DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Session: June 2021 – July 2021

LABORATORY MANUAL

Semester : Minor Course

Subject Code : UE18CS305

Subject : OPERATING SYSTEMS – LABORATORY

Lab In charge : Prof. Chandravva Hebbi

Learning Outcomes

At the end of the course the student will be able to

1. Write programs to implement the basic functionality of an operating system and its components.
2. Write programs to implement the various scheduling algorithms and their performance tradeoffs.
3. Produce algorithmic solutions to process synchronization problems.
4. Write programs to implement memory, file and device management.

WEEK 1 LAB

1. **Execute and familiarize with Linux environment and commands**
   - **Getting used to basic commands on Linux Operating System – Process creation, Process monitoring, Process states, Linux File system tree, Linux File system commands**

1. Getting help in Unix

- man – view manual pages for Unix commands

2. Unix Shell Commands

- clear – clear screen
- history – show history of previous commands

3. Time and Date commands

- date – show current date and time
- sleep – wait for a given number of seconds
- uptime – find out how long the system has been up

4. Unix users commands

These commands allow you to get basic information about Unix users in your environment.

- whoami – show your username
- id – print user identity
- groups – show which groups user belongs to
- passwd – change user password
- who – find out who is logged into the system
- last – show history of logins into the system

5. Unix file operations

Navigating filesystem and managing files and access permissions:

- ls – list files and directories
- cp – copy files (work in progress)
- rm – remove files and directories (work in progress)
- mv – rename or move files and directories to another location
- chmod – change file/directory access permissions
- chown – change file/directory ownership

6. Text file operations in Unix

Most of important configuration in Unix is in clear text files, these commands will let you quickly inspect files or view logs:

- cat – concatenate files and show contents to the standard output
- more – basic pagination when viewing text files or parsing Unix commands output

- less – an improved pagination tool for viewing text files (better than more command)
- head – show the first 10 lines of text file (you can specify any number of lines)
- tail – show the last 10 lines of text file (any number can be specified)
- grep – search for patterns in text files

7. Unix directory management commands

Navigating filesystems and managing directories:

- cd – change directory
- pwd – confirm current directory
- ln – make links and symlinks to files and directories
- mkdir – make new directory
- rmdir – remove directories in Unix

8. Unix system status commands

Most useful commands for reviewing hostname configuration and vital stats:

- hostname – show or set server hostname
- w – display system load, who's logged in and what they are doing
- uname – print Unix system information

9. Reboot

- shutdown – graceful shutdown and reboot of your system
- halt – ungraceful (without stopping OS services) shutdown
- reboot – ungraceful reboot (without stopping OS services)

10. Networking commands in Unix

Most useful commands for inspecting network setup and exploring network connections and ports:

- ifconfig – show and set IP addresses (found almost everywhere)
- ip – show and set IP addresses (in recent Linux versions)
- ping – check if remote host is reachable via ICMP ping
- netstat – show network stats and routing information

- iptables – manage firewall rules on a Linux server
- netstat – network statistics and network routing information
- traceroute – tracing ICMP routes to a remote host

11. Process management

Listing processes and confirming their status, and stopping processes if needed:

- ps – list processes
- top – show tasks and system status
- kill – kill a process (stop application running)

12. Remote access commands

ssh is really the only way to go, but it's important to know telnet as well:

- telnet – clear-text (insecure) remote access protocol
- ssh – Secure SHell – encrypted remote access client
    - check out the SSH reference!

13. File transfers commands

Always useful to know how to copy files between servers or just download some package from the web:

- ftp – clear-text (insecure!) File Transfer Protocol client
- sftp – secure (encrypted) version of FTP
- scp – secure (encrypted) version of cp command
- wget – download files from remote servers, HTTP/HTTPS and FTP

14. Privileged Access

- su – switch user (commonly used to become root)
- sudo – run commands with elevated (usually root-like) privileges
    - be sure to check out sudo reference

15. Unix system status commands

- who -r – confirm current run-level of your Unix/Linux OS
- uname – print Unix system information: hostname, kernel version, etc

2. **Write a C program to display an array in reverse using index.**
   Create Makefile (ex: make.mk below) and other files as shown below: (Hint: Refer to Makefile tutorial sent before to create these files)

   Client.c – contains main function to collect input on array elements from the user and calls reverse_array function
   Server.c – contains reverse_array function and prints the reversed array (use a separate function to print the reversed array)
   Header.h – contains function prototypes
   make.mk – contains targets and their dependencies

   Program Execution and Expected Output

   $make -f make.mk

   $./a.out

   ENTER SIZE OF AN ARRAY

   4

   ENTER ELEMENTS OF AN ARRAY

   1
   2
   3
   4

   Input array is

   1234

   Reversed array is

   4321

Submission

1. Basic LINUX commands (**ANY 10**) executed in the lab should be submitted in the following way:
   Command: What does the command do?
   Any two options (i.e flags or arguments) regarding the command
   Outcome of the command
2. Main program and all sub programs (dependency files, header file and Makefile) should be submitted. Steps to execute make and output of the program should be submitted.
3. Answer the following questions (Brief answers only)

   - **Why do we use Makefile?**

- **Is Makefile a shell script?**
- **What does “clean” do in Makefile?**
- **How does make learn about the last modified files to be complied?**
- **What does Cflags in Makefile mean?**
- **Why do we use -f option with make command?**

Reference Links:

1. https://www.unixtutorial.org/basic-unix-commands
2. https://www.tutorialspoint.com/makefile/makefile_example.htm