

Active reconnaissance and Information Gathering

ACCA7033

Dr. Ayman El Hajjar

2026/27

Lab environment

PLEASE READ

- You will find that the lab is showing activities on the Linux server. 192.168.56.101, however this is for one server.
- For your assignment, you are expected to conduct all activities on both the Linux and the Windows servers.
- Below are the IP addresses I am using for my machines. Your machines can have different ip Addresses
 - Kali Linux IP address- **192.168.56.103**– this is on my machien. on your it could be different
 - Linux Server IP address **192.168.56.101** – this is on my machien. on your it could be different
 - Windows Server IP address **192.168.56.99**
- In some activities you need **only** your kali Linux machine to be connected to the internet.
- Resolvconf is a dns name server. On the new kali linx (2020) it doesn't come pre installed. To install it type **sudo apt-get install resolvconf**
- It is needed for nmap for example to identify the domain name resolver for your machine.
- Your dhcp server should have given you the correct dns address
- Let us enable resolvconf service first and then start it.

- **sudo systemctl enable resolvconf.service**
- **sudo systemctl start resolvconf.service**
- **sudo systemctl status resolvconf.service** - To check status
- Sometimes Linux fails to obtain an IP address from the DHCP server. A simple invoke of a **dhclient** request will allow it to dynamically obtain one. You type **sudo dhclient**
 - This specifically happens when you change the network interface from NAT to host only.
 - You can also manually disconnect the network interface and connect it again.

0.1 Taking advantage of robots.txt

- One step further into reconnaissance, we need to figure out if there is any page or directory in the site that is not linked to what is shown to the common user. For example, a login page to the intranet or to the **content management systems (CMS)** administration. Finding a site similar to this will expand our testing surface considerably and can give us some important clues about the application and its infrastructure.
- We will use the **robots.txt** file to discover some files and directories that may not be linked to anywhere in the main application.
- Browse to <http://192.168.56.101/drupal/>
- Now we add robots.txt to the URL.
- This file tells search engines that the indexing of the directories should avoid certain directories. For example modules folder should not be crawled by the search engine spider.
- Let's browse to <http://192.168.56.101/drupal/modules/>

How these tools work

- robots.txt is a file used by web servers to tell search engines about the directories or files that they should index and what they are not allowed to look into.
- Taking the perspective of an attacker, this tells us if there is a directory in the server that is accessible but hidden to the public using what is called "security through obscurity" (that is, assuming that users won't discover the existence of something, if they are not told about it).
- Have a look at the robots.txt file at this address: <http://192.168.56.101/drupal/robots.txt>

- You should spend some time on both servers, trying to identifying what services they contain.
- Reconnaissance and information gathering both form an essential stage in every penetration test or auditing activity;
- The more information we have about our target, the more options we will have when it comes to finding vulnerabilities and/or exploiting them.

1 The Network mapper Nmap

One of the most important stages of an attack is information gathering. To be able to launch an attack, we need to gather basic information about our target. So, the more information we get, the higher the probability of a successful attack.

1.1 Finding Files and folders

DirBuster is a tool created to discover, by brute force, the existing files and directories in a web server. We will use it in this recipe to search for a specific list of files and directories. We will use a text file that contains the list of words that we will ask DirBuster to look for.

How this tool works

DirBuster is a mixture of crawler and brute forcer; it follows all links in the pages it finds but also tries different names for possible files. These names may be in a file similar to the one we used or may be automatically generated by DirBuster using the option of "pure brute force" and setting the character set and minimum and maximum lengths for the generated words.

To determine if a file exists or not, DirBuster uses the response codes from the server. The most common responses are listed, as follows:

- **200. OK:** The file exists and the user can read it.
- **404. File not found:** The file does not exist in the server.
- **301. Moved permanently:** This is a redirect to a given URL.
- **401. Unauthorized:** Authentication is required to access this file.
- **403. Forbidden:** Request was valid but the server refuses to respond.

