Computer Networking Laboratory Project
Paper Code EC692
Academic Session : 2023-2024 EVEN
Course Coordinators – Mr. Sujoy Mondal, Mr. Kalyan Biswas, and Mr. Anindya Basu

Open Port Scanner

Submitted by

Satwik Basu, University Roll- 11700321005
Debargha Biswas, University Roll- 11700321059
Somdatta Mandal, University Roll- 11700321060
Indranil Lohar, University Roll- 11700321057
Piyush Prasad, University Roll- 11700321105

1. Introduction

This project extends a basic port scanner by adding functionality to send a custom message to identified open ports on a target host. This allows for basic interaction with open ports, but caution is necessary due to security and service compatibility concerns.

2. Application

- Network security assessments (with proper permission)
- Identifying and potentially interacting with specific services running on open ports (limited functionality)

3. Implementation steps

Imported libraries: The code imports the socket library for network communication.

1. scan_port function:
   - Creates a socket connection and sets a timeout.
   - Attempts to connect to the target host and port.
   - If successful (result == 0), prints that the port is open and returns True.
   - If unsuccessful, prints that the port is closed and returns False.
   - Handles socket errors and prints them.
2. send_message function (New):
   - Creates a socket connection and sets a timeout.
   - Connects to the target host and specified open port.
   - Encodes the message and sends it to the open port.
   - Prints a confirmation message upon successful sending.
   - Handles socket errors and prints them.
3. start_scan_and_send function:
   - Takes user input for target host, starting port, ending port, and message to send.
   - Iterates through ports in the specified range.
   - Calls scan_port to check if the port is open.

- If the port is open, calls send_message to send the message to that port.

4. Technology used

- Python programming language
- Socket library for network communication

5. Requirements

- Python 3.x installed
- No external libraries required (apart from socket library)

6. Conclusion

This extended port scanner demonstrates the ability to send messages to open ports. Remember to use it responsibly and ethically, considering potential security implications and service compatibility.

7. Source code

```
#extended_port_scanner.py
import socket

def scan_port(host, port):
    try:
        sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        sock.settimeout(1)  # Timeout after 1 second
        result = sock.connect_ex((host, port))
        if result == 0:
            print(f"Port {port} is open")
            return True
        else:
            print(f"Port {port} is closed")
            return False
        sock.close()
    except socket.error as e:
        print(f"Error: {e}")
        return False

def send_message(host, port, message):
    try:
        sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        sock.settimeout(1)  # Timeout after 1 second
```

```
        sock.connect((host, port))
        sock.sendall(message.encode())
        print(f"Message sent to port {port}")
        sock.close()
    except socket.error as e:
        print(f"Error: {e}")

def start_scan_and_send(host, start_port, end_port, message):
    print(f"Scanning ports {start_port} to {end_port} on host {host}...")
    for port in range(start_port, end_port + 1):
        if scan_port(host, port):
            send_message(host, port, message)

if __name__ == "__main__":
    HOST = input("Enter target host IP: ")  # Change this to the target host's IP address
    START_PORT = int(input("Enter starting port: "))  # Specify the starting port for scanning
    END_PORT = int(input("Enter ending port: "))  # Specify the ending port for scanning
    MESSAGE = input("Enter message to send: ")  # Specify the message to send
    start_scan_and_send(HOST, START_PORT, END_PORT, MESSAGE)
```

References

[1] **Python Documentation:** socket — Low-level networking interface: Official documentation for Python's socket module, which provides low-level networking interface.

[2] **Port Scanning**: Port scanning - Wikipedia: Wikipedia page on port scanning, explaining the concept and techniques used in port scanning.

[3] **Socket Programming:** Socket Programming in Python (Guide): Real Python tutorial on socket programming in Python, covering both client and server-side programming.