Nmap Full Tutorial for Beginners

Complete Course Notes — Network Reconnaissance, Scanning Techniques & Scripting Engine

SECTION 1: Introduction to Nmap

1.1 What is Nmap?

Nmap (Network Mapper) is one of the most well-known and widely used open-source network reconnaissance and auditing tools in existence. It was first released in September 1997 and has since become an essential tool for network administrators, security professionals, and penetration testers worldwide.

- Full name: Network Mapper
- First released: September 1997
- Purpose: mapping networks, discovering hosts, scanning ports, detecting services and OS types, and evading firewalls
- Famous pop-culture appearances: featured in films including The Matrix as a real-world hacking tool
- Written in C, C++, Python, and Lua
- Open source and free to download from nmap.org

1.2 Platform Support

Platform	Notes
Linux	Nmap's native platform — born on Linux; full feature support; requires root/sudo for advanced scans
macOS (Mac OS X)	Full support via Terminal — used throughout this course
Windows	Available as a Windows executable; some features require Administrator privileges
Kali Linux	A Linux distribution specifically designed for penetration testing; Nmap is pre-installed along with hundreds of other security tools
Virtual Machine (VirtualBox, VMware)	Run a Linux VM on any host OS to get full Linux Nmap capability without dual-booting

1.3 Zenmap — Nmap's GUI Interface

- Zenmap is the official graphical user interface (GUI) for Nmap
- Suitable for users who prefer not to use the command line
- Not covered in this course — the course focuses exclusively on the Nmap CLI for full control and understanding
- Available as a free download from nmap.org alongside the standard installer

1.4 The Main Nmap Help Menu

Typing Nmap and pressing Enter (with no arguments) displays the full help menu. This is the first thing to do when learning Nmap. The menu groups options into the following categories:

- Host Discovery — how to find live hosts on a network
- Scan Techniques — which type of TCP/UDP scan to use
- Port Specification — which ports to target
- Service/Version Detection — determine what software is running on each port
- Script Scanning (NSE) — powerful scripting engine for advanced tasks
- OS Detection — identify the operating system of the target
- Timing and Performance — control scan speed for firewall evasion
- Firewall/IDS Evasion — techniques to bypass security controls
- Output — save results to text, greppable, or XML files
- Examples — built-in command examples to help beginners get started

Important reminder: Always obtain explicit permission before scanning any network or device you do not own. Unauthorized scanning is illegal in most jurisdictions.

SECTION 2: TCP Fundamentals — Understanding the Foundation of Nmap Scanning

2.1 The TCP Three-Way Handshake

To understand how Nmap scanning works, you must first understand how a normal TCP connection is established. The Three-Way Handshake is the process by which two hosts negotiate and establish a TCP connection:

Step	Who Sends It
Step 1 — SYN	Host A (initiator)
Step 2 — SYN-ACK	Host B (responder)
Step 3 — ACK	Host A

Key Nmap insight: Nmap has full control over the TCP flags in each packet it sends. It can set any combination of flags to 0 or 1 — much like flipping a light switch on or off. This ability to craft custom TCP packets is the foundation of Nmap's advanced scanning power.

2.2 TCP Connection Termination

There are two ways to close a TCP connection:

- Normal (graceful) close using FIN flag:
  - Host A sends a FIN packet → Host B enters CLOSE_WAIT state, releasing resources → Host B sends its own FIN → Host A sends ACK → connection fully closed
- Abrupt close using RST (Reset) flag:
  - Either host sends a RST packet — immediately terminating the connection without the graceful close process
  - Nmap uses RST extensively: it sends a SYN, receives SYN-ACK, then sends a RST to abort the connection — avoiding a full handshake (the Stealth/SYN scan technique)

2.3 Port States Detected by Nmap

Port State	Meaning
Open	An application is actively listening on this port and accepting connections. This is what you are looking for when mapping services.
Closed	No application is listening on this port. The target sends back a RST (reset) packet in response to probes. Port is reachable but unused.
Filtered	A firewall, IDS, or router rule is blocking probes to this port. Nmap cannot determine whether the port is open or closed — probes receive no response (or ICMP unreachable messages).
Unfiltered	The port is accessible (not filtered), but Nmap cannot determine whether it is open or closed from the probe response. Seen in ACK scans.
Open Filtered	Nmap cannot determine if the port is open or filtered — typical of UDP, FIN, NULL, and Xmas scans when there is no response.

