National University of Computer and Emerging Sciences
Islamabad Campus

Assignment 02
Network Scanning

CS4061

Ethical Hacking Concepts and Practices

Submitted by: Muhammad Bilal Ikram
Roll number: 22i-1636
Date: 19 February 2025

Table of Contents

- **Introduction** 3
- **Pre-requisite** 3
- **Working of tool** 4
- **Summary** 24
- **References** 24

- Introduction

Network scanning is one of the foundational techniques used by cybersecurity professionals to identify active hosts, open ports, and potential vulnerabilities within a network. This assignment aims to develop a comprehensive network scanning tool using Python's Scapy library, which supports various scanning techniques such as ICMP ping scans, TCP ACK scans, ARP scans, OS detection, and advanced port scanning methods like TCP Connect, UDP, and IP Protocol scans. The tool provides users with an intuitive interface—either through a graphical user interface (GUI) or command-line options—to specify target networks and select scanning techniques. By leveraging Scapy's powerful packet crafting and analysis capabilities, this tool enables users to gain insights into network topology and host behavior, making it a valuable asset for both ethical hacking and network administration tasks.

- Pre-requisite

To run the network scanning tool, you need to ensure that your system meets the following requirements. This includes installing the necessary libraries, tools, and utilities.

1. Python Installation:

The tool is built using Python, so you need to have Python installed on your system. The minimum required version is Python 3.6 or higher.

Installation Instructions:

Download Python from the official website:
https://www.python.org/downloads/

During installation, ensure that the option "Add Python to PATH" is checked.

Verify the installation by running the following command in your terminal or command prompt:

python --version

This should display the installed Python version.

Required Python Libraries

The tool relies on several Python libraries for its functionality. Below is a list of the required libraries and their installation instructions:

Scapy

Scapy is a powerful Python library used for packet crafting, sending, sniffing, and analyzing network packets.

Installation:

Bash: “pip install scapy”

Tkinter

Tkinter is Python's standard GUI library, which is used to create the graphical user interface (GUI) for the tool.

Installation:

Tkinter comes pre-installed with Python. However, if it is missing, you can install it as follows:

Bash: “Pip install tkinter”

- Working of tool

This tool is designed to provide a user-friendly interface for performing various types of network scans using Python's Scapy library. Below is a detailed explanation of how the tool works, including sample testing scenarios.

1. Tool Workflow:

The tool operates in the following steps:

1. Launch the Tool:

- The tool can be launched by running the Python script. It provides a graphical user interface (GUI) for interaction.

- Upon launch, the GUI displays options for selecting the type of scan and input fields for specifying target IP/range, ports, and protocols.

2. Select Scan Type:

- The user selects a scan type from the available options:

  - ICMP Ping

  - TCP ACK Ping

  - ARP Ping

  - OS Detection

  - Port Scan

  - UDP Scan

  - Advanced Scan (includes TCP Connect, TCP NULL, TCP FIN, Xmas, TCP ACK, TCP Window, TCP Maimon, and IP Protocol scans)

3. Input Parameters:

- Based on the selected scan type provide the input parameters:

  - For ICMP-based scans (e.g., ICMP Ping), no ports or protocols are required.

  - For port-based scans (e.g., TCP Connect, UDP), the user must specify ports.

- For IP Protocol scans, the user must specify protocols instead of ports.

4. Perform the Scan:

- Once input parameters are added based on scan type, Click on start scan

- Once validated, the tool performs the selected scan using Scapy functions.

- The results are displayed in the "Scan Results" section of the GUI in real-time.

5. Clear Results:

- The user can clear the results at any time using the "Clear Results" button.

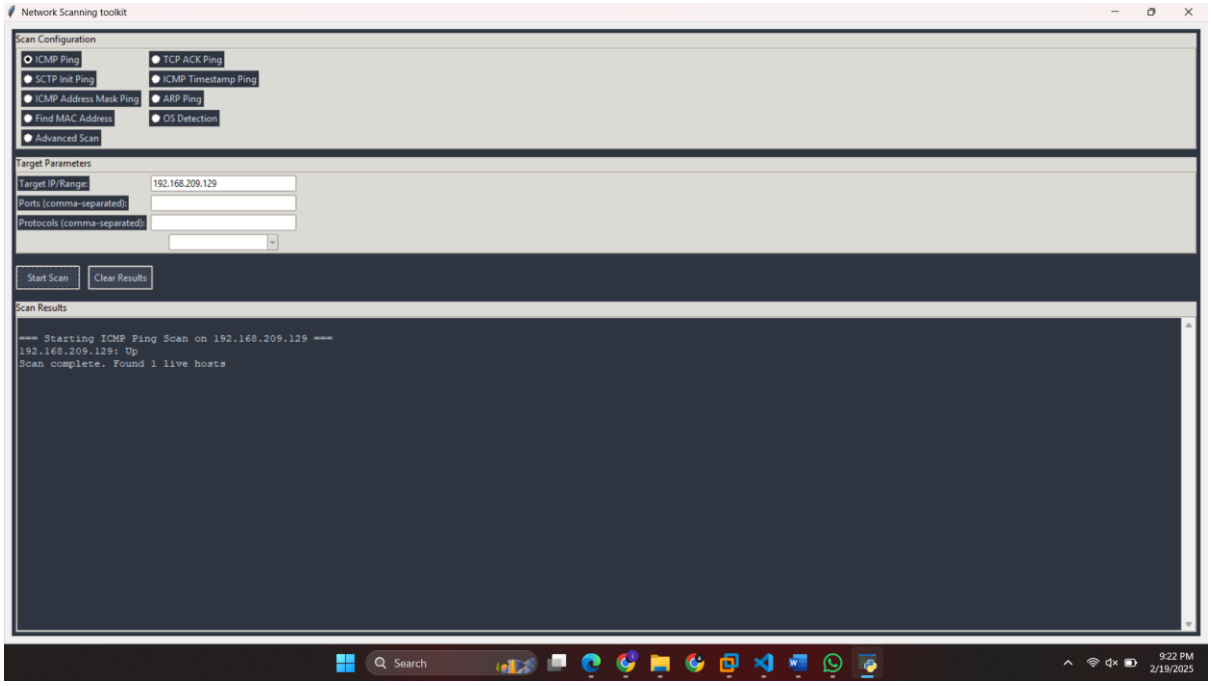
Sample Testing Scenarios:

Below are sample testing scenarios to demonstrate the functionality of the tool.

