Network scanner tools .

**Introduction to the Network Security Scanning Project**

In today’s interconnected world, ensuring the security of a network is more critical than ever. Networks, especially in business and educational environments, are constantly exposed to a wide variety of threats—ranging from unauthorized access to data breaches and denial-of-service attacks. For network administrators, cybersecurity professionals, and anyone interested in strengthening their network defenses, knowing how to detect vulnerabilities and secure devices is an essential skill.

This project aims to address these concerns by providing a **network scanning and vulnerability detection tool**. The core functionality of this tool is to **scan a network**, identify **connected devices**, and then evaluate their **open ports** and **potential vulnerabilities**. By doing so, this project helps users assess the **security posture** of their network and devices, identify possible weaknesses, and take proactive steps to secure their infrastructure.

The tool was designed with an easy-to-use command-line interface (CLI), making it simple to specify the network range to scan, select devices to investigate, and retrieve detailed information about their **operating systems**, **open ports**, and **known vulnerabilities**. The scanning process is powered by two popular tools in the security industry: **Nmap** and **Scapy**. These tools are widely used for **network exploration**, **security auditing**, and **vulnerability detection**.

**Key Features of the Project:**

1. **Network Scanning:**
   * The tool performs an **ARP scan** to discover all the active devices in a given IP range. It identifies devices by their **IP addresses** and **MAC addresses**, which provides a clear overview of the devices connected to the network.
2. **Operating System Detection:**
   * By leveraging **Nmap’s OS detection feature**, the tool can detect the **operating systems** running on the identified devices. This information is crucial because different operating systems often have distinct vulnerabilities.
3. **Open Ports Scanning:**
   * The tool scans each device for **open ports**, revealing which network services are available for interaction. Open ports can be potential entry points for attackers, making it vital to identify and secure them.
4. **Vulnerability Scanning:**
   * Once open ports are identified, the tool uses **Nmap’s vulnerability scripts** to scan these ports for **known vulnerabilities**. This helps highlight weaknesses in services that could be exploited by attackers.

Code :-

It is provided in the directory in which the file is shared

Explanation of the code :-

**Overview of the Project**

The script is designed to help you **scan your network**, discover devices, and check for potential vulnerabilities. If you’ve ever wondered how to check which devices are connected to your network, what services they’re running, or whether any of those services have security weaknesses, this tool does just that.

**1. scan\_network(ip\_range)**

**What It Does:**

This function’s job is to **scan the network** and find out which devices are connected. It does this by sending **ARP requests** to the devices in the provided IP range.

**Why It’s Important:**

Before you can do anything with your network, you need to know which devices are online. This is like walking into a room and checking who’s there.

**How It Works:**

* It sends out an ARP request to every IP address in the range you provide (like 192.168.1.0/24), asking **“Hey, are you there?”**
* If a device responds, the script **records its IP address and MAC address**, letting you know it’s online.

This is basically how your router keeps track of devices that are connected.

**2. detect\_os(ip)**

**What It Does:**

Now that we know which devices are online, it’s time to figure out **what operating system (OS)** they’re running. This function uses **Nmap**, a popular tool for discovering information about devices, to detect the OS.

**Why It’s Important:**

Knowing the operating system of a device can help you identify potential security risks. For example, certain vulnerabilities only affect specific versions of Windows or Linux. This can help you tailor your security tests.

**How It Works:**

* The function tells Nmap to run an **OS detection scan** using its fingerprinting technique.
* It looks for matching patterns in how the device responds to the scan, and if a match is found, it returns the name of the OS.
* If it can’t figure out the OS, it just says **"OS could not be determined."**

**3. scan\_ports(ip)**

**What It Does:**

This function checks which **ports are open** on a device. Ports are like doors through which data flows into a device, and some of them might be vulnerable to attacks.

**Why It’s Important:**

If there are open ports, that means services (like web servers, databases, etc.) are running. Some of these services might have known weaknesses that could be exploited. Finding open ports gives you insight into potential attack surfaces.

**How It Works:**

* The function runs a **full port scan** on the device, checking every possible TCP port (1-65535).
* It then looks at the open ports and tells you what services are running on them. For example, if port 80 is open, you might find that a **web server** is running.