2.4 Common Ports and Their Services

Port	Protocol
21	TCP
22	TCP
25	TCP
53	UDP / TCP
67	UDP
80	TCP
137	UDP
161	UDP
443	TCP
445	TCP
1900	UDP

SECTION 3: Host Discovery — Finding Live Hosts

3.1 Why Host Discovery?

Host discovery is always the first step in a network scan or penetration test. Before you scan for open ports and services, you need to know which hosts are actually alive and reachable. Scanning dead or non-existent hosts wastes time and resources.

3.2 Ping Sweep (-sP)

The Ping Sweep sends ICMP echo requests and TCP packets with the ACK flag set to each host in a range. Hosts that respond are considered alive.

```
nmap -sP 192.168.1.0/24
```

- Sends: ICMP echo request (ping) + TCP ACK packet
- A host is considered alive if it sends back: an ICMP echo reply OR a TCP RST packet
- Does NOT scan for open ports — only checks which hosts are up
- Limitation: many firewalls block ICMP and TCP ACK packets — see alternatives below

3.3 Customising Host Discovery Probes

Because firewalls commonly block standard ICMP pings and TCP ACK packets, Nmap provides alternative discovery methods:

TCP SYN Ping (-PS):

- Sends a TCP packet with the SYN flag set instead of ACK — bypasses many stateful firewalls that block unsolicited ACK packets

```
nmap -sP -PS 192.168.1.0/24
```

- Can also specify the source port to make the probe look more legitimate:

```
nmap -sP -PS22 192.168.1.0/24
```

TCP ACK Ping (-PA):

- Sends a TCP packet with the ACK flag set — useful against different firewall configurations

```
nmap -sP -PA 192.168.1.0/24
```

ARP Ping (-PR):

- Sends ARP (Address Resolution Protocol) requests — the most reliable method for discovering hosts on a LOCAL network
- Every host on the local network MUST respond to ARP requests — they cannot be blocked by host-based firewalls
- ARP ping only works within the same network segment (local/internal network) — it cannot cross routers

```
sudo nmap -sP -PR 192.168.1.0/24
```

- After an ARP scan, view the ARP table to see discovered MAC addresses:

```
arp -a
```

3.4 Scanning from a Target List File (-iL)

Instead of specifying hosts on the command line, you can create a text file containing one host or range per line and pass it to Nmap:

```
nano list_of_targets.txt
nmap -iL list_of_targets.txt
```

- Useful when scanning a large number of specific hosts or subnets
- The file can contain IP addresses, hostnames, CIDR ranges, or any combination

3.5 RTT Timeout Tuning for Distant Targets

RTT (Round Trip Time) is the time it takes for a probe to travel to the target and the response to return. When scanning targets with high latency (e.g. international targets), you must increase Nmap's timeout values or probes will be incorrectly marked as filtered/lost.

```
nmap --max-rtt-timeout 250ms --initial-rtt-timeout 150ms <target>
```

- --max-rtt-timeout: maximum time to wait for a response (e.g. 250ms for a target with ~75ms ping)
- --initial-rtt-timeout: starting value for the adaptive RTT algorithm (e.g. 150ms — roughly 2x the measured ping)
- Rule of thumb: set max RTT to 3–4x the measured ping time to be conservative

SECTION 4: Port Scanning Techniques

4.1 TCP Connect Scan (-sT) — Full Handshake

The TCP Connect scan completes the full three-way handshake (SYN → SYN-ACK → ACK) with each target port. It is the default scan type when Nmap is run WITHOUT root/Sudo privileges.

```
nmap -sT <target>
```

