Information Gathering and Vulnerability Scanning

Introduction

You will briefly learn about some of the common tools and techniques used. From there, the module digs deeper into the process of vulnerability scanning and how scanning tools work, including how to analyze vulnerability scanner results to provide useful deliverables and explore the process of leveraging the gathered information in the exploitation phase. The module concludes with coverage of some of the common challenges to consider when performing vulnerability scans.

Performing Passive Reconnaissance

Reconnaissance :

Reconnaissance is always the initial step in a cyber attack. An attacker must first gather information about the target in order to be successful. In fact, the term reconnaissance is widely used in the military world to describe the gathering of information about the enemy, such as information about the enemy's location, capabilities, and movements. This type of information is needed to successfully perform an attack. Reconnaissance in a penetration testing engagement typically consists of scanning and enumeration. But what does reconnaissance look like from an attacker's perspective?

> *If you know the enemy and know yourself, you need not fear the result of a hundred battles. If you know yourself but not the enemy, for every victory gained you will also suffer a defeat. If you know neither the enemy nor yourself, you will succumb in every battle."*
>
> **— Sun Tzu, The Art of War**

Active Reconnaissance vs. Passive Reconnaissance

Active reconnaissance is a method of information gathering in which the tools used actually send out probes to the target network or systems in order to elicit responses that are then used to determine the posture of the network or system. These probes can use various protocols and multiple levels of aggressiveness, typically based on what is being scanned and when. For example, you might be scanning a device such as a printer that does not have a very robust TCP/IP stack or network hardware. By sending active probes, you might crash such a device. Most modern devices do not have this problem; however, it is possible, so when doing active scanning, you should be conscious of this and adjust your scanner settings accordingly.

Common active reconnaissance tools and methods include the following:

- **Host enumeration**
- **Network enumeration**
- **User enumeration**
- **Group enumeration**
- **Network share enumeration**
- **Web page enumeration**
- **Application enumeration**
- **Service enumeration**
- **Packet crafting**

Passive reconnaissance is a method of information gathering in which the tools do not interact directly with the target device or network. There are multiple methods of passive reconnaissance. Some involve using third-party databases to gather information. Others also use tools in such a way that they will not be detected by the target. These tools, in particular, work by simply listening to the traffic on the network and using intelligence to deduce information about the device communication on the network. This approach is much less invasive on a network, and it is highly unlikely for this type of reconnaissance to crash a system such as a printer. Because it does not produce any traffic, it is also unlikely to be detected and does not raise any flags on the network that it is surveying. Another scenario in which a passive scanner would come in handy would be for a penetration tester who needs to perform analysis on a production network that cannot be disrupted. The passive reconnaissance technique that you use depends on the type of information that you wish to obtain. One of the most important aspects of learning about penetration testing is developing a good methodology that will help you select the appropriate tools and technologies to use during the engagement.

Common passive reconnaissance tools and methods include the following:

- **Domain enumeration**
- **Packet inspection**
- **Open-source intelligence (OSINT)**
- **Recon-ng**
- **Eavesdropping**

The objectives of OSINT are:

- To determine the digital footprint of the organization.

- Determine what data about the organization is available to cyber criminals.

Access the OSINT Framework

- Go to the OSINT Framework site at https://osintframework.com/
- **WhatsMyName(T)**
- Go to https://smart.myosint.training/.

SpiderFoot

SpiderFoot : Designed to automate the process of gathering information from various sources about a given target, which could be an IP address, domain name, hostname, network subnet, or even a person's name or email address. It's widely used in cybersecurity, penetration testing, and digital forensics for the purpose of reconnaissance.

SpiderFoot seeds its scan with one of the following:

- **Domain names**
- **IP addresses**
- **Subnet addresses**
- **Autonomous System Numbers (ASN)**
- **Email addresses**
- **Phone numbers**
- **Personal names**

BuiltWith

API-Key: 0ddcd802-1eed-4bfa-b5c9-48b3912bb589

Hunter.io

API-Key: a654e7f9676a06bfc60403041021a73c813ab387

IntelligenceX

API-Key: 1e23462a-66c8-45b7-ae29-af428308ba05

Onion.link

API-Key: AIzaSyDwygm5Kta8x4L-HxCHPYdPLsjK1-Ud8kU

Investigate recon-ng

Recon-ng is an **OSINT framework** that is similar to the **Metasploit** exploitation framework or the **Social-Engineering Toolkit** (SET). It consists of a series of modules that can be run in their own workspaces.
The modules can be configured to run with option settings that are specific to the module. This simplifies running Recon-ng at the command line because options for the modules are independently set within the workspace. When you run the module, it uses these settings to perform its searches.

```
┌──(kali㉿Kali)-[~]
└─$ recon-ng
[recon-ng][default] > workspaces help
[recon-ng][default] > workspaces list
[recon-ng][default] > workspaces test
[recon-ng][default] > workspaces load test
[recon-ng][test] > help
```

Investigate the module marketplace

```
[recon-ng][default] > modules search
[recon-ng][default] > marketplace search
[recon-ng][default] > marketplace info nmap
[recon-ng][default] > marketplace serach bing
[recon-ng][default] > marketplace search whois
```

Install a new module

```
[recon-ng][default] > marketplace install all
[recon-ng][default] > marketplace install recon/domains-hosts/hackertarget
```

Run the new modules

```
/** create workspaces serviceOne
[recon-ng][default] > workspaces create serviceOne

/** Loading module into workspace
[recon-ng][serviceOne] > modules load hackertarget
[recon-ng][serviceOne][hackertarget] > info
```

/** info about Module hackertarget

Name: HackerTarget Lookup
Author: Michael Henriksen (@michenriksen)

Version: 1.1
Description:
Uses the HackerTarget.com API to find host names. Updates the 'hosts' table with the results.

```
Options:
  Name    Current Value  Required  Description
  ------  -------------  --------  -----------
  SOURCE  default        yes       source of input (see 'info' for details)
```

Source Options:
default SELECT DISTINCT domain FROM domains WHERE domain IS NOT NULL
<string> string representing a single input
<path> path to a file containing a list of inputs
query <sql> database query returning one column of inputs

/** change SOURCE current value
[recon-ng][serviceOne][hackertarget] > **options set source hackxor.net**

...

```
Options:
  Name    Current Value  Required  Description
  ------  -------------  --------  -----------
  SOURCE  hackxor.net    yes       source of input (see 'info' for details)
```

...
[recon-ng][serviceOne][hackertarget] > **run**
[recon-ng][serviceOne][hackertarget] > **dashboard**
[recon-ng][serviceOne][hackertarget] > **show hosts**

Investigate the web interface

```
┌──(kali㉿Kali)-[~]
└─$ recon-web
```

Find Interesting Files with Recon-ng

Install another module

```
┌──(kali㉿Kali)-[~]
└─$ recon-ng
[recon-ng][default] > marketplace search
```

Install and load the plugin

```
[recon-ng][default] > marketplace search
[recon-ng][default] > marketplace install discovery/info_disclosure/interesti…
[recon-ng][default] > workspaces load serviceOne
[recon-ng][serviceOne] > modules load discovery/info_disclosure/interesti…
[recon-ng][serviceOne][interesting_files] > info
```

```
Options:
  Name      Current Value                                  Required  Description
  --------  -------------                                  --------  -----------
  CSV_FILE  /home/kali/.recon-ng/data/interesting_files_verify.csv  yes       custom
filename map
  DOWNLOAD  True                                           yes       download discovered files
  PORT      80                                             yes       request port
  PROTOCOL  http                                           yes       request protocol
  SOURCE    default
```

```
/** change SOURCE current value
[recon-ng][serviceOne][interesting_files] > options set source hackxor.net
```

```
Options:
  Name      Current Value                                  Required  Description
  --------  -------------                                  --------  -----------
  CSV_FILE  /home/kali/.recon-ng/data/interesting_files_verify.csv  yes       custom
filename map
  DOWNLOAD  True                                           yes       download discovered files
  PORT      80                                             yes       request port
  PROTOCOL  http                                           yes       request protocol
  SOURCE    hackxor.net
```

[recon-ng][serviceOne][**interesting_files**] > **run**
[recon-ng][serviceOne][**interesting_files**] > **dashboard**
[recon-ng][serviceOne][**interesting_files**] > **show hosts**

DNS Lookups

```
┌──(kali㉿Kali)-[~]
└─$ dnsrecon -d h4cker.org
```

You can use other basic DNS tools, such as the **nslookup**, **host**, and **dig** Linux commands, to perform name resolution and obtain additional information about a domain. Example 3-2 shows how the Dig tool is used to show the DNS resolution details for h4cker.org.

```
;; ANSWER SECTION:
h4cker.org.         2767    IN      A       185.199.108.153
h4cker.org.         2767    IN      A       185.199.111.153
h4cker.org.         2767    IN      A       185.199.109.153
h4cker.org.         2767    IN      A       185.199.110.153
```

The highlighted lines show the **IP addresses associated with h4cker.org**. Similarly, you can use the dig < domain > mx command to ***obtain the email servers*** **used by h4cker.org** (mail exchanger [MX] record)

```
┌──(kali㉿Kali)-[~]
└─$ dig h4cker.org mx
```

```
h4cker.org.        2302    IN      MX      10 alt1.gmr-smtp-in.l.google.com.
h4cker.org.        2302    IN      MX      5 gmr-smtp-in.l.google.com.
h4cker.org.        2302    IN      MX      40 alt4.gmr-smtp-in.l.google.com.
h4cker.org.        2302    IN      MX      30 alt3.gmr-smtp-in.l.google.com.
h4cker.org.        2302    IN      MX      20 alt2.gmr-smtp-in.l.google.com.
```

Identification of Technical and Administrative Contacts

```
┌──(kali㉿Kali)-[~]
└─$ whois tesla.com
```

Showing Technical and Administrative Email Contact

```
┌──(kali㉿Kali)-[~]
└─$ whois cisco.com | grep '@cisco.com'
```

```
Registrant Email: infosec@cisco.com
Admin Email: infosec@cisco.com
Tech Email: infosec@cisco.com
```

Various tools of passive reconnaissance listed in Omar Santos' GitHub repository at https://github.com/The-Art-of-Hacking/h4cker/tree/master/osint.

Use nslookup to Obtain Domain and IP Address Information

```
┌──(kali㉿Kali)-[~]
└─$ man nslookup
```

use set type=mx To find **mail server**

```
┌──(kali㉿Kali)-[~]
└─$ nslookup  -query=mx cisco.com
```

```
┌──(kali㉿Kali)-[~]
└─$ nslookup
> set type=mx
> h4cker.org
;; communications error to 10.0.2.3#53: timed out
Server:         10.0.2.3
Address:        10.0.2.3#53

Non-authoritative answer:
h4cker.org      mail exchanger = 10 alt1.gmr-smtp-in.l.google.com.
h4cker.org      mail exchanger = 5 gmr-smtp-in.l.google.com.
h4cker.org      mail exchanger = 20 alt2.gmr-smtp-in.l.google.com.
h4cker.org      mail exchanger = 40 alt4.gmr-smtp-in.l.google.com.
h4cker.org      mail exchanger = 30 alt3.gmr-smtp-in.l.google.com.

Authoritative answers can be found from:
>
```

use set type=ns To find **Domain name servers configuration**

```
┌──(kali㉿Kali)-[~]
└─$ nslookup -query=ns cisco.com
```

Or

```
┌──(kali㉿Kali)-[~]
└─$ nslookup
      > set type=ns
      > cisco.com
```

ns

Find IPv4 and IPv6 Addresses of the server ns1

```
> set type=ns
> cisco.com
```

```
;; communications error to 192.168.1.1#53: timed out
Server:         192.168.1.1
Address:        192.168.1.1#53

Non-authoritative answer:
cisco.com       nameserver = ns1.cisco.com.
cisco.com       nameserver = ns3.cisco.com.
cisco.com       nameserver = ns2.cisco.com.

using a in type to return IPV4

        Find IPv4
        ┌──(kali㉿Kali)-[~]
        └─$ nslookup
                > set type=a
                > ns1.cisco.com
        or
        ┌──(kali㉿Kali)-[~]
        └─$ nslookup -type=a ns1.cisco.com



        Find IPv6
        ┌──(kali㉿Kali)-[~]
        └─$ nslookup
                > set type=aaaa
                > ns1.cisco.com

        ┌──(kali㉿Kali)-[~]
        └─$ nslookup -type=aaaa ns1.cisco.com
```

Change the server used to perform lookups

Occasionally it is desirable to use a different DNS server to perform lookups. This may be necessary if the local DNS server is unable to resolve an address or resolves the host name to an internal private address and you need to obtain the internet accessible address of the host.

The any query type can retrieve much, or all, of the information contained in the DNS record for a host name. Often text records that can provide additional details about the domain are contained in **DNS records**. Using the **8.8.8.8 Google DNS server**, find the **DNS records for skillsforall.com.**

```
┌──(kali㉿Kali)-[~]
└─$ nslookup
        > server 8.8.8.8
         > set type=any
         > skillsforall.com
```

Imagine you're trying to resolve the domain name example.com to its IP address. Normally, your computer uses the DNS server provided by your Internet Service Provider (ISP) to resolve domain names to IP addresses. However, there might be situations where you want to use a different DNS server. Here's why and how you might do it:

Why Use a Different DNS Server?

1. Faster Response Times: Some DNS servers might respond faster than others, improving your internet browsing experience.
2. Bypassing Restrictions: The local DNS might resolve certain domain names to incorrect IP addresses due to censorship or network policies, preventing access to those sites.
3. Testing and Troubleshooting: You might want to test how your website is resolved from different parts of the world, or you're troubleshooting DNS issues.
4. Privacy Concerns: Some users prefer DNS servers that do not log queries or that offer additional privacy features.

Example Scenario

You suspect that your ISP's DNS server is resolving example.com to an outdated IP address, or perhaps it's not resolving the domain at all because of some temporary issues. To get around this problem, you decide to use Google's public DNS server (8.8.8.8) to resolve the domain name.

How to Use a Different DNS Server for Lookup

Using the `nslookup` command-line tool, you can specify which DNS server to use for your query. Here's how you would do it:

```bash
nslookup example.com 8.8.8.8
```

This command tells **`nslookup`** to resolve example.com using the DNS server at 8.8.8.8 (Google's DNS).

What Happens in the Background

5. You Issue the Command: By running the command above, you're instructing your computer to bypass the local DNS settings.
6. Query Sent to Specified DNS Server: Your computer sends a DNS query directly to 8.8.8.8, asking for the IP address associated with example.com.
7. Response from the DNS Server: Google's DNS server looks up its records for example.com and returns the corresponding IP address to your computer.
8. Access the Website: Your computer uses this IP address to connect to example.com, potentially bypassing any issues present with your local DNS server's resolution.

Conclusion

By manually specifying a DNS server for your queries, you can overcome local DNS issues, access a more up-to-date or accurate resolution of domain names, and enjoy a more flexible and controlled browsing experience. This method is particularly useful for web developers, network administrators, and users in environments with restrictive DNS policies.

Use the whois function to obtain domain information

The whois tool queries domain registration information, rather than the **DNS server records**. It is another form of **passive reconnaissance** that can identify where the domain is registered, **technical** and **administrative contact information**, and physical locations. Be aware that information contained in domain registrations can be set to private and often the contact information is that of the hosting service, rather than the organization itself.

Compare whois output for various organizations

```
┌──(kali㉿Kali)-[~]
└─$ whois cisco.com
```

Use whois to determine IP address registration information

c. Because organizations may use the same IP networks for other externally facing servers, knowing the address ranges is valuable for determining which networks to target during a penetration test. Use the whois tool to obtain the IP address allocations for the IP networks where the other Cisco DNS servers are located.

Compare the output of the nslookup and dig Functions

Dig is a Linux function that performs DNS queries. The format of a Dig query is similar to that of Nslookup. To resolve the hostname cisco.com to an IP address, use the syntax dig [hostname].

```
┌──(kali㉿Kali)-[~]
└─$ dig cisco.com

Dig queries only the A record type  Only in IPv4
Nslookup queries both the A and AAAA records. IPv4 and IPv6
```

```
┌──(kali㉿Kali)-[~]
└─$ dig cisco.com AAAA

┌──(kali㉿Kali)-[~]
└─$ dig cisco.com 8.8.8.8 ns

Using dig and nslookup with any

┌──(kali㉿Kali)-[~]
└─$ dig skillsforall.com  any
┌──(kali㉿Kali)-[~]
└─$ nslookup -type=any skillsforall.com
```

Perform Reverse DNS lookups

Now that you can perform DNS lookups and use Whois to determine IP address ranges, use Dig to find additional host names. Reverse DNS (rDNS) lookups use the IP address to query for the host names of the services that resolve to that address.

Using dig to perform rDNS Lookups

using **-x** option to retrieve the hostname and record type of ns1.cisco.com DNS server(72.163.5.201)

```
┌──(kali㉿Kali)-[~]
└─$ dig  -x 72.163.5.201
```

The **PTR** (Pointer) record is used to map an IP address to a domain name.

Use the host utility to perform rDNS Lookups

The Host utility is a function in Linux that performs lookups to convert IP addresses to host names. Use this utility to find another host on the 72.163.0.0/16 network

```
┌──(kali㉿Kali)-[~]
└─$ host  72.163.10.1
```

Host can also be used to perform a quick IP address lookup for a known hostname

```
┌──(kali㉿Kali)-[~]
└─$ host  hsrp-72-163-10-1.cisco.com
```

URIs often contain aliases for the host name of the server hosting the website. the output of the host command can list the servers that respond to that URL

The information about **aliases** is useful when trying to determine where the actual website or service is located.

Use nslookup to perform rDNS Lookups

Nslookup is used primarily to perform IP address lookups for known host names. It can also be used to perform rDNS lookups to return a host name assigned to a known IP address.

Use Nslookup to find hostnames associated with an IP address.

In non-interactive mode the syntax to do an rDNS query is nslookup [ip address].

```
┌──(kali㉿Kali)-[~]
└─$ nslookup 72.163.5.201
```

To use interactive mode

```
┌──(kali㉿Kali)-[~]
└─$ nslookup
> 72.163.5.201
```

Cloud vs. Self-Hosted Applications and related Subdomains

A company can own a **domain** and related **subdomain**, but its applications might be hosted in the cloud . For example, Netflix (at the time of writing) owns the domain **netflix.com**, Which resolves to **IPv4 addresses** 3.230.129.93, 52.3.144.142, and 54.237.226.164 (as demonstrated in Example below with Linux host command)

DNS Name Resolution for netflix.com

```
┌──(kali㉿Kali)-[~]
└─$ host  netflix.com

netflix.com has address 54.73.148.110
netflix.com has address 54.155.246.232
netflix.com has address 18.200.8.190
netflix.com has IPv6 address 2a05:d018:76c:b684:8ab7:ac02:667b:e863
netflix.com has IPv6 address 2a05:d018:76c:b683:a2cd:4240:8669:6d4
netflix.com has IPv6 address 2a05:d018:76c:b685:e8ab:afd3:af51:3aed
netflix.com mail is handled by 1 aspmx.l.google.com.
netflix.com mail is handled by 5 alt2.aspmx.l.google.com.
netflix.com mail is handled by 10 aspmx2.googlemail.com.
netflix.com mail is handled by 5 alt1.aspmx.l.google.com.
netflix.com mail is handled by 10 aspmx3.googlemail.com.
```

Ownership of IP Addresses and Applications Hosted in the Cloud

```
┌──(kali㉿Kali)-[~]
└─$ whois  54.73.148.110 | grep OrgName
    OrgName:        Amazon Technologies Inc.
    OrgName:        Amazon.com, Inc.
```

Social Media Scraping

Attackers can easily gather valuable information about victims by scraping social media sites such as Twitter, LinkedIn, Facebook, and Instagram. People

post too many things online. They publicly talk about their hobbies, the restaurants they visit, what they do for work, work promotions, where they travel for business and pleasure, and much more.

Often attackers use other information such as key contacts (company stakeholders) and their job responsibilities. Attackers can use all that information to perform different types of **social engineering attacks** (**including spear phishing and whaling**). You will learn details about social engineering attacks in Module 4, “Social Engineering Attacks.”

