University Tunis El Manar
National Engineering School of Tunis

Information and Communication Technologies Department

Final Year Project I Report

Created by :

MIGHRI Manar
CHERNI Oussema

Class:

1 Telecom 1

Introduction to Computer Security: Protect Your Computer Against Common Threats with a Focus on Backdoors and How to Prevent Them

---

Supervised by: **Mr. HAMMAMI Hamza**

2023 - 2024

ABSTRACT

This project report provides a comprehensive study on Windows computer security, delving into various aspects crucial for understanding and mitigating cyber threats targeting Windows systems. The report begins with a general introduction, laying the groundwork for comprehending the complexities of Windows computer security. It then proceeds to explore different types of attacks and malware specifically designed to exploit vulnerabilities in Windows systems, covering topics such as denial-of-service attacks, man-in-the-middle attacks, and phishing attacks. Additionally, it examines vulnerability exploitation techniques, including various types of exploits targeting Windows or Microsoft applications. The report also analyzes malware tailored to exploit Windows vulnerabilities, such as spyware, Trojans, rootkits, and ransomware. Furthermore, it offers guidance on setting up a demonstration laboratory and presenting key solutions to overcome cyber threats targeting Windows computers. In conclusion, this project report serves as a valuable resource for understanding the challenges posed by Windows computer security and provides insights into effective defense strategies and countermeasures against cyber threats.

DEDICATION

To our parents

To our friends and loved ones.

MIGHRI Manar & CHERNI Oussema

ACKNOWLEDGEMENTS

We would like to express our heartfelt gratitude towards all those who have played an important and major role in the completion of our internship. First, we are deeply thankful to Mr. Hamza Hammami for his constant support and guidance, his priceless advice, and his valuable expertise, which played a significant role in enriching this experience.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

GENERAL INTRODUCTION

We live in a technologically driven age where computers are both omnipresent and essential. The risks that endanger the security and integrity of our computer systems change along with the digital landscape. Because of the ever-present interplay between innovation and malice, cybersecurity requires a thorough understanding to secure individual, corporate, and social assets.

The goal of this project is to shed light on the complex field of computer security, with a specific focus on backdoors and prevention. Our mission is to provide people with the information and resources they need to defend their digital domains from common threats, with a particular emphasis on understanding, preventing, and mitigating cyber attacks.

Our efforts will always prioritize ethical issues as we explore the complexities of cybersecurity. The project is designed with the highest regard for ethical and legal requirements, guaranteeing that our investigation into computer security stays within the bounds of responsible and regulated research.

The report comprises three key chapters. The first chapter introduces the fundamentals of computer security within the Windows environment, detailing prevalent malware and exploits. The second chapter examines attacks directed at Windows computers and the malware tailored to compromise them. Finally, Chapter 3 focuses on the setup of a demonstration laboratory, including configuring virtual machines, installing Njrat for creating backdoors, executing exploits, and discussing protection techniques against Windows-based attacks. Together, these chapters provide a comprehensive understanding of security challenges in the Windows ecosystem.

CHAPTER 1

INTRODUCTION TO WINDOWS COMPUTER SECURITY

Contents

1 Introduction

To effectively counter cyber threats, it's crucial to comprehend their primary target: operating systems. An operating system (OS) serves as a vital interface between users and computers, facilitating essential functions like memory and file management, input/output handling, security enforcement, process management, error detection, performance control, and peripheral device management.

Among the array of operating systems available, Windows stands out as one of the most widely recognized and utilized platforms. Introduced by Microsoft in 1985, Windows has evolved from its inception with Windows 1.0 to the latest iteration, Windows 11, offering users a user-friendly interface, multitasking capabilities, and extensive peripheral device support. However, its popularity also renders it a prime target for security threats.

This chapter serves as an introduction to Windows computer security, aiming to explore and understand the fundamental components of Windows and the potential threats to its stability.

Figure 1.1: Evolution of Windows operating system

Windows Operating System Technology

Recognized for its sophisticated architecture, Windows operating systems feature a comprehensive array of components that collaborate to provide a seamless and secure user experience. This section will delve into the primary components of Windows operating systems, elucidating their roles in the overall functioning of the system.[3].

2 Components of Operating System

In this section, we will delve into the core elements that shape the landscape of modern operating systems. These components serve as the foundation for how our computers operate, facilitating everything from hardware management to user interaction.

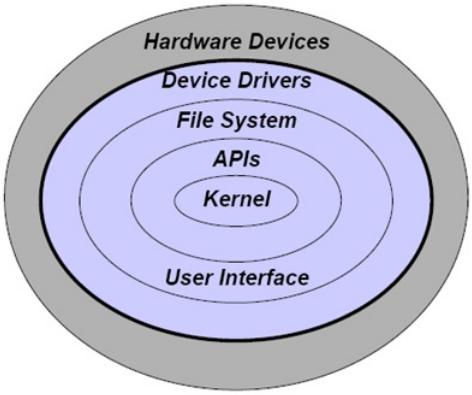

Figure 1.2: operating-system-components

- **Kernel**

  The kernel serves as the core component of the operating system, responsible for managing hardware resources, memory, and system processes. In modern Windows computers, the kernel ensures stable and efficient operation by providing essential services for hardware interaction and process management.

- **APIs (Application Programming Interfaces)**

  APIs play a crucial role in facilitating communication between applications and the operating system. They provide a standardized way for software developers to access system resources and services, enabling the development of diverse and feature-rich applications for Windows platforms.

- **File System**

  The file system is responsible for organizing and managing data stored on storage devices such as hard drives and SSDs. In modern Windows computers, the File Allocation Table (FAT) file system, although considered legacy, still plays a role in maintaining compatibility with older storage devices and file formats

- **User Interface (UI)**

  The user interface serves as the primary means through which users interact with their computers and applications. In modern Windows computers, the UI provides a familiar and intuitive experience, allowing users to navigate the system, access applications, and perform tasks efficiently.

- **Device Drivers**

  Device drivers are essential software components that enable the operating system to communicate with hardware devices such as printers, graphics cards, and network adapters. In modern Windows computers, device drivers ensure hardware compatibility and functionality, allowing users to utilize a wide range of peripherals and accessories.

- **System Services**

  System services encompass a broad spectrum of background processes and functionalities provided by the operating system. In modern Windows computers, these services include networking, security, power management, and system maintenance, ensuring smooth and reliable operation of the system.

- **Registry**

  The registry acts as a central database that stores various configuration settings for the Windows operating system, applications, and hardware. It functions as a comprehensive repository, housing information such as program locations, user preferences, hardware configurations, and file associations.

While the operating system serves as the foundational framework for computing operations, its vulnerabilities can be exploited by malicious software, commonly known as malware. In the next section, we will delve into different types of malware and their implications on data privacy and system stability.

3 Malware: a Barrier to Windows Computer Security

In this section, we will explore the realm of malware, a significant challenge to the security of Windows-based systems. We'll examine the various types of malicious software and how they can affect the privacy and stability of Windows computers.

3.1 Definition

Malicious software, commonly known as malware, encompasses a broad spectrum of software designed with the intent to cause harm. Its objectives range from disrupting regular computer operations to gathering sensitive information, infiltrating private networks, and displaying unwanted advertisements or spam. While the term 'malware' was coined by Yisrael Radai in 1990, these types of software were previously referred to as computer viruses. Essentially, malware acts as a general term encompassing various intrusive and harmful software, such as Trojan horses, worms, adware, and spyware. Some malware operates covertly, stealthily siphoning off private data, while others directly cause damage, sometimes even extorting payment from unsuspecting victims[1].

Most infamous early malware

The Morris worm, also referred to as the Internet Worm or Great Worm, stands out as one of the earliest and most notorious instances of malicious software. Created to exploit weaknesses in Unix systems, it swiftly spread through the nascent internet, infecting thousands of computers.

This worm spread by taking advantage of vulnerabilities in widely used network services like Sendmail and Finger, subsequently replicating itself on vulnerable systems. However, a programming flaw led to unintended consequences: infected machines were hit multiple times, resulting in unresponsiveness or crashes. This unintended effect significantly slowed down large sections of the internet.

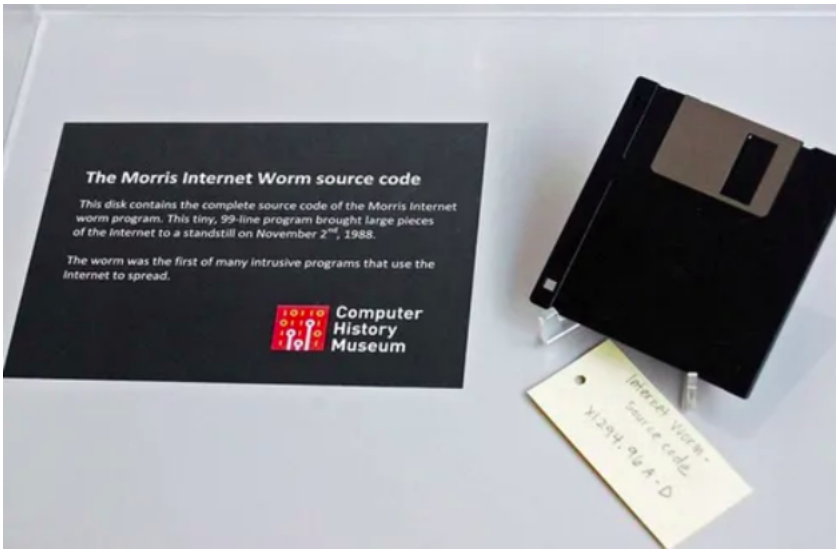

Figure 1.3: The Morris worm source code

3.2 Different types of malware

Malware	Overview
Ransomware	Ransomware refers to a type of software that employs encryption to block a victim's access to their data until a ransom is provided. Following infection, it often presents a notification to the victim, indicating that their files or system have been encrypted and requesting payment for their release.
Fileless malware	Fileless malware operates by manipulating essential operating system files like PowerShell or WMI, without the need for initial installation. Since these modified files appear legitimate to the operating system, antivirus software often fails to detect a fileless attack. Such malware employs tactics like code injection or process hollowing to remain hidden and execute malicious actions.
Spyware	Spyware gathers user information without their awareness or permission, encompassing sensitive data such as passwords, PINs, payment details, and unstructured messages. Its reach extends beyond desktop browsers to include critical applications and mobile devices.
Adware	Adware monitors a user's browsing behavior to tailor the ads they see. While it shares similarities with spyware, adware does not install software on a user's computer or capture keystrokes. However, it can affect device performance and diminish the user experience.
Worms	Worms exploit operating system vulnerabilities to infiltrate networks. They can infect systems through software vulnerabilities or via phishing or smishing attacks. Once embedded, worms can alter or delete files, introduce additional malicious software, or reproduce within the system until its resources are exhausted.
Trojan	A Trojan is a form of malware that masquerades as legitimate software, often disguising itself as native operating system programs or innocuous files such as free downloads. They are typically installed using social engineering tactics like phishing or bait websites.
Rootkits	A rootkit is a type of software that grants unauthorized remote access to a victim's computer, providing full administrative privileges to malicious actors. It can be inserted into applications, kernels, hypervisors, or firmware. Once operational, this malicious program establishes a backdoor exploit and may introduce further malware.

