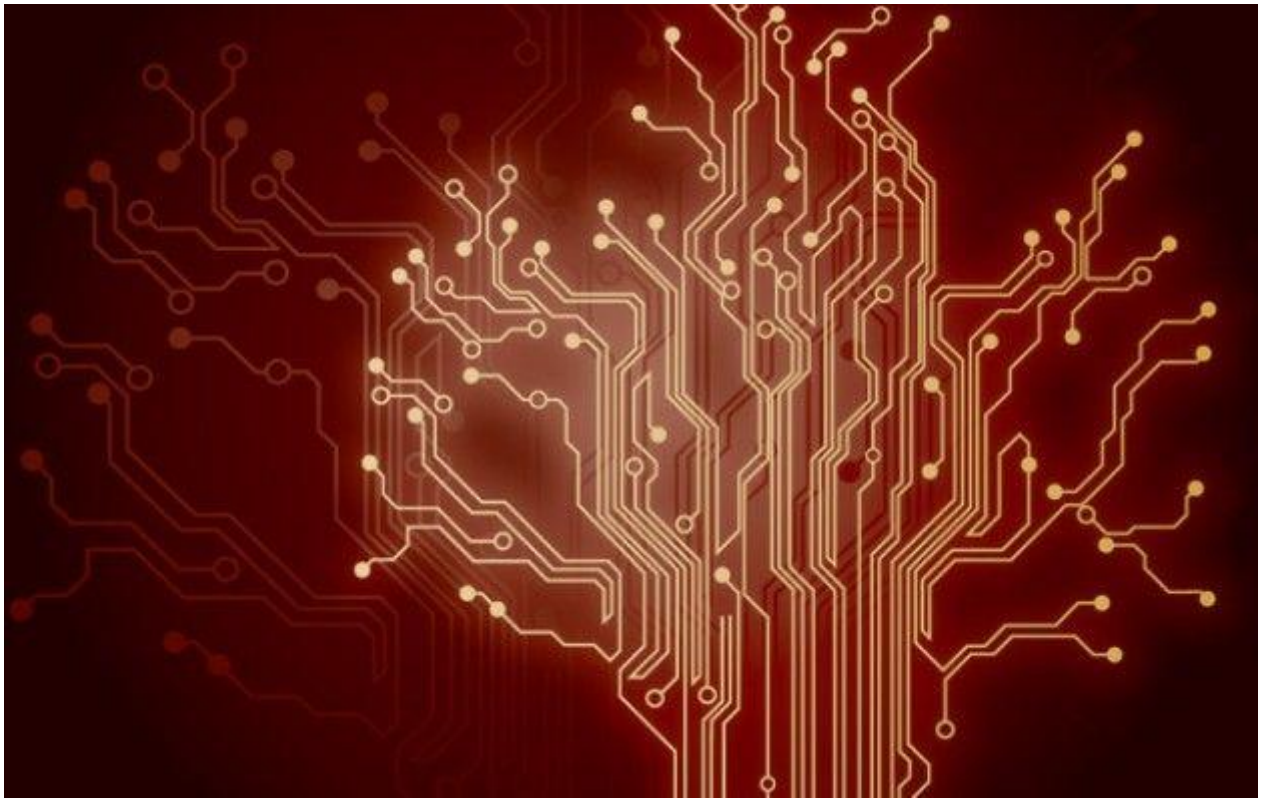
NETWORK LAB REPORT

CO7 : Implement Application Layer Protocols using TCP/UDP socket as applicable.



Atanu Ghosh

BCSE-III (2019-2023) 5th sem, Section: A-1,

Roll: **001910501005**, Date: 12/11/2021

ASSIGNMENT-7

Implement Application Layer Protocols using TCP/UDP socket as applicable.

PROTOCOLS IMPLEMENTED

- **BGP** (Border Gateway Protocol)
- **DHCP** (Dynamic Host Configuration Protocol)
- **FTP** (File Transfer Protocol)

.