Attackers often leverage job listings in websites like Indeed, LinkedIn, CareerBuilder, and individual company websites to obtain information about the technologies these companies use (for example, the technology stack of an organization). Let’s say a company is looking for a Cisco firewall administrator, an Ansible expert, and a Mongo database architect in Raleigh, North Carolina. The attacker didn’t have to launch any tools to learn that the company is using Cisco firewalls, MongoDB, and Ansible for automation in its Raleigh office.

Similarly, attackers have also created job posts to attract people to apply for those positions. Then they interview their victims to try to get them to talk about what they do at work and the technologies used by their employer. Think about it: When people are trying to get a job, they want to “show off” and often reveal too much information about the technologies, architectures, and applications that they use in their current roles.

Employee Intelligence Gathering

Gather Information Through Social Media

Step 1 : Conduct the investigation
Conduct a search for your name and usernames, using different search engines. **Open an incognito or private window** for this lab. This will prevent any of your saved login information from populating and you

should not see any cached content as you search. Use variations of your name as well, with and without a middle name or initial, married, and maiden names, etc.

Open an incognito:
Windows, Linux, or Chrome OS: Press Ctrl + Shift + n. Mac: Press ⌘ + Shift + n.

Conduct an audit of all the social media sites and accounts you have used. Look for identifiable information and behavioral patterns that could be helpful to an attacker. This includes things like your **place of employment**, **where you live**, **online shopping preferences**, your **daily schedule**, **vacation plans**, **political beliefs**, **interests**, **hobbies**, **education**, **cultural beliefs**, **family members**, **pets**, etc.

Items to investigate include:

1- All your social media profiles - name, birthdate, contact information, etc.

2- Your status updates - life events, work relationships and status, political and religious beliefs.

3- Location data - hometown information and geo check-ins.

4- Shared content - pictures and comments you have posted and those where you are mentioned or tagged.

5- Posts from friends and family.

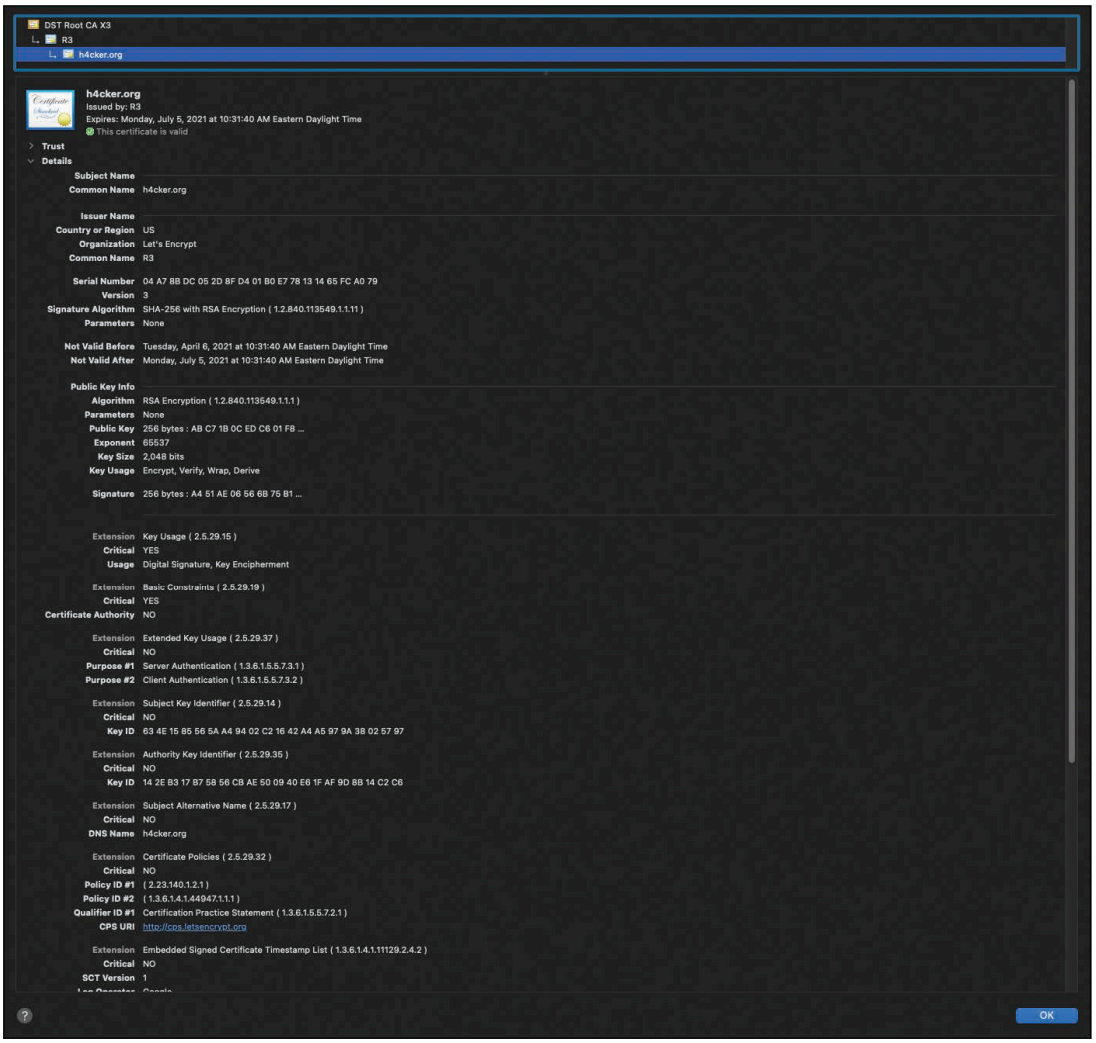
6- Any online discussions you have joined or participated in.

Cryptographic Flaws

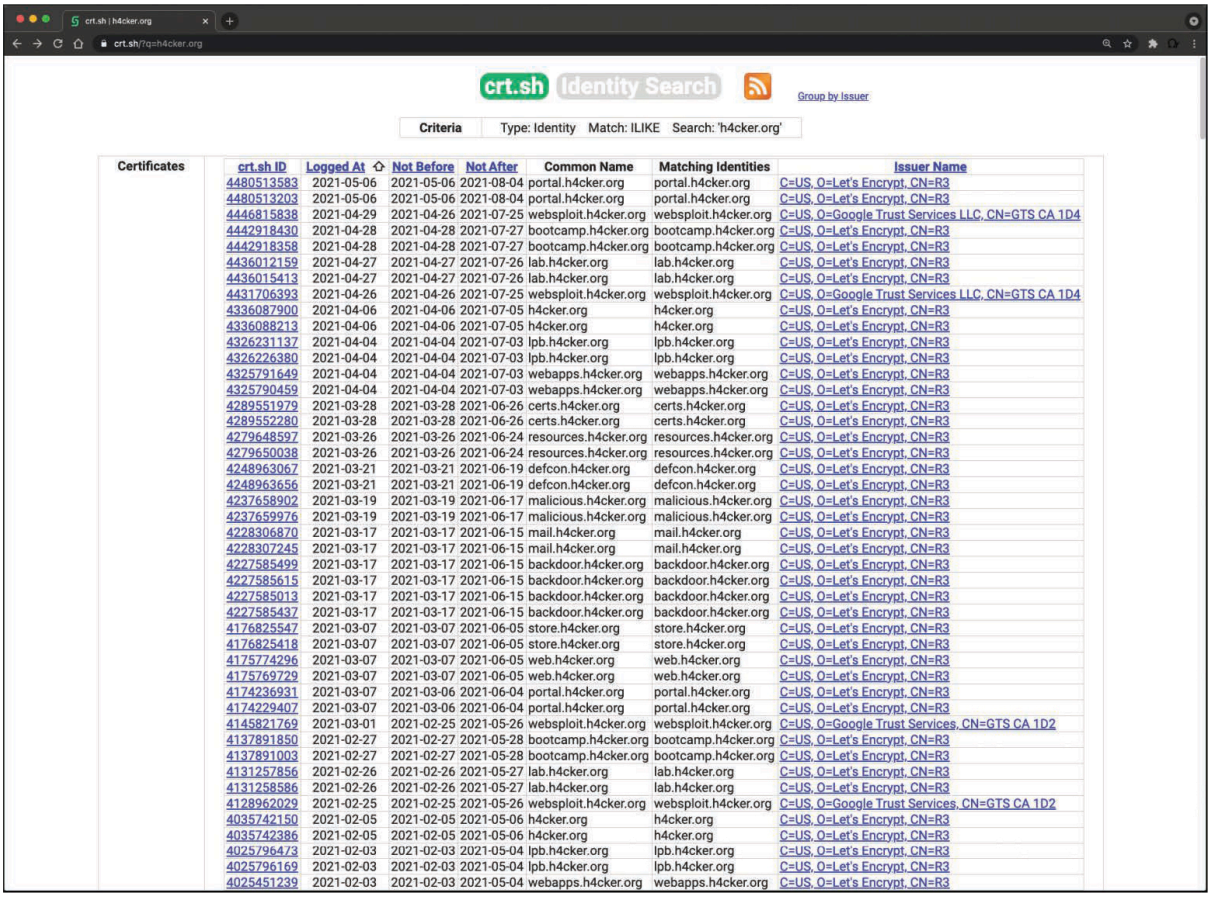
During the reconnaissance phase, attackers often can inspect **Secure Sockets Layer** (**SSL**) certificates to obtain information about the organization, potential cryptographic flaws, and weak implementations. You can find a lot inside digital certificates: the certificate serial number, the subject common name, the uniform **resource identifier (URI)** of the server it was assigned to, the organization name, **Online Certificate Status Protocol (OCSP)** information, the **certificate revocation list (CRL)** URI, and so on.

Obtain detailed information about certificate transparency from https://certificate.transparency.dev.

The Digital Certificate Assigned to h4cker.org



Tools such as **crt.sh** enable you to obtain detailed certificate transparency information about any given domain. Figure below shows the result of the query **https://crt.sh/?q=h4cker.org** in crt.sh for the domain **h4cker.org. we can see in the search results multiple subdomains that were known to the attacker before.**



crt.sh ID	Logged At	Not Before	Not After	Common Name	Matching Identities	Issuer Name
4480513583	2021-05-06	2021-05-06	2021-08-04	portal.h4cker.org	portal.h4cker.org	C=US, O=Let's Encrypt, CN=R3
4480513203	2021-05-06	2021-05-06	2021-08-04	portal.h4cker.org	portal.h4cker.org	C=US, O=Let's Encrypt, CN=R3
4446815838	2021-04-29	2021-04-26	2021-07-25	websploit.h4cker.org	websploit.h4cker.org	C=US, O=Google Trust Services LLC, CN=GTS CA 1D4
4442918430	2021-04-28	2021-04-28	2021-07-27	bootcamp.h4cker.org	bootcamp.h4cker.org	C=US, O=Let's Encrypt, CN=R3
4442918358	2021-04-28	2021-04-28	2021-07-27	bootcamp.h4cker.org	bootcamp.h4cker.org	C=US, O=Let's Encrypt, CN=R3
4436012159	2021-04-27	2021-04-27	2021-07-26	lab.h4cker.org	lab.h4cker.org	C=US, O=Let's Encrypt, CN=R3
4436015413	2021-04-27	2021-04-27	2021-07-26	lab.h4cker.org	lab.h4cker.org	C=US, O=Let's Encrypt, CN=R3
4431706393	2021-04-26	2021-04-26	2021-07-25	websploit.h4cker.org	websploit.h4cker.org	C=US, O=Google Trust Services LLC, CN=GTS CA 1D4
4336087900	2021-04-06	2021-04-06	2021-07-05	h4cker.org	h4cker.org	C=US, O=Let's Encrypt, CN=R3
4336088213	2021-04-06	2021-04-06	2021-07-05	h4cker.org	h4cker.org	C=US, O=Let's Encrypt, CN=R3
4326231137	2021-04-04	2021-04-04	2021-07-03	lpb.h4cker.org	lpb.h4cker.org	C=US, O=Let's Encrypt, CN=R3
4326226380	2021-04-04	2021-04-04	2021-07-03	lpb.h4cker.org	lpb.h4cker.org	C=US, O=Let's Encrypt, CN=R3
4325791649	2021-04-04	2021-04-04	2021-07-03	webapps.h4cker.org	webapps.h4cker.org	C=US, O=Let's Encrypt, CN=R3
4325790459	2021-04-04	2021-04-04	2021-07-03	webapps.h4cker.org	webapps.h4cker.org	C=US, O=Let's Encrypt, CN=R3
4289551979	2021-03-28	2021-03-28	2021-06-26	certs.h4cker.org	certs.h4cker.org	C=US, O=Let's Encrypt, CN=R3
4289552280	2021-03-28	2021-03-28	2021-06-26	certs.h4cker.org	certs.h4cker.org	C=US, O=Let's Encrypt, CN=R3
4279648597	2021-03-26	2021-03-26	2021-06-24	resources.h4cker.org	resources.h4cker.org	C=US, O=Let's Encrypt, CN=R3
4279650038	2021-03-26	2021-03-26	2021-06-24	resources.h4cker.org	resources.h4cker.org	C=US, O=Let's Encrypt, CN=R3
4248963067	2021-03-21	2021-03-21	2021-06-19	defcon.h4cker.org	defcon.h4cker.org	C=US, O=Let's Encrypt, CN=R3
4248963656	2021-03-21	2021-03-21	2021-06-19	defcon.h4cker.org	defcon.h4cker.org	C=US, O=Let's Encrypt, CN=R3
4237658902	2021-03-19	2021-03-19	2021-06-17	malicious.h4cker.org	malicious.h4cker.org	C=US, O=Let's Encrypt, CN=R3
4237659976	2021-03-19	2021-03-19	2021-06-17	malicious.h4cker.org	malicious.h4cker.org	C=US, O=Let's Encrypt, CN=R3
4228306870	2021-03-17	2021-03-17	2021-06-15	mail.h4cker.org	mail.h4cker.org	C=US, O=Let's Encrypt, CN=R3
4228307245	2021-03-17	2021-03-17	2021-06-15	mail.h4cker.org	mail.h4cker.org	C=US, O=Let's Encrypt, CN=R3
4227585499	2021-03-17	2021-03-17	2021-06-15	backdoor.h4cker.org	backdoor.h4cker.org	C=US, O=Let's Encrypt, CN=R3
4227585615	2021-03-17	2021-03-17	2021-06-15	backdoor.h4cker.org	backdoor.h4cker.org	C=US, O=Let's Encrypt, CN=R3
4227585013	2021-03-17	2021-03-17	2021-06-15	backdoor.h4cker.org	backdoor.h4cker.org	C=US, O=Let's Encrypt, CN=R3
4227585437	2021-03-17	2021-03-17	2021-06-15	backdoor.h4cker.org	backdoor.h4cker.org	C=US, O=Let's Encrypt, CN=R3
4176825547	2021-03-07	2021-03-07	2021-06-05	store.h4cker.org	store.h4cker.org	C=US, O=Let's Encrypt, CN=R3
4176825418	2021-03-07	2021-03-07	2021-06-05	store.h4cker.org	store.h4cker.org	C=US, O=Let's Encrypt, CN=R3
4175774296	2021-03-07	2021-03-07	2021-06-05	web.h4cker.org	web.h4cker.org	C=US, O=Let's Encrypt, CN=R3
4175769729	2021-03-07	2021-03-07	2021-06-05	web.h4cker.org	web.h4cker.org	C=US, O=Let's Encrypt, CN=R3
4174236931	2021-03-07	2021-03-06	2021-06-04	portal.h4cker.org	portal.h4cker.org	C=US, O=Let's Encrypt, CN=R3
4174229407	2021-03-07	2021-03-06	2021-06-04	portal.h4cker.org	portal.h4cker.org	C=US, O=Let's Encrypt, CN=R3
4145821769	2021-03-01	2021-02-25	2021-05-26	websploit.h4cker.org	websploit.h4cker.org	C=US, O=Google Trust Services, CN=GTS CA 1D2
4137891850	2021-02-27	2021-02-27	2021-05-28	bootcamp.h4cker.org	bootcamp.h4cker.org	C=US, O=Let's Encrypt, CN=R3
4137891003	2021-02-27	2021-02-27	2021-05-28	bootcamp.h4cker.org	bootcamp.h4cker.org	C=US, O=Let's Encrypt, CN=R3
4131257856	2021-02-26	2021-02-26	2021-05-27	lab.h4cker.org	lab.h4cker.org	C=US, O=Let's Encrypt, CN=R3
4131258586	2021-02-26	2021-02-26	2021-05-27	lab.h4cker.org	lab.h4cker.org	C=US, O=Let's Encrypt, CN=R3
4128962029	2021-02-25	2021-02-25	2021-05-26	websploit.h4cker.org	websploit.h4cker.org	C=US, O=Google Trust Services, CN=GTS CA 1D2
4035742150	2021-02-05	2021-02-05	2021-05-06	h4cker.org	h4cker.org	C=US, O=Let's Encrypt, CN=R3
4035742386	2021-02-05	2021-02-05	2021-05-06	h4cker.org	h4cker.org	C=US, O=Let's Encrypt, CN=R3
4025796473	2021-02-03	2021-02-03	2021-05-04	lpb.h4cker.org	lpb.h4cker.org	C=US, O=Let's Encrypt, CN=R3
4025796169	2021-02-03	2021-02-03	2021-05-04	lpb.h4cker.org	lpb.h4cker.org	C=US, O=Let's Encrypt, CN=R3
4025451239	2021-02-03	2021-02-03	2021-05-04	webapps.h4cker.org	webapps.h4cker.org	C=US, O=Let's Encrypt, CN=R3

Finding Information from SSL Certificates

Objectives :

1 . View certificate information on Hosts
2 . Access Detailed certificate Information
3 . Use SSL Analysis Tools in kali
4 . Use Kali Tools to Gather Certificate Information

SSL/TLS certificates provide two broad functions.

First, they provide a way that the ownership of a website can be validated by people who are accessing it.

Second, they provide a means by which communication between a client and server is encrypted so that it cannot be read or altered by unauthorized parties.

They also provide the information required for a browser to create a secure, encrypted connection to a web site over the HTTPS protocol. Certificates are used behind the scenes as users browse the internet. In most cases, users are not aware that they are in use. The users become aware of them if a certificate is missing, out of date, or misconfigured.

The format of public key certificate information is specified by the X.509 standard

In addition, **certain versions of software, such as OpenSSL**, have widely known vulnerabilities that can be exploited, including vulnerability to the heartbleed bug. In addition, it is possible that some certificates could use weak encryption algorithms.

1. View certificate information on Hosts

Some SSL certificates are stored locally on network hosts. These certificates allow secure communication between a host and a server through a certificate chain. A host stores intermediate and root certificates as part of the SSL authentication process.

Step 1: View site Certificates from a browser.

a. Navigate to skillsforall.com

b. **In most browsers, a padlock icon appears next to the URL of the site that is currently displayed. Click the padlock icon and explore the settings available.**

step 2: View stored certificates in the operating system

a. **Microsoft Windows has a security management application that is part of the Microsoft Management Console. Enter certmgr.msc in the search box and press Enter to open it.**
b.
c. **In Kali, you can find the stored certificates in the /usr/share/ca-certificates/mozilla folder. Right-click a certificate and select Open With "ViewFile" to access the information for a certificate.**
d.
b. **Access information about trusted root and intermediate certificates in Windows by selecting the appropriate certificate folders in the management app.**
c.
d. **In Kali, access the certificates folder and use ls -l | grep root to list root certificate files, or search for the word root in the file manager window.**

2. Access Detailed Certificate Information Online

In **OSINT**, CT logs can be used to gather information about SSL/TLS certificates used by an organization or a specific domain. By analyzing CT logs, analysts can identify certificate issuances and their associated domains, as well as any anomalies or irregularities in certificate issuance. CT logs can also be used to monitor for any unauthorized SSL/TLS certificate issuance, which could indicate a potential security breach.

CT logs can be accessed through various CT log servers and APIs. There are also several CT monitoring tools available, such as **CertSpotter** and **Censys**, which can help automate the process of monitoring CT logs for specific domains or **SSL/TLS** certificates.