Malware	Overview
Keyloggers	Keyloggers record each keystroke made by users, enabling attackers to gather login credentials, financial data, and other confidential information.
Botnet	A Botnet is a network of computers infected with malware, controlled by a bot herder. This individual manages the botnet infrastructure, utilizing the compromised computers to execute attacks that may crash a target's network, inject malware, collect credentials, or perform CPU-intensive tasks.
Malspam	Malspam (malicious spam) delivers malware payloads through emails containing malicious content, such as virus or malware-laden attachments.
Scareware	Scareware deceives users into believing their computer is infected with a virus. Typically, users encounter scareware in the form of pop-up messages warning of an infection, aiming to coerce them into installing fake antivirus software. Once downloaded, this fake antivirus may introduce malware to the system.
Wiper	A Wiper is a type of malware designed solely to erase user data, making it irretrievable. Wipers are employed to disrupt computer networks in both public and private sectors. Threat actors also utilize wipers to conceal evidence following an intrusion, weakening the victim's ability to respond effectively.
Virus	A Virus cannot operate or replicate without the host application it has infected running. This reliance on a host application distinguishes viruses from trojans, which necessitate user downloads, and worms, which do not require applications to execute. Some instances of malware fit into multiple categories; for instance, Stuxnet is classified as a worm, virus, and rootkit.

[2]

Table 1.1: Different types of malware

3.3 Consequences of malware on data privacy and the stability of Windows computers

Malware poses significant threats to both data privacy and system stability. In this section, we will delve into the impact of malware on Windows-based systems.

3.3.1 Impact on Data Privacy

- **Data Theft**

  Malware such as spyware and keyloggers is engineered to clandestinely infiltrate systems and gather sensitive information without the user's awareness. This malicious software can monitor user activities, capture keystrokes, and log browsing history, providing attackers access to passwords, credit card numbers, and other personal data. The theft of such sensitive information compromises data privacy and exposes individuals to identity theft, financial fraud, and various cybercrimes.

- **Data Loss**

  In the event of a ransomware or wiper attack, victims without proper backups may find themselves unable to restore their data, resulting in permanent loss of critical files, documents, and personal information. These attacks can have severe repercussions for individuals, businesses, and organizations, leading to financial losses, operational disruptions, and reputational damage.

- **Data Leakage**

  Facilitated by various means such as backdoors, command-and-control channels, and covert communication protocols established by malware, data leakage enables attackers to siphon sensitive information from compromised systems and transfer it to remote servers. This includes intellectual property, confidential documents, customer records, and proprietary data. Such unauthorized disclosure poses a significant risk, potentially resulting in legal and regulatory penalties, as well as damage to trust and reputation.

3.3.2 Impact on System Stability

- **Performance Degradation**

  Malware consumes critical system resources such as CPU, memory, and disk space, resulting in performance degradation and slowdowns in processing speed. Resource-intensive activities by malware, such as continuous background scanning, file encryption, or network communication, can monopolize system resources, leading to slowness and unresponsiveness. System performance degradation may manifest as sluggish application launch times, delayed responses to user input, and an overall reduction in system operation efficiency, impacting productivity and user experience.

- **Software and System Corruption**

  Malware has the potential to corrupt files, alter system settings, and tamper with vital components of the operating system, resulting in software and system instability. File-corrupting malware may modify or delete crucial system files, applications, or configuration files, resulting in software malfunctions, application crashes, or system errors. Modification of system settings by malware can disrupt system functionality, compromise security configurations, and jeopardize system integrity, rendering it susceptible to further exploitation and compromise.

- **Network Disruption**

  Certain types of malware, such as worms and botnets, have the ability to spread rapidly across networks, causing congestion, bandwidth issues, and network downtime. Worms propagate by exploiting vulnerabilities in network services or by employing social engineering techniques to trick users into executing malicious code, leading to widespread infection and network disruption. Botnets, comprising compromised computers controlled by a centralized command-and-control infrastructure, can orchestrate coordinated attacks, such as distributed denial-of-service (DDoS) attacks, flooding network resources and resulting in service outages. Network disruption stemming from malware infections impacts overall system stability, disrupts business operations, and undermines the availability and reliability of network services.

4 Common Exploits and Vulnerabilities

Computer exploits are specialized programs or pieces of code crafted to take advantage of vulnerabilities or security flaws within software. In this section, we will delve into the intricacies of exploits, examining various types including Macro, HTA, CHM, JS-WWEB, and WinRAR exploits.

4.1 What is an exploit in terms of computer security?

An exploit is a type of program designed to target a specific vulnerability within software or hardware components. It can vary from a complete application to a simple sequence of code and data, or even just a sequence of commands.

In essence, an exploit serves as a tool that allows a hacker to leverage a security flaw for their benefit. Any program or code that can be programmed to take advantage of a vulnerability within software or hardware qualifies as a security exploit.[4].

What is the difference between exploits and vulnerabilities?

Vulnerabilities and exploits are interconnected, although they have distinct meanings. Vulnerabilities pertain to weaknesses in software applications. However, not all vulnerabilities can be used to inject malware into targeted computers. Some vulnerabilities are not exploitable due to the presence of other security measures. In 2019, a dangerous vulnerability called Bluekeep was uncovered in Windows 7, prompting the NSA to issue a security warning.

Exploits, on the other hand, are attacks that take advantage of software vulnerabilities to cause undesirable effects on targeted systems, such as the introduction of malicious software or unauthorized access by hackers. The existence of a vulnerability does not pose an immediate threat unless someone discovers how to create an exploit for it. However, once a vulnerability is identified, it is almost certain that someone will attempt to develop an exploit for it.

Let's imagine the program as a house. The doors are securely closed, but someone left a window open on the second floor. This is a vulnerability. If the thief (that is, the hacker) wants to exploit this vulnerability to break into the house, they need to use an exploit, like a ladder. When they use the ladder to reach the second floor, the thief can exploit that

open window and get in.

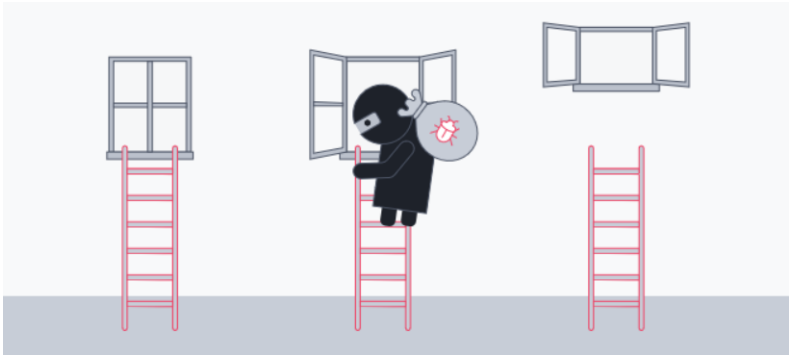

Figure 1.4: An illustration that explains the concept of exploits

In the illustration above, the left window is locked, so there are no vulnerabilities there. The right window is open and vulnerable, but it's placed too high to exploit. The middle window is open, vulnerable, and it's close enough to the ground to be exploitable.

4.2 Some examples of exploits targeting Windows or Microsoft applications

Exploit	Overview	Consequences
Macro	Macro Exploits involve the manipulation of macros within Microsoft Office applications to execute harmful code. Originally designed to automate tasks, macros can unwittingly serve as a vector for malware distribution. Attackers exploit vulnerabilities within macros to infiltrate systems and carry out malicious actions.	Macro viruses, embedded within documents, can rapidly spread via email attachments or compromised files. Upon activation, they have the potential to compromise entire systems, resulting in data theft, malware installation, and widespread disruption. Notable incidents such as the Melissa Virus underscore the significant financial and operational damages caused by macro exploits.

Exploit	Overview	Consequences
HTA	HTA Exploits revolve around the exploitation of HTML Applications (HTAs) in Windows environments. HTAs, capable of executing code with elevated privileges, provide attackers with the means to deploy malicious scripts and compromise systems.	Vulnerabilities in HTAs allow attackers to create malicious files, distribute them through various channels, and execute harmful code on victims' systems. Exploited vulnerabilities may lead to unauthorized access, malware deployment, and data theft. High-profile cases like CVE-2017-0199 highlight the severity of HTA exploits and their potential impact on system security.
CHM	CHM Exploits target Compiled HTML Help (CHM) files, commonly used for storing documentation and resources. Attackers exploit vulnerabilities within CHM files to execute malicious code and compromise system security.	By crafting malicious CHM files and distributing them through spear-phishing or other methods, attackers can exploit vulnerabilities in Windows systems. These exploits can result in the execution of malicious scripts, installation of malware, and unauthorized access to victim systems. Noteworthy instances like CVE-2017-8570 illustrate the real-world implications of CHM exploits and the potential for significant data breaches and system compromise.
JS-WEB	JS-WEB Exploits encompass security vulnerabilities associated with JavaScript execution within web environments. Malicious JavaScript code can exploit flaws in web applications, leading to various forms of cyberattacks.	JavaScript vulnerabilities, such as Cross-Site Scripting (XSS) and Clickjacking, enable attackers to execute malicious code within users' browsers. These exploits may result in compromised user data, unauthorized access, and the dissemination of malware. The manipulation of the Document Object Model (DOM) presents additional risks, allowing attackers to covertly alter web page behavior and deceive users.

Exploit	Overview	Consequences
WinRAR	WinRAR Exploits target vulnerabilities in the WinRAR file compression software, enabling attackers to execute malicious code on targeted systems. Vulnerabilities like CVE-2018-20250 expose users to potential security risks when handling compressed files.	Exploiting vulnerabilities such as CVE-2018-20250 permits attackers to create malicious compressed files that, upon decompression, execute harmful code on vulnerable systems. These exploits can lead to malware installation, system compromise, and data breaches.

Table 1.2: Different types of exploits targeting Windows

5 Conclusion

As we conclude this chapter and prepare to transition to the next, we have gained a comprehensive understanding of the myriad threats that jeopardize the security of Windows computers. We have explored a diverse array of attacks that specifically target Windows systems, including the exploitation of vulnerabilities. Additionally, we have ventured into the intricate realm of malicious software tailored for Windows, discerning the distinct characteristics of ransomware, spyware, and other harmful programs.

Armed with this knowledge, we are now poised to delve deeper into the realm of cyber threats in Chapter 2. By further analyzing various attacks and malware that target Windows computers, our aim is to fortify our defenses and effectively mitigate risks.

CHAPTER 2

ATTACKS AND MALWARE TARGETING WINDOWS COMPUTERS

Contents

1 Introduction

Windows computers continue to be highly sought-after targets for malicious actors worldwide. Given their extensive use in both personal and professional settings, they offer numerous opportunities for a range of sophisticated attacks and malware. This chapter has explored the diverse threats that exist within Windows systems, from techniques for exploiting vulnerabilities to malware crafted specifically to infiltrate these environments.

