SECTION 4 NETWORK PROGRAMMING AND ADMINISTRATION EXERCISES

4.0 INTRODUCTION

In the earlier sections, you studied the Unix, Linux and C language basics. This section contains more practical information to help you know best about Socket programming, it contains different lab exercises based on Unix and C language. We hope these exercises will provide you practice for socket programming. Towards the end of this section, we have given the list of Unix commands frequently required by the Unix users, further we have given a list of port numbers to indicate the TCP/IP services which will be helpful to you during socket programming.

To successfully complete this section, the learner should have the following knowledge and skills prior to starting the section. S/he must have:

- Studied the corresponding course material of BCS-052 and completed the assignments.
- Proficiency to work with Unix and Linux and C interface.
- Knowledge of networking concepts, including network operating system, client-server relationship, and local area network (LAN).

Also, to successfully complete this section, the learner should adhere to the following:

- Before attending the lab session, the learner must already have written steps/algorithms in his/her lab record. This activity should be treated as home-work that is to be done before attending the lab session.
- The learner must have already thoroughly studied the corresponding units of the course material (BCS-052) before attempting to write steps/algorithms for the problems given in a particular lab session.
- Ensure that you include comments in your lab exercises. This is a practice, which will enable others to understand your program and enable you to understand the program written by you after a long time.

4.1 OBJECTIVES

After completing this section, you should be able to:

- understand the practical issue of TCP/IP;
- know the different task performed for network administration;

- develop network applications; and
- know the TCP/IP services.

4.2 LAB SESSIONS

It contains different lab exercises based on Unix, Linux and C language to provide you hand-on experience on Unix and Linux, to sharpen your programming skills and to provide knowledge necessary for developing network applications. We hope these exercises will provide the learners, practice for socket programming. Before attending the lab session, the learner must have already written steps/algorithms in his/her lab record. This activity should be treated as homework that is to be done before attending the lab session. The learner must have already thoroughly studied the corresponding units of the course material (BCS-052) before attempting to write steps/algorithms for the problems given in a particular lab session. Ensure that you include comments in your lab exercises. This is a practice, which will enable others to understand your program and enable you to understand the program written by you after a long time.

Session 1: Unix Basics

This session is your first introduction with Unix. You can try different commands available in Unix for system and network administrator. Let us start:

Exercise 1: Run the following commands and write the use of each command:

ipconfig	ping	telnet	diskperf	netdiag
netstat	pathping	ftp/tftp	fc	sfc
nbtstat	rcp	lpr	tracert	verifier
nslookup	route	lpq	net session	drivers
nettime	rsh	chkdsk	hostname	net account

Exercise 2: Find your Ethernet physical address. Configure the IPv6 address in your eth0 interface and update the DNS server according your addresses.

Exercise 3: Write the command to remove read permission for Group and Other on the file 'green'.

Exercise 4: Write the command to add search permission for everyone for your home directory (~/).

Exercise 5: Find all files in your directory starting at ~ that were modified or accessed within the last 2 days.

Exercise 6: Find and print all files in your file space whose size is less than 50 bytes.

Session 2: Socket Setup and Data Transfer

Exercise 1: Write the code in C language for a function **swap**, which exchanges two-socket address structures. Give at least two implementations in your solutions. Also compare the solutions and explain why and how it is better.

Exercise 2: Write a new function named inet_pton_loose that handles these scenarios:

- If the address family is AF_INET and inet_pton returns 0, call inet_aton and see if it succeeds.

- If the address family is AF_INET6 and inet_pton returns 0, call inet_aton and if it succeeds, return the IPv4-mapped IPv6 address.

Exercise 3: Write the client and server programs in C language for establishing termination of connection between client and server using TCP. Assume the server can handle only one client.

Exercise 4: Write the client and server programs in C language for simple data (hello) transfer between client and server using UDP. Client will send *hello server* message to the server program. In its reply the server will send *hello client* message. The server and client programs should reside on different computers in a network.

Exercise 5: Write the Echo client and server programs in C language using UDP. The Echo clients should verify whether the text string they received from the server is the same text string that they sent or not. Also use the shutdown system call to improve the performance programs.

Session 3: Basic Linux Administration

Exercise 1: Create a new user guest1 in the Guest Group using GUI tool in Linux. Now add another new user in a different group than the group of guest. Assume both users wants to write on a same text file "myfile", create an optimal file permission for this text file.

Exercise 2: When new media (like a flash drive, an external hard drive, an SD card, etc) is added to a system, it must be mounted to the file system. Show how Mounting/Unmounting is done in Linux File Systems with read only permission.

