

Faculty of Sciences and Techniques
Department of Computer Science
IT Security

Hacker House Booklab

Engineering Student:
Maman Aboubakar Moïse

October 6, 2026

Contents

1 Introduction **3**

2 Hacking Bussiness **3**

3 Hacking Ethically and Legally **4**
3.1 Penetration Testing Methodology 4
3.2 Authorization 4
3.3 Responsible Disclosure 5
3.4 Bug Bounty Programs 5

4 Building the Hack Box **5**
4.1 Verifying download 5
4.2 Disk Encryption 6
4.3 Essentials Software 6

5 Setting up VirtualBox **7**

6 Open Source Intelligence Gathering **8**
6.1 Google Dorking the Shadows 8
6.1.1 A brief introduction to passwd and shadow files 8
6.2 The Google Hacking Database 10
6.3 OSINT Framework Recon-ng 11
6.3.1 Recon-ng under the hood 14
6.4 Harvesting the Web 15
6.5 Document Metadata 16
6.6 Maltego 19
6.7 Social Media Network 19
6.8 Shodan 20
6.9 Protecting Against OSINT 21

7 The Domain Name System **21**
7.1 The Implications of Hacking DNS 21
7.2 DNS hierarchy 21
7.3 Basic Query 22
7.4 Authority and Zone 24
7.5 DNS Resource Records 24
7.6 Bind9 25
7.7 DNS Hacking Toolkit 27
7.8 Finding Host 28
7.9 Finding the SOA with Dig 30
7.10 Port Scanning with Nmap 31
7.11 Digging for Information 33
7.12 Information Leak CHAOS 36
7.13 Zone Transfer Requests 37
7.14 Information-Gathering Tools 38
7.15 Searching for Vulnerabilities and Exploits 39

7.16 DNS Traffic Amplification 39

8 In progress... 39

1 Introduction

This document is the report of my comprehension of some topics and all the hands-on activities that i have done through the book **"Hands on Hacking - Become an Expert at Next Gen Penetration Testing and Purple Teaming"**. This book really give you the essentials to understand how some systems work and start hacking them in a safe environment. This book is the one which influenced me to have this desire to become a purple team profile. I really enjoyed reading this book. Thanks to **Matthew Hickey, Jennifer Arcuri, the Hacker House Team and anyone that has contributed** for making this book.

2 Hacking Bussiness

Data is now the most valuable thing in this word, unfortunately we need to face this reality of that all computer are broke. Now the internet has became so wide specially with this era of IoT, which means that even a fridge are connected to the internet. When we talk about information security there is two traditional type of team.

Blue Team

It is a group of security professionals who try to protect asset (anything that an organization need to protect) of an organization against internal or external threats and ensure that security measures are effective within a organization.

Red Team

A red team is an independent group of security professionals who try to mimics external or internal threat and attack systems of organizations using advanced tactic,technique and procedure, however we need to note that they have the authorization to perform that.

The problem here is when a company paid the red team for the service after finding vulnerabilities and giving instruction to mitigate them, it can lead to misunderstanding when the organization blue team don't have all the skills necessary to understand the report content, maybe the company don't have enough money to pay a red team so they can only search for the service of a penetration tester (a security professional who can perform a security assessment on a target system). Companies want effective and inexpensive cybersecurity to protect their systems. This is why the purple teaming is the solution.

Purple Teaming

The purple team is the fusion of the red and blue team to become one single team. Which mean that those security experts have both this defensive and offensive background ,they have necessary skills of understanding in information technology (IT), software development lifecycles, social engineering, penetration testing, vulnerability management, patching, system configuration, and hardening to standards . They can perform a security assessment and in the same time implement the solution to mitigate, fix or patch vulnerabilities found.

3 Hacking Ethically and Legally

Most of the time the word hacker is assigned to those who break into system for illegal activities. But we need to known that term is neutral. Any ethical hacker must avoid breaking laws. Before taking any actions we need to have an understanding of laws where we live or where the system which we are going to test is. For example in America we have laws like the **Computer Fraud and Abuse Act, Data Protection Act and in Europe we have the example of the RGPD**. We have to always get written permission from the system, network, or environment owner when we are planning a test and ideally when you intend to perform scanning activities.

3.1 Penetration Testing Methodology

Performing a penetration testing involve using two ways the first one is using methodologies which give a set of steps to follow along. As exemple we have :

- Penetration Testing Execution Standard which give guidelines from the interaction with the client to the final report. PTES gives specially technical task to follow.
- Open Source Security Testing Methodology Manual
- Open Source Wide Application security Project Testing Framework which gives guideline to perform security assessment specially on web application.
- MITRE Adversarial Tactics, Techniques, and Common Knowledge (ATT&CK) define operation used by APT groups.

The other way involve thinking Out of the Box and making your own path . Both can also be used on a penetration testing.

3.2 Authorization

It is imperative to have a written authorization from the client before attempting a penetration test on his system. Both the client and the penetration tester must agree on what will be done, what is the scope , the objectives , what kind of communication process will be set and so on. Everything must be clear to both, no place to ambiguity.

3.3 Responsible Disclosure

Responsible disclosure is the practice of first informing and then working with product vendors to resolve a vulnerability. A zero day vulnerability is vulnerability which is not yet patched or even known by the product vendor or publicly known. When founding this type of flaw we need to use a secure communication canal while submitting that to the client. For that we can use the PGP to encrypted mail. This encryption mechanism is already implemented on thunderbird and can easily be activated. The vendor can also refuse to take into consideration the vulnerability even though you have submitted it to him, in this case you need to make it public to force them to take it seriously.

3.4 Bug Bounty Programs

Bug Bounty programs are programs where vendors or enterprises gives the opportunity to security professionals or anyone to find vulnerabilities legally with their permission and giving money reward.

4 Building the Hack Box

The minimal requirement for this lab are :

- A modern AMD or Intel CPU
- 4 GB of RAM
- 50–100 GB of hard-disk drive (HDD) or solid-state drive (SSD) capacity
- Internet access for downloading software and running certain demonstrations

Linux is the best route to go for security testing, so it's is recommended to have a linux distribution as a daily driver to get used to this ecosystem. The cool fact is i'm already using linux distribution as a daily driver, i have chosen fedora because i have read a lot of reddit publication on it, also because it make the balance between stability and getting the latest kernel and technology, fun fact linus torvalds also use fedora by the way. There is a bunch of linux distribution each one is made for specific need. Some distribution are define as being stable other one are rolling release distribution which means they get the latest packages even if it can lead to bug on the system you are like a beta tester for this kind of distro if i can say that.

4.1 Verifying download

Any program that you downloaded through the web should be verified. This help to check the integrity of the program. Now most of web page give you the hash of the program so that when you download the program you can compare it hash with the one given . Now let try this with the sha256 hash of our virtualbox rpm package that we have downloaded.



Figure 1

Let try to generate the hash from the downloaded package.

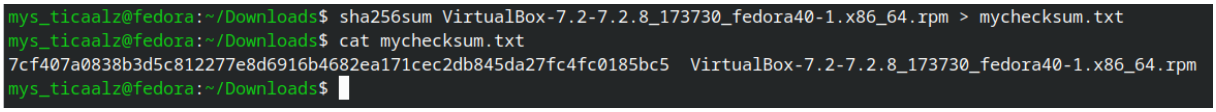


Figure 2

We will use the diff command which compare content of two file and try to found differences if no differences was found nothing will show up. As shown below our package was not altered while downloading which mean our packet is identical and we can use it for our purpose.

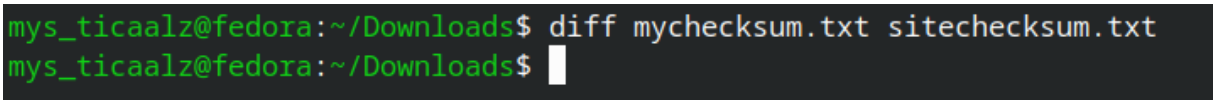


Figure 3

4.2 Disk Encryption

Disk encryption can now be enabled on linux while installing the operating system, for windows we need to use the bitlocker. So even when your machine get stolen or can be access physically, they cannot access the content easily because it is encrypted. The only way to decrypt it to use the passphrase given to you while encrypting or the key file. You need to store those information in self place.

4.3 Essentials Software

There is some essentials software to have such as :

- Firewall : it set rule for the outgoing or ongoing network traffic. By default ubuntu came with the ufw (Un-complicated firewall), for fedora it is the firewalld. There is other firewall that you can install like iptables or it new version nftables.
- PasswordManager : it help to manage all your passwords used. the recommendation is to use KeepassX for a local solution and for an online solution there is 1Password.
- Email : For communication that involves sensitive data it better to use a email client that implement encryption mechanism like the PGP for that thunderbird make the job.

5 Setting up VirtualBox

We will create a host-only network named vboxnet1 to which we will connect the machine's adapters so that they can communicate together and with the host also, is like they are connected to the same switch. The dhcp server enabled will give to any machine connected an ip address between 192.168.57.101 to 192.168.57.254.

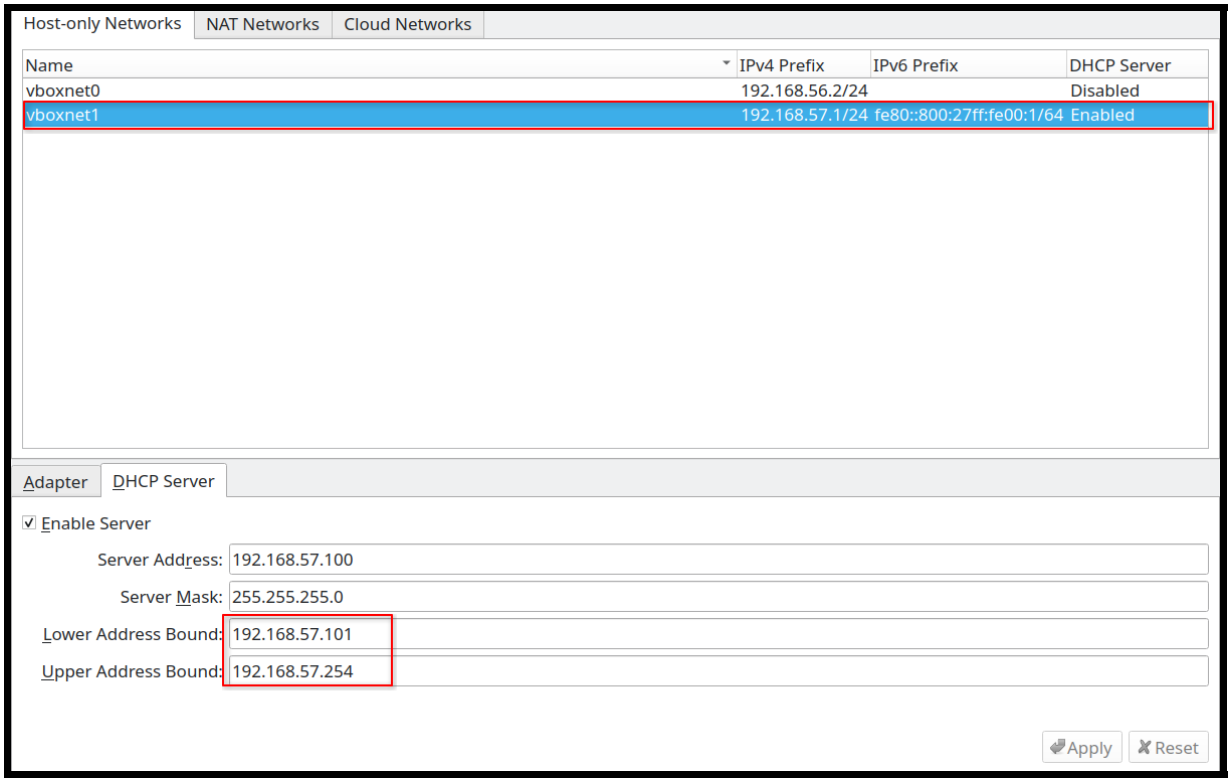


Figure 4

For this lab we will have a kali linux machine with the hacker house mail server and also the booklab machine .