2 Presentation of Different Attacks Targeting Windows Computers

In the upcoming section, we will delve into an array of attacks targeting the Windows operating system. From the brute force of DOS/DDoS assaults to the covert tactics of man-in-the-middle attacks, and from the deceptive lures of phishing attempts to the exploitation of vulnerabilities, we will navigate through the diverse strategies employed by threat actors in their quest to compromise Windows systems.

2.1 DOS/DDOS attacks

2.1.1 What is a denial-of-service attack?

A denial-of-service (DoS) attack is a type of cyber attack where a malicious actor seeks to make a computer or device unavailable to its intended users by disrupting the normal functioning of the device. In a DoS attack, a single computer or network of computers is used to flood the targeted system with an overwhelming volume of requests, resulting in a denial of service to legitimate users. This flood of requests consumes the system's resources, such as bandwidth, processing power, or memory, rendering it incapable of responding to legitimate user requests.

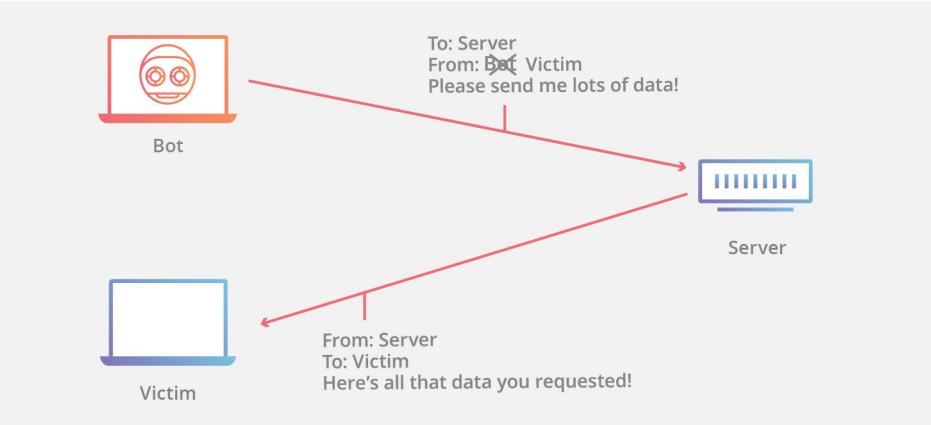

Figure 2.1: DOS attack

A distributed denial-of-service (DDoS) attack occurs when numerous sources, like a botnet, launch a form of DoS attack. A botnet comprises computers infected with malware and controlled by a malicious actor. Every infected device, referred to as a bot, adds to the assault by flooding the target with requests. These botnets can serve diverse malicious objectives, such as spamming, data theft, ransomware, fake ad clicks, and executing DDoS attacks.

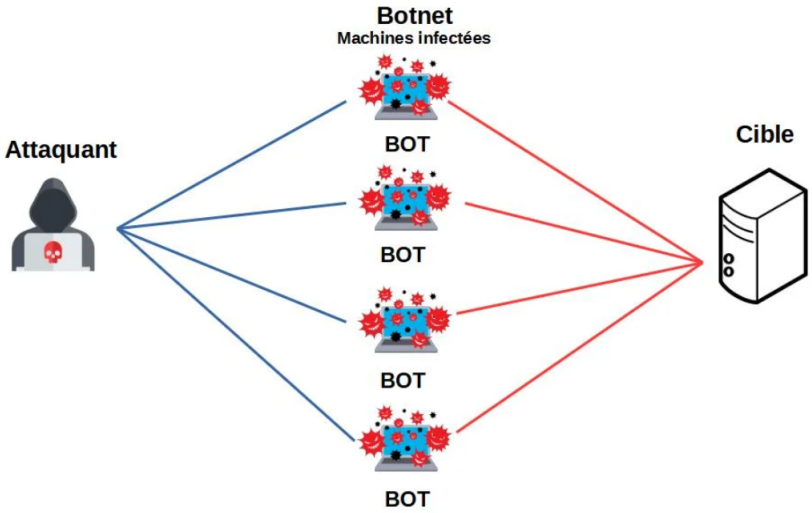

Figure 2.2: DDOS attack
[18]

2.1.2 Types of DDoS attacks

2.1.2.1 The SYN Flood Attack

SYN flood attacks work by exploiting the TCP connection handshake process.The TCP Handshake is a three-step process used by a client to establish a connection with a server. It begins with the client sending a SYN (synchronize) packet to the server, signaling its intent to start a connection. The server then acknowledges the request with a SYN-ACK (synchronize-acknowledgment) packet, indicating its readiness to establish the connection. Finally,the client completes the handshake by sending an ACK (acknowledgment) packet back to the server, establishing the connection.

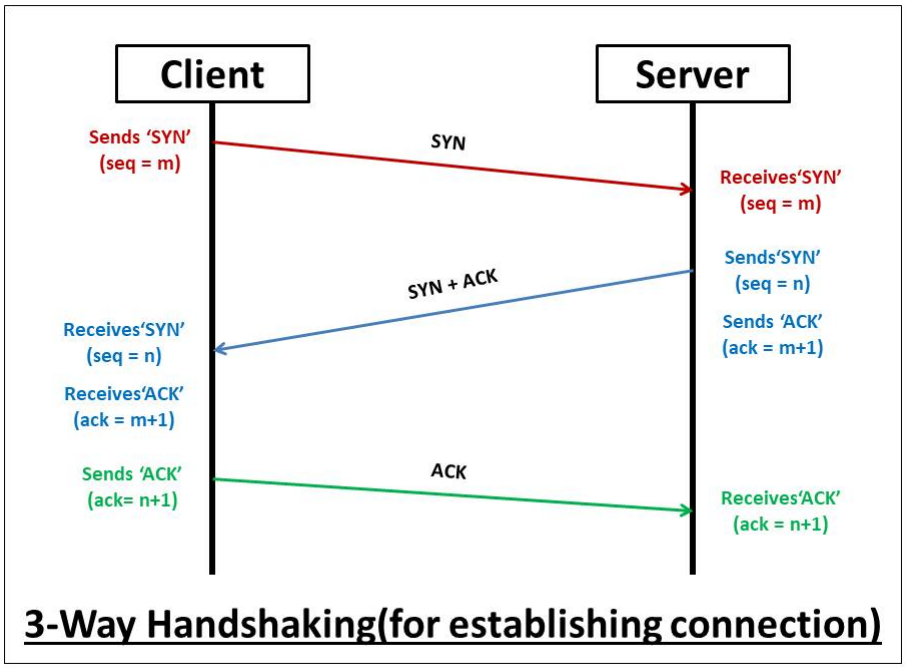

Figure 2.3: TCP connection

However, in a SYN Flood Attack, which is an example of an Application-Layer Attack, this process is disrupted. In this attack, the attacker overwhelms the targeted server with a large number of SYN packets but intentionally does not respond to the server's SYN-ACK replies. As a result, the server is left waiting for the final ACK packet to complete the handshake and establish the connection. By withholding this crucial packet, the attacker ties up the server's resources, such as memory and CPU processing power, as it waits for these incomplete connections to time out. This situation leads to resource exhaustion as the server grapples with an inundation of half-open connections. Consequently, legitimate

users trying to establish connections are unable to do so because of the overwhelmed resources of the server. The consequence is a server that becomes sluggish or unresponsive to genuine requests, effectively denying service to legitimate users.

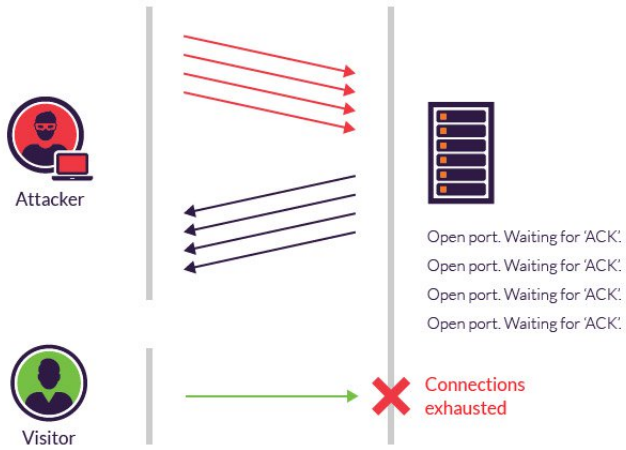

Figure 2.4: SYN flood attack

When a SYN Flood attack occurs, 'netstat' or 'ss' displays a large number of connections with the 'SYN-RECV' state.

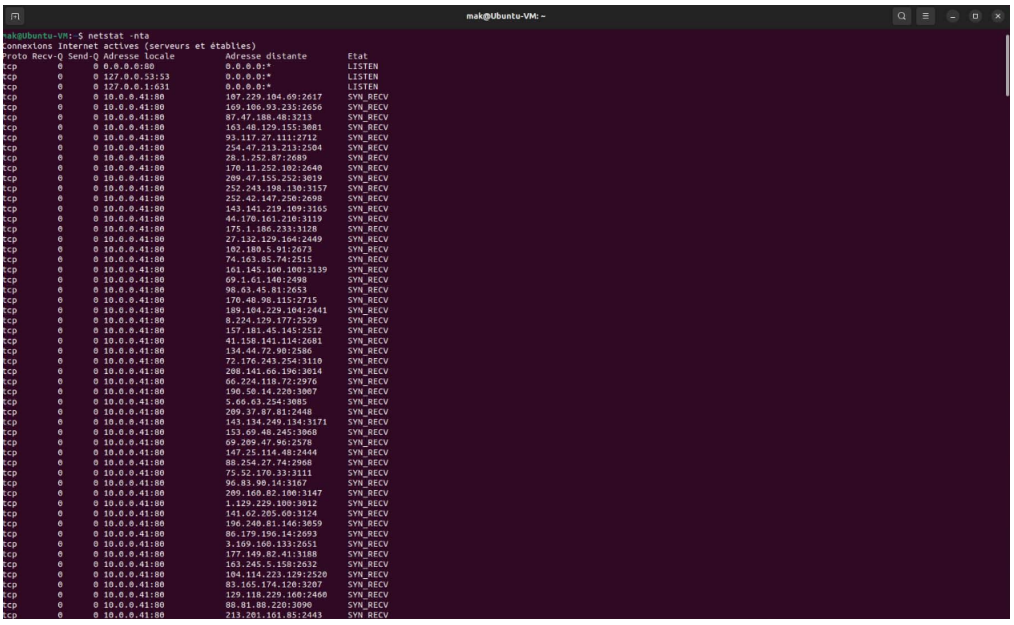

Figure 2.5: High volume of connections in the 'SYN-RECV' state

In essence, to execute a SYN Flood-type DoS attack, an attacker capitalizes on the

server's response pattern: upon receiving an initial SYN packet, the server replies with one or more SYN/ACK packets, anticipating the final ACK packet to complete the handshake—a step the attacker deliberately withholds. This flooding of unanswered requests can saturate the network layer of the operating system, rendering it incapable of processing new connections. Additionally, if the server becomes overwhelmed, system resources such as CPU and memory can be entirely consumed [19].

