Course Code : BCSL-056

Course Title : Network Programming and Administration Lab

Assignment Number : BCA(V)/L-056/Assignment/2024-25

Maximum Marks : 100

Weightage : 25% www.ignousite.com

Last Date of Submission : 31st October, 2024 (for July 2024 session)

: 30th April, 2025 (for January 2025 session)

Q1 (a): Write and execute a TCP client and a server program in C-language to perform the following tasks:

The TCP client program sends two strings to the TCP server program to find length of these to strings and return the sum of lenghts of these two strings. Also the TCP server program sends the concatenated strings to the client.

Ans. To implement the TCP client and server programs in C, we'll follow these steps:

1. **Server Program**: It will accept two strings from the client, calculate their lengths, sum the lengths, and send the concatenated result and sum of lengths back to the client.
2. **Client Program**: It will send two strings to the server and receive the concatenated string and the sum of lengths from the server.

Let's start with the code for the TCP server program:

TCP Server Program (server.c)

```
// tcp_server.c

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <unistd.h>

#include <arpa/inet.h>


#define PORT 8080

#define BUFFER_SIZE 1024


int main() {

   int server_fd, new_socket;

   struct sockaddr_in address;
```

```
int addrlen = sizeof(address);

char buffer[BUFFER_SIZE] = {0};

char response[BUFFER_SIZE] = {0};


// Create socket file descriptor

if ((server_fd = socket(AF_INET, SOCK_STREAM, 0)) == 0) {

   perror("socket failed");

   exit(EXIT_FAILURE);

}


// Set server address and port

address.sin_family = AF_INET;

address.sin_addr.s_addr = INADDR_ANY;

address.sin_port = htons(PORT);


// Bind the socket to the address and port

if (bind(server_fd, (struct sockaddr *)&address, sizeof(address)) < 0) {

   perror("bind failed");

   close(server_fd);

   exit(EXIT_FAILURE);

}


// Listen for incoming connections

if (listen(server_fd, 3) < 0) {

   perror("listen");

   close(server_fd);

   exit(EXIT_FAILURE);

}
```

```
// Accept incoming connection
if ((new_socket = accept(server_fd, (struct sockaddr *)&address, (socklen_t*)&addrlen)) < 0) {
    perror("accept");
    close(server_fd);
    exit(EXIT_FAILURE);
}

// Read strings from client
char str1[BUFFER_SIZE] = {0};
char str2[BUFFER_SIZE] = {0};
read(new_socket, str1, BUFFER_SIZE);
read(new_socket, str2, BUFFER_SIZE);

// Calculate lengths and concatenate strings
int len1 = strlen(str1);
int len2 = strlen(str2);
int total_length = len1 + len2;

snprintf(response, sizeof(response), "Total Length: %d\nConcatenated String: %s%s", total_length, str1, str2);

// Send response to client
send(new_socket, response, strlen(response), 0);

// Close connections
close(new_socket);
close(server_fd);
```

```
    return 0;

}
```

TCP Client Program (client.c)

```
// tcp_client.c

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <unistd.h>

#include <arpa/inet.h>


#define PORT 8080

#define BUFFER_SIZE 1024


int main() {

    int sock = 0;

    struct sockaddr_in serv_addr;

    char *str1 = "Hello";

    char *str2 = "World";

    char buffer[BUFFER_SIZE] = {0};


    // Create socket file descriptor

    if ((sock = socket(AF_INET, SOCK_STREAM, 0)) < 0) {

        printf("\n Socket creation error \n");

        return -1;

    }


    serv_addr.sin_family = AF_INET;

    serv_addr.sin_port = htons(PORT);
```

```
  // Convert IPv4 and IPv6 addresses from text to binary form
  if (inet_pton(AF_INET, "127.0.0.1", &serv_addr.sin_addr) <= 0) {
    printf("\nInvalid address/ Address not supported \n");
    return -1;
  }

  // Connect to server
  if (connect(sock, (struct sockaddr *)&serv_addr, sizeof(serv_addr)) < 0) {
    printf("\nConnection Failed \n");
    return -1;
  }
  // Send strings to server
  send(sock, str1, strlen(str1), 0);
  send(sock, str2, strlen(str2), 0);
    // Receive response from server
  read(sock, buffer, BUFFER_SIZE);
  printf("Server Response:\n%s\n", buffer);
    // Close the socket
  close(sock);
   return 0;
}
```

