Scan Your Local Network for Open Ports

Install Nmap from Official Website

- Go to: https://nmap.org/download.html
- Download and install the appropriate version for your OS:
- **Windows:** Includes Zenmap (GUI) optionally.
- **Linux/macOS:** You can also install via package manager:
- **Ubuntu:** sudo apt install nmap
- **macOS (Homebrew):** brew install nmap

Find Your Local IP Range

Open a terminal or command prompt and run:

- **Windows:**
  ipconfig
- **Linux/macOS:**
  ifconfig or ip a

Identify your local IP (e.g., 192.168.1.5) and subnet mask.

Based on subnet mask, determine IP range:

If subnet is 255.255.255.0, your range is likely 192.168.1.0/24

Run TCP SYN Scan

nmap -sS 192.168.1.0/24

- -sS: Stealth SYN scan (doesn't complete full TCP handshake).
- Requires root/admin privileges on Unix systems.

Note Down IPs and Open Ports

- Nmap will output active hosts and the ports that are open on each.
- Example output snippet:

  Nmap scan report for 192.168.1.10

PORT	STATE SERVICE
22/tcp	open ssh
80/tcp	open http

Analyse Packets with Wireshark

Open Wireshark.

Start capturing on the network interface.

While scanning, observe traffic to/from your IP.

Filter examples:

- tcp.port == 80
- ip.addr == 192.168.1.10

Research Common Services

Look up each open port to determine service:

22 → SSH

80 → HTTP

443 → HTTPS

3389 → Remote Desktop

- **Use: https://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xhtml**

Identify Potential Security Risks

- **Check for:**
  - Open ports that shouldn't be accessible (e.g., Telnet 23)
  - Services with default or weak authentication
  - Unpatched services (outdated versions)
- **Consider using nmap -sV to detect service versions:**

  nmap -sV 192.168.1.10

  Save Scan Results
- **Text File**

  nmap -sS 192.168.1.0/24 -oN scan_results.txt

HTML File (requires xsltproc):

nmap -sS 192.168.1.0/24 -oX scan_results.xml

xsltproc scan_results.xml -o scan_results.html

Port Scanning

Port scanning is the act of probing a host or range of hosts to discover open ports and the services running on them. It's a fundamental step in network reconnaissance (foot printing), used by both penetration testers and attackers.

Types of Port Scans:

TCP Connect Scan (-sT): Completes the TCP handshake (less stealthy).

TCP SYN Scan (-sS): Half-open scan (sends SYN, waits for SYN-ACK, then resets).

UDP Scan (-sU): Probes for open UDP ports (slower, more complex).

ACK, Xmas, NULL, FIN Scans: Used to bypass certain firewall rules or detect filtering.

TCP SYN Scan (Half-Open Scan)

A TCP SYN scan is the most common and efficient technique used by nmap for stealthy port scanning.

How it works:

- Sends a SYN packet to a target port.
- If SYN-ACK is received → port is open.
- If RST is received → port is closed.
- If no response or ICMP unreachable → filtered (likely blocked by firewall).
- The scanner never completes the TCP handshake, hence "half-open."

Why it's stealthier:

- It avoids logging in some basic systems because it doesn’t complete the connection**.**

IP Ranges and CIDR Notation

CIDR	Subnet Mask	IP Range Size	Example Range
/24	**255.255.255.0**	**256 IPs**	**192.168.1.0 – 192.168.1.255**
/16	**255.255.0.0**	**65,536 IPs**	**192.168.0.0 – 192.168.255.255**
/30	**255.255.255.252**	**4 IPs**	**Often used for point-to-point**

TCP Scanning

- Protocol: Connection-oriented (3-way handshake)
- Command: nmap -sS (SYN scan)
- Speed: Fast and reliable
- Response logic:
  - SYN-ACK → port is open
  - RST → port is closed
- Detection: Can be logged by IDS/IPS
- Use Cases: Detects web servers, SSH, FTP, RDP, mail, etc.

UDP Scanning

- Protocol: Connectionless (no handshake)
- Command: nmap -sU
- Speed: Slow due to timeouts
- Response logic:
  - ICMP Port Unreachable → port is closed
  - No response → port is open filtered
  - Valid UDP reply → port is open
- Detection: Often stealthier, harder to detect
- Use Cases: Finds DNS, SNMP, NTP, DHCP, etc.

How can open ports be secured?

1. Close Unused Ports

- Regularly audit open ports with tools like nmap or netstat.
- Disable or uninstall services you don't need.

sudo systemctl disable unused-service

2. Use Firewalls to Restrict Access

- Block unnecessary inbound and outbound traffic.
- Only allow specific IP ranges (whitelisting).
- Examples:
    - **iptables**, **ufw**, **pfSense**, **Windows Defender Firewall**.

    sudo ufw allow from 192.168.1.0/24 to any port 22 proto tcp

    sudo ufw deny 23/tcp # Block Telnet

3. Apply Access Control Lists (ACLs)

- Restrict access to certain services (e.g., SSH, RDP) by IP or role.
- Configure **router, switch, or firewall ACLs** to limit port reachability.

4. Use Secure Protocols

- Replace insecure services:
    - Replace **Telnet** (port 23) → with **SSH** (port 22)
    - Replace **FTP** (port 21) → with **SFTP** or **FTPS**
    - Replace **HTTP** (port 80) → with **HTTPS** (port 443)

5. Enable Authentication and Encryption

- For exposed services:
    - Enforce **strong passwords or keys**
    - Use **MFA** where supported
    - Apply **TLS/SSL encryption**

6. Change Default Ports (Obfuscation)

- Not security by itself, but reduces automated attacks.
    - e.g., Move SSH from port 22 to 2222 or higher.

sudo nano /etc/ssh/sshd_config # Change Port 22 to 2222

7. Patch and Update Regularly

- Keep services behind open ports **fully updated**.
- Vulnerable versions (e.g., old Apache, SMBv1) are easily exploited.

8. Monitor and Log Port Activity

- Use tools like:
    - **Fail2Ban** to ban IPs after failed login attempts.
    - **IDS/IPS** like Snort or Suricata.
    - **SIEM** to track and alert on port usage anomalies.

9. Use Port Knocking or VPN for Sensitive Services

- **Port Knocking**: Keeps a port closed until a specific sequence of connection attempts is made.
- **VPN**: Only expose sensitive services (e.g., RDP, SSH) inside a secure VPN tunnel.

10. Implement Network Segmentation

- Keep critical services in separate **zones** (e.g., DMZ, internal VLANs).
- Use firewalls to control inter-zone traffic.

Core Role of a Firewall Regarding Ports

1. Block or Allow Traffic on Specific Ports

- Firewalls can **allow (open)** or **block (close)** ports based on rules.
- Example: Block port **23** (Telnet), allow port **443** (HTTPS).

2. Filter Traffic Based on Source/Destination

- Limit access to ports by:
    - **Source IP**: e.g., only allow SSH on port 22 from internal IPs.
    - **Destination IP**: e.g., deny traffic to untrusted servers.
    - **Port ranges or specific ports**: fine-tuned control.

3. Prevent Unauthorized Port Access

- Stops **unauthorized scans**, **exploit attempts**, or **brute-force attacks** by blocking unused or vulnerable ports.
- Can **detect and alert** on suspicious port activity (e.g., port scanning).

4. Enable Segmentation and Zone-Based Security

- Firewalls define boundaries (e.g., between a **DMZ**, **internal network**, and the **internet**) and control which ports are allowed between zones.

5. Log and Monitor Port Activity

- Logs all **accepted/denied traffic** for each port.
- Useful for:
    - **Incident response**
    - **Security auditing**
    - **Forensics**

Examples

On a Host Firewall (e.g., Linux UFW):

sudo ufw allow 22/tcp # Allow SSH

sudo ufw deny 23/tcp # Block Telnet

On a Network Firewall (e.g., Cisco, Fortinet):

access-list 100 deny tcp any any eq 23

access-list 100 permit tcp any any eq 443

What Is a Port Scan?

A port scan is a technique used to probe a target system or network to discover which ports are open, closed, or filtered, and what services are running on those ports.

Think of it like knocking on every door in a building to see which ones are unlocked — but in the digital world.

What Does a Port Scan Reveal?

A port scan can uncover:

- Open ports (e.g., 22 for SSH, 80 for HTTP)
- Services behind those ports (e.g., Apache, MySQL)
- Service versions (with advanced scans)
- Operating system fingerprints
- Misconfigured or vulnerable services

Why Do Attackers Perform Port Scans?

Attackers use port scans during the reconnaissance phase to map the target's attack surface and identify potential vulnerabilities.

Why It's Dangerous (If You're Defending)

- Exposed ports can reveal **critical services** attackers can exploit.
- Misconfigured services may allow **unauthenticated access**.
- Unpatched versions may contain **known vulnerabilities** (CVEs).
- Silent scans may **go unnoticed** without proper monitoring (e.g., stealthy SYN scans).

How to Defend Against Port Scans

- Use **firewalls** to block unused ports.
- Implement **intrusion detection systems (IDS)** like **Snort** or **Suricata**.
- Monitor logs for **scan signatures** (e.g., many SYNs to many ports/IPs).
- Use **rate-limiting** and **honeypots** to trap or mislead attackers.