2.1.2.2 The UDP Flood Attack

The User Datagram Protocol (UDP) serves as a fundamental component of Internet communication, providing a lightweight and efficient method for transmitting data packets. Unlike the Transmission Control Protocol (TCP), UDP operates without the need for a connection establishment phase, allowing for faster data transmission at the cost of reliability and error checking. It is specifically chosen for time-sensitive applications such as gaming, video streaming, or Domain Name System (DNS) queries. UDP allows for faster communication as it does not take the time to establish a solid connection with the destination before transferring data. Since establishing the connection takes time, eliminating this step speeds up the data transfer.

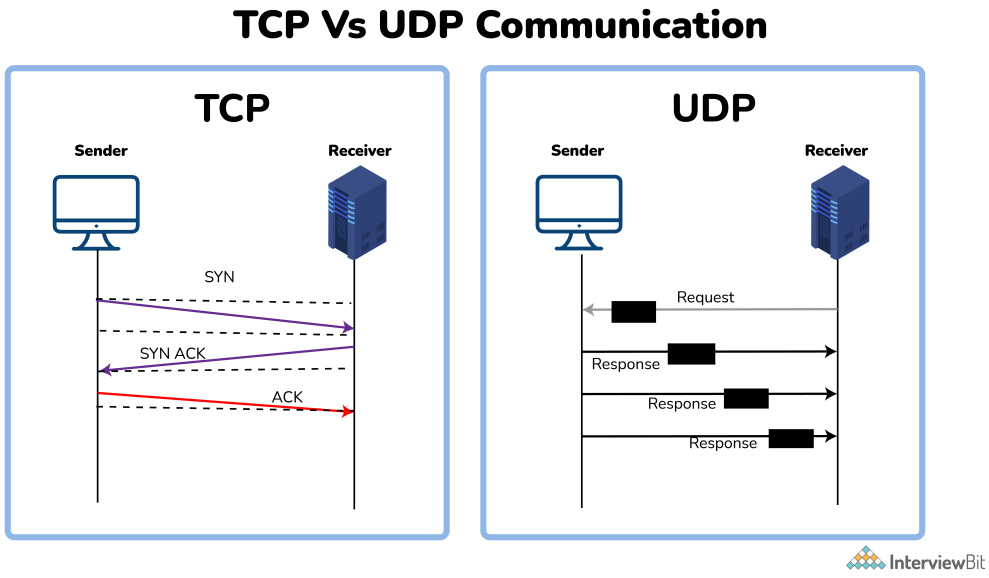

Figure 2.6: TCP and UDP communication

However, the UDP protocol can also lead to the loss of data packets between the source and destination. It can also facilitate the execution of a Distributed Denial of Service (DDoS) attack by a hacker. In these malicious activities, bad actors flood a targeted system

with an overwhelming deluge of UDP packets, exploiting its connectionless nature. This flood of packets overwhelms the target system's resources, resulting in network congestion, severe slowdowns, and ultimately, denial of service (DoS) to legitimate users attempting to access the affected service or system. When a server receives a UDP packet on a specific port, it undergoes two steps in response. First, it checks for any active programs currently listening for requests at the designated port. If no programs are found to be receiving packets on that port, the server then sends an ICMP (ping) packet back to the sender, notifying them that the destination is unreachable.

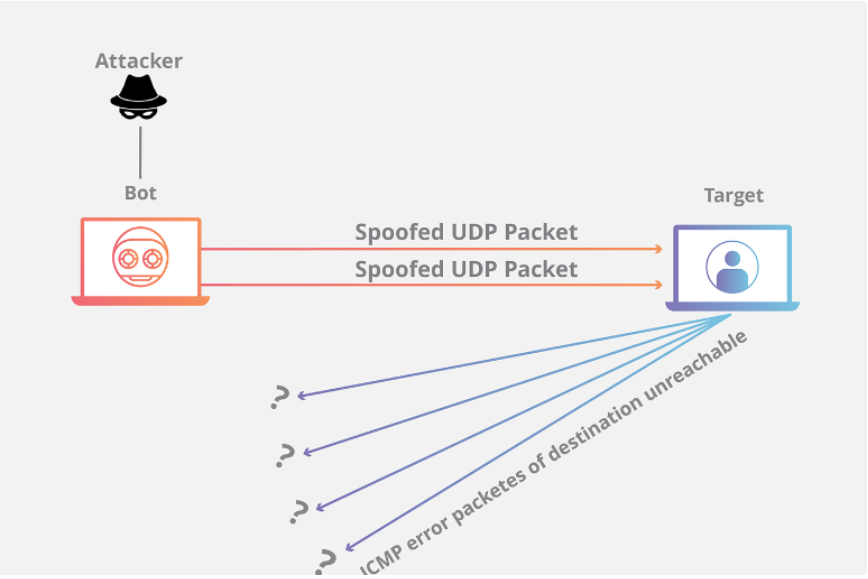

Figure 2.7: UDP flood attack

Suppose we have a target server with an online game service listening for UDP packets on port 12345. The game server, configured to receive UDP packets on this port, becomes the focus of a malicious UDP flood attack initiated by an attacker using a "booter" or "stressor" tool. This attacker floods the server with a massive volume of spoofed UDP packets, quickly overwhelming the server's bandwidth, memory, and processor. As a result, the game server struggles to handle the influx of packets, leading to a denial of service (DoS) situation. Legitimate players attempting to connect and play experience significant disruptions, such as delays, frequent disconnections, or an inability to access the server altogether. In response to the flood of packets, the server, unable to process them all, may send ICMP packets to the attacker indicating that the port is "unreachable" or "inaccessible," marking the success of the attack in saturating and disrupting the game server's service.

Figure 2.8: Example of an error message following a UDP flood attack [20]

2.1.2.3 The HTTP Flood Attack

An HTTP request, or Hypertext Transfer Protocol, serves as the standard mechanism through which web browsers and servers communicate to exchange data on the Internet. Essentially, when a user accesses a website, their browser sends HTTP requests to the server to retrieve various resources needed to display the page, such as images, scripts, or CSS files.

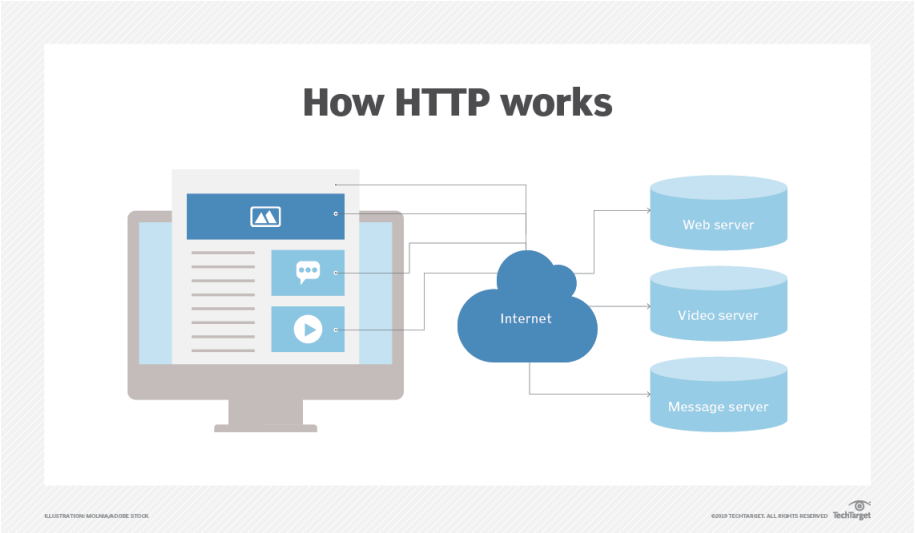

Figure 2.9: How HTTP works

The HTTP Flood attack serves as an illustration of an Application-Layer Attack, posing a significant risk to the availability of web servers. Its aim is to overwhelm server resources by generating a large volume of simultaneous HTTP requests. Perpetrators of this cyberattack frequently exploit botnets, which are networks of compromised computers, to coordinate the influx of requests. The main goal is to exhaust the resources of the targeted server, potentially leading to significantly delayed response times or even complete unavailability of the website. What sets HTTP Flood apart from other Denial of Service (DoS) attacks is its use of HTTP requests, enabling it to imitate legitimate traffic and thus complicating detection.

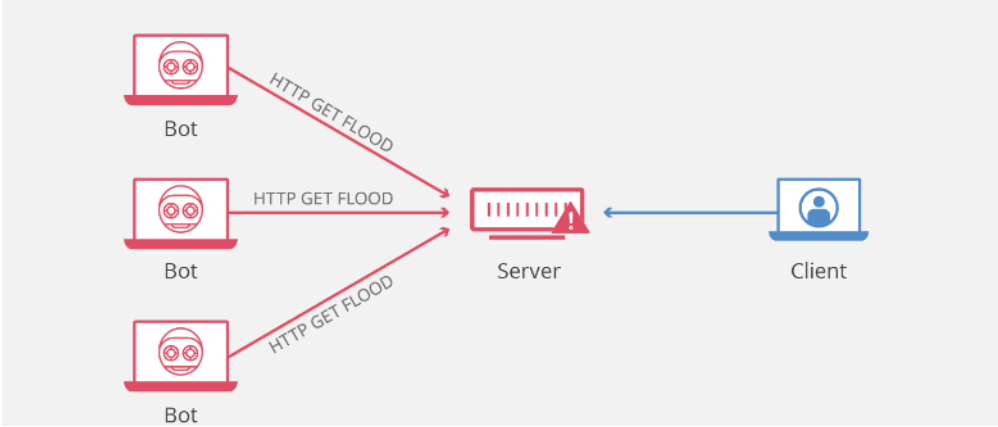

Figure 2.10: HTTP flood attack

This attack method can take various forms, such as intensive repetition of legitimate GET or POST requests, or malformed requests designed to overwhelm the server's processing mechanisms.

HTTP GET attack

In this type of attack, numerous computers or devices are synchronized to send a multitude of requests for images, files, or other resources from a specific server. As the targeted server becomes overwhelmed with the influx of incoming requests and responses, it becomes unable to process additional requests from legitimate sources of traffic, leading to a denial-of-service situation.

HTTP POST attack

Usually, when a form is submitted on a website, the server is tasked with processing the incoming request and storing the data in a persistence layer, commonly a database. Managing the form data and executing the requisite database operations can be consider-

ably resource-intensive compared to the processing power and bandwidth needed to send the initial POST request. Exploiting this difference in resource consumption, the attack involves bombarding the targeted server with numerous POST requests until its capacity is overwhelmed, resulting in a denial-of-service situation.

2.1.2.4 The DNS Flood Attack

The Domain Name System (DNS) Flood attack is an example of a Volume-Based or Volumetric Attack. It is a malicious attempt to disrupt the normal operation of a DNS server by overwhelming it with an excessive volume of DNS queries. The DNS system serves as the backbone of the internet by translating human-readable domain names into IP addresses that computers understand. In a DNS Flood attack, attackers aim to flood the targeted DNS server with an overwhelming number of requests, exhausting its resources and leading to service degradation or complete unavailability. This type of attack can be particularly effective because DNS servers are often not designed to handle extremely high query rates.

