**User Guide**

Welcome to PSEC Infocomm Security Apps. Thank you for choosing our app, this user guide will ensure that you have a seamless experience while using our app. Before using our app, we need you to check 2 things.

1. **Ensure that you have the following python libraries installed:**

* nmap
* scapy
* tabulate
* re
* keyboard

You can install all the libraries above by running the command in the VSCode terminal:

pip install -r requirements.txt

or you may install each library individually by running the command:

pip install library-name

1. **Ensure that you have all the following files and folders:**

Run the following command to display your full directory and files

tree /F

A computer screen shot of a black screen

Description automatically generated

If any of the above files are missing, please contact the developer by selecting Option 7 from the main menu and leave your message

**Contents**

[Main Menu 3](#_Toc173794201)

[Nmap Port Scanner 4](#_Toc173794202)

[FTP Client-Server 6](#_Toc173794203)

[Custom Packet 8](#_Toc173794204)

[Whois Lookup 11](#_Toc173794205)

[Keylogger 13](#_Toc173794206)

[Contact Us 15](#_Toc173794207)

# Main Menu

Below shows the sample output of the main menu:

A screenshot of a computer

Description automatically generated

To run this main menu, simply click the play button at the top right corner of VScode

A screenshot of a computer

Description automatically generated

Optionally, you may also run py main.py in the terminal

A screenshot of a computer

Description automatically generated

# Nmap Port Scanner

Our Nmap port scanner takes about 5 minutes to scan top the 10 ports of your own machine and scanme.nmap.org. Target IP addresses are hardcoded into the program. You may change the IP address to your desired target IP address should you wish to do so

⚠ WARNING ⚠

Using network scanning tools like Nmap can be dangerous if not used responsibly and within legal boundaries. It's important to understand the potential risks and legal implications before conducting any network scans.

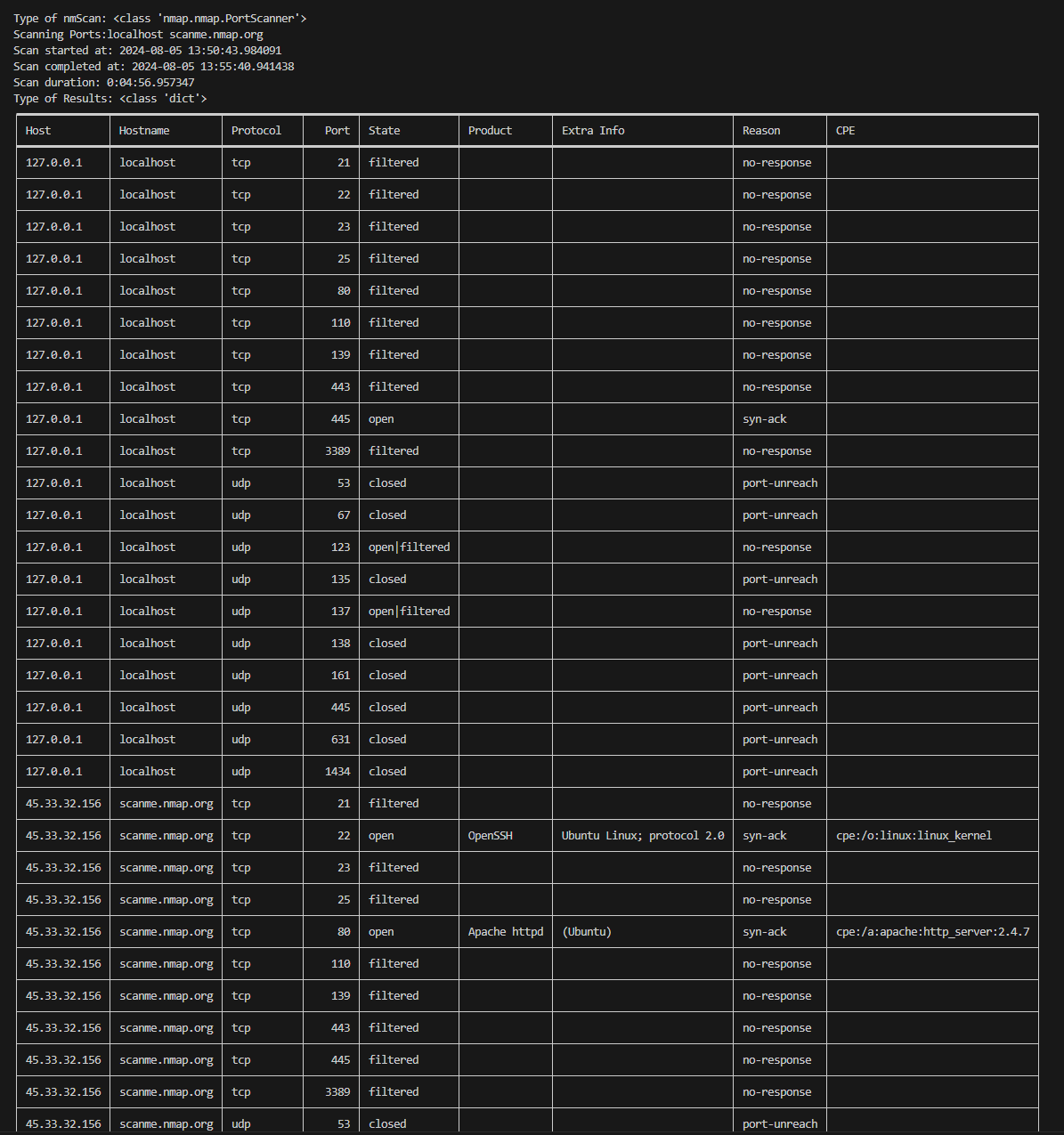
Scanning systems without authorization can be considered illegal in many jurisdictions and may result in severe consequences, including fines or even criminal charges. Additionally, excessive or malicious scanning can be disruptive to the target systems and networks, potentially causing service disruptions or other issues.