- Create a text file **dictionary.txt** containing the following:
 - info
 - server-status

- server-info
 - cgi-bin
 - robots.txt
 - phpmyadmin
 - admin
 - login
 - chat
- To create the file, you can use a text editor- For example
 - save the file
 - Navigate to **Kali Linux** ⇒ **Web application analysis** ⇒ **Web Crawlers & Directory Bruteforce** ⇒ **dirbuster**.
 - On the DirBuster's window, set the target URL to **http://192.168.56.101:80**
 - Set the number of threads to 20.
 - Select **List based brute force** and click on **Browse**.
 - In the browsing window, select the file we just created (**dictionary.txt**).
 - Uncheck the **Be Recursive** option.
 - Click on **Start**.
 - If we go to the **Results** tab, we will see that DirBuster has found at least two of the files in our dictionary: **server-status** and **phpmyadmin**. The response code 200 means that the file or directory exists and can be read. PhpMyAdmin is a web-based MySQL database administrator; finding a directory with this name tells us that there is a DBMS in the server and it may contain relevant information about the application and its users.
 - You should repeat this activity for all web applications on both servers Any information you obtain will be useful later on.

1.2 DNS Enumeration

Enumeration is a process that allows us to gather information from a network. We will examine DNS enumeration technique. DNS enumeration is the process of locating all DNS servers and DNS entries for an organization. DNS enumeration will allow us to gather critical information about the organization such as usernames, computer names, IP addresses, and so on.

- You should not do this on any website without permission

- **DNS enumeration** is a tool that is pre installed on Kali Linux.
- Kali Linux tools are pre-installed in bin folder of the user.

commands for enumeration

```
dnsenum --enum www.hackthissite.org

dnsenum --enum cwscenario.site
```

- We should get an output with information like host, name server(s), mail server(s), and if we are lucky, a zone transfer.
- Explore the various options using the **man** command to see the additional options. Those options can help if you are looking for something specific.

Note

The tools we have used are not comprehensive and there are many others either pre - installed with Kali or available as open source.

You are encouraged to explore those tools.

1.3 Default scanning with Nmap

Nmap is probably the most used port scanner in the world. It can be used to identify live hosts, scan TCP and UDP open ports, detect firewalls, get versions of services running in remote hosts, and even, with the use of scripts, find and exploit vulnerabilities.

- We first need to see if the server is answering to a ping or if the host is up.
- Let us start with a default nmap scan

Default scan - Internet

- **nmap scanme.nmap.org**
 - Note: There are many different type of scans types. First in this command, we did not use root privilege. The result of this command are limited. You have just used scan type "connect"
- Have a look and see if you were able to detect any open services or services behind ports.

- In the first command, we were connected to the internet and we scanned nmap scanme site.
- Now let us scan our vulnerable machine OWASP. Make sure you switch to host only network.

Default scan

- **nmap 192.168.56.101**
 - Have a look and see if you were able to detect any open services or services behind ports.

1.4 Ports scans

- The basic default Nmap scan `nmap [target]` executes a simple port scan that returns a list of ports. In addition, it returns a service name from a database distributed with Nmap and the port state for each of the listed ports.
- Nmap categorizes ports into the following states:
 - **Open:** Open indicates that a service is listening for connections on this port.
 - **Closed:** Closed indicates that the probes were received, but it was concluded that there was no service running on this port.
 - **Filtered:** Filtered indicates that there were no signs that the probes were received and the state could not be established. It also indicates that the probes are being dropped by some kind of filtering.
 - **Unfiltered:** Unfiltered indicates that the probes were received but a state could not be established.
 - **Open/Filtered:** This indicates that the port was filtered or open but the state could not be established.
 - **Close/Filtered:** This indicates that the port was filtered or closed but the state could not be established.
- The simplest of those scans is the one we used in the previous section.
- You can also identify a DNS server different then the one we set up previously or let it use the system DNS nameserver.
 - **nmap -dns-servers 8.8.8.8,8.8.4.4 scanme.nmap.org**

1.4.1 Specific scan ranges of ports

- Sometimes nmap throw at you a warning that not all ports were scanned.
- This is more common when you are scanning large networks - Nmap by default only scans the trop 1000 ports
- This is not useful if services are running on higher ports - up to 65535 ports exist. Remember the header for ports in an IP address is 16 bits $2^{16} = 65536$.
 - You can specify a specific port by using **-pflag** following by either the range or by one port

- For example **nmap -p 80-3000 nmap.scanme.org** or **nmap -p 80 nmap.org**
- You can do the same for any other network.