How to Compile and Run

1. **Compile the Server and Client Programs:**

```
gcc tcp_server.c -o tcp_server

gcc tcp_client.c -o tcp_client
```

2. Run the Server Program:

```
./tcp_server
```

3. Run the Client Program (in another terminal):

```
      ./tcp_client
```

Output:

```
Server Response:
Total Length: 10
Concatenated String: HelloWorld
```

(b) Run the following Linux commands on your machine and show the output:

- **cat**
- **sort**
- **ping**
- **more**
- **df-h**
- **tail - f**

Ans.

Cat Cat(concatenate) command is very frequently used in Linux. It reads data from the file and gives their content as output. It helps us to create, view, concatenate files. So let us see some frequently used cat commands.

Command:

$cat filename

Output:

It will show content of given filename

sort: The sort command is used to sort the lines of text files in ascending order.

Example usage:

sort file.txt

Input (assuming "file.txt" contains unsorted numbers):

5
2
9
1

Output:

```
1
2
5
9
```

ping: The ping command is used to test the reachability of a host on a network by sending ICMP echo requests and receiving ICMP echo replies.

Example usage:

ping google.com

Output:

```
PING google.com (172.217.167.238) 56(84) bytes of data.
64 bytes from google.com (172.217.167.238): icmp_seq=1 ttl=117 time=20.4 ms
64 bytes from google.com (172.217.167.238): icmp_seq=2 ttl=117 time=20.5 ms
...
```

more: The more command is used to display the contents of a text file one screen at a time.

Example usage:

more longfile.txt

Output (displays a portion of the file and waits for user input):

```
This is the first line of a long text file.
It contains multiple lines of text.
...
-- More -- (25%)
```

df-h To display information about disk drives in human-readable format (kilobytes, megabytes, gigabytes and so on), invoke the df command with the -h option:
$ df -h

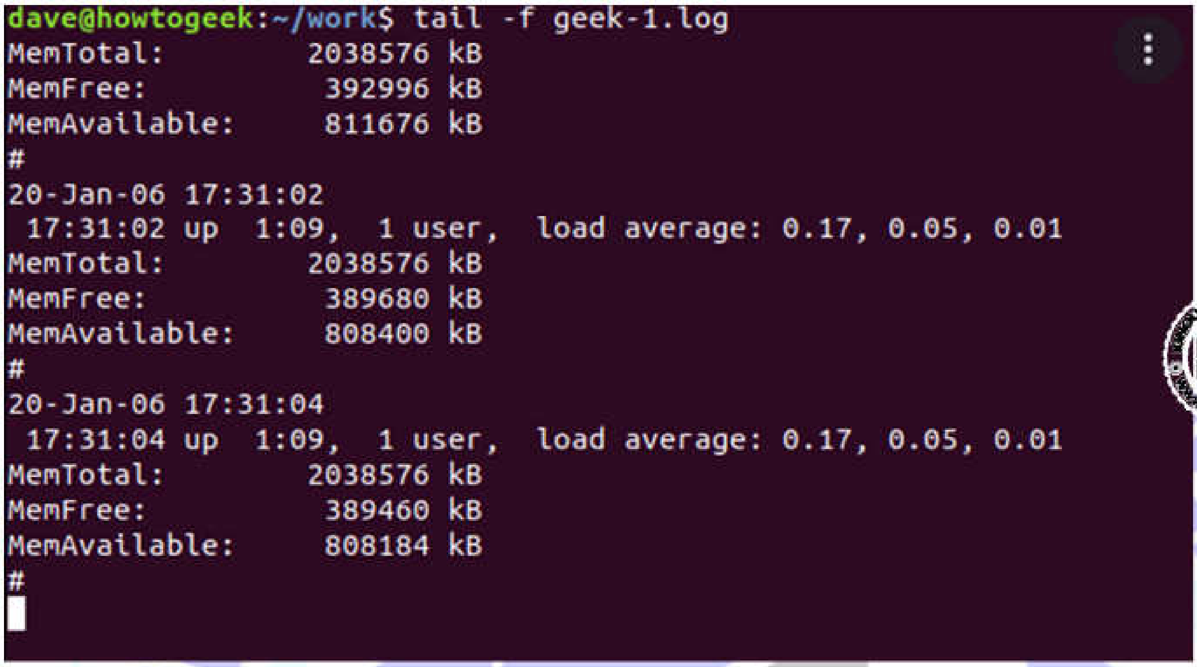
Output:

```
Filesystem       Size   Used   Avail  Use%   Mounted on
dev              7.8G   0      7.8G   0%     /dev
run              7.9G   1.8M   7.9G   1%     /run
/dev/nvme0n1p3   212G   176G   27G    88%    /
tmpfs            7.9G   145M   7.7G   2%     /dev/shm
tmpfs            7.9G   0      7.9G   0%     /sys/fs/cgroup
tmpfs            7.9G   24K    7.9G   1%     /tmp
/dev/nvme0n1p1   511M   106M   406M   21%    /boot
/dev/sda1        459G   165G   271G   38%    /data
tmpfs            1.6G   16K    1.6G   1%     /run/user/1000
```