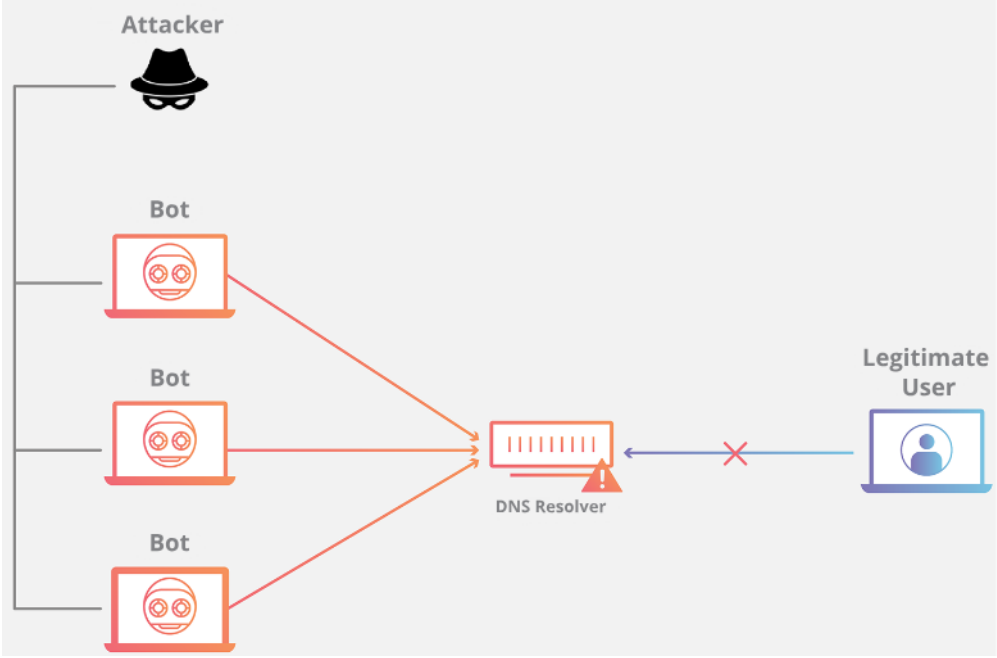

Figure 2.11: DNS flood attack

2.2 Man In The Middle attack

2.2.1 Definition

A man-in-the-middle (MITM) attack is used by hackers to insert themselves into the communication between a user and an application. By doing this, they can secretly intercept and potentially manipulate the data exchange without anyone knowing. The main goal of a MITM attack is to secretly gather personal information like login details, financial account information, or credit card numbers. The targets of these attacks are typically users of online banking, software services, e-commerce websites, and any platform that requires user authentication.

The information obtained during a MITM attack is a powerful tool for various malicious activities, including identity theft, unauthorized financial transactions, and even secret changes to account settings.

Imagine a MITM attack as a situation where a sneaky mail carrier secretly opens your bank statement, carefully writes down your account information, seals the envelope without leaving any evidence, and then delivers it to your house. This is a hidden invasion of your privacy and security, completely unnoticed by the unsuspecting receiver.[7] [8] [9]

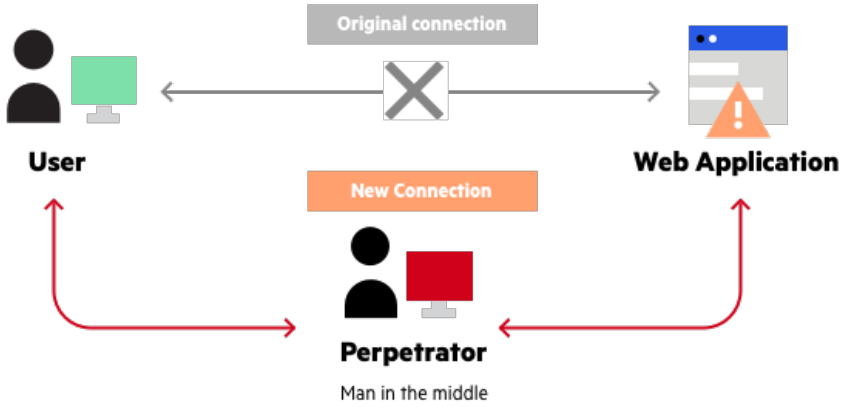

Figure 2.12: Man-In-The-Middle attack illustration

2.2.2 Execution

To execute a MITM attack, hackers employ a variety of malicious techniques. One common and uncomplicated method is a passive attack, where the attacker sets up unsecured,

harmful WiFi hotspots for public use. These hotspots are usually named after their location and do not require a password. When a person connects to one of these hotspots, the attacker can fully observe any online data transmission.
To adopt a more proactive method of intercepting, attackers have the option to initiate any of the following attacks:

2.2.2.1 ARP poisoning

In the context of MITM attacks, ARP poisoning serves as the initial step to intercept communication between the victim and the legitimate network gateway, such as a router.

ARP, or the Address Resolution Protocol, is a method used to convert IP addresses into MAC addresses. When devices in a network need to communicate, they send out ARP-queries to discover the MAC addresses of other machines.

ARP poisoning, also known as ARP spoofing, involves manipulating the ARP cache of devices on a local network. By sending spoofed ARP messages, the attacker associates their MAC address with the IP address of the router or another legitimate device on the network. This causes traffic intended for the legitimate device to be redirected to the attacker's machine.

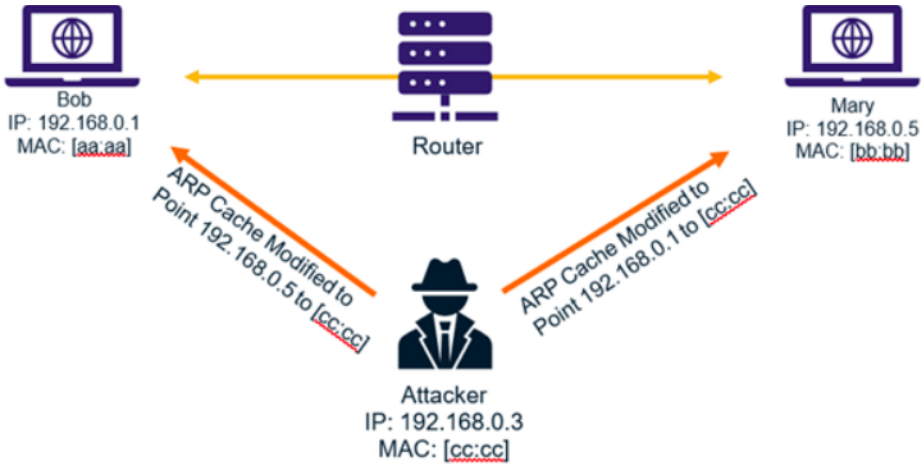

Figure 2.13: ARP spoofing attack

2.2.2.2 IP spoofing

IP spoofing occurs when a malicious actor manipulates the original IP address of a packet, typically substituting it with a false IP address, often to make it appear as if the traffic

originates from a legitimate source. Conversely, hackers can also use IP spoofing to conceal the actual destination's IP address. The underlying mechanism enabling IP spoofing on the internet lies in the structure of data transmission.

Data traffic on the internet is fragmented into packets for transmission and reception. These packets are sent individually and reassembled at their destination, which could be a recipient's device or a server hosting a website. Each data packet carries an IP header containing crucial information such as the source and destination IP addresses. Under normal circumstances, these data packets are transmitted using the TCP/IP protocol.

However, the TCP/IP protocol contains a vulnerability. It requires the completion of a three-way TCP handshake to facilitate the exchange of information between two parties. The process unfolds as follows:

1. The source device initiates the handshake by sending a SYN (synchronize) message to the receiver, establishing a connection and enabling both devices to synchronize their sequence numbers.
2. Upon receiving the SYN message, the receiver responds with an ACK (acknowledgment) message, confirming the receipt of the SYN message.
3. Finally, the source device sends a SYN-ACK message back to the receiver, affirming the establishment of a secure connection.

In an IP spoofing attack, the hacker intervenes during the TCP handshake, typically before step 3, where the source device sends its SYN-ACK message. Instead of allowing the legitimate source to complete the handshake, the hacker sends a counterfeit confirmation, incorporating their device's address (MAC address) and a falsified IP address representing the original sender. Consequently, the receiver is misled into believing that it has established a connection with the legitimate sender when in reality, it is communicating with an IP address that has been spoofed.

After a successful attempt of being a man in the middle, what can hackers do?

Once the attacker has successfully established an ARP or IP spoofing attack, they can employ packet sniffing techniques to capture and analyze the traffic passing between the victim and the legitimate network gateway. This allows the attacker to intercept sensitive information, such as login credentials, financial data, or other confidential information exchanged between the victim and the network.

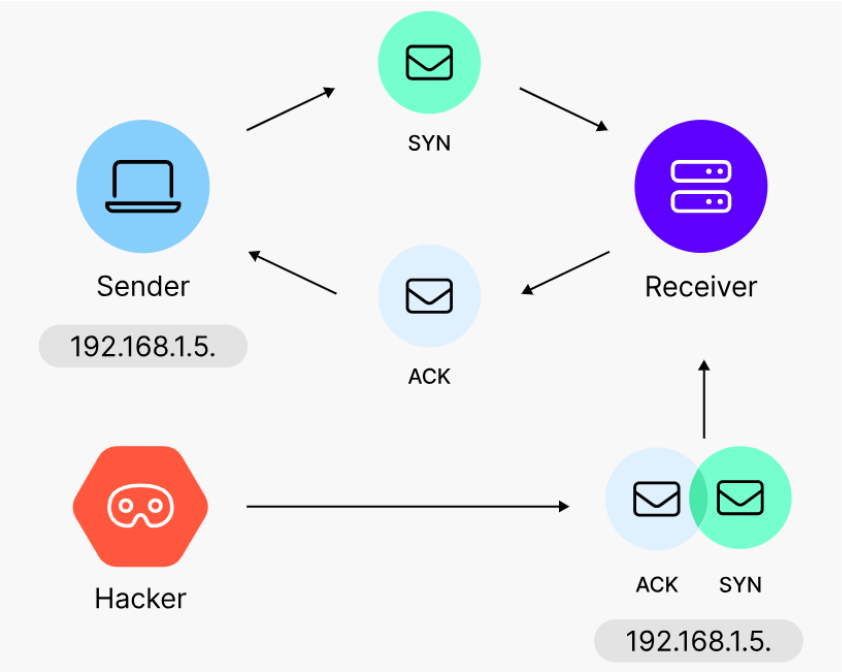

Figure 2.14: IP spoofing illustration

2.2.3 Packet sniffing

2.2.3.1 A case where the victim visits an HTTP website

When a user visits an HTTP website, their browser sends a request to the web server over the network. This request is transmitted in plain text, making it susceptible to interception by attackers. For example, an attacker on the same local network as the victim can execute ARP poisoning to intercept traffic between the victim's device and the router. By poisoning the ARP cache, the attacker redirects traffic through their machine. He can then use packet sniffing tools such as Wireshark to capture and analyze the plaintext HTTP requests and responses exchanged between the victim and the web server. This allows the attacker to view any sensitive information transmitted over HTTP, such as login credentials or personal data.

