```
├──> Phase 2: Vulnerability Scanning Automation
 │      │
 │      ├── Set Up Automated Web Scanning
 │      │       ├── Integrate OWASP ZAP for Basic Web Scans
 │      │       └── Write Python Script to Trigger ZAP Scans
 │      │
 │      └── Set Up Network Scanning Automation
 │              ├── Integrate Nmap for Basic Network Scans
 │              └── Automate Nmap Execution with Python Scripts
 │
```

Step 1: Set Up Automated Web Scanning with OWASP ZAP

1. Install OWASP ZAP

- Download and install OWASP ZAP from the official website.

2. Install Python Dependency

Use the `python-owasp-zap-v2.4` library to interact with ZAP.
bash
Copy code

```
pip install python-owasp-zap-v2.4
```

-

3. Write Python Script to Trigger OWASP ZAP Scans

zap_scan.py:

python
Copy code

```
from zapv2 import ZAPv2
import time

# ZAP proxy details
zap = ZAPv2(proxies={'http': 'http://127.0.0.1:8080', 'https':
'http://127.0.0.1:8080'})

# Target URL for scanning
target_url = 'http://example.com'

# Start ZAP spider (crawling the target)
print(f"Spidering target: {target_url}")
```

```
zap.spider.scan(target_url)
time.sleep(5)

while int(zap.spider.status()) < 100:
    print(f"Spider progress: {zap.spider.status()}%")
    time.sleep(2)

print("Spider completed!")

# Start active scan
print(f"Starting active scan on: {target_url}")
zap.ascan.scan(target_url)

while int(zap.ascan.status()) < 100:
    print(f"Active scan progress: {zap.ascan.status()}%")
    time.sleep(5)

print("Active scan completed!")

# Print vulnerabilities found
alerts = zap.core.alerts()
for alert in alerts:
    print(f"Risk: {alert['risk']}, Description:
{alert['description']}")
```

Step 2: Set Up Network Scanning Automation with Nmap

1. Install Nmap

Install Nmap using the package manager for your operating system:
bash
Copy code

```
sudo apt-get install nmap
```

-

2. Install Python Dependency

Use the `python-nmap` library to interact with Nmap.
bash
Copy code

```
pip install python-nmap
```

-

3. Write Python Script to Trigger Nmap Scans

nmap_scan.py:

python
Copy code
```
import nmap

# Initialize the Nmap scanner
nm = nmap.PortScanner()

# Define the target and the scan arguments
target = '192.168.1.1'
scan_arguments = '-sS -O -Pn'

# Perform the scan
print(f"Scanning target: {target}")
nm.scan(hosts=target, arguments=scan_arguments)

# Output scan results
for host in nm.all_hosts():
    print(f"Host: {host} ({nm[host].hostname()})")
    print(f"State: {nm[host].state()}")
    for protocol in nm[host].all_protocols():
        print(f"Protocol: {protocol}")
        ports = nm[host][protocol].keys()
        for port in ports:
            print(f"Port: {port}, State:
{nm[host][protocol][port]['state']}")
```

Step 3: Automate Both Scans

Create a master script to automate both **OWASP ZAP** and **Nmap** scans.

vulnerability_scan_automation.py:

python
Copy code
```
import subprocess
import zap_scan
import nmap_scan
```

```
def run_owasp_zap():
    print("Starting OWASP ZAP Scan...")
    zap_scan.run()  # Call the ZAP script (assumes it's structured
as a function)

def run_nmap_scan():
    print("Starting Nmap Scan...")
    subprocess.run(['python', 'nmap_scan.py'])  # Trigger Nmap
script

if __name__ == "__main__":
    print("Vulnerability Scanning Automation Started!")
    run_owasp_zap()
    run_nmap_scan()
    print("All scans completed!")
```

Directory Structure

Your directory structure should look like this:

bash
Copy code
```
/vulnerability_scanning
    zap_scan.py          # Script for OWASP ZAP scans
    nmap_scan.py         # Script for Nmap scans
    vulnerability_scan_automation.py  # Master script for automation
```

Running the Scripts

Start OWASP ZAP in Headless Mode (if automating on a server):
bash
Copy code
```
zap.sh -daemon
```

1.

Run the Master Script:
bash
Copy code
```
python vulnerability_scan_automation.py
```

2.