Nmap Script Documentation

Purpose of the Scripts

Nmap is a powerful network scanning tool used for network discovery and security auditing. The provided scripts automate various scanning tasks to make the process efficient and consistent.

The first script, Full Network Scan, scans an entire subnet for all open ports, providing a comprehensive view of the network's devices and their exposed services. The second script, Top Ports Scan, focuses on scanning the top 20 most used ports on a specific target IP, which helps quickly identify critical services running on the host. The third script, OS Detection, is designed to detect the operating system of a target IP, which is useful for identifying potential vulnerabilities related to specific OS versions.

Usage Instructions

1. To use these scripts, ensure Python and the nmap library are installed. You can install the library using the command:
   pip install python-nmap
2. Save each script as a .py file and run them from the command line or any Python IDE.
3. For the Full Network Scan, run the following command:
   python full_network_scan.py
4. This script requires a subnet parameter, which specifies the subnet to scan (e.g., 192.168.1.0/24).
5. For the Top Ports Scan, use the command:
   python top_ports_scan.py
6. This script requires a target_ip parameter, which is the IP address of the target (e.g., 192.168.1.1).
7. For the OS Detection, run:
   python os_detection.py
8. This script also requires a target_ip parameter, specifying the IP address of the target (e.g., 192.168.1.1).

Expected Outputs or Results

The Full Network Scan script generates a detailed list of all active hosts in the subnet and their open ports. The results are saved in a timestamped text file.

The Top Ports Scan script produces a CSV file containing details of the scanned top ports, including the host, port, state, and service.

The OS Detection script prints the detected operating system details, including the OS family and accuracy.

Dependencies Needed

These scripts depend on the python-nmap library, which provides a Pythonic interface to Nmap.

Line-by-Line Explanation

Full Network Scan

The script begins by importing the necessary libraries:

```
import nmap
import datetime
```

The nmap library is used for network scanning, and datetime is used for timestamping the result files.

Next, the function full_network_scan is defined, which takes a subnet parameter. Inside the function, a PortScanner object is initialized:

```
scanner = nmap.PortScanner()
```

A timestamped filename is generated for the results:

```
timestamp = datetime.datetime.now().strftime("%Y%m%d_%H%M%S")
result_file = f"full_network_scan_{timestamp}.txt"
```

The subnet is scanned for all ports using the following command:

```
scanner.scan(hosts=subnet, arguments='-p-')
```

The results are written to a file:

```
with open(result_file, "w") as file:
    for host in scanner.all_hosts():
        file.write(f"Host: {host}\n")
        file.write(f"State: {scanner[host].state()}\n")
        for proto in scanner[host].all_protocols():
            file.write(f"Protocol: {proto}\n")
            for port in scanner[host][proto].keys():
                state = scanner[host][proto][port]['state']
                file.write(f"Port: {port}, State: {state}\n")
```

Finally, a confirmation message is printed:

```
print(f"Scan complete. Results saved to {result_file}")
```

Top Ports Scan

The script starts with the same imports and initializes a PortScanner object. A timestamped CSV file is created for the results:

```
timestamp = datetime.datetime.now().strftime("%Y%m%d_%H%M%S")
result_file = f"top_ports_scan_{timestamp}.csv"
```

The top 20 ports on the target IP are scanned:

```
scanner.scan(target_ip, arguments='--top-ports 20')
```

The results are written to a CSV file with headers:

```
with open(result_file, mode='w', newline='') as file:
    writer = csv.writer(file)
    writer.writerow(['Host', 'Port', 'State', 'Service'])
    for host in scanner.all_hosts():
        for proto in scanner[host].all_protocols():
            for port in scanner[host][proto].keys():
                state = scanner[host][proto][port]['state']
                service = scanner[host][proto][port]['name']
                writer.writerow([host, port, state, service])
```

A completion message is printed:

```
print(f"Scan complete. Results saved to {result_file}")
```

OS Detection

The os_detection function begins by initializing a PortScanner object and printing the target IP:

```
scanner = nmap.PortScanner()
print(f"Detecting OS for {target_ip}")
```

The OS detection scan is executed:

```
scanner.scan(target_ip, arguments='-O')
```

If OS information is available, it is printed:

```
if 'osclass' in scanner[target_ip]:
    for os_class in scanner[target_ip]['osclass']:
        print(f"OS: {os_class['osfamily']}, Accuracy: {os_class['accuracy']}%")
else:
    print("No OS information available")
```

Why These Scripts Are Needed ?

Automating Nmap scans ensures consistency, saves time, and reduces human error during security assessments. The Full Network Scan provides comprehensive visibility into all devices and services in a subnet, helping to identify potential security risks. The Top Ports Scan quickly identifies critical services on a host, which is crucial for prioritizing security efforts. The OS Detection helps in understanding the target's OS, allowing for tailored vulnerability assessments based on the specific operating system.