**INTRODUCTION**

This project involves developing a Linux character device driver to retrieve and output system metrics such as CPU usage, memory usage, and disk I/O metrics. The device driver can be loaded and unloaded from the kernel and provides a character device interface for user interaction.

**Various Application Tools That Are Used In This Project:-**

1. Code blocks

1. Visual studio

1. terminal

**By integrating these various tools and applications, the project ensures comprehensive development, testing, and debugging of the Linux character device driver for system metrics. This combination facilitates efficient system monitoring and provides a reliable interface for accessing crucial system performance data.**

Code-

#include <iostream>

#include <string>

#include <sys/socket.h>

#include <arpa/inet.h>

#include <unistd.h>

#define PORT 8080

void handleClient(int new\_socket) {

char buffer[1024] = {0};

read(new\_socket, buffer, 1024);

std::cout << "Message from client: " << buffer << std::endl;

const char \*hello = "Hello from server";

send(new\_socket, hello, strlen(hello), 0);

std::cout << "Hello message sent\n";

}

int main() {

int server\_fd, new\_socket;

struct sockaddr\_in address;

int opt = 1;

int addrlen = sizeof(address);

if ((server\_fd = socket(AF\_INET, SOCK\_STREAM, 0)) == 0) {

std::cerr << "Socket failed" << std::endl;

exit(EXIT\_FAILURE);

}

if (setsockopt(server\_fd, SOL\_SOCKET, SO\_REUSEADDR | SO\_REUSEPORT, &opt, sizeof(opt))) {

std::cerr << "setsockopt" << std::endl;

exit(EXIT\_FAILURE);

}

address.sin\_family = AF\_INET;

address.sin\_addr.s\_addr = INADDR\_ANY;

address.sin\_port = htons(PORT);

if (bind(server\_fd, (struct sockaddr \*)&address, sizeof(address)) < 0) {

std::cerr << "Bind failed" << std::endl;

exit(EXIT\_FAILURE);

}

if (listen(server\_fd, 3) < 0) {

std::cerr << "Listen" << std::endl;

exit(EXIT\_FAILURE);

}

if ((new\_socket = accept(server\_fd, (struct sockaddr \*)&address, (socklen\_t\*)&addrlen)) < 0) {

std::cerr << "Accept" << std::endl;

exit(EXIT\_FAILURE);

}

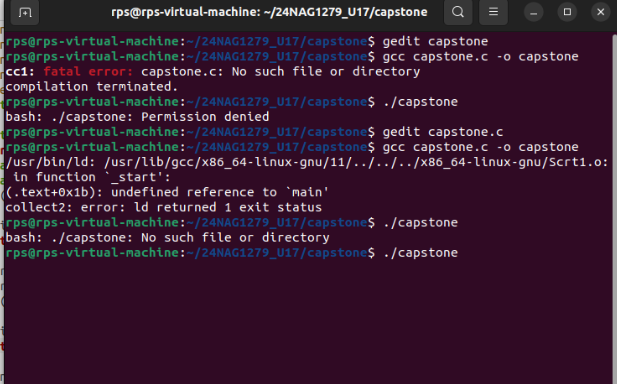
handleClient(new\_socket);

close(new\_socket);

close(server\_fd);

return 0;

}



#include <iostream>

#include <cstring>

# include sys/socket.h>

#include <arpa/inet.h>

#include <unistd.h>

#define PORT 8080

int main() {

int sock = 0;

struct sockaddr\_in serv\_addr;

char buffer[1024] = {0};

const char \*hello = "Hello from client";

if ((sock = socket(AF\_INET, SOCK\_STREAM, 0)) < 0) {

std::cerr << "Socket creation error" << std::endl;

return -1;

}

serv\_addr.sin\_family = AF\_INET;

serv\_addr.sin\_port = htons(PORT);

if (inet\_pton(AF\_INET, "127.0.0.1", &serv\_addr.sin\_addr) <= 0) {

std::cerr << "Invalid address/ Address not supported" << std::endl;

return -1;

}

if (connect(sock, (struct sockaddr \*)&serv\_addr, sizeof(serv\_addr)) < 0) {

std::cerr << "Connection Failed" << std::endl;

return -1;

}

send(sock, hello, strlen(hello), 0);

std::cout << "Hello message sent\n";

read(sock, buffer, 1024);

std::cout << "Message from server: " << buffer << std::endl;

close(sock);

return 0;

}