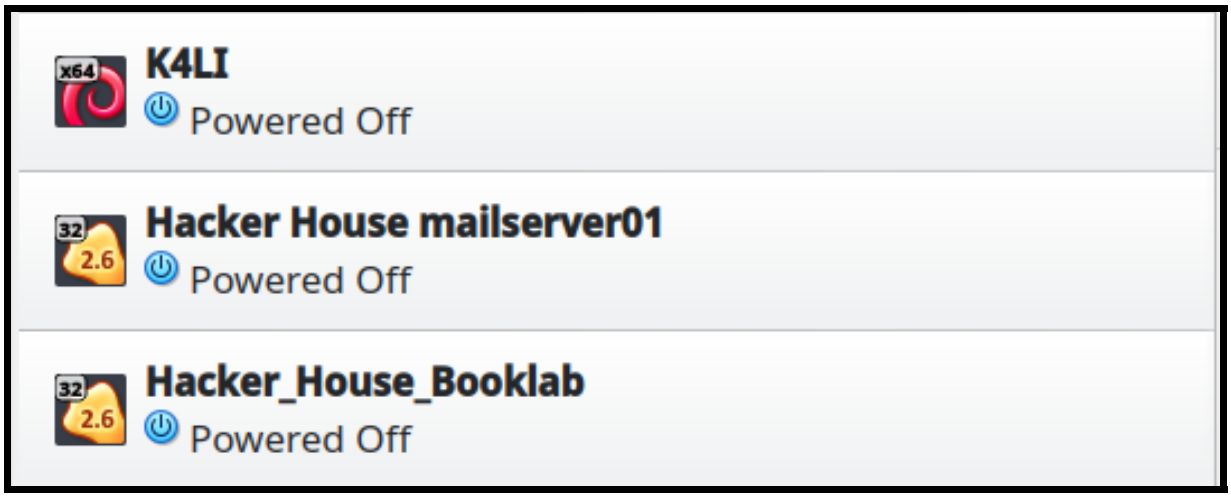


Figure 5

6 Open Source Intelligence Gathering

Open source intelligence is the process of getting valuable information through public data resources, this is known as a passive attack or passive reconnaissance because we are not interacting directly with the target. We should not limit your research to only one public database or search engine. The type or piece of information to look for are :

- Usernames, profile names and/or email addresses
- Passwords (private keys, PINs, and so forth)
- Domain names
- Hostnames
- IP addresses (external and internal)
- Software and operating system types, names, and versions
- Technical documentation, such as user guides for systems

6.1 Google Dorking the Shadows

Google Dorking is using advanced search queries on Google search engine to retrieve data from it's database. This can be used to identify vulnerable site and data. As exemple we can use the following dorks: **inurl:/etc/passwd root:x:0:0:root:/root:/bin/bash**. This query will ask the search engine to show pages that contain on their url the string "/etc/passwd" along with the "root:x:0:0:root:/root:/bin/bash" as part of the page content.

6.1.1 A brief introduction to passwd and shadow files

On unix and linux system users credentials are stored in passwd and shadow files. Those file are world readable but only writable by the root. The figure below is an example of the content of a passwd file each field of a line is separated by ":".

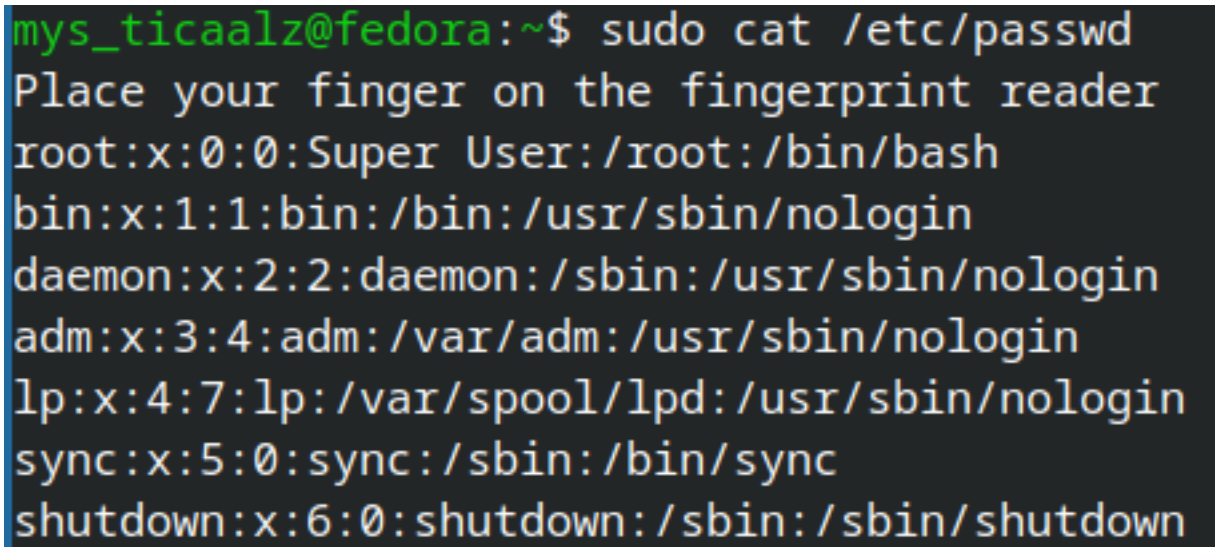


Figure 6

Each line represent a user, not all are human user because the operating system can also create users for services. The composition of a line follows this structure :

username:password_field:userid:groupid:description:personal_directory:shell_type

The shadow file will contain the stored of passwords set for user and other information related to the password like the amount of day remaining before changing it or the number of time that you can the the password.

```
mys_ticaalz@fedora:~$ sudo cat /etc/shadow
Place your finger on the fingerprint reader
root:[redacted]7:::::::
bin:!*:20294::::::
daemon:!*:20294::::::
adm:!*:20294::::::
lp:!*:20294::::::
sync:!*:20294::::::
shutdown:!*:20294::::::
halt:!*:20294::::::
```

Figure 7

After using the dorks given as example we have found an interesting result as show below which was a vulnerable site. But clicking on it is considered as illegal so we haven't done that. I also try to contact this company but no real reponse from them.

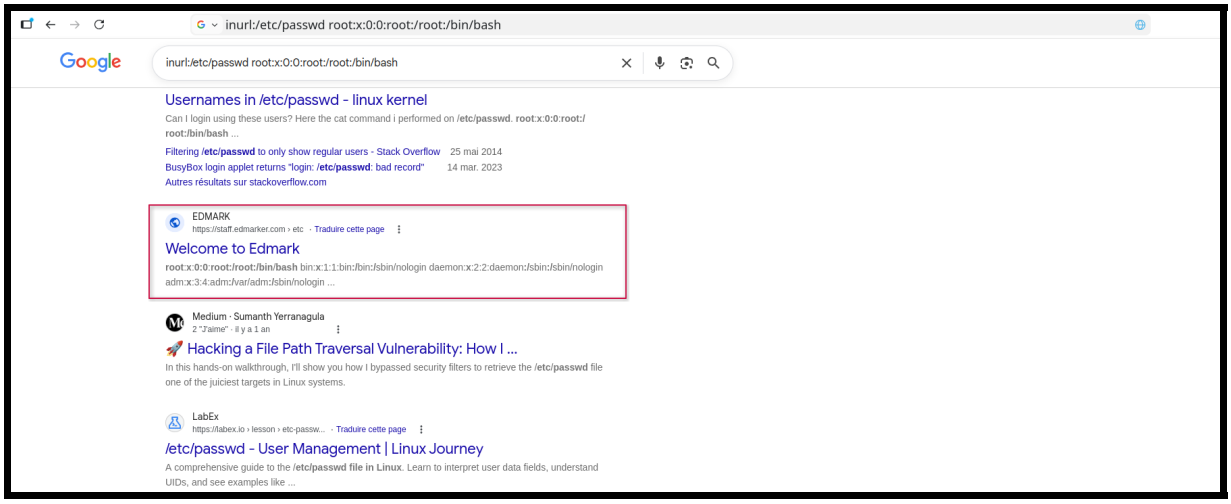


Figure 8

The following query will search for doc files of web pages that contain the text "default password" and which url contain the word "gov".

filetype:doc inurl:gov intext:"default password"

6.2 The Google Hacking Database

The google hacking database is a section of the exploit-db.com website that content all type of dorking for special cases.

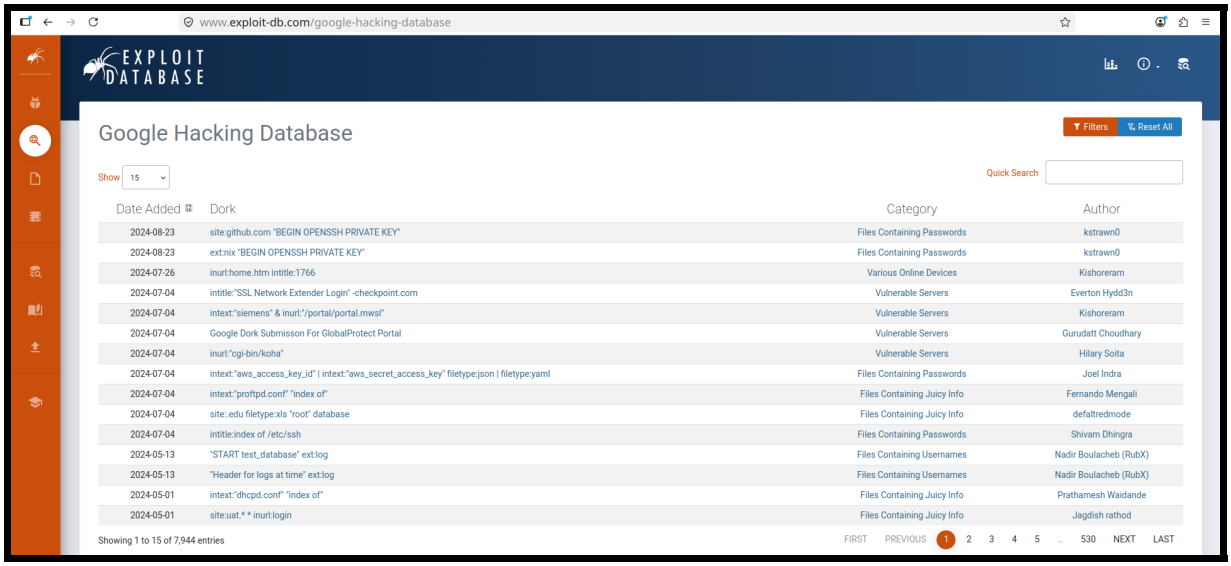


Figure 9

Another example of dork is **inurl:"q=user/password"**, this one search for url that contain the input name q with the value user/password. For website that match this dork it means that they use drupal as a content management system which is a very valuable information.