3. Use SSL Analysis Tools in Kali

Tool	Description	Recon, Exploitation, or Utility
sslscan	Queries SSL services to determine what cyphers are supported	**Reconnaissance**
ssldump	Analyze and decode SSL traffic	**Exploitation**
sslh	Running multiple services on port 443	**Utility**
sslsplit	Enable MitM attacks on SSL encrypted network connections	**Exploitation**
sslyze	Analyze the SSL configuration of a server by connecting to it	**Reconnaissance**

4. Use Kali Tools to Gather Certificate Information

As you know, sslscan is a Kali tool reconnaissance that will gather information about SSL certificates that are associated with domains. It is a command line utility. We will use sslscan to gather information about certificates and use another utility, called **aha**, to output the results to an HTML file.

Step 1: Install aha.

The application aha creates a standard HTML file that captures the output of terminal commands to standard HTML files. Aha captures any color coding and basic formatting of the command output. It also has command line options that allow you to specify your own formatting, such as background color, stylesheets to apply, and word wrap, among other settings.

Update apt package information with the apt update command.

```
┌──(kali㉿Kali)-[~]
└─$ sudo apt update
```

Install **aha** ,the **option -y** assumes yes is the answers to all prompts and can run non-interactively. In this case, you are giving permission to install aha.

```
┌──(kali㉿Kali)-[~]
└─$ sudo apt install -y aha
```

Stap 2 : Run sslscan and save the output to an HTML file

Execute the command to run **sslscan** .

```
┌──(kali㉿Kali)-[~]
└─$ sslscan skillsforall.com
```

While sslscan provides options for outputting results in text or XML file formats, the readability of HTML and the preservation of color coding is provided by aha. To use aha, pipe the output of the sslscan command to aha and then redirect the output of aha to a HTML file.

```
┌──(kali㉿Kali)-[~]
└─$ sslscan skillsforall.com | aha > sfa_cert.html
```

sslscan will save the file in the Kali Home directory as indicated by the prompt. **You can add a path to the filename or run the terminal from a destination directory to save it elsewhere.**

- **Red background text – NULL cipher. No encryption was used.**
- **Red – broken cipher (less than or equal to 40-bit), vulnerable or broken protocol such as SSLv2 or SSLv3 or broken certificate signing algorithm such as MD5.**
- **Yellow – weak cipher (less than or equal to 56-bit) or weak signing algorithm such as SHA-1.**
- **Purple – anonymous cipher such as ADH or AECDH.**

Execute the command to run **sslscan** .

```
┌──(kali㉿Kali)-[~]
└─$ sslscan skillsforall.com
```

Execute the command to run **sslyze** .

```
┌──(kali㉿Kali)-[~]
└─$ sslyze skillsforall.com
```

Execute the command to run **ssldump** .

```
┌──(kali㉿Kali)-[~]
└─$ sudo ssldump -i eth0 port 443
```

Execute the command to run **sslh**

Scénario:

1-Configuration **sslh**

```
┌──(kali㉿Kali)-[~]
```

```
└─$ sudo nano /etc/default/sslh/

# Run 'sslh' in foreground.
DAEMON_OPTS="--user sslh --listen 0.0.0.0:443 --ssh 127.0.0.1:22 --http
127.0.0.1:80 --pidfile /var/run/sslh/sslh.pid"

┌──(kali㉿Kali)-[~]
└─$ sudo systemctl restart sslh

Get information about domain :
┌──(kali㉿Kali)-[~]
└─$ whois  skillsforall.com

use port 22 or 443
┌──(kali㉿Kali)-[~]
└─$ sslh user@skillsforall.com  -p 22 or 443
```

Company Reputation and Security Posture

Security breaches can have a direct impact on a company's reputation. Attackers can leverage information from past security breaches that an organization might have experienced. They may, for example, leverage the following data while trying to gather information about their victims:

1. **Password dumps**
2. **File metadata**
3. **Strategic search engine analysis/enumeration**
4. **Website archiving/caching**
5. **Public source code repositories**

Password Dumps

Attackers can leverage password dumps from previous breaches. There are a number of ways that an attacker can get access to such password dumps, such as by using **Pastebin**, **dark web** websites, and even **GitHub** in some cases. Several different tools and websites make this task very easy. An example of a tool that allows you to **find email addresses and passwords** exposed in previous breaches is **h8mail**. You can install h8mail by using the **pip3 install h8mail** command.

Installing and Using **h8mail** .

```
┌──(kali㉿Kali)-[~]
└─$ pip3 install h8mail

┌──(kali㉿Kali)-[~]
└─$ h8mail -t  target@email.com
```

Search for **multiple emails** from a file

```
┌──(kali㉿Kali)-[~]
└─$ sudo  nano  targets.txt
```

```
┌──(kali㉿Kali)-[~]
└─$ cat   targets.txt | nl

1  target1@exampl.com
2  target1@exampl.com
3  target1@exampl.com

┌──(kali㉿Kali)-[~]
└─$ h8mail -t  targets.txt
```

Using configuration files for **API keys:**

To enhance the search , H8mail allows you to use services like HavelBeenPwned, Snusbase, or others that require API keys. specify the configuration file

```
┌──(kali㉿Kali)-[~]
└─$ h8mail -t  target@email.com -c config.ini
```

Advanced features

Custom search sources: You can configure **h8mail** to use additional or custom breach databases by modifying the configuration file.
Output: You can specify an output file to save your search results.

```
┌──(kali㉿Kali)-[~]
```

```
└─$ h8mail -t  target@email.com -o reslut.txt
```

```
┌──(kali㉿Kali)-[~/kali_folder2]
└─$ h8mail -g
```

```
┌──(kali㉿Kali)-[~/kali_folder2]
└─$ nano h8mail_config.ini
```

```
[h8mail]
; h8mail will automatically detect present keys & launch services accordingly
; Uncomment to activate
;hunterio =
snusbase_token = handpick-bugbear-panda
;;weleakinfo_priv =
;;weleakinfo_pub =
;hibp =
;leak-lookup_pub = 1bf94ff907f68d511de9a610a6ff9263
;leak-lookup_priv =
;emailrep =
;dehashed_email =
;dehashed_key =
;intelx_key =
;intelx_maxfile = 10
;breachdirectory_user =
;breachdirectory_pass =
```

```
┌──(kali㉿Kali)-[~/kali_folder2]
└─$ h8mail -t brahimlion38@gmail.com -sk -c h8mail_config.ini
```

```
┌──(kali㉿Kali)-[~/kali_folder2]
└─$ h8mail -t "2023_2024" -q password -sk -c h8mail_config.ini
```

Using hunter.io

h8mail_config.ini

```
[h8mail]
; h8mail will automatically detect present keys & launch services accordingly
; Uncomment to activate
;hunterio =a654e7f9676a06bfc60403041021a73c813ab387
;snusbase_token = handpick-bugbear-panda
;;weleakinfo_priv =
;;weleakinfo_pub =
;hibp =
;leak-lookup_pub = 1bf94ff907f68d511de9a610a6ff9263
;leak-lookup_priv =
;emailrep =
;dehashed_email =
;dehashed_key =
;intelx_key =
;intelx_maxfile = 10
;breachdirectory_user =
;breachdirectory_pass =
```

```
┌──(kali㉿Kali)-[~/kali_folder2]
└─$ h8mail -t "brahimlion38@gmail.com" -sk -c h8mail_config.ini -o results.csv
```

```
┌──(kali㉿Kali)-[~/kali_folder2]
└─$ ls
h8mail_config.ini  kali_folder3  results.csv  targetsEmail.txt

┌──(kali㉿Kali)-[~/kali_folder2]
└─$ h8mail -t results.csv -sk -c h8mail_config.ini
```

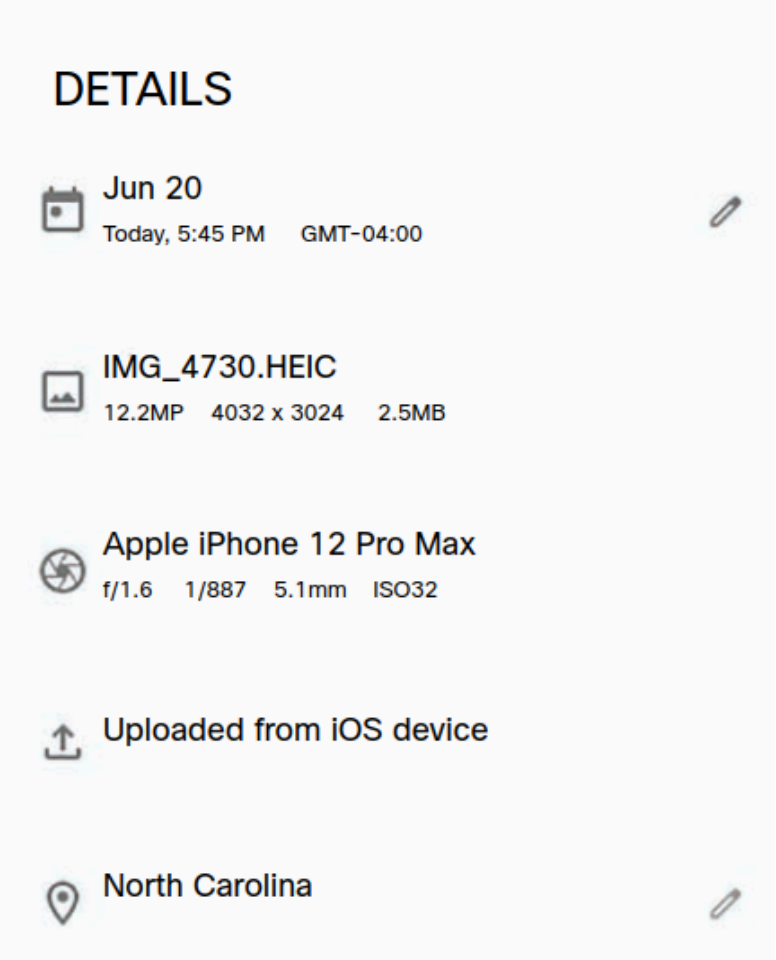
The following are additional tools that allow you to search for breach data dumps:

- WhatBreach: *https://github.com/Ekultek/WhatBreach*
- LeakLooker: *https://github.com/woj-ciech/LeakLooker*
- Buster: *https://github.com/sham00n/buster*
- Scavenger: *https://github.com/rndinfosecguy/Scavenger*
- PwnDB: *https://github.com/davidtavarez/pwndb*

Tools like **h8mail** and **WhatBreach** take advantage of breached data repositories of websites such as **haveibeenpwned.com** and **snusbase.com**. Historically, websites such as **weleakinfo.com** (seized by the **FBI**) have been used by criminals to dump information from past security breaches.

File Metadata

You can obtain a lot of information from metadata in files such as **images**, Microsoft **Word documents**, **Excel files**, **PowerPoint files**, and more. For instance, Exchangeable Image File Format (Exif) is a specification that defines the formats for images, sound, and supplementary tags used by digital **cameras**, **mobile phones**, **scanners**, and other systems that process image and sound files. Figure 3-3 shows the Exif data of a digital image (picture) captured by an iPhone.



Several tools can show Exif details. One of the most popular of them, **ExifTool**, is demonstrated in Example 3-10. This example shows the Exif metadata details of the same image whose details are shown in Figure 3-3.

```
Using  ExifTool .

        ┌──(kali㉿Kali)-[~]
        └─$ exiftool  IMG_4730.jpg


Get GPS Informations .

        ┌──(kali㉿Kali)-[~]
        └─$ exiftool  IMG_4730.jpg | grep  GPS


Verbose  Mode

        ┌──(kali㉿Kali)-[~]
        └─$ exiftool  -v IMG_4730.jpg
```

Strategic Search Engine Analysis/Enumeration

Most of us use search engines such as **DuckDuckGo**, **Bing**, and **Google** to locate information. What you might not know is that search engines, such as Google, can perform much more powerful searches than most people ever dream of. Google can translate documents, perform news searches, and do images searches. In addition, **hackers** and **attackers** can use it to do something that has been termed **Google hacking**.

By using basic search techniques combined with advanced operations, **both you and attackers can use Google as a powerful vulnerability search tool**. The following are some advanced operations:

Filetype: Directs Google to search only within the text of a particular type of file **(for example, filetype:xls)**

Inurl: Directs Google to search only within the specified URL of a document **(for example, inurl:serach-text)**

Link: Directs Google to search within hyperlinks for a specific term **(for example, link: www.domain.com)**

Intitle: Directs Google to search for a term within the title of a document **(for example, intitle, "index of /etc")**

By using these advanced operations in combination with key terms, both you and attackers can get Google to uncover many pieces of sensitive information that shouldn't be revealed. These search strings are often called **Google dorks**.

Demo Google dorks. :

Tosse how Google dorking works, enter the following into Google :
Use Case: use with search engines like Google to find web server log files
. intext:JSESSIONID OR intext:PHPSESSID inurl:access.log ext:log

Use Case: If you're looking for presentations about cybersecurity, you might use filetype:ppt cybersecurity to find PowerPoint presentations on the topic.
. filetype:ppt cybersecurity

Use Case: To find pages that have "profile" in their URL, possibly indicating individual profiles
. inurl:profile.

Use Case : You can use advanced operators to search for many types of data. The following is another example of a Google search string (or Google dork) that can reveal passwords of web applications:

```
"public $user =" | "public $password = " | "public $secret ="
| "public $db =" ext:txt | ext:log -git
```

https://www.exploit-db.com/google-hacking-database

Now that we have discussed some basic Google search techniques, let's look at advanced Google hacking. We recommend that you visit the Google Hacking Database (GHDB) repositories at *https://www.exploit-db.com/google-hacking-database/*. GHDB has the following search categories:

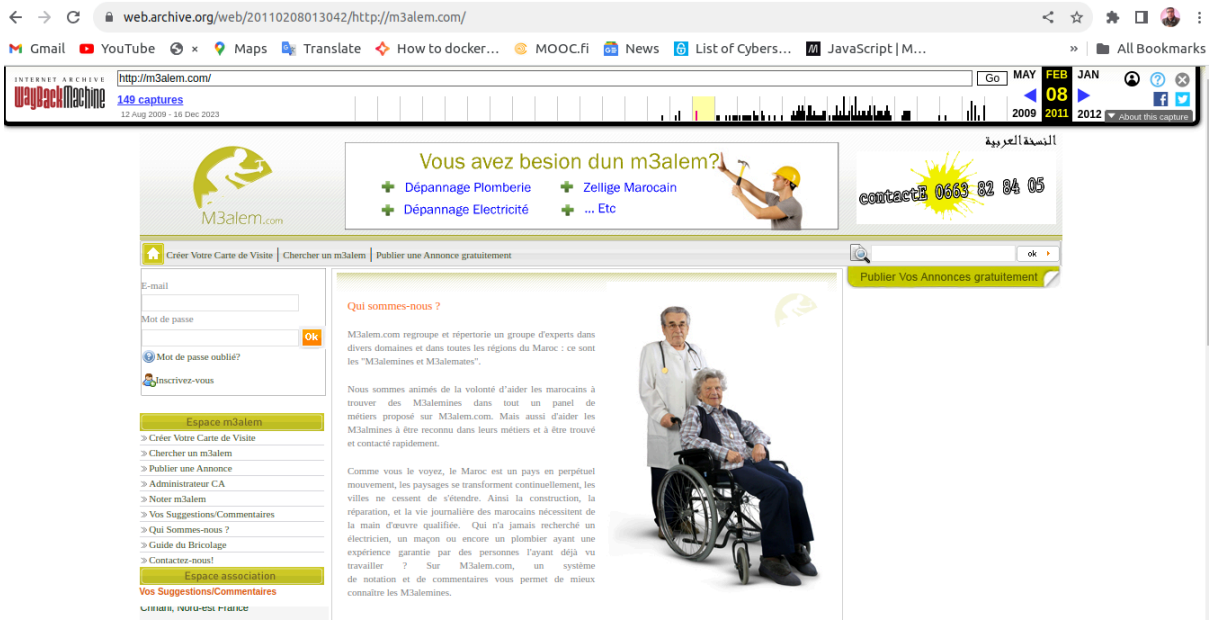
- Footholds
- Files containing usernames
- Sensitive directories
- Web server detection
- Vulnerable files
- Vulnerable servers
- Error messages
- Files containing juicy info
- Files containing passwords
- Sensitive online shopping info
- Network or vulnerability data
- Pages containing login portals
- Various online devices
- Advisories and vulnerabilities

GHDB is a community effort. Anyone can upload a new Google dork to perform these types of searches. Once you start playing with the dorks in GHDB, you will be surprised by the unbelievable things found through Google hacking. GHDB has made using Google dorks very easy, and there are other options as well. Later in this module, you will learn about additional tools that can be used to perform similar searches (such as Recon-ng).

Website Archiving/Caching

Several organizations archive and cache website data on the Internet. One of the most popular repositories is the “**Wayback Machine**” of Internet Archive (*https://archive.org/web*).

The Wayback Machine allows you to go back in time on the Internet. For example, Figure 3-4 shows what Cisco’s website looked like on November 3, 1999. You can access the archive of the site shown in Figure below



Public Source Code Repositories

An attacker can obtain extremely valuable information from public source code repositories such as GitHub and GitLab. Most of the applications and products we consume today use open-source software that is freely available in these public repositories. Attackers can find vulnerabilities in those software packages and use them to their advantage. Similarly, as a penetration tester, you can obtain valuable information from these public repositories. Even if you do not immediately find security vulnerabilities in the code, **these repositories can give you insights into the architecture and underlying code used in the organization’s applications and infrastructure.**

Find Information about Email Breaches

It is possible to learn more about a person or organization by searching on a known email address. It is useful to determine if employees of a company have had their work email addresses compromised. Several online services provide the ability to search on individual email addresses and entire domains to reveal breaches. Some of those sites are:

- haveibeenpwned.com
- f-secure.com
- hacknotice.com
- breachdirectory.com
- keepersecurity.com

Tool to find email addresses for a domain

Using a tool called **EmailHarvester** to find information about a domain, including email addresses of personnel.

Using **emailharvester** to find information about a domain including email .

```
┌──(kali㉿Kali)-[~]
└─$ emailharvester -h

┌──(kali㉿Kali)-[~]
└─$ emailharvester -d h4cker.org -s ~/folder_path/file_name

┌──(kali㉿Kali)-[~]
└─$ emailharvester -d h4cker.org -l 300 -e googles -s ~/kali_course/scanning/results
```

Search engines and platforms
(ASK, Reddit, Yahoo, Bing, Google, and Baidu)

The -x or --proxy option in commands like those used with EmailHarvester allows you to specify a proxy server through which all requests from the tool will be routed. Using a proxy can be beneficial for several reasons, including anonymity, bypassing IP-based rate limits or blocks, and accessing content restricted to certain geographical locations.

```
Using  emailharvester to find email  .

        ┌──(kali㉿Kali)-[~]
        └─$ emailharvester  -d example.com -l 100 -e bing -x http:// 127.0.0.1:8080
```

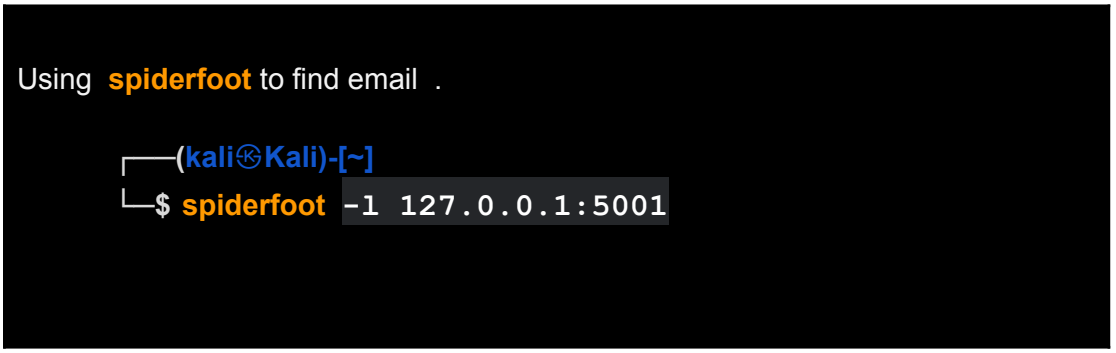
Anonymity: Routing your requests through a proxy can help mask your real IP address, making it more difficult for the target server to identify and block you based on your IP. This can be particularly useful when scanning for information to avoid leaving a direct trail back to your own machine.