DESIGN

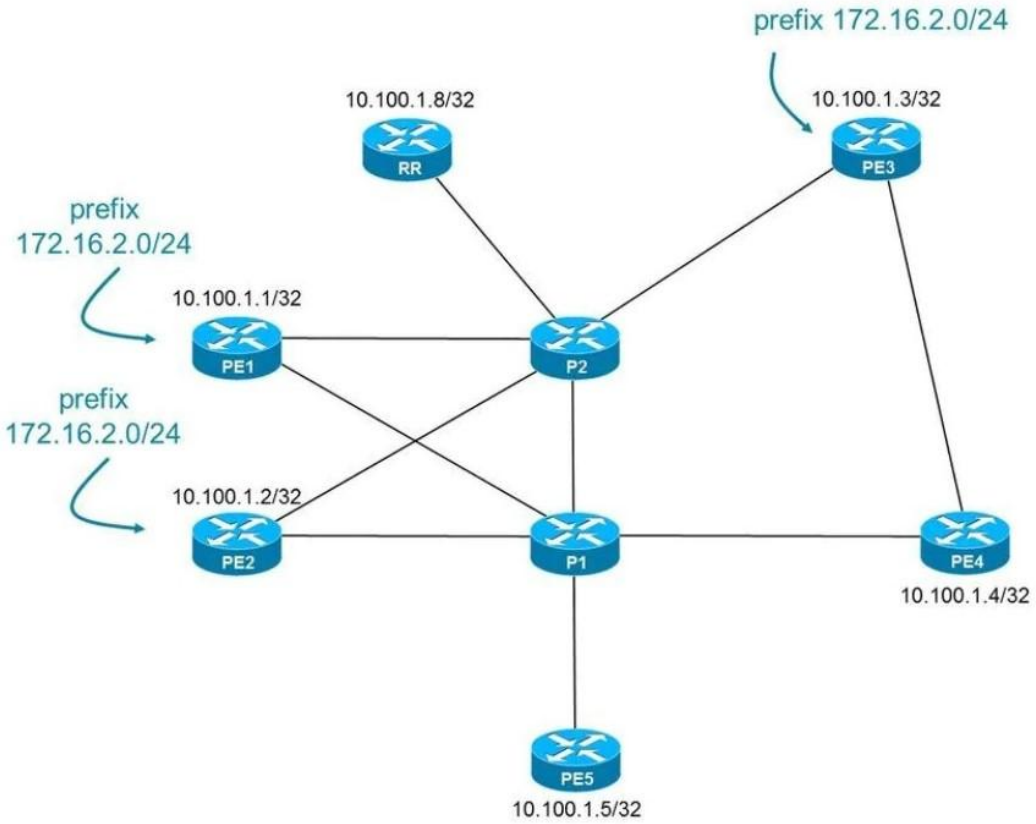


Fig : Schematic Diagram of a BGP Server

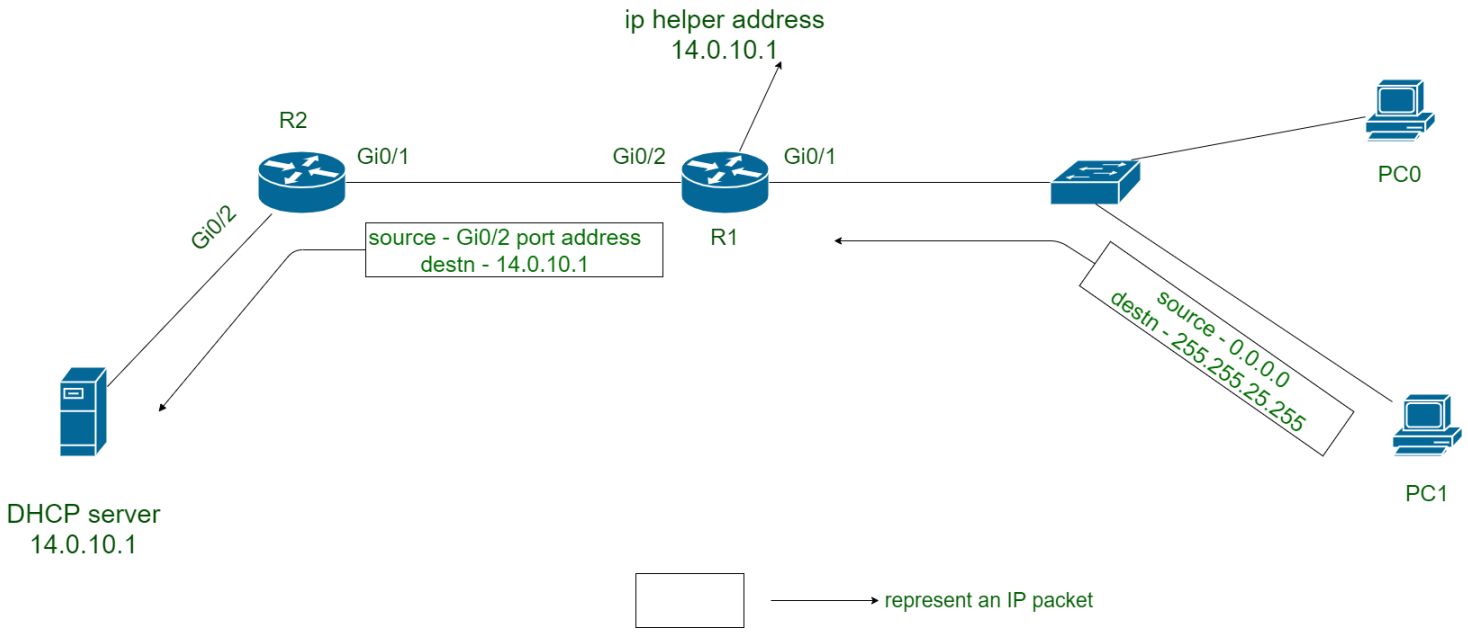


Fig : Schematic Diagram of a DHCP Server (1)