tail -f This option is mainly used by system administration to monitor the growth of the log files written by many Unix program as they are running. This option shows the last ten lines of a file and will update when new lines are added. As new lines are written to the log, the console will update with the new lines. The prompt doesn't return even after work is over so, we have to use the interrupt key to abort this command. In general, the applications writes error messages to log files. You can use the -f option to check for the error messages as and when they appear in the log file.

$ tail -f

Output:


```
dave@howtogeek:~/work$ tail -f geek-1.log
MemTotal:        2038576 kB
MemFree:          392996 kB
MemAvailable:     811676 kB
#
20-Jan-06 17:31:02
 17:31:02 up  1:09,  1 user,  load average: 0.17, 0.05, 0.01
MemTotal:        2038576 kB
MemFree:          389680 kB
MemAvailable:     808400 kB
#
20-Jan-06 17:31:04
 17:31:04 up  1:09,  1 user,  load average: 0.17, 0.05, 0.01
MemTotal:        2038576 kB
MemFree:          389460 kB
MemAvailable:     808184 kB
#
```

Q2. (a) Configure and test the Telnet server in Linux.

Ans. To configure and test a Telnet server on a Linux system, you'll need to install and set up the Telnet server software. Here's a step-by-step guide:

Step 1: Install Telnet Server

1. **Update your package list**:

```
sudo apt update
```

2. **Install the Telnet server package**. Depending on your Linux distribution, the package may be called telnetd or inetutils-telnetd.

For Debian/Ubuntu:

```
sudo apt install telnetd
```

For CentOS/RHEL:

```
sudo yum install telnet-server

sudo yum install xinetd
```

Step 2: Configure Telnet Server

1. **Edit the /etc/inetd.conf or /etc/xinetd.d/telnet file** (depending on your system setup):

If your system uses inetd:

```
sudo nano /etc/inetd.conf
```

Ensure you have the following line uncommented or added:

```
telnet stream tcp nowait telnetd /usr/sbin/tcpd /usr/sbin/in.telnetd
```

If your system uses xinetd, edit the /etc/xinetd.d/telnet configuration file:

```
sudo nano /etc/xinetd.d/telnet
```

Ensure it contains:

```
service telnet
{
    flags          = REUSE
    socket_type    = stream
    wait           = no
    user           = root
    server         = /usr/sbin/in.telnetd
    log_on_failure  += USERID
    disable        = no
}
```

Change disable = yes to disable = no if necessary.