Bypassing Restrictions: If the server has rate-limited or blocked your IP address, using a proxy can allow you to continue your work unimpeded. Additionally, proxies can help you access content or services that are geographically restricted by making requests appear as if they originate from a different location.

Testing: Proxies can be used to test how systems behave from different network environments, simulating accesses from various locations around the world.

Use spiderfoot to research email addresses

SpiderFoot is designed to automate the process of **gathering information from various sources about a given target,** which could be an **IP address**, **domain name**, **hostname**, **network subnet,** or even a **person's name** or **email address**. It's widely used in cybersecurity, penetration testing, and digital forensics for the purpose of reconnaissance.



Ahmia
Purpose: Searches the Ahmia search engine, which indexes hidden services on the Tor network. While not directly related to email scans, it can be useful for uncovering hidden services that may be tied to an email address.

AccountFinder
Purpose: Attempts to find social media accounts and other online services associated with an email address. This is directly relevant to your goal of identifying accounts registered to an email address on social media sites.

Archive.org (The Wayback Machine)
Purpose: Looks up an email address in the Wayback Machine's archived web pages. This can provide historical context about the email address's mentions on the internet, including technical forums and past link shares.

Bing
Purpose: Searches Bing for any mentions of the email address online. This can include social media mentions, forum posts, and other instances where the email address appears on the web.

Leak-Lookup API-key (57db17115101f5721eca219be0cc1c7c)

Purpose: This module queries Leak-Lookup's database to find if the email address has been involved in any data breaches. This is crucial for assessing the exposure of the email address in question.

CommonCrawl
Purpose: Searches the CommonCrawl dataset for mentions of the email address. CommonCrawl's archive of the web can reveal where and how the email has been used across a vast array of sites.

Dehashed
Purpose: Similar to Leak-Lookup, Dehashed searches for the email address in databases of leaked and compromised accounts. This can reveal whether the email address has been part of any breaches.

DuckDuckGo
Purpose: Performs searches on DuckDuckGo for the email address, potentially uncovering mentions across various sites, forums, and other online platforms.

EmailCrawlr
Purpose: Looks for information associated with an email address, such as social media profiles, online presence, and potentially other emails linked to the person.

View File Metadata

File metadata can provide hackers with insights into organizations and personnel. For example, metadata within an image file can reveal the device that was used to create the image. This can reveal information that can be used to determine if the device is potentially vulnerable. Some files have metadata consisting of comments, the author’s name, usernames, the operating system, or the location at which the file was created. Metadata varies by the type of file and the device on which it was created. Hackers can use this information to piece together a means of attack.

In general, files that are posted on the public internet should have their metadata stripped or at least scrutinized. You can use **ExifTool**, among others, to remove or edit tags from individual files or a directory of files.

ExifTool comes in a GUI version that is available for Windows, MacOS, and Linux.

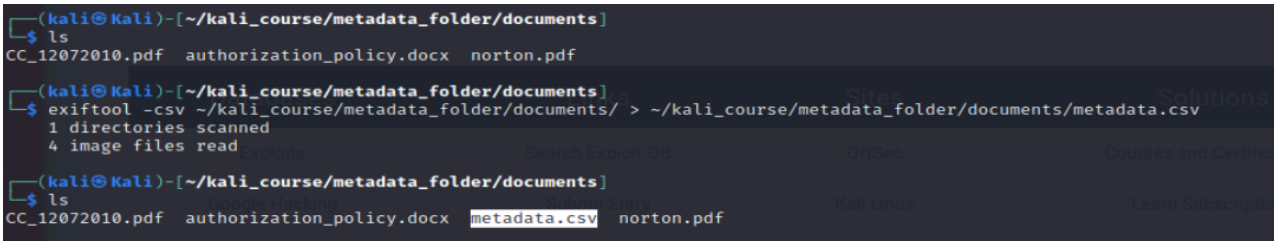
Install ExifTool

- navigate to https://www.kali.org/tools.
- Search for **libimage-exiftool-perl**
  or

```
┌──(kali㉿Kali)-[~]
└─$ sudo apt update

┌──(kali㉿Kali)-[~]
└─$ sudo apt install libimage-exiftool-perl
```

File Type : Documents, Audio, Video, Graphics, Archives

Running **Exiftool** for entire Folder



d. You can save the metadata for each image in the folder, or for individual images by adding the -csv option. For example:

```
┌──(kali㉿Kali)-[~]
└─$ exiftool -csv > /path/to/out.csv <File Or Dir>
```

f. On the internet, research some of the values that you find in the file. For example, the tag **CREATOR: gd-jpeg v1.0** indicates that the image was generated by the PHP GD library version 1.0. Search the internet for **PHP GD vulnerability** to learn more.

Advanced Searches (https://www.exploit-db.com/)

Google Advanced Searches(Dorking)

For most people, Google is a tool for searching for text, images, videos, and news on the internet by using simple string queries. However, for some, Google is a powerful and useful hacking tool and can be used for performing passive reconnaissance by using advanced search operators. The practice of using advanced Google search operators to find information and vulnerable servers is called Google dorking or Google hacking. Google dorking is used by hackers to try to find information that was never intended to be revealed publicly. It is a useful technique for conducting passive reconnaissance in penetration tests.

Note: When performing advanced queries, you may be prompted by Google to prove you are not a robot. If this occurs, as it probably will after several searches, simply complete the captcha and continue.

Explore Google dorking.

Table below shows the advanced search operators

Operator	Description
allintext:	Restricts results to pages with all words in the page text.
filetype:	Restricts results to pages of the specified file type (.pdf, .doc, .ppt,..)

intitle:	Restricts results to pages with a certain word (or words) in the title.
inurl:	Restricts results to pages with a certain word(or words) in the URL.
site:	Restricts results to pages from the specified domain.

Google advanced search example :

- ethical hacker **site:**pearson.com
- ethical hacker **inurl:**pearson.com
- ethical hacker **site:**pearson.com **filetype:**pdf

- **site:**company.ma **inurl:**admin
- **site:**company.ma **intitle:**login
- **site:**company.ma **intext:**employee **filetype**:pdf
- **site:**linkedin.com **intitle:**company **filetype**:pdf

- ethical hacker **intitle:**certification

- **allintext:**free ethical hacker practice test questions

Add quotes around searched text for more specified results

- **allintext:**free **“**ethical hacker practice test questions**”**

Google advanced search form example :

Type advanced search in the **Google search** window.

Conduct passive reconnaissance with advanced search operators.

Advanced search operators are useful for narrowing down search results as you have seen. This makes them useful for performing passive reconnaissance as well. Hackers will use advanced search operators to find vulnerabilities and information about potential targets. While the results of the searches may seem harmless on their own, when pieced together, they can provide valuable intelligence to a hacker. The hacker hopes to find sites or files that the target company did not intend to make public, or to find information that can be used for future attacks, such as social engineering attacks.

When performing these searches, use a target company of your choice. **Passive reconnaissance is legal but stop there because using any information you uncover for active reconnaissance is not**. If you do find vulnerabilities, consider informing the company so that they can correct the issue.

The Google Hacking Database

The Google Hacking Database (GHDB) is an index of user-created dorks that are designed to uncover interesting, and potentially sensitive, information that was unintentionally made publicly available on the internet.

Step 1: Explore the Google Hacking Database main page.

a. Do a Google search for GHDB. The first returned page should be The Google Hacking Database.
b. On the GHDB main page, click the Filters button in the top right of the window.

This allows you to filter the database results by Category or Author. There is also a Quick Search.

https://www.exploit-db.com/

Categories	Google Dork	Description
Footholds	inurl:adminpanel site:gov.*	This google dork indexes pages containing Admin Login Panels of government.
	mail/u/0 filetype:pdf	Pages Exposing internal Documents.
	intitle:"index of" "httpd.pid"	This file contains the Process ID (PID) of the Apache server's main process when it's running.
	intitle:"Index of /" +.htaccess	Access to the parent directory and more...
	intitle:"index of" env.cgi	Access to the env.cgi file

Categories	Google Dork	Description
Files Containing Usernames	intitle:"index of" "/usernames"	Display Files Containing Usernames
	intext:"-----BEGIN CERTIFICATE-----" ext:txt	This particular query aims to find text files (.txt) on the internet that contain SSL/TLS certificates
	jdbc:sqlserver://localhost:1433 + username + password ext:yml | ext:java	Exposed usernames and passwords
	filetype:csv intext:"Secret access key"	Exposed Secret access key

Step 4: Combine Category filters with search terms.

You can combine category filters with search terms to further refine and filter results to specific information.

a. Select Files Containing Passwords in the Categories drop down.
b. In the Quick Search window, type db_pass. This will return dork searches for database passwords.

The Wayback Machine

Website security has evolved over the decades. Websites used to publish information that is no longer considered safe. Webpage archives can reveal interesting information that is no longer available. The Wayback Machine is a useful tool for passively collecting information about a target that could be used in social engineering or other attacks. The Wayback Machine is an archive of the entire internet. It accesses every website and crawls it while taking screenshots and logging the data to a database. These endpoints can then be queried to pull down every path the site has ever crawled.

Open-Source intelligence (OSINT) Gathering

Open-source intelligence (OSINT) gathering is a method of gathering publicly available intelligence sources to collect and analyze information about a target. **OSINT** is “open source” because collecting the information does not require any type of covert methods. Typically, the information can be found on the Internet. The larger the online presence of the target, the more information that will be available. This type of collection can often start with a simple Google search, which can reveal a significant amount of information about a target. It will at least give you enough information to know what direction to go with your information-gathering process. Let's cover two tools that can be used for **OSINT** gathering: **Recong-ng** and **Shodan.**

Recong-ng

This module covers a number of individual sources and tools used for information gathering. These tools are all very effective for their specific uses; however, wouldn’t it be great if there were a tool that could pull together all these different functions? This is where **Recon-ng comes in. It is a framework developed by Tim Tomes of Black Hills Information Security.** This tool was **developed in Python** with Metasploit **msfconsole** in mind. If you have used the **Metasploit console** before, Recon-ng should be familiar and easy to understand.

Recon-ng is a modular framework, which makes it easy to develop and integrate new functionality. It is highly effective in social networking site enumeration because of its use of application programming interfaces (APIs) to gather information. It also includes a reporting feature that allows you to export data in different report formats. Because you will always need to provide some kind of deliverable in any testing you do, Recon-ng is especially valuable.

Using Recon-ng tool steps

```
┌──(kali㉿Kali)-[~]
└─$ recon-ng
                                  /\
                                 / \\ /\
Sponsored by...                 /\  /\/  \\V  \/\
                               / \\/ // \\\\\ \\ \/\
                              // // BLACK HILLS \/ \\
                             www.blackhillsinfosec.com
```

[recon-ng][default] > help

Available modules

[recon-ng][default] > marketplace search

Refresh the marketplace modules

[recon-ng][default] > marketplace search

Find different subdomains of domains (h4cker.org), Using bing_domain_web

[recon-ng][default] > marketplace search **bing**

install a module that note need D or K

[recon-ng][default] > marketplace install **recon/domains-hosts/bing_domain_web**

Load a module

[recon-ng][default] > modules load **recon/domains-hosts/bing_domain_web**

After loading module prompt changed

[recon-ng][default][bing_domain_web] > info

Execute module to find subdomains

[recon-ng][default][bing_domain_web] > options set SOURCE h4cker.org
SOURCE => h4cker.org
[recon-ng][default][bing_domain_web] > run

```
H4CKER.ORG
---------
[*] URL: https://www.bing.com/search?first=0&q=domain%3Ah4cker.org
```

shodan.io (https://www.shodan.io/)

Shodan is an organization that **scans the Internet 24 hours a day, 365** days a year. The results of those scans are stored in a database that can be queried at shodan.io or by using an API. You can use Shodan to query for vulnerable hosts, Internet of Things (IoT) devices, and many other systems that should not be exposed or connected to the public Internet. Figure 3-5 shows different categories of systems found by Shodan scans, including industrial control systems (ICS), databases, network infrastructure devices, and video games.

Bug Bounty

TIP Keep in mind that even though this is public information, you should not interact with any systems shown in Shodan results without permission from the owner. If the owner has a **bug bounty program**, you may get recognition and a reward for finding the affected system. A bug bounty is a program designed to reward security researchers and ethical hackers for finding vulnerabilities in a product, an application, or a system. In most cases, the compensation is monetary. **Omar Santos** has included several resources in his GitHub repository on how to get started in bug bounties; see https://github.com/The-Art-of-Hacking/h4cker/tree/master/bug-bounties.

Shodan Searches

On the Search Query Fundamentals screen, **the banner** is identified as the fundamental unit of data.

Use the shodan website to search for Vulnerable IoT Devices

Shodan provides a method to filter your search results using the syntax filter:value with no spaces. If the value contains spaces, such as city:"los angeles", you must enclose the value in double quotes. Some of the most popular search filters are:

country:XX Searches for a 2 digit country code
city:city-name Searches for a city by name
region:region-or-state-name Searches for a specific state or region
product:product-name Searches for a specific product by name
version:XX Searches for a specific product version
vuln:XX Searches for vulnerabilities that match a specific CVE number

```
country:US city:"Los Angeles" product:Apache version:2.4
```

A common configuration issue found on the internet is FTP servers that permit anonymous logins. Use the search string to find the FTP servers in San Jose, California.

```
port:21 country:US region:CA city:"San Jose" 230
```

This search uses the standard FTP TCP port 21, with location filters, and a text search for 230. 230 is the FTP successful login response code.

Use Shodan to search for specific products or services.

We can use shodan to search for a specific product, such as Apache services open on port 80. Formulate a query to find the Apache services in city

```
Apache port:80 city:”casablanca”
```

Use shodan from the CLI to Perform a Search

Find an API Key by selecting **Account > Overview** from the top right of the shodan web site screen.

```
┌──(kali㉿Kali)-[~]
└─$ shodan  init  S9caFIKbWYf50eYqgS92AYaKmcPYLyuv

Display the list of Shodan commands available
┌──(kali㉿Kali)-[~]
└─$ shodan  -h

Execute the same search
┌──(kali㉿Kali)-[~]
└─$ shodan  search webcam

Use the shodan myip command to find the registered IP address that
corresponds to your device.
The IP address returned is the source IP address that will be added to
any packets sent from your device to a destination on the internet.

┌──(kali㉿Kali)-[~]
└─$ shodan  myip

Return the summary information about a query

┌──(kali㉿Kali)-[~]
└─$ shodan  stats webcam
```

Performing Active Reconnaissance

Active Reconnaissance :

As mentioned earlier in this module, with each step of the information gathering phase, the goal is to gather additional information about the target. The process of gathering this information is called enumeration. So, let's talk about what kind of enumeration you would typically be doing in a penetration test. In an earlier example, we looked at the enumeration of hosts exposed to the Internet by h4cker.org. External enumeration of hosts is usually one of **the first things you do in a penetration test. Determining the Internet-facing hosts of a target network can help you identify the systems that are most exposed.** Obviously, a device that is publicly accessible over the Internet is open to attack from malicious actors all over the world. After you identify those systems, you then need to identify which services are accessible. A server should be behind a firewall, allowing minimal exposure to the services it is running. Sometimes, however, services that are not expected are exposed. To determine if a network is running any such services, you can run a port scan to enumerate the services that are running on the exposed hosts.

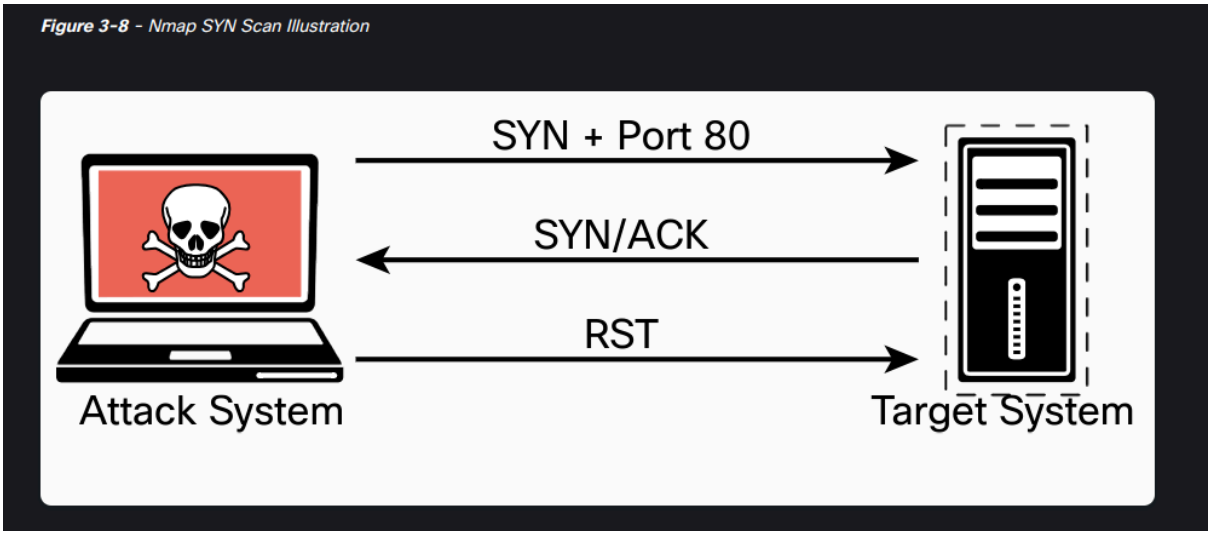
Port Scans

A port scan is an active scan in which the scanning tool sends various types of probes to the target IP address and then examines the responses to determine whether the service is actually listening. For instance, with an Nmap SYN scan, the tool sends a TCP SYN packet to the TCP port it is probing. This process is also referred to as half-open scanning because it does not open a full TCP connection. If the response is a SYN/ACK, this would indicate that the port is actually in a listening state. If the response to the SYN packet is an RST (reset), this would indicate that the port is closed or is not in a listening state. If the SYN probe does not receive any response, Nmap marks it as filtered because it cannot determine if the port is open or closed. Table 3-2 defines the SYN scan responses when using Nmap.

SYN Scan Responses

Nmap Port Status Reported	Response from target	Nmap Analysis
open	TCP SYN-ACK	The service is listening on the port.
Closed	TCP RST	The service is not listening on the port.
Filtered	No response from target or ICMP destination unreachable	The port is firewalled

Figure 3-8 - Nmap SYN Scan Illustration



Nmap Scan Types

The following sections cover some of the most common Nmap scanning options used for specific scenarios, including the following:

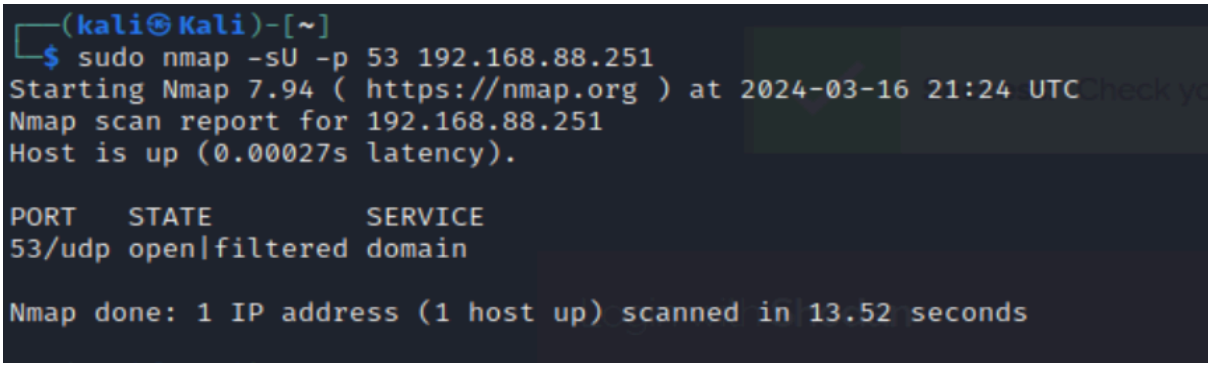
- TCP Connect Scan (**-sT)**
- UDP Scan (**-sU**)
- TCP FIN Scan (**-sF**)
- Host Discovery Scan (**-sn**)
- Timing Options (**-T 0-5**)
-

Omar Santos has also created an Nmap Cheat Sheet that includes all options and is available in his GitHub repository, *https://github.com/The-Art-of-Hacking/h4cker/blob/master/cheat_sheets/NMAP_cheat_sheet.md*.

