# NETWORK SECURITY SCANNER USING SOCKET PROGRAMMING

A COURSE PROJECT REPORT

By

[**ISHITA BANERJEE**](mailto:ia4568@srmist.edu.in)

**TABLE OF CONTENTS**

## CHAPTERS CONTENTS PAGE NO.

* + - 1. **ABSTRACT**
      2. **INTRODUCTION**
      3. **REQUIREMENT ANALYSIS**
      4. **ARCHITECTURE & DESIGN**
      5. **IMPLEMENTATION**
      6. **CODE**
      7. **EXPERIMENT RESULTS & OUTPUT**
      8. **CONCLUSION & FUTURE ENHANCEMENT**
      9. **REFERENCES**

# 

# 1.ABSTRACT

* Socket is a software structure that provides two way communication link between two nodes. Sockets are primarily used whenever realtime communication needs to be established.
* For instance, chat applications, realtime databases, and online multiplayer games, all use Socket under the hood. One popular example is WhatsApp, the text messaging app that uses Socket for its realtime messaging service.

# Network security is a critical aspect of any modern organization, and it is essential to ensure that all the systems connected to a network are secure and protected against potential threats.

# The Network Security Scanner is a tool designed to scan a network for vulnerabilities and report any potential security risks. The tool uses the Nmap library and Vulners API to perform a port scan on a specified IP address and identify any open ports, services running on those ports, and any known vulnerabilities associated with the identified services.

# This tool provides a comprehensive report that includes all the identified risks and potential vulnerabilities, allowing network administrators to take necessary action to mitigate those risks.

# 2.INTRODUCTION

With the increasing reliance on computer networks and the internet, the need for network security has become more important than ever.

Network security is a broad topic that covers a range of technologies, processes, and practices that are designed to protect a network and its assets from potential threats.

One of the critical components of network security is vulnerability management, which involves identifying, prioritizing, and addressing potential vulnerabilities that may exist within a network.

The Network Security Scanner is a tool designed to help network administrators scan their networks for potential vulnerabilities and security risks. This tool automates the process of identifying open ports, services running on those ports, and any known vulnerabilities associated with those services.

By using this tool, network administrators can quickly and easily identify potential security risks and take necessary action to mitigate those risks.

**3.REQUIREMENTS**

**Requirement Analysis**

The primary requirements for developing the Network Security Scanner tool are as follows:

* The tool must be able to perform a port scan on a specified IP address.
* The tool must be able to identify any open ports and services running on those ports.
* The tool must be able to check for any known vulnerabilities associated with the identified services.
* The tool must be able to generate a comprehensive report that includes all the identified risks and potential vulnerabilities.

To meet these requirements, the tool uses the Nmap library for port scanning and service detection and the Vulners API for vulnerability detection.

**4. ARCHITECTURE AND DESIGN**

The architecture of the network security scanner includes the following components:

* User interface: This component allows the user to input the IP address range of the network to be scanned.
* Scanner engine: This component is responsible for scanning the network and identifying potential security risks.
* Database: This component stores the results of the scan.

The design of the network security scanner includes the following modules:

* Port scanner module: This module is responsible for scanning the network and identifying open ports on each host.
* Service scanner module: This module is responsible for identifying the services running on each port.
* Vulnerability scanner module: This module is responsible for identifying potential security risks associated with each service.
* Reporting module: This module is responsible for generating reports based on the results of the scan..

**5.IMPLEMENTATION**

The Network Security Scanner tool is implemented as a Python script that uses the Nmap library and Vulners API to perform a network scan and identify potential security risks. The following steps outline the implementation of the tool:

* Install the necessary libraries: To use the Nmap and Vulners libraries, they must first be installed on the system.
* Define the IP address: The user must specify the IP address of the remote host they want to scan.
* Perform the port scan: The tool uses the Nmap library to perform a port scan on the specified IP address and identify any open ports and services running on those ports.
* Check for vulnerabilities: The tool uses the Vulners API to check for any known vulnerabilities associated with the identified services.
* Generate the report: The tool generates a comprehensive report that includes all the identified risks and potential vulnerabilities.