Exercise 3: Write a command to enable authentication for single-user mode on Linux.

Exercise 4: Write a command to disable Interactive Hotkey Startup at Boot in Linux.

Exercise 5: Setup a Network Connection in Linux. Configure it for Wireless networking. Once internet connection is established use **ss, netstat, tcptrack, Iftop** commands and check the results.

Session 4: Advanced Data Transfer

Exercise 1: Write the client and server programs in C language for connectionless communication between two different Unix computers in the same TCP/IP network. The server process receive a byte from the client process should and send back an should acknowledgement to the client process.

Exercise 2: Write the client and server programs in C language, where the server can exchange text with many client processes. A client process starts the communication with an input "start". After this the client process waits for the answer from the server. If server permits, it can further send any text message (with restriction of not more than 1000 words in a day). The communication goes on in this way until the client process sends the message "stop" to the server.

Session 5: Flow and Error Control

Exercise 1: Assume Client program is running on Machine A and server program on B. Write a program to ensure that the data received by server on machine B is the same data which was sent by the client program on machine A. Implement the scheme through which you can recover/calculate the lost data. Write the client and server programs in C language for showing the result.

Exercise 2: Write programs in C language for implementing the sliding window protocol of window size 5.

Session 6: Routing

Exercise 1: Write the client and server programs in C language for implementing the broadcasting in the local network.

Exercise 2: Write the program in C language for implementing the IP Routing protocol using Address tables.

Session 7: Utility Development

Exercise 1: Write the program in C language for implementing the utility similar to "Ping".

Help: Ping is actually an acronym for the words 'Packet INternet Groper'. The Ping utility is essentially a system administrator's tool that is used to see if a computer is operating and also to see if network connections are intact. Ping uses the Internet Control Message Protocol (ICMP) echo function, which is detailed in RFC 792.

Exercise 2: Write the program in C language for implementing address resolution using DNS tables.

Session 8: Advance Linux Administration

Exercise 1: Configure and launch an FTP server than install and test an FTP client in Linux.

Exercise 2: Configure a remote server and transfer a Directory to Remote Server.

Exercise 3: Create and Configure samba Server in Linux. Also, transfer files from client side.

Exercise 4: At the root level, check the status of services apmd, sendmail, and cups. Use chkconfig to turn cups off in runlevels 2, 3, 4, and 5. Use the run level editor GUI to turn 'cups' back on.

Exercise 5: Assume a user is running HTTP at the port number 80. You are required to change port number of HTTP for your client to 8080.

Session 9: Mail Transfer

Exercise 1: Write the program in C language for implementing the client for Simple mail transfer protocol.

Exercise 2: Write the program in C language for implementing the server for Simple mail transfer protocol. Where Server can handle maximum 5 clients concurrently.

Session 10: Client/Server Computing

Exercise 1: Write the client and server programs in C language, where client will send a file containing a C-program, server will compile and executes the file given by the client and if error occurs during compilation or execution server will send back the appropriate message to the client otherwise server will send the executable file to the client.

4.3 LIST OF UNIX COMMANDS

In this appendix, we have summarized some of the basic Unix commands you need to get started. For further details on UNIX commands use the **man** command.

Setup and Status Commands

logout	end your UNIX session
passwd	change password by prompting for old and new passwords
stty	set terminal options
date	display or set the date
finger	display information about users
ps	display information about processes
env	display or change current environment
set	C shell command to set shell variables
alias	C shell command to define command abbreviations
history	C shell command to display recent commands

File and Directory Commands

cat	concatenate and display file(s)
more	paginator - allows you to browse through a text file
less	more versatile paginator than more
mv	move or rename files
cp	copy files
rm	remove files
ls	list contents of directory
mkdir	make a directory
rmdir	remove a directory
cd	change working directory
pwd	print working directory name
du	summarize disk usage
chmod	change mode (access permissions) of a file or directory
file	determine the type of file
quota -v	displays current disk usage for this account
ls -a	list all files and directories
cd ~	change to home-directory
cd ..	change to parent directory
head file	display the first few lines of a file
tail file	display the last few lines of a file
grep 'keyword' file	search a file for keywords
command > file	redirect standard output to a file

command >> file	append standard output to a file
command < file	redirect standard input from a file
command1 | command2	pipe the output of command1 to the input of command2
cat file1 file2 > file0	concatenate file1 and file2 to file0
sort	sort data