UDP Scan (-sU)

The majority of the time, you will be scanning for TCP ports, as this is how you connect to most services running on target systems. However, you might encounter some instances in which you need to scan for UDP ports – for example, if you are trying to enumerate a DNS, SNMP, or DHCP server. These services all use UDP for communication between client and server. To scan UDP ports, Nmap sends a UDP packet to all ports specified in the command-line configuration. It waits to hear back from the target. If it receives an ICMP port unreachable message back from a target, that port is marked as closed. If it receives no response from the target UDP port, Nmap marks the port as open/filtered. Table 3-4 shows the UDP scan responses.

NOTE You should be aware that **ICMP** (**Internet Control Message Protocol**.) unreachable messages can sometimes be rate limited, and when they are, a UDP port scan can take much longer. ICMP rate limiting is primarily used for throttling worm or virus-like behavior and should normally be configured to allow 1% to 5% of available inbound bandwidth (at 10Mbps or 100Mbps speeds) or 100kpbs to 10,000kbps (at 1Gbps or 10Gbps speeds) to be used for ICMP traffic.



TCP FIN Scan (-sF)

There are times when a SYN scan might be picked up by a network filter or firewall. In such a case, you need to employ a different type of packet in a port scan. With the TCP FIN scan, a FIN packet is sent to a target port. If the port is actually closed, the target system sends back an RST packet. If nothing is received from the target port, you can consider the port open because the normal behavior would be to ignore the FIN packet. Table 3-5 shows the TCP FIN scan responses.

NOTE A TCP FIN scan is not useful when scanning Windows-based systems, as they respond with RST packets, regardless of the port state.

Host Discovery Scan (-sn)

A host discovery scan is one of the most common types of scans used to enumerate hosts on a network because it can use different types of ICMP messages to determine whether a host is online and responding on a network.

NOTE The default for the -sn scan option is to send an ICMP echo

request packet to the target, a TCP SYN to port 443, a TCP ACK to port 80, and an ICMP timestamp request. This is documented at https://nmap.org/book/man-host-discovery.html. If the target responds to the ICMP echo or the aforementioned packets, then it is considered alive.

Timing Options (-T0-5) Nmap scanner provides six timing templates that can be specified with the -T option and the template number (0 through 5) or name. Nmap timing templates enable you to dictate how aggressive a scan will be, while leaving Nmap to pick the exact timing values. These are the timing options:

- **T0 (Paranoid)** : Very slow, **used for IDS evasion**
- **T1 (Sneaky)** : Quite slow, **used for IDS evasion**
- **T2 (Polite)** : Slows down to consume less bandwidth, runs about 10 times slower than the default
- **T3 (Normal)** : Default, a dynamic timing model based on target responsiveness
- **T4 (Aggressive)** : Assumes a fast and reliable network and may overwhelm targets
- **T5 (Insane)** : Very aggressive; will likely overwhelm targets or miss open ports

Types of Enumeration

This section covers enumeration techniques that should be performed in the information-gathering phase of a penetration test. you will learn how and when these enumeration techniques should be used. This section also includes examples of performing these types of enumeration by using Nmap as well as a deep dive into packet crafting with Scapy.

- **Host Enumeration**

- **User Enumeration**
- **Group Enumeration**
- **Network Share Enumeration**
- **Additional SMB Enumeration Examples**
- **Web Page Enumeration/Web Application Enumeration**
- **Service Enumeration**
- **Exploring Enumeration via Packet Crafting**

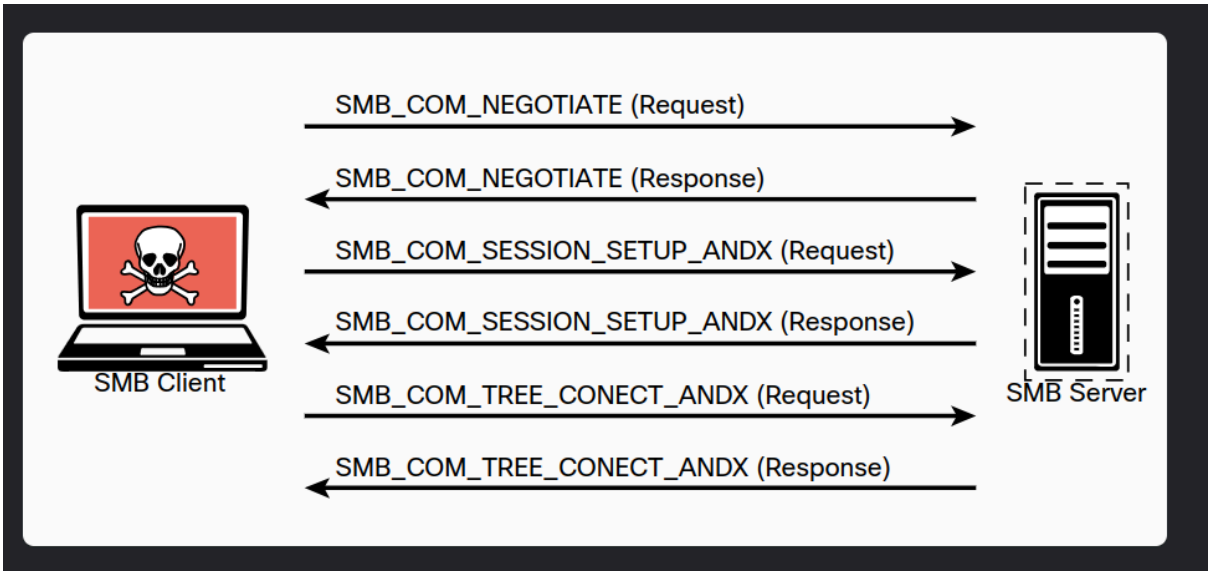
Host Enumeration

The enumeration of hosts is one of the first tasks you need to perform in the information-gathering phase of a penetration test. ***Host enumeration*** is performed **internally** and **externally**. When performed externally, you typically want to limit the IP addresses you are scanning to just the ones that are part of the scope of the test. This reduces the chance of inadvertently scanning an IP address that you are not authorized to test. When performing an internal host enumeration, you typically scan the full subnet or subnets of IP addresses being used by the target. Host enumeration is usually performed using a tool such as Nmap or Masscan; however, vulnerability scanners also perform this task as part of their automated testing.A sample Nmap ping scan being used for host enumeration on the network 192.168.88.0/24. In earlier versions of Nmap, the Nmap ping scan option was -sP (not -sn).

User Enumeration

Gathering a valid list of users is the step in cracking a set of credentials. When you have the username, you can then begin brute-force attempts to get the account password.You perform ***user enumeration*** when you have gained access to the internal network. On a Windows network, you can do this by manipulating the **Server Message Block (SMB)** protocol, which uses TCP port 445. Figure below illustrates how a typical SMB implementation works.

```
┌──(kali㉿Kali)-[~]
└─$ nmap - -script smb-enum-users.nse 192.168.88.251
```



- SMB_COM_NEGOTIATE: This message allows the client to tell the server what protocols, flags, and options it would like to use. The response from the server is also an SMB_COM_NEGOTIATE message. This response is relayed to the client about which protocols, flags, and options it prefers. This information can be configured on the server itself. A misconfiguration sometimes reveals information that you can use in penetration testing. For instance, the server might be configured to allow messages without signatures. You can determine if the server is using share- or user-level authentication mechanisms and whether the server allows plaintext passwords. The response from the server also provides additional information, such as the time and time zone the server is using. This is necessary information for many penetration testing tasks.
- SMB_COM_SESSION_SETUP_ANDX : After the client and server have negotiated the protocols, flags, and options they will use for communication, the authentication process begins. Authentication is the primary function of the SMB_COM_SESSION_SETUP_ANDX message. The information sent in this message includes the client username, password, and domain. If this information is not encrypted, it is easy to sniff it right off the network. Even if it is encrypted, if the mechanism being used is not sufficient, the information can be revealed using tools such as Lanman and NTLM in the case of

Microsoft Windows implementations. The following example shows this message being used with the smb-enum-users.nse script:

```
nmap --script smb-enum-users.nse <host>
```

Group Enumeration

For a penetration tester, ***group enumeration i***s helpful in determining the authorization roles that are being used in the target environment. The Nmap NSE script for enumerating SMB groups is smb-enum-groups. This script attempts to pull a list of groups from a remote Windows machine. You can also reveal the list of users who are members of those groups. The syntax of the command is as follows:

```
┌──(kali㉿Kali)-[~]
└─$ nmap  - -script  smb-enum-groups.nse -p445 <host>

┌──(kali㉿Kali)-[~]
└─$ nmap  - -script  smb-enum-groups.nse -p445  192.168.56.3
```

Network Share Enumeration

Identifying systems on a network that are sharing files, folders, and printers is helpful in building out an attack surface of an internal network. The Nmap **smb-enum-shares.nse** NSE script uses **Microsoft Remote Procedure Call (MSRPC)** for **network share enumeration.**

```
┌──(kali㉿Kali)-[~]
└─$ nmap  - -script  smb-enum-shares.nse -p 445 <host>
```

```
┌──(kali㉿Kali)-[~]
└─$ nmap  - -script  smb-enum-shares.nse -p 445  192.168.56.3
```

Additional SMB Enumeration Examples

SMB (Server Message Block) enumeration is a critical task in network security and penetration testing, used to gather information about network shares, users, groups, and other services provided by servers running the SMB protocol. SMB is commonly used in Windows environments to provide shared access to files, printers, and serial ports. However, Linux systems can also offer SMB services using Samba. Enumerating SMB can reveal valuable information that can be used for further penetration testing or to identify misconfigurations and security vulnerabilities.

Tools for SMB Enumeration:

Nmap: A powerful network scanner that can identify open ports and services, including SMB. Nmap can use scripts (smb-enum-shares.nse, smb-enum-users.nse, etc.) from the Nmap Scripting Engine (NSE) to enumerate SMB information.

Enum4linux: A tool specifically designed for enumerating information from Windows and Samba systems. It can gather information about SMB shares, users, and groups, among other things.

SMBClient: A command-line tool that allows users to connect to SMB shares, list available shares on a server, and perform actions similar to FTP.

Metasploit: Contains several modules that can be used to enumerate SMB versions, shares, and vulnerabilities.

The system used in earlier examples(with the IP 192.168.88.251) is running **Linux** and **Samba**.However, it is not easy to determine that it is a Linux system from the results of previous scans. An easy way to perform additional enumeration and fingerprinting of the applications and operating system running on a host is by using the **nmap-sC** command. The -sC option runs the most common NS scripts based on the ports found to be open on the target system.

Note: You can locate the installed NSE scripts in kali Linux and Parrot OS by simply using the **locate *.nse commande**. The site *https://nmap.org/book/man-nse.html* includes a detailed explanation of the NSE and how to create new scripts using the **Lua programming language.**

```
┌──(kali㉿Kali)-[~]
└─$ sudo samba -V
  Version 4.18.5-Debian
```

You can also use tools such as **enum4linux** to enumerate Samba shares, including user accounts, shares, and other configurations.

```
┌──(kali㉿Kali)-[~]
└─$ enum4linux 192.168.88.251
```

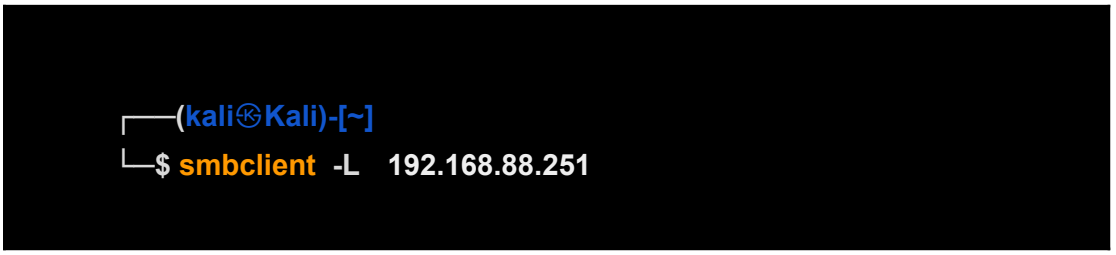
There is a Python-based enum4linux implementation called enum4linux-ng that can be downloaded from *https://github.com/cddmp/enum4linux-ng*.

Example 3-30 shows an example of SMB enumeration using enum4linux-ng.

Example 3-30 - Enumeration Using enum4linux-ng

```
┌──(kali㉿Kali)-[~]--[~/enum4linux-ng]
└─$ ./enum4linux-ng.py -As 192.168.88.251
ENUM4LINUX - next generation

 ==========================
|    Target Information    |
 ==========================
[*] Target ........... 197.146.174.161
[*] Username ......... ''
[*] Random Username .. 'cykulgme'
[*] Password ......... ''
[*] Timeout .......... 5 second(s)
```

The highlighted lines in Example below show the enumerated users, Samba version, and shared folders, You can also use simple tools such as **smbclient** to enumerate shares and other information from a system **running SMB**, as demonstrated



Web Page Enumeration/Web Application Enumeration

Once you have identified that a web server is running on a target host, the next step is to look at the web application and begin to map out the attack surface performing ***web page enumeration*** or often referred to as ***web application enumeration***. You can map out the attack surface of a web application in a few different ways. The handy **Nmap** tool actually has an NSE script available for brute forcing the directory and file paths of web applications. Armed with a list of known files and directories used by common web applications, it probes the server for each of the items on the list. Based on the response from the server, it can determine whether those paths exist. This is handy for identifying things like the Apache or Tomcat default manager page that are commonly left on web servers and can be potential paths for exploitation.

```
┌──(kali㉿Kali)-[~]
└─$ nmap  -sV –script=http-enum -p 80   192.168.88.251
```

Another web server enumeration tool we should talk about is **Nikto,** an open-source web vulnerability scanner that has been around for many years. It's not as robust as the commercial web vulnerability scanner; however, it is very handy for running a quick script to enumerate information about a web server and the applications it is hosting. Because of the speed at which **Nikto** works to scan a web server, it is very noisy. It provides a number of options for scanning, including the capability to authenticate to a web application that requires a username and password.

```
┌──(kali㉿Kali)-[~]
└─$ nikto  -h   192.168.88.251
```

Service Enumeration

Service enumeration is the process of identifying the services running on a remote system,and it is a primary focus of what **Nmap** does as a port scanner.Earlier discussion in this module highlights the various scan types and how they can be used to bypass filters. When you are connected to a system that is on a directly connected network segment, you can run some additional scripts to enumerate further. A port scan takes the perspective of a credentialed remote user. The Nmap **smb-enum-processes** NSE script enumerates services on Windows systems, and it does so by using credentials of a user who has access to read the status of services that are running. This is a handy tool for remotely querying a Windows system to determine the exact list of services running.

```
┌──(kali㉿Kali)-[~]
└─$ nmap  --script smb-enum-processes.nse  --script -args
smbusername=<username>, smbpass=<password> -p445 <host>
```

Exploring Enumeration via Packet Crafting

When it comes to enumeration via packet crafting and generation, **Scapy** is one of pentesters' favorite tools and frameworks. **Scapy** is a very comprehensive Python-based framework or ecosystem for packet generation. This section looks at some of the simple ways you can use this tool to perform basic network reconnaissance.

> **NOTE** Scapy must be run with **root permissions** to be able to modify packets.

```
┌──(kali㉿Kali)-[~]
└─$ sudo  scapy
```

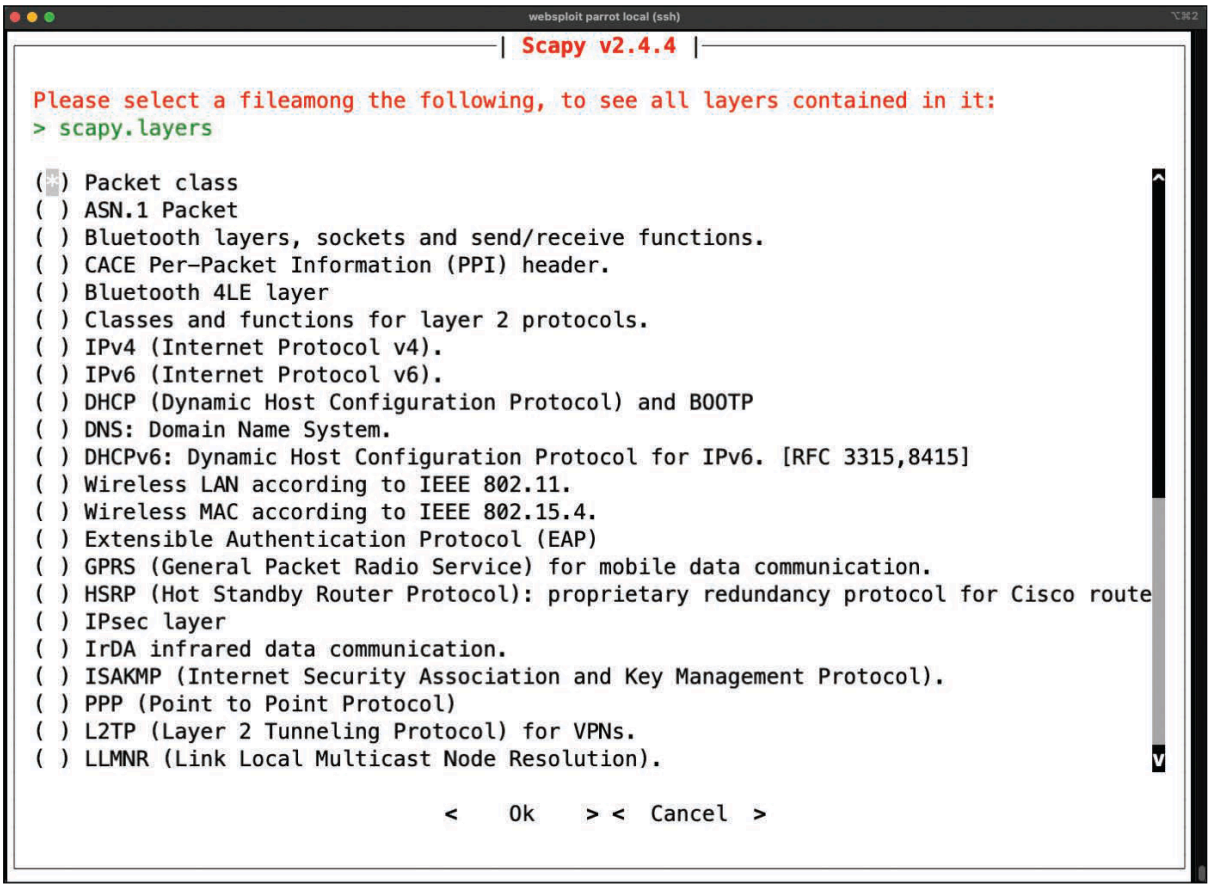
List all available format and protocols

```
┌──(kali㉿Kali)-[~]
└─$ >>>  ls()
```

Use ls() with option TCP to display all fields of a specific protocol.

```
┌──(kali㉿Kali)-[~]
└─$ >>>  ls(TCP)
```

```
Display another screen

┌──(kali㉿Kali)-[~]
└─$ >>> explore()
```



An example shows how easy it is to begin crafting packets. In this example, a simple ICMP packet is crafted with malicious_payload as the payload being sent to the destination host 192.168.88.251.

```
┌──(kali㉿Kali)-[~]
└─$ >> send(IP(dst="192.168.88.251")/ICMP()/"malicious_payload")
```