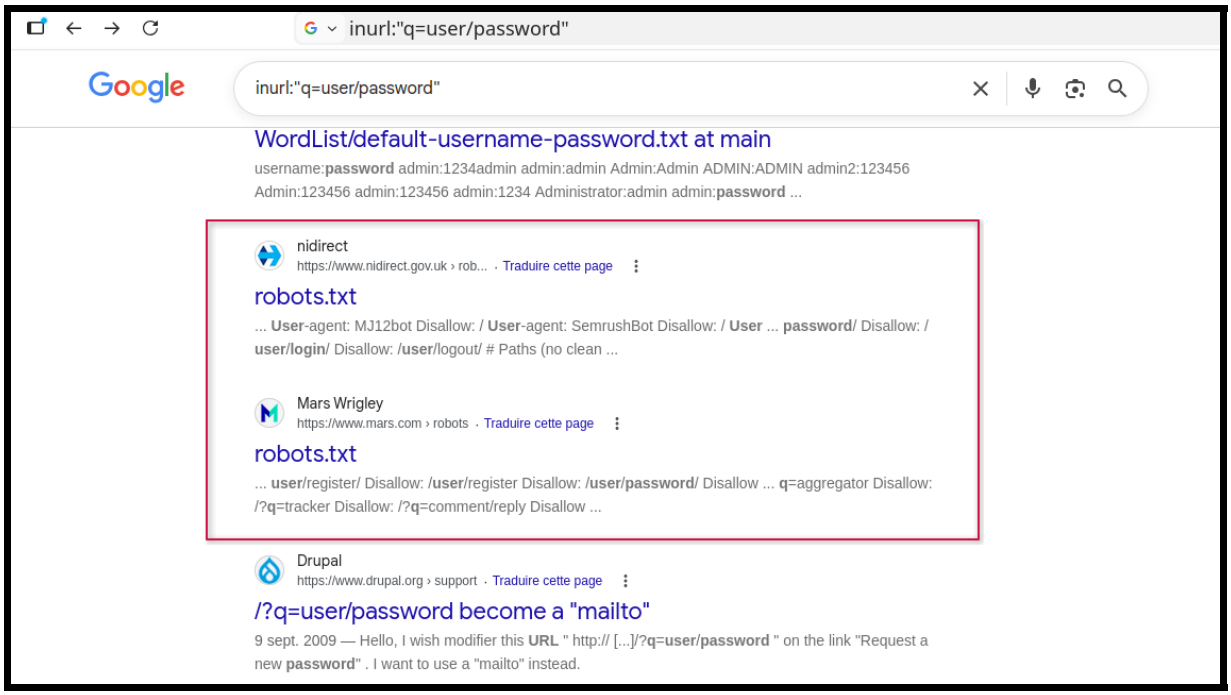


Figure 10

6.3 OSINT Framework Recon-ng

Some task of the information gathering process can be very repetitive, to automate them a security professional named **LaNMaSteR53** developed recong-ng. This tool helps to automate some aspects of reconnaissance through a command line. Recon-ng uses workspace to organize results for various use case. It can use API also to access other systems services. This tool is preinstalled on Kali and can be lunch by simply typing **recon-ng** from the terminal.
After launching it we can us **help** as command to list all the available commands.

```
[recon-ng][default] > help

Commands (type [help|?] <topic>):
--------------------------------
back            Exits the current context
dashboard       Displays a summary of activity
db              Interfaces with the workspace's database
exit            Exits the framework
help            Displays this menu
index           Creates a module index (dev only)
keys            Manages third party resource credentials
marketplace     Interfaces with the module marketplace
modules         Interfaces with installed modules
options         Manages the current context options
pdb             Starts a Python Debugger session (dev only)
script          Records and executes command scripts
shell           Executes shell commands
show            Shows various framework items
snapshots       Manages workspace snapshots
spool           Spools output to a file
workspaces      Manages workspaces

[recon-ng][default] >
```

Figure 11

To have more details about a special command we use **?** <**command_name**>

```
[recon-ng][default] > ? workspaces
Manages workspaces

Usage: workspaces <create|list|load|remove> [...]

[recon-ng][default] >
```

Figure 12

Now we will create a new workspace called hackerhouse using the following command:

workspaces create hackerhouse

If we want to show the list of all the workspaces created we use the option **list**.

```
[recon-ng][hackerhouse] > workspaces list

  +-------------------------------------+
  |  Workspaces |       Modified        |
  +-------------------------------------+
  | cisco       | 2026-02-16 18:31:17 |
  | default     | 2026-02-19 12:11:08 |
  | hackerhouse | 2026-05-23 11:24:47 |
  +-------------------------------------+

[recon-ng][hackerhouse] >
```

Figure 13

Recon-ng has it own marketplace where you can find module to use based on a special need. All things related to the marketplace can be done using the **marketplace** command. We will install all the module using the command **marketplace install all** as show below.

```
[recon-ng][hackerhouse] > marketplace install all
[*] Module installed: discovery/info_disclosure/cache_snoop
[*] Module installed: discovery/info_disclosure/interesting_files
[*] Module installed: exploitation/injection/command_injector
[*] Module installed: exploitation/injection/xpath_bruter
[*] Module installed: import/csv_file
[*] Module installed: import/list
[*] Module installed: import/masscan
[*] Module installed: import/nmap
[*] Module installed: recon/companies-contacts/bing_linkedin_cache
[*] Module installed: recon/companies-contacts/censys_email_address
```

Figure 14

Listing all the modules installed require to write the command **module search** as shown below, but when it is preceded by a word it will try to find module that match it.

```
[recon-ng][hackerhouse] > modules search

  Discovery
  ---------
    discovery/info_disclosure/cache_snoop
    discovery/info_disclosure/interesting_files

  Exploitation
  ------------
    exploitation/injection/command_injector
    exploitation/injection/xpath_bruter

  Import
  ------
    import/csv_file
    import/list
```

Figure 15

To select an installed module we use the command **module load** followed by the module name or the full path to that module. Let's take a look to the profiler module and use it. To show information about the selected module we use the command **info**.This module search on the web website where the given usernames are used and store them into the recon-ng database.

```
[recon-ng][hackerhouse] > modules load recon/profiles-profiles/profiler
[recon-ng][hackerhouse][profiler] > info

      Name: OSINT HUMINT Profile Collector
    Author: Micah Hoffman (@WebBreacher), Brendan Burke (@gbinv)
   Version: 1.2

Description:
  Takes each username from the profiles table and searches a variety of web sites for those users. The
  list of valid sites comes from the parent project at https://github.com/WebBreacher/WhatsMyName

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

Source Options:
  default        SELECT DISTINCT username FROM profiles WHERE username 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

Comments:
  * Note: The global timeout option may need to be increased to support slower sites.
  * Warning: Using this module behind a filtering proxy may cause false negatives as some of these
  sites may be blocked.
```

Figure 16

The default SOURCE option indicates that the module will use as input usernames from a table named profiles. But we can change it to specify a single input like one username or a path to file that contain usernames. We will use the username **omarsantos** and insert it on that table. For that we will use the following command: **db insert profiles omarsantos~~~~**. We added 4 tildes to specify that the 4 more columns after the username are empty so we don't get errors. The 2 others columns (rowid,module) have been filled automatically. We can add also full email or part of an email (without the @ and the domain name) because they work also well there. To show off the content of a table we use the command **show** followed by the name of that table.

```
[recon-ng][hackerhouse][profiler] > db insert profiles omarsantos~~~~
[*] 1 rows affected.
[recon-ng][hackerhouse][profiler] > show profiles

  +-------------------------------------------------------------------------------+
  | rowid | username  | resource | url | category | notes |    module    |
  +-------------------------------------------------------------------------------+
  | 1     | omarsantos |          |     |          |       | user_defined |
  +-------------------------------------------------------------------------------+

[*] 1 rows returned
[recon-ng][hackerhouse][profiler] >
```

Figure 17

Now we can run the process using simply the **run** command and wait after it finish to retrieve the information puts in that database table.

```
SUMMARY

[*] 88 total (88 new) profiles found.
[recon-ng][hackerhouse][profiler] > show profiles

+-----------------------------------------------------------------------------------------------------------------------------------------------+
| rowid | username  |         resource         |                                                 url                                                 |  category  | notes |  module  |
+-----------------------------------------------------------------------------------------------------------------------------------------------+
| 2     | omarsantos | about.me                | https://about.me/omarsantos                                                                         | social     |       | profiler |
| 3     | omarsantos | ADVFN                   | https://uk.advfn.com/forum/profile/omarsantos                                                       | finance    |       | profiler |
| 4     | omarsantos | Bandlab                 | https://www.bandlab.com/omarsantos                                                                  | music      |       | profiler |
| 5     | omarsantos | Arduino (Project Hub)   | https://projecthub.arduino.cc/omarsantos                                                            | tech       |       | profiler |
| 6     | omarsantos | Bandcamp                | https://bandcamp.com/omarsantos                                                                     | music      |       | profiler |
| 7     | omarsantos | Bio Sites               | https://bio.site/omarsantos                                                                         | social     |       | profiler |
| 8     | omarsantos | Blogspot                | http://omarsantos.blogspot.com                                                                      | blog       |       | profiler |
| 9     | omarsantos | Bluesky 2               | https://bsky.app/profile/omarsantos.bsky.social                                                     | social     |       | profiler |
| 10    | omarsantos | Calendy                 | https://calendly.com/omarsantos                                                                     |            |       |          |
```

Figure 18

6.3.1 Recon-ng under the hood

Recon-ng uses a SQLite Databases to store data for each workspace. Those databases are found in the personal directory specially in:

.recon/workspaces/workspace_name/data.db

We can use the sqlitebrowser command to take a look to that database, see the schema of it, explore the data contained within or also export it.

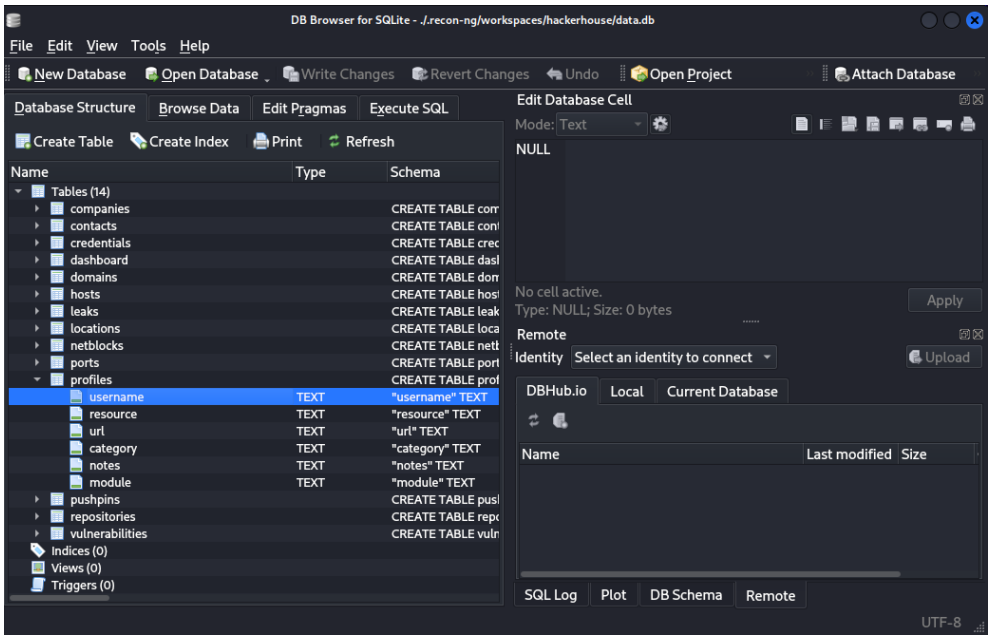


Figure 19

6.4 Harvesting the Web

There is a python program which come preinstalled on kali called **theHarverster**. This tools harvest the web trough a submitted search engine to find subdomains, ips, emails all related to a given domain name. Let's try it, for our example below the tools will search for data related to the domain name google.com using duckduckgo as search engine and limit the finding results to 50.

```
┌──(kali㉿kali)-[~]
└─$ theHarvester -d google.com -b duckduckgo -l 50
Read proxies.yaml from /etc/theHarvester/proxies.yaml
*******************************************************************
*  _   _                                            _             *
* | |_| |__    ___     ^  /\__ _ _ ____   _____  ___| |_ ___ _ __  *
* | __|  _ \ / _ \   / /_/ / _` | '__\ \ / / _ \/ __| __/ _ \ '__| *
* | |_| | | |  __/  / __  / (_| | |   \ V /  __/\__ \ ||  __/ |    *
*  \__|_| |_|\___| v /_/ \__,_|_|    \_/ \___||___/\__\___|_|    *
*                                                                 *
* theHarvester 4.10.1                                             *
* Coded by Christian Martorella                                   *
* Edge-Security Research                                          *
* cmartorella@edge-security.com                                   *
*                                                                 *
*******************************************************************

[*] Target: google.com

[*] Searching Duckduckgo.

[*] No IPs found.

[*] No emails found.

[*] No people found.

[*] Hosts found: 12
---------------------
2Fencrypted.google.com
2Fpatents.google.com
2Fsupport.google.com
Encrypted.google.com
accounts.google.com
encrypted.google.com
```

Figure 20

6.5 Document Metadata

Entreprise or Organization often publish documents like annual reports, brochures and so on through their website. Those documents can be images, pdf files, Microsoft Word documents or Excel spreadsheet.Unfortunately, those documents can leak useful information contained on their metadata such as usernames, email addresses, date and time information, and Exchangeable Image File Format (Exif) data which is usually associated with photographs and can contain camera details, GPS coordinates (the location where the photo was taken), as well as software type and version information. Those data are not part of the document body that why most of the time people are not aware of them.

