

Week 3: OSINT, Information gathering, Reconnaissance and Enumeration activities

6COSC002W

Dr. Ayman El Hajjar

Week 3/4

STOP and READ

Before you start your work in the lab you need to ensure the following steps when setting up your lab environment:

1. Delete all Virtual machines that already exist inside VirtualBox to ensure that you are starting your labs from a clean state
2. Before you import the VMs you need from the C:\VirtualMachines folder, check if you have the latest Kali version. If not download them by following the lab environment document.
3. The latest Kali version is "Kali-Linux-2022.3-UoW-virtualbox-amd64"
4. Make sure you import the VMs to C:\VirtualMachines

Requirements and Notes

- **NOTE-** In some activities you need **only** your Kali Linux machine to be connected to the internet.
- In this lab, you need the following VM machines
 1. Kali Linux
 2. OWASP Vulnerable machine
- This is not a HACKING TUTORIAL and YOU ARE RESPONSIBLE FOR YOUR ACTS OUTSIDE THIS LAB



Food for thought: Kali Linux Slogan

1 Getting to know web applications on a vulnerable VM

OWASP-bwa contains many web applications, intentionally made vulnerable to the most common attacks. Some of them are focused on the practice of some specific technique while others try to replicate real-world applications that happen to have vulnerabilities.

1. With OWASP vm running, open your Kali Linux host's web browser and go to <http://192.168.56.102>. You will see a list of all applications the server contains.
2. For **Damn vulnerable Web Application**. Use username **admin** and password **admin**. We can see a menu on the left; this menu contains links to all the vulnerabilities that we can practice in this application: Brute Force, Command Execution, SQL Injection, and so on.
3. **OWASP WebGoat.NET**. This is a .NET application where we will be able to practice file and code injection attacks, cross-site scripting, and encryption vulnerabilities. It also has a WebGoat Coins Customer Portal that simulates a shopping application and can be used to practice not only the exploitation of vulnerabilities but also their identification

The applications in the home page are organized in the following six groups:

- **Training applications:** These are the ones that have sections dedicated to practice-specific vulnerabilities or attack techniques; some of them include tutorials, explanations, or other kind of guidance.
- **Realistic, intentionally vulnerable applications:** Applications that act as real-world applications (stores, blogs, and social networks) and are intentionally left vulnerable by their developers for the sake of training.
- **Old (vulnerable) versions of real applications:** Old versions of real applications, such as WordPress and Joomla are known to have exploitable vulnerabilities; these are useful to test our vulnerability identification skills.

- **Applications for testing tools:** The applications in this group can be used as a benchmark for automated vulnerability scanners.
- **Demonstration pages / small applications:** These are small applications that have only one or a few vulnerabilities, for demonstration purposes only. OWASP demonstration application: OWASP AppSensor is an interesting application, it simulates a social network and could have some vulnerabilities in it. But it will log any attack attempts, which is useful when trying to learn; for example, how to bypass some security devices such as a web application firewall.



Figure 1: OWASP Web Interface

Passive Reconnaissance: Open Source Intelligence (OSINT)

PLEASE READ BEFORE YOU START THIS SECTION

For this section of the lab, you need to make sure that your kali Linux machine is connected to the Internet (NAT)

For any other sections, if you need to be connected to the Internet, to install an application or for any other reasons, this will be made clear to you.

OSINT can be carried out by conducting different activities on the internet to search for information. Tools are there to put all of those activities in one place and to automate them.

There are many tools to use to conduct your OSINT. We will be using some, but feel free to explore others.

You can see what type of sources can be used in OSINT investigation using the OSINT Framework

Feel free to use **cwscenario.site** domain for all your OSINT investigations. This is a domain I own and only use for testing.