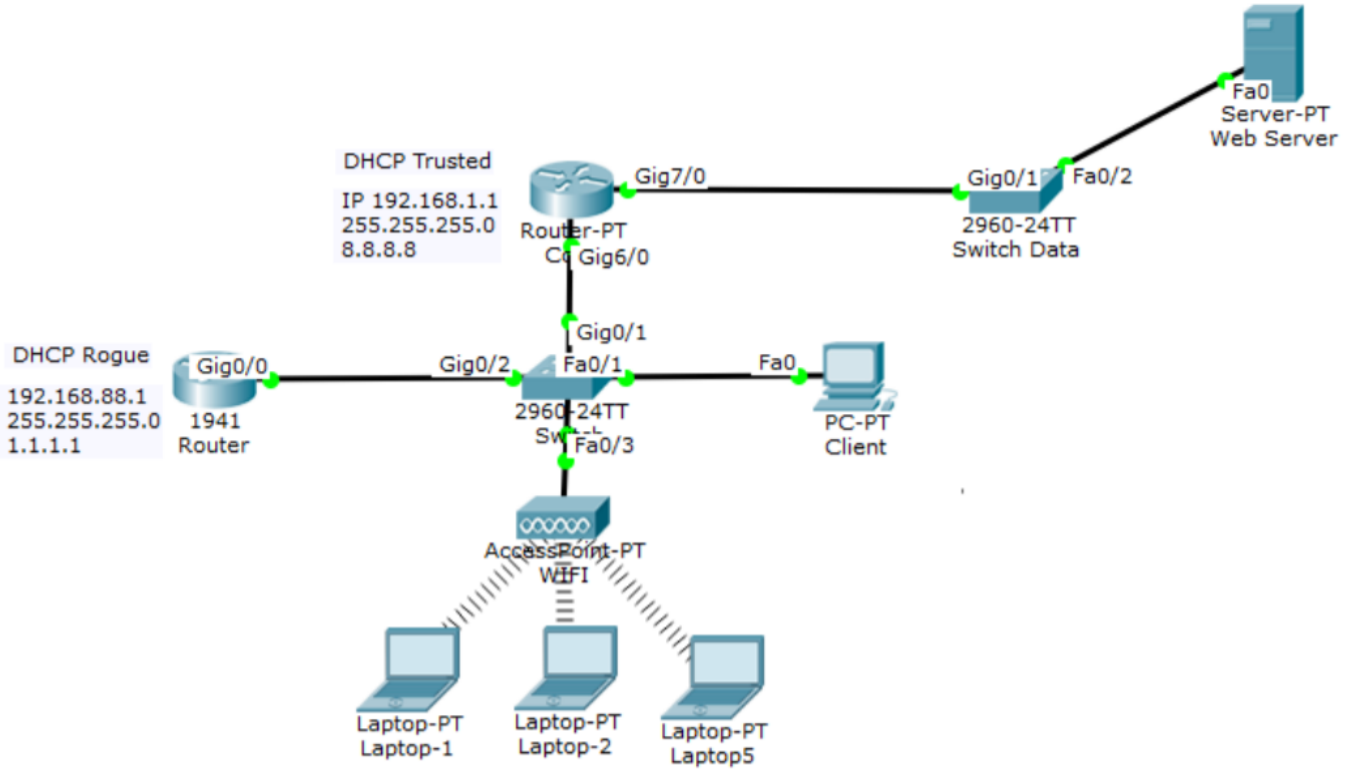


Fig : Schematic Diagram of a DHCP Server (2)