2.2.3.2 A case where the victim visits an HTTPS website

When a user visits an HTTPS website, their browser establishes a secure connection with the web server using the Transport Layer Security (TLS) protocol. This encryption ensures that the data exchanged between the client and server is protected from eavesdropping or tampering.

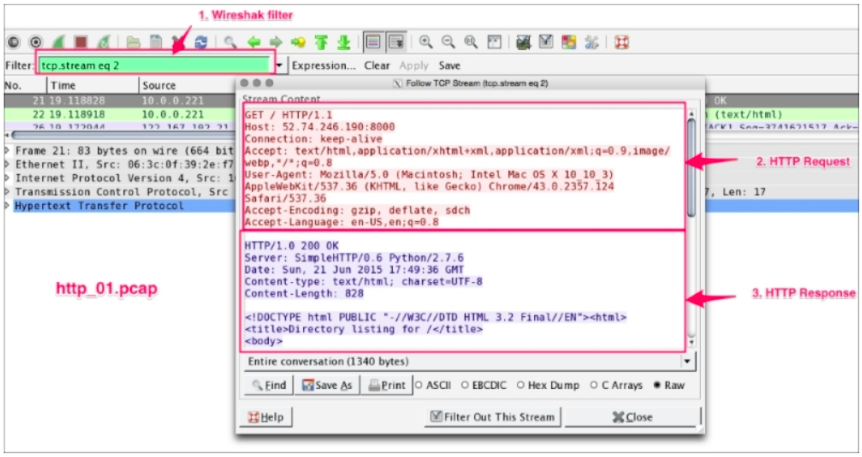

Figure 2.15: An example of an HTTP request analyzed by Wireshark

During the TLS handshake, the web server presents its digital certificate to the client, which contains its public key and is signed by a trusted Certificate Authority (CA). The client verifies the certificate's authenticity to ensure it's from a legitimate source.

Compromised CA or Invalid Certificate

In some MITM attack scenarios, the attacker may compromise a trusted CA or present the victim with an invalid or forged digital certificate. This can occur through various means, such as exploiting vulnerabilities in the certificate issuance process or using fraudulent certificates. If the attacker succeeds in presenting the victim with a fake or compromised certificate, the victim's browser may trust it and establish a secure connection with the attacker's server instead of the legitimate web server.

With the victim's traffic routed through the attacker's server, the attacker can decrypt the encrypted HTTPS traffic using the private key associated with the fake certificate. This allows the attacker to inspect and potentially modify the traffic before forwarding it to the legitimate web server, creating a seamless man-in-the-middle interception scenario even with HTTPS encryption in place.

2.3 Phishing attacks

As internet usage continues to surge, individuals are increasingly sharing their details online, making a vast amount of sensitive information and financial transactions susceptible to cybercriminals. Among the many forms of cybercrime, phishing stands out as a highly

effective method that allows criminals to deceive users and pilfer crucial data. Originating from the first reported phishing attack in 1990, this tactic has since evolved into a more sophisticated and pervasive threat. Today, phishing ranks among the most prevalent forms of fraudulent activity on the internet, capable of inflicting significant losses on victims, including identity theft, exposure of sensitive information, and compromised corporate or governmental secrets.

2.3.1 What is a Phishing attack

Phishing is One of the social engineering crimes that allow the attacker to perform identity theft. This is often done by impersonating a reputable organization through email or other means, leading the user to trust the message and disclose their details to the attacker. In phishing attacks, the phisher typically sends emails containing malicious links that redirect users to harmful websites. Additionally, attackers may use Voice over IP (VoIP), Short Message Service (SMS), or Instant Messaging (IM) to carry out their schemes. Phishers have also moved towards more targeted approaches, known as "spear-phishing," where they tailor emails to specific individuals rather than sending mass messages to random targets.

2.3.2 General phishing attack process

Figure 1 illustrates the typical flow of a phishing attack, consisting of four distinct phases. These phases are detailed in the Proposed Phishing Anatomy. In many instances, the phishing process begins with the collection of information about the target. Following this, the attacker determines the most suitable method for the attack during the planning phase. The subsequent stage, known as the preparation phase, involves the attacker scouring for vulnerabilities to exploit and ensnare the victim. The actual attack takes place in the third phase, during which the phisher patiently awaits a response from the target. Subsequently, in the final phase known as the valuables acquisition stage, the attacker can reap the rewards of their efforts.

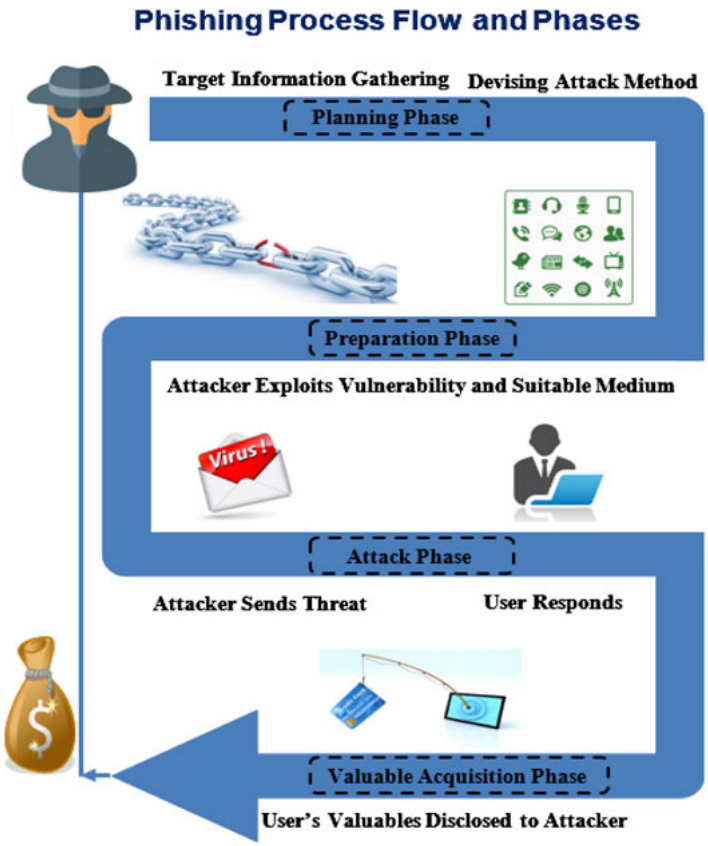

Figure 2.16: General phishing attack process

2.3.3 Real-World Phishing Examples

This part delves into real-world instances of phishing attacks to shed light on the intricate nature of recent schemes. Illustrated in Figure 2 is a snapshot of a suspicious phishing email that managed to circumvent a University's spam filters and land in the recipient's inbox. The email employs a tactic of urgency or importance in its subject line, using the term 'important,' aiming to elicit a psychological response from the recipient to compel them to click on the "View message" button. Notably, the email contains an embedded button that raises suspicion; upon hovering over it, the displayed Uniform Resource Locator (URL) does not correspond to the claimed destination. Additionally, the sender's address raises doubts, as it is unfamiliar to the recipient. Clicking on the deceptive attachment button could lead to the installation of malicious software, such as viruses or worms, onto the recipient's device. Alternatively, it might redirect the victim to a counterfeit login page, enabling the phisher to acquire the victim's credentials.

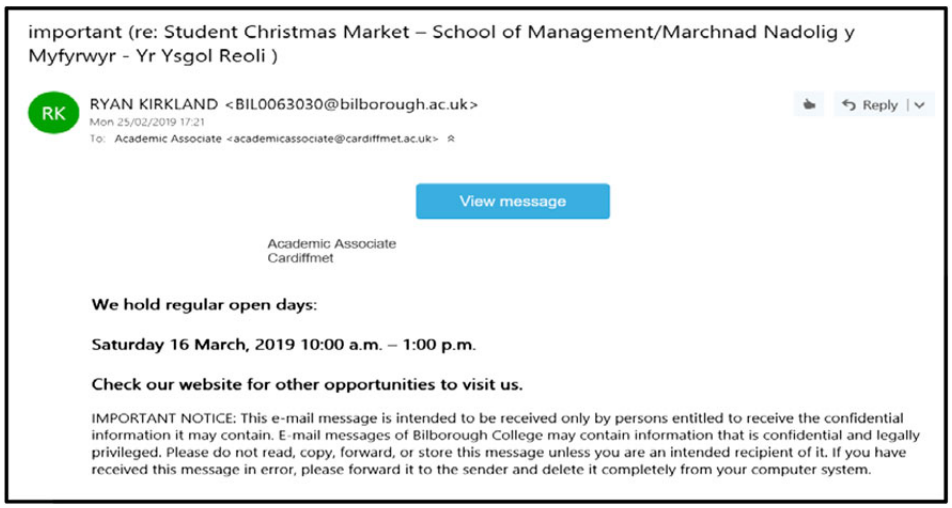

Figure 2.17: Screenshot of a real suspicious phishing email received by the authors' institution in February 2019

In recent times, Scammers have exploited the COVID-19 pandemic, circulating fraudulent messages themed around the virus. They targeted people's anxieties and urgent need for information, focusing on PPE-like facemasks due to high demand.
Displayed in Figure 3 is an instance of a phishing email, in which the perpetrator posed as the recipient's neighbor. The email presented a false narrative of the sender being infected and on the brink of death due to the virus.

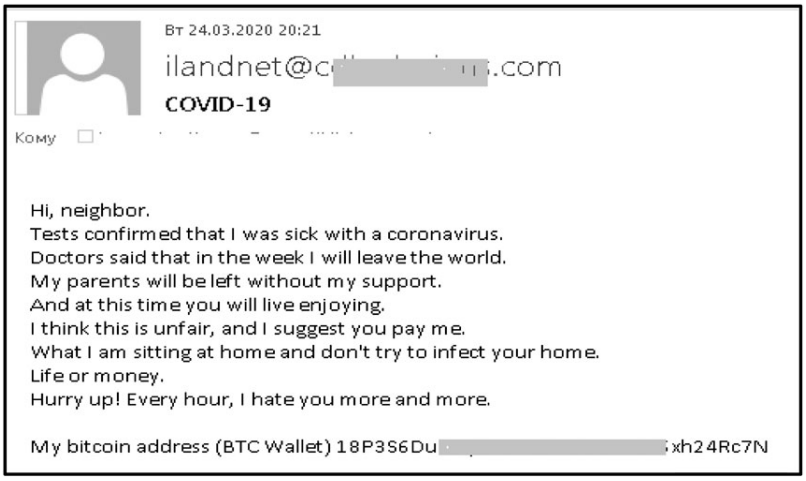

Figure 2.18: Screenshot of a coronavirus-related phishing email

In January 2019, a phishing attack was uncovered by an Akamai security researcher. The attack used Google Translate to hide suspicious URLs under the legitimate-looking

"www.translate.google.com" address, tricking users into logging in. These scams included requests for Netflix payment details and fake PayPal login pages in promoted tweets. The absence of HTTPS and URL misspellings were telltale signs of phishing. Figure 4A shows a phishing email received by the Federal Trade Commission (FTC), prompting users to update their payment information for Netflix due to billing issues.