1.1 TheHarvester

- Since theHarvester searches for information already are online, it means your search will go unnoticed and will not raise any alarm. This is passive reconnaissance and you are not actually searching the web server of the company but public information about them that are available on the Internet.
- There are many options to use with theHarvester tool, if you type **theHarvester -h** on an open terminal, you will see a collection of different options you can use.
- We will focus on **-d** option to specify the domain, **-b** option to specify the data source (google, linkedin, bing, twitter, etc..) . We also need to use **-l** option for certain sources as they block your request if the number of requests is not set.
- Lets see an example below

```
theHarvester -d westminster.ac.uk -b google -l 100
```

- you can also specify all sources

```
theHarvester -d cwscenario.site -b all
```

- It is useful to save the output of your OSINT so that you can use it to carry further attacks at a later stage. You can use the **-f** option to specify the filename to save the output to. I will save them in an html format so that I can open them in a browser.

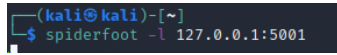


Figure 2: Starting spiderfoot on localhost port 5001

theHarvester -d cwsenario.site -b all -f myresults.html

1.2 SpiderFoot

- Another OSINT you should explore is Spiderfoot.
- SpiderFoot is preinstalled on Kali.
- To start you need to open a terminal and run it. You need to specify the server address and the port number to use for its application.

spiderfoot -l 127.0.0.1:5001

- While Spiderfoot is running on the terminal, open a new browser and point it to 127.0.0.1:5001 (you can specify any other port number you want. I suggest you use a number larger than 5000)
- Spiderfoot is module based, some of the modules requires the use of API keys. In some cases, these modules will not function without properly setting up the API keys first. In other modules, the API keys are only to speed up the scanning. Check this website for more information on the modules. Ignore the installation part.
- At the beginning, and since we are conducting an OSINT investigation, at least at this stage.
- Click on "New Scan"
- Choose "Passive scan" and give a name for the scan and for ip address you can choose either an IP address or a domain name to investigate.
- Give it some time until it finishes running. Results will come gradually until the scan finishes.
- Once it is done, explore the obtained results.

1.3 Recon-ng

- Recon-ng is an information-gathering tool that uses many different sources to gather data, for example, on Google, Twitter, and Shodan.
- Let's try to gather public information about our target.
- Let's start recon-ng by typing:

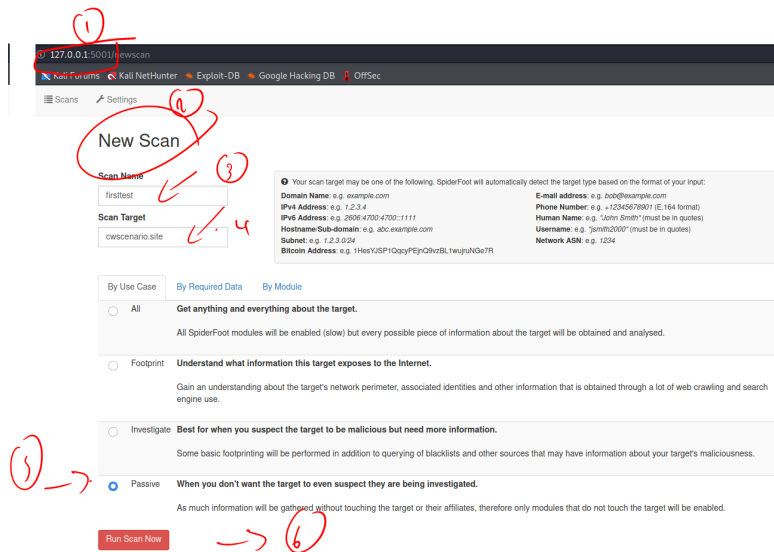


Figure 3: Creating new scan on my testing domain name. Feel free to use this domain for testing.

recon-ng

- You notice that recon-ng throws a warning that no modules are install.
- let us install the hackertarget first. Hackertarget is a module that search for sub-domains on any given domains.

marketplace install hackertarget

- To use hackertarget module. We type :

modules load hackertarget

- Now we need to set the target to use for our investigation:

options set source cwsenario.site

- Let's run the module now:

run

- **Note:** my domain will not bring a lot of results, you can use any other website as these are public information and we are not touching the server.
- There are many other modules that you can explore, for example you can install all marketplace modules by typing:

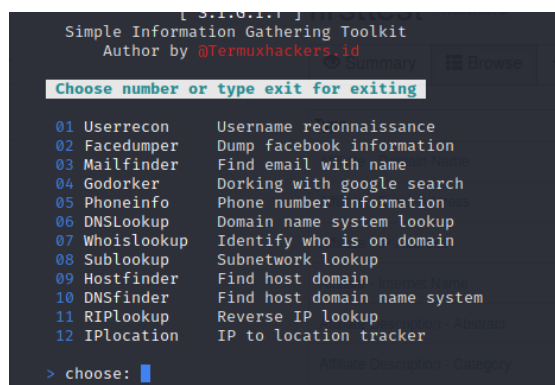
marketplace install all

- to search the module use

marketplace search

1.4 SIGIT - Simple Information Gathering Toolkit

- Let us now clone the SIGIT tool, another powerful information gathering tool.
- To clone it and install, we type:
 - `sudo git clone https://github.com/termuxhackers-id/SIGIT.git`
 - `cd SIGIT`
 - * we navigate to folder
 - `sudo chmod +x installkali.sh`
 - * we make the Kali installer "installkali.sh" executable
 - `sudo ./installkali.sh`
 - * we execute the installer
- Let's run it now and see what information we can obtain. To run it open a terminal and type:
sigit
- SIGIT brings an interface with a number of options you can use.
- Choose the one you need and put its number.
- Try for example your username, or the username you usually use for many services online and see if it can find it.
- You can also try to find a specific email with specific names and many other information.



```
[ 3.1.0.1.1 ]
Simple Information Gathering Toolkit
Author by @Termuxhackers.id

Choose number or type exit for exiting

01 Userrecon      Username reconnaissance
02 Facedumper     Dump facebook information
03 Mailfinder     Find email with name
04 Godorker       Dorking with google search
05 Phoneinfo      Phone number information
06 DNSLookup      Domain name system lookup
07 Whoislookup    Identify who is on domain
08 Sublookup      Subnetwork lookup
09 Hostfinder     Find host domain
10 DNSfinder      Find host domain name system
11 RIPLookup      Reverse IP lookup
12 IPlocation     IP to location tracker

> choose: █
```

Figure 4: SIGIT - Simple Information Gathering Toolkit

```
[ 3.1.0.1.7 ]
Simple Information Gathering Toolkit
Author by @Termuxhackers.id

Choose number or type exit for exiting

01 Userrecon      Username reconnaissance
02 Facedumper     Dump facebook information
03 Mailfinder     Find email with name
04 Godorker       Dorking with google search
05 Phoneinfo      Phone number information
06 DNSLookup      Domain name system lookup
07 Whoislookup    Identify who is on domain
08 Sublookup      Subnetwork lookup
09 Hostfinder     Find host domain
10 DNSfinder      Find host domain name system
11 RIPlookup      Reverse IP lookup
12 IPlocation     IP to location tracker

> choose: █
```

Figure 5: SIGIT - Simple Information Gathering Toolkit

Information gathering

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.

PLEASE READ BEFORE YOU START THIS SECTION

In this section, both the OWASP VM and Kali VM should be running.

Both VMs should be set on host only adapter.

We will be mainly using nmap however there are many other tools we can be used. I suggest you explore the various tools available pre-installed on Kali Linux in your own time. To access them you click on kali home button, and select → **01 - Information Gathering**. Yopu will find tools about OSINT, Information Gathering, and tools to enumerate networks ports and addresses, services and protocols

Setup DNS server

- Before we start with the Information Gathering activity, we need to setup the DNS server, otherwise nmap will keep on complaining that it cannot find the DNS server for our host only connection.
- Type:
 - `sudo nano \etc\resolv.conf`
- add
 - `nameserver 6.7.8.9`

Scanning and identifying services 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:

Is Server Responding?