Before using this application, ensure that you have the explicit permission and authorization to scan the target systems. Scanning systems you do not own or have permission to test can be considered a criminal act and should be avoided.

If you are unsure about the legality or appropriateness of a scan, it is strongly recommended to consult with a legal professional or cybersecurity expert before proceeding. Responsible and authorized use of network scanning tools can be valuable for security research and testing, but improper use can have serious consequences.

By using this application, you acknowledge and agree to use it only for lawful and authorized purposes. The developer of this application is not responsible for any misuse or illegal activities conducted with this tool.

The below shows an example of a successful top 10 port scan of your localhost machine and scanme.nmap.org. As you can see, port 445 is open on this localhost machine, and therefore leaves a potential vulnerability in this target machine. Port 445 is commonly used for the Server Message Block (SMB) protocol, which is a file-sharing and networking protocol used in Windows operating systems.   
  
SMB over port 445 allows users to access shared files, printers, and other resources on a network. This is a common way for Windows computers to share files and collaborate on a local network, but it can also be used for remote administration and management of Windows systems, allowing administrators to access and control remote computers on the network.



However, port 445 is also commonly targeted by cyber attackers, as it can provide access to sensitive data and enable the spread of worms and other malicious code. This is because many Windows systems have SMB enabled by default, and if not properly secured, can be vulnerable to exploitation.

# FTP Client-Server

This FTP client-server application allows you to upload and download files between a local directory (on the client-side) and a remote FTP server directory. The application uses the ftplib and pyftpdlib libraries in Python to handle the FTP protocols.

**Getting Started**

1. **Running the FTP Server**:
   * Ensure you have Python installed on your system.
   * Navigate to the directory containing the ftp-server.py file.
   * Run the ftp-server.py script to start the FTP server.
   * The server will start running on localhost:8088.
2. **Running the FTP Client**:
   * Navigate to the directory containing the main.py file.
   * Run the main.py script to start the FTP client.
   * The client will connect to the FTP server running on localhost:8088.

**Using the FTP Client**

When you run the main.py script, you will see the main menu with the following options:

1. **Download file from ftpServerData to ftpClientData**:
   * This option allows you to download a file from the FTP server's ftpServerData directory to the client's ftpClientData directory.
   * The program will first display the contents of the ftpServerData directory.
   * You will then be prompted to enter the name of the file you want to download.
   * The file will be downloaded and saved in the ftpClientData directory.
2. **Upload file from ftpClientData to ftpServerData**:
   * This option allows you to upload a file from the client's ftpClientData directory to the FTP server's ftpServerData directory.
   * The program will first display the contents of the ftpClientData directory.
   * You will then be prompted to enter the name of the file you want to upload.
   * The file will be uploaded to the ftpServerData directory on the FTP server.
3. **Exit**:
   * This option allows you to exit the FTP client application.