Note

Downloading those documents from the client website is mostly considered as an active reconnaissance because you are interacting with the client network.

There is a tool called Metagoofil which is used to download documents from a given domain name and to analyze their metadata. This tool uses Google to search for documents, and it automatically downloads them to a folder before extracting their metadata. So let's try it, for our example below we are gonna use it to check pdf and Microsoft word documents related to redhat while specifying some limits and the directory where the downloaded documents will be.

metagoofil -d redhat.com -t pdf,doc -l 200 -n 50 -o redhatfiles

```
┌──(kali㉿kali)-[~]
└─$ metagoofil -d redhat.com -t doc,pdf -l 200 -n 50 -o redhatfiles
[*] Searching for 200 .doc files and waiting 30.0 seconds between searches
[*] Results: 0 .doc files found
[*] Searching for 200 .pdf files and waiting 30.0 seconds between searches
[*] Results: 0 .pdf files found
[+] Done!

┌──(kali㉿kali)-[~]
└─$
```

Figure 21

Metagoofil is an old tool so we don't get great result with it but there is a version more up to date on github at this link .

Reminder

Always try to check the source code of tools from unverified repository before using them. If you don't know or you are not sure of what it does, run it in a virtual environment or just don't run it at all !

Some metadata extraction tools can usually be found in Linux distribution's package repository. An example is Exiftool, which can be run on any JPEG images to extract Exif data. For our example we will download a image from the hacker house website using this link. After giving it as argument to exiftool we get the result below.

exiftool image1.jpeg

```
┌──(kali㉿kali)-[~/Hands_On_Hacking]
└─$ exiftool image1.jpeg
ExifTool Version Number         : 13.50
File Name                       : image1.jpeg
Directory                       : .
File Size                       : 2.7 MB
File Modification Date/Time     : 2026:06:18 21:32:00-04:00
File Access Date/Time           : 2026:06:18 21:32:00-04:00
File Inode Change Date/Time     : 2026:06:18 21:32:00-04:00
File Permissions                : -rw-rw-r--
File Type                       : JPEG
File Type Extension             : jpg
MIME Type                       : image/jpeg
JFIF Version                    : 1.01
Exif Byte Order                 : Big-endian (Motorola, MM)
Make                            : Apple
Camera Model Name               : iPhone 6
X Resolution                    : 72
Y Resolution                    : 72
Resolution Unit                 : inches
Software                        : 11.3.1
Modify Date                     : 2018:09:27 11:21:35
Exposure Time                   : 1/898
F Number                        : 2.2
Exposure Program                : Program AE
ISO                             : 32
Exif Version                    : 0221
Date/Time Original              : 2018:09:27 11:21:35
Create Date                     : 2018:09:27 11:21:35
Components Configuration        : Y, Cb, Cr, -
Shutter Speed Value             : 1/898
Aperture Value                  : 2.2
Brightness Value                : 9.196232339
Exposure Compensation           : 0
Metering Mode                   : Multi-segment
```

Figure 22

```
Flash                           : Off, Did not fire
Focal Length                    : 4.2 mm
Subject Area                    : 1631 1223 1795 1077
Maker Note Version              : 9
Run Time Flags                  : Valid
Run Time Value                  : 30645718499916
Run Time Scale                  : 1000000000
Run Time Epoch                  : 0
AE Stable                       : Yes
AE Target                       : 153
AE Average                      : 166
AF Stable                       : Yes
Acceleration Vector             : -0.9815188172 0.03043882641 -0.1194911988
HDR Image Type                  : Unknown (2)
Image Capture Type              : Unknown (3)
Live Photo Video Index          : 0
Photos App Feature Flags        : 0
Sub Sec Time Original           : 901
Sub Sec Time Digitized          : 901
Flashpix Version                : 0100
Color Space                     : sRGB
Exif Image Width                : 3264
Exif Image Height               : 2448
Sensing Method                  : One-chip color area
Scene Type                      : Directly photographed
Custom Rendered                 : HDR (no original saved)
Exposure Mode                   : Auto
White Balance                   : Auto
Focal Length In 35mm Format     : 29 mm
Scene Capture Type              : Standard
Lens Info                       : 4.15mm f/2.2
Lens Make                       : Apple
Lens Model                      : iPhone 6 back camera 4.15mm f/2.2
GPS Latitude Ref                : North
GPS Longitude Ref               : West
GPS Altitude Ref                : Above Sea Level
```

Figure 23

```
GPS Time Stamp                  : 10:21:34
GPS Speed Ref                   : km/h
GPS Speed                       : 0
GPS Img Direction Ref           : True North
GPS Img Direction               : 104.5822785
GPS Dest Bearing Ref            : True North
GPS Dest Bearing                : 104.5822785
GPS Date Stamp                  : 2018:09:27
GPS Horizontal Positioning Error: 5 m
XMP Toolkit                     : XMP Core 5.4.0
Creator Tool                    : 11.3.1
Date Created                    : 2018:09:27 11:21:35
Current IPTC Digest             : d41d8cd98f00b204e9800998ecf8427e
IPTC Digest                     : d41d8cd98f00b204e9800998ecf8427e
Image Width                     : 3264
Image Height                    : 2448
Encoding Process                : Baseline DCT, Huffman coding
Bits Per Sample                 : 8
Color Components                : 3
Y Cb Cr Sub Sampling            : YCbCr4:2:0 (2 2)
Run Time Since Power Up         : 8:30:46
Aperture                        : 2.2
Image Size                      : 3264×2448
Megapixels                      : 8.0
Scale Factor To 35 mm Equivalent: 7.0
Shutter Speed                   : 1/898
Create Date                     : 2018:09:27 11:21:35.901
Date/Time Original              : 2018:09:27 11:21:35.901
GPS Altitude                    : 63.3 m Above Sea Level
GPS Date/Time                   : 2018:09:27 10:21:34Z
GPS Latitude                    : 52 deg 13' 55.19" N
GPS Longitude                   : 0 deg 53' 26.06" W
Circle Of Confusion             : 0.004 mm
Field Of View                   : 63.7 deg
Focal Length 35mm Equiv         : 4.2 mm (35 mm equivalent: 29.0 mm)
GPS Position                    : 52 deg 13' 55.19" N, 0 deg 53' 26.06" W
```

Figure 24

If an image file contains a thumbnail, you can extract it using the following command:

exiftool -b -ThumbnailImage <ImageFileName> <ThumbnailFileName>

6.6 Maltego

Maltego is another commercial osint tools which has a graphical user interface, it queries dns records, various api, search engines and so on to gather information and makes graphical representation of the relations between each entity found like username, ip addresses, domain name, social media accounts. All those process are handled by the Maltego server's which means you need to be careful about information you give to it.

6.7 Social Media Network

Nowadays Business stuffs require having a social media network specially for networking among each other. One of the most used by organizations and companies is Linkedin. Employee most often post events, something they have done or sharing their contacts or where they work and so on. An attacker can manually or use a tool on github to gather

valuable information about employees or organization and perform social engineering attack.

6.8 Shodan

Shodan is search engine for everything that is connected to the internet. It scans the internet every time. A great tool for OSINT. It will give information about ports and services running on them or even potential vulnerabilities, you can perform your research based on a domain name, an ip address, a device name or type, the country and so on. You can use your API Keys if you want to use the CLI tool.

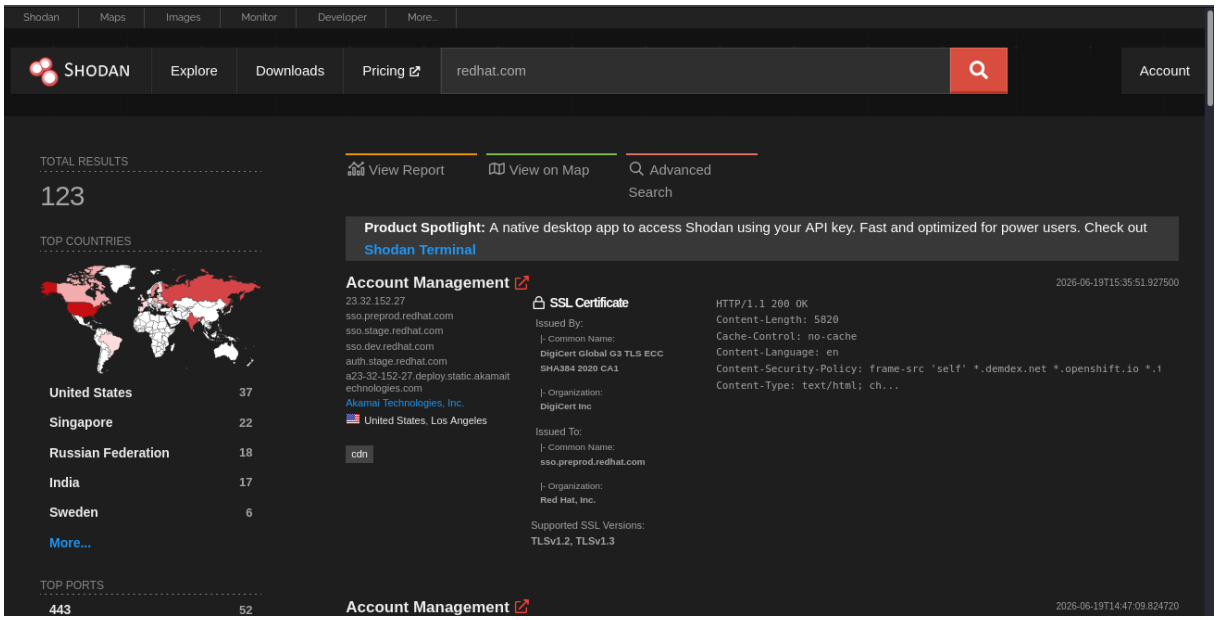


Figure 25

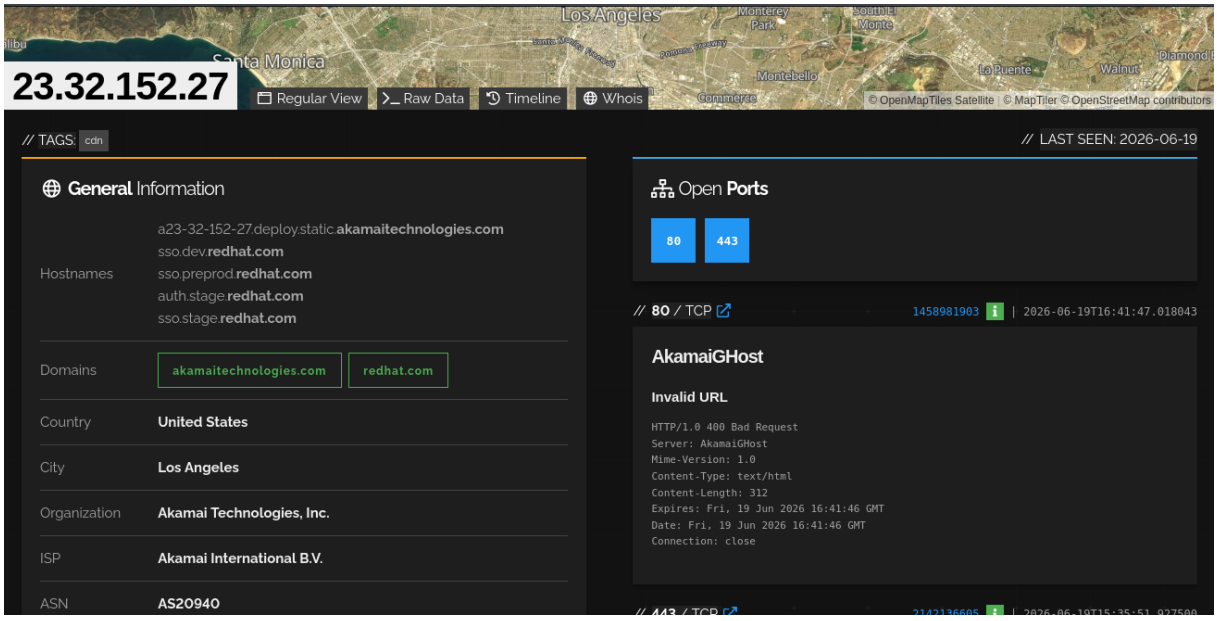


Figure 26

6.9 Protecting Against OSINT

There is no great deal a company or a single person can do to protect itself against passive reconnaissance. Once a data is published or leaked on the internet it is difficult to somehow "lock it down". What enterprise and employees can do is to be aware of what they publish on the internet and to constantly monitor their public 'footprint'.
Use a unique and strong password for each service and to help you to manage them you can use a password manager. When the service support the multifactor authentication enable it. **The multifactor authentication** is a mechanism which require 2 or more factors for a person to authenticate itself. There is 3 type of authentication :

- Known-Based Authentication (What you know) : password
- Have-Based Authentication (What you have) : IDcard
- Biometric-Based Authentication (What you are) : fingerprint

7 The Domain Name System

To communicate within a network such as the internet computers have a **32-bit address (4 bytes)** which is use to identify uniquely a host on that network. In decimal representation we have something like **59.193.57.6** each byte is separated by a dot. So for we as human, it will be very difficult to memorize those number if we want to connect or access a service from another computer in a network. That why the **Domain Name System** was created, this system is used to resolve domain names into ip addresses.