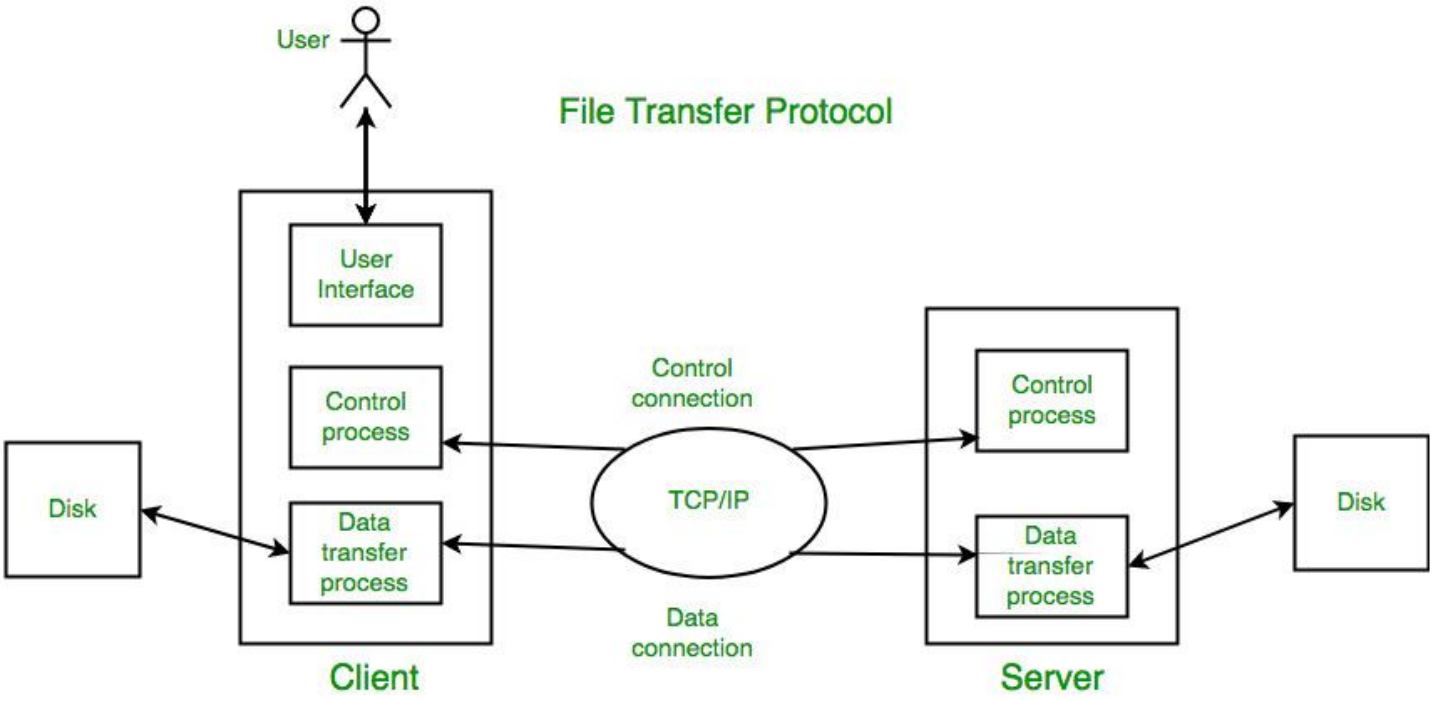


Fig : Schematic Diagram of a FTP Server

IMPLEMENTATION

bgp_server.py

```
#!/usr/bin/env python3.9

"""BGP Server implementation in python"""

import socket

HOST = "127.0.0.1"      # Standard loopback interface address (localhost)
BGP_PORT = 54400        # Port to listen on (non-privileged ports are > 1023)


class BGPServer:
    """BGP Server class"""

    def __init__(self):
        """Initialize BGP Server"""

        self.sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        self.sock.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
        self.address = ""
        self.name = ""
        self.data = ""
        self.msg = ""
        self.log = {}

    def start_bgp(self):
        """Start The BGP Server"""

        print("BGP Server started!!")
        while True:
            self.sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
            self.sock.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
            self.sock.bind((HOST, BGP_PORT))
            print("Listening for a connection on its own port....")
            self.sock.listen(1)
            conn, addr = self.sock.accept()
            self.msg = conn.recv(1024).decode("utf-8")

            if self.msg == "Request to add host to the log!":
                conn.send(bytes("Request granted! Send host name!", "utf-8"))
                self.name = conn.recv(1024).decode("utf-8")
                conn.send(bytes("Send address!", "utf-8"))
```

```
                self.address = conn.recv(1024).decode("utf-8")
                self.log[self.name] = self.address
                conn.close()
                self.sock.close()
                print("The server log now : ", end="")
                print(self.log)
                print("BGP Server is still running!")

            else:
                conn.send(bytes("Request granted! Send host name!", "utf-8"))
                self.name = conn.recv(1024).decode("utf-8")
                conn.send(bytes("Send data to be transferred!", "utf-8"))
                self.data = conn.recv(1024).decode("utf-8")
                conn.close()
                self.sock.close()
                self.sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
                self.sock.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
                addr = self.log[self.name]
                self.sock.connect((HOST, int(addr[13:-2])))
                self.sock.send(bytes("Requesting data transfer by {}".format(self.name), "utf-8"))
                self.msg = self.sock.recv(1024).decode("utf-8")
                print("Message received : {}".format(self.msg))
                self.sock.send(bytes(self.data, "utf-8"))
                self.sock.close()
                print("BGP Server still running!")


if __name__ == "__main__":
    bgp_server = BGPServer()
    bgp_server.start_bgp()
```