- `nmap -sn 192.168.56.102`

- 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 that it is up. If you don't get this, make sure you have OWASP is up and is on the host only network.
- Now that we know that it's up, let's see which ports are open

Open ports?

```
nmap 192.168.56.102
```

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.

- Now, we will tell Nmap to ask the server for the versions of services it is running and to guess the operating system based on that.

Guess Operating System

```
sudo nmap -sV -O 192.168.56.102
```

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

Food for thought

- - Put together all the information you gathered for this activity and think about what those information? what can you use them? what did they help you identify? How they can be potentially used at a later stage to help you attack the vulnerable machine?
- Type **man nmap** and read about the various options you have used.

Identifying active machines

Before attempting a target to conduct penetration testing on, we first need to identify the active machines that are on the target network range. 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.

- Let's begin the process of locating active machines by opening a terminal window.
- Using Nmap we can find if a host is up or not.

commands for determining network range

```
nmap -sP 192.168.56.102
```

- You can also use Nmap to find a collection of hosts in the same network if they are active or no.

commands for determining network range

```
nmap -sP 192.168.56.*
```

Finding open ports

With the knowledge of the victim's network range and the active machines, we'll proceed with the port scanning process to retrieve the open TCP and UDP ports and access points.

- Let's begin the process of finding the open ports by opening a terminal window

Finding open ports

```
nmap 192.168.56.102
```

- We can also explicitly specify the ports to scan (in this case, we are specifying only the first 1000 ports)

Finding open ports

```
nmap -p 1-1000 192.168.56.102
```

- We can also explicitly specify which port to scan on a specific host as in 1 below or for the whole network active devices as in 2 below.

Finding open ports

```
nmap -p 22 192.168.56.102
```

```
nmap -p 22 192.168.56.*
```

- You can output the result to a specific file in a specified format

Finding open ports

```
nmap -p 22 192.168.56.* -oG ~/Desktop/myresults.txt
```

- **Note:** The symbol ~ means the home directory. This is the same as navigating to /home/kali. Kali is the username in this case.

Fingerprinting for Operating systems

At this point of the information gathering process, we should now have documented a list of IP addresses, active machines, and open ports identified from the target organization. The next step in the process is determining the running operating system of the active machines in order to know the type of systems we're pentesting.

- Let's begin the process of OS fingerprinting from a terminal window
- Using **Nmap**, we issue the following command with the **-O** option to enable the OS detection feature.

OS fingerprinting

```
nmap -O 192.168.56.102
```

- The terminal should warn you that this command requires root privilege! This is a very noisy command and nmap is warning you that an Intrusion detection system will pick it up.
- Root privilege means to use **sudo** before the command to get Super User privilege.

Fingerprinting for services

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.

- Let's begin the process of service fingerprinting by opening a terminal window
- Open a terminal window and issue the following command

Service fingerprinting

```
nmap -sV 192.168.56.102
```

Spoofing & Decoy Scan

When we are scanning machines that are not ours, we often want to hide our IP (our identity). Obviously, every packet must contain our source address or else the response from the target system will not know where to return to. ¹ The same applies to spoofing our IP when using nmap. We CAN spoof our IP address (-S) in nmap, but as a result, any response and any info we are trying to gather will return to the spoofed IP. Not very useful, if we are scanning for info gathering. A better solution is to obfuscate our IP address. In other words, bury our IP address among many IP addresses so that the network/security admin can't pinpoint the source of the scan. Nmap allows us to use decoy IP addresses so that it looks like many IP addresses are scanning the target.