1.4.2 Specific scan ranges of IP address

- Nmap supports IP address ranges in different formats, and it is essential that we know how to deal with them.
 - The simplest form is to use the range of IP addresses **from** and **to**
 - For example **nmap 192.168.56.100-103**- In this example we are scanning the range from 100 to 103.
 - You can also use different notations:
 - * **nmap 192.168.56.*** - The whole network
 - * **nmap 192.168.56./24** - The whole subnet

1.5 Russian Roulette- Random targets list scan

- Nmap allows you to scan random targets list of n hosts using **-iR** flag. For example you can scan 100 hosts by typing
 - **sudo nmap -iR 100**
 - This can be useful if you are looking for a specific vulnerability on the Internet. For example if you are looking for a random NFS shares online server you type **nmap -p2049 --open -iR 0** where 0 represents unlimited ip addresses to scan.
 - Use this with caution , you dont know what you are scanning. Port scanning without permission is not very welcome, and it is even illegal in some countries.

1.6 Logging scans

- If you are carrying an active reconnaissance assessment over a large network or a server that has a large number of services running behind it, it is a good idea to log all your scans to a file.
- There are three different types of scans for scan. These are .xml, .nmap and .gnmap.
 - To record all the reconnaissance activity in a log you add **-oA** flag and the name of the output log.
 - For example **nmap scanme.nmap.org -oA sitelog** or **nmap 192.168.56.101 -oA servers-log**

1.7 Tracing routes

- Ping scans allows including trace route information of the targets. Use the Nmap option **-traceroute** to trace the route from the scanning machine to the target host:

– **sudo nmap -sn --traceroute cwsenario.site**

1.8 Host detection methods

A simple way would be by performing a **ping** on the target network. Of course, this can be rejected or known by a host, and we don't want that.

- In order to scan a host effectively it is essential to identify whether they are "alive" or "online".
- Many administrators hide their systems from the internet and they can appear offline until further probed.
- one way in doing this is by using ping in command line - An ICMP echo request that will revoke a message saying yes I am here - or no answer.
- You can run a ping only sweep using Nmap with **-sn** flag. This will force nmap simply to check whether it is alive or not regardless of the port it is behind.

Live hosts in a network

- **nmap -sn -n 192.168.56.***
- We have just scanned the whole network range of 192.168.56.0 to 192.168.56.255 looking for live hosts. The * represents the whole network range- This is class c.

1.9 Agnostic scan

- Even with the previous scan, a system can still hide from ping sweeps.
- Nmap offers a flag that provides agnostic method for scanning.
- In the previous command, nmap was using default port addresses. Which as we said before means that it could be skipping many services on higher than 1000 port number.
 - **nmap -Pn scanme.nmap.org**
 - Compare your results with the a simple ping command for the same site as we did in the previous section using **nmap -sn scanme.nmap.org**
 - Try my testing site **nmap -Pn cwsenario.site**

List scan

- You can also add a list scan flag **-sL**. This is useful if you want to get reverse DNS lookups and to understand the live hosts.
- For example
- **nmap -sL cwscenario.site**

1.10 Scanning UDP services

- Since UDP is a connectionless service, scanning them is more difficult since they don't respond and even when they do respond, they take a large amount of time making them slow and less reliable than TCP scans.
- Many important services can run on UDP and it is essential that you scan UDP services as well. For example some VPN services have their listening ports set on UDP only. DNS only listen exclusively on UDP.
- To scan with UDP you need to invoke the flag **-sU**; **It will be slow. Make sure you have do this at home when you have plenty of time**
- UDP scans also requires elevated privilege.