**4. scan\_vulnerabilities(ip, ports)**

**What It Does:**

This function checks the open ports for any **known vulnerabilities**. Nmap has built-in scripts that can look for **common security flaws** in services running on open ports.

**Why It’s Important:**

Knowing that a service is running is helpful, but knowing if that service is **vulnerable** is crucial. For example, an outdated web server might have a known vulnerability that can be exploited by an attacker.

**How It Works:**

* For each open port, the function runs Nmap’s **vulnerability scanning scripts**.
* These scripts check if the service running on the port has any known weaknesses, such as bugs or misconfigurations.
* It then returns any vulnerabilities found, which can include things like **CVE IDs** (common vulnerability identifiers) or exploit details.

**5. scan\_device\_for\_ports\_and\_vulnerabilities(device)**

**What It Does:**

This function ties everything together. For a given device, it first scans for **open ports** and then checks those ports for any **vulnerabilities**.

**Why It’s Important:**

Instead of manually running separate scans for ports and vulnerabilities, this function does both at once and displays the results clearly.

**How It Works:**

* First, it calls scan\_ports() to find out which ports are open on the device.
* Then, it checks those open ports for vulnerabilities using scan\_vulnerabilities().
* Finally, it prints out the open ports and any vulnerabilities that were found.

It’s like a quick security check-up for each device on your network.

**6. Main Functionality (if \_\_name\_\_ == "\_\_main\_\_":)**

**What It Does:**

This is the heart of the script. It asks the user for a **network range**, finds the devices in that range, detects their **operating systems**, and then lets the user choose a device to scan for **open ports and vulnerabilities**.

**Why It’s Important:**

This is the part where everything comes together. It takes the user’s input, runs all the scans, and displays the results in a nice, interactive way.

**How It Works:**

* The user inputs a network range (like 192.168.1.0/24), and the script scans the network to discover the devices.
* For each device, it uses multiple threads to **speed up the OS detection**.
* The user is then shown a list of devices and their operating systems.
* The user selects a device, and the script proceeds to scan that device for **open ports** and **vulnerabilities**.

**Conclusion**

So, in simple terms, this script allows you to **map out your network**, **identify devices**, **see which ports are open**, and **check if there are any vulnerabilities** that could put your network at risk.

It’s like doing a health check-up for your network, finding the **devices**, figuring out **what software they run**, seeing if they have **weak spots**, and then deciding what to do about it. It’s an essential tool for **security professionals** or anyone who wants to keep their network safe.

Output :-



Explanation of output :-

**Summary of the Output:**

The output is the result of scanning the network **192.168.126.0/24**, where several devices were discovered. Among these, the primary target for further scanning was **192.168.126.129**, which is the IP of the **Metasploitable2** virtual machine, an intentionally vulnerable system commonly used for penetration testing and learning about security flaws.

Here’s a breakdown of what the output reveals:

**Discovered Devices:**

1. **IP: 192.168.126.1**  
   **OS:** Microsoft Windows 11 21H2
   * This is likely your **gateway router** or another machine in the network running **Windows 11**. It’s important to know the OS of devices, as it helps in targeting specific vulnerabilities later.
2. **IP: 192.168.126.2**  
   **OS:** VMware Player virtual NAT device
   * This device seems to be a **virtual NAT device**, which is probably used for managing virtual machine network traffic.
3. **IP: 192.168.126.129**  
   **OS:** Linux 2.6.9 - 2.6.33
   * This is the **Metasploitable2** machine, running an outdated version of **Linux**. This device is the primary target of your scan, and it’s intentionally insecure with a number of known vulnerabilities.
4. **IP: 192.168.126.254**  
   **OS:** OS could not be determined
   * This device could not be identified, possibly due to firewall settings or a non-responding machine.

**Device Scanned: 192.168.126.129 (Metasploitable2)**

When you chose to scan **192.168.126.129**, the script found several **open ports** on this device, each representing a service that could potentially have vulnerabilities.

**Open Ports:**

* **Ports 21, 22, 23, 25, 53, 80, 111, 139, 445, 512, 513, 514, 1099, 1524, 2049, 2121, 3306, 3632, 5432, 5900, 6000, 6667, 6697, 8009, 8180, 8787, and others** are all open on the device.
  + These ports correspond to well-known services such as **FTP**, **SSH**, **Telnet**, **HTTP**, **SMTP**, **MySQL**, **VNC**, **RPC**, **NFS**, and more.
  + Some of these ports, especially **FTP (21)**, **Telnet (23)**, and **MySQL (3306)**, are often targeted in attacks due to their common presence and known vulnerabilities.
  + The sheer number of open ports on this machine indicates that it’s running several **unnecessary services**, which increases its attack surface.