dhcp_server.py

```
#!/usr/bin/env python3.9

"""DHCP Server implementation in python"""

import socket

HOST = "127.0.0.1"      # Standard loopback interface address (localhost)
DHCP_PORT = 54100       # Port to listen on (non-privileged ports are > 1023)
BGP_PORT = 54400
```

```
class DHCPServer:
    """DHCP Server class"""

    def __init__(self):
        """Initialize DHCP Server"""

        self.sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        self.sock.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
        self.name = ""
        self.msg = ""

    def start_dhcp(self):
        """Start The DHCP Server"""

        print("DHCP Server started!!")
        while True:
            self.sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
            self.sock.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
            self.sock.bind((HOST, DHCP_PORT))
            print("Listening for a connection on its own port....")
            self.sock.listen(5)
            conn, addr = self.sock.accept()
            print("Connection by : {}".format(addr))
            self.msg = conn.recv(1024).decode("utf-8")
            self.name = conn.recv(1024).decode("utf-8")
            print("{} received from {} named {}".format(self.msg, addr, self.name))
            conn.send(bytes(str(addr), "utf-8"))
            conn.close()
            self.sock.close()
            self.sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
            self.sock.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
            self.sock.connect((HOST, BGP_PORT))
            self.sock.send(bytes("Request to add host to the log!", "utf-8"))
            self.msg = self.sock.recv(1024).decode("utf-8")
            print("Message received : {}".format(self.msg))
            self.sock.send(bytes(self.name, "utf-8"))
            self.msg = self.sock.recv(1024).decode("utf-8")
            print("Message received : {}".format(self.msg))
            self.sock.send(bytes(str(addr), "utf-8"))
            print("{} added to the log!".format(self.name))
            print("DHCP Server still running!")


if __name__ == "__main__":
    dhcp_server = DHCPServer()
    dhcp_server.start_dhcp()
```

ftp_server.py

```
#!/usr/bin/env python3.9

"""FTP Server implementation in python"""
import socket

HOST = "127.0.0.1"      # Standard loopback interface address (localhost)
FTP_PORT = 54200        # Port to listen on (non-privileged ports are > 1023)

class FTPServer:
    """FTP Server class"""

    def __init__(self):
        """Initialize FTP Server"""
        self.sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        self.sock.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
        self.file_name = ""
        self.name = ""
        self.data = ""

    def start_ftp(self):
        """Start The FTP Server"""
        print("FTP Server started!!")
        while True:
            self.sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
            self.sock.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
            self.sock.bind((HOST, FTP_PORT))
            print("Listening for a connection on its own port....")
            self.sock.listen(5)
            conn, addr = self.sock.accept()
            self.name = conn.recv(1024).decode("utf-8")
            self.file_name = conn.recv(1024).decode("utf-8")
            print("{} with address {}'s requesting file {}".format(self.name, addr, self.file_name))
            with open(self.file_name, "r", encoding="utf-8") as fptr:
                self.data = fptr.read()
            conn.send(bytes(self.data, "utf-8"))
            print("File sent successfully")
            conn.close()
            self.sock.close()
            print("FTP Server still running!")


if __name__ == "__main__":
    ftp_server = FTPServer()
    ftp_server.start_ftp()
```