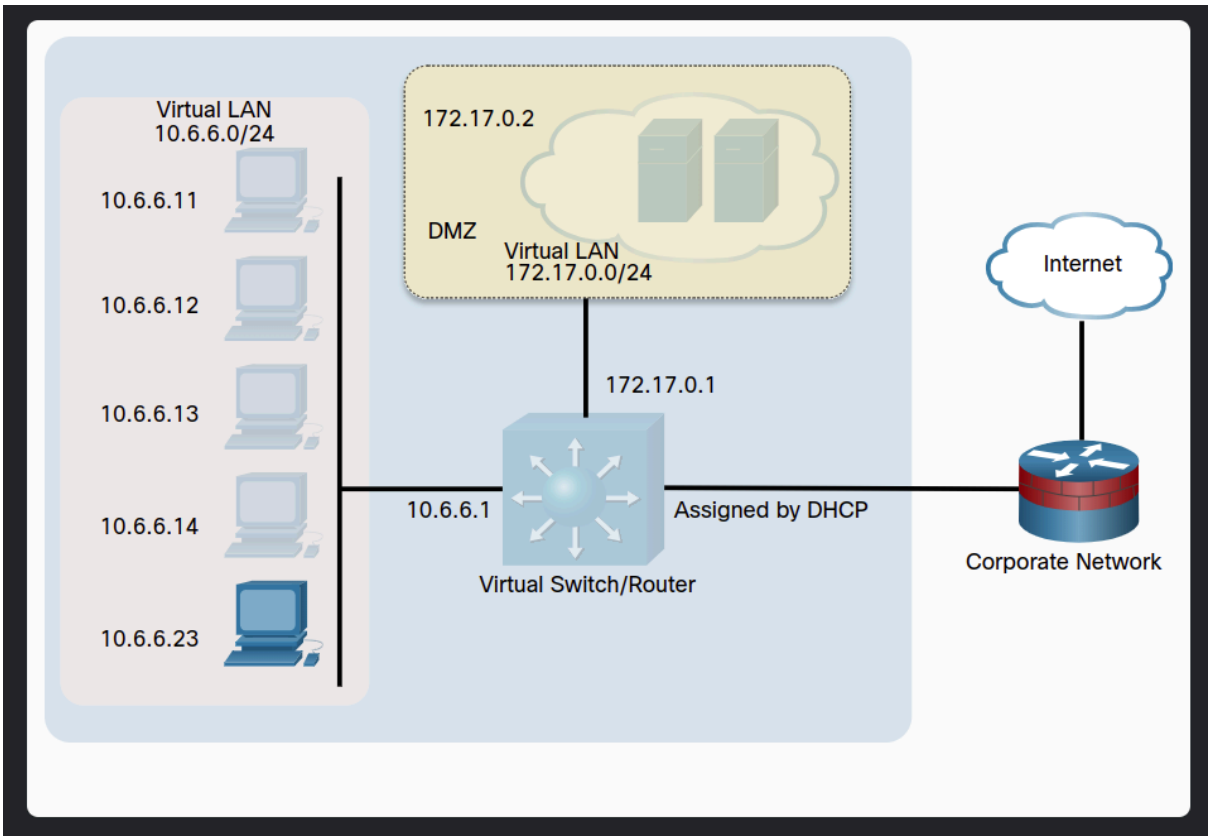
Shows the ICMP packet received by the target system (192.168.88.225/Vulnhost-1).The **tshark** packet capture tool is used to capture the crafted ICMP packet.

```
┌──(kali㉿Kali)-[~]
└─$ sudo tshark host 192.168.78.142


┌──(kali㉿Kali)-[~]
└─$ sudo scapy
>>> ans, unans = sr(IP(dst='192.168.88.251')/TCP(dport=445,flags='S'))
```

You can use Scapy as a scanner in many different ways. Omar Santos has several examples of Python scripts to perform network and system scanning using Scapy at his GitHub repository; see *https://github.com/The-Art-of-Hacking/h4cker/blob/master/python_ruby_and_bash*. However, you can do a simple TCP SYN scan to any given port, as demonstrated in Example 3-40. In this example, a TCP port 445 SYN packet is sent to the host with the IP address 192.168.88.251. The output indicates that it received one answer, but it doesn't specify what the actual answer (response) was.

Enumeration with Nmap

Topology



Nmap is a powerful open-source tool for network mapping and discovery. In this lab, you will use **Nmap** as part of your **active reconnaissance strategy.**

Investigate Nmap

```
┌──(kali㉿Kali)-[~]
└─$ nmap -V
```

Option	Description
-A	Aggressive scan that enables OS detection, version detection, script scanning and traceroute
-O	Enables OS detection
-p <port ranges>	Allows for specific ports or port ranges to be scanned
-sF	Performs TCP FIN scan
-sn	Performs host discovery scan
-sS	Performs TCP SYN scan
-sT	Performs TCP Connect scan
-sV	Probes open ports to determine service/version info

-T<0-5>	**Sets the timing of the scan. Higher numbers produce results faster. Slower scans elude detection better.**
-v	**Increases the verbosity of the output**
--open	**Only reports open (or possibly open) ports**

Perform Basic Nmap Scans

To quickly scan the DMZ for active hosts, you can perform a discovery scan. In a discovery scan, the scanning host sends an ICMP echo request (ping), a TCP SYN to port 443, a TCP ACK to port 80, and an ICMP timestamp request. A response to any of the requests indicates that the host is up and the IP protocol stack on the host is functioning. Enter the following command to scan the DMZ network:

```
┌──(kali㉿Kali)-[~]
└─$ nmap -sn 10.2.6.0/24
```

The host 10.6.6.23 was identified as suspicious in a **Wireshark capture**, and it is necessary to perform additional reconnaissance to discover more about the computer and its services. Use the nmap command to execute a default scan on the target host.

```
┌──(kali㉿Kali)-[~]
└─$ nmap 10.6.6.23
```

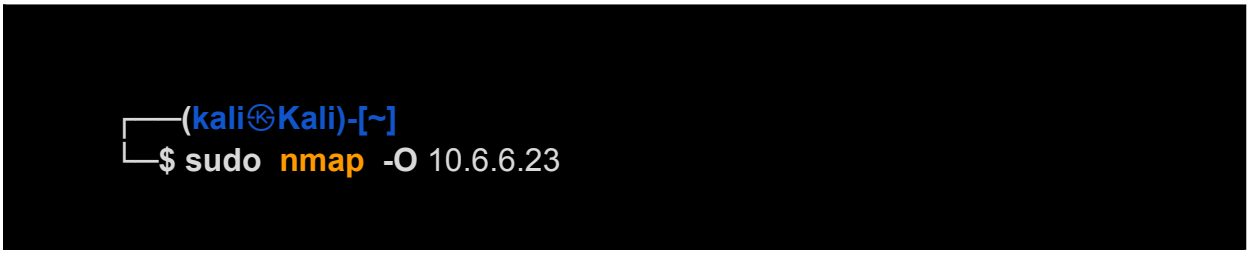
By default, Nmap performs a connect scan of 1000 most common TCP ports. This makes use of the operating system's networking software to establish a full TCP connection. This type of scan creates a lot of networking traffic and increases the probability of detection by intrusion detection services. You can also specify a TCP connect scan using the command option **nmap -sT.**

Status	Response Received	Interpretation
Open	TCP SYN-ACK	There is a service listening on the identified port.
Closed	TCP RST	There is no service listening on the identified port.
Filtered	No response, or an ICMP destination unreachable message received.	The port is being filtered by a firewall.

ICMP, short for Internet Control Message Protocol, is used in IP (Internet Protocol) networks for sending error messages and operational information indicating, for example, that a requested service is not available or that a host or router could not be reached. ICMP is integral to the IP layer and is used to report problems, not to carry application data. Some common uses include the ping command (for testing reachability) and traceroute (for diagnosing the route path and measuring transit delays of packets across an IP network).

The -O option can be used to further determine information about the operating system running on the target host. Some Nmap options require additional permissions and must be run as root or using the sudo

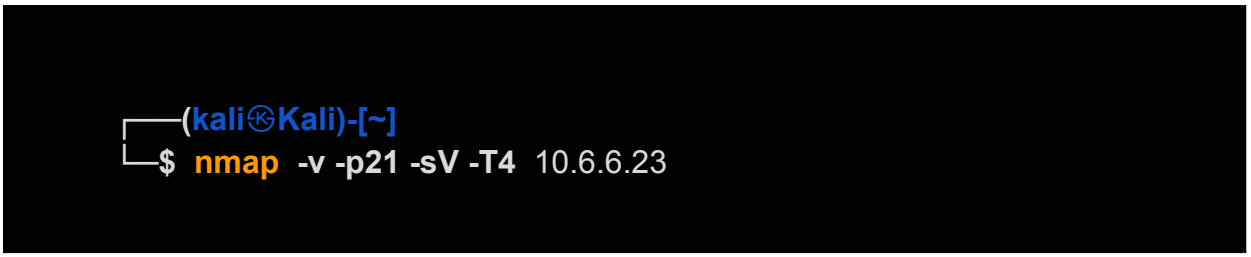
command. To find operating system information on the target host, use the nmap -O command. Enter the password of kali when prompted.



Obtain additional information about the host and services

To provide additional information about the target computer, it is possible to combine different options into a single command line.

The previous command identified several potentially open ports on the 10.6.6.23 host. **You can use -v, -P and -sV to find additional information about the services running on the open ports.** This command provides information about the FTP service running on the port 21 on the target in verbose mode, with the timing set to fast (-T4)



The -A option executes OS detection, version detection, script scanning, and traceroute. The -A scan can be very intrusive and therefore **will be detected by many IDS systems,** so ensure that you have permission before attempting this scan outside of the lab environment. To gather more information regarding the FTP service, enter the command nmap -p21 -sV -A 10.6.6.23

An **Intrusion Detection System (IDS)** is a security technology that monitors network or system activities for malicious actions or policy violations. It reports these activities to an administrator or collects centrally for analysis. IDS can detect suspicious activity through signature-based methods (looking for specific patterns, such as malware signatures) or anomaly-based methods (detecting deviations from normal behavior).

Investigate SMB Services with Scripts

The Server Message Block (SMB) protocol is a network file sharing protocol supported on Windows computers and by SAMBA on Linux. SMB enables applications to read and write files or request services over a network. Open public shared devices such as **print** servers on a network, can be accessed through SMB

The earlier scan of open ports on the target computer indicates that the SMB ports 139 and 445 are open. Find more information on these ports using the -A and -p command options. The -A option executes several functions including running the default scripts.

Specify more than one pot to scan bt listing them separately with a comma between them.

```
┌──(kali㉿Kali)-[~]
└─$ nmap -A -p139, 445 10.6.6.23
```

Nmap contains the powerful **Nmap Scripting Engine (NSE)**, which enables the programming of various Nmap options and conditional actions to be taken as a result of the responses. **NSE** has built-in scripts that enumerate users, groups, and network shares. One of the more commonly used scripts for SMB discovery is the **smb-enum-users.nse** script.

```
┌──(kali㉿Kali)-[~]
└─$ nmap -- script smb-enum-users.nse -p139, 445 10.6.6.23
```

A serious security concern is the existence of publicly shared directories (folders). You can enumerate the network shares using another NSE script, **smb-enum-shares.nse**. To discover shared directories on the target computer.

```
┌──(kali㉿Kali)-[~]
└─$ nmap -- script smb-enum-shares.nse -p445 10.6.6.23

smb-vuln- * designed to detect various SMB vulnerabilities in the taget

┌──(kali㉿Kali)-[~]
└─$ nmap -- script 'smb-vuln-*' -p 445 10.6.6.23

To list all Nmap scripts: ls /usr/share/nmap/scripts/
┌──(kali㉿Kali)-[~]
└─$ ls /usr/share/namp/scripts
```

Packet Inspection and Eavesdropping

As a penetration tester, you can use tools like **Wireshark**, **tshark**, and **tcpdump** to collect packet captures for packet inspection and eavesdropping. Anyone who has been involved with networking or security has at some point used these tools to capture and analyze traffic on a network. For a penetration tester, such tools can be convenient for performing passive reconnaissance. Of course, this type of reconnaissance requires either a physical or a wireless connection to

the target. If you are concerned about being detected, you are probably better off attempting a wireless connection because it would not require you to be inside the building. Many times, a company's wireless footprint bleeds outside its physical walls. This gives a penetration tester an opportunity to potentially collect information about the target and possibly gain access to the network to sniff traffic

Tools of -active or -passive reconnaissance

Nikto	-**active**
Nmap	-**active**
tshark	-**passive**
tcpdump	-**passive**
Wireshark	-**passive**
enum4linux	-**active**

Packet Crafting with Scapy

Investigate the Scapy tool

Scapy is a multi-purpose tool originally written by Philippe Biondi. In this part, you will load the Scapy tool and explore some of its capabilities. Tools like Scapy should only be used to scan or communicate with machines that you own or have written permission to access.
https://scapy.readthedocs.io/en/latest/introduction.html

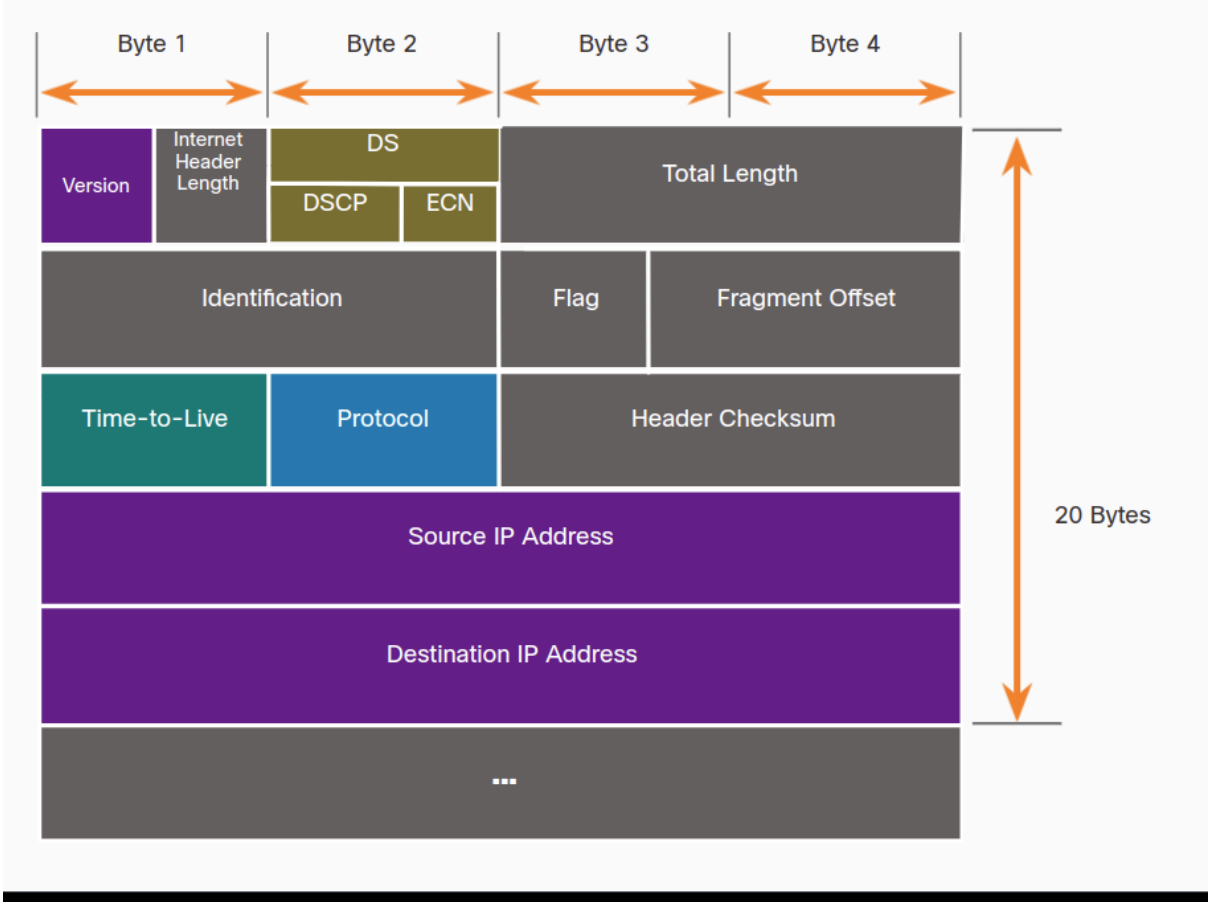
Use Scapy interactive command mode

The commands to craft and send packets require root privileges to run. Use the **sudo su** command to obtain root privileges before starting Scapy

```
┌──(kali㉿Kali)-[~]
└─$ sudo su

┌──(root㉿Kali)-[/home/kali]
└─#

┌──(root㉿Kali)-[/home/kali]
└─# scapy

list all of the available default formats and protocols  included in scapy

>>> ls()

To list  the available fields in an IP packet header

>>> ls(IP)
```

TFTP is a protocol **used to send and receive files on a LAN segment**. It is commonly used to back up configuration files on networking devices.

Examine the fields in an IPv4 packet header.



- **Version (4 bits):** Specifies the IP protocol version, IPv4 in this case.

- **Internet Header Length (IHL) (4 bits):** Indicates the header length in 32-bit words. The minimum value is 5, which corresponds to a header length of 20 bytes.

- **Type of Service (ToS) (8 bits):** Now referred to as Differentiated Services Code Point (DSCP) and Explicit Congestion Notification (ECN), it is used to specify the packet's priority and QoS.

- **Total Length (16 bits):** Specifies the entire packet size in bytes, including header and data.

- **Identification (16 bits):** Used in fragmentation to reassemble packets.

- **Flags (3 bits):** Controls and identifies fragments.
  Bit 0: Reserved, must be zero.
  Bit 1: Don't Fragment (DF).
  Bit 2: More Fragments (MF).

- **Fragment Offset (13 bits):** Indicates the position of the fragment's data in the original data packet.

- **Time to Live (TTL) (8 bits):** Limits the packet's lifespan to prevent it from looping indefinitely. It is decremented by each router, and the packet is discarded when TTL reaches zero.

- **Protocol (8 bits):** Identifies the protocol of the payload data (e.g., TCP is 6, UDP is 17).

- **Header Checksum (16 bits):** Used for error-checking of the header.

- **Source IP Address (32 bits):** The IP address of the originating host.

- **Destination IP Address (32 bits):** The IP address of the final destination.

Use Scapy to sniff network Traffic

Scapy can be used to capture and display network traffic, similar to a **tcpdump** or **tshark** packet collection.

```
┌──(root㉿Kali)-[/home/kali]
└─# scapy

Use the sniff function to collect traffic using the default eth0 interface of VM

>>> sniff()
```

Open a second terminal window and ping an internet address, such as www.cisco.com. Remember to specify the count using the -c argument.

```
┌──(kali㉿Kali)-[~]
└─$ ping -c 5  cisco.com
```

Return to the terminal window that is running the Scapy tool. Press CTRL-C to stop the capture. You should receive output similar to what is shown here:

```
>>>
^C<Sniffed: TCP:3 UDP:68 ICMP:56 Other:10>
```

View the captured traffic using the summary() function. The a=_ assigns the variable a to hold the output of the sniff() function. The underscore (_) in Python is used to temporarily hold the output of the last function executed.

```
View the capture traffic with summary
        >>> a=_
        >>> a.summary()
```

Capture and save traffic on a specific interface

In this step, you will capture traffic to and from a device connected to a virtual network in your kali Linux VM.

Examine the collected packets

In this step, you will filter the collected traffic yo include only **ICMP** traffic, limit the number of packets being collected, and view the individual packet details.

```
Use interface ID associated with 10.6.6.1 (br-internal) to capture ten ICMP packets sent and received on the internal VM network.
 sniff(iface="br-internal",filter = "icmp",count = 10)
        >>> sniff( iface="br-internal",filter = "icmp",count = 10 )
```

Open a second terminal window and ping the host at 10.6.6.23

```
        ┌──(kali㉿Kali)-[~]
        └─$ ping -c 10  10.6.6.23
```

```
    >>> <Sniffed: TCP:0 UDP:0 ICMP:10 Other:0>

View the capture traffic with nsummary()
        >>> a=_
        >>> a.nsummary()
```

Create and Send an ICMP Packet

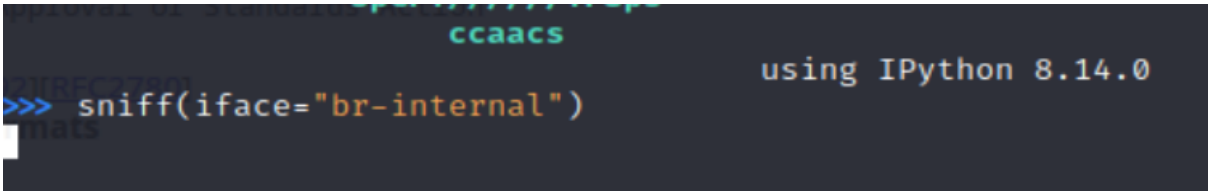
ICMP is a protocol designed to send control messages between network devices for various purposes. There are **many types of ICMP packets, with echo-request and echo-reply** the most familiar to IT

technicians. To see a list of the message types that can be sent and received using ICMP, navigate to https://www.iana.org/assignments/icmp-parameters/icmp-parameters.xhtml

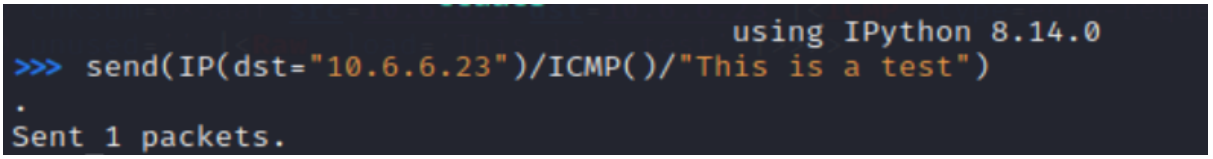
Create and Send a TCP SYN Packet.