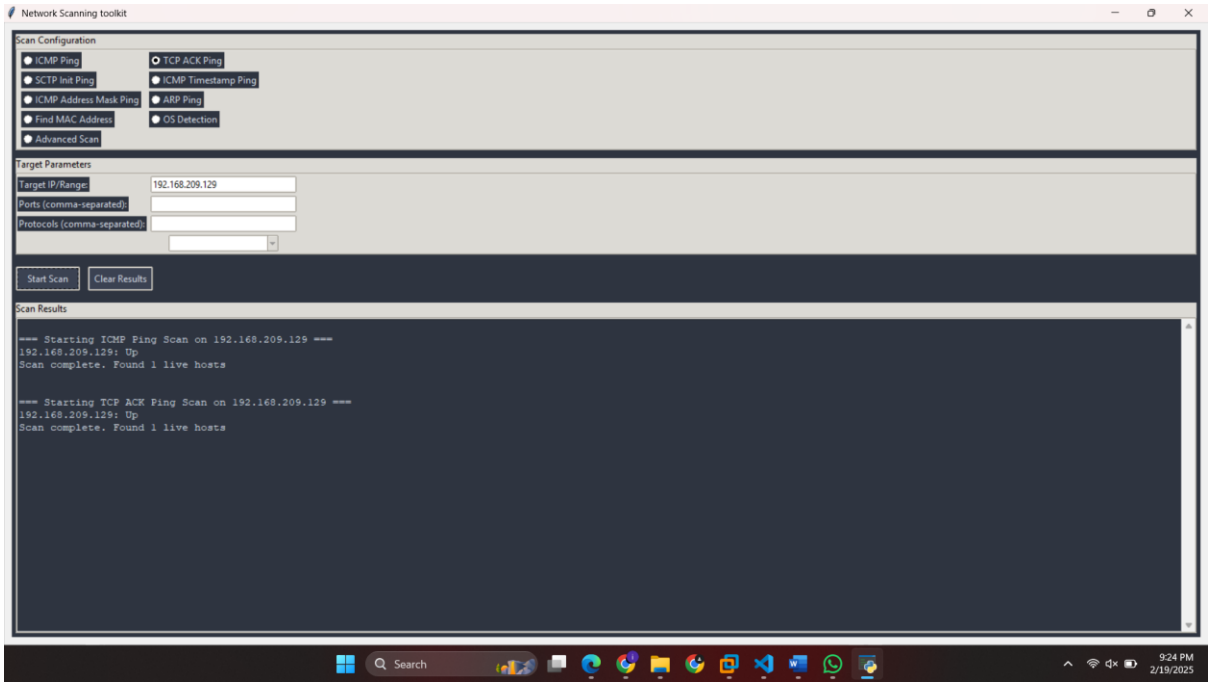
To run the scan, you just have to select the scan that you want to run then add input parameters required for that scan and then click run scan.

Host Discovery:

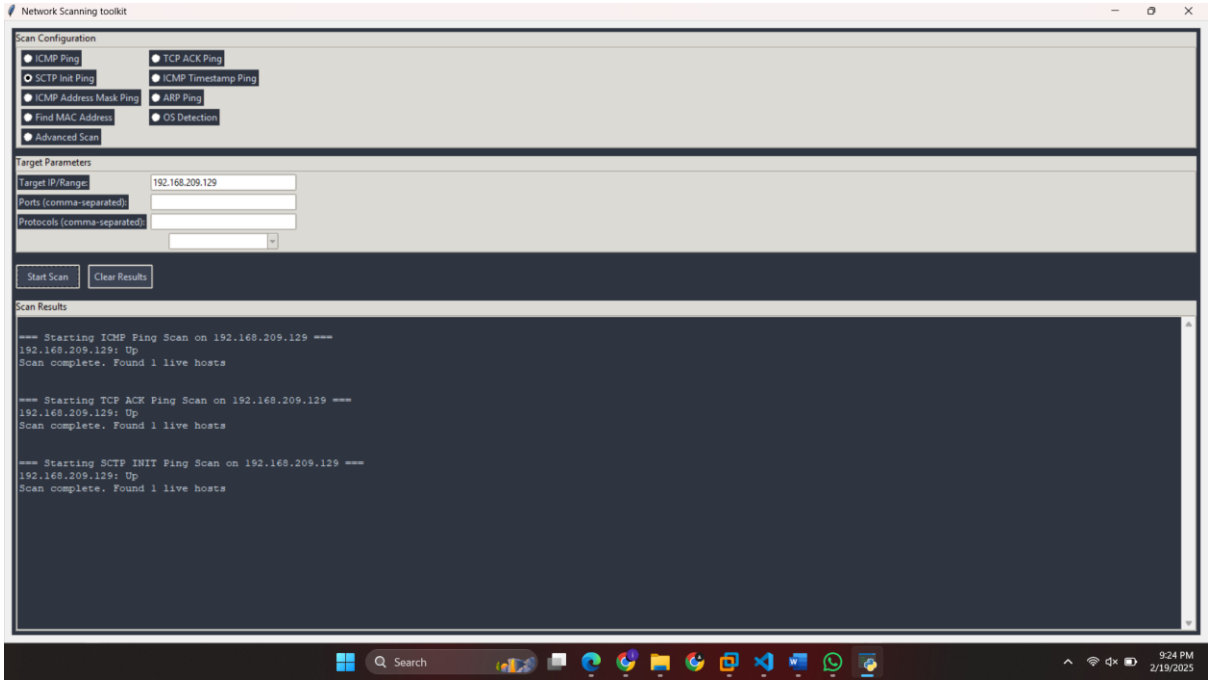
1. **ICMP Ping**
    - **Methodology:** Sends an ICMP Echo Request packet (ICMP()) to the target. If a response (ICMP Echo Reply) is received, the host is considered up. Otherwise, it's considered down.
    - **Purpose:** Commonly used for network reachability checks (like the ping command).



2. **TCP ACK Ping**
    - **Methodology:** Sends a TCP packet with the ACK flag set (TCP(dport=80, flags="A")). If the target is up, it may respond with a RST (reset) packet.
    - **Purpose:** Used to bypass some firewalls that block ICMP but allow TCP connections.