host.py

```
#!/usr/bin/env python3.9

"""Host for connecting to the servers"""

import socket


HOST = "127.0.0.1"      # Standard loopback interface address (localhost)
BGP_PORT = 54400        # Port to listen on (non-privileged ports are > 1023)
DHCP_PORT = 54100
FTP_PORT = 54200


class Host:
    """Host Class for implementing connections"""


    def __init__(self):
        "Initialize The Hosts"

        self.sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        self.sock.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
        self.own_address = ""
        self.file_name = ""
        self.receiver = ""
        self.name = ""
        self.data = ""
        self.msg = ""


    def connect_hosts(self):
        """Connects The Hosts To Specific Servers"""

        print("Host started!!")
        self.sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        self.sock.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
        self.sock.connect((HOST, DHCP_PORT))
        self.sock.send(bytes("Requesting port number of the host!!", "utf-8"))
        self.name = input("Enter the name of the host : ")
        self.sock.send(bytes(self.name, "utf-8"))
        self.own_address = self.sock.recv(1024).decode("utf-8")
        print("The host address : {}".format(self.own_address))
        self.sock.close()
```

```
    while True:

        print("\n")
        print("+-------------------------------------------+")
        print("|    You want to >>                         |")
        print("|    1. Request file from FTP server        |")
        print("|    2. Send a msg to another HOST          |")
        print("|    3. Receive a msg from another HOST     |")
        print("|    4. Exit!!!!                            |")
        print("+-------------------------------------------+")
        choice = input("Enter Your Choice (1/2/3/4) : ")
        print("\n")

        if choice == "1":
            self.sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
            self.sock.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
            self.sock.connect((HOST, FTP_PORT))
            self.sock.send(bytes(self.name, "utf-8"))
            self.file_name = input("Enter filename to be searched : ")
