**SERVER**

#include <stdio.h>

#include <stdlib.h>

#include <unistd.h>

#include <sys/types.h>

#include <sys/socket.h>

#include <netinet/in.h>

#include <fcntl.h>

#include <string.h>

#include <errno.h>

#include <arpa/inet.h> // Add this header for inet\_ntoa

#define PORT 8888

#define BUFFER\_SIZE 1024

void handle\_client(int client\_socket) {

char buffer[BUFFER\_SIZE];

ssize\_t bytes\_received;

char \*file\_name;

int file\_fd;

ssize\_t bytes\_sent;

// Receive the file name from the client

bytes\_received = recv(client\_socket, buffer, BUFFER\_SIZE - 1, 0);

if (bytes\_received < 0) {

perror("recv");

close(client\_socket);

return;

}

buffer[bytes\_received] = '\0';

file\_name = buffer;

// Open the file

file\_fd = open(file\_name, O\_RDONLY);

if (file\_fd < 0) {

// Send a message indicating the file does not exist

send(client\_socket, "File not found", strlen("File not found"), 0);

close(client\_socket);

return;

} else {

// Send the process ID to the client

sprintf(buffer, "PID: %d\n", getpid());

send(client\_socket, buffer, strlen(buffer), 0);

// Send the file contents to the client

while ((bytes\_sent = sendfile(client\_socket, file\_fd, NULL, BUFFER\_SIZE)) > 0) {

// Continue sending until the file is fully sent

}

if (bytes\_sent < 0) {

perror("sendfile");

}

// Close the file

close(file\_fd);

}

// Close the client socket

close(client\_socket);

}

int main() {

int server\_socket, client\_socket;

struct sockaddr\_in server\_addr, client\_addr;

socklen\_t client\_len = sizeof(client\_addr);

pid\_t pid;

// Create a TCP socket

server\_socket = socket(AF\_INET, SOCK\_STREAM, 0);

if (server\_socket < 0) {

perror("socket");

exit(EXIT\_FAILURE);

}

// Bind the socket to a specific IP address and port

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

server\_addr.sin\_addr.s\_addr = htonl(INADDR\_ANY);

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

if (bind(server\_socket, (struct sockaddr \*)&server\_addr, sizeof(server\_addr)) < 0) {

perror("bind");

exit(EXIT\_FAILURE);

}

// Enable the server to accept connections

if (listen(server\_socket, 5) < 0) {

perror("listen");

exit(EXIT\_FAILURE);

}

printf("Server PID: %d\n", getpid());

while (1) {

// Accept a client connection

client\_socket = accept(server\_socket, (struct sockaddr \*)&client\_addr, &client\_len);

if (client\_socket < 0) {

perror("accept");

continue; // Continue accepting other clients

}

printf("Connection from %s:%d\n", inet\_ntoa(client\_addr.sin\_addr), ntohs(client\_addr.sin\_port));

// Create a new process to handle the client

pid = fork();

if (pid < 0) {

perror("fork");

close(client\_socket);

continue; // Continue accepting other clients

}

if (pid == 0) {

// Child process

handle\_client(client\_socket);

exit(EXIT\_SUCCESS);

} else {

// Parent process

close(client\_socket);

}

}

// Close the server socket

close(server\_socket);

return 0;

}

**CLIENT**

#include <stdio.h>

#include <stdlib.h>

#include <unistd.h>

#include <sys/types.h>

#include <sys/socket.h>

#include <netinet/in.h>

#include <netdb.h>

#include <string.h>

#define SERVER\_IP "127.0.0.1"

#define PORT 8888

#define BUFFER\_SIZE 1024

int main() {

int client\_socket;

struct sockaddr\_in server\_addr;

struct hostent \*server;

char file\_name[BUFFER\_SIZE];

ssize\_t bytes\_sent, bytes\_received;

char buffer[BUFFER\_SIZE];

// Create a TCP socket

client\_socket = socket(AF\_INET, SOCK\_STREAM, 0);

if (client\_socket < 0) {

perror("socket");

exit(EXIT\_FAILURE);

}

// Get the server's IP address

server = gethostbyname(SERVER\_IP);

if (server == NULL) {

fprintf(stderr, "Error: no such host\n");

exit(EXIT\_FAILURE);

}

// Configure the server address

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

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

memcpy(&server\_addr.sin\_addr.s\_addr, server->h\_addr, server->h\_length);

// Connect to the server

if (connect(client\_socket, (struct sockaddr \*)&server\_addr, sizeof(server\_addr)) < 0) {

perror("connect");

exit(EXIT\_FAILURE);

}

// Get the file name from the user

printf("Enter file name: ");

scanf("%s", file\_name);

// Send the file name to the server

bytes\_sent = send(client\_socket, file\_name, strlen(file\_name), 0);

if (bytes\_sent < 0) {

perror("send");

exit(EXIT\_FAILURE);

}

// Receive the server's response

bytes\_received = recv(client\_socket, buffer, BUFFER\_SIZE, 0);

if (bytes\_received < 0) {

perror("recv");

exit(EXIT\_FAILURE);

}

buffer[bytes\_received] = '\0';

printf("%s\n", buffer);

// Close the socket

close(client\_socket);

return 0;

}

OUTPUT

SERVER

Server PID:5781

Connection from 127.0.0.1:49562

CLIENT

Enter the file name: hello.txt

PID:5336

Hello

ABY PIOUS VINOY

S6 CSE

ROLL NO : 57