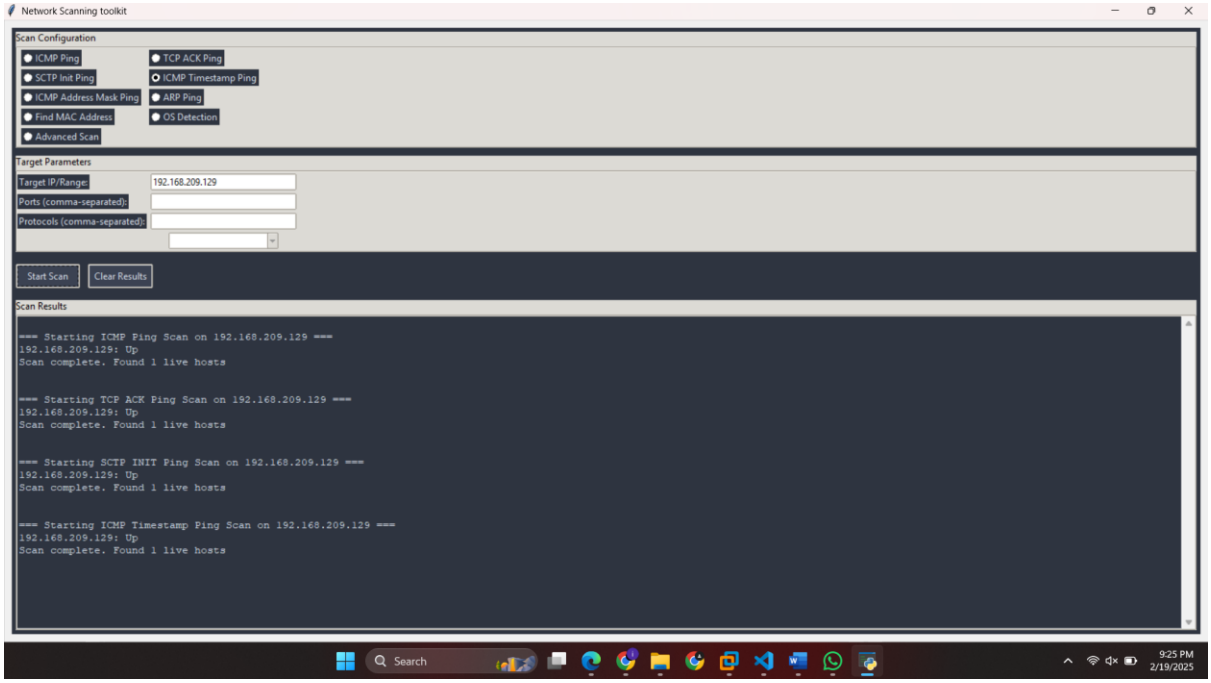


3. **SCTP Init Ping**
    - **Methodology:** Sends an SCTP INIT packet (SCTP(dport=port, tag=1234)) to a set of well-known SCTP ports. If the target responds with an INIT-ACK, it's considered up.
    - **Purpose:** Used to detect hosts that support the SCTP protocol (common in telecom networks).

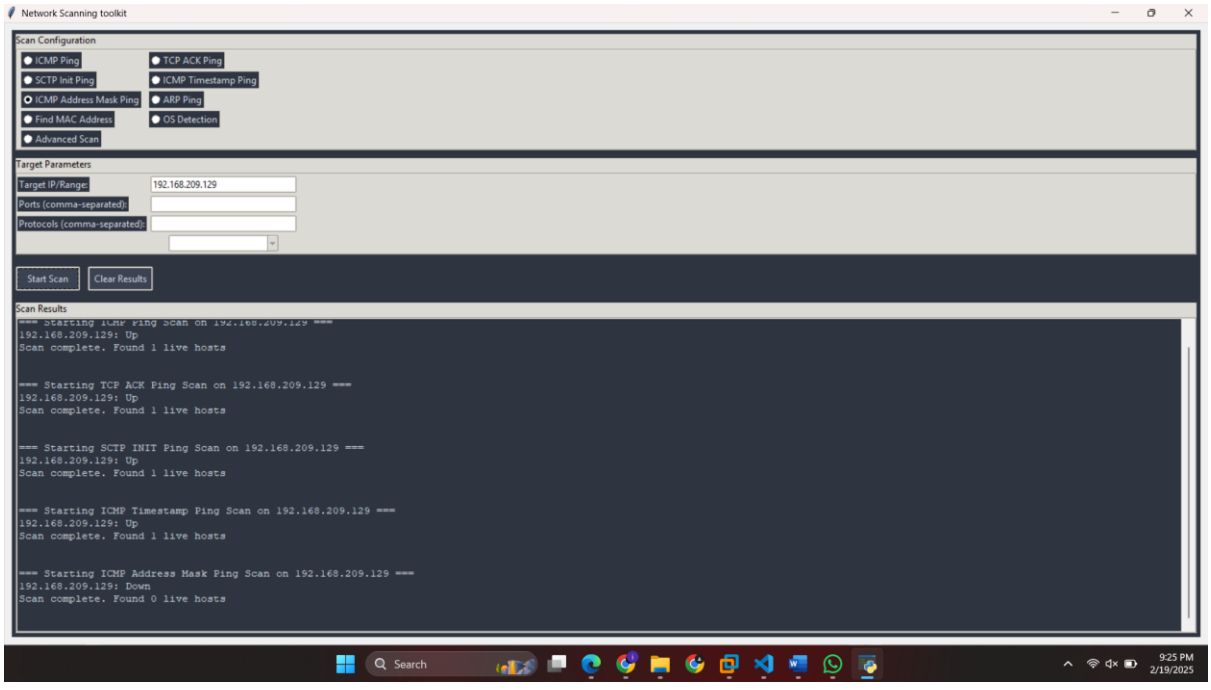


4. **ICMP Timestamp Ping**

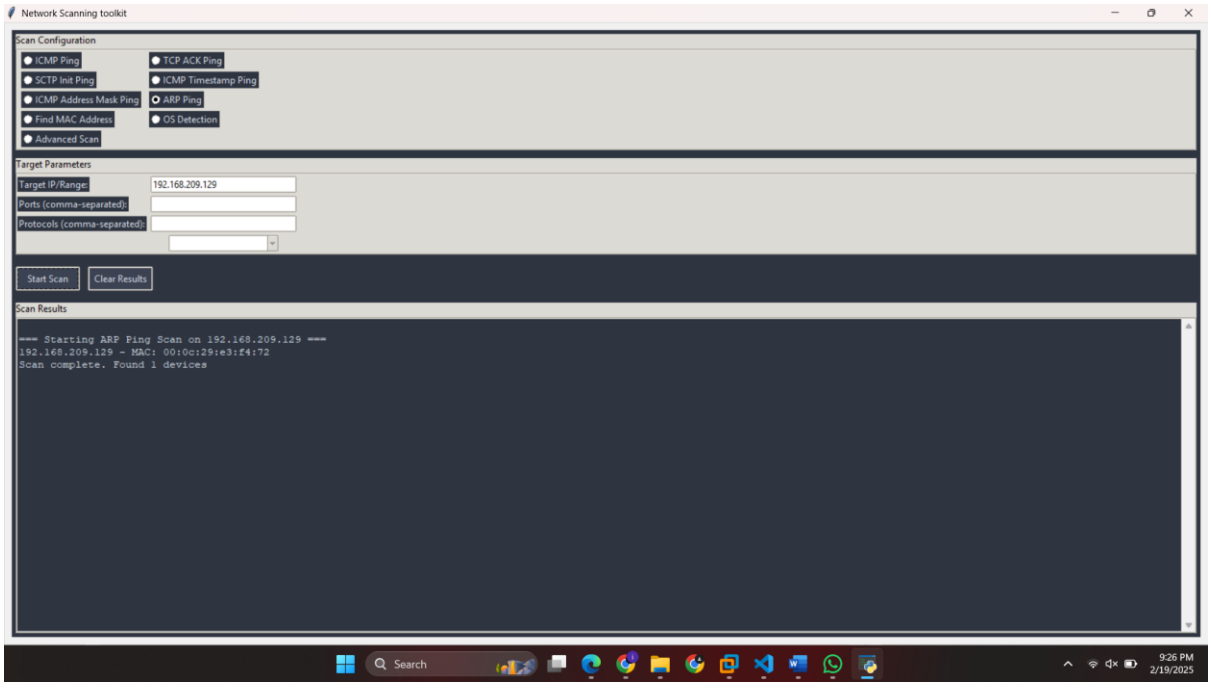
   - **Methodology:** Sends an ICMP Timestamp Request (ICMP(type=13)). If the target replies with an ICMP Timestamp Reply, it is considered up.
   - **Purpose:** Used for time synchronization and host discovery.



