Task 01 — Networking Fundamentals, Nmap Scanning, and Automation Scripting
Author: Yash Chandrashekhar Karnik
Date: 2025-10-27

Objective
Summarise and document the work required for Task 01: install and use Nmap to scan an authorized target, analyze results, create a network diagram, research two services and their common vulnerabilities, and produce a Python automation script that performs a SYN scan and writes a report.

Environment / Target (example)
Target name: Local test VM (Metasploitable-like)

- IP: 192.168.0.103 (real test target provided)
- Network: Lab NAT/Host-only network (ensure authorized access)

Repository: https://github.com/yk47/Networking-Fundamentals-Nmap-Scanning-and-Automation-Scripting (source files and scan outputs).

---

1) Install and verify Nmap

1. Download and install: https://nmap.org/
2. Verify installation in terminal:
   nmap -v
3. (Optional) Install python-nmap for automation:

pip install python-nmap

---

2) Nmap scans to run (commands)
Run these on your authorized target and save outputs to files.

- **SYN scan (stealth):**
  - nmap -sS 192.168.0.103 -oN syn_scan.txt
- **TCP connect scan (full connect):**
  - nmap -sT 192.168.0.103 -oN tcp_scan.txt
- **UDP scan:**
  - nmap -sU 192.168.0.103 -oN udp_scan.txt

Note: Aggressive (version, scripts, OS detection, all ports): sudo nmap -sV -sC -O -p- 192.168.0.103 -oN aggressive_scan.txt

---

3) How to analyze scan outputs
Look in the saved .txt files for lines like:
PORT STATE SERVICE VERSION
22/tcp open ssh OpenSSH 7.2p2 Ubuntu 4ubuntu2.8
80/tcp open http Apache httpd 2.4.18
139/tcp open netbios-ssn Samba smbd 3.X
Create a table of findings listing port, protocol, state, service, and version (if present). See example below.

4) Example findings table (replace with your actual scan output)

Port	State	Service	Product	Version	Notes
21	open	ftp	vsftpd	2.3.4	Anonymous/backdoor historic risk
22	open	ssh	OpenSSH	4.7p1 Debian 8ubuntu1	Very old OpenSSH
23	open	telnet	telnetd	-	Cleartext authentication
25	open	smtp	Postfix	-	Mail server; check relaying
53	open	domain	ISC BIND	9.4.2	Old BIND version
80	open	http	Apache httpd	2.2.8 (Ubuntu) DAV/2	Outdated Apache
111	open	rpcbind	rpcbind	2	RPC service
139	open	netbios-ssn	Samba smbd	3.X - 4.X	SMB exposure
445	open	netbios-ssn	Samba smbd	3.X - 4.X	SMB exposure
512	open	exec	netkit-rsh rexecd	-	rsh/rlogin services
513	open	login	rlogind	-	rlogin
514	open	tcpwrapped	-	-	tcpwrapped
1099	open	java-rmi	GNU Classpath grmiregistry	-	RMI service
1524	open	bindshell	Metasploitable root shell	-	Backdoor/bindshell
2049	open	nfs	nfs	2-4	NFS exports
2121	open	ftp	ProFTPD	1.3.1	Alternate FTP
3306	open	mysql	MySQL	5.0.51a-3ubuntu5	Very old MySQL
5432	open	postgresql	PostgreSQL DB	8.3.x	Old Postgres
5900	open	vnc	VNC	-	VNC protocol 3.3
6000	open	X11	X11	-	X11 display (access denied)
6667	open	irc	UnrealIRCd	-	Historic vulnerable IRC daemon
8009	open	ajp13	Apache Jserv	-	AJP connector
8180	open	http	Apache Tomcat/Coyote	1.1	Tomcat HTTP

5) Two researched services (short notes)

Service 1: MySQL (port 3306)

- Purpose: Relational database server. Risks: Very old MySQL 5.0.51a has multiple known vulnerabilities; weak/default credentials may allow full DB compromise. Mitigations: Update to supported versions, restrict access, use strong passwords and TLS.

Service 2: vsftpd (port 21)

- Purpose: FTP server. Risks: vsftpd 2.3.4 has historical backdoor incidents and transmits credentials in cleartext; anonymous FTP can leak files. Mitigations: Use SFTP (via SSH), disable anonymous access, or configure FTPS/TLS.

For a real assessment, check CVE entries for the detected service/version (use NVD, MITRE CVE or vendor advisories).