7.1 The Implications of Hacking DNS

Companies rely on name server, taking down those entities will prevent employees and customer to access services which will have a negative impact on credibility, reputation and incoming of the company. Attacker can perform a Distributed Denial of Service (DDOS) on a name server or even the root name server which occurs constantly using botnet which is a network of infected machines called zombies that will send an amount of traffic to the target to take it down.
The way DNS is implemented can also be used to perform a DDOS attack. The DNS cache of a single victim or a company's name server, this means that an attacker will send false details about a site that the victim wants to access which will bring him to the copycat version from the attacker instead of the real one.

7.2 DNS hierarchy

DNS uses a hierarchy both to store information and to serve that information when requested. At the top of the hierarchy are the root server. They store information about the **top level domain** such as **.com, .net, .org** but also **country top level domain** such as **.ne, .ma, .fr**. The .com name servers store information about domain name like ducduckgo.com, redhat.com. They cannot give you the ip address of these hosts but they can point you in the right direction of where to get the answer. The servers responsible for google.com and all of its subdomains would need to be queried in order to reach websites

or services belonging to Google. The dot represents (.) the root level and domain names are read from right to left.

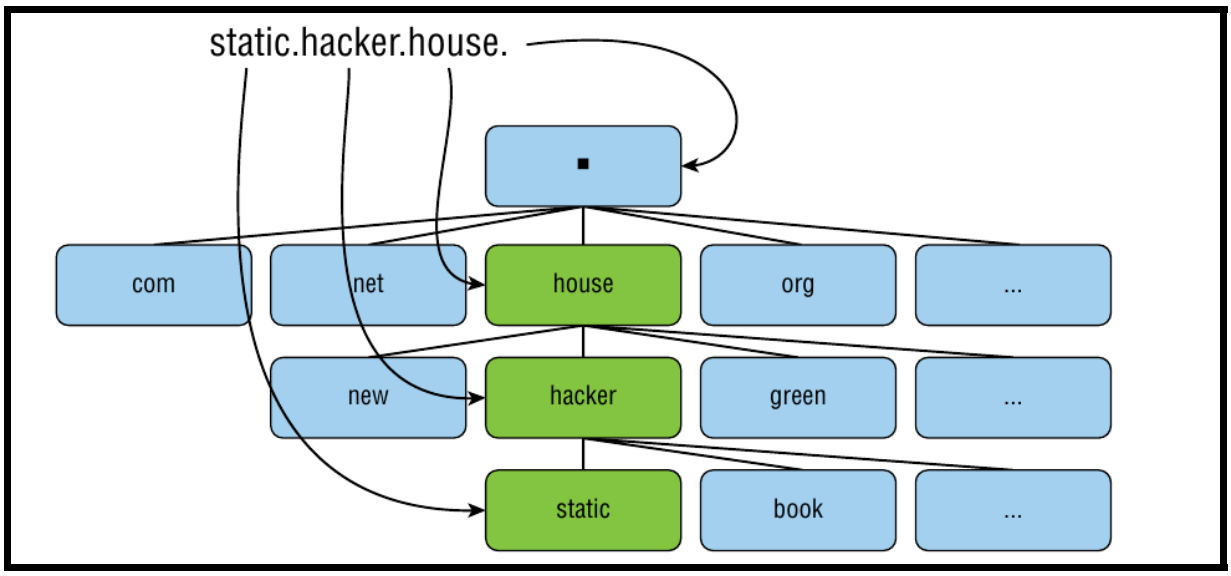


Figure 27

7.3 Basic Query

First when you type a domain name like redhat.com on a web browser, the computer will try to resolve it by checking if the answer is stored on his cache then in the file /etc/host on UNIX-like system or C:\Windows\System 32\drivers\etc\hosts on Windows system. If the answer isn't there it will query the name server within the local network most of the time it's the router. It will check on its cache if someone has already requested the domain, if not the action taken by the name server will depend if the query is recursive or not, or if the name server is set to handle recursive lookup. In the recursive way, the name server becomes the client and query other name servers until it found the answer and return it to client, in the iterative way it will just point out the client where it can eventually get answer based on the top level domain of the query. There is 13 root name servers but in reality their more physical server than. This is made to ensure redundancy and speed.

HOSTNAME	IP ADDRESSES	MANAGER
a.root-servers.net	198.41.0.4 2001:503:ba3e::2:30	VeriSign, Inc.
b.root-servers.net	199.9.14.201 2001:500:200::b	University of Southern California (ISI)
c.root-servers.net	192.33.4.12 2001:500:2::c	Cogent Communications
d.root-servers.net	199.7.91.13 2001:500:2d::d	University of Maryland
e.root-servers.net	192.203.230.10 2001:500:a8::e	NASA (Ames Research Center)

Figure 28

HOSTNAME	IP ADDRESSES	MANAGER
f.root-servers.net	192.5.5.241 2001:500:2f::f	Internet Systems Consortium, Inc.
g.root-servers.net	192.112.36.4 2001:500:12::d0d	U.S. Department of Defense (NIC)
h.root-servers.net	198.97.190.53 2001:500:1::53	U.S. Army (Research Lab)
i.root-servers.net	192.36.148.17 2001:7fe::53	Netnod
j.root-servers.net	192.58.128.30 2001:503:c27::2:30	VeriSign, Inc.
k.root-servers.net	193.0.14.129 2001:7fd::1	RIPE NCC
l.root-servers.net	199.7.83.42 2001:500:9f::42	ICANN
m.root-servers.net	202.12.27.33 2001:dc3::35	WIDE Project

Figure 29

7.4 Authority and Zone

Some name servers have authority over a particular subsection of the Domain Name System which means they are the only ones that can answer query to a specific domain. Those name servers are called Start of Authority (SOA) but other name server can answer the query if they had already checked and cached it. The SOA for duckduckgo.com is responsible for domains only within the duckduckgo.com space. It could not have given the answer to plus.google.com or any other google.com domain query because it is not authoritative for the google.com space. Actually, there is usually more than one name server for a company like Google, but this is for redundancy; they both do the same job, and both are considered the SOA.

7.5 DNS Resource Records

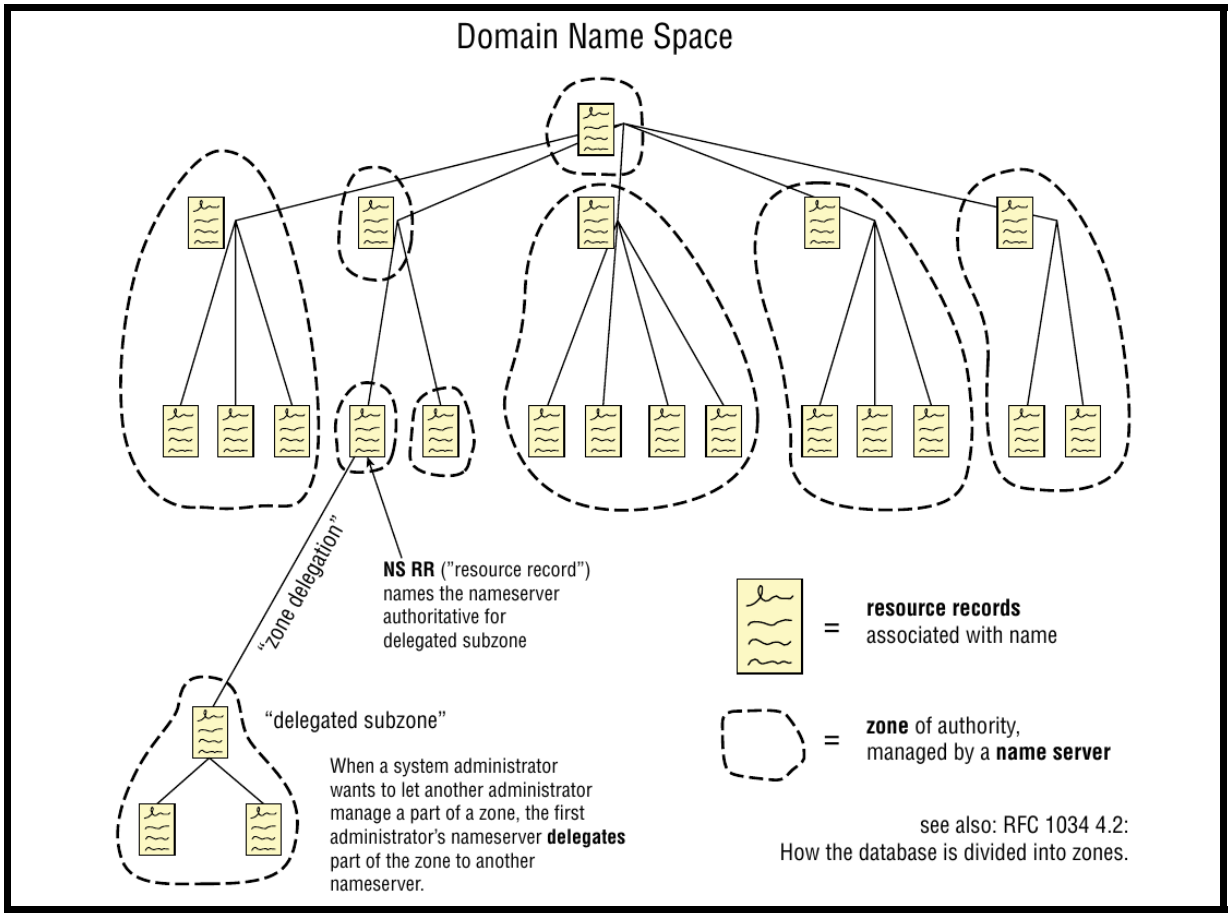


Figure 30

The data stored in a name server is called a ressource record. A zone is a specific portion of the DNS namespace that is under the authority of a name server. The different type of a ressource record are :

- **Address of Host (A)**: This represent an ipv4 address of a host.
- **Address of Host (AAAA)**: This represent an ipv6 address of a host.

- **Canonical Name (CN)**: This represent an alias which is used to specify other hostname for a subdomain or a fully qualified domain.
- **Mail Exchange (MX)**: This contains the ip address or hostname of a mail server.
- **Name Server (NS)**: This contains informations about the name server.
- **Start of Authority (SOA)**: This contains informations about the primary name server of a zone.
- **Pointer (PTR)**: This contains the hostname for an ip address.
- **Text (TXT)**: This just text record that are used for notes.