**Folder Structure**

The project has the following folder structure:

* ftpClientData: This directory contains the files that the client can download from or upload to the FTP server.
* ftpServerData: This directory contains the files that are hosted on the FTP server.

**Troubleshooting**

* Ensure that the ftp-server.py script is running before running the main.py script.
* Check the network connectivity and firewall settings to ensure that the client can connect to the FTP server.
* Verify that the file names and paths are correct when downloading or uploading files.

If you encounter any issues or have further questions, please contact the project maintainer.

The following video tutorial will explain how the FTP Client-server will work

A screenshot of a computer

Description automatically generated



# Custom Packet

The custom packet sender is a Python script that allows you to create and send custom network packets using the Scapy library. It supports sending TCP, UDP, and ICMP echo request packets with the ability to specify source/destination addresses and ports, packet type, and payload data. It can be a valuable tool for network testing, troubleshooting, and exploration.

**Features**

* Send TCP, UDP, or ICMP echo request packets
* Customizable source and destination IP addresses and ports
* Ability to specify payload data for the packet
* Option to send multiple packets in a single run
* Validation of input parameters to ensure valid values

**Usage**

1. **Run the script**: To use the custom packet sender, simply run the Python script.
2. **Enter Source Address**: The script will prompt you to enter the source IP address or URL for the packet. It will validate the input to ensure a valid URL format.
3. **Enter Source Port**: Next, you will be asked to enter the source port number. The script will validate that the port is between 1 and 65535.
4. **Enter Destination Address**: You will then be prompted to enter the destination IP address or URL for the packet. Again, the input will be validated for a valid URL format.
5. **Enter Destination Port**: You will then be asked to enter the destination port number, which will be validated to be between 1 and 65535.
6. **Select Packet Type**: The script will then prompt you to select the packet type: TCP, UDP, or ICMP echo request. This is case-sensitive, so be sure to enter "T", "U", or "I" as appropriate.
   * Note: For ICMP echo request, the port number will be ignored.
7. **Enter Packet Data**: You can optionally enter raw data to include in the packet. If left blank, the default data "DISM-DISM-DISM-DISM" will be used.
8. **Enter Packet Count**: Finally, you will be asked to enter the number of packets to send, between 1 and 65535.
9. **Start Sending Packets**: The script will then prompt you to enter "Y" to start sending the packets. If you enter anything else, the script will return to the main menu.

The script will send the specified number of packets and display the number of packets successfully sent.

**Examples**

1. **Send a TCP packet**:
   * Source Address: www.example.com
   * Source Port: 12345
   * Destination Address: www.google.com
   * Destination Port: 80
   * Packet Type: T
   * Packet Data: Hello, world!
   * Packet Count: 10
2. **Send a UDP packet**:
   * Source Address: 10.0.0.1
   * Source Port: 54321
   * Destination Address: 8.8.8.8
   * Destination Port: 53
   * Packet Type: U
   * Packet Data: DNS query
   * Packet Count: 5
3. **Send an** **ICMP echo request**:
   * Source Address: www.sp.com
   * Source Port: (ignored)
   * Destination Address: www.pp.com
   * Destination Port: (ignored)
   * Packet Type: I
   * Packet Data: Hola
   * Packet Count: 3

Here is an example on sending a custom packet using ICMP echo request:

A screenshot of a computer

Description automatically generated

Below shows the successful wireshark capture:

A screenshot of a computer

Description automatically generated

The custom packet sender is a powerful and versatile tool that can be invaluable for network professionals and hobbyists alike. Its ability to create and send custom network packets using the Scapy library provides users with a high degree of flexibility and control, allowing them to test network behavior, troubleshoot issues, and explore the intricacies of network protocols. As networks continue to grow in complexity and new technologies emerge, the custom packet sender can serve as an indispensable tool for analyzing and understanding network behavior, identifying vulnerabilities, and developing more robust and secure network systems.

Moving forward, the continued development and refinement of the custom packet sender, along with the expansion of its capabilities and integrations, could further enhance its value and solidify its place as a essential tool in the network professional's arsenal. Ultimately, the custom packet sender represents a valuable addition to the network testing and analysis landscape, empowering users to gain deeper insights and take more proactive control of their network environments.

# Whois Lookup

The whoislookup.py script is a Python-based tool that allows you to perform a Whois lookup on an IP address or domain name. Whois is a protocol that provides information about the registration and ownership of internet domain names and IP addresses.