UDP services

- **sudo nmap -sU cwscenario.site**
- **sudo nmap -sU 192.168.52.102**

1.11 Scanning TCP services

- The two basic scans for Nmap using TCP are connect scan using the flag **-sT** and the SYN stealth scan **-sS**
- These are also called full and half connection scans.
 - **-sT** flag- TCP connect scan is the default TCP scan type when SYN scan is not an option. This is the case when a user does not have raw packet privileges or is scanning IPv6 networks. Instead of writing raw packets as most other scan types do, Nmap asks the underlying system to establish a connection with the target machine and port by issuing the connect system call. This can be dangerous and raise alarms as you are establishing a full connection
 - **-sS** flag- half connection scan for TCP SYN scan- SYN scan is the default and most popular scan option for good reasons. It can be performed quickly, scanning thousands of ports per second on a fast network not hampered by

restrictive firewalls. SYN scan is relatively unobtrusive and stealthy, since it never completes TCP connections. It also allows clear, reliable differentiation between the open, closed, and filtered states. This technique is often referred to as half-open scanning, because you don't open a full TCP connection. You send a SYN packet, as if you are going to open a real connection and then wait for a response. A SYN/ACK indicates the port is listening (open), while a RST (reset) is indicative of a non-listener. If no response is received after several retransmissions, the port is marked as filtered. The port is also marked filtered if an ICMP unreachable error (type 3, code 1, 2, 3, 9, 10, or 13) is received.

TCP scans services

- **sudo nmap -sT cwsenario.site**
- **sudo nmap -sT 192.168.52.102**
- **sudo nmap -sS cwsenario.site**
- **sudo nmap -sS 192.168.52.102**

- You can also specifically use TCP SYNC flag by invoking **-PS** flag.
 - For example you can type **nmap -sn -PA 192.168.56.1/24**
 - I am using sn to stop ports scanning and only identify hosts,
 - The **-PA** flag tells Nmap to use a ACK SYN ping scan. This type of ping scan works in the following way:
 - * Since we are expecting an acknowledgement, only hosts that are online will respond.
 - * Nmap sends an empty TCP packet and wait for an ACK for each host.
- You can also specifically use TCP ACK flag by invoking **-PS** flag.
 - For example you can type **nmap -sn -PS 192.168.56.1/24**
 - I am using sn to stop ports scanning and only identify hosts,
 - The **-PS** flag tells Nmap to use a TCP SYN ping scan. This type of ping scan works in the following way:
 - * Nmap sends a TCP SYN packet to port 80.
 - * If the port is closed, the host responds with an RST packet.
 - * If the port is open, the host responds with a TCP SYN/ACK packet indicating that a connection can be established.
 - * Afterward, an RST packet is sent to reset this connection.

- You can either specify the port on to scan on with **-PS** flag by entering the port number.
 - * For example you can type **nmap -sn -PS80,21 192.168.56.1/24** - To scan on ports 80 and 21.

1.11.1 Special TCP services

- The problem with the two previous flags is that if services are running behind an Intrusion detection system (IDS) they might be dropping SYN scans and full scans and thus you are not obtaining the full figure of the network services running and live.
- Three other TCP scan types can be used. They are **FIN**, **Xmas tree** and **NULL** scans.
 - FIN scan using **-sF** flag. Type **sudo nmap -sF cwsenario.site** - There is no abnormal response, which is expected as there is already a service running on port 80 and the service will terminal normally.
 - The next scan type is called the Xmas Tree scan—so-called because it is as if a packet is lit up like a Christmas tree! The Xmas Tree scan **-sX** flags works by flagging FIN, URG, and PUSH flags on a packet header.
 - The final scan is called Null scan, a null scan **-sN** simply sets no flags on the packet header sent to the target port. This means you will be launching a scan that targets every single possible flag in TCP.
 - It is worth noting that those special scans do not work against Microsoft windows hosts.

1.12 Scanning using ICMP packets services

- Ping scans are used to determine if a host is online. ICMP echo request messages were designed specifically for this task, and naturally, ping scans use these packets to reliably detect the status of a host.
- We can use ICMP echo request with Nmap using **-PE** flag.

ICMP echo request

- **sudo nmap -PE cwsenario.site**
- **sudo nmap -PE 192.168.52.102**

1.13 discovering hosts with ARP ping scans

- ARP ping scans are the most effective way of detecting hosts in LAN networks.

- This makes them the preferred technique when scanning local Ethernet networks, and Nmap will use it even if other ping options were specified.
- Nmap uses its own algorithm to optimize this scanning technique with the flag **-PR**.