            self.sock.send(bytes(self.file_name, "utf-8"))
            self.data = self.sock.recv(1024).decode("utf-8")
            print("The contents of the file : {}".format(self.data))
            self.sock.close()
            continue

        if choice == "2":
            self.sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
            self.sock.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
            self.sock.connect((HOST, BGP_PORT))
            self.sock.send(bytes("Requesting file transfer to another host!", "utf-8"))
            self.msg = self.sock.recv(1024).decode("utf-8")
            print("Message received : {}".format(self.msg))
            self.receiver = input("Enter name of receiver : ")
            self.sock.send(bytes(self.receiver, "utf-8"))
            self.data = input("Enter data to be transferred : ")
            self.sock.send(bytes(self.data, "utf-8"))
            self.sock.close()
            continue

        if choice == "3":
            self.sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
            self.sock.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
            self.sock.bind((HOST, int(self.own_address[13:-2])))
            self.sock.listen(5)
            print("Listening for a connection on its own port....")
            conn, addr = self.sock.accept()
            print("Connection established!!")
            print("Connected to address : {}".format(addr))
```

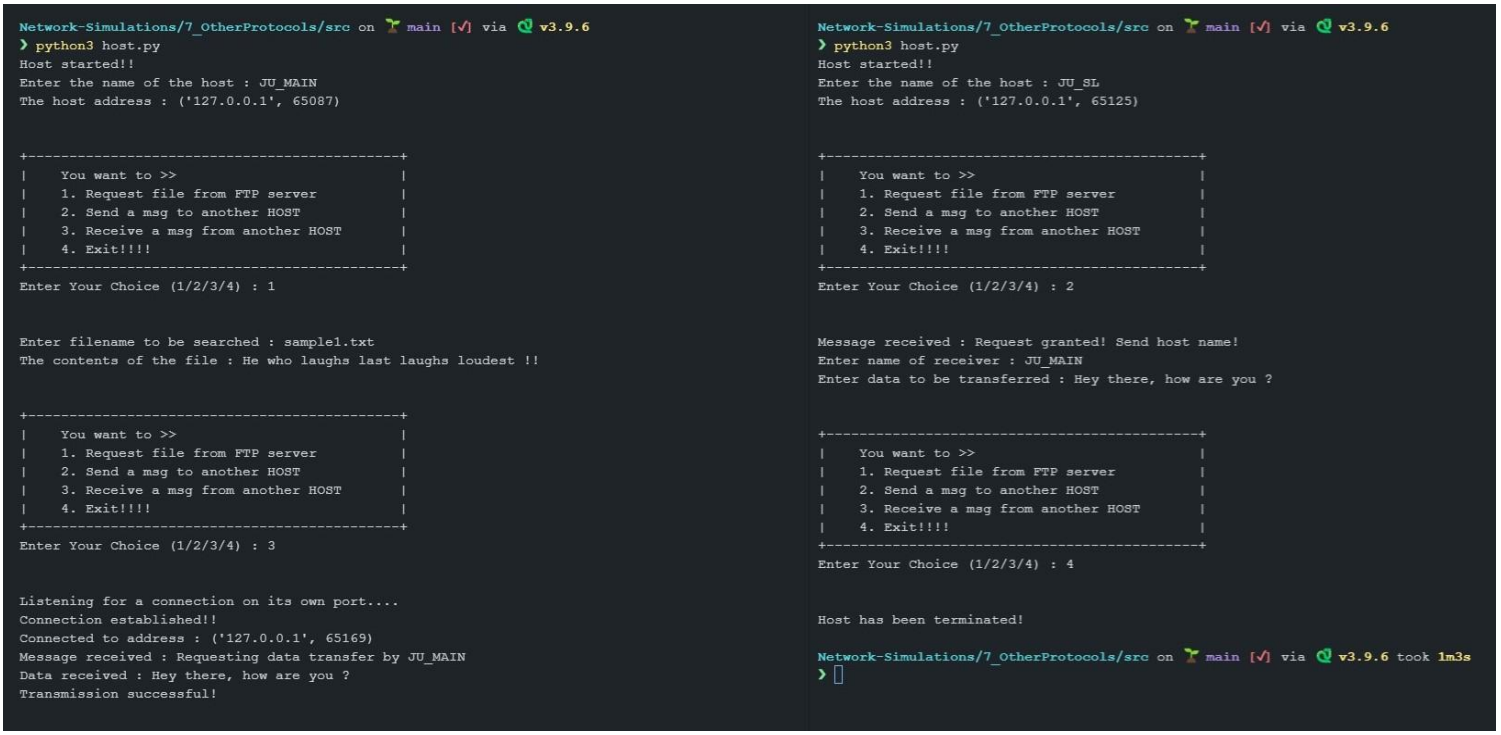
```
                self.msg = conn.recv(1024).decode("utf-8")
                print("Message received : {}".format(self.msg))
                conn.send(bytes("Request granted! Send data!", "utf-8"))
                self.data = conn.recv(1024).decode("utf-8")
                print("Data received : {}".format(self.data))
                print("Transmission successful!")
                conn.close()
                self.sock.close()
                continue

            if choice == "4":
                print("Host has been terminated!")
                break

            if choice not in ["1", "2", "3", "4"]:
                print("Invalid choice! Reselect 1/2/3/4.")
                continue


if __name__ == "__main__":
    host = Host()
    host.connect_hosts()
```