The Whois protocol was developed in the early days of the internet to provide information about the registration and ownership of domain names and IP addresses. The original purpose was to allow network administrators and users to contact responsible parties for reporting issues or resolving technical problems.

The first Whois protocol was defined in the 1980s, as a simple text-based lookup system querying a central database of domain and IP registrations. Over the years, the Whois system evolved to accommodate the growing internet, with various extensions and improvements to the protocol. Today, Whois remains an essential part of the internet's infrastructure, providing a valuable resource for domain research, network troubleshooting, and intellectual property enforcement, despite some criticism over privacy and data accuracy concerns. The whoislookup.py script in this guide uses the python-whois library to provide a convenient Whois lookup interface.

This script will take an IP address or domain name as input and display the corresponding Whois information, including details such as the registrant's name, contact information, and the domain's registration and expiration dates.

The script will prompt you to enter an IP address or domain name, and then display the corresponding Whois information.

**Functionality**

The whoislookup.py script has the following functionality:

1. **IP Address Lookup**: If the user enters an IP address, the script will perform a Whois lookup on the IP address and display the associated information.
2. **Domain Name Lookup**: If the user enters a domain name, the script will perform a Whois lookup on the domain and display the associated information.
3. **Error Handling**: The script includes error handling to catch any exceptions that may occur during the Whois lookup process and display an appropriate error message.

Here are some sample inputs you can use to test your whois\_lookup() function:

IP Address:

* 8.8.8.8
* 192.168.1.1
* 1.1.1.1

Domain Name:

* google.com
* github.com
* example.org
* anthropic.com

**Output**

A screenshot of a computer

Description automatically generated

The output of the whoislookup.py script will display the Whois information for the entered IP address or domain name. The information will be presented in a key-value format, where each piece of information (such as the registrant's name, contact details, and registration/expiration dates) will be displayed on a new line.

**Limitations**

* The script relies on the python-whois library, which may not have complete or up-to-date Whois information for all IP addresses or domain names.
* The script does not provide any advanced features, such as batch processing or saving the Whois information to a file.
* The script may not work correctly with certain special characters or non-standard domain names or IP addresses.

The whoislookup.py script provides a simple and straightforward way to perform Whois lookups on IP addresses and domain names from within VSCode. While it has some limitations, it can be a useful tool for developers, network administrators, or anyone who needs to quickly obtain Whois information.

# Keylogger

The keylogger.py script is a simple yet powerful tool that allows you to record all keystrokes made on your computer. This can be useful for a variety of purposes, such as monitoring user activity, troubleshooting software issues, capturing sensitive information or serving as intrusion detection.

**Features**

* Logs all keystrokes, including special characters and modifier keys (e.g., shift, ctrl, alt).
* Records the timestamp of each keystroke.
* Saves the log file in the current directory with the name keystrokes.txt.

**Log File**

The logged keystrokes will be saved in the keystrokes.txt file located in the same directory as the keylogger.py file. Each entry in the log file will contain the timestamp and the corresponding keystroke.

**Important Notes**

* This keylogger is designed for educational and testing purposes only. Use it responsibly and respect the privacy of others.
* The keylogger will continue to run until you stop it by pressing Ctrl + C. Make sure to stop the script when you're done monitoring the keystrokes.

A screenshot of a computer

Description automatically generated

To stop the keylogger, press Ctrl + C in the terminal or command prompt.

The below shows a sample keylogger capture in the keystrokes.txt file:

A screenshot of a computer

Description automatically generated

The log file will continue to grow in size as more keystrokes are recorded. It's a good idea to periodically review and manage the log file to avoid excessive disk usage.

**Limitations**

The keylogger only logs keystrokes and does not capture other user activities, such as mouse movements or screen captures.

The keylogger.py script provides a simple and straightforward way to record keystrokes on your computer from within VSCode. While it has some limitations, it can be a useful tool for various purposes, such as testing, troubleshooting, or monitoring user activity. Remember to use this tool responsibly and respect the privacy of others.

# Contact Us

We are all human, we are not perfect and we all make mistakes. At PSEC Information Security Apps, we are always striving to serve you better. That is why we want to hear from you!

A screenshot of a computer

Description automatically generated

All your contact information and comments will be saved here in the contact\_messages.txt file for our developers to get back to you!

A screenshot of a computer

Description automatically generated

I hope this user guide helps you understand how to use PSEC Information Security Apps. Do let us know if you have any further questions!