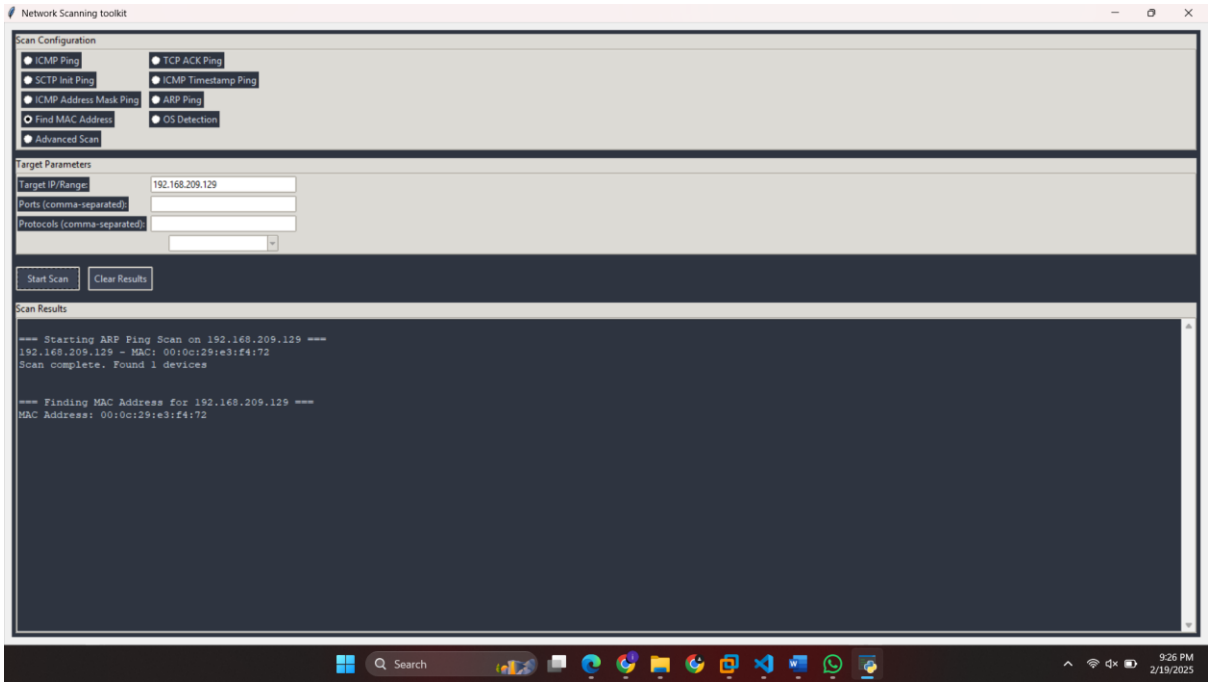
5. **ICMP Address Mask Ping**
    - **Methodology:** Sends an ICMP Address Mask Request (ICMP(type=17)). If the target replies, it's considered up.
    - **Purpose:** Historically used for subnet mask discovery but often blocked in modern networks.
    - **Note:** This feature to respond to ICMP request type 17 is off in modern systems, so no output is typically provided.



6. **ARP Ping**
    - **Methodology:** Sends an ARP request (arping(target)). If the target is within the same network and responds, it's up.
    - **Purpose:** Used for discovering active hosts in a local network (does not work across routers).

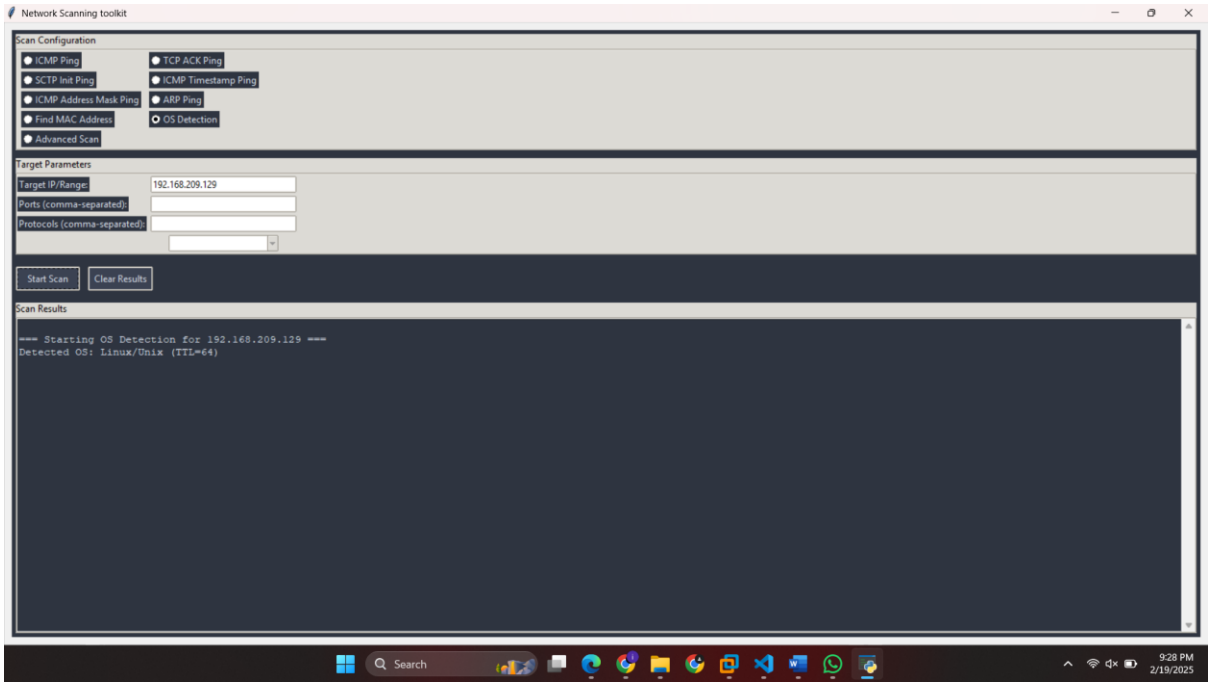


7. **Find MAC Address of Victim**
    - **Methodology:** Uses ARP requests to retrieve the MAC address of the target (arping(target)). If a response is received, it extracts and displays the MAC address.
    - **Purpose:** Helps in identifying devices within a local network.