**Vulnerabilities Found:**

* The scan also looked for known **vulnerabilities** associated with these open ports. Specifically, it checked for vulnerabilities related to **SMB** (Server Message Block), which is commonly targeted in network attacks.
  + **smb-vuln-ms10-061**: This vulnerability is related to a flaw in **Windows SMB** implementations. The result here indicates that **Metasploitable2** is **not vulnerable** to this specific issue, as shown by false.
  + **smb-vuln-regsvc-dos**: This vulnerability is related to a **Denial of Service (DoS)** attack affecting **Windows services**. However, the scan encountered an **error** while attempting to check this vulnerability. This could mean the script wasn’t able to execute properly, possibly due to network issues or misconfigurations.
  + **smb-vuln-ms10-054**: Similar to **ms10-061**, this is another vulnerability affecting **Windows SMB**. Again, the result is false, indicating no issue here.
  + The errors related to **smb-vuln-regsvc-dos** could be due to **script execution failures**, but it doesn't imply a major security risk unless further investigated.

**Detailed Explanation of the Key Findings:**

* **Metasploitable2’s Open Ports:**  
  Metasploitable2 has a number of **open ports**, some of which are intentionally left vulnerable to help people practice penetration testing. For example:
  + **Port 21 (FTP):** Known to have vulnerabilities like **anonymous login**, allowing attackers to gain access to files.
  + **Port 23 (Telnet):** Telnet is an outdated protocol that transmits data in plaintext, which is vulnerable to interception and attacks.
  + **Port 80 (HTTP):** HTTP servers often have vulnerabilities like **SQL injection**, **XSS**, or outdated versions of web applications.
  + **Port 3306 (MySQL):** Often targeted for **SQL injection** attacks or weak/default credentials.
* **Vulnerabilities Found:**
  + **SMB Vulnerabilities (ms10-061, ms10-054):** These are vulnerabilities in the **Windows SMB protocol** that could allow an attacker to exploit weaknesses in file-sharing features. However, these are not affecting the Metasploitable2 machine in this case.
  + **Script Errors in SMB Vulnerability Checks:** The errors encountered when attempting to check for **smb-vuln-regsvc-dos** might suggest that the scan couldn't complete successfully for that specific vulnerability. It’s worth investigating further if the goal is to check for Denial of Service vulnerabilities.

**Conclusion:**

From this scan, we can gather several important insights:

1. **Metasploitable2 (192.168.126.129)** is running many open services that might have vulnerabilities.
2. While some **SMB vulnerabilities** weren’t found or were skipped due to errors, the machine still has a **large attack surface** due to its open ports.
3. This scan is a useful starting point for a penetration test, as it shows which services might be exploited and which vulnerabilities can be tested further.

Overall, this scan confirms that **Metasploitable2** is a vulnerable system, as intended for learning purposes, and offers multiple attack vectors for testing. If this were a real-world system, you’d likely want to close unnecessary ports, update services, and patch known vulnerabilities.

**Conclusion:**

In conclusion, this project provides a robust and comprehensive solution for network security scanning, focusing on discovering devices, identifying their operating systems, scanning for open ports, and detecting known vulnerabilities. By leveraging powerful tools like **Nmap** and **Scapy**, it enables efficient detection of security gaps in a network, offering valuable insights into potential attack vectors. This project serves as a practical demonstration of how network security assessments can be automated, providing a foundational understanding of the vulnerabilities that can affect various network services.

As we move forward, there are numerous opportunities to expand and improve this project. Future growth could involve incorporating additional vulnerability detection scripts, enhancing the accuracy of OS detection, and expanding the types of scans available (e.g., service version detection, SSL/TLS testing). Integrating real-time monitoring and alerts, or even creating a web-based user interface, would provide an even more user-friendly experience.

I plan to continue refining and expanding this project, incorporating more advanced features and improving its usability to make it an even more powerful tool for network security assessments.