- Pros: works without root privileges; reliable and accurate
- Cons: very noisy — the full handshake is logged by the target's firewall and IDS; slower than SYN scan
- When to use: when root privileges are unavailable; when testing a target, you are authorized to connect to fully

4.2 TCP SYN Scan (-sS) — Stealth Scan ★ Default

The TCP SYN scan (also called the Stealth Scan or Half-Open scan) is Nmap's default scan type when run with root privileges. It sends a SYN packet, receives the SYN-ACK, and then immediately sends an RST to abort the connection — never completing the handshake.

```
sudo nmap -sS <target>
```

- Port is OPEN if: SYN-ACK is received (target is listening)
- Port is CLOSED if: RST is received (target rejected the connection)
- Port is FILTERED if: no response (firewall dropped the packet)
- Pros: fast; stealthy — many older firewalls and IDS do not log incomplete connections; gives clear open/closed/filtered view
- Cons: requires root/Sudo privileges

The SYN scan is the recommended starting point for nearly all Nmap scans. Start here before trying more specialized techniques.

4.3 TCP ACK Scan (-sA) — Firewall Detection

The ACK scan sends TCP packets with only the ACK flag set. It is NOT used to find open ports — it is used to map firewall rules and determine whether ports are filtered or unfiltered.

```
sudo nmap -sA <target>
```

- Unfiltered: the target responds with RST — the port is reachable (no firewall blocking)
- Filtered: no response — a firewall is dropping the ACK packets
- Key use: determine whether a host is behind a firewall and which ports the firewall is blocking
- Example: scanning your own internal host → all ports unfiltered; scanning nmap.org → all 993 ports filtered (behind a firewall), with only 7 ports accessible through firewall rules

4.4 FIN Scan (-sF), NULL Scan (-sN), and Xmas Scan (-sX)

These three scans use non-standard TCP flag combinations. When a port is CLOSED, it responds with RST. When a port is OPEN (or filtered), there is no response. This behavior allows these scans to infer port state without the SYN flag — bypassing firewalls configured to block SYN packets.

Scan Type	Flags Set
FIN Scan (-sF)	FIN = 1 only
NULL Scan (-sN)	All flags = 0
Xmas Tree Scan (-sX)	URG + PSH + FIN = 1

- Limitation: these scans may return open filtered for many ports (cannot distinguish open from filtered in the absence of a response)
- Best used when SYN scans are being blocked by stateless firewalls that only examine the SYN flag

Strategy: start with -sS. If blocked, try -sF, then -sX. Try custom flag combinations with --scan flags for stubborn systems.

4.5 Custom Scan Flags (--scan flags)

Nmap allows you to craft completely custom TCP packets by specifying exactly which flags to set — giving you total control over the TCP header.

```
sudo nmap --scanflags SYNURG <target>
```

- You specify flag names directly: SYN, ACK, FIN, RST, URG, PSH, NULL
- Not all combinations will produce useful results — experimentation is required
- Requires root privileges
- Useful for evading specific IDS signatures that flag known Nmap scan patterns

4.6 UDP Scan (-sU)

Many important services use UDP rather than TCP — including DNS (port 53), DHCP (port 67), SNMP (port 161), and UPnP (port 1900). UDP scanning is fundamentally different from TCP scanning because UDP is connectionless.

```
sudo nmap -sU --max-retries 1 <target>
```

- How UDP scanning works: Nmap sends a UDP packet to each port
  - Open port: either no response (most common) OR an application-specific UDP response
  - Closed port: ICMP port unreachable message returned
  - Filtered: no response and no ICMP message (firewall blocking)
- Why it is slow: open and filtered ports typically don't respond — Nmap must wait and retry before concluding a port is filtered
- Speed-up: use --max-retries 1 to try each port only once (less accurate but much faster)
- Example result: port 53 (DNS) and port 137 (NetBIOS) open on a typical home gateway