Hosts discovery using ARP scans

- **sudo nmap -sn -PR 192.168.56.101**
- Note that the network interface MAC address was returned for your target IP address
- You can even see the whole ARP resolution request by using **--packet-trace option** flag
- Type **sudo nmap -sn -PR --packet-trace 192.168.56.101**

Why this is useful

- Suppose the server you are targeting employs an Intrusion detection system, since we now know the MAC address of the target we can spoof it and evade the IDS. **sudo nmap -sn -PR --spoof-mac 08:00:27:88:93:F6 192.168.56.101**
- Note that the network interface MAC address was returned for your target IP address
- You can even see the whole ARP resolution request by using **--packet-trace option** flag
- Type **sudo nmap -sn -PR --packet-trace 192.168.56.101** -Note this is my virtual machine MAC address, your one will be different.

1.14 Fingerprinting

1.14.1 Fingerprinting for Operating system detection

- Determining the services running on specific ports will ensure a successful pentest on the target network. It will also remove any doubts left resulting from the OS fingerprinting process.
- Nmap **-O** flag works by comparing the fingerprint identified on a host by a large database of fingerprints for services, servers and operating systems used worldwide.
- It uses the Common Platform Enumeration (CPE) as the naming scheme for service and operating system detection. This convention is used in the information security industry to identify packages, platforms, and systems.

- The complete documentation of the tests and probes sent during OS detection can be found at [OS detection methods site](#)

Operating system detection

- **`nmap -O 192.168.156.102`**

1.14.2 Fingerprinting for service detection

- Running a service version scan is very simple; all we need to do is add an additional flag, `-sV`. This means that we're conducting a service version scan, which can demonstrate which version of each software is running.
- This is particularly useful if someone is running a service on a non-default port (that does not match up with `/etc/services`)—in such instances, it's even more important to be able to figure out exactly what's running.

Default scan - Internet

- **service version scans**
 - [nmap site](#) **`nmap -sV -n scanme.nmap.org`**
 - [one server](#) **`nmap -sV -n 192.168.56.101`**
- Now let us compare the results of this command to the previous one.
- Can you imagine what is the importance of the information you obtained from this command in comparison with the previous one?

Understanding the reason flag

- If you think of the TCP three way handshake, you can ask Nmap to give you the reason behind the different status of the flags you can have **open, closed and filtered**.
- Read these definitions on port flags on the Nmap site
 - [OWASP](#) **`nmap -sV -n --reason 192.168.56.101`**

1.15 Optimising scans

1.15.1 Increasing version detection intensity

- You can increase or decrease the amount of probes to use during version detection by changing the intensity level of the scan with the argument `--version-intensity [0-9]`, as follows:
- **`nmap -sV --version-intensity 9 192.168.56.101`**

- This Nmap option is incredibly effective against services running on nondefault ports due to configuration changes.

1.15.2 Aggressive detection mode

- Nmap has a special flag to activate aggressive detection, namely **-A**. Aggressive mode enables OS detection (**-O**), version detection (**-sV**), script scanning (**-sC**), and traceroute (**--traceroute**). This mode sends a lot more probes, and it is more likely to be detected, but provides a lot of valuable host information. You can try aggressive detection with the following command:
- **nmap -A 192.168.56.101**

1.15.3 Configuring OS detection

- If OS detection fails, you can use **--osscan-guess** to force Nmap to guess the Operating system
- **nmap --osscan-guess 192.168.56.101**
- This Nmap option is incredibly effective against services running on nondefault ports due to configuration changes.

1.15.4 Increasing verbosity in scans

- Verbosity lets timing, parallelism, and internal debugging information to display straight to the console while scans run.
- This can be great to figure out when we need to try to optimize scans in one of the several ways. When running a scan in increased verbosity, you can also hit Enter to see how far the scan has progressed, and how far it has to go before completing its current target file. There are several different levels of verbosity, but I usually use the third-level.
 - The first level of verbosity gives a very basic information about a scan's progress, and can be invoked by using the **-v** flag.
 - The second level of verbosity gives more information, including some network and packet information, and can be invoked by using **-vv** as a flag.
 - Lastly, triple verbosity—which gives out the most information of a scan—can be invoked with the **-v3** or **-vvv** flag.
 - If you want to make Nmap less verbose than normal, you can also use the **--reduce-verbosity** flag.