In this part, you will use Scapy to determine id port 445, Microsoft Windows drive share port, is open on the target system at 10.6.6.23.

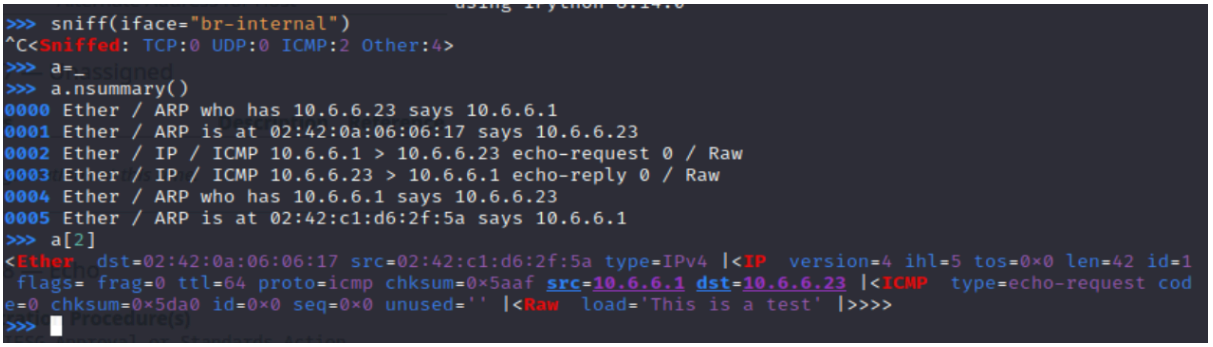
original Scapy



custom Scapy



original Scapy : view the capture ICMP packets



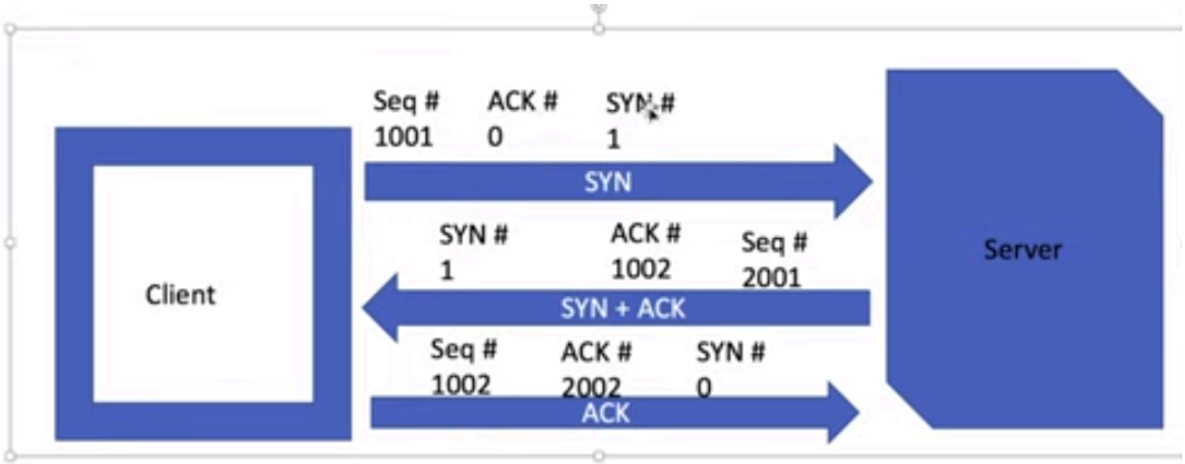
In original Scapy stop the packet capture by CTRL-C

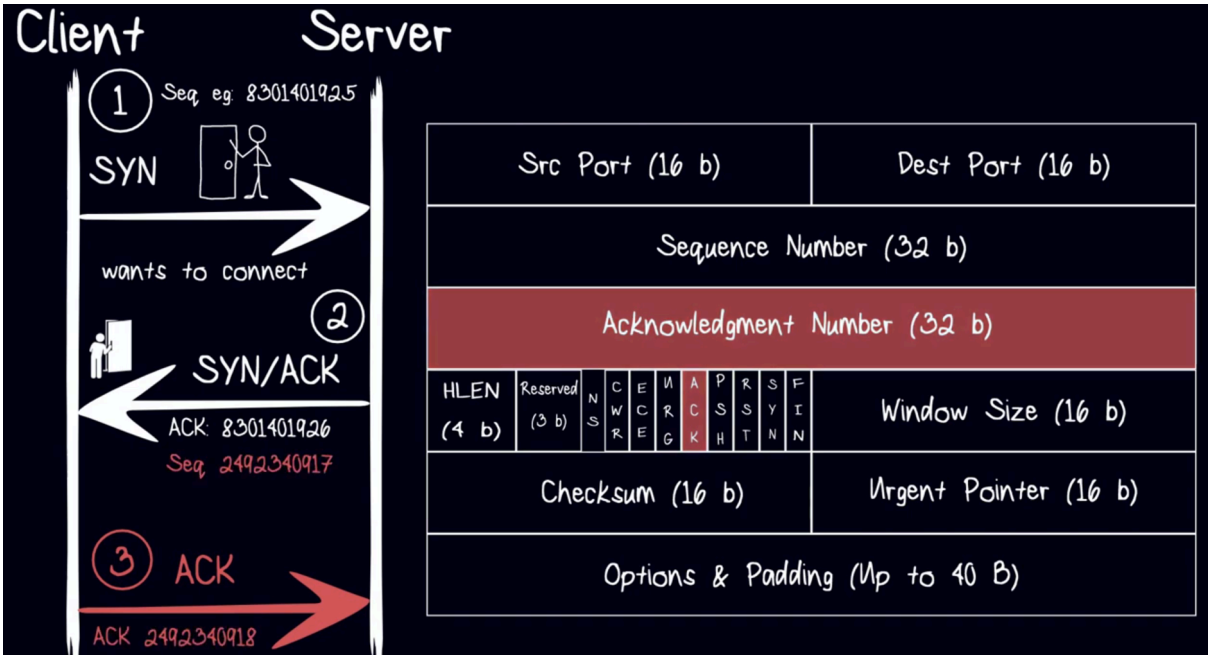
```
>>> sniff(iface="br-internal")
^C<Sniffed: TCP:3 UDP:24 ICMP:0 Other:6>
>>>
```

Display the detail of TCP packet

```
>>> a[2]
<Ether  dst=02:42:0a:06:06:17 src=02:42:c1:d6:2f:5a type=IPv4 |<IP  version=4 ihl=5 tos=0x0
 len=40  id=1 flags= frag=0 ttl=64 proto=tcp chksum=0x5aac src=10.6.6.1 dst=10.6.6.23 |<TCP
 sport=ftp_data dport=microsoft_ds seq=0 ack=0 dataofs=5 reserved=0 flags=S window=8192 chk
sum=0x6dee urgptr=0 |>>>
>>>
```

TCP Handshake, SYN, SYN / ACK, ACK





2. How could creating an ICMP echo-request packet with a spoofed source address create a denial of service attack on against a target host?

Sending thousands of packets to different hosts with same spoofed source address will cause all of the echo-reply packets to be sent to the target host. This will result in a distributed denial of service attack.

Network Sniffing with Wireshark

In this lab, you will use Linux utility **tcpdump** to capture and save network traffic. you will then use **Wireshark** to investigate the traffic capture.

- **Prepare the host to capture network traffic.**
- **Capture and save network traffic.**
- **View and Analyze the Packet capture.**

Wireshark is a network packet capture utility that can be used by network administrators to troubleshoot network problems. It can also be used to eavesdrop on network communications to passively collect

information about users and services. **Wireshark is considered a passive** tool because it does not create traffic on the network.

Prepare the host to capture Network traffic.

```
Display current directory
┌──(kali㉿Kali)-[~]
└─$ pwd

Display IP and MAC addresses fron eth0 internet Interface
inet 10.0.2.15
 ether 08:00:27:4a:f3:6e

┌──(kali㉿Kali)-[~]
└─$ ifconfig

Determine the default gateway
┌──(kali㉿Kali)-[~]
└─$ ip route

Determine the DNS server IP address
┌──(kali㉿Kali)-[~]
└─$ cat /etc/resolv.conf
```

Capture and Save network Traffic

In this part, you will use **tcpdump** from **CLI** to capture traffic. You will use command options to save the traffic to a packet capture **(pcap)** file. These records can then be analyzed using different applications that read pcap files, including **Wireshark**.

```
Display the Interface name eth0
┌──(kali㉿Kali)-[~]
└─$ ifconfig

Execute tcpdump with sudo and display in the browser “skillsforall.com” login
┌──(kali㉿Kali)-[~]
└─$ sudo tcpdump -i eth0 -s 0 -w packetdump.pcap

List the pakectdump.pcap file
┌──(kali㉿Kali)-[~]
```

```
└─$ ls packetdump.pcap
```

View and Analyze the Packet Capture.

In this part, you will use **Wireshark** to analyze the packet capture file that you created in the previous part of this lab.

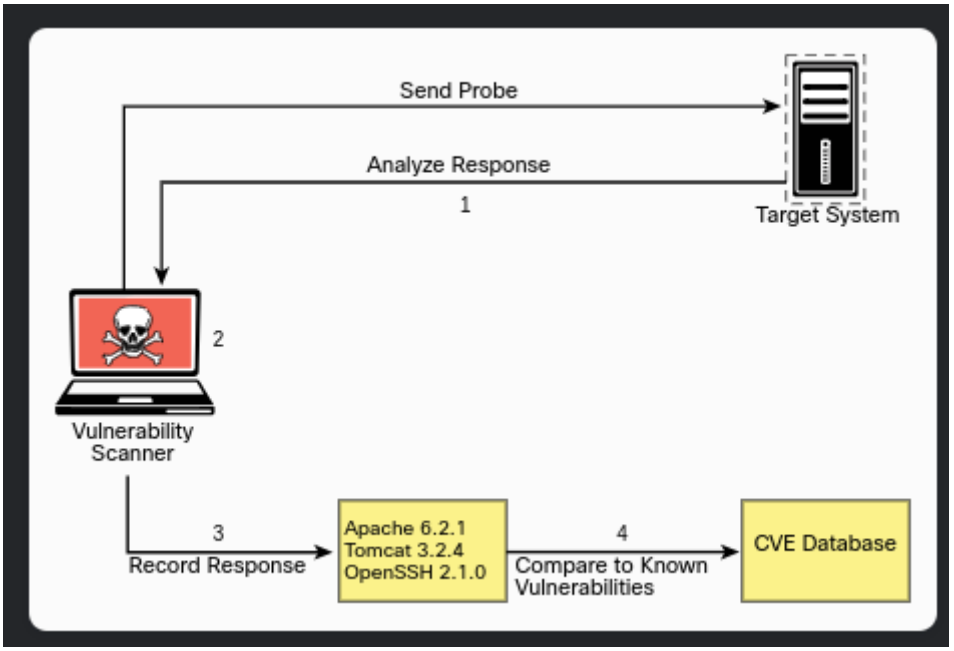
```
Open Wireshark
┌──(kali㉿Kali)-[~]
└─$ wireshark

In Wireshark application
Use File -> Open enter packetdump.pcap file
```

Understanding the Art of Performing Vulnerability Scans

Once you have identified the target hosts that are available and the services that are listening on those hosts, you can begin to probe those services to determine if there are any weaknesses; this is what vulnerability scanners do. **Vulnerability scanners use a number of different methods to determine whether a service is vulnerable**. The primary method is to identify the version of the software that is running on the open service and try to match it with an already known vulnerability. For instance, if a vulnerability scanner determines that a Linux server is running an outdated version of the Apache web server that is vulnerable to remote exploitation, it reports that vulnerability as a finding.

How a Typical Automated Vulnerability Scanner Works

Now let's take look at how typical vulnerability scanners work, they are all different in some ways, but most of them follow a similar process:



step-1: In the discovery phase, the scanner uses a tool such as **Nmap** to perform host and port enumeration. Using the results of the host and port enumeration, the scanner begins to probe open ports for more information.

step-2: When the scanner has enough information about the open port to determine what software and version are running on that port, it records that information in a database for further analysis. The scanner can use various methods to make this determination, including using **banner information**.

step-3: The scanner tries to determine if the software that is listening on the target system is susceptible to any known vulnerabilities. It does this by correlating a database of known vulnerabilities against the information recorded in the database about the target services.

step-4: The scanner produces a report on what it suspects could be vulnerable. Keep in mind that results are often false positives and need

to be validated. At the very least, this type of tool gives you an idea of where to look for vulnerabilities that might be exploitable.

Step 1	In the discovery phase, the scanner uses a tool such as Nmap to perform host and port enumeration.
Step 2	The scanner records what software and version are running on an open port into a database for further analysis.
Step 3	The scanner tries to determine if the software that is listening on the target system is susceptible to any known vulnerabilities.
Step 4	The scanner produces a report on what it suspects could be vulnerable.

Type of Vulnerability Scans

The type of vulnerability scan to use is usually driven by the scan policy that is created in the automated vulnerability scanning tool. Each tool has many options available for scanning. You can often just choose to do a full scan that will operate all scanning options. although you might not be able to use every option (for instance, if you are scanning a production environment or a device that is prone to crashing when scanning occurs). In such situations, **you must be careful to select only the scan options** that are less likely to cause issues.

Scan Types:

- **Unauthenticated Scans**
- **Authenticated Scans**
- **Discovery Scans**
- **Full Scans**
- **Stealth Scans**
- **Compliance Scans**

Unauthenticated Scans:

By default, vulnerability scanners do not use credentials to scan a target.

If you provide only the IP address of the target and click Scan, the tool will begin enumerating the host from the perspective of an unauthenticated remote attacker. An unauthenticated Scan shows only the network services that are exposed to the network. The scanner attempts to enumerate the ports open on the target host. If the service is not listening on the network segment that the scanner is connected to, or if it is firewalled, the scanner will report the port as closed and move on. However, this does not mean that there is not a vulnerability.
Sometimes it is possible to access ports that are not exposed to the network via SSH port forwarding and other tricks. It is still important to run a credentialed (or authenticated) scan when possible.

NOTE Authenticated scans may provide a lower rate of false positives than unauthenticated scans.

Authenticated Scans

In some cases, it is best to run an authenticated scan against the target to get a full picture of the attack surface. An authenticated scan requires you to provide the scanner with a set of credentials that have root-level access to the system**. The scanner actually logs in to the target via SSH or some other mechanism**. It then runs commands like **netstat** to gather information from the system.

```
netstat command in root -level access showing different result without root
┌──(kali㉿Kali)-[~]
└─$ netstat -tunap

┌──(root㉿Kali)-[/home/kali]
└─$ netstat -tunap
```

Discovery Scans

A discovery scan aims to map a target's attack surface, with a port scan being a key component. Tools like **Nmap** may be used for port scanning, integrating results into a database for deeper analysis. If **open ports such as 80, 22, and 443** are found, the scanner investigates the services on these ports, like **Apache Tomcat on ports 80 and 443**, and **OpenSSH on port 22**. It then examines these services further, such as web server specifics and SSH configurations, to pinpoint vulnerabilities for subsequent testing.

Full Scans

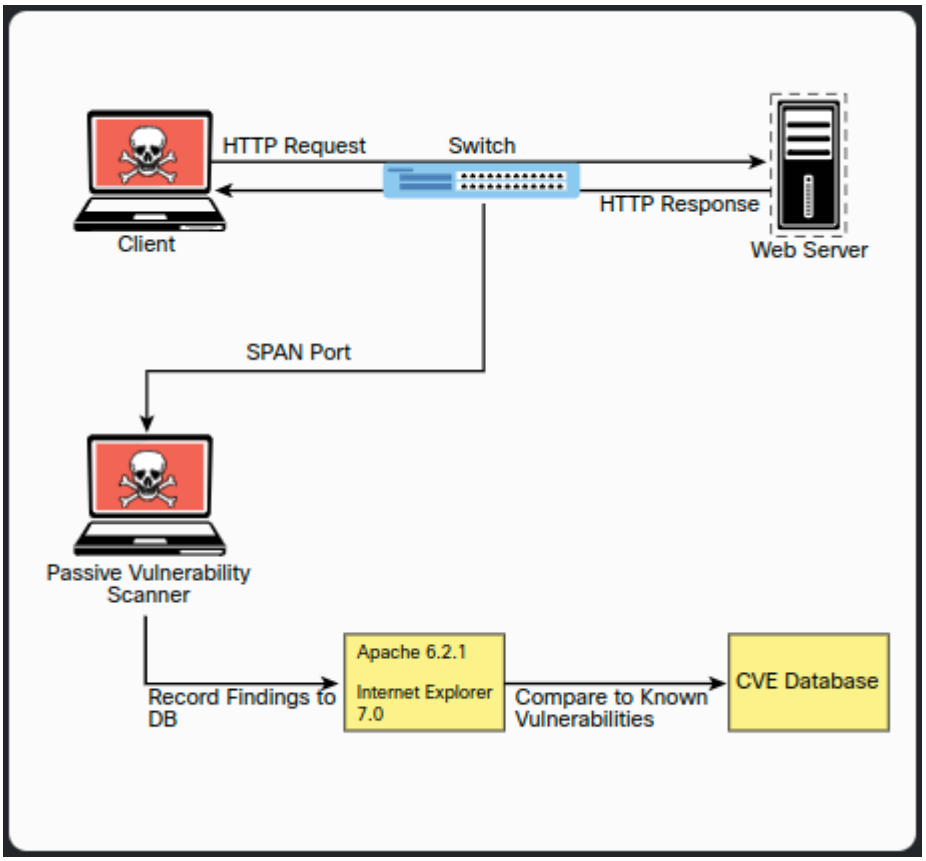
As mentioned previously, a full scan typically involves enabling every scanning option in the scan policy. The options vary based on the scanner, **but most vulnerability scanners have their categories of options defined similarly**. For instance, they are typically organized by operating system, device manufacturer, device type, protocol, compliance, and type of attack, and the rest of the options might fall into a miscellaneous category. Below the list of the plugin categories from the Nessus vulnerability scanner.

Family	Count
AIX Local Security Checks	11416
Amazon Linux Local Security Checks	1048
Backdoors	114
Brute force attack	26
CGI abuses	3841
CGI abuses : XS	666
CISCO	918
CentOS Local Security Checks	2585
DNS	172
Databases	577
Debian Local Security Checks	5532
Default Unix Accounts	168
Denial of Service	109
F5 Networks Local Security Checks	607
FTP	255
Fedora Local Security Checks	12634
Firewalls	240

Stealth Scans

There are sometimes situations in which you must scan an environment that is in a production state. **In such situations, there is typically a requirement for running a scan without alerting the defensive position of the environment**; such a scan is called a stealth scan. In this case, you will want to implement a vulnerability scanner in a manner that makes the target less likely to detect the activity. Vulnerability scanners are pretty noisy; **however, there are some options you can configure to make a scan quieter.** For example, as discussed earlier in this module, there are different types of Nmap scans, and they can be detected by network intrusion prevention systems (IPSs) or host firewalls. You have learned that a SYN scan is a fairly stealthy type of scan to run. This same concept applies to vulnerability scanners because they all use some kind of port scanner to enumerate the target. These same options are available in the vulnerability scanner's configuration. You can also disable any plugins/attacks that might be especially likely to generate noisy traffic, such as any that perform denial-of-service attacks, which would definitely arouse some concerns on the target network.