SECTION 5: Service Version Detection and OS Detection

5.1 Service Version Detection (-sV)

Knowing that port 80 is open is useful. Knowing that port 80 is running Apache 2.4.6 is far more valuable — it allows you to check for known vulnerabilities (CVEs) for that specific version.

```
nmap -sV <target>
```

- Nmap sends application-specific probes to open ports and analyses the responses
- Returns: service name, protocol, version number, and additional details
- Example output: Port 22 open → OpenSSH 7.4; Port 80 open → Apache httpd 2.4.6
- Once you have a version number, search the CVE database (cve.mitre.org or nvd.nist.gov) for known exploits and patches

5.2 Version Intensity (--version-intensity)

You can control how aggressively Nmap probes for version information — higher intensity = more probes = more accurate but slower.

```
nmap -sV --version-intensity 7 <target>
```

Intensity Level	Behaviour
0 — Lightest	Only the most likely probes sent. Very fast but may miss some services.
2 — Light	Faster scan with reduced accuracy. Good for quick reconnaissance.
7 (Default)	Nmap's balanced default — good accuracy without excessive delay.
9 — Maximum	Tries every possible probe. Most accurate but very slow.

5.3 OS Detection (-O)

OS detection uses a combination of TCP/IP stack probing techniques to fingerprint the operating system running on the target device. This is invaluable for identifying which systems need which patches.

```
sudo nmap -O <target>
```

- How it works: Nmap sends up to five different probe types and analyses responses:
  - Sequence Generation probe — 6 TCP SYN packets to measure sequence number behavior
  - ICMP echo request probes
  - TCP probes with various flag combinations (FIN, NULL, Xmas, etc.)
  - UDP probes
- The response patterns are compared against Nmap's fingerprint database (nmap-os-db) containing thousands of OS signatures
- Results include: device type (phone, router, computer), OS family (Linux, Windows, Android), specific OS version/variant
- Example: detecting a device as 'Phone — Android (CyanogenMod)' from TCP/IP stack behavior alone
- Requires root/Sudo privileges
- Combined with -sV, provides a complete picture: OS + services + versions → basis for targeted vulnerability research

SECTION 6: Firewall and IDS Evasion Techniques

6.1 Why Evasion?

Modern networks are protected by firewalls and Intrusion Detection Systems (IDS). These tools often recognize Nmap's standard scanning patterns and block them. Nmap provides a range of techniques to disguise scans and evade detection — all useful for penetration testing with proper authorization.

6.2 Decoy Scan (-D)

The Decoy scan makes it appear to the target as if the scan is coming from multiple IP addresses simultaneously — making it much harder to identify the real attacker.

```
sudo nmap -D <decoy_IP> <target>
```

- The target sees packets from multiple source IPs — your real IP is hidden amongst the decoys
- Use a real IP on your network as a decoy (e.g. your gateway): nmap -D 192.168.1.1 <target>
- Or generate random spoofed IPs automatically:
  ```
  sudo nmap -D RND:5 <target>
  ```
  - RND:5 generates 5 random decoy IP addresses
- Requires root privileges (crafting raw packets with spoofed source IPs)

6.3 Source Port Spoofing (--source-port)

Some misconfigured firewalls trust traffic originating from well-known ports (e.g. DNS port 53, HTTP port 80). By setting the source port of your Nmap scan to one of these trusted ports, the firewall may permit packets it would otherwise block.

```
sudo nmap --source-port 53 <target>
```

- Using source port 53 (DNS) makes packets appear to originate from a DNS server — often allowed through firewalls
- Using source port 80 (HTTP) makes packets appear to be web server responses — often permitted

6.4 Packet Fragmentation (-f and --mtu)

Fragmentation divides Nmap's probe packets into very small pieces. Many firewalls and IDS cannot reassemble and inspect tiny fragmented packets — they simply pass them through.