- We can do this by using the -D switch. the D switch will simply give several IP addresses. In another word, different IP addresses will show up on the victim machine including yours but it will be difficult to pinpoint which one is yours.
- Open a terminal window and issue the following command

Service fingerprinting

```
sudo nmap -sS 192.168.56.102 -D 10.0.0.1,10.0.0.2,10.0.0.4
```

UDP scan

Up until this point, all of our scans have been for TCP ports. Some services and ports use UDP to communicate to the outside world. Our previous scan types (-sS and -sT) will not find UDP ports as they are only looking for TCP ports. Some services only run on UDP, such NTP (port 123) and SNMP (port 161). To find these ports and services, we need to do a UDP scan. We can do this with the -sU switch:

¹<https://null-byte.wonderhowto.com/how-to/hack-like-pro-advanced-nmap-for-reconnaissance-0151619/>

- To find these ports and services, we need to do a UDP scan. We can do this with the -sU switch

Service fingerprinting

```
sudo nmap -sU 192.168.56.102
```

- **Note:** UDP scan takes longer than TCP scans. This is because UDP is a connectionless protocol.
- Scanning all UDP ports will take longer and is resource intensive due to the lack of the three way handshake so all will go through and no need for acknowledgment regardless whether it is used or not.

Reason

Note in the output from the UDP scan above that some ports are reported as open/filtered. This indicates that nmap cannot determine whether the port is open or it is filtered by a device such as a firewall. Unlike TCP ports that respond with a RST packet when they are closed, UDP ports respond with an ICMP packet when they are closed. This can make scans far less reliable, as often the ICMP response is blocked or dropped by intermediate devices (firewalls or routers). Nmap has a switch that will return the reason why it has placed a particular port in a particular state. For instance, we can run the same UDP scan as above with the --reason switch and nmap will return the same results, but this time will give us the reason it has determined the particular state of the port.

Service fingerprinting

```
sudo nmap -sU --reason 192.168.56.102
```

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

- mousepad scanlist.txt
- Inside this file put several IP
- To scan all IP addresses in the list type:

```
nmap -iL scanlist.txt
```

- **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>.

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
- amap
- Metasploit scanning modules

An example of one of them below. If you cannot find it, you can install it by typing **sudo apt-get install amap**

- Using **amap**, we can also identify the application running on a specific port or a range of ports, as shown in the following example.

Service fingerprinting

- You will need to install it first. To install it, type:
sudo apt-get install amap
- Once it is installed, we can use it as following:
amap -bq 192.168.56.102 20-1000
- Look at the results and compare them with what you obtained with nmap.

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.² These scripts can be used for:

- Network discovery
- More sophisticated and accurate OS version detection
- Vulnerability detection
- Backdoor detection
- Vulnerability exploitation

²<https://null-byte.wonderhowto.com/how-to/hack-like-pro-using-nmap-scripting-engine-nse-for-reconnaissance-0158681/>

Locate Nmap Scripting Engine

Nmap is installed on most pentesting distributions. To locate Nmap scripting Engine, you need to move to the script where they are installed.

Nmap Scripting Engine location

- `cd /usr/share/nmap/scripts`

Food for thought

- Can you categorize the various scripts ? which script you think will be useful when gathering information about our OWASP Vulnerable machine?

Example: Identifying a web application Firewall

- Nmap includes a couple of scripts to test for the presence of a WAF. Let's try some on our vulnerable-vm.

Detecting WAF

```
nmap -p 80,443 --script=http-waf-detect 192.168.56.102
```

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.

fingerprinting WAF

```
nmap -p 80,443 --script=http-waf-fingerprint www.tryhackme.com  
nmap -p 80,443 --script=http-waf-fingerprint www.cwscenario.site
```

- Now, let's try the same command on a server that actually has a firewall protecting it. Here, we will use the university website.

Detecting WAF

```
nmap -p 80,443 --script=http-waf-detect www.tryhackme.com  
nmap -p 80,443 --script=http-waf-detect www.cwscenario.site
```

- We can do both as well.