1.15.5 Nmap timing optimisation

- The easiest way to make a scan run faster is to use the built-in timing flags. These flags are invoked using **-T** and a number from 1 (slowest) to 5 (fastest). The default scanning speed is **-T3**, right in the middle.
- There are a few risks to use significantly faster scanning, since it creates certain unreliable aspects in the scan. Particularly, if your network interface is known to be reliable, these options should be used with caution!
- If we are working on a penetration testing engagement with intrusion detection systems and intrusion prevention systems running, it can be very beneficial to run these slow scans.
 - You can invoke a timing flag by typing for example **nmap -T2 192.168.56.101** or **time nmap -T2 192.168.56.101**

1.15.6 Increasing and decreasing parallelism

- You can increase or decrease the number of scans running in parallel to make your scans more efficient.
- For example by increasing the value of **--min-parallelism** flag for up to 10 or 12—you can force Nmap to scan at least that fast.
- Nmap will still make the scan run faster if it needs to, which reduces some of the risk.
- On the other hand, it's possible to set the value of **--max-parallelism** flag as low as 1.

Parallelism optimization

- **nmap -sU --min-parallelism 10 192.168.56.101**

1.15.7 Dealing with stuck hosts.

- Sometimes hosts can take very long time to respond. If you are dealing with a large number of hosts, it is better to skip it and come back to it later.
- You can use **--host-timeout** flag and specify the value in minutes before the scan moves to another host.

Stuck hosts

- **nmap -Pn 10 192.168.56.* --host-timeout 1m**

1.16 List of targets

Many times we want to scan a list of IP addresses and not an entire subnet. We can use any text editor and create a list of IP addresses and "feed" it to nmap. This is the list of IP addresses I want to scan.

Service fingerprinting

- `gedit scanlist.txt`
 - Inside this file put several IP
 - Type `nmap -iL scanlist.txt` to scan all IP addresses in the list.
 - **Note:** You need to be in the same location as your scanlist file for the command to run.
- Nmap is a port scanner, this means that it sends packets to a number of TCP or UDP ports on the indicated IP address and checks if there is a response. If there is, it means the port is open; hence, a service is running on that port.
 - In the first command, with the `-sn` parameter, we instructed Nmap to only check if the server was responding to the ICMP requests (or pings). Our server responded, so it is alive.
 - The second command is the simplest way to call Nmap; it only specifies the target IP address. What this does is ping the server; if it responds then Nmap sends probes to a list of 1,000 TCP ports to see which one responds and then reports the results with the ones that responded.
 - The third command adds the following two tasks to the second one:
 - `-sV` asks for the banner—header or self identification—of each open port found, which is what it uses as the version
 - `-O` tells Nmap to try to guess the operating system running on the target using the information collected from open ports and versions

In each command we used, you can use **man** command to look at the various options that it can be used with.

- Other useful parameters when using Nmap are:
 - `-sT`: By default, when it is run as a root user, Nmap uses a type of scan known as the SYN scan. Using this parameter we force the scanner to perform a full connect scan. It is slower and will leave a record in the server's logs but it is less likely to be detected by an intrusion detection system.

- `-Pn`: If we already know that the host is alive or is not responding to pings, we can use this parameter to tell Nmap to skip the ping test and scan all the specified targets, assuming they are up.
- `-v`: This is the verbose mode. Nmap will show more information about what it is doing and the responses it gets. This parameter can be used multiple times in the same command: the more it's used, the more verbose it gets (that is, `-vv` or `-v -v -v -v`).
- `-p N1,N2,...,Nn`: We might want to use this parameter if we want to test specific ports or some non-standard ports, where `N1` to `Nn` are the port numbers that we want Nmap to scan. For example, to scan ports 21, 80 to 90, and 137, the parameters will be: `-p 21,80-90,137`.
- `--script=script_name`: Nmap includes a lot of useful scripts for vulnerability checking, scanning or identification, login test, command execution, user enumeration, and so on. Use this parameter to tell Nmap to run scripts over the target's open ports. You may want to check the use of some Nmap scripts at: <https://nmap.org/nsedoc/scripts>.

