Day 01: Nmap Scan and Automation Report

- **Prepared by:** Girija Shankar Sahoo
- **Date:** July 28, 2025

Nmap, short for Network Mapper, is a free and open-source utility for network discovery and security auditing. It is a powerful tool used to discover hosts and services on a computer network by sending packets and analyzing the responses.

Nmap is used to discover and audit computer networks by identifying what devices are running, what services they offer, and what operating systems they use.

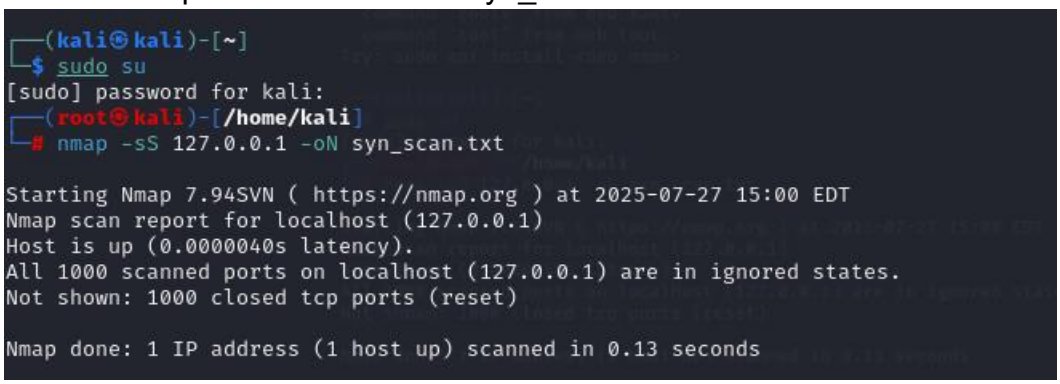
Target Scope

- **Target IP Address:** 127.0.0.1
- **Hostname:** localhost
- **Description:** The target is the local loopback interface of the Kali Linux operating system, used as a safe and authorized environment for this security assessment.

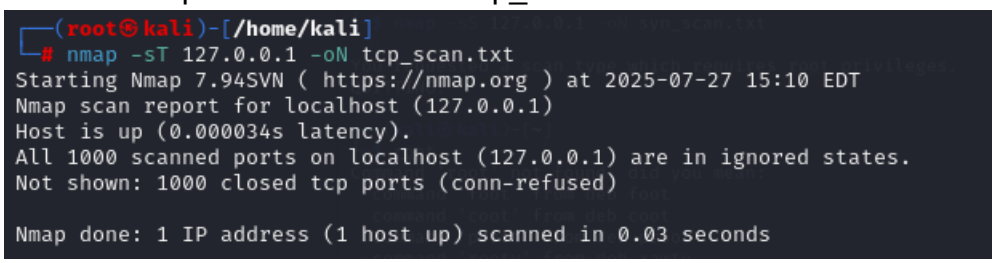
Methodology

Initial reconnaissance was performed using manual Nmap commands to establish a baseline understanding of the target. Subsequently, a Python script was developed to automate these scans.

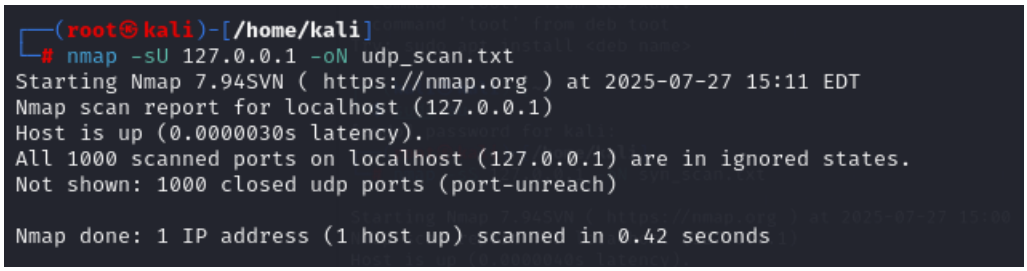
- **Manual Commands Executed:**
  nmap -sS 127.0.0.1 -oN syn_scan.txt



nmap -sT 127.0.0.1 -oN tcp_scan.txt



nmap -sU 127.0.0.1 -oN udp_scan.txt



PYTHON SCRIPT:

```
import nmap
import datetime

def main():
    target = input("Please enter the target IP address: ")
    scanner = nmap.PortScanner()
    print(f"\n[+] Scanning target: {target}...")

    try:
        scanner.scan(target, arguments='-sS -sV')
        print("[+] Scan complete!")

    except nmap.PortScannerError:
        print("[-] Nmap not found. Please install it and ensure it's in your system's
PATH.")
        return

    report_file = "scan_report.txt"
    print(f"[+] Generating report: {report_file}...")
```

```
    with open(report_file, "w") as f:
        f.write("--- Nmap Scan Report ---\n\n")
        f.write(f"Scan performed at: {datetime.datetime.now().strftime('%Y-%m-%d
%H:%M:%S')}\n")
        f.write(f"Target IP: {target}\n\n")

        if not scanner.all_hosts():
            f.write("No hosts found or host is down.\n")

        else:
            for host in scanner.all_hosts():
                f.write(f"Host: {host} ({scanner[host].hostname()})\n")
                f.write(f"State: {scanner[host].state()}\n\n")

                f.write("--- Open Ports and Services ---\n")
                f.write("{:<10} {:<10} {:<20} {}\n".format('PORT', 'STATE', 'SERVICE',
'VERSION'))

                for proto in scanner[host].all_protocols():
                    ports = scanner[host][proto].keys()
                    sorted_ports = sorted(ports)

                    for port in sorted_ports:
                        state = scanner[host][proto][port]['state']
                        name = scanner[host][proto][port]['name']
                        version = scanner[host][proto][port]['version']

                        f.write("{:<10} {:<10} {:<20} {}\n".format(port, state, name, version))
        f.write("\n--- Scan Complete ---\n")

    print("[+] Report saved successfully.")
```

4. Findings and Analysis

```
1 ── Nmap Scan Report ──
2
3 Scan performed at: 2025-07-27 15:33:11
4 Target IP: 127.0.0.1
5
6 Host: 127.0.0.1 (localhost)
7 State: up
8
9 ── Open Ports and Services ──
10 PORT       STATE      SERVICE             VERSION
11
12 ── Scan Complete ──
13
```

5. Network Topology Diagram


```
1 ── Nmap Scan Report ──
2
3 Scan performed at: 2025-07-27 15:33:11
4 Target IP: 127.0.0.1
5
6 Host: 127.0.0.1 (localhost)
7 State: up
8
9 ── Open Ports and Services ──
10 PORT       STATE      SERVICE             VERSION
11
12 ── Scan Complete ──
13
```


6. Conclusion and Learnings

This project provided valuable hands-on experience with fundamental cybersecurity tools and practices. Key learnings include the functional differences between Nmap scan types (-sS, -sT, -sU), the importance of version scanning (-sV) for vulnerability assessment, and the efficiency gained by automating reconnaissance tasks with Python. The exercise demonstrated that even a standard localhost environment can be running services with potential security considerations that require analysis and hardening.