Detecting WAF

```
nmap -p 80,443 --script http-waf-detect,http-waf-fingerprint  
www.tryhackme.com
```


- A **web application firewall (WAF)** is a device or a piece of software that checks packages sent to a web server in order to identify and block those that might be malicious, usually based on signatures or regular expressions.
- We can end up dealing with a lot of problems in our penetration test if an undetected WAF blocks our requests or bans our IP address. When performing a penetration test, the reconnaissance phase must include the detection and identification of a WAF, **intrusion detection system (IDS)**, or **intrusion prevention system (IPS)**. This is required in order to take the necessary measures to prevent being **blocked** or **banned**.
- we will use different methods, along with the tools included in Kali Linux, to detect and identify the presence of a web application firewall between our target and us.

Food for thought

- What is the difference between detect waf and fingerprint waf scripts?

Mapping the network

Further information gathering tools can be used such as threat assessment with **Maltego** or mapping the network with **casefile**. Explore those tools and use resources from the internet to try to collect all the information and tools you have gathered or used using those software. Other software also are pre installed in Kali Linux.

Determining network range

With the gathered information obtained by the previous tool, we can now focus on determining the IP addresses range from the target network. in this section, we will explore dmitry, at tool that allows to determine the network range.

- Let's begin the process of determining the network range by opening a terminal window.

commands for determining network range

- On a local network - kali is set as host only adapter

```
sudo dmitry -wnspb 192.168.56.102 -o ~/Desktop/dmitry-result
```
- You can also use dmitry on a website - You will need to be connected to the internet on kali.

```
sudo dmitry -wnspb cwsenario.site -o ~/Desktop/dmitry-result
```

Reconnaissance

Information from simply looking at code

- Looking into a web page's source code allows us to understand some of the programming logic, detect the obvious vulnerabilities, and also have a reference when testing, as we will be able to compare the code before and after a test and use that comparison to modify our next attempt.
- Lets look at the source code of an application and arrive at some conclusions from that. We will look specifically at an application called WackoPicko.
- WackoPicko has been developed as a *real* web application with following features:³

- **Authentication:** WackoPicko provides personalized content to registered users.
 - **Upload Pictures:** When a photo is uploaded to WackoPicko by a registered user, other users can comment on it, as well as purchase the right to a high-quality version.
 - **Comments on Pictures:** Once a picture is uploaded into WackoPicko, all registered users can comment on the photo by filling a form
 - **Purchase Pictures:** A registered user on WackoPicko can purchase the high-quality version of a picture.
 - **Search:** The search feature offers the possibility to filter pictures by looking for strings in the tags of the images
 - **Guestbook:** A guestbook page provides a way to receive feedback from all visitors to the WackoPicko web site.
 - **Admin Area:** WackoPicko has a special area for administrators only, which enables the creation of new users.
- With the source code we can discover the libraries or external files that the page is using and where the links go. Also, as can be seen in the preceding image, this page has some hidden input fields. The selected one is **MAX_FILE_SIZE**;
 - this means that, when we are uploading a file, this field determines the maximum size allowed for the file we are uploading. So, if we alter this value, we might be able to upload a file bigger than what is expected by the application; this represents an important security issue.

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

³<https://www.aldeid.com/wiki/WackoPicko>

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://192.168.56.102/vicnum/`
- Now we add `robots.txt` to the URL.
- This file tells search engines that the indexing of the directories `jotto` and `cgi-bin` is not allowed for every browser (user agent). However, this doesn't mean that we cannot browse them.
- Let's browse to `http://192.168.56.102/vicnum/cgi-bin/`
- We can click and navigate directly to any of the Perl scripts in this directory.
- Let's browse to `http://192.168.56.102/vicnum/jotto/`
- Click on the file named `jotto`. Jotto is a game about guessing five-character words; could this be the list of possible answers? Check it by playing the game; if it is, we have already hacked the game!

How these tools works

`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).

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.

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

- cgi-bin
- robots.txt
- phpmyadmin
- admin
- login
- save the file
- Navigate to **Kali Linux** \Rightarrow **Web application analysis** \Rightarrow **Web Crawlers & Directory Bruteforce** \Rightarrow **dirbuster**.
- On the DirBuster's window, set the target URL to **http://192.168.56.102:80**
- 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: **cgi-bin** 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.

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.

Web 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.