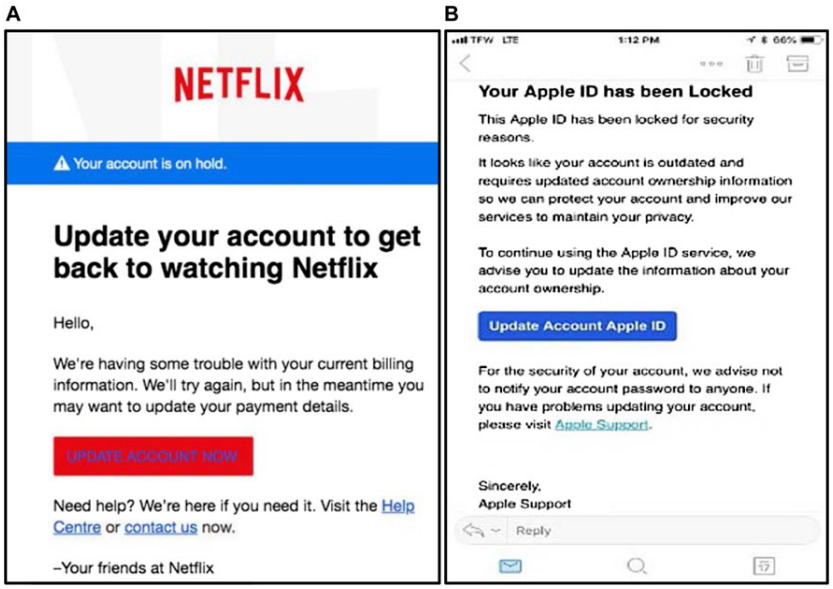

Figure 2.19: Screenshot of (A) the Netflix scam email and (B)fraudulent text message(Apple)

[21] [22] [23]

2.4 Vulnerability exploitation

2.4.1 Introduction

In this segment, we'll embark on a comprehensive exploration of vulnerability exploitation, a critical facet of cybersecurity. We'll delve into the mechanisms behind some of the most prevalent attack vectors, including those targeting WinRAR, JS-web, macros, HTA, and CHM files.

2.4.2 Macro Exploits

What is a macro?

A Macro in the context of Microsoft Office applications refers to a set of instructions or commands that can be recorded and saved for later use to automate tasks. The macros are designed to make the user's life easier by speeding up or eliminating repetitive tasks. However, like any software that lets users run customized scripts in the background, Office suites can be exploited by attackers to execute harmful code and compromise victims. Typically, the macro-malware plays a role as a loader in the infection process. It will first download and execute another payload, and then end its operation. Popular programs that use macros heavily include Microsoft Word, Excel, PowerPoint, and other apps in the Microsoft Office suite[5].

How does a macro virus work?

To activate macro viruses, cybercriminals create malicious code, and then insert this code into documents. When a user opens an infected document and enables macros, the malicious code embedded in the macros is executed. This code can perform a variety of harmful actions, such as installing malware, encrypting files, or stealing sensitive information.

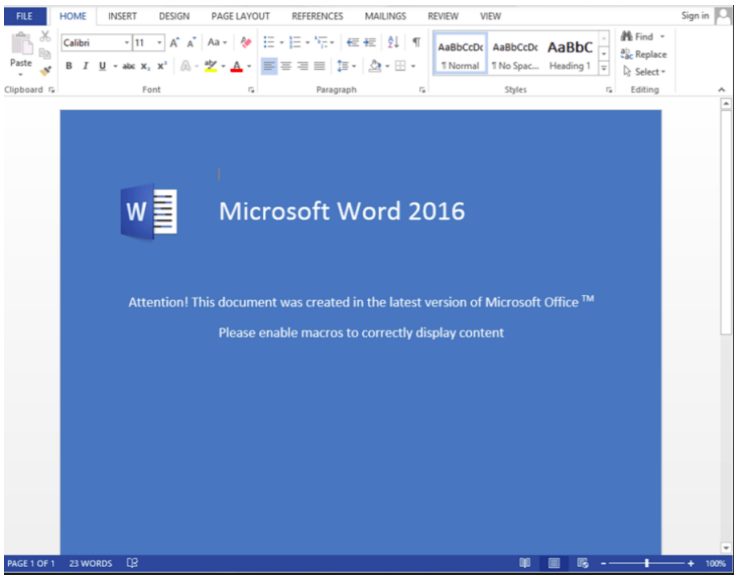

Figure 2.20: An example of a common MS Office macro lure

How do macro viruses spread?

Many macro viruses spread as infected email attachments, while others spread by Downloading infected files from a network, or by Inserting physical media, like a flash drive or CD, that contains an infected file.
Once activated, the malware can infiltrate other Office files on the victim's computer or even the Office file templates themselves. This compromises every document created on the machine, leading to an increased chance of user error. Furthermore, macros are written in a macro language and can run on different platforms, making their spread even more effortless. Threat actors might use techniques such as obfuscating to guarantee that their malware won't be detected by firewalls or IDSs (Intrusion detector systems). Obfuscating macro-malware is relatively simple. Online tools are readily available to hide malicious code snippets within other file components, such as text labels or Excel cells, to avoid detection. Additionally, macro threats can operate "off the land" by solely using common tools to execute code in memory, without writing anything to the disk.

Figure 2.21: Macro attack illustration

The Melissa Virus (1999)

The Melissa Virus made global headlines in 1999 as a rapidly spreading email virus targeting Microsoft Word and Microsoft Outlook. The attack started when its creator posted a list, supposedly containing passwords to several adult websites, in an online community. When victims downloaded and opened the file in Word, they triggered the virus instead.

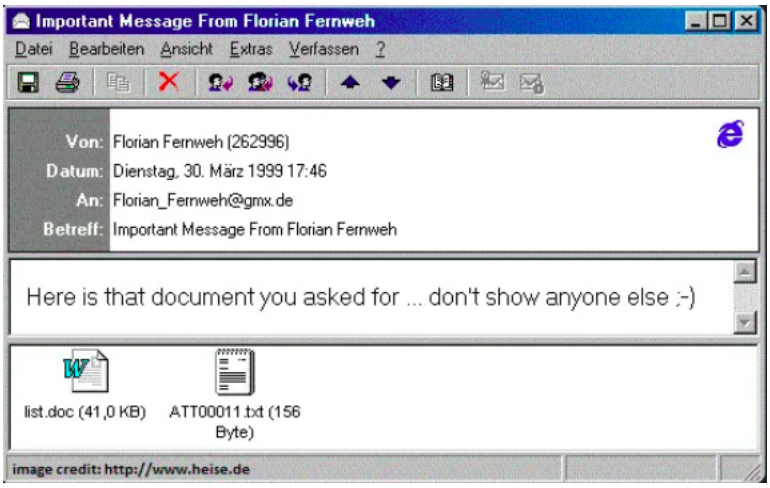

Figure 2.22: The E-mail that was sent to victims during the Melissa macro attack

The Melissa Virus's macros caused Microsoft Outlook to mass-mail the infected list to the first 50 contacts in the victim's address list. When those people opened the file, the virus infected their computers, and the cycle repeated — ultimately inflicting an estimated 80 million dollars in damages.

2.4.3 HTA Exploits

What is an HTA?

An HTML Application (HTA) is a program specifically designed for Microsoft Windows, providing a versatile platform for creating rich user interfaces and executing code with elevated privileges. HTAs are crafted using a combination of HTML, Dynamic HTML, and scripting languages supported by Internet Explorer, such as VBScript or JScript. Unlike traditional web applications, HTAs have the `.hta` extension and are not subject to the security constraints imposed by web browsers. Instead, they are considered "fully trusted" applications, capable of executing code without limitations.

HTAs can be executed using the program `mshta.exe` or by simply double-clicking on the file. `mshta.exe` is typically bundled with Internet Explorer and serves as the engine responsible for launching HTAs. Upon execution, `mshta.exe` invokes the Internet Explorer rendering engine (mshtml) and any necessary language engines, enabling the HTA to run seamlessly within the Windows environment.

How can an HTA file be exploited?

Despite their versatility and power, HTA files also introduce security risks, as they can execute code with elevated privileges on the local system. Attackers may exploit vulnerabilities in HTA files to execute malicious code and compromise the security of a user's system. Here's how an HTA exploit typically occurs:

- **Crafting Malicious HTA File**
  Attackers create a malicious HTA file containing code designed to exploit vulnerabilities in the Windows operating system or associated software components. This code may include techniques such as code injection, cross-site scripting (XSS), or leveraging vulnerabilities in ActiveX controls.
  ActiveX:
  ActiveX is a framework developed by Microsoft that allows software components to be reused in various applications or web pages. It enables developers to create interactive and rich content by incorporating pre-built controls, such as buttons, menus, and multimedia players, into their applications or web pages.
  ActiveX Controls:
  ActiveX controls are small, reusable software components that can be embedded within applications or web pages to add functionality. These controls can interact with the underlying system and perform tasks such as displaying content, handling user input, or accessing system resources.

- **Delivery**
  The malicious HTA file is then distributed to potential victims through various means, including email attachments, compromised websites, or disguised as legitimate software updates. Attackers may employ social engineering tactics to trick users into opening the malicious file.

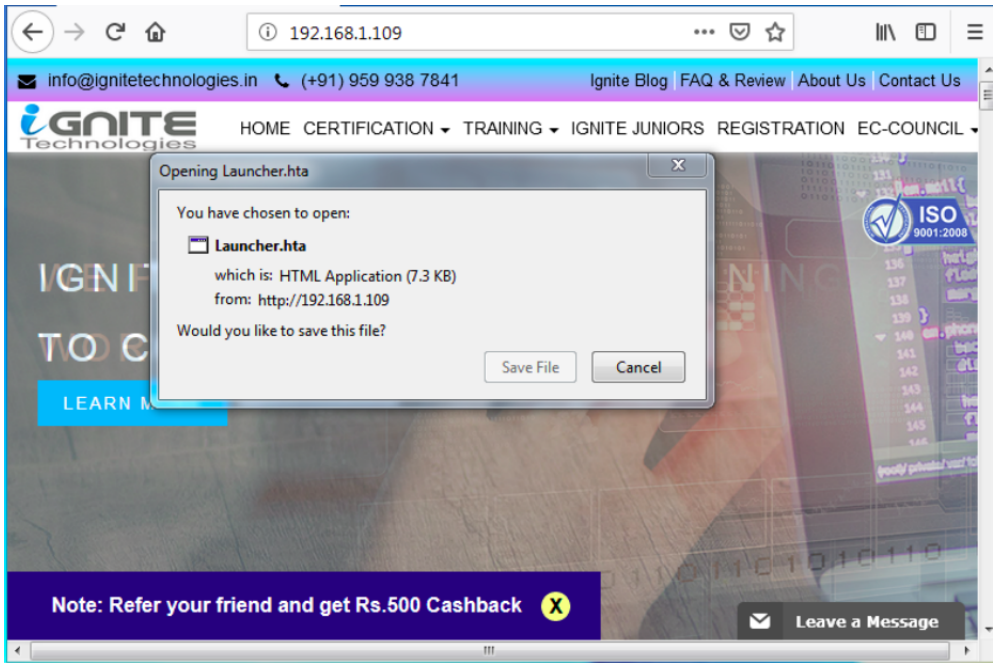

Figure 2.23: An image showing that when the victim browses the above malicious link, the file will be saved and automatically executed.