RESULTS (SCREENSHOTS)

```
Network-Simulations/7_OtherProtocols/src on  main [✓] via  v3.9.6
❯ python3 bgp_server.py
BGP Server started!!
Listening for a connection on its own port....
The server log now : {'JU_MAIN': "('127.0.0.1', 65087)"}
BGP Server is still running!
Listening for a connection on its own port....
The server log now : {'JU_MAIN': "('127.0.0.1', 65087)", 'JU_SL': "('127.0.0.1', 65125)"}
BGP Server is still running!
Listening for a connection on its own port....
Message received : Request granted! Send data!
BGP Server still running!
Listening for a connection on its own port....
```

```
Network-Simulations/7_OtherProtocols/src on  main [✓] via  v3.9.6
❯ python3 dhcp_server.py
DHCP Server started!!
Listening for a connection on its own port....
Connection by : ('127.0.0.1', 65087)
Requesting port number of the host!! received from ('127.0.0.1', 65087) named JU_MAIN
Message received : Request granted! Send host name!
Message received : Send address!
JU_MAIN added to the log!
DHCP Server still running!
Listening for a connection on its own port....
Connection by : ('127.0.0.1', 65125)
Requesting port number of the host!! received from ('127.0.0.1', 65125) named JU_SL
Message received : Request granted! Send host name!
Message received : Send address!
JU_SL added to the log!
DHCP Server still running!
Listening for a connection on its own port....
```

```
Network-Simulations/7_OtherProtocols/src on  main [✓] via  v3.9.6
❯ python3 ftp_server.py
FTP Server started!!
Listening for a connection on its own port....
JU_MAIN with address ('127.0.0.1', 65108)'s requesting file sample1.txt
File sent successfully
FTP Server still running!
Listening for a connection on its own port....
```



COMMENTS

This assignment has helped me in understanding different networking protocols of application layer by researching and implementing them. It has also helped in understanding the demerits of this protocol, and how such demerits are overcome.