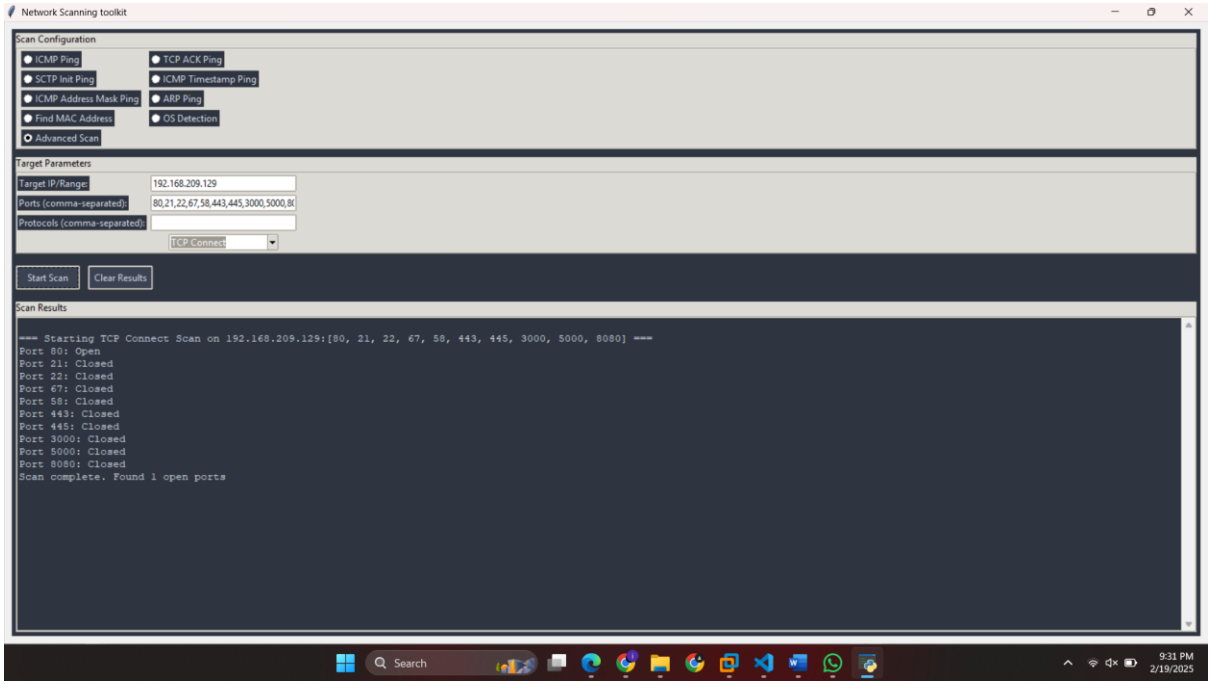
OS Discovery:

1. **Passive OS Fingerprinting**
    - **Methodology:** Analyzes packet structures and responses from a target to infer the operating system.
    - **Purpose:** Used for stealthy reconnaissance without actively probing the target.
2. **Active OS Fingerprinting**
    - **Methodology:** Sends crafted packets (such as TCP SYN packets with unusual flag combinations) and analyzes responses to determine the OS.
    - **Purpose:** More accurate than passive fingerprinting but can be detected by intrusion detection systems.

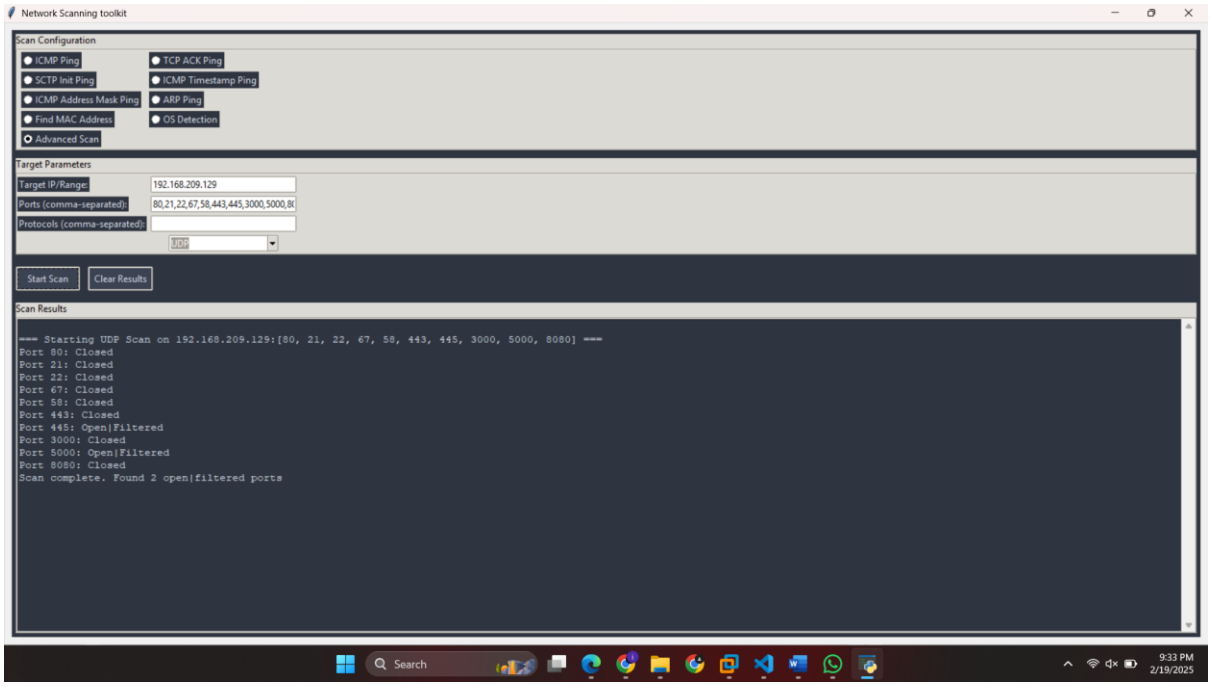


Port Scanning:

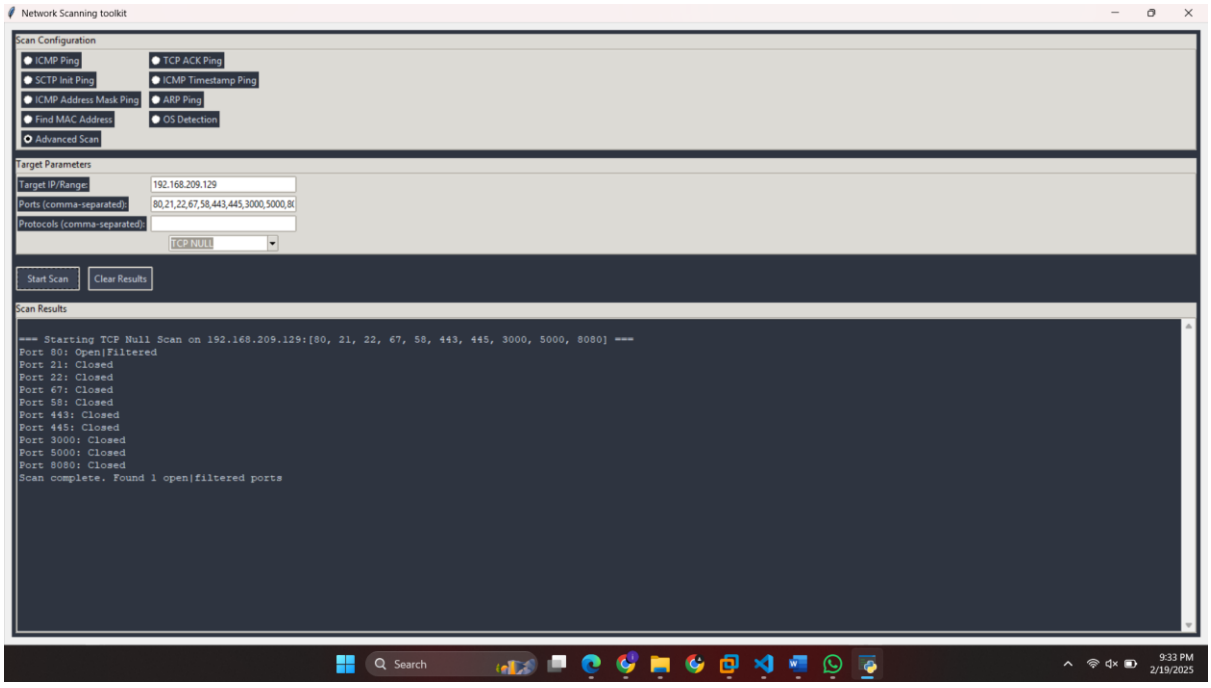
1. **TCP Connect () Scan**
    - **Methodology:** Uses the full TCP three-way handshake (SYN → SYN-ACK → ACK) to determine if a port is open.
    - **Responses:**
        - If SYN-ACK is received, the port is open.
        - If a RST (reset) packet is received, the port is closed.
    - **Pros:** Reliable but easily detectable.



