Tcp\_client.py

|  |
| --- |
| import sys |
|  | from socket import socket, AF\_INET, SOCK\_STREAM |
|  |  |
|  | server\_name = sys.argv[1].encode() |
|  | server\_port = int(sys.argv[2]) |
|  | client\_socket = socket(AF\_INET, SOCK\_STREAM) |
|  | client\_socket.connect((server\_name, server\_port)) |
|  |  |
|  | sentence = input("\nEnter sentence: ").encode() |
|  | client\_socket.send(sentence) |
|  | modified\_sentence = client\_socket.recv(1024) |
|  | print(f"{server\_name.decode()} > {modified\_sentence.decode()}") |
|  | client\_socket.close() |

Tcp\_server.py

|  |
| --- |
| from socket import socket, AF\_INET, SOCK\_STREAM |
|  |  |
|  | server\_name = "10.0.4.26" |
|  | server\_port = 12000 |
|  | server\_socket = socket(AF\_INET, SOCK\_STREAM) |
|  | server\_socket.bind((server\_name, server\_port)) |
|  | server\_socket.listen(1) |
|  |  |
|  | print(f"Server {server\_name} is ready to receive on port {server\_port}") |
|  |  |
|  | while True: |
|  | connection\_socket, address = server\_socket.accept() |
|  | sentence = connection\_socket.recv(1024) |
|  | sentence = sentence.upper() |
|  | connection\_socket.send(sentence) |
|  | connection\_socket.close() |

Udp\_client.py

|  |
| --- |
| import sys |
|  | from socket import socket, AF\_INET, SOCK\_DGRAM |
|  |  |
|  | server\_name = sys.argv[1].encode() |
|  | server\_port = int(sys.argv[2]) |
|  | client\_socket = socket(AF\_INET, SOCK\_DGRAM) |
|  |  |
|  | sentence = input("\nEnter sentence: ").encode() |
|  | client\_socket.sendto(sentence, (server\_name, server\_port)) |
|  | modified\_sentence, server\_address = client\_socket.recvfrom(2048) |
|  | print(f"{server\_address[0]} > {modified\_sentence.decode()}") |
|  | client\_socket.close() |

Udp\_server.py

|  |
| --- |
| import sys |
|  | from socket import socket, AF\_INET, SOCK\_DGRAM |
|  |  |
|  | server\_name = "10.0.4.26" |
|  | server\_port = 12000 |
|  | server\_socket = socket(AF\_INET, SOCK\_DGRAM) |
|  | server\_socket.bind((server\_name, server\_port)) |
|  |  |
|  | print(f"Server {server\_name} is ready to receive on port {server\_port}") |
|  |  |
|  | while True: |
|  | message, client\_address = server\_socket.recvfrom(2048) |
|  | message = message.upper() |
|  | server\_socket.sendto(message, client\_address) |

Web\_proxy.py

|  |
| --- |
| import os |
|  | import sys |
|  | import threading |
|  | from socket import socket, AF\_INET, SOCK\_STREAM, error |
|  |  |
|  | NUM\_REQS = 50 |
|  | BUF\_SIZE = 999999 |
|  |  |
|  |  |
|  | def proxy\_server\_thread(client\_conn, client\_addr): |
|  | request = client\_conn.recv(BUF\_SIZE) |
|  | request\_first\_line = request.decode().split("\n")[0] |
|  | url = request\_first\_line.split(" ")[1] |
|  | print("From", "\t", client\_addr[0], "\t", |
|  | "Request", "\t", request\_first\_line) |
|  |  |
|  | http\_pos = url.find("://") |
|  | if http\_pos == -1: |
|  | temp = url |
|  | else: |
|  | temp = url[(http\_pos + 3):] |
|  |  |
|  | port\_pos = temp.find(":") |
|  |  |
|  | webserver\_pos = temp.find("/") |
|  | if webserver\_pos == -1: |
|  | webserver\_pos = len(temp) |
|  |  |
|  | webserver = "" |
|  | port = -1 |
|  | if port\_pos == -1 or webserver\_pos < port\_pos: |
|  | port = 80 |
|  | webserver = temp[:webserver\_pos] |
|  | else: |
|  | port = int((temp[(port\_pos + 1):])[: webserver\_pos - port\_pos - 1]) |
|  | webserver = temp[:port\_pos] |
|  |  |
|  | try: |
|  | s = socket(AF\_INET, SOCK\_STREAM) |
|  | s.connect((webserver, port)) |
|  | s.send(request) |
|  | while 1: |
|  | response = s.recv(BUF\_SIZE) |
|  | response\_first\_line = response.decode( |
|  | "utf8", "ignore").partition("\n")[0] |
|  | print( |
|  | "To", "\t", client\_addr[0], "\t", "Response", "\t", response\_first\_line |
|  | ) |
|  | if len(response) > 0: |
|  | client\_conn.send(response) |
|  | else: |
|  | break |
|  | s.close() |
|  | client\_conn.close() |
|  | except error: |
|  | if s: |
|  | s.close() |
|  | if client\_conn: |
|  | client\_conn.close() |
|  | print(client\_addr[0], "\t", "Peer reset", "\t", request\_first\_line) |
|  | sys.exit(1) |
|  |  |
|  |  |
|  | def proxy\_server(): |
|  | if len(sys.argv) < 2: |
|  | print("Using Default port 8080 since no port was mentioned.") |
|  | port = 8080 |
|  | else: |
|  | port = int(sys.argv[1]) |
|  | host = "" |
|  | print("Proxy server 10.0.4.26 Running on localhost :", port) |
|  | try: |
|  | s = socket(AF\_INET, SOCK\_STREAM) |
|  | s.bind((host, port)) |
|  | s.listen(NUM\_REQS) |
|  | except error: |
|  | if s: |
|  | s.close() |
|  | print("Could not open socket:") |
|  | sys.exit(1) |
|  | while 1: |
|  | client\_conn, client\_addr = s.accept() |
|  | threading.\_start\_new\_thread( |
|  | proxy\_server\_thread, (client\_conn, client\_addr)) |
|  | s.close() |
|  |  |
|  |  |
|  | if \_\_name\_\_ == "\_\_main\_\_": |
|  | proxy\_server() |