Passive Vulnerability Scanner Diagram



Compliance Scans

Compliance scans are network and application tests (scans) typically driven by the market or governance that the environment serves and regulatory compliance. An example of this would be the information security environment for a healthcare entity, which must adhere to the requirements sent forth by the Health Insurance **Portability** and **Accountability** Act (**HIPAA**). This is where a vulnerability scanner comes into play. It is possible to use a vulnerability scanner to address the specific requirements that a policy requires. Vulnerability scanners often have the capability to import a compliance policy file. This policy file can typically map to specific plugins/attacks that the scanner is able to perform. Once the policy is imported, the specific set of compliance checks can be run against a target system.

The challenge with compliance requirements is that there are many different types for different industries and government agencies, and they can all be interpreted in various ways. Some of the checks might be straightforward. If a requirement check is looking for a specific command to be run and that the output be a 1 instead of a 0, that is very simple for a vulnerability scanner to determine; however, many requirements leave more to be interpreted. This makes it very difficult for a tool like a vulnerability scanner to make a determination. Most vulnerability scanners also have the capability to create custom compliance policies. This is a valuable option for penetration testers, who typically want to fine-tune the scanner policy for each engagement.

Unauthenticated scan	The scan shows only the network services exposed to the network. The scanner attempts to enumerate the ports open on the target host.
Authenticated scan	The scan requires that a set of credentials with root-level access to the system be provided to the scanner.
Discovery scan	The scan is primarily meant to identify the attack surface of a target. Performing a port scan is a major part of this type of scan.
Full scan	The scan typically enables every scanning option in the policy.
Stealth scan	The scan requires running scanning processes without alerting the defensive position of the environment.
Compliance scan	The scan typically tests networks and applications driven by requirements from the market, governance, or regulations that serve the environment.

Vulnerability Scanning with kali Tools

Now we want to catalog the vulnerabilities that are found on various systems, networks, and applications, vulnerability scanners link discovered vulnerabilities with public vulnerability info, such as **CVE numbers**. From there, you can uncover details about the vulnerabilities and explore means of exploiting them for the next phase of the penetration test.

In this lab we will use **Nmap** and **Greenbone** Vulnerability Management (**GVM**) to scan the system to identify potential vulnerabilities.

Run a Nmap Scan on a target Computer.

Use **Nmap** and **NSE scripts** to uncover potential vulnerabilities in a target host .

Use ping to determine if the target is reachable over the network.

```
┌──(kali㉿Kali)-[~]
└─$ ping  -c5 10.6.6.23
```

-c5 option tells the ping to stop after five tries.

Identify open ports and services.

Review the results of **Nmap** scan on the host with the IP **address 10.6.6.23.**

Use ping to determine if the target is reachable over the network.

```
┌──(kali㉿Kali)-[~]
└─$ nmap  -sV 10.6.6.23
```

-sV option to determine service and version numbers of the applications.

Identify the operating system running on the target computer using **nmap -O**

```
┌──(kali㉿Kali)-[~]
└─$ sudo nmap -O 10.6.6.23
```

Use the Nmap Vulners script to scan for vulnerabilities.

The **Vulners script** knows vulnerabilities and the corresponding **CVE**. **The Vulners script uses the open port and software version information to search for common platform enumeration (CPE) names that relate to the identified service**. It then makes a request to a remote server to find out if any known vulnerabilities exist for that CPE.

Use the **nmap -script** to launch **vulners script**

```
┌──(kali㉿Kali)-[~]
└─$ nmap -sV --script vulners --script -args mincvss=4 10.6.6.23

PORT    STATE SERVICE     VERSION
21/tcp  open  ftp         vsftpd 3.0.3
| vulners:
|   cpe:/a:vsftpd:vsftpd:3.0.3:
|       PRION:CVE-2021-3618     5.8
https://vulners.com/prion/PRION:CVE-2021-3618
|_      PRION:CVE-2021-30047    5.0
https://vulners.com/prion/PRION:CVE-2021-30047
22/tcp  open  ssh         OpenSSH 7.9p1 Debian 10+deb10u2 (protocol 2.0)
vulners:
|   cpe:/a:openbsd:openssh:7.9p1:
|       PRION:CVE-2019-6111     5.8
```

-sV: Service version detection; probes open ports to determine service/application version.
--script vulners: Uses the vulners NSE script to check for vulnerabilities based on the versions of services detected. The vulners script queries the Vulners vulnerability database.
--script-args mincvss=4: Sets a script argument **(mincvss=4**) specifying that only vulnerabilities with a CVSS score of 4 or higher should be reported.
10.6.6.23: The target IP address to scan.

CVSS stands for Common Vulnerability Scoring System. It's a standardized framework for rating the severity of software vulnerabilities. **CVSS scores range from 0 to 10**, where 0 indicates no security impact and 10 represents the most critical vulnerabilities. The score is calculated based on various metrics related to the ease of exploitation, the impacts on confidentiality, integrity, and availability, and other factors. Higher CVSS scores indicate more severe vulnerabilities.

National Vulnerability Database at NIST web site:
(https://nvd.nist.gov/vuln/search)

Use GVM to scan for vulnerabilities.
GVM (**G**reenbone **V**ulnerability **M**anagement) is part of the Open Source vulnerability Management suite of products produced by **Greenbone Networks GmbH**. The GVM scanner is one of the most

widely used open-source vulnerability scanners. Unlike Nmap, **GVM uses a graphical user interface** to initiate scans and report vulnerability scan results.

Verify the GVM Product Installation.
The GVM service using the **sudo gvm-check-setup** command. This command verifies that the setup is completed correctly and the necessary files are available. The verification will flag any issues that need fixing and will provide the commands to use to fix the issues.

```
Verify the setup of the GVM service and necessary files available  with
gvm-check-setup command

┌──(kali㉿Kali)-[~]
└─$ sudo  gvm-check-setup

Stop the VGM Clean Up

┌──(kali㉿Kali)-[~]
└─$ sudo  gvm-stop
```

Open the GVM Scanner GUI

```
Start the GVM scanner using  gvm-start command

┌──(kali㉿Kali)-[~]
└─$ sudo  gvm-start
```

Challenges to Consider When Running Vulnerability Scan

The previous section have touched on a number of different things that should factor into how you perform your scanning. The sections that

follow go into further detail about some of the specific things you should consider when building a scanning policy and actually performing scans. For more information, below are challenges to consider.

Considering the Best Time to Run a Scan

The Timing of when to run a scan is typically of most concern when you are scanning a production network. If you are scanning a device in a lab environment, there is normally not much concern because a lab environment is not being used by critical applications. There are a few reasons running a scan on a production network should be done carefully. **First, the network traffic that is being generated by a vulnerability scan can and will cause a lot of noise on the network. It can also cause significant congestion, especially when your scans are traversing multiple network hops.**

Another consideration in choosing a time to run a scan is the fact that **many of the options or plugins that are performed in a vulnerability scan can and will crash the target device as well as the network infrastructure.** For this reason, you should be sure that when scanning on a production network, you are scanning at times that will have the least possible impact on end users and services. **Most of the time, scanning in the early hours of the day, when no one is using a network for critical purposes, is best.**

Determining What Protocols Are in Use

One of the **first things you need to know about a network or target device before you begin running vulnerability scans is what protocols are being used.** If a target device is using both TCP and UDP protocols for services that are running, and you only run a vulnerability scan against TCP ports, Then you are going to miss any vulnerabilities that might be found on the UDP services.

Network Topology

The network topology should always be considered when it comes to vulnerability scanning. Of course, **scanning across a WAN connection is never recommended** because it would significantly impact any of the devices along the path. **The rule of thumb when determining where in the network topology to run a vulnerability scan is that it should always be performed as close to the target as possible**. For example, if you are scanning a Windows server that is sitting inside your screened subnet (formerly known as the demilitarized zone, or DMZ), the best location for your vulnerability scanner is adjacent to the server on the screened subnet. By placing it there, you can eliminate any concerns about impacting devices that your scanner traffic is traversing.

Aside from the impact on the network infrastructure, another concern is that any device that you traverse could also affect the results of your scanner. This is mostly a concern when traversing a firewall device; in addition, other network infrastructure devices could possibly impact the results as well.

Bandwidth Limitations

Let's take a moment to consider the effects of bandwidth limitations on vulnerability scanning. Obviously, any time you flood a network with a bunch of traffic, it is going to cause an issue with the amount of bandwidth that is available. **As a penetration testing professional, you need to be cognizant of how you are affecting the bandwidth of the networks or systems you are scanning.** Specifically, depending on the amount of bandwidth you have between the scanner and the target, you might need to adjust your scanner settings to accommodate lower-bandwidth situations. If you are scanning across a VPN or WAN link that most likely has limited bandwidth, you will want to adjust your scanning options so that you are not causing bandwidth consumption

issues. The settings that need to be adjusted are typically those related to flooding and denial-of-service (DoS) type attacks.

Query Throttling

To work around the issue of bandwidth limitations and vulnerability scanning, slowing down the traffic created by your scanner can often help. This is often referred to as ***query throttling***, and it can typically be achieved by modifying the options of the scanning policy. One way to do this is to reduce the number of attack threads that are being sent to the target at the same time. There isn't a specific rule of thumb for the number of threads. It really depends on the robustness of the target. Some targets are more fragile than others. Another way to accomplish this is to reduce the scope of the plugins/attacks that the scanner is checking for. If you know that the target device is a Linux server, you can disable the attacks for other operating systems, such as Windows. Even though the attacks won't work against the Linux server, it still needs to receive and respond to the traffic. This additional traffic can cause a bottleneck in processing and network traffic consumption. Limiting the number of requests that the target would need to respond to would reduce the risk of causing issues such as crashing on the target and result in a more successful scan.

Fragile Systems / Nontraditional Assets

When using a vulnerability scanner against your internal network, you must take into consideration the devices on the network that might not be able to stand up to the traffic that is hurled at them by a vulnerability scanner. For these systems, you might need to either adjust the scanning options to reduce the risk of crashing the devices or completely exempt the specific devices from being scanned. Unfortunately, by exempting the specific devices, you reduce the overall security of the environment.

Printers are often considered "fragile systems." Historically, they have been devices that have not been able to withstand vulnerability scanning

attempts. With the surge in IoT devices, today there are many more devices that may be considered fragile, and you need to consider them when planning for vulnerability scanning. The typical way to address fragile devices is to exempt them from a scan; however, these devices can pose a risk to the environment and do need to be scanned. To address this issue, you can “throttle” the scan frequency as well as the options used in the scan policy to reduce the likelihood of crashing the device.

Understanding How to Analyze Vulnerability Scan Results

As you might already know, running a vulenreability scan is really the easy part of the information grathering and vulnerability identification process. The majority of the work goes into analyzing the results you obtain from the tools you use for vulnerability scanning. these tools are not foolproof; they can provide false positives, and the false positives need to be sorted out to determine what the actual vulnerabilities are.

For example, say that you are part of an information security team doing internal vulnerability scans on your network. You run your vulnerability scanning tool of choice and then export a report of the findings from the

scan. Next, you turn over the report to the endpoint team to address all the issues noted in the report. The endpoint team begins to address the issues one by one. Its process would likely include an investigation of an endpoint to determine how to best mitigate a finding about it. If the report that you provide includes false positives, the endpoint team will end up wasting a lot of time chasing down issues that don't actually exist. This can obviously cause some problems between the security team and the endpoint team. This scenario can also be applied to other situations.

When you are providing a report as a deliverable of a paid penetration testing assignment, it is especially important that the report be accurate. Say that you have been hired to identify vulnerabilities on a customer's network. Your deliverable is to provide a full report of security issues that need to be addressed to protect the customer's environment. Turning over a report that includes false positives will waste your customer's time and will likely result in your losing the customer's repeat business. As you can see from this discussion, reducing the false positives in vulnerability scans is very important.

So how do you go about eliminating false positives? The process involves a detailed and thorough look into the results that your vulnerability scanning tool has provided. Suppose that the results of a scan reveal that there is a possible remote code execution vulnerability in the Apache web server that is running on the target server. This type of finding is likely to be flagged as a high-severity vulnerability, and it should therefore be prioritized. To determine if this is a valid finding, you would first want to take a look at what the vulnerability scanner did to come to this conclusion. Did it pull the version information directly from the system by using a credentialed scan, or was it determined by remotely connecting to the port? As you know, the results of a credentialed scan are more likely than remote analysis to be valid.

Because the method of harvesting version information varies based on the scanner and the service, you should be able to take a look at the details of the findings in a report to determine how the information was gathered. From there, if possible, you would want to connect directly to

the target that is reporting a particular vulnerability and try to manually determine the version information of that service. Once you validate that the version reported by the scanner does actually match what is on the system, you also need to dig deeper into the details of the vulnerability. Each vulnerability will typically map to one or many items in the Common Vulnerabilities and Exposures (CVE) list. You need to take a look at the particulars of those CVE items to understand the criteria because a vulnerability may be flagged based on only one piece of information (such as the version number of the Apache server pulled from the banner). When you dig into the CVE details, you might find that for the vulnerability to be exploitable, this version of Apache must be running on a specific version or distribution of Linux. Most vulnerability scanners are able to correlate multiple pieces of information to make the determination. However, some Linux operating systems, such as Red Hat, report an older version of a service that has actually been patched for the specific vulnerability. This is called backporting. So, as you can see, there is more to it than just running a scan. Of course, the number-one method of validating a finding from a vulnerability scan is to exploit the vulnerability, as discussed in many of the upcoming modules.

Source for Further Investigation of Vulnerabilities

US-CERT

The U.S. **Computer Emergency Readiness Team** (US-CERT) was established to **protect the Internet infrastructure of the United States**. The main goal of US-CERT is to work with **public**- and **private**-sector agencies to increase the efficiency of vulnerability data sharing. The work done by US-CERT is meant to improve the nation's cybersecurity posture. US-CERT operates as an entity under the Department of Homeland Security as part of the National Cybersecurity and Communications Integration Center (NCCIC). You can access US-CERT resources by visiting https://www.us-cert.gov.

The CERT Division of Carnegie Mellon University

The CERT Division of the Software Engineering Institute of Carnegie Mellon University is a cybersecurity center whose experts **help coordinate vulnerability disclosures across the industry.** CERT researches security vulnerabilities and contributes to many different cybersecurity efforts in the industry. CERT also **develops and delivers training to many organizations to help them improve their cybersecurity practices and programs.** You can obtain additional information about CERT at https://cert.org.

NIST

The **National Institute of Standards and Technology** (NIST) is an agency of the U.S. **Department of Commerce**. **Its core focus is to promote innovation and industrial competitiveness.** NIST is responsible for the creation of the NIST Cybersecurity Framework (NIST CSF; see https://www.nist.gov/cyberframework). This framework includes a policy on computer security guidance. Version 1 of the NIST framework was published in 2014 for the purpose of guiding the security of critical infrastructure; however, it is commonly used by private industry for guidance in risk management. In 2018, NIST released version 1.1, which is designed to assist organizations in assessing the risks they encounter. In general, the framework outlines the standards and industry best practices that can be used to improve organizations' cybersecurity posture. Anyone who is responsible for making decisions related to cybersecurity in an organization should consult this framework for guidance on standards and best practices.

JPCERT

Similar to the US-CERT, the **Japan Computer Emergency Response Team** (JPCERT) is an organization that works with service providers, security vendors, and **private-sector** and **government agencies** to provide incident response capabilities, increase cybersecurity awareness, conduct research and analysis of security incidents, and work with other international CERT teams. The JPCERT is responsible for Computer Security Incident Response Team (CSIRT) **activities in the Japanese and Asia Pacific region**. You can access JP-CERT resources by visiting https://www.jpcert.or.jp/english/.

CAPEC

The **Common Attack Pattern Enumeration and Classification** (CAPEC) is a **community-driven effort** to catalog the attack patterns **seen in the wild** so that they can be used to more efficiently identify active threats. CAPEC, which is maintained by MITRE, acts as a dictionary of known **attacks that have been seen in the real world**.

CVE

Common Vulnerabilities and Exposures (CVE) is an effort that reaches across **international cybersecurity communities.** It was created in 1999 with the idea of consolidating cybersecurity tools and databases. A **CVE ID is composed of the letters CVE** followed by the year of publication and four or more digits in the sequence number portion of the ID (for example, **CVE-YYYY-NNNN** with four digits in the sequence number, CVE-YYYY-NNNNN with five digits in the sequence number, CVE-YYYY-NNNNNNN with seven digits in the sequence number, and so on). You can obtain additional information about CVE at https://cve.mitre.org.

CWE

Common Weakness Enumeration (CWE), at a **high level, is a list of software weaknesses**. The purpose of **CWE is to create a common language to describe software security weaknesses** that are the root causes of given vulnerabilities. CWE provides a common baseline for weakness identification to aid the mitigation process. You can obtain additional information about CWE at MITRE's site: https://cwe.mitre.org.

CVSS

Vulnerabilities pose risks to systems and networks, with their severity quantified by the **Common Vulnerability Scoring System** (CVSS), a standard maintained by FIRST. CVSS scores, **ranging from 0 to 10,** are

based on three components: **base**, **temporal**, and **environmental**. The base score reflects a vulnerability's inherent characteristics, the temporal score its changes over time, and the environmental score its impact within a specific organizational context. The base score, mandatory for a vulnerability rating, considers exploitability, impact on confidentiality, integrity, and availability, and any scope change affecting other systems. CVSS aids in understanding and prioritizing vulnerabilities based on their potential impact.FIRST provides additional examples at https://www.first.org/cvss/.

Investigate Vulnerability Information Sources

CVEs vs CWE

Waht is the difference between a CVE and a CWE ?

CVEs are about specific vulnerabilities;
CWEs are about the categories of weaknesses that lead to vulnerabilities.

What is the relationship between **CVE**, **CWE**, **NVD**, and **CVSS** ?

CVE lists vulnerabilities that have been discovered.
CWE classifies these vulnerabilities.
NVD provides details.
CVSS provides severity ratings.

How to Deal with a Vulnerability

A penetration tester, your goal is to identify weaknesses that can be exploited. As previously discussed, vulnerability scanning is a method of identifying potential exploits. After you identify a vulnerability, you need

to verify it.There are many ways to determine if a vulnerability scanner's findings are valid. The ultimate validation is exploitation.

To determine if a vulnerability is exploitable, you need to first identify an exploit for a vulnerability. Suppose your vulnerability scanner reports that there is an outdated version of Apache Struts that is vulnerable to a remotely exploitable unauthenticated defect. One of the first things you would want to do is to determine if there is a readily available exploit. Many times, this can be found with an exploitation framework such as

Metasploit. As a general rule, if a vulnerability has a matching module in Metasploit, it should almost always be considered high severity.That being said, there are also other methods for finding exploits, and you can always **write your own exploits**.

How do you prioritize your findings for the next phase of your penetration test ? To determine the priority, you need to answer a few question :

- What is the **severity** of the vulnerability ?
- How many **systems** does the vulnerability apply to ?
- How was the **vulnerability detected** ?
- Was the vulnerability found with an **automated scanner** or **manuall**y ?
- What is the value of the **device** on which the vulnerability was found ?
- Is this **device critical** to your **business** or **infrastructure** ?
- What is the **attack vector**, and does it apply to your **environment** ?
- Is there a **possible workaround** or **mitigation available** ?

Answering these questions can help you determine the priority you should assign to the vulnerabilities found. Standard protocol would have you start with the highest-severity vulnerabilities that have the greatest likelihood of being exploited. If these vulnerabilities are actually valid,

they might already be compromised. (If at any time during a penetration test, you find that a system is being actively exploited, you should report it right away to the system owner.) Next, you should address any vulnerabilities that are on critical systems, regardless of the severity level. It is possible that there might be an exploit chain available to an attacker that would allow a lower-severity vulnerability to become critical. You need to protect critical systems first. Next, you might want to prioritize based on how many systems are affected by the finding.

If a large number of systems are affected, then this would raise the priority because many exploits on this vulnerability would have a higher impact on your environment. These are suggested guidelines, but when it comes to prioritization of vulnerability management and mitigation, it really depends on the specific environment.