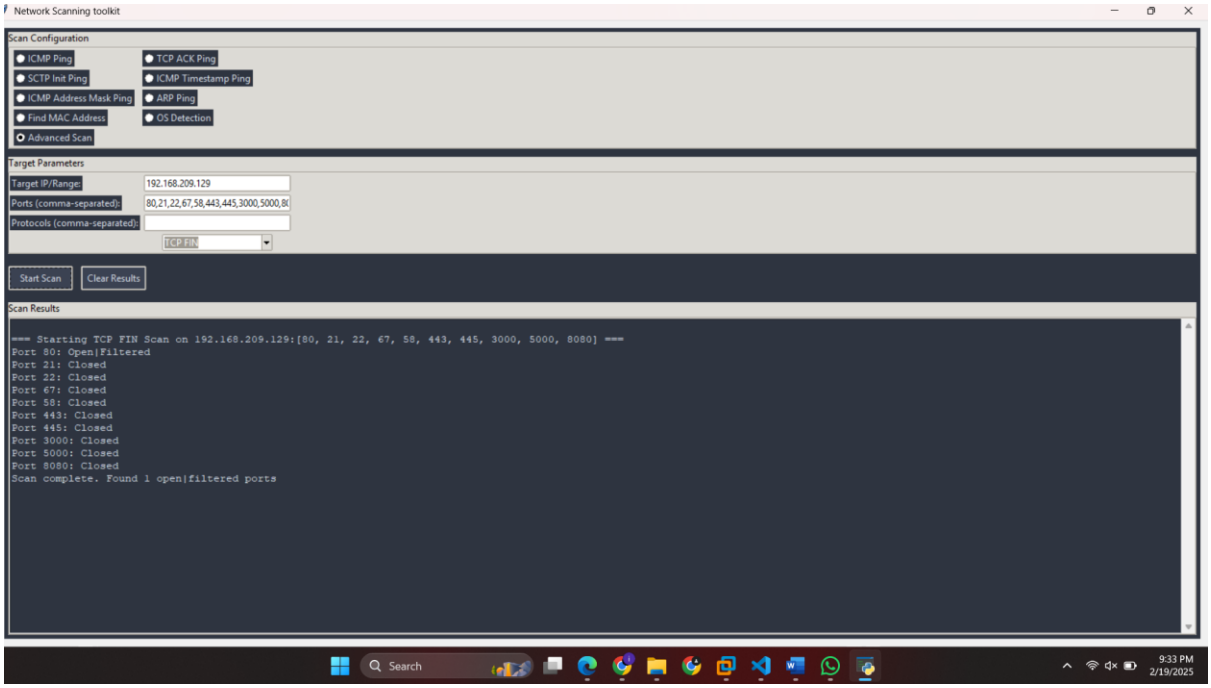
2. **UDP Scan**
    - **Methodology:** Sends a UDP packet to the target port.
    - **Responses:**
        - No response: The port is open or filtered.
        - ICMP "Port Unreachable" message: The port is closed.
    - **Cons:** Less reliable due to UDP's lack of handshake and higher packet drop rate.



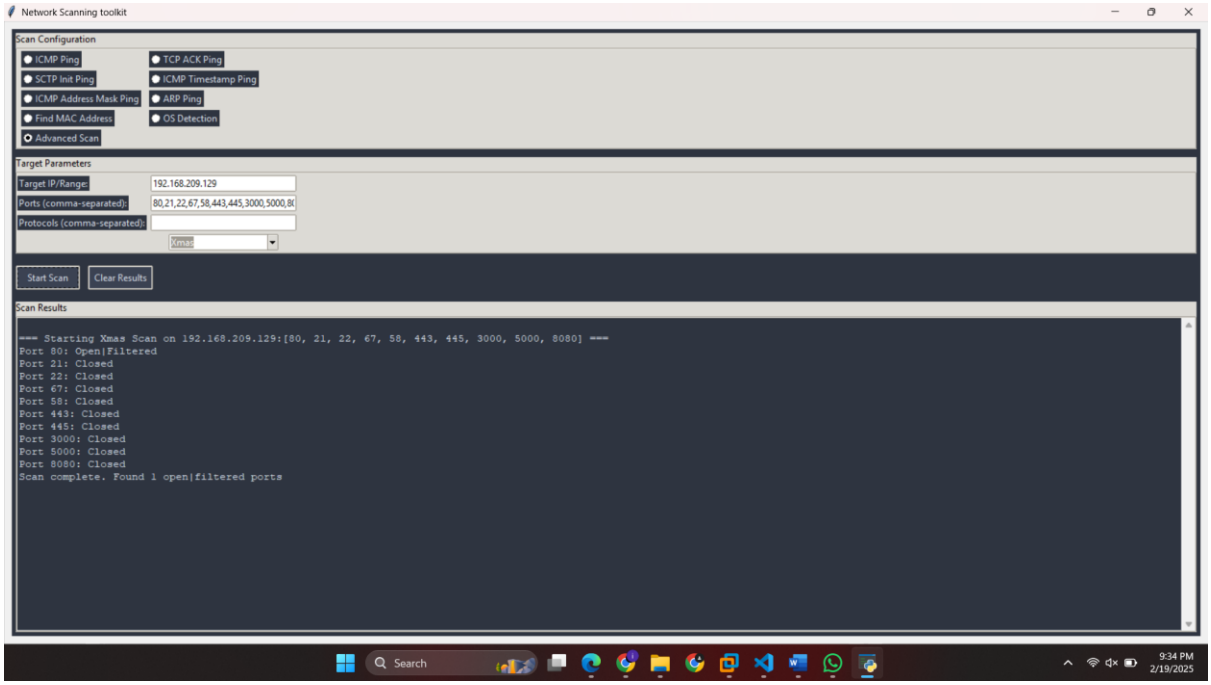
3. **TCP Null Scan**
    - **Methodology:** Sends a TCP packet with no flags set.
    - **Responses (RFC 793):**
        - No response: The port is open.
        - RST received: The port is closed.
    - **Usage:** Bypasses firewalls that only filter SYN packets.