Editing Tools

pico	simple text editor
vi	screen oriented (visual) display editor
diff	show differences between the contents of files
grep	search a file for a pattern
sort	sort and collate lines of a file (only works on one file at a time)
wc	count lines, words, and characters in a file
Look	look up specified words in the system dictionary
awk	pattern scanning and processing language
gnuemacs	advanced text editor

Formatting and Printing Commands

lpq	view printer queue
lpr	send file to printer queue to be printed
Lprm	remove job from printer spooling queue
enscript	converts text files to POSTSCRIPT format for printing
lprloc	locations & names of printers, prices per page
pacinfo	current billing info for this account

Program Controls, Pipes, and Filters

CTRL-C	interrupt current process or command
CTRL-D	generate end-of-file character
CTRL-S	stop flow of output to screen
CTRL-Q	resume flow of output to screen
CTRL-Z	suspend current process or command
jobs	lists background jobs
bg	run a current or specified job in the background
fg	bring the current or specified job to the foreground
fg %1	foreground job number 1
!!	repeat entire last command line
!$	repeat last word of last command line
sleep	suspend execution for an interval
kill	terminate a process

nice	run a command at low priority
renice	alter priority of running process
&	run process in background when placed at end of command line
>	redirect the output of a command into a file
>>	redirect and append the output of a command to the end of a file
<	redirect a file to the input of a command
>&	redirect standard output and standard error of a command into a file (C shell only)
|	pipe the output of one command into another
kill %1	kill job number 1
ps	list current processes
kill 26152	kill process number 26152
who	list users currently logged in
a2ps -Pprinter textfile	print text file to named printer
lpr -Pprinter psfile	print postscript file to named printer
*	match any number of characters
?	match one character
man command	read the online manual page for a command
whatis command	brief description of a command
apropos keyword	match commands with keyword in their man pages
ls -lag	list access rights for all files
command &	run command in background

Other Tools and Applications

pine	electronic mail
bc	desk calculator
man	print UNIX manual page to screen
elm	another electronic mail program

4.4 LIST OF TCP/IP PORTS

The Internet Assigned Numbers Authority (IANA) specifies TCP/IP port numbers. As we discussed earlier in the course, all the port numbers are divided into three categories based on port number ranges: the Well Known Ports, the Registered Ports, and the Dynamic and/or Private Ports. The well known ports are those from 0 through 1023, registered ports are those from 1024 through 49151 and the Dynamic and/or Private Ports are those from 49152 through 65535. These ports are not used by any defined application. The following tables indicate the official (if the application-port combination is in the Internet Assigned Numbers Authority list) well-known and registered ports numbers.

Well-Known Ports (0 to 1023)

Port number	Description
0	Reserved; do not use
1	TCPMUX
5	RJE (Remote Job Entry)
7	ECHO protocol
9	DISCARD protocol
13	DAYTIME protocol
17	QOTD (Quote of the Day) protocol
18	Message Send Protocol
19	CHARGEN (Character Generator) protocol
20	FTP - data port
21	FTP - control (command) port
22	SSH (Secure Shell) - used for secure logins, file transfers and port forwarding
23	Telnet protocol - unencrypted text communications
25	SMTP - used for sending E-mails
37	TIME protocol
38	Route Access Protocol
39	Resource Location Protocol
41	Graphics
42	Host Name Server
49	TACACS Login Host protocol
53	DNS (Domain Name Server)
67	BOOTP (BootStrap Protocol) server; also used by DHCP (Dynamic Host Configuration Protocol)
68	BOOTP client; also used by DHCP
69	TFTP (Trivial File Transfer Protocol)
70	Gopher protocol
79	Finger protocol
80	HTTP (Hyper Text Transfer Protocol) - used for transferring web pages
88	Kerberos - authenticating agent
109	POP2 (Post Office Protocol version 2) - used for retrieving E-mails
110	POP3 (Post Office Protocol version 3) - used for retrieving E-mails
113	Ident - old server identification system, still used by IRC servers to identify its users