- **Execution**
  When a user opens the malicious HTA file, it is executed by the Microsoft HTML Application Host (`mshta.exe`). This grants the HTA file the ability to execute code with elevated privileges, enabling it to perform malicious actions on the victim's system.

- **Exploiting Vulnerabilities**
  The specific vulnerabilities exploited by HTA files can vary widely, depending on the target system's configuration and the techniques employed by the attacker. Common vulnerabilities include weaknesses in the HTML Application framework, the Windows operating system, or third-party software components accessed by the HTA file.

- **Payload**
  Once executed, the malicious HTA file may perform a variety of malicious actions, including installing malware, stealing sensitive information, or gaining unauthorized access to the victim's system. The payload of the exploit is limited only by the capabilities of the scripting language and the privileges granted to the HTA file.

Most infamous HTA attack: CVE-2017-0199

One notable real-life example of an HTA exploit is the "CVE-2017-0199" vulnerability, also known as the Microsoft Office/WordPad Remote Code Execution Vulnerability. This exploit involved crafting a malicious HTA file and embedding it within a Microsoft Office document, such as a Word or Excel file. When the victim opened the document, the embedded HTA file would execute, allowing the attacker to execute arbitrary code on the victim's system.

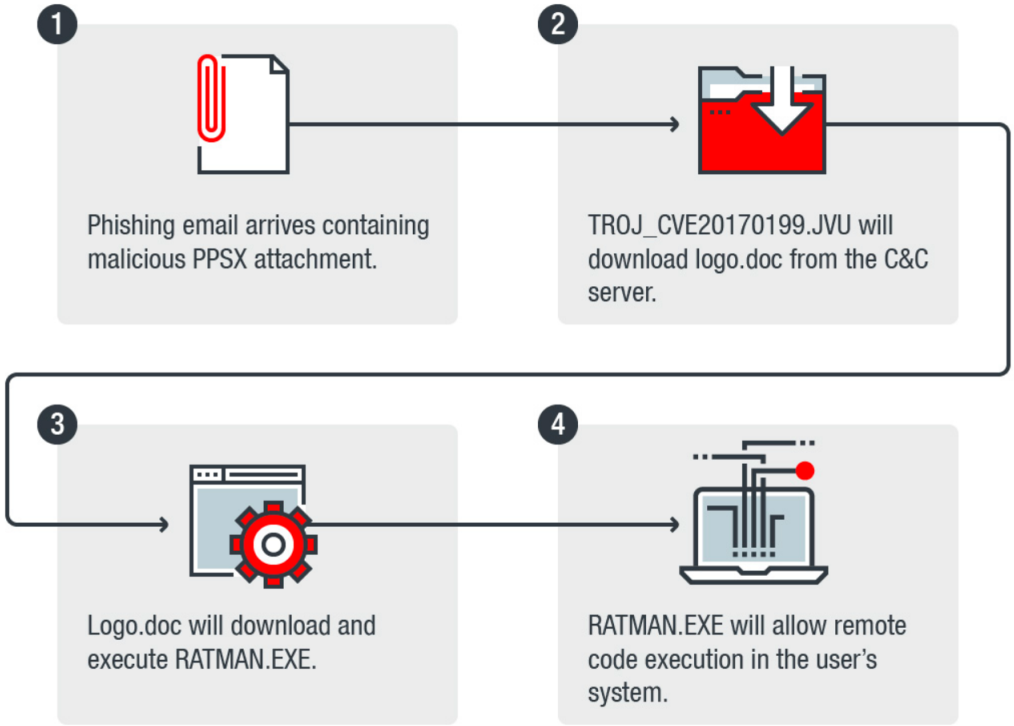

Figure 2.24: What exactly happened with the CVE-2017-0199

CVE-2017-0199 exploited a vulnerability in the way Microsoft Office and WordPad parsed specially crafted OLE (Object Linking and Embedding) objects within documents. By embedding a malicious HTA file within an OLE object, attackers could bypass security measures and execute code with elevated privileges, potentially leading to system compromise and data theft.

2.4.4 CHM Exploits

What is a CHM file?

A Compiled HTML Help (CHM) file is a type of file format commonly used for storing documentation, help files and other resources in a compressed HTML format. CHM files are often distributed with software applications and provide users with access to documentation and support resources directly from their local system.

Unlike regular HTML files, CHM files are packaged as a single compressed file with the extension `.chm`. They can contain a table of contents, index, and various HTML pages, all of which can include active content such as scripts and multimedia elements.

How can a CHM file be exploited?

Similar to other file formats, CHM files can be exploited by attackers to execute malicious code and compromise the security of a user's system. Here's how a CHM exploit typically occurs:

- **CHM File Creation**: Attackers create a malicious CHM file containing code designed to exploit vulnerabilities in the way Windows handles CHM files or associated software components. This code may include techniques such as buffer overflow exploits, code injection, or leveraging vulnerabilities in ActiveX controls.
- **CHM File Dissemination**: The attack begins with the distribution of a malicious CHM file, likely through spear-phishing emails. The CHM file is sent to potential victims, enticing them to open it.
- **Embedded HTML and Script**: The CHM file contains an embedded HTML document along with a script and decoy content. The embedded script is designed to execute when the CHM file is opened.
- **Execution via hh.exe**: Upon opening the CHM file, the embedded script executes via the HTML Help executable (hh.exe). This allows the attacker to initiate the malicious actions encoded in the script.
- **Legitimate Executable Vulnerability**: The script may exploit vulnerabilities in a legitimate executable that is vulnerable to DLL(Dynamic Link Library) search order hijacking. This vulnerability allows the attacker to load and execute a malicious loader DLL.

- **Malicious Loader DLL**: The malicious loader DLL is created and executed, enabling the attacker to gain control over the victim's system.

  What is DLL?
  DLL (Dynamic Link Library) is a file format used to hold multiple codes and procedures for Windows programs. DLL files allow several programs to share the same functions, thus conserving memory space. These files contain code and data that can be used by multiple programs simultaneously, without requiring the code to be replicated in each program's memory space.

  In the context of the CHM exploit, it refers to the exploitation of a vulnerability in a legitimate executable program that allows an attacker to load and execute their code (the malicious loader DLL) into the memory space of that program.

- **Actor-Controlled C2 (Command and Control) Infrastructure**: The attacker establishes communication with a command and control server controlled by the threat actor. This server is hosted on dynamically assigned domains (DDNS Domains) to evade detection.

  What is DDNS?
  DDNS(Dynamic Domain Name System) is a method of automatically updating a domain name's DNS (Domain Name System) records in real time to reflect changes in the underlying IP address of the resource it points to. By using DDNS services, attackers can update the IP address associated with their C2 domain in real time, making it more difficult for security teams to track and block malicious traffic.

- **ReVBShell as VBE File**: The attacker uses the ReVBShell tool disguised as a Visual Basic Script (VBE) file to establish contact with the actor-controlled command and control infrastructure.

- **Additional Tooling**: The attacker downloads additional tooling, such as Bisonal, to further exploit the compromised system or expand the scope of the attack.

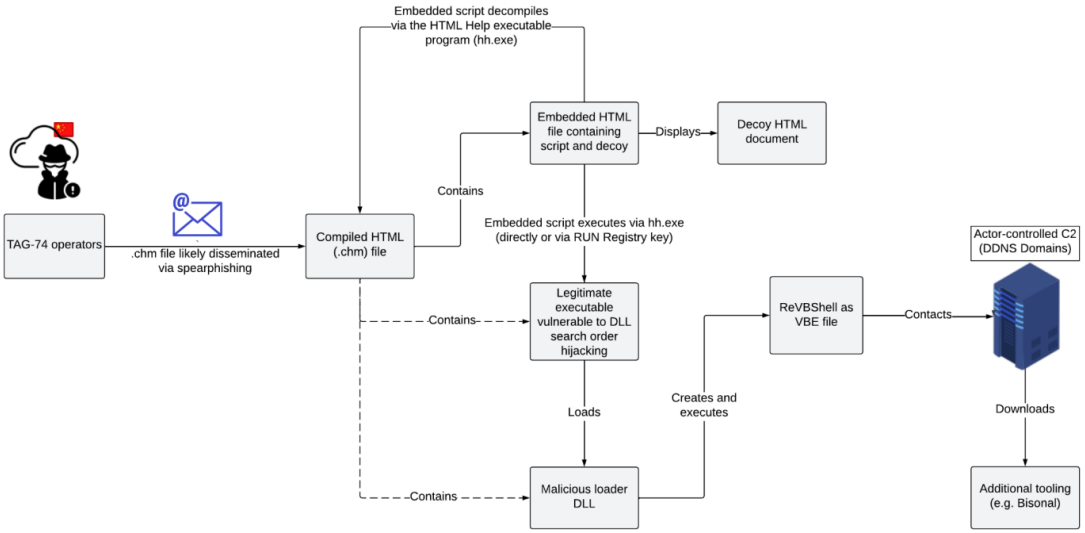

Figure 2.25: A diagram illustrating the process of a CHM (Compiled HTML Help) exploit

Example of a CHM exploit: CVE-2017-8570

One notable real-life example of a CHM exploit is the "CVE-2017-8570" vulnerability, also known as the Microsoft Office/WordPad Remote Code Execution Vulnerability. These exploits involved crafting a malicious CHM file and embedding it within a Microsoft Office document, such as a Word or Excel file. When the victim opened the document, the embedded CHM file would execute, allowing the attacker to execute arbitrary code on the victim's system.

CVE-2017-8570 exploited a vulnerability in the way Microsoft Office and WordPad parsed specially crafted OLE (Object Linking and Embedding) objects within documents. By embedding a malicious CHM file within an OLE object, attackers could bypass security measures and execute code with elevated privileges, potentially leading to system compromise and data theft.

2.4.5 JS-WEB Exploits

What is a JS-WEB?

JS-WEB refers to security vulnerabilities and issues associated with the execution of JavaScript code within a web environment, such as a user's web browser. JavaScript, a widely used programming language for creating dynamic and interactive web pages, can

become a target for malicious activities when its code is not adequately secured.

How it can be exploited?

Cross-Site Scripting (XSS)

Cross-site scripting (XSS) is a common security vulnerability in web applications that allows an attacker to inject and execute malicious JavaScript code on users' web browsers. This vulnerability occurs when user-provided data is not properly escaped, filtered, or validated by the web application before being dynamically displayed on a web page.

How it work?

The attacker exploits a flaw in the web application to insert malicious JavaScript code into input fields, comments, URLs, or other insecure entry points of the application. When this infected data is sent back to end-users and displayed on a web page, the malicious JavaScript code is then executed on the client-side, within the user's browser. It can perform a variety of harmful actions, ranging from stealing user sessions to stealing authentication cookies, redirecting to malicious websites, displaying