There is a lot of software that implements a dns name server such as **DJBDNS, PowerDNS, MaraDNS, MSDNS or DNSmasq** but the most used which is considered as a defacto standard is **BIND (Berkley Internet Name Domain)**.

7.6 Bind9

Now we are gonna implement a basic dns name server for a testing purpose and to see how it really works. First we will install bind9 on our ubuntu server using the following command:

sudo apt install bind9

```
shinra@souleater:~$ sudo apt install bind9
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  bind9-utils dns-root-data
Suggested packages:
  bind-doc
The following NEW packages will be installed:
  bind9 bind9-utils dns-root-data
0 upgraded, 3 newly installed, 0 to remove and 0 not upgraded.
Need to get 423 kB of archives.
After this operation, 1,633 kB of additional disk space will be used.
Do you want to continue? [Y/n]
```

Figure 31

The configuration for Bind9 are stored in the file /etc/bind/named.conf. This file include other files that define zones.

```
shinra@souleater:/etc/bind$ cat named.conf
// This is the primary configuration file for the BIND DNS server named.
//
// Please read /usr/share/doc/bind9/README.Debian for information on the
// structure of BIND configuration files in Debian, *BEFORE* you customize
// this configuration file.
//
// If you are just adding zones, please do that in /etc/bind/named.conf.local

include "/etc/bind/named.conf.options";
include "/etc/bind/named.conf.local";
include "/etc/bind/named.conf.default-zones";
shinra@souleater:/etc/bind$ _
```

Figure 32

So let's create a zone in the /etc/bind/named.conf.local by defining the domain,the type of the server as master and also the file that will contain the records.

```
//
// Do any local configuration here
//
zone "mydomain.com"
{
    type master;
    file "/etc/bind/db.mydomain.com";
};
// Consider adding the 1918 zones here, if they are not used in your
// organization
//include "/etc/bind/zones.rfc1918";
```

Figure 33

Now we will create the db file by copying the template from the db.local file and make changes to match our context.

```
$TTL    604800
@       IN      SOA     ns.mydomain.com. root.mydomain.com. (
                              2         ; Serial
                         604800         ; Refresh
                          86400         ; Retry
                        2419200         ; Expire
                         604800 )       ; Negative Cache TTL
@       IN      NS      ns.mydomain.com.
ns      IN      A       192.168.56.1
www     IN      A       192.168.56.10
```

Figure 34

Let's talk about the composition of the file above. the **@** is used to define the origin source which correspond to the domain name declared for the zone. But if the variable **$ORIGIN is set** within this file the @ will correspond to it. The variable **$TTL** is used to define the expiration time in seconds of the data stored in the file this will help caching name server to update their data by asking for new data to the master. **The first column** defines the name which is mapped to record data. **The second column** define the record class, it can have two value : IN that stand for INTERNET and CHAOS . **The third column** define the record type which can be AAAA, NS, A, CN, MX. And the last one is for the record data which is mapped to the name in the first column. The data can be a fully qualified domain (NS,MX) , text or ip address. When we use MX record type there is another column after the record type that can be set to define the priority between the mail servers.
Now let's restart the service to make it use the new configuration and query the name server through the resolver dig.

```
shinra@souleater:/etc/bind$ sudo systemctl start bind9.service
shinra@souleater:/etc/bind$ dig @localhost mydomain.com

; <<>> DiG 9.18.39-0ubuntu0.24.04.5-Ubuntu <<>> @localhost mydomain.com
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 64951
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 0, AUTHORITY: 1, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1232
; COOKIE: 32c5075d6f5efc2d010000006a42d4819b04250356c42557 (good)
;; QUESTION SECTION:
;mydomain.com.                  IN      A

;; AUTHORITY SECTION:
mydomain.com.           604800  IN      SOA     ns.mydomain.com. root.mydomain.com. 2 604800 86400 2419200 604800

;; Query time: 0 msec
;; SERVER: 127.0.0.1#53(localhost) (UDP)
;; WHEN: Mon Jun 29 20:24:33 UTC 2026
;; MSG SIZE  rcvd: 113

shinra@souleater:/etc/bind$
```

Figure 35

7.7 DNS Hacking Toolkit

Here a list of tools that can be used to hack dns system.

- Dig: A DNS querying tool
- NSlookup: An alternative DNS querying tool
- DNSrecon: A script for carrying out reconnaissance on a name server
- DNSenum: A tool for enumerating information from a name server
- Fierce: A name server scanning tool, similar to Dnsrecon and dnsenum
- Host: Another DNS querying tool
- WHOIS: The name of a protocol and tool for querying information

- DNSspoof: A tool for spoofing DNS packets
- Dsniff : A tool for sniffing DNS caches and packets
- Hping3: A tool for creating and injecting custom packets useful in many scenarios
- Scapy: A packet injection tool
- Nmap: A port scanner for scanning any host and returning useful information
- Searchsploit: A tool for searching for known exploit scripts and information
- MSFconsole: A modular command-line tool that is part of the popular
- Metasploit penetration testing framework
- Wireshark: A tool for inspecting packets on your network

7.8 Finding Host

Before hacking a client's name server we need first to known where it is. For that we can use the protocol **whois** which query public data record for information related to a domain name. Let's use it to get information related to **redhat.com**

```
┌──(kali㉿kali)-[~]
└─$ whois redhat.com
   Domain Name: REDHAT.COM
   Registry Domain ID: 4832854_DOMAIN_COM-VRSN
   Registrar WHOIS Server: whois.comlaude.com
   Registrar URL: http://www.comlaude.com
   Updated Date: 2026-04-25T23:09:40Z
   Creation Date: 1994-05-26T04:00:00Z
   Registry Expiry Date: 2027-05-25T04:00:00Z
   Registrar: Nom-iq Ltd. dba COM LAUDE
   Registrar IANA ID: 470
   Registrar Abuse Contact Email: abuse@comlaude.com
   Registrar Abuse Contact Phone: +442074218250
   Domain Status: clientDeleteProhibited https://icann.org/epp#clientDeleteProhibited
   Domain Status: clientTransferProhibited https://icann.org/epp#clientTransferProhibited
   Domain Status: clientUpdateProhibited https://icann.org/epp#clientUpdateProhibited
   Name Server: DNS1.P01.NSONE.NET
   Name Server: DNS1.P02.NSONE.NET
   Name Server: DNS2.P01.NSONE.NET
   Name Server: DNS2.P02.NSONE.NET
   Name Server: DNS3.P01.NSONE.NET
   Name Server: DNS4.P01.NSONE.NET
   DNSSEC: unsigned
   URL of the ICANN Whois Inaccuracy Complaint Form: https://www.icann.org/wicf/
>>> Last update of whois database: 2026-07-04T18:18:22Z <<<
```

Figure 36

One way to get a list of hostname and subdomains including name server is to brute-force them. For that we can use the module brute_hosts from **recon-ng**. After launching recon-ng we loaded the module as show below.

```
[recon-ng][default] > modules search brute_host
[*] Searching installed modules for 'brute_host' ...

  Recon
  -----
    recon/domains-hosts/brute_hosts

[recon-ng][default] > module load brute_hosts
[!] Invalid command: module load brute_hosts.
[recon-ng][default] > module load recon/domains-hosts/brute_hosts
[!] Invalid command: module load recon/domains-hosts/brute_hosts.
[recon-ng][default] > modules load brute_hosts
[recon-ng][default][brute_hosts] > info

      Name: DNS Hostname Brute Forcer
    Author: Tim Tomes (@lanmaster53)
   Version: 1.0

Description:
  Brute forces host names using DNS. Updates the 'hosts' table with the results.

Options:
  Name      Current Value                          Required  Description
  --------  -------------                          --------  -----------
  SOURCE    default                                yes       source of input (see 'info' for details)
  WORDLIST  /home/kali/.recon-ng/data/hostnames.txt  yes       path to hostname wordlist

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

[recon-ng][default][brute_hosts] >
```

Figure 37

We have set the domain name as the source value and run the module through the command **run**. After the module was run we used the command **show hosts** to see the results.

```
[*] zera.hacker.house ⇒ No record found.
[*] zebra.hacker.house ⇒ No record found.
[*] za.hacker.house ⇒ No record found.
[*] zlog.hacker.house ⇒ No record found.
[*] zeus.hacker.house ⇒ No record found.
[*] zulu.hacker.house ⇒ No record found.
[*] zm.hacker.house ⇒ No record found.
[*] zw.hacker.house ⇒ No record found.

------
SUMMARY
------
[*] 3 total (3 new) hosts found.
[recon-ng][default][brute_hosts] > show hosts

  +-------+--------------------+-----------------+--------+---------+----------+-----------+-------+-------------+
  | rowid |        host        |    ip_address   | region | country | latitude | longitude | notes |    module   |
  +-------+--------------------+-----------------+--------+---------+----------+-----------+-------+-------------+
  | 1     | static.hacker.house | 148.113.220.86 |        |         |          |           |       | brute_hosts |
  | 2     | vpn.hacker.house    | 137.74.227.71  |        |         |          |           |       | brute_hosts |
  | 3     | www.hacker.house    | 148.113.220.86 |        |         |          |           |       | brute_hosts |
  +-------+--------------------+-----------------+--------+---------+----------+-----------+-------+-------------+

[*] 3 rows returned
[recon-ng][default][brute_hosts] >
```

Figure 38

Another tool that can used to find subdomain or domain ip address is **host**, it perform a dns lookup and give also additional info such as mail servers of the given domain name.

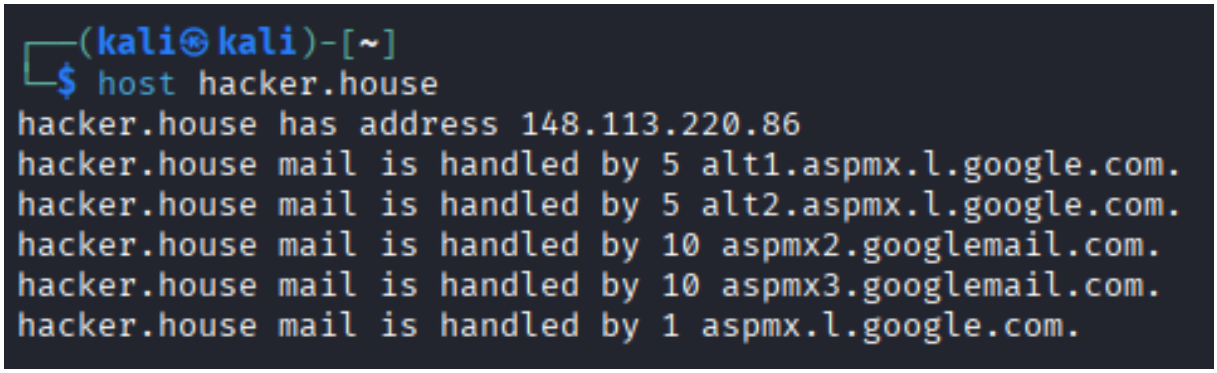


Figure 39

It can also be used to perform a reverse lookup which means looking for the the fully qualified domain name of a given ip address.

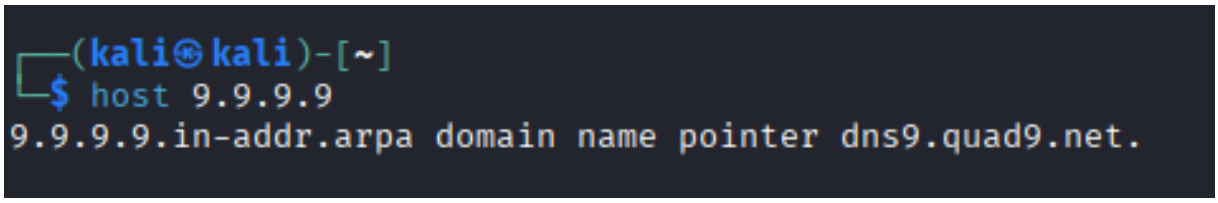


Figure 40

7.9 Finding the SOA with Dig

We can use dig to request name server in order to find the SOA of a given domain name. As show below we can see that sometime the SOA is not part of the domain name specially for small enterprise which don't have enough budget to get their own name server so they rely on a hosting company. There is also the admin email which is **admin@hacker.house** even though is not explicit due to how it is represented on the resource record.