2. **Restart the inetd or xinetd service**:

For systems using inetd:

```
sudo systemctl restart inetd
```

For systems using xinetd:

```
sudo systemctl restart xinetd
```

Or use the service command:

```
sudo service xinetd restart
```

Step 3: Allow Telnet Port in Firewall

1. **Open port 23 in the firewall**:

For ufw (common on Debian/Ubuntu systems):

```
sudo ufw allow 23/tcp
```

For firewalld (common on CentOS/RHEL systems):

```
sudo firewall-cmd --permanent --add-port=23/tcp
sudo firewall-cmd --reload
```

Step 4: Test the Telnet Server

1. **Connect to the Telnet server**:

From the same machine or a different machine, use Telnet to connect:

```
telnet localhost
```

Or, if connecting from another machine, replace localhost with the server's IP address:

```
telnet <server-ip-address>
```

2. **Login**:

Once connected, you'll be prompted to enter a username and password. Use valid credentials of an existing user on the Telnet server.

Security Considerations

Telnet is **not secure** as it transmits data, including usernames and passwords, in plaintext. For secure remote access, consider using SSH (Secure Shell) instead.

Always restrict Telnet access to trusted IP addresses using firewall rules or by configuring access controls in /etc/hosts.allow and /etc/hosts.deny.

Troubleshooting

If you cannot connect, check if the Telnet server is running:

```
sudo netstat -tulnp | grep telnet
```

Ensure that no firewall rules are blocking port 23.

Check the Telnet server logs for any errors:

```
sudo tail -f /var/log/syslog    # For Debian/Ubuntu

sudo tail -f /var/log/messages  # For CentOS/RHEL
```

Following these steps will help you set up and test a Telnet server on your Linux machine successfully.

(b). Configure the DHCP server on the Linux operating system. Write all the steps involved in configuration. Sort each column of the table and show the result.

Ans. Configuring a DHCP (Dynamic Host Configuration Protocol) server on a Linux operating system involves several steps. Here's a general outline of the process along with the commands you can use. Note that the exact steps may vary depending on your Linux distribution.

Step 1: Install DHCP Server

1. Open a terminal.
2. Update your package repository:

sudo apt update (for Debian/Ubuntu)
sudo yum update (for CentOS/Red Hat)

3. Install the DHCP server package:

sudo apt install isc-dhcp-server (for Debian/Ubuntu)
sudo yum install dhcp (for CentOS/Red Hat)

Step 2: Configure DHCP Server

1. Edit the DHCP configuration file using a text editor (e.g., nano, vim):

sudo nano /etc/dhcp/dhcpd.conf

2. Add or modify the DHCP configuration settings as needed. For example:

```
subnet 192.168.1.0 netmask 255.255.255.0 {
  range 192.168.1.10 192.168.1.50;
  option routers 192.168.1.1;
  option domain-name-servers 8.8.8.8, 8.8.4.4;
}
```

Step 3: Start and Enable DHCP Server

1. Start the DHCP service:

sudo systemctl start isc-dhcp-server (for Debian/Ubuntu)
sudo systemctl start dhcpd (for CentOS/Red Hat)

2. Enable the DHCP service to start at boot:

sudo systemctl enable isc-dhcp-server (for Debian/Ubuntu)
sudo systemctl enable dhcpd (for CentOS/Red Hat)

Step 4: Verify DHCP Server Status

1. Check the status of the DHCP service:

sudo systemctl status isc-dhcp-server (for Debian/Ubuntu)
sudo systemctl status dhcpd (for CentOS/Red Hat)

Sorting Columns in a Table:

If you have a table in a text file that you want to sort, you can use the sort command. For example, let's say you have a file named "data.txt" with the following content:

```
Name     IP Address    MAC Address

HostA    192.168.1.10   00:11:22:33:44:55
```

```
HostB    192.168.1.20   AA:BB:CC:DD:EE:FF

HostC    192.168.1.30   11:22:33:44:55:66
```

You can sort the table by a specific column using the sort command. For example, to sort by the "Name" column:

sort -k1 data.txt

Output:

```
HostA    192.168.1.10   00:11:22:33:44:55

HostB    192.168.1.20   AA:BB:CC:DD:EE:FF

HostC    192.168.1.30   11:22:33:44:55:66
```

Similarly, you can sort by other columns. For instance, to sort by the "IP Address" column:

```
sort -k2 data.txt
```

Output:

```
HostA    192.168.1.10   00:11:22:33:44:55

HostB    192.168.1.20   AA:BB:CC:DD:EE:FF

HostC    192.168.1.30   11:22:33:44:55:66
```