Task 1 — Local Network Port Scan

Comprehensive Report and Source Code Documentation

README Overview

~~~
# Task 1 — Local Network Port Scan

Objective-
Discover open ports on devices in your local network and produce a concise report that explains findings and potential security exposure.

What this repo contains-
- `nmap_scan.sh` — ready-to-run shell script to scan your local network and save results.
- `parse_nmap.py` — Python script that parses `nmap` grepable output (`.gnmap`) and generates `scan_summary.csv` and `scan_summary.html`.
- `sample_scan.gnmap` — a sample `nmap` grepable output for testing the parser (no real network scanning included).
- `report_template.md` — a short report template you can fill and submit.
- `images/` — includes `network_diagram.png` and `ports_chart.png` illustrating the task.
- `.gitignore` and `LICENSE`.

**Important notes before running**
- Do **not** scan networks you do not own or have explicit permission to scan.
- Run this on your home or lab network only.
- The shell script requires `nmap` installed and should be run from a UNIX-like shell with appropriate privileges (use `sudo` if needed).

---

## Quick steps to produce results (local machine)

1. Make the script executable:
```bash
chmod +x nmap_scan.sh
```

2. Edit `nmap_scan.sh` to set your local IP range (default example `192.168.1.0/24`) if needed.

3. Run the scan (example; may require sudo):
```bash
./nmap_scan.sh
```
This produces:
- `scan_results.gnmap` (grepable)
- `scan_results.xml` (XML)
- `scan_results.txt` (normal output)

4. Parse results:
```bash
python3 parse_nmap.py scan_results.gnmap
```
This creates:
- `scan_summary.csv`
- `scan_summary.html`

5. Fill `report_template.md` with your findings, add screenshots from `images/`, then push to GitHub.

---
~~~

Source Code: nmap_scan.sh

```bash
#!/usr/bin/env bash
# nmap_scan.sh
# Usage: ./nmap_scan.sh [IP_RANGE]
# Example: ./nmap_scan.sh 192.168.1.0/24

IP_RANGE=${1:-192.168.1.0/24}
OUT_PREFIX="scan_results"

echo "[*] Starting Nmap TCP SYN scan (requires privileges for -sS)..."
echo "[*] Target range: $IP_RANGE"

# Full port scan (-p-) with TCP SYN (-sS), treat hosts as up if they respond to ping (-Pn is omitted)
# Save grepable and XML outputs for parsing
sudo nmap -sS -p- --min-rate 1000 -T4 -oG ${OUT_PREFIX}.gnmap -oX ${OUT_PREFIX}.xml ${IP_RANGE} | tee ${OUT_PREFIX}.txt

echo "[*] Scan complete. Outputs written to ${OUT_PREFIX}.gnmap, ${OUT_PREFIX}.xml, ${OUT_PREFIX}.txt"
echo "[*] Parse the grepable output with: python3 parse_nmap.py ${OUT_PREFIX}.gnmap"
```

Source Code: parse_nmap.py

```python
#!/usr/bin/env python3
"""parse_nmap.py

Parses an nmap grepable output (.gnmap) and produces:
- scan_summary.csv (host,ip,port,state,service)
- scan_summary.html (simple HTML report)

Usage:
    python3 parse_nmap.py sample_scan.gnmap
"""

import sys, csv, html
from pathlib import Path

if len(sys.argv) < 2:
    print('Usage: python3 parse_nmap.py <scan.gnmap>')
    sys.exit(1)

gnmap_file = Path(sys.argv[1])
if not gnmap_file.exists():
    print('File not found:', gnmap_file)
    sys.exit(1)

hosts = []
with gnmap_file.open() as fh:
    for line in fh:
        line=line.strip()
        if not line or line.startswith('#'):
            continue
        # gnmap host line format: Host: 192.168.1.10 ()    Status: Up
        # or: Host: 192.168.1.10 ()  Ports: 22/open/tcp//ssh///,80/open/tcp//http///  Ignored State: closed (997)
        if line.startswith('Host:'):
            parts = line.split('\t')
            host_part = parts[0]
            ip = host_part.split()[1]
            ports = ''
            for p in parts:
                if p.startswith('Ports:'):
                    ports = p[len('Ports:'):].strip()
            if ports:
                for entry in ports.split(','):
                    entry = entry.strip()
                    if not entry:
                        continue
                    # entry example: 22/open/tcp//ssh///
                    f = entry.split('/')
                    if len(f) >= 3:
                        port = f[0]
                        state = f[1]
                        proto = f[2]
                        service = f[4] if len(f) > 4 else ''
                        hosts.append({'ip': ip, 'port': port, 'state': state, 'proto': proto, 'service': service})
# Write CSV
```

```
csv_file = gnmap_file.with_name('scan_summary.csv')
with csv_file.open('w', newline='') as csvf:
    writer = csv.DictWriter(csvf, fieldnames=['ip','port','proto','state','service'])
    writer.writeheader()
    for h in hosts:
        writer.writerow(h)

# Write simple HTML
html_file = gnmap_file.with_name('scan_summary.html')
with html_file.open('w') as fh:
    fh.write('<!doctype html><html><head><meta charset="utf-8"><title>Scan Summary</title></head><body>')
    fh.write('<h1>Scan Summary</h1>')
    fh.write('<table border="1" cellpadding="6" cellspacing="0">')
    fh.write('<tr><th>IP</th><th>Port</th><th>Protocol</th><th>State</th><th>Service</th></tr>')
    for h in hosts:
        fh.write(f"<tr><td>{html.escape(h['ip'])}</td><td>{html.escape(h['port'])}</td><td>{html.escape(h['proto'])}</td><td>{html.escape(h['state'])}</td><td
    fh.write('</table></body></html>')

print('[*] Parsed', len(hosts), 'open/filtered ports.')
print('[*] Outputs:', csv_file.name, html_file.name)
```

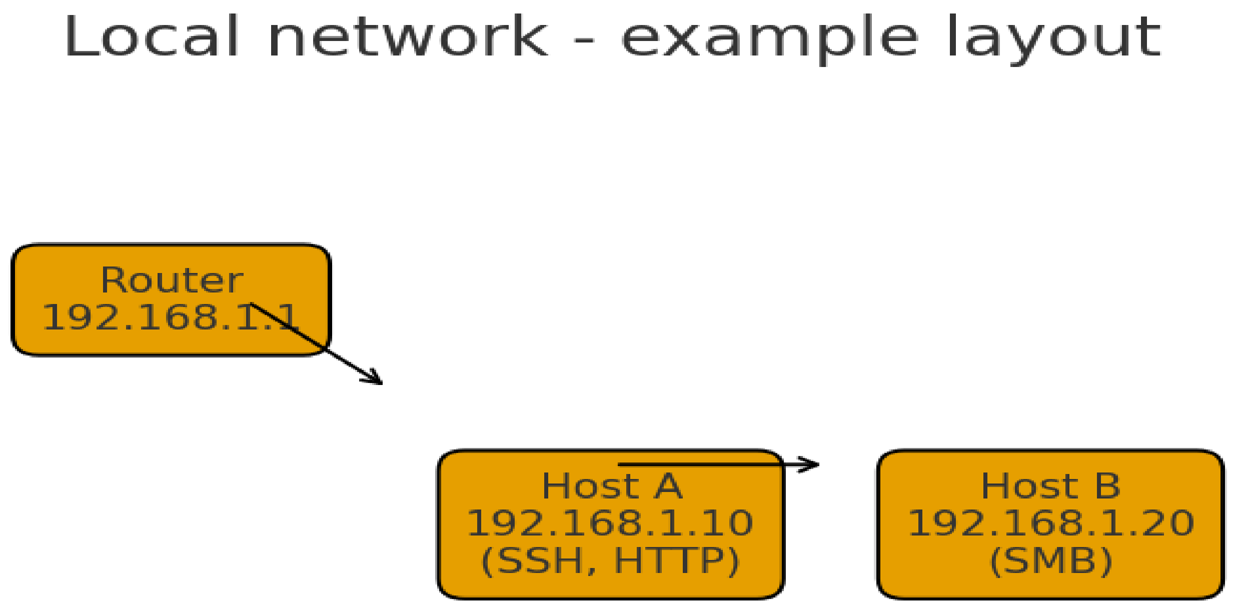
Sample Scan Output (sample_scan.gnmap)

```
# Nmap grepable sample (simulated)
# Generated for demo purposes only
Host: 192.168.1.1 ()    Status: Up
Host: 192.168.1.10 ()   Status: Up   Ports: 22/open/tcp//ssh///,80/open/tcp//http///,631/closed/tcp//ipp///
Host: 192.168.1.15 ()    Status: Up    Ports: 53/open/udp//domain///,161/closed/udp//snmp///
Host: 192.168.1.20 ()    Status: Up    Ports: 139/open/tcp//netbios-ssn///,445/open/tcp//microsoft-ds///
```

Report Template

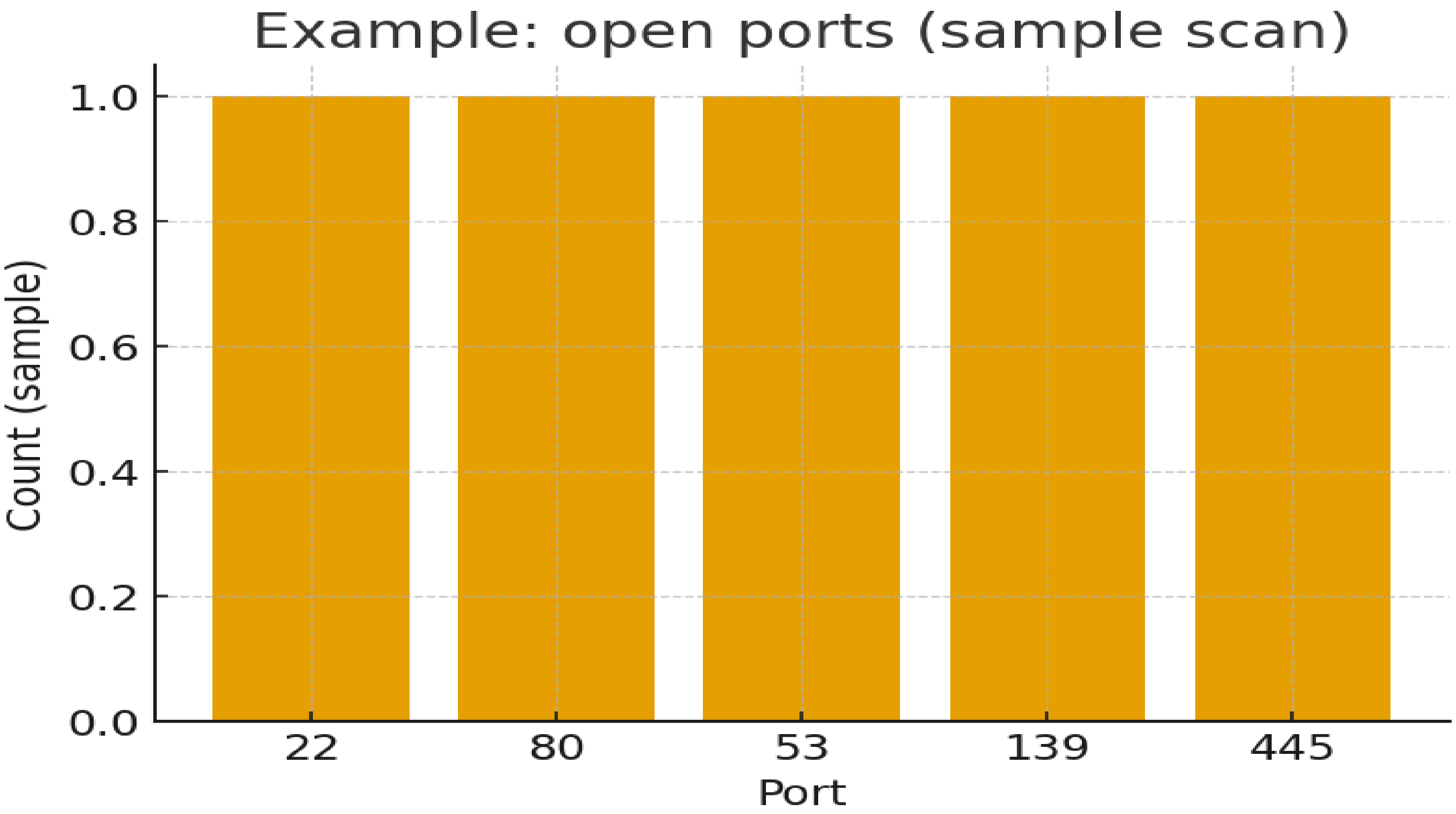
```
# Scan Report - Task 1

**Target range:** _fill in the IP range scanned_

**Date:** _fill in date_

## Summary
_One-paragraph summary of the scan and number of hosts discovered with open ports._

## Findings
- Host: 192.168.1.10 — open ports: 22 (ssh), 80 (http)
- Host: 192.168.1.20 — open ports: 139 (netbios-ssn), 445 (microsoft-ds)

## Potential Risks
- Services like netbios and SMB (ports 139/445) can expose file shares; ensure they are patched and not exposed to untrusted networks.
- SSH and HTTP should enforce strong authentication and up-to-date software.

## Remediation / Recommendations
- Disable unnecessary services.
- Apply OS and application patches.
- Use host-based and network firewalls to restrict access.
- Use strong credentials and remove default accounts.

## Evidence
- Attach `scan_summary.html`, screenshots, and `scan_results.txt` as evidence.
```

Network diagram.png



Local network - example layout

Ports chart.png



License