1.17 Using Nmap scripting Engine (NSE) for Reconnaissance

- The Nmap scripting engine is one of Nmap's most powerful and, at the same time, most flexible features. It allows users to write their own scripts and share these scripts with other users for the purposes of networking, reconnaissance, etc.¹ These scripts can be used for:
 - Network discovery
 - More sophisticated and accurate OS version detection
 - Vulnerability detection
 - Backdoor detection
 - Vulnerability exploitation
- The NSE is a framework that runs code written in the programming language Lua with specific flags that the engine can parse.
- Lua is a lightweight, fast, and interpreted programming language—one that has the most fame for scripting user interfaces for computer games such as World of Warcraft—that has a similar syntax to other contemporary interpreted languages.
- For now, we will be using pre-written scripts
- This website lists all the scripts that are part of the official Nmap script repository
Nmap official script repository

¹<https://null-byte.wonderhowto.com/how-to/hack-like-pro-using-nmap-scripting-engine-nse-for-reconnaissance-0158681/>

- These scripts are broken down into several categories, each with their own specific use case. It's worth noting that sometimes these scripts can be in several categories, depending on the full functionality of the script. The categories and their definitions are as follows:
 - **Auth:** These scripts attempt to authenticate to services, and can verify found credentials
 - **Broadcast:** These scripts broadcast certain protocols in order to find out whether or not they are listening
 - **Brute:** These scripts attempt brute force or dictionary-based attacks against network services
 - **Default:** This is the default category of scripts that may run when a scan is initiated
 - **Discovery:** These scripts attempt to enumerate sensitive information from hosts and network services
 - **Denial of Service (DoS):** These scripts may cause disruption to the service that is being scanned
 - **Exploit:** These scripts attempt to execute an exploit that exploits a given vulnerability
 - **External:** These scripts query third-party databases, such as DNS blacklists, to gather additional information about targets
 - **Fuzzer:** These scripts send random "garbage" information to services in order to attempt to find flaws in the software
 - **Intrusive:** These scripts are an umbrella category for any script that may cause damage or be intrusive to the service itself
 - **Malware:** These scripts attempt to find instances of the known malware.
 - **Safe:** These scripts are verified to not cause harm to servers
 - **Version:** These scripts attempt to identify specific versions—as well as information disclosures—from specific services in a more in-depth way than normal service version detection
 - **Vuln:** These scripts identify the known vulnerabilities in services

1.17.1 Running Nmap Scripts

- Let us first update the database of scripts by typing **sudo nmap --script-updatedb**
- Once you have the scripts up to date, you can now either run them as groups or individual inside the groups.
- For example to run them as a group

How to run scripts

- **nmap 192.168.56.101 --script default** -- For the default group
 - **nmap 192.168.56.101 --script "http-*" --** will scan all scripts related to a web server http
 - Or if you are looking for a specific script you can visit the folder where all scripts are stored by typing **cd /usr/share/nmap/scripts**.
 - You can type **ls** to list those scripts.
 - You can also use pipeline with **grep** to find a specific keyword. For example
 - **ls | grep drupal**
-
- Nmap includes for example a couple of scripts to test for the presence of a WAF. Let's try this on our vulnerable-vm.
 - **nmap -p 80,443 --script=http-waf-detect 192.168.56.101**
 - You can also check if WAF is installed on other websites when you are connected to the Internet. Make sure vulnerable machine should be off when you connect to the Internet.
 - **nmap -p 80,443 --script=http-waf-fingerprint cwsenario.site**
 - You can also enable scripts and combine them with the numerous flags of Nmap.
 - Have a look at this example **sudo nmap --script dns-brute cwsenario.site -T5**
 - Note the T5 otherwise the IDS on the system will block me. Try with lower timing and more deep scanning.

2 Other scanners tools

- Although it's the most popular, Nmap is not the only port scanner available and, depending on varying tastes, maybe not the best either. There are some other alternatives included in Kali Linux, such as:
 - unicornscan
 - hping3
 - masscan
 - Metasploit scanning modules
- Using **masscan**, we can also identify the application running on a specific port or a range of ports, as shown in the following example.

- `sudo masscan 192.168.56.101 -p80`