then
        echo -e " It is a regular file "
        elif [ -d "$filename" ] ; then
        echo -e "It is a directory"
        else
        echo -e " cannot determine the type of $filename"
        fi
else
echo "file does not exist"
fi
```

case' statement.

```
Syntax
case variable in
option 1)
     statements to be executed if option 1 matches;;
Option 2)
     statements to be executed if option 2 matches;;
*)
```

```
    default statement
esac
example:- #!/bin/bash
echo -e " Welcome to shell scripting"
read -p " enter a word to determine whether it is a file or directory:  " var
echo $(file $var)
echo
check=$(file $var | cut -d " " -f2)
echo $check
case $check in
"ASCII")
    echo -e " File $var is a plain text file ";;
"Bourne-Again")
    echo -e " File $var is a shell script file ";;
*)
    echo -e " File $var is of unknown type "S
esac
```

Operators in shell

Shell supports the following categories of operators

- Arithmetic operators
- Relational operators
- String operators
- File operators
- Logical operators

Iterative constructs in shell

Shell scripting supports the use of looping constructs just as in programming languages. The following are the iterative constructs in shell.

- for loop
- while loop
- until loop

Syntax of "for loop"

for variable in <list>; do

<statements>

Done

Syntax of "while" loop

while [condition] ; do

<statements>

Done

Syntax of "until" loop

until [condition] ; do

<statements>

Done

Difference between while and until loops

The "while" loop construct executes the statements when the given condition evaluates to true. On the other hand, the "until" loop executes the statements as long as the condition evaluates to false

Conclusion

Unix is really a great operating system with many different advantages and significant contributions to the development of technology.Although there are still many bad points, we believe that you will learn a lot when you get acquainted with Unix.Unix is a worthwhile operating system for us to learn and use. With the above article, you can answer yourself the definition of Unix and related information.UNIX programming has changed a lot, because UNIX and the subsystems that run on it are much more complicated and the language technologies have evolved. But it's still recognizably the same UNIX; and, while many of us no longer program in C, we still acknowledge C as the official UNIX reference and implementation language. This is true even if the UNIX is really Linux, the language is Python, and the application is a web site that's running Apache and MySQL