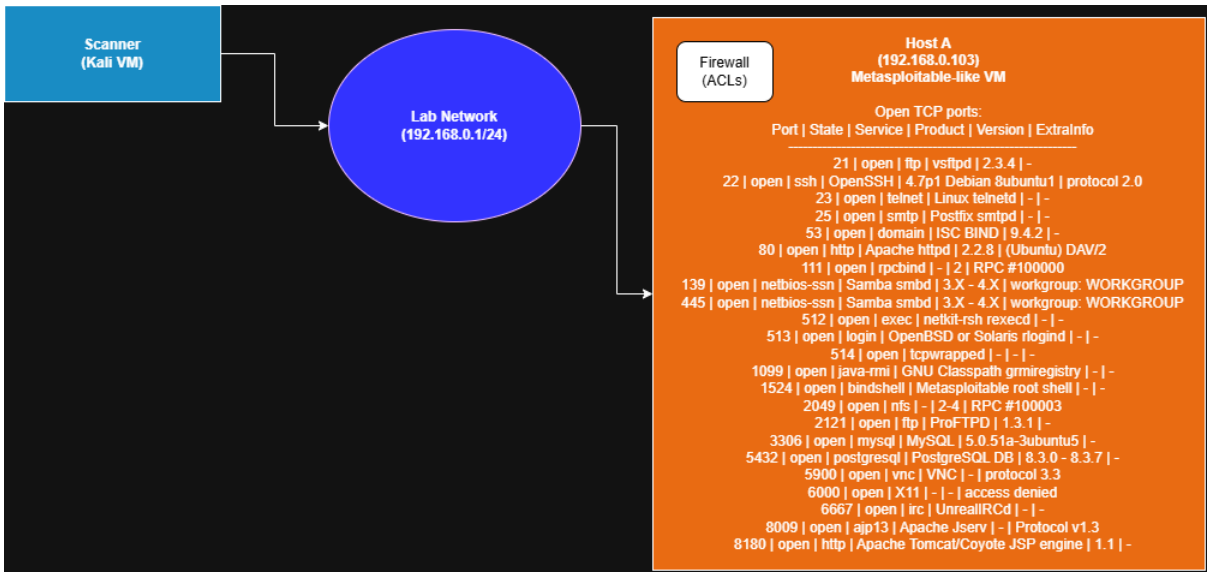
---

6) Network diagram

Diagram link (editable):

https://drive.google.com/file/d/17bRnm4C_eO9hMIltNhkn4n1xhrkAQZ4u/view?usp=sharing

Network diagram image not found locally; link provided above.



---

7) Sample Report structure (what to include in task01_report.pdf)

High-priority items:

- Disable Telnet/rsh/rlogin (ports 23, 512-514) and replace with SSH (key-based auth).
- Patch or remove obsolete services (MySQL 5.0, Apache 2.2.8, BIND 9.4.2, UnrealIRCd).
- Firewall and restrict access to database and admin ports to trusted IPs only.
- Isolate vulnerable lab VMs on a separate network (no internet exposure).

8) Python automation script

Below is a ready-to-run Python script using python-nmap that performs a SYN scan (-sS) and writes a scan_report.txt file. Replace target input with your authorized host.

```
#!/usr/bin/env python3
"""
nmap_automation.py

Performs a SYN scan (-sS) using python-nmap and writes a text report.
Requires: nmap installed on system and python-nmap library (`pip install python-nmap`).
"""

import nmap
import argparse
from datetime import datetime


def run_syn_scan(target, ports=None, timeout=120):
    """Run an nmap SYN scan (-sS) against target. Optionally specify comma-separated ports."""
    nm = nmap.PortScanner()

    # Build nmap arguments: -sS for SYN scan, -sV for service/version
    nmap_args = '-sS -sV'
    if ports:
        scan_target = f"{target} -p {ports}"
    else:
        scan_target = target

    print(f"Running: nmap {nmap_args} {scan_target}")
    # run the scan (scan() accepts arguments string for nmap command)
    nm.scan(hosts=target, arguments=nmap_args)
    return nm


def parse_scan(nm, target):
    """Parse PortScanner object and return structured results for target."""
    results = {
        'host': target,
        'state': None,
        'addresses': {},
        'ports': []
    }

    if target in nm.all_hosts():
        host_info = nm[target]
```

```
        results['state'] = host_info.state()
        if 'addresses' in host_info:
            results['addresses'] = host_info['addresses']

        # iterate over protocols (tcp, udp)
        for proto in host_info.all_protocols():
            lports = host_info[proto].keys()
            for port in sorted(lports):
                port_info = host_info[proto][port]
                results['ports'].append({
                    'port': port,
                    'protocol': proto,
                    'state': port_info.get('state', ''),
                    'service': port_info.get('name', ''),
                    'product': port_info.get('product', ''),
                    'version': port_info.get('version', ''),
                    'extrainfo': port_info.get('extrainfo', '')
                })
    else:
        results['state'] = 'down or no response'

    return results


def write_report(report_file, nm_results):
    """Write a human-readable text report to report_file."""
    with open(report_file, 'w') as f:
        f.write(f"Scan timestamp: {datetime.utcnow().isoformat()}Z\n")
        f.write(f"Target: {nm_results['host']}\n")
        f.write(f"Host state: {nm_results['state']}\n\n")

        f.write("Open ports and services:\n")
        f.write("Port\tProto\tState\tService\tProduct\tVersion\n")
        for p in nm_results['ports']:

f.write(f"{p['port']}\t{p['protocol']}\t{p['state']}\t{p['service']}\t{p['product']}\t{p['version']}\n")

        f.write('\nCompletion: Scan finished.\n')

    print(f"Report written to {report_file}")


def main():
    parser = argparse.ArgumentParser(description='Nmap SYN scan automation (python-
nmap)')
```

```
    parser.add_argument('target', help='Target IP or hostname to scan')
    parser.add_argument('--ports', help='Optional comma-separated port range or list (e.g. 1-1000 or 22,80,443)')
    parser.add_argument('--out', default='scan_report.txt', help='Output report filename')

    args = parser.parse_args()

    # Run scan
    nm = run_syn_scan(args.target, ports=args.ports)

    # Parse results
    nm_results = parse_scan(nm, args.target)

    # Write text report
    write_report(args.out, nm_results)


if __name__ == '__main__':
    main()
```

How to run:
python3 nmap_automation.py 192.168.56.101 --ports 1-1024 --out scan_report.txt
This will produce scan_report.txt with timestamp, target, and a table of open ports and detected services.

---

9) Example scan_report.txt (sample content)
Included files in repository (see GitHub): syn_scan.txt, tcp_scan.txt, udp_scan.txt, scan_report.txt, nmap_automation.py, os_detect.txt, and screenshots. (If you need these embedded in the Word file, upload them to the workspace or push to the repo and I will include them.)
Repository URL: https://github.com/yk47/Networking-Fundamentals-Nmap-Scanning-and-Automation-Scripting

---

Report prepared by: Yash Chandrashekhar Karnik
Generated: 2025-10-27 14:16:23