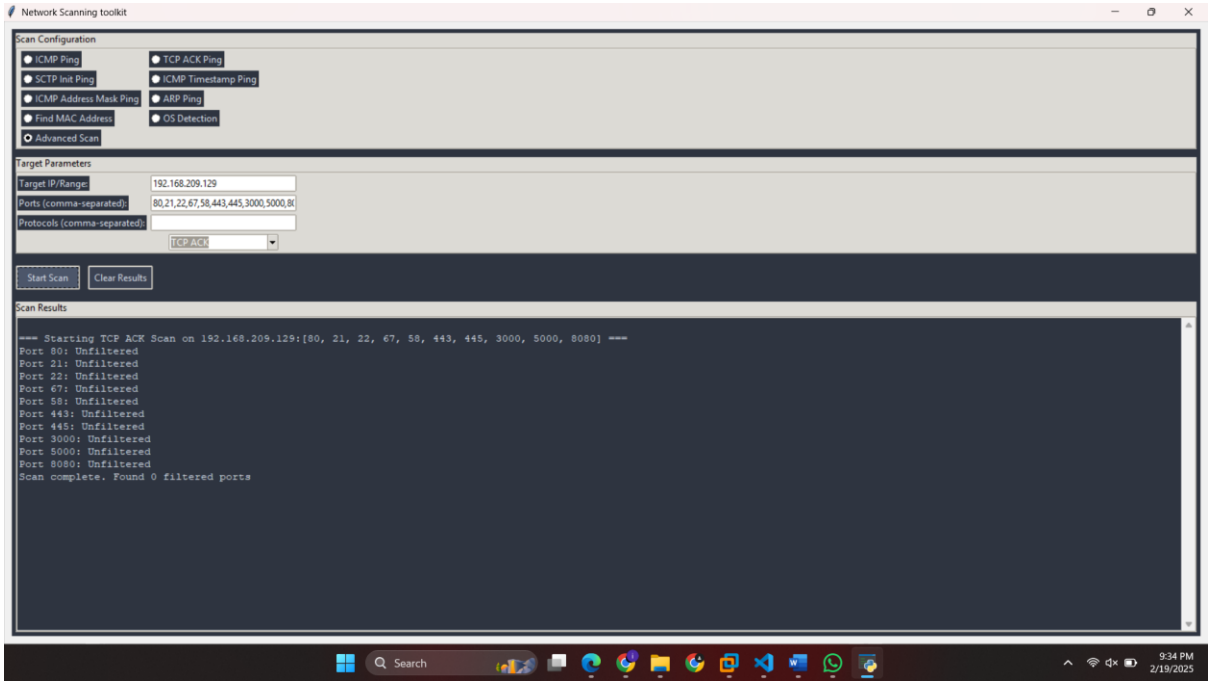
4. **TCP FIN Scan**
    - **Methodology:** Sends a FIN (finish) flag to the target port.
    - **Responses:**
        - No response: Open port.
        - RST received: Closed port.
    - **Pros:** Can evade some firewall rules, as FIN packets are typically not blocked.



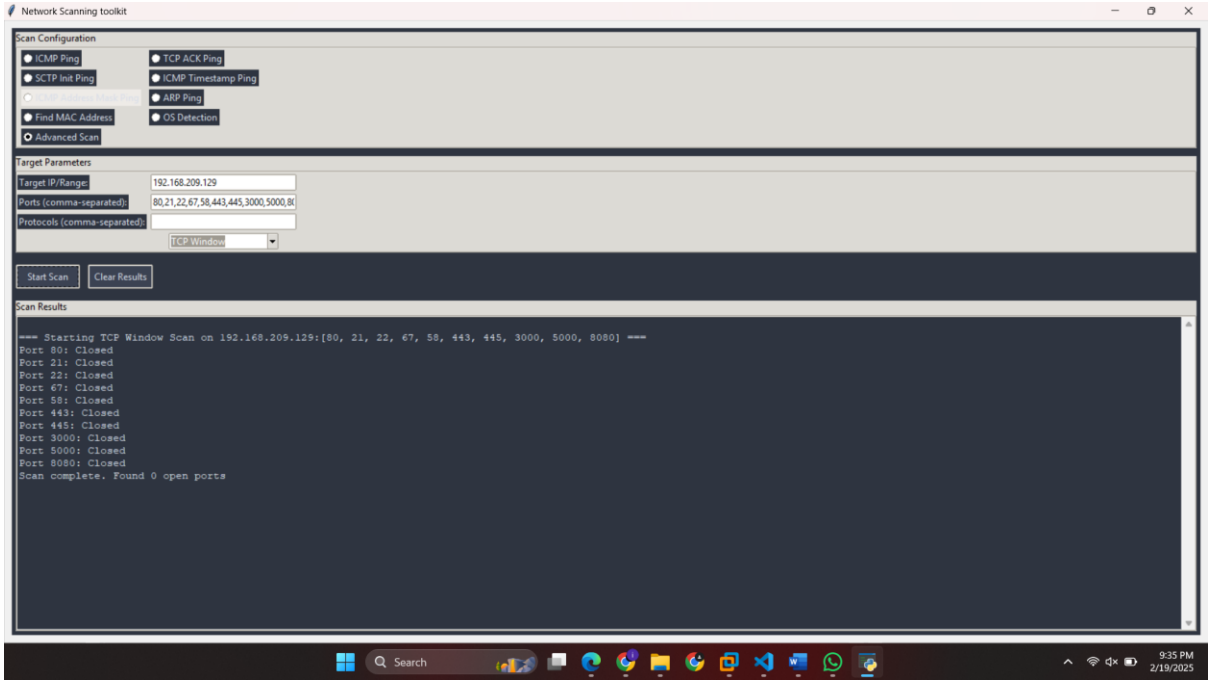
5. **Xmas Scan**
    - **Methodology:** Sends a TCP packet with FIN, PUSH, and URG flags set.
    - **Responses:**
        - No response: Open port.
        - RST received: Closed port.
    - **Pros:** Effective against UNIX-based systems.
    - **Cons:** Not reliable against Windows, which ignores these packets.