Figure 41

Now that we known the name server we can query to find it ip address by submitting it fully qualified domain name.

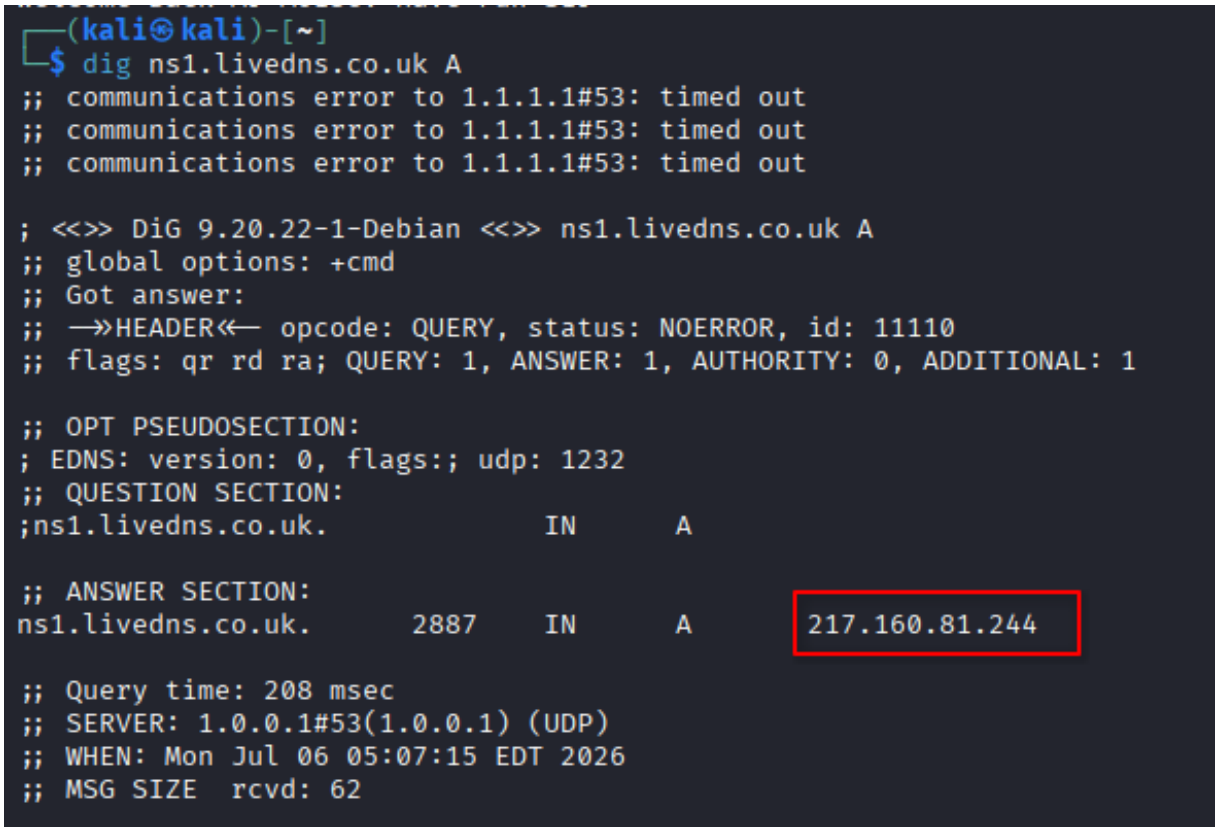


Figure 42

7.10 Port Scanning with Nmap

Now we gonna start hacking our virtual name server from the booklab by scanning and probing it to find more useful information and, ideally some security vulnerabilities. We have the ip address of our target as show below. The ip address can change because it is allocated dynamically, so when maybe a shut down the virtual machine and run it after, i could eventually get another ip address.



Figure 43

Let 's run a simple nmap scan to find the available service running on our target.

```
┌──(kali㉿kali)-[~]
└─$ nmap 192.168.57.108
Starting Nmap 7.99 ( https://nmap.org ) at 2026-07-06 07:23 -0400
Nmap scan report for 192.168.57.108
Host is up (0.00010s latency).
Not shown: 982 closed tcp ports (reset)
PORT      STATE SERVICE
21/tcp    open  ftp
22/tcp    open  ssh
23/tcp    open  telnet
25/tcp    open  smtp
53/tcp    open  domain
80/tcp    open  http
111/tcp   open  rpcbind
139/tcp   open  netbios-ssn
389/tcp   open  ldap
443/tcp   open  https
445/tcp   open  microsoft-ds
873/tcp   open  rsync
2049/tcp  open  nfs
3128/tcp  open  squid-http
3306/tcp  open  mysql
5432/tcp  open  postgresql
8080/tcp  open  http-proxy
10000/tcp open  snet-sensor-mgmt
MAC Address: 08:00:27:12:86:06 (Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 0.24 seconds
```

Figure 44

By default nmap scan for services running on TCP, to be able to find service running on UDP we must add the option **-sU**. Scanning UDP services take more time compared to TCP.

```
┌──(kali㉿kali)-[~]
└─$ nmap -sU 192.168.57.108
Starting Nmap 7.99 ( https://nmap.org ) at 2026-07-06 07:55 -0400
Nmap scan report for 192.168.57.108
Host is up (0.00062s latency).
Not shown: 989 closed udp ports (port-unreach)
PORT      STATE         SERVICE
53/udp    open          domain
68/udp    open|filtered dhcpc
69/udp    open|filtered tftp
111/udp   open          rpcbind
137/udp   open          netbios-ns
138/udp   open|filtered netbios-dgm
161/udp   open          snmp
500/udp   open          isakmp
2049/udp  open          nfs
4500/udp  open|filtered nat-t-ike
10000/udp open          ndmp
MAC Address: 08:00:27:12:86:06 (Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 1054.69 seconds
```

Figure 45

7.11 Digging for Information

Unless it is told to query a specific name server, Dig will try each of the servers listed in /etc/resolv.conf . If no usable server addresses are found, Dig will send the query to the localhost. To query a specific name server we use the following syntax:

dig @<nameserver> <domain_name>

```
┌──(kali㉿kali)-[~]
└─$ dig @192.168.57.108 nsa.gov

; <<>> DiG 9.20.22-1-Debian <<>> @192.168.57.108 nsa.gov
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 13620
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 0, AUTHORITY: 1, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;nsa.gov.                       IN      A

;; AUTHORITY SECTION:
nsa.gov.                600     IN      SOA     ns1.nsa.gov. root.nsa.gov. 2007010401 3600 600 86400 600

;; Query time: 4 msec
;; SERVER: 192.168.57.108#53(192.168.57.108) (UDP)
;; WHEN: Mon Jul 06 15:43:07 EDT 2026
;; MSG SIZE  rcvd: 81
```

Figure 46

The flag aa set shows us that the answer come from the SOA name server of the domain. Now that we known the hostname of the SOA we can query the name server to get the A record .

```
┌──(kali㉿kali)-[~]
└─$ dig @192.168.57.109 ns1.nsa.gov A

; <<>> DiG 9.20.22-1-Debian <<>> @192.168.57.109 ns1.nsa.gov A
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 20294
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 2, ADDITIONAL: 2

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;ns1.nsa.gov.                   IN      A

;; ANSWER SECTION:
ns1.nsa.gov.            3600    IN      A       10.1.0.50

;; AUTHORITY SECTION:
nsa.gov.                3600    IN      NS      ns1.nsa.gov.
nsa.gov.                3600    IN      NS      ns2.nsa.gov.

;; ADDITIONAL SECTION:
ns2.nsa.gov.            3600    IN      A       10.1.0.51

;; Query time: 8 msec
;; SERVER: 192.168.57.109#53(192.168.57.109) (UDP)
;; WHEN: Thu Jul 09 14:15:02 EDT 2026
;; MSG SIZE  rcvd: 104
```

Figure 47

The name server give us the queried record and also some additional information about the existence of a second name server for the domain. Dig can also be used to query any type of record for example we can ask for a MX record which represent mail server hostname. By default dig query A records.

```
└─$ dig @192.168.57.109 nsa.gov MX

; <<>> DiG 9.20.22-1-Debian <<>> @192.168.57.109 nsa.gov MX
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 40659
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 2, AUTHORITY: 2, ADDITIONAL: 5

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;nsa.gov.                       IN      MX

;; ANSWER SECTION:
nsa.gov.                3600    IN      MX      10 mail1.nsa.gov.
nsa.gov.                3600    IN      MX      20 mail2.nsa.gov.

;; AUTHORITY SECTION:
nsa.gov.                3600    IN      NS      ns2.nsa.gov.
nsa.gov.                3600    IN      NS      ns1.nsa.gov.

;; ADDITIONAL SECTION:
mail1.nsa.gov.          3600    IN      A       10.1.0.25
mail2.nsa.gov.          3600    IN      A       10.1.0.26
ns1.nsa.gov.            3600    IN      A       10.1.0.50
ns2.nsa.gov.            3600    IN      A       10.1.0.51

;; Query time: 0 msec
;; SERVER: 192.168.57.109#53(192.168.57.109) (UDP)
;; WHEN: Thu Jul 09 18:14:24 EDT 2026
;; MSG SIZE  rcvd: 180
```

Figure 48

Nslookup is also another tools that can be used to query name server.

```
┌──(kali㉿kali)-[~]
└─$ nslookup
> set querytype=SOA
> google.com
Server:         1.1.1.1
Address:        1.1.1.1#53

Non-authoritative answer:
google.com
        origin = ns1.google.com
        mail addr = dns-admin.google.com
        serial = 944394919
        refresh = 900
        retry = 900
        expire = 1800
        minimum = 60

Authoritative answers can be found from:
>
```

Figure 49

We can sniff into the traffic via wireshark to see what happen at the lower level. As show in the figure below two packet are sent one for the query and the other for the answer. We can see the ip addresses involved into this communication,the ports used one for the client which is randomly chose by the operating system and the other which is the standard port for dns. There are also some flags set, for example to set the name server to perform recursive lookup (**rd**) or to specify if the answer came from the SOA (**aa**).

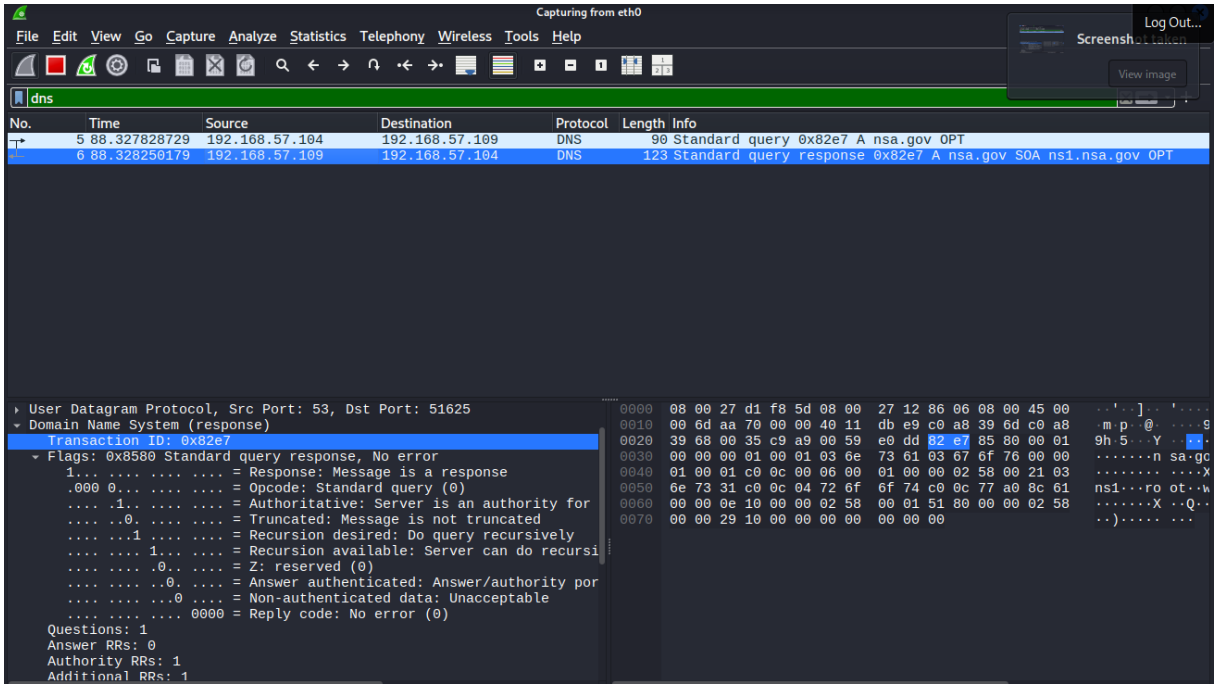


Figure 50

7.12 Information Leak CHAOS

CHAOS is one of the network like the ARPANET that exist before the internet that we know. A misconfiguration of the server can leak important informations. The BIND instance running which means the software itself is able to accept CHAOSNET queries. We already know the IN record class which is used by default within a query but there is also the CHAOS class which is used by BIND to store information about itself, like metadata. The queries look like domain name but with bind at the end.