118	SQL Services
119	NNTP (Network News Transfer Protocol) - used for retrieving newsgroups messages
123	NTP (Network Time Protocol) - used for time synchronization
137	NetBIOS NetBIOS Name Service
138	NetBIOS NetBIOS Datagram Service
139	NetBIOS NetBIOS Session Service
143	IMAP4 (Internet Message Access Protocol 4) - used for retrieving E-mails
156	SQL Service
161	SNMP (Simple Network Management Protocol)
162	SNMPTRAP
179	BGP (Border Gateway Protocol)
194	IRC (Internet Relay Chat)
213	IPX
369	Rpc2portmap
389	LDAP (Lightweight Directory Access Protocol)
401	UPS Uninterruptible Power Supply
427	SLP (Service Location Protocol)
443	HTTPS - HTTP Protocol over TLS/SSLSSL (encrypted transmission)
445	Microsoft-DS (Active Directory, Windows shares, Sasser-worm, Agobot, Zobotworm)
445	Microsoft-DS SMB file sharing
464	Kpasswd
465	SMTP over SSL - CONFLICT with registered Cisco protocol
500	Isakmp
514	syslog protocol - used for system logging
530	Rpc
540	UUCP (Unix-to-Unix Copy Protocol)
542	commerce (Commerce Applications) (RFC maintained by: Randy Epstein [repstein at host.net])
554	RTSP (Real Time Streaming Protocol)
563	Nntp protocol over TLS/SSL (NNTPS)
587	email message submission (SMTP) (RFC 2476)
591	FileMaker 6.0 Web Sharing (HTTP Alternate, see port 80)
593	HTTP RPC Ep Map
636	LDAP over SSL (encrypted transmission)
666	id Software's Doom multiplayer game played over TCP (666 is a reference to the Number of the Beast)
691	MS Exchange Routing

873	rsync File synchronisation protocol
989	Ftp Protocol (data) over TLS/SSL
990	Ftp Protocol (control) over TLS/SSL
992	Telnet protocol over TLS/SSL
993	IMAP4 over SSL (encrypted transmission)
995	POP3 over SSL (encrypted transmission)

Registered Ports (1024 – 49151)

Port	Description
1080	SOCKS proxy
1099	RMI Registry
1099	RMI Registry
1194	OpenVPN
1198	The cajo project Free dynamic transparent distributed computing in Java
1214	Kazaa
1223	TGP: "TrulyGlobal Protocol" aka "The Gur Protocol"
1337	menandmice.com DNS (not to be confused with standard DNS port). Often used on compromised/infected computers - "1337" a "Leet speak" version of "Elite". See unregistered use below.
1352	IBM Lotus Notes/Domino RCP
1387	Computer Aided Design Software Inc LM (cadsi-lm)
1387	Computer Aided Design Software Inc LM (cadsi-lm)
1414	IBM MQSeries
1433	Microsoft SQL database system
1434	Microsoft SQL Monitor
1434	Microsoft SQL Monitor
1494	Citrix MetaFrame ICA Client
1547	Laplink
1547	Laplink
1723	Microsoft PPTP VPN
1723	Microsoft PPTP VPN
1863	MSN Messenger
1900	Microsoft SSDP Enables discovery of UPnP devices
1935	Macromedia Flash Communications Server MX
1984	Big Brother - network monitoring tool
2000	Cisco SCCP (Skinny)
2000	Cisco SCCP (Skinny)
2427	Cisco MGCP
2809	IBM WebSphere Application Server Node Agent

2967	Symantec AntiVirus Corporate Edition
3050	gds_db
3050	gds_db
3074	xbox live
3128	HTTP used by web caches and the default port for the Squid cache
3306	MySQL Database system
3389	Microsoft Terminal Server (RDP) officially registered as Windows Based Terminal (WBT)
3396	Novell NDPS Printer Agent
3689	DAAP Digital Audio Access Protocol used by Apple's ITunes
3690	Subversion version control system
3784	Ventrilo VoIP program used by Ventrilo
3785	Ventrilo VoIP program used by Ventrilo
6891-6900	MSN Messenger (File transfer)
6901	MSN Messenger (Voice)
11371	OpenPGP HTTP Keyserver

4.5 SUMMARY

In this section, we have given different lab exercises based on Unix, Linux and C language to provide to practical experience of socket programming. These exercises will help to develop/manage different network application by your own. Further this section covered the list of Unix commands frequently required by the Unix users, and a list of port numbers to indicate the TCP/IP services which will helpful to you during socket programming.

4.6 FURTHER READINGS

1) Brian W.Kernighan and Dennis M. Ritchie; *The C programming language* Prentice Hall.

2) Douglas E. Comer and David L. Stevens, “*Internetworking with TCP/IP. Vol.3: Client-server programming and applications BSD socket version*”, Prentice Hall.

3) W. Richard Stevens, “*TCP/IP Illustrated. Vol. 1: The protocols*”, Addison Wesley.

4) W. Richard Stevens, “*UNIX Network Programming*”, Prentice Hall.

5) http://www.linux.org

6) UNIX and Linux System Administration Handbook, Prentice Hall