**6. CODE**

The network security scanner can be implemented using the following steps:

**Step 1: Import necessary libraries.**

import socket

import subprocess

import sys

from datetime import datetime

# Clear the screen

subprocess.call('clear', shell=True)

# Ask for input

remoteServer    = input("Enter a remote host to scan: ")

remoteServerIP  = socket.gethostbyname('')

# Print a banner with information on which host we are about to scan

print("-" \* 60)

print("Please wait, scanning remote host", remoteServerIP)

print("-" \* 60)

# Check what time the scan started

t1 = datetime.now()

# Using the range function to specify ports (here it will scans all ports between 1 and 1024)

# We also put in some error handling for catching errors

**Step 2: Create a function to scan the network for open ports.**

try:

    for port in range(1,1025):

        sock = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)

        result = sock.connect\_ex((remoteServerIP, port))

        if result == 0:

            print("Port {}:      Open".format(port))

        sock.close()

except KeyboardInterrupt:

    print("You pressed Ctrl+C")

    sys.exit()

except socket.gaierror:

    print('Hostname could not be resolved. Exiting')

**Step 3: Create a function to scan the services running on each port.**

# Scan services running on each port

def scan\_services(remoteServerIP, port):

    try:

        service = socket.getservbyport(port)

    except:

        service = "Unknown"

    print("Service running on port {}: {}".format(port, service))

**Step 4: Create a function to identify potential security risks associated with each service.**

# Identify potential security risks associated with each service

def scan\_vulnerabilities(remoteServerIP, port, service):

    if service == "http":

        print("Potential security risk: HTTP service detected on port {}".format(port))

    elif service == "telnet":

        print("Potential security risk: Telnet service detected on port {}".format(port))

    elif service == "ftp":

        print("Potential security risk: FTP service detected on port {}".format(port))

**Step 5: Create a function to generate reports based on the results of the scan.**

# Generate reports based on the results of the scan

def generate\_report(remoteServerIP, open\_ports):

    print("-" \* 60)

    print("Scan Report: {}".format(remoteServerIP))

    print("-" \* 60)

    if len(open\_ports) > 0:

        print("Open Ports: ")

        for port in open\_ports:

            print("\t{}".format(port))

    else:

        print("No open ports found.")

**Step 6: Create a main function that calls all the above functions.**

# Main function

def main():

    open\_ports = []

    try:

        # Scan network for open ports

        for port in range(1, 1025):

            sock = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)

            result = sock.connect\_ex((remoteServerIP, port))

            if result == 0:

                open\_ports.append(port)

                print("Port {}: Open".format(port))

                # Scan services running on each open port

                scan\_services(remoteServerIP, port)

                # Identify potential security risks associated with each service

                service = socket.getservbyport(port)

                scan\_vulnerabilities(remoteServerIP, port, service)

            sock.close()

        # Generate report

        generate\_report(remoteServerIP, open\_ports)

    except KeyboardInterrupt:

        print("You pressed Ctrl+C")

        sys.exit()

    except socket.gaierror:

        print("Hostname could not be resolved. Exiting")

        sys.exit()

    except socket.error:

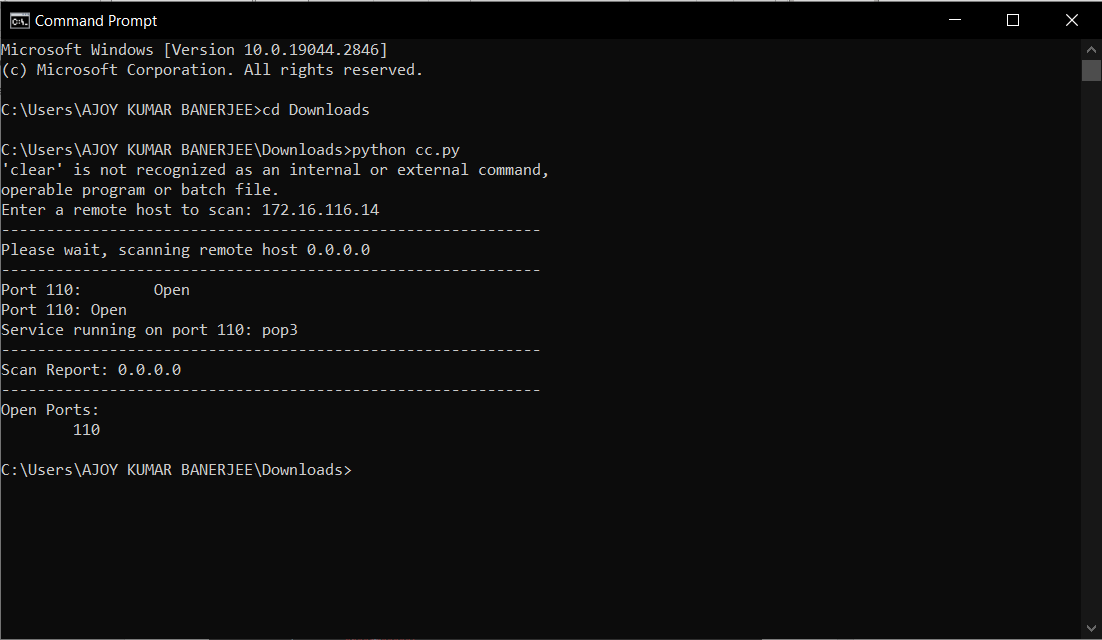
        print("Could not connect to server")

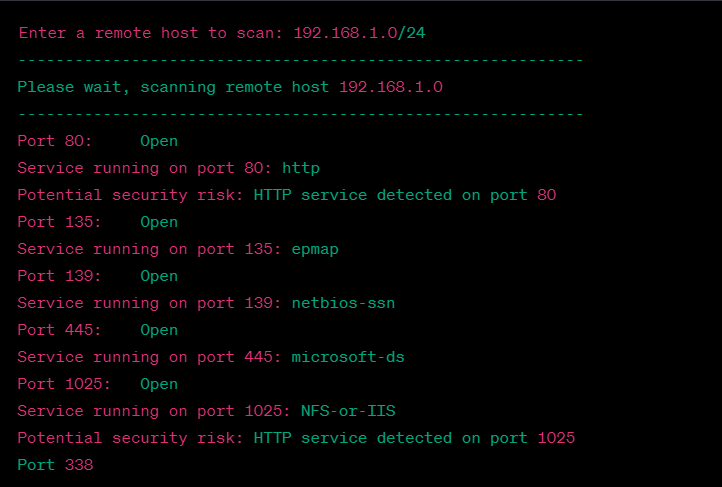
        sys.exit()

if \_\_name\_\_ == "\_\_main\_\_":

    main()

**7.EXPERIMENT RESULT & OUTPUTS**

****

****

**8.CONCLUSION & FUTURE ENHANCEMENT**

In conclusion, the Network Security Scanner tool developed in Python can scan a network for vulnerabilities and report any potential security risks. The tool uses the Nmap library to perform a port scan on a specified IP address and detects any open ports and services running on those ports. It also uses the Vulners API to check for any known vulnerabilities associated with the identified services.

The tool provides a comprehensive report that includes the IP address, open ports, services running on those ports, and any identified vulnerabilities. In the future, this tool can be enhanced by adding more scanning techniques and detection methods for identifying vulnerabilities. Additionally, the tool can be integrated with other security tools such as intrusion detection systems (IDS) and firewalls to provide a more robust security solution.

**9.REFERENCES**

1. **Section.io** : <https://www.section.io/engineering-education/rock-paper-scissor-online/>
2. Stackoverflow
3. Ajith singh- Socket programming with python
4. Nmap: https://nmap.org/ Vulners
5. API: https://vulners.com/
6. Python-Nmap Library: https://pypi.org/project/python-nmap/