- Fragment option (-f): splits packets into 8-byte fragments
  ```
  sudo nmap -f <target>
  ```
- MTU option (--mtu): allows custom packet size (must be a multiple of 8)
  ```
  sudo nmap --mtu 16 <target>
  ```
- Useful for: evading older firewalls and IDS that cannot handle fragmented packets
- Note: most modern firewalls reassemble fragments before inspection — effectiveness varies

6.5 Data Length (--data-length)

Nmap's probes have a characteristic size that is known to many firewall vendors and IDS signature databases. Adding extra random data to each packet changes the packet size, making Nmap's probes harder to fingerprint.

```
sudo nmap --data-length 10 <target>
```

- Adds a specified number of random bytes to each probe packet
- The example above adds 10 extra bytes — the probe no longer matches Nmap's known probe signature
- Does not affect the functionality of the scan — just disguises the packet size

6.6 MAC Address Spoofing (--spoof-mac)

At Layer 2 (Data Link layer), devices are identified by their MAC address. Spoofing your MAC address prevents the target from linking the scan to your real network adapter.

```
sudo nmap --spoof-mac 0 <target>
```

- 0 as the parameter generates a completely random MAC address
- You can also specify a known vendor (Apple, Dell, Microsoft) to generate a realistic-looking MAC:
  ```
  sudo nmap --spoof-mac Apple <target>
  ```
- No registered vendor behind a random MAC may itself be suspicious — using a known vendor MAC looks more legitimate
- Only effective at Layer 2 (local network) — does not affect IP-level identification

6.7 Timing Templates (-T)

Nmap provides six timing templates that control scan speed. Slower scans are less likely to trigger IDS alerts; faster scans are noisier but quicker.

Template	Name
-T0	Paranoid
-T1	Sneaky
-T2	Polite
-T3	Normal (Default)
-T4	Aggressive
-T5	Insane

6.8 Evasion Techniques — Summary Table

Technique	Flag
Decoy Scan	-D <IP> or -D RND:N
Source Port Spoof	--source-port <port>
Fragmentation	-f
Custom MTU	--mtu <size>
Data Length	--data-length <bytes>
MAC Spoofing	--spoof-mac <vendor/0>
Slow Timing	-T0 to -T2

SECTION 7: Output Options — Saving and Analyzing Results

7.1 Why Save Output?

When scanning dozens or hundreds of hosts, reviewing results in the terminal is impractical. Nmap supports three output formats that allow results to be saved, shared, searched, and processed by other tools.

7.2 Output Format Options

Format	Flag
Normal Text	-oN <filename>
Grepable	-oG <filename>
XML	-oX <filename>
All Formats	-oA <basename>

7.3 Using Grep with Greppable Output

The greppable (-oG) format is designed to be filtered with the Linux grep command. For example, to find which hosts in a scan have port 21 (FTP) open:

```
grep '21/open' live.txt
```

- This instantly shows only the hosts with port 21 open — invaluable when scanning large networks
- You can chain grep commands for complex filtering (e.g. hosts with both port 22 and port 80 open)

7.4 Verbosity and Debugging Options

Option	Flag
Verbose Mode	-v (or -vv)
Reason	--reason
Debug Mode	-d (or -d1 to -d9)
Packet Trace	--packet-trace

Useful learning tip: Run the same scan with -v and without it, then with --reason, to understand what Nmap is detecting and how it reaches its conclusions.

SECTION 8: Nmap Scripting Engine (NSE)

8.1 What is the NSE?

The Nmap Scripting Engine (NSE) is one of Nmap's most powerful features. It allows users to run pre-written scripts (or write their own) to automate complex network tasks — from vulnerability detection to service enumeration to brute-force attacks.

- Scripts are written in Lua — a lightweight scripting language
- This course covers using existing scripts — not writing new ones
- Scripts are stored locally in the Nmap scripts directory (typically /usr/share/nmap/scripts/)
- Usage: nmap --script <script_name> <target>
- WARNING: some NSE scripts use aggressive techniques that can harm or crash target systems. Always obtain permission before running scripts against any target.

8.2 NSE Script Categories

Category	Description
auth	Authentication testing — check for default or weak credentials
broadcast	Discover hosts using broadcast messages (no specific target needed)
brute	Brute-force credential attacks against services
default	Run when -sC flag is used — the most commonly useful safe scripts
discovery	Gather information about network services and hosts
dos	Denial-of-Service tests — use with extreme caution, authorised environments only
exploit	Attempt to exploit known vulnerabilities — use only with explicit permission
external	Scripts that query external databases and APIs
fuzzer	Send unexpected or malformed input to services to find bugs
intrusive	Aggressive scripts likely to be detected or cause harm — authorised use only
malware	Detect malware infection on target systems
safe	Non-intrusive scripts unlikely to cause harm or crash targets
version	Extend service version detection capabilities
vuln	Check for known vulnerabilities on detected services

8.3 SNMP Scripts

SNMP (Simple Network Management Protocol) is used to monitor and manage network devices. SNMP versions 1 and 2 have no real security — they use a 'community string' (essentially a plain-text password, usually 'public' or 'private') instead of proper authentication. Misconfigured SNMP can expose extensive system information.

- SNMP runs on UDP port 161
- SNMP versions 1 and 2: no encryption, weak community string authentication — treat as insecure
- Commonly found misconfigured on: printers, IP cameras, routers, switches, and older servers

Key SNMP NSE Scripts:

Script	Command Example
snmp-win32-software	sudo nmap -sU -p161 --script snmp-win32-software <target>
snmp-sysdescr	sudo nmap -sU -p161 --script snmp-sysdescr <target>
snmp-interfaces	sudo nmap -sU -p161 --script snmp-interfaces <target>
snmp-win32-users	sudo nmap -sU -p161 --script snmp-win32-users <target>

8.4 NetBIOS Scripts

NetBIOS (Network Basic Input/Output System) is an older protocol still used in local area networks for device naming and communication. It uses UDP port 137.

```
sudo nmap -sU -p137 --script nbstat <target>
```

- Returns the NetBIOS name of the target device
- Useful for identifying device hostnames on Windows networks
- Works reliably against gateways and routers that support NetBIOS

8.5 SMB Scripts

SMB (Server Message Block) is a protocol used by Windows systems (and others) to share files, printers, and other resources over a network. SMB operates on TCP port 445.

- SMB versions: SMBv1 (legacy, dangerous — responsible for WannaCry), SMBv2, SMBv3 (most secure)

Key SMB NSE Scripts:

Script	Command Example
smb-os-discovery	nmap --script smb-os-discovery <target>
smb-enum-shares	nmap --script smb-enum-shares <target>

- Security note: writable SMB shares are a serious risk — an attacker could drop malware into a shared folder that is then executed by other users

8.6 HTTP Enumeration Script

Many network devices (routers, printers, cameras, NAS devices) run internal web servers for administration. The HTTP enumeration script attempts to discover directories and pages on these web servers.

```
nmap --script http-enum -sV <target>
```

- Discovers: /admin, /login, /help, /status directories and other potentially interesting paths
- Combined with -sV, also reveals the web server software and version (e.g. lighttpd 1.4.35)
- Cross-reference any discovered version against CVE databases for known vulnerabilities

8.7 DNS Cache Snooping Script

DNS cache snooping is an intelligence-gathering technique that queries a DNS server to determine which domain names are currently cached — revealing which websites the DNS server's clients have recently visited.

```
sudo nmap -sU -p53 --script dns-cache-snoop <target>
```

- Nmap queries the DNS server for the 50 most popular domains
- Two methods to detect a cached entry:
  - Time difference: a cached DNS response arrives faster than an uncached one (no external lookup needed)
  - Recursion flag: a DNS request with recursion flag = 0 is only answered if the domain is already cached
- Reveals: which popular websites/services users on this network have visited recently
- Target: your gateway (which usually acts as the local DNS server)

8.8 DHCP Discovery Script

DHCP (Dynamic Host Configuration Protocol) automatically assigns IP addresses and network configuration to devices that join a network. The DHCP Discovery script allows you to query the DHCP server for information about itself and the network.

```
sudo nmap -sU -p67 --script dhcp-discover <target>
```

- Sends a DHCP INFORM message (requests information without requesting a new IP address)
- Returns: DHCP server IP, DNS server IP, gateway/router IP, subnet mask, lease time
- Use case: identify whether a rogue/unauthorised DHCP server exists on the network alongside the legitimate one
- A rogue DHCP server can redirect all network traffic through an attacker's machine (man-in-the-middle attack)

8.9 UPnP Info Script

UPnP (Universal Plug and Play) allows devices on a network to automatically discover and communicate with each other — routers, smart TVs, printers, cameras, and computers. UPnP runs on UDP port 1900.

```
sudo nmap -sU -p1900 --script upnp-info <target>
```

- Sends a UPnP discovery message to the target
- Returns: device name, manufacturer, model number, firmware version, MAC address, location URL of the XML service description file
- The location URL can be opened in a browser to view the full XML description of all UPnP services the device offers
- Security risk: UPnP is often misconfigured and exposes internal device details and port mapping capabilities — attackers on the local network can potentially redirect ports through your router
- Example output: device = Netgear R500 access point, firmware = 1.0.0.20, location = http://192.168.1.1:80/upnp.xml

SECTION 9: Nmap Command Quick Reference

9.1 Essential Command Cheat Sheet

Purpose	Command
Show Nmap help menu	nmap
Ping sweep (host discovery)	nmap -sP 192.168.1.0/24
TCP SYN ping discovery	sudo nmap -sP -PS 192.168.1.0/24
ARP ping (local network)	sudo nmap -sP -PR 192.168.1.0/24
Scan from file	nmap -iL targets.txt
Full TCP Connect scan	nmap -sT <target>
SYN Stealth scan (default)	sudo nmap -sS <target>
ACK scan (firewall detection)	sudo nmap -sA <target>
FIN scan	sudo nmap -sF <target>
NULL scan	sudo nmap -sN <target>
Xmas Tree scan	sudo nmap -sX <target>
Custom TCP flags	sudo nmap --scanflags SYNFIN <target>
UDP scan (fast)	sudo nmap -sU --max-retries 1 <target>
Service version detection	nmap -sV <target>
OS detection	sudo nmap -O <target>
Service + OS detection	sudo nmap -sV -O <target>
Aggressive scan (sV+sC+O+traceroute)	sudo nmap -A <target>
Decoy scan	sudo nmap -D RND:5 <target>
Source port spoof	sudo nmap --source-port 53 <target>
Packet fragmentation	sudo nmap -f <target>
Custom MTU fragmentation	sudo nmap --mtu 16 <target>
Add data length	sudo nmap --data-length 10 <target>
MAC address spoof (random)	sudo nmap --spoof-mac 0 <target>
Slow timing (T1)	sudo nmap -T1 <target>
Save to text file	nmap -oN output.txt <target>
Save to grepable file	nmap -oG output.txt <target>
Save to XML file	nmap -oX output.xml <target>
Verbose mode	nmap -v <target>
Show reason for port state	nmap --reason <target>
Debug mode	nmap -d <target>
SNMP software list	sudo nmap -sU -p161 --script snmp-win32-software <target>
SNMP system description	sudo nmap -sU -p161 --script snmp-sysdescr <target>
SNMP interfaces	sudo nmap -sU -p161 --script snmp-interfaces <target>

Purpose	Command
SNMP Windows users	sudo nmap -sU -p161 --script snmp-win32-users <target>
NetBIOS name	sudo nmap -sU -p137 --script nbstat <target>
SMB OS discovery	nmap --script smb-os-discovery <target>
SMB share enumeration	nmap --script smb-enum-shares <target>
HTTP directory enumeration	nmap --script http-enum -sV <target>
DNS cache snooping	sudo nmap -sU -p53 --script dns-cache-snoop <target>
DHCP discovery	sudo nmap -sU -p67 --script dhcp-discover <target>
UPnP info	sudo nmap -sU -p1900 --script upnp-info <target>

— End of Course Notes —