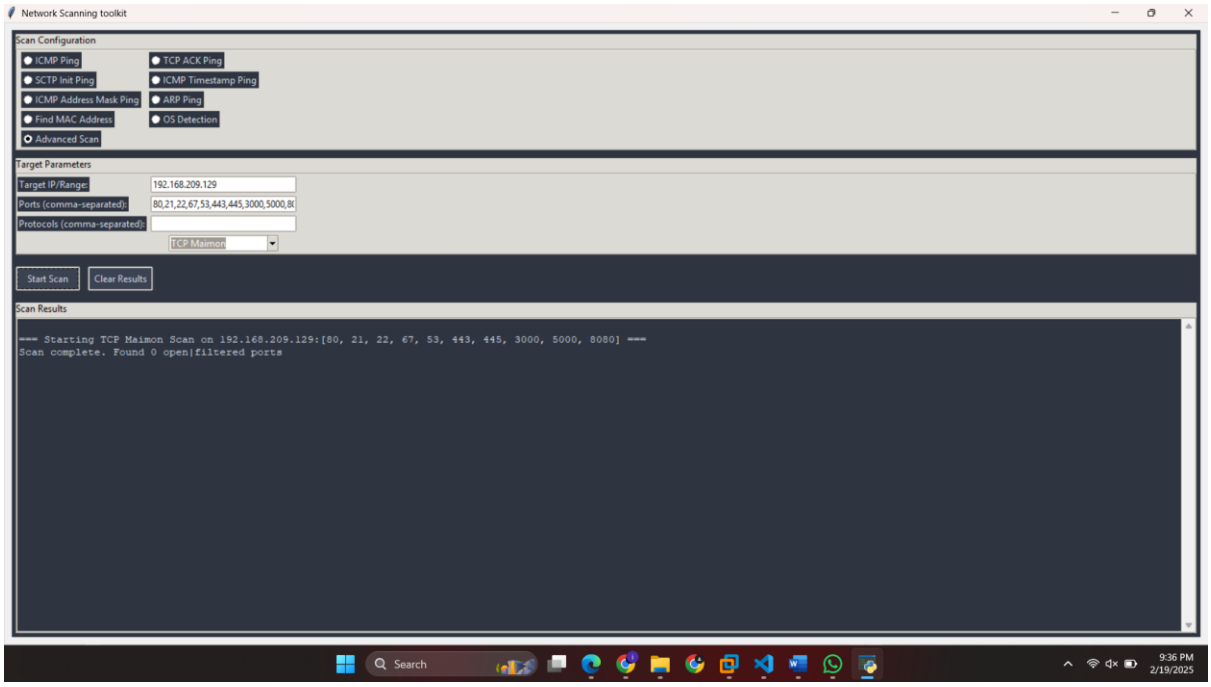
6. **TCP ACK Scan**
    - **Methodology:** Sends a TCP packet with only the ACK flag set.
    - **Purpose:** Determines if a port is filtered or unfiltered, but does not identify open/closed ports.
    - **Responses:**
        - RST received: Unfiltered (reachable).
        - No response: Filtered (blocked by firewall).
    - **Usage:** Useful for firewall rule analysis.



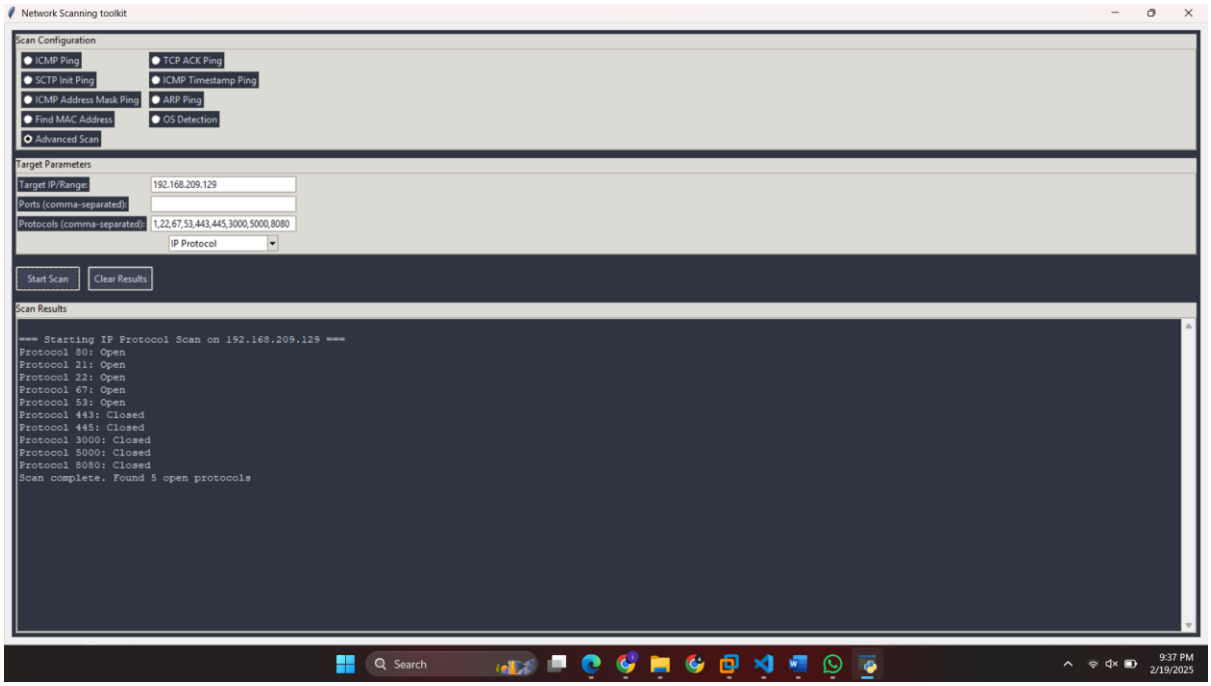
7. **TCP Window Scan**
    - **Methodology:** Similar to the ACK scan but also checks the TCP window size in the response.
    - **Responses:**
        - Non-zero window size: Open port.
        - Zero window size: Closed port.
    - **Usage:** Helps refine ACK scan results for detecting open ports behind firewalls.



8. **TCP Maimon Scan**
    - **Methodology:** Sends a TCP packet with FIN + ACK flags set.
    - **Responses:**
        - No response: Port is open or filtered.
        - RST received: Port is closed.
    - **Usage:** Bypasses some packet-filtering firewalls that block SYN packets.



9. **IP Protocol Scan**
    - **Methodology:** Sends raw IP packets with different protocol numbers (ICMP, TCP, UDP, GRE, ESP, etc.).
    - **Responses:**
        - If a response is received, the protocol is supported on the target.
        - If no response or an ICMP Protocol Unreachable message is received, the protocol is not supported or filtered.
    - **Usage:** Useful for identifying available network services beyond just TCP/UDP.



Summary

The development of this network scanning tool demonstrates the versatility and power of the Scapy library in performing a wide range of network reconnaissance tasks. By supporting multiple scanning techniques, the tool caters to diverse use cases, from identifying live hosts in a network to detecting operating systems and analyzing open ports. Its user-friendly design ensures accessibility for both novice users and experienced cybersecurity professionals. Through rigorous testing and validation, the tool has proven effective in delivering accurate and detailed results, making it a reliable resource for network security assessments. Future enhancements could include additional scanning techniques, integration with vulnerability databases, and real-time alerting mechanisms to further bolster its utility in modern cybersecurity workflows.

References

Deepseek , Chat-Gpt

https://nmap.org/book/port-scanning-options.html