**深 圳 大 学 实 验 报 告**

**课程名称：­ 计算机网络（Computer Networks）**

**实验名称： Application Layer Assignment**

**学院： 电子与信息工程学院**

**专业： 电子信息工程专业**

**指导教师： 毕宿志**

**报告人： 杨烨 班级： 06 学号： 2022280380**

**实验时间： 2023年10月**

**实验报告提交时间： 2023年11月3号**

**教务部制**

|  |
| --- |
| 1. **Purpose of experiment：**   (1) Proficient in socket library programming in python;  (2) Learn to create sockets to achieve communication between two processes;  (3) To achieve the transmission of two end files;  (4) Simulation delay and calculation;   1. **Experimental principle:**   Through socket programming, create a web server to send its required file content to the client.  A simple ping program is implemented through UDP protocol to measure round-trip delay (RTT).  Through socket programming, create a web server to send its required file content to the client.   1. **Content:**   **Assignment 1: Web Server Lab:**  Define a handle\_request() function to handle the HTTP request, which parses the request into a method, path, and protocol, looks for a file based on the request path, and returns a response.  Define a start\_server() function to start the server. This function creates a TCP socket and binds it to the local address and port, then listens for client connections. When the client connects to the server, the server receives the client request and invokes the handle\_request() function to process the request. The server then constructs the response and sends it back to the client. Finally close the connection and wait for the next client to connect.  In the main program, the start\_server() function is called to start the server. When the program is run, the server will run on port 8080 of the local address (localhost).  The code runs as shown in the following figure:    Figure 1 - http server response result  **Assignment 2: UDP Pinger:**  The first program is a client program that simulates the Ping command on a network. It does this by creating a UDP socket, setting the timeout time and server address, sending a ping message and receiving a pong response.  The specific programming ideas are as follows:  Import the required modules including socket, datetime, statistics, and random.  Set the server address and port, and the number of pings to send.  Create a UDP socket and set the timeout to 1 second.  Create an empty list to store round trip times (RTT).  Using a loop from 1 to the specified number of ping messages, do the following:  Gets the current time as the start time for sending the ping message.  Generates random packet loss. If no packet loss occurs, the ping message is sent.  Receive the pong message returned from the server and get the time when the message was received as the end time.  Calculates the round trip time (RTT) and adds it to the RTT list.  Delay by 1 second to simulate real network latency.  Print the response message and RTT.  If the list of RTT received is not empty, the average RTT is calculated and printed.  Close the socket.  The second programming is a simple UDP server program to process the ping message sent by the client and reply to the pong response. It does this by creating a UDP socket, binding to the specified server address and port, and then looping in a ping message and sending a pong response.  The specific programming ideas are as follows:  Import the socket module.  Set the server address and port number.  Create a UDP socket.  Bind the socket to the server address and port.  Enter the infinite loop and do the following:  Receives the ping message from the client and obtains the IP address of the client.  Build a pong response message and send it back to the client.  Close the socket.  These two programs implement a simple Ping client and Pong server, respectively, which can be used to simulate Ping/Pong operations in a network. The client sends a ping message, the server replies with a pong message after receiving the message, and the client calculates and prints the round trip time (RTT). Through this process, the situation of measuring network latency can be simulated.  The experimental results are shown in the figure below:    Figure 2 - Server response result  **Assignment 3: FTP**  The first piece of code is a simple Ping client program that simulates Ping operations on a network. It does this by creating a UDP socket, setting the timeout time and server address, sending a ping message and receiving a pong response. The specific programming ideas are as follows:  Import the required modules including socket, datetime, statistics, and random.  Set the server address and port, and the number of pings to send.  Create a UDP socket and set the timeout to 1 second.  Create an empty list to store round trip times (RTT).  Using a loop from 1 to the specified number of ping messages, do the following:  Gets the current time as the start time for sending the ping message.  Generates random packet loss. If no packet loss occurs, the ping message is sent.  Receive the pong message returned from the server and get the time when the message was received as the end time.  Calculates the round trip time (RTT) and adds it to the RTT list.  Delay by 1 second to simulate real network latency.  Print the response message and RTT.  If the list of RTT received is not empty, the average RTT is calculated and printed.  Close the socket.  The experimental results are shown in the figure below:    Figure 3 - Client request result  The second piece of code is a simple file transfer client program that requests the specified file from the server and saves it locally. It does this by creating a TCP socket, connecting to the specified server address and port, and then sending the file name to the server. After receiving the file name, the server reads the contents of the corresponding file and sends it back to the client. After receiving the data, the client saves it as a new file. The specific programming ideas are as follows:  Import the socket module.  Set the server address, port, and buffer size.  Define a run\_client function that takes as an argument the file name to be transferred.  In the run\_client function, create a TCP socket and connect to the server using the connect method.  Print a message indicating a successful connection to the server.  The file name is sent to the server using the send method.  The response sent by the server is received using the recv method and decoded as a string.  Check whether the response begins with "File", and if so, print an error message; Otherwise, do the following:  Create a new file with the received file name as the file name.  Use a loop to receive data sent by the server and write it to a file until empty data is received.  Print the message that the file received successfully.  Close the client socket.  In the main function, first get the file name entered by the user and call the run\_client function to run the client program.  The experimental results are shown in the figure below:    Figure 4 - Server response result |
| 1. **Conclusion and discussion:**   **Conclusion:**  Socket is an API used for inter-process communication in computer network. It provides a series of functions to realize network communication at different levels. socket programming in the computer network is a very efficient programming function library, which integrates the creation of sockets, accept the transmission of data and other functions, using it can solve many network transmission level problems  **Discussion:**  In this programming experiment, I used the socket library in python to build the server and control the text transfer protocol, and successfully realized the code I needed; In this experiment, I improved my programming ability and logical algorithm ability, mastered various library functions and their use, and learned about computer network, how to realize the communication between two processes and how to use the server to transfer files for clients. |
| 指导教师批阅意见：  成绩评定：  指导教师签字：  年 月 日  备注： |

注：1、报告内的项目或内容设置，可根据实际情况加以调整和补充。

2、教师批改学生实验报告时间应在学生提交实验报告时间后10日内。