dig @<NameServer> <Class> <DomainName> <RecordType>

As example we will query our name server to get the version of bind running on it using the chaos record class. We will use the following syntax below. Getting information about the version of bind running we can further investigate for possible vulnerability related to the specific version that could eventually be exploited.

dig @192.168.57.110 chaos version.bind txt

```
┌──(kali㉿kali)-[~]
└─$ dig @192.168.57.110 chaos version.bind txt

; <<>> DiG 9.20.22-1-Debian <<>> @192.168.57.110 chaos version.bind txt
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 46760
;; flags: qr aa rd; QUERY: 1, ANSWER: 1, AUTHORITY: 1, ADDITIONAL: 1
;; WARNING: recursion requested but not available

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;version.bind.                  CH      TXT

;; ANSWER SECTION:
version.bind.           0       CH      TXT     "9.8.1"

;; AUTHORITY SECTION:
version.bind.           0       CH      NS      version.bind.

;; Query time: 0 msec
;; SERVER: 192.168.57.110#53(192.168.57.110) (UDP)
;; WHEN: Fri Jul 10 07:21:51 EDT 2026
;; MSG SIZE  rcvd: 73
```

Figure 51

Other information can be gather such as the local hostname of the server (**hostname.bind**), the authors responsable of the version (**authors.bind**) or internal id (ID.server).

7.13 Zone Transfer Requests

Zone transfer requests is a process used to copy a dns setup from one name server to another. So to perform such transaction it needs to use a connection-oriented protocol which is TCP, that why when we see an open port 53 running on tcp there is a possibility to perform a zone transfert. Using the special record class AXFR shows all the records for a zone. After query the server we see a bunch of hostname, ip address and some text record also left there as note.

```
┌──(kali㉿kali)-[~]
└─$ dig @192.168.57.110 AXFR nsa.gov

; <<>> DiG 9.20.22-1-Debian <<>> @192.168.57.110 AXFR nsa.gov
; (1 server found)
;; global options: +cmd
nsa.gov.                3600    IN      SOA     ns1.nsa.gov. root.nsa.gov. 2007010401 3600 600 86400 600
nsa.gov.                3600    IN      NS      ns1.nsa.gov.
nsa.gov.                3600    IN      NS      ns2.nsa.gov.
nsa.gov.                3600    IN      MX      10 mail1.nsa.gov.
nsa.gov.                3600    IN      MX      20 mail2.nsa.gov.
fedora.nsa.gov.         3600    IN      TXT     "The black sparrow password"
fedora.nsa.gov.         3600    IN      AAAA    fd7f:bad6:99f2::1337
fedora.nsa.gov.         3600    IN      A       10.1.0.80
firewall.nsa.gov.       3600    IN      A       10.1.0.105
fw.nsa.gov.             3600    IN      A       10.1.0.102
mail1.nsa.gov.          3600    IN      TXT     "v=spf1 a mx ip4:10.1.0.25 ~all"
mail1.nsa.gov.          3600    IN      A       10.1.0.25
mail2.nsa.gov.          3600    IN      TXT     "v=spf1 a mx ip4:10.1.0.26 ~all"
mail2.nsa.gov.          3600    IN      A       10.1.0.26
ns1.nsa.gov.            3600    IN      A       10.1.0.50
ns2.nsa.gov.            3600    IN      A       10.1.0.51
prism.nsa.gov.          3600    IN      A       172.16.40.1
prism6.nsa.gov.         3600    IN      AAAA    ::1
sigint.nsa.gov.         3600    IN      A       10.1.0.101
snowden.nsa.gov.        3600    IN      A       172.16.40.1
vpn.nsa.gov.            3600    IN      A       10.1.0.103
web.nsa.gov.            3600    IN      CNAME   fedora.nsa.gov.
webmail.nsa.gov.        3600    IN      A       10.1.0.104
www.nsa.gov.            3600    IN      CNAME   fedora.nsa.gov.
xkeyscore.nsa.gov.      3600    IN      TXT     "knock twice to enter"
xkeyscore.nsa.gov.      3600    IN      A       10.1.0.100
nsa.gov.                3600    IN      SOA     ns1.nsa.gov. root.nsa.gov. 2007010401 3600 600 86400 600
;; Query time: 0 msec
;; SERVER: 192.168.57.110#53(192.168.57.110) (TCP)
```

Figure 52

7.14 Information-Gathering Tools

There is a lot of tools availables for information gathering in the internet. We will just choose some of them and try to use them.

- **dnsrecon :** This tool is used for scanning and enumerating hosts from a given name server. **Fierce** is also another tools that perform the same thing but with a larger wordlist.

dnsrecon -n 192.168.57.12 -d nsa.gov

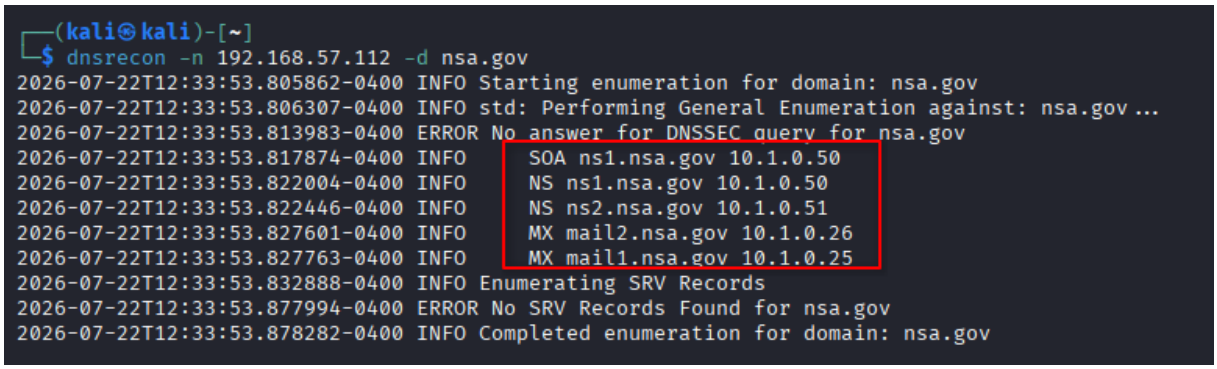


Figure 53

- **dnsenum :** This tool is also used for enumerating hosts from a given name server. It automatically try to perform a zone transfert on ns1 and ns2 name server but those one don't really exist on my virtual network that why it didn't work.

dnsenum –dnsserver192.168.57.12 nsa.gov

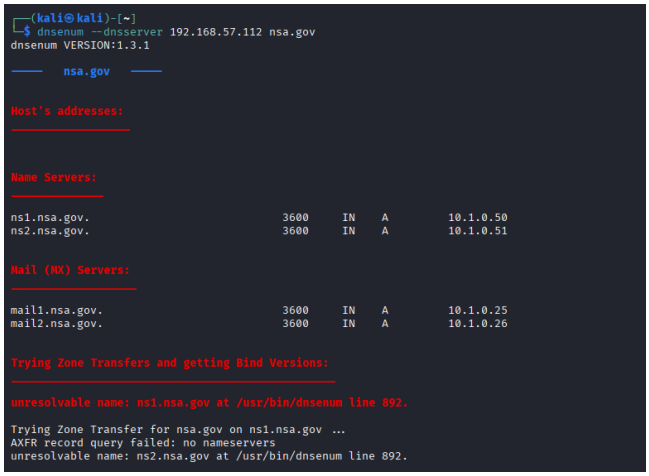


Figure 54

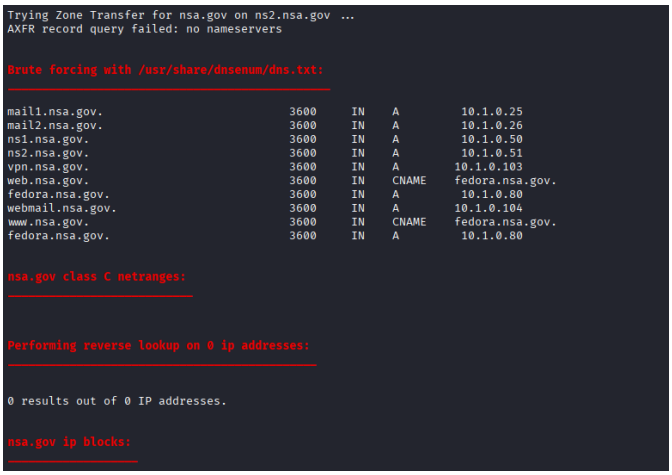


Figure 55

Is important to not completely rely on only one tool or just tools in general. The best way is to mix both manual process and automated process through different tools, and try to compare and gather the maximum of information from both.

7.15 Searching for Vulnerabilities and Exploits

Now that we have gather some information from our target name server specially the version of bind running on it, we will search for possible vulnerability and exploit targeting that version. As for OSINT we need to search information from different ressources and not only one. For that we gonna use the **searchsploit** tool that query a copy of an exploits database from **www.exploit-db.com**. The figure above shows us that there is possibly 4 exploits for our bind version.

```
┌──(kali㉿kali)-[~]
└─$ searchsploit bind isc
------------------------------------------------------------------------------
 Exploit Title
------------------------------------------------------------------------------
ISC BIND (Linux/BSD) - Remote Buffer Overflow (1)
ISC BIND (Multiple OSes) - Remote Buffer Overflow (2)
ISC BIND 4.9.7 -T1B - named SIGINT / SIGIOT Symlink
ISC BIND 4.9.7/8.x - Traffic Amplification and NS Route Discovery
ISC BIND 8 - Remote Cache Poisoning (1)
ISC BIND 8 - Remote Cache Poisoning (2)
ISC BIND 8.1 - Host Remote Buffer Overflow
ISC BIND 8.2.2 / IRIX 6.5.17 / Solaris 7.0 - NXT Overflow / Denial of Service
ISC BIND 8.2.2-P5 - Denial of Service
ISC BIND 8.2.x - 'TSIG' Remote Stack Overflow (1)
ISC BIND 8.2.x - 'TSIG' Remote Stack Overflow (2)
ISC BIND 8.2.x - 'TSIG' Remote Stack Overflow (3)
ISC BIND 8.2.x - 'TSIG' Remote Stack Overflow (4)
ISC BIND 8.3.x - OPT Record Large UDP Denial of Service
ISC BIND 9 - Denial of Service
ISC BIND 9 - Remote Dynamic Update Message Denial of Service (PoC)
ISC BIND 9 - TKEY (PoC)
ISC BIND 9 - TKEY Remote Denial of Service (PoC)
Microsoft Windows Kernel - 'win32k!NtQueryCompositionSurfaceBinding' Stack Memory Disclosure
Zabbix 2.0.5 - Cleartext ldap_bind_Password Password Disclosure (Metasploit)
------------------------------------------------------------------------------
Shellcodes: No Results
```

Figure 56

7.16 DNS Traffic Amplification

8 In progress...

In progress....