CSC 251 ♦ NETWORK SECURITY

FINAL COURSE PROJECT: TCP PORT SCANNER
DUE 10:00 PM ON THURSDAY, 4 MAY 2023

Network Scanners and Port Scanners are essential tools while trying to understand the layout of a network and the services that a specific host is running. They are often used for network diagnostics, but also as a precursor to launching an attack, since they identify potentially vulnerable services. In this project, you are called to design Port Scanner.

[IMPORTANT] It is required that you create everything from scratch instead of relying on pre-existing port scanners such as Nmap. If you do utilize any code snippets from online sources, be sure to document them in your code.

Please adhere to the ethics considerations:

- You may wish to develop your program on your own Unix-based system (e.g., Linux, macOS, WSL on Windows 10) and scan a subnet and hosts that instructor designate.
- It is not cool to scan hosts on the Internet when you do not have permission to do so. Since port scanners are sometimes used to prepare for an attack, network administrators build tools to detect their use. Hence, by scanning a host, you may cause an alarm to be raised. Even if the target machine is not being monitored for probes, routers along the path from the scanner to the target may detect the "attack".

The required features are as follows:

- First, you port scanner must check whether the target host is alive
- If the target host is alive, then probe(scan) the host for a given set of ports using any of the following scanning modes based on user selection:
    - Note: **All modes** should be implemented, and a user should be able to select one
    - Mode 1: Normal Port Scanning (full TCP connect to the remote IP address)
        - When this mode is requested, you should also grab the banner sent by the server
    - Mode 2: TCP SYN Scanning (only send the initial SYN Packet and then send RST when client responds with SYN/ACK)
    - Mode 3: TCP FIN Scanning
- Any of the above host/port scanning methods must also be able to be done sequentially or in random order (give them as options for a user)
    - Option 1: Probe all 2^{16} TCP ports on a targeted host
        - Option 1-1: Scan the ports in order (i.e., from 0 to 65,535)
        - Option 1-2: Scan in random order (e.g., instead of first scanning port 1, then port 2, then port 3, etc., randomize the order of ports)
    - Option 2: Probe only well-known TCP ports (i.e., from 0 to 1023)
        - Option 2-1: Scan the ports in order
        - Option 2-2: Scan in random order

- For each open port, port scanner should report both the port number and the service that normally runs on that port
    - The service can be found by using the getservbyport() and socket.getservbyport() calls in C and Python, respectively
- After finishing the port scanning, your portscanner must report how long it took to conduct the command as well as the number of ports that were found to be closed/open

The design should follow:

- The command-line usage for the program should be:
    - Python3 port_scanner.py [-options] target_ip
        - For options, you can use argparse module in Python
    - Examples of options:
        - -mode [normal/syn/fin]
        - -order [order/random]
        - -ports [all/known]
- The following is a sample output:

```
Starting port scan                  at 2011-01-21 01:30 PST
Interesting ports on 172.16.48.130:
Not shown: 992 closed ports
PORT     STATE SERVICE
22/tcp   open  ssh
80/tcp   open  http
111/tcp  open  rpcbind
2049/tcp open  nfs
3306/tcp open  mysql
5000/tcp open  upnp
6000/tcp open  X11
8000/tcp open  http-alt

scan done!1 IP address (1 host up) scanned in 0.23 seconds
```

Notes:

- Use local IDEs (e.g., pycharm, vscode) instead of replit
- You may find some of following libraries/tools useful: socket, scapy
- You can work in a group. The maximum number allowed in a group is two.

Submission:

- A README.md file including:
    - A list of all files required included in your project
    - An explanation of how to run your projects
- All files necessary to run your code
- A discussion of at least one major challenge, and how you overcame it.
- If you work in a group,
    - a description of your specific contributions to the project