

SCANNING

Module 4: Scanning & Enumeration (Full Expanded Version)

Objective:

To equip learners with the skills to **actively interact with a target network**, identify **open ports**, **running services**, **OS details**, and extract **critical technical information** using specialized tools.

◆ **Lesson 4.1 – What is Scanning?**

Definition:

Scanning is the process of **actively probing a system** or network to identify:

- Live hosts (computers, routers, servers)
- Open ports (entry points)
- Services running (e.g., FTP, HTTP, SSH)
- Operating systems (Windows, Linux, etc.)
- Possible vulnerabilities

Purpose:

Think of scanning as a **digital knock** on each door of a building to check:

- Which doors are open?
 - What's behind those doors?
 - Are there any weak doors (vulnerabilities)?
-

◆ **Lesson 4.2 – Types of Scanning**

1. Ping Sweep (Host Discovery)

- **Purpose:** Identify which systems are online in a network.
- **How:** Sends ICMP echo requests (ping) to all IPs in a subnet.
- **Tool:** `nmap -sn 192.168.1.0/24`

2. Port Scanning

- **Purpose:** Find which ports are open on a system.
- Each port represents a **specific service** (e.g., 80 = HTTP).
- **Tools:** Nmap, Masscan

3. Service Version Detection

- **Purpose:** Identify software and version behind an open port.
- Why? Old versions = exploitable.
- `nmap -sV target.com`

4. OS Fingerprinting

- **Purpose:** Identify target's OS by analyzing packet responses.
- `nmap -O target.com`

5. Stealth Scan (SYN Scan)

- Sends partial connection requests to avoid detection by firewalls.
- `nmap -sS target.com`

Scan Type	Description	Detectable?
TCP Connect	Full connection to target port	Yes
SYN Scan	Half-open connection (stealth)	No/Low
UDP Scan	For non-TCP services	Sometimes
Xmas/FIN Scan	Obscure techniques for evasion	Rarely

◆ Lesson 4.3 – Mastering Nmap

Nmap (Network Mapper) is the hacker's go-to tool for scanning and reconnaissance.

🔧 **Common Nmap Commands:**

Command	Purpose
<code>nmap -sn 192.168.1.0/24</code>	Ping sweep
<code>nmap -sS target.com</code>	SYN stealth scan
<code>nmap -sV target.com</code>	Service version detection
<code>nmap -O target.com</code>	OS fingerprinting
<code>nmap -A target.com</code>	Aggressive scan (OS + services + traceroute)
<code>nmap -p- target.com</code>	Scan all 65535 ports

Pro Tip:

Use `-T4` for faster scans and `--script vuln` to detect vulnerabilities:

```
bash
CopyEdit
nmap -A -T4 --script vuln target.com
```

3. Essential Scanning Tools

Nmap (Network Mapper)

Most powerful & flexible scanner.

Common Nmap commands:

```
bash
CopyEdit
nmap -sS 192.168.1.1      # Stealth SYN Scan
nmap -sV 192.168.1.1     # Service version detection
nmap -O 192.168.1.1      # OS detection
nmap -T4 -A 192.168.1.1  # Aggressive scan
```

Masscan

Faster than Nmap. Useful for scanning large IP ranges.

```
bash
CopyEdit
```

```
masscan -p1-65535 192.168.1.0/24 --rate=1000
```

Unicornscan

Advanced alternative to Nmap for stealthy scanning.

Netcat

Useful for manual testing and banner grabbing.

```
bash
CopyEdit
nc -nv 192.168.1.1 80
```

Definition:

Enumeration is the phase where an ethical hacker **extracts structured and detailed information** from the discovered systems **after scanning** confirms the system is live.

If scanning is “checking doors”, enumeration is peeking inside to see what’s stored inside.

Information Gathered:

- Active user accounts
- Shared folders
- Network shares
- System uptime
- Software/OS version
- Printer and drive details

Lesson 4.5 – Enumeration Protocols & Targets

Protocol	Enumeration Details	Tool
SMB/NetBIOS	Users, groups, shares	Enum4linux

Protocol	Enumeration Details	Tool
SNMP	Network devices, config	SNMPwalk
LDAP	Directory services	ldapsearch
DNS	Zone transfers, subdomains	dig, host
FTP	Banner grabbing, anonymous login	telnet, netcat
SMTP	Valid email accounts	Telnet, Nmap scripts
HTTP	Directory structure, headers	Dirb, Gobuster, Nikto

◆ Lesson 4.6 – Tools for Scanning & Enumeration

🔧 Scanning Tools:

- **Nmap** – Full-featured scanner
- **Masscan** – Super fast port scanner
- **Netdiscover** – LAN discovery tool
- **Angry IP Scanner** – GUI-based ping sweep

🔍 Enumeration Tools:

Tool	Description
Enum4linux	NetBIOS & SMB enumeration
SNMPwalk	SNMP data extraction
Dirb , Gobuster	Discover hidden web directories
Nikto	Web vulnerability scanner
Hydra	Brute-force login services
WhatWeb , Wappalyzer	Identify web technologies

◆ Lesson 4.7 – Real Hacking Scenario

🕵️ Example Flow:

1. You scan `192.168.1.100` using `nmap -sS -sV -O`.
2. Port 21 (FTP), 80 (HTTP), 445 (SMB) are open.
3. Use `enum4linux` → find users: `admin` , `guest`

4. Try SMB login with default creds → access `\\192.168.1.100\public`
5. Inside: find a text file with database credentials 🔑

You now have a valid entry point!

Practical Lab Suggestion:

Set up a vulnerable VM (like **Metasploitable 2** or **TryHackMe Room**) and practice:

- Scanning with Nmap
- Enumerating SMB with Enum4linux
- Using `dirb` on exposed web server

Summary Chart:

Phase	Description	Example Tool
Host Discovery	Identify live systems	Nmap, Netdiscover
Port Scanning	Find open ports	Nmap, Masscan
Service Detection	Identify services & versions	Nmap <code>-sV</code>
OS Fingerprinting	Detect OS type	Nmap <code>-O</code>
Enumeration	Extract system data via services	Enum4linux, SNMPwalk, Dirb

Port Scanning and Recon with nmap, Part 02: The nmap scripts (nse)

First introduced by Fyodor in 1998 in Phrack magazine, nmap has been a staple of every hacker/pentester's toolbox for over 20 years. The nmap tool is mature, well-documented, and robust, but the NSE (Nmap Scripting Engine) takes nmap to a whole other level!

NSE transforms the functionality of nmap from a classic, port-scanning tool to a tool capable of vulnerability scanning, network discovery, fuzzing, password cracker and even exploitation. Presently, there are 603 scripts built into Kali but there are new scripts and capabilities being developed almost daily by an active and dedicated open-source community. These scripts are partitioned into a few categories, including;

1. auth
2. broadcast

3. brute
4. default
5. discovery
6. dos (denial of service)
7. exploit
8. external
9. fuzzer
10. intrusive
11. malware
12. safe
13. version
14. vuln (vulnerability)

NSE is a fully developed scripting language with scripts utilizing the Lua scripting language.

Basic Syntax

The syntax for using nmap scripts is similar to that of the basic nmap command with the exception of the keyword “-script=”. To invoke an NSE, you can use the keyword -script= followed by the name of the script or the category of the script and finally followed by the IP address of the target system such as;

```
kali > nmap -script vuln 192.168.1.101
```

This command will run a series of scripts in the “vuln” category and only output data if a vulnerability is found.

Let’s take a closer look at nmap scripts (NSE) in our Kali 2020 system.

Step #1: Fire up Kali and Open a terminal

Let’s begin by firing up our Kali Linux 2020 and opening a terminal as seen below.



Step #2: Search for NSE Scripts

There are a number of NSE scripts built into our Kali 2020. To find them, we can use the Linux command `locate`, followed by the wildcard `*` and then the extension of every nmap scripts `".nse"`. This should locate all files with that extension which should all include all nmap scripts.

kali > locate *.nse


```
kali@kali:~$ locate *.nse
/usr/share/exploitdb/exploits/hardware/webapps/31527.nse
/usr/share/exploitdb/exploits/multiple/remote/33310.nse
/usr/share/legion/scripts/nmap/shodan-api.nse
/usr/share/legion/scripts/nmap/shodan-hq.nse
/usr/share/legion/scripts/nmap/vulners.nse
/usr/share/nmap/scripts/acarsd-info.nse
/usr/share/nmap/scripts/address-info.nse
/usr/share/nmap/scripts/afp-brute.nse
/usr/share/nmap/scripts/afp-ls.nse
/usr/share/nmap/scripts/afp-path-vuln.nse
/usr/share/nmap/scripts/afp-serverinfo.nse
/usr/share/nmap/scripts/afp-showmount.nse
/usr/share/nmap/scripts/ajp-auth.nse
/usr/share/nmap/scripts/ajp-brute.nse
/usr/share/nmap/scripts/ajp-headers.nse
/usr/share/nmap/scripts/ajp-methods.nse
/usr/share/nmap/scripts/ajp-request.nse
/usr/share/nmap/scripts/allseeingeeye-info.nse
/usr/share/nmap/scripts/amqp-info.nse
/usr/share/nmap/scripts/asn-query.nse
/usr/share/nmap/scripts/auth-owners.nse
/usr/share/nmap/scripts/auth-spoof.nse
/usr/share/nmap/scripts/backorifice-brute.nse
/usr/share/nmap/scripts/backorifice-info.nse
/usr/share/nmap/scripts/bacnet-info.nse
/usr/share/nmap/scripts/banner.nse
/usr/share/nmap/scripts/bitcoin-getaddr.nse
/usr/share/nmap/scripts/bitcoin-info.nse
/usr/share/nmap/scripts/bitcoinrpc-info.nse
/usr/share/nmap/scripts/bittorrent-discovery.nse
/usr/share/nmap/scripts/bjnp-discover.nse
/usr/share/nmap/scripts/broadcast-ataoe-discover.nse
/usr/share/nmap/scripts/broadcast-avahi-dos.nse
/usr/share/nmap/scripts/broadcast-bjnp-discover.nse
```

As you can see there are quite a few nmap scripts. To save the list to a file, simply enter;

```
kali > locate *.nse >nmapscripts
```

```
kali@kali:~$ locate *.nse > nmapscripts
kali@kali:~$ cat -n nmapscripts
 1 /usr/share/exploitdb/exploits/hardware/webapps/31527.nse
 2 /usr/share/exploitdb/exploits/multiple/remote/33310.nse
 3 /usr/share/legion/scripts/nmap/shodan-api.nse
 4 /usr/share/legion/scripts/nmap/shodan-hq.nse
 5 /usr/share/legion/scripts/nmap/vulners.nse
 6 /usr/share/nmap/scripts/acarsd-info.nse
 7 /usr/share/nmap/scripts/address-info.nse
 8 /usr/share/nmap/scripts/afp-brute.nse
 9 /usr/share/nmap/scripts/afp-ls.nse
10 /usr/share/nmap/scripts/afp-path-vuln.nse
11 /usr/share/nmap/scripts/afp-serverinfo.nse
12 /usr/share/nmap/scripts/afp-showmount.nse
13 /usr/share/nmap/scripts/ajp-auth.nse
14 /usr/share/nmap/scripts/ajp-brute.nse
15 /usr/share/nmap/scripts/ajp-headers.nse
16 /usr/share/nmap/scripts/ajp-methods.nse
17 /usr/share/nmap/scripts/ajp-request.nse
18 /usr/share/nmap/scripts/allseeingeye-info.nse
19 /usr/share/nmap/scripts/amqp-info.nse
20 /usr/share/nmap/scripts/asn-query.nse
21 /usr/share/nmap/scripts/auth-owners.nse
22 /usr/share/nmap/scripts/auth-spoof.nse
23 /usr/share/nmap/scripts/backorifice-brute.nse
```

Then, to see the total number of scripts, simply use the cat command followed by the -n option and the name of the file such as;

kali > cat -n nmapscripts

```
589 /usr/share/nmap/scripts/vnc-title.nse
590 /usr/share/nmap/scripts/voldemort-info.nse
591 /usr/share/nmap/scripts/vtam-enum.nse
592 /usr/share/nmap/scripts/vulners.nse
593 /usr/share/nmap/scripts/vuze-dht-info.nse
594 /usr/share/nmap/scripts/wdb-version.nse
595 /usr/share/nmap/scripts/weblogic-t3-info.nse
596 /usr/share/nmap/scripts/whois-domain.nse
597 /usr/share/nmap/scripts/whois-ip.nse
598 /usr/share/nmap/scripts/wsdd-discover.nse
599 /usr/share/nmap/scripts/x11-access.nse
600 /usr/share/nmap/scripts/xdmcp-discover.nse
601 /usr/share/nmap/scripts/xmlrpc-methods.nse
602 /usr/share/nmap/scripts/xmpp-brute.nse
603 /usr/share/nmap/scripts/xmpp-info.nse
kali@kali:~$
```


As you can see, there are 603 nmap scripts installed on Kali Linux 2020. For further information on these scripts, you can refer to the nmap scripts website at;

<https://nmap.org/nsedoc/>

As you can see below, each script has an explanation of its function on this page.

Scripts	
acarsd-info	Retrieves information from a listening acarsd daemon. Acarsd decodes ACARS (Aircraft Communication Addressing and Reporting System) data in real time. The information retrieved by this script includes the daemon version, API version, administrator e-mail address and listening frequency.
address-info	Shows extra information about IPv6 addresses, such as embedded MAC or IPv4 addresses when available.
afp-brute	Performs password guessing against Apple Filing Protocol (AFP).
afp-ls	Attempts to get useful information about files from AFP volumes. The output is intended to resemble the output of ls.
afp-path-vuln	Detects the Mac OS X AFP directory traversal vulnerability, CVE-2010-0533.
afp-serverinfo	Shows AFP server information. This information includes the server's hostname, IPv4 and IPv6 addresses, and hardware type (for example Macmini or MacBookPro).
afp-showmount	Shows AFP shares and ACLs.
ajp-auth	Retrieves the authentication scheme and realm of an AJP service (Apache JServ Protocol) that requires authentication.
ajp-brute	Performs brute force passwords auditing against the Apache JServ protocol. The Apache JServ Protocol is commonly used by web servers to communicate with back-end Java application server containers.
ajp-headers	Performs a HEAD or GET request against either the root directory or any optional directory of an Apache JServ Protocol server and returns the server response headers.
ajp-methods	Discovers which options are supported by the AJP (Apache JServ Protocol) server by sending an OPTIONS request and lists potentially risky methods.
ajp-request	Requests a URI over the Apache JServ Protocol and displays the result (or stores it in a file). Different AJP methods such as, GET, HEAD, TRACE, PUT or DELETE may be used.
allseeingeye-info	Detects the All-Seeing Eye service. Provided by some game servers for querying the server's status.
amqp-info	Gathers information (a list of all server properties) from an AMQP (advanced message queuing protocol) server.
asn-query	Maps IP addresses to autonomous system (AS) numbers.

Step #3: NSE Help

Although there are rudimentary explanations of each script on the nmap web site, if you need more information on a script, you can use the "--script-help=" switch followed by the name of the script or script category. For instance, if I wanted more information on the script category "vuln", I could simply enter;

kali > nmap -script-help=vuln

```

https://nmap.org/nsedoc/scripts/http-shellshock.html
Attempts to exploit the "shellshock" vulnerability (CVE-2014-6271 and CVE-2014-7169) in web applications.

To detect this vulnerability the script executes a command that prints a
random string and then attempts to find it inside the response body. Web apps that
don't print back information won't be detected with this method.

By default the script injects the payload in the HTTP headers User-Agent,
Cookie, Referer and also uses the payload as the header name.

Vulnerability originally discovered by Stephane Chazelas.

References:
* http://www.openwall.com/lists/oss-security/2014/09/24/10
* http://seclists.org/oss-sec/2014/q3/685
* https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2014-7169
* http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2014-6271

http-slowloris-check
Categories: vuln safe
https://nmap.org/nsedoc/scripts/http-slowloris-check.html
Tests a web server for vulnerability to the Slowloris DoS attack without
actually launching a DoS attack.

Slowloris was described at Defcon 17 by RSnake
(see http://ha.ckers.org/slowloris/).

This script opens two connections to the server, each without the final CRLF.
After 10 seconds, second connection sends additional header. Both connections
then wait for server timeout. If second connection gets a timeout 10 or more
seconds after the first one, we can conclude that sending additional header
prolonged its timeout and that the server is vulnerable to slowloris DoS
attack.

A "LIKELY VULNERABLE" result means a server is subject to timeout-extension
attack, but depending on the http server's architecture and resource limits, a
full denial-of-service is not always possible. Complete testing requires
triggering the actual DoS condition and measuring server responsiveness.

You can specify custom http User-agent field with <code>http.useragent</code>
script argument.

Idea from Qualys blogpost:
* https://community.qualys.com/blogs/securitylabs/2011/07/07/identifying-slow-http-attack-vulnerabilities-on-web-applications

```

Note above that nmap returns each script in the category with a detailed explanation.

Step #4: Test vuln scripts category against OWASP-BWA Linux Server

Let's try using an entire category of nmap scripts against the OWASP BWA server. With nmap scripts, you can run every script in the category (see the categories above) by simply using the `-script`

switch followed by the name of the category, such as "vuln".

The vuln category includes scripts to test for known vulnerabilities in a target. To run the entire vuln category against our target system, we can simply enter;

kali > nmap -script vuln 192.168.100.102

```

kali@kali:~$ nmap --script vuln 192.168.100.102
Starting Nmap 7.80 ( https://nmap.org ) at 2020-09-04 12:24 EDT
Nmap scan report for 192.168.100.102
Host is up (0.0019s latency).
Not shown: 991 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
|_clamav-exec: ERROR: Script execution failed (use -d to debug)
80/tcp    open  http
|_clamav-exec: ERROR: Script execution failed (use -d to debug)
http-cookie-flags:
  /mono/:
    ASP.NET_SessionId:
      httponly flag not set
http-cross-domain-policy:
  VULNERABLE:
    Cross-domain and Client Access policies.
    State: VULNERABLE
    A cross-domain policy file specifies the permissions that a web client such as Java, Adobe Flash, Adobe Reader,
    etc. use to access data across different domains. A client access policy file is similar to cross-domain policy
    but is used for M$ Silverlight applications. Overly permissive configurations enables Cross-site Request
    Forgery attacks, and may allow third parties to access sensitive data meant for the user.
    Check results:
    /crossdomain.xml:
      <?xml version="1.0"?>
      <!DOCTYPE cross-domain-policy SYSTEM "http://www.macromedia.com/xml/dtds/cross-domain-policy.dtd">
      <cross-domain-policy>
        <allow-access-from domain="*" />
      </cross-domain-policy>
    Extra information:
      Trusted domains:*

References:
  https://www.adobe.com/devnet-docs/acrobatetk/tools/AppSec/CrossDomain_PolicyFile_Specification.pdf
  http://acunetix.com/vulnerabilities/web/insecure-clientaccesspolicy-xml-file
  http://gursevkalra.blogspot.com/2013/08/bypassing-same-origin-policy-with-flash.html
  https://www.owasp.org/index.php/Test_RIA_cross_domain_policy_%28OTG-CONFIG-008%29
  http://sethsec.blogspot.com/2014/03/exploiting-misconfigured-crossdomainxml.html
  https://www.adobe.com/devnet/articles/crossdomain_policy_file_spec.html
http-csrf:
  Spidering limited to: maxdepth=3; maxpagecount=20; withinhost=192.168.100.102
  Found the following possible CSRF vulnerabilities:

    Path: http://192.168.100.102:80/railsgoat/
    Form id:
    Form action: /railsgoat/signup

```

We can be more specific and look for SQL Injection vulnerabilities by using the "http-sql-injection" script. First, let's take a look at its help screen.

kali > nmap --scripts-help=http-sql-injection

```

http-sql-injection
Categories: intrusive vuln
https://nmap.org/nsedoc/scripts/http-sql-injection.html
Spiders an HTTP server looking for URLs containing queries vulnerable to an SQL
injection attack. It also extracts forms from found websites and tries to identify
fields that are vulnerable.

The script spiders an HTTP server looking for URLs containing queries. It then
proceeds to combine found SQL commands with susceptible URLs in order to
obtain errors. The errors are analysed to see if the URL is vulnerable to
attack. This uses the most basic form of SQL injection but anything more
complicated is better suited to a standalone tool.

We may not have access to the target web server's true hostname, which can prevent access to
virtually hosted sites.

```

nmap provides a detailed explanation of this script after you hit enter. This help screen explains that this script spiders an HTTP server for URLs vulnerable to SQL injection attack.

Now let's try executing this script against our OWASP BWA server.

kali > nmap -script=http-sql-injection 192.168.1.101

```
kali@kali:~$ nmap --script=http-sql-injection 192.168.1.101
Starting Nmap 7.80 ( https://nmap.org ) at 2020-09-24 14:12 EDT
mass_dns: warning: Unable to determine any DNS servers. Reverse DNS is disabled. Try using --system-dns or specify valid servers with --dns-servers
Nmap scan report for 192.168.1.101
Host is up (0.0013s latency).
Not shown: 991 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    open  http
http-sql-injection:
  Possible sqli for queries:
    http://192.168.1.101:80/mutillidae/index.php?do=toggle-bubble-hints%27%20OR%20sqlspider%page=X2fowaspbwaX2fmutillidae-gitX2fhome.php
    http://192.168.1.101:80/mutillidae/index.php?do=toggle-security%27%20OR%20sqlspider%page=X2fowaspbwaX2fmutillidae-gitX2fhome.php
    http://192.168.1.101:80/mutillidae/includes/pop-up-help-context-generator.php?pagename=X2fowaspbwaX2fmutillidae-gitX2fhome.php%27%20OR%20sqlspider
    http://192.168.1.101:80/mutillidae/index.php?popUpNotificationCode=HPPH0%27%20OR%20sqlspider%page=home.php
    http://192.168.1.101:80/mutillidae/index.php?do=toggle-hints%27%20OR%20sqlspider%page=X2fowaspbwaX2fmutillidae-gitX2fhome.php
    http://192.168.1.101:80/mutillidae/index.php?do=toggle-enforce-ssl%27%20OR%20sqlspider%page=X2fowaspbwaX2fmutillidae-gitX2fhome.php
139/tcp    open  netbios-ssn
143/tcp    open  imap
443/tcp    open  https
445/tcp    open  microsoft-ds
5001/tcp    open  complex-link
8080/tcp    open  http-proxy
8081/tcp    open  blackice-icecap
Nmap done: 1 IP address (1 host up) scanned in 69.11 seconds
```

As you can see above, this script located 6 potential SQL injection vulnerabilities in that website.

Step 5: Test Other nmap Scripts within the http Category

Let's try some another script in the 'http' category. What if we were looking for login forms to brute force the username and password? We could find each of these using a script called "http-auth-finder". Let's run it against our OWASP BWA server and see whether it can locate the authentication forms.

kali > nmap -script="http-auth-finder" 192.168.1.101

```
kali@kali:~$ nmap --script="http-auth-finder" 192.168.1.101
Starting Nmap 7.80 ( https://nmap.org ) at 2020-09-23 15:26 EDT
mass_dns: warning: Unable to determine any DNS servers. Reverse DNS is disabled. Try using --system-dns or specify valid servers with --dns-servers
Nmap scan report for 192.168.1.101
Host is up (0.034s latency).
Not shown: 991 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    open  http
http-auth-finder:
  Spidering limited to: maxdepth=3; maxpagecount=20; withinhost=192.168.1.101
    url      method
    http://192.168.1.101:80/phpBB2/  FORM
    http://192.168.1.101:80/ghost/  FORM
139/tcp    open  netbios-ssn
143/tcp    open  imap
443/tcp    open  https
445/tcp    open  microsoft-ds
5001/tcp    open  complex-link
8080/tcp    open  http-proxy
8081/tcp    open  blackice-icecap
```

As you can see above, this script found two forms for authentication to this system. Now that we know where they are, we can use tools such as BurpSuite or THC-Hydra to brute force the authentication.

Step #6: Test nmap scripts against Windows 7 for EternalBlue vulnerability

Next, let's try using nmap scripts against an unpatched Windows 7 Professional system. These unpatched Windows 7 systems are often vulnerable to the EternalBlue exploit developed by the US NSA and released by the Shadowbrokers in 2017.

Let's first take a look at the help screen for the EternalBlue vulnerability scanner script in nmap scripts. We can find it by searching for it by its Microsoft designated vulnerability number "ms17-010" and entering;

```
kali > locate *.nse | grep ms17-010
```

Now that we have located the appropriate script for this task, let's view its help screen.

```
kali > nmap --script-help=smb-vuln-ms17-010.nse
```

```
kali@kali:~$ nmap --script-help=smb-vuln-ms17-010.nse
Starting Nmap 7.80 ( https://nmap.org ) at 2020-09-24 14:28 EDT

smb-vuln-ms17-010
Categories: vuln safe
https://nmap.org/nsedoc/scripts/smb-vuln-ms17-010.html
Attempts to detect if a Microsoft SMBv1 server is vulnerable to a remote code
execution vulnerability (ms17-010, a.k.a. EternalBlue).
The vulnerability is actively exploited by WannaCry and Petya ransomware and other malware.

The script connects to the $IPC tree, executes a transaction on FID 0 and
checks if the error "STATUS_INSUFF_SERVER_RESOURCES" is returned to
determine if the target is not patched against ms17-010. Additionally it checks
for known error codes returned by patched systems.

Tested on Windows XP, 2003, 7, 8, 8.1, 10, 2008, 2012 and 2016.

References:
* https://technet.microsoft.com/en-us/library/security/ms17-010.aspx
* https://blogs.technet.microsoft.com/msrc/2017/05/12/customer-guidance-for-wannacrypt-attacks/
* https://msdn.microsoft.com/en-us/library/ee441489.aspx
* https://github.com/rapid7/metasploit-framework/blob/master/modules/auxiliary/scanner/smb/smb_ms17_010.rb
* https://github.com/cldrn/nmap-nse-scripts/wiki/Notes-about-smb-vuln-ms17-010
```

As you can see above, this script "attempts to detect if a Microsoft SMBv1 server is vulnerable". It also points out that this vulnerability is actively exploited by WannaCry and Petya ransomware. For more information on EternalBlue, [click here](#).

Finally, let's run this script against our Windows 7 system and see whether its vulnerable to this malicious malware.

```
kali > nmap --script=smb-vuln-ms17-010.nse 192.168.1.103
```

```

kali@kali:~$ nmap --script=smb-vuln-ms17-010.nse 192.168.1.103
Starting Nmap 7.80 ( https://nmap.org ) at 2020-09-24 14:29 EDT
mass_dns: warning: Unable to determine any DNS servers. Reverse DNS is disabled. Try using --system-dns or specify valid serv
Nmap scan report for 192.168.1.103
Host is up (0.0046s latency).
Not shown: 990 closed ports
PORT      STATE SERVICE
135/tcp   open  msrpc
139/tcp   open  netbios-ssn
445/tcp   open  microsoft-ds
3389/tcp  open  ms-wbt-server
49152/tcp open  unknown
49153/tcp open  unknown
49154/tcp open  unknown
49155/tcp open  unknown
49156/tcp open  unknown
49157/tcp open  unknown

Host script results:
smb-vuln-ms17-010:
VULNERABLE:
Remote Code Execution vulnerability in Microsoft SMBv1 servers (ms17-010)
State: VULNERABLE
IDs: CVE:CVE-2017-0143
Risk factor: HIGH
A critical remote code execution vulnerability exists in Microsoft SMBv1
servers (ms17-010).

Disclosure date: 2017-03-14
References:
https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2017-0143
https://blogs.technet.microsoft.com/msrc/2017/05/12/customer-guidance-for-wannacrypt-attacks/
https://technet.microsoft.com/en-us/library/security/ms17-010.aspx

Nmap done: 1 IP address (1 host up) scanned in 2.28 seconds

```

Summary

nmap is a great tool for any pentester/hacker and should be in everyone's toolbox. When the NSE scripts are added to nmap, it becomes a versatile tool for vulnerability testing, fuzzing, brute forcing and even exploitation! With all this capability at the hacker/pentester's fingertips, nmap and nmap scripts can supplant a number of tools in your toolbox.