HTB Academy — Nmap Module

Network Enumeration with Nmap

Author: Calm Ay (Rasaq Ayomide)

Platform: Hack The Box Academy

Module: Network Enumeration with Nmap

Date: May 30, 2026

Topics: Host Discovery, Port Scanning, Service Enumeration, NSE, IDS/IPS Evasion

---

1. Host Discovery

Host discovery is the first step in any network engagement. Before scanning ports, we need to identify which hosts are alive on the network. Nmap provides multiple methods depending on the network environment and firewall configurations.

Scan entire network range:

```
sudo nmap 10.129.2.0/24 -sn -oA tnet | grep for | cut -d' ' -f5
```

Scan from IP list:

```
sudo nmap -sn -oA tnet -iL hosts.lst | grep for | cut -d' ' -f5
```

Single host with ICMP echo + packet trace:

```
sudo nmap 10.129.2.18 -sn -oA host -PE --packet-trace --disable-arp-ping
```

> **Key insight:** By default Nmap uses ARP ping on local networks before ICMP. Use --disable-arp-ping to force ICMP echo requests. TTL values in ICMP replies reveal the OS: TTL=64 (Linux/Unix), TTL=128 (Windows), TTL=255 (Cisco).

Lab Question — OS Identification via TTL:

TTL=128 in ICMP reply → OS: Windows

2. Host and Port Scanning

After confirming hosts are alive, the next step is identifying open ports and their states. Nmap has 6 port states: open, closed, filtered, unfiltered, open|filtered, closed|filtered.

Port states reference:

open: Connection established — service is running

closed: RST flag received — port accessible but no service

filtered: No response or error — firewall likely blocking

unfiltered: Port accessible but state undetermined (TCP-ACK scan only)

open|filtered: No response — firewall or filter may be present

closed|filtered: Only in IP ID idle scans — cannot determine state

Common scanning commands:

```
# Top 10 ports
sudo nmap 10.129.2.28 --top-ports=10

# SYN scan with packet trace
sudo nmap 10.129.2.28 -p 21 --packet-trace -Pn -n --disable-arp-ping

# Connect scan (full TCP handshake)
sudo nmap 10.129.2.28 -p 443 -sT --packet-trace -Pn -n

# UDP scan (top 100 ports)
sudo nmap 10.129.2.28 -F -sU
```

> **SYN vs Connect scan:** SYN scan (-sS) is stealthier — doesn't complete the handshake. Connect scan (-sT) is more accurate but logs on target. SYN scan requires root privileges for raw packet creation.

3. Saving Scan Results

Always save scan output — critical for documentation, comparison, and reporting. Nmap supports three output formats.

```
# Save all formats at once
sudo nmap 10.129.2.28 -p- -oA target

# Convert XML to HTML report
xsltproc target.xml -o target.html
```

Normal (.nmap): Human-readable text output

Grepable (.gnmap): One line per host, easy to grep/parse

XML (.xml): Machine-readable, convertible to HTML with xsltproc

> Lab Answer: Highest TCP port found = 31337

4. Service Enumeration

Service version detection identifies exact software and versions running on open ports. This is critical for finding matching CVEs and public exploits.

```
# Full port scan with version detection
sudo nmap 10.129.2.28 -p- -sV
```

```
# With progress stats every 5 seconds
sudo nmap 10.129.2.28 -p- -sV --stats-every=5s

# Verbose - shows open ports as discovered
sudo nmap 10.129.2.28 -p- -sV -v
```

Banner Grabbing with netcat:

```
nc -nv 10.129.2.28 25
# Returns: 220 inlane ESMTP Postfix (Ubuntu)
```

Key insight: Nmap doesn't always show full banner information. Always manually banner grab with nc on interesting ports — extra details like OS distribution (Ubuntu, Debian) are often revealed in the raw banner but not in Nmap's standard output.

Lab — Flag in service banner:

Port 31337 returned 'Elite?' as service name — Nmap couldn't identify it. Banner grabbing with nc revealed the flag:

```
nc -nv 10.129.20.254 31337
# Returns: 220 HTB{pr0F7pDv3r510nb4nn3r}
```

HTB{pr0F7pDv3r510nb4nn3r}

5. Nmap Scripting Engine (NSE)

NSE allows Lua scripts to interact with services for deeper enumeration, vulnerability detection, and exploitation. 14 script categories available.

NSE usage examples:

```
# Default scripts
sudo nmap <target> -sC

# Specific scripts
sudo nmap <target> --script banner,smtp-commands

# Script category
sudo nmap <target> --script vuln

# Aggressive scan (sV + OS + traceroute + sC)
sudo nmap <target> -p 80 -A
```

Vulnerability scan on web port:

```
sudo nmap 10.129.20.254 -p 80 -sV --script vuln
```

Finding: http-enum script identified /robots.txt. Always follow up on every path http-enum finds — robots.txt contained the flag.

```
curl http://10.129.20.254/robots.txt
# Returns: HTB{873nniuc71bu6usbs1i96as6dsv26}
```

HTB{873nniuc71bu6usbs1i96as6dsv26}

6. Firewall and IDS/IPS Evasion

When targets are protected by IDS/IPS systems, standard scans trigger alerts. Evasion techniques minimize noise and bypass detection.

6.1 Easy Lab — OS Detection (Ubuntu)

Standard OS detection was blocked. Used stealthy timing and disabled noisy probes. SSH banner revealed the OS distribution more reliably than nmap OS guesses:

```
# Stealthy OS scan
sudo nmap 10.129.2.80 -O -Pn -n --disable-arp-ping -T2 --source-port 53

# Banner grab SSH for exact OS
nc -nv 10.129.2.80 22
# Returns: SSH-2.0-OpenSSH_7.6p1 Ubuntu-4ubuntu0.7
```

Answer: Ubuntu (from SSH banner — nmap only said Linux 98%)

6.2 Medium Lab — DNS Version via UDP (Flag)

DNS runs on UDP port 53. The version.bind CHAOS TXT query retrieves DNS server version. DNS traffic is typically whitelisted by firewalls making this a reliable evasion technique:

```
dig version.bind CHAOS TXT @10.129.2.48
```

HTB{GoTtgUnyze9Psw4vGjcuMpHRp}

6.3 Hard Lab — Source Port 53 Evasion (Flag)

Stricter firewall only allowed traffic from source port 53 (DNS). FTP service was moved to non-standard port 50000. Used --source-port 53 in nmap to discover the port, then ncat with source port 53 bound to tun0 interface to retrieve the banner:

```
# Discover open ports with source port 53
sudo nmap 10.129.2.47 -p- -Pn -n --disable-arp-ping --source-port 53 -T2
# Found: port 50000 open

# Connect with source port 53 bound to tun0
sudo ncat --source-port 53 -s 10.10.14.69 10.129.2.47 50000
```

Issue encountered: dnsmasq was using port 53 on 0.0.0.0. Fix: bind ncat to tun0 IP specifically with -s flag to avoid conflict.

HTB{kjnsdf2n982n1827eh76238s98di1w6}

6.4 Evasion Techniques Reference

-T0 to -T2: Slow timing — reduces scan speed to avoid rate-based detection

-Pn: Skip host discovery — avoids ICMP alerts

--disable-arp-ping: Prevents ARP broadcast alerts on local networks

-n: No DNS resolution — reduces outbound traffic

--source-port 53: Disguise traffic as DNS — usually whitelisted by firewalls

-D RND:5: Add random decoy IPs — confuses IDS attribution

--data-length 25: Append random data to packets — breaks signature matching

-f: Fragment packets — evades some packet inspection systems

--scan-delay: Add delay between probes — avoids rate-limit triggers

7. Key Takeaways

Always save scan output: Use -oA to save all formats — critical for documentation and comparison

TTL reveals OS: TTL=64→Linux, TTL=128→Windows, TTL=255→Cisco. Decreases by 1 per hop.

Banner grab manually: nc often reveals more than nmap — OS distro, exact versions, hidden flags

NSE http-enum finds paths: Always follow up every path http-enum discovers — check robots.txt, readme.html

Source port 53 bypasses firewalls: DNS traffic (port 53) is commonly whitelisted — use as source port for evasion

Non-standard ports: Services get moved off default ports — always run full -p- scans

DNS version.bind query: dig version.bind CHAOS TXT reveals DNS server version via UDP

UDP is often overlooked: Admins forget UDP ports — always run -sU scan on key ports (53, 161, 137)

Tools Used: Nmap, netcat (nc), ncat, dig, xsltproc

Written by Calm Ay — linkedin.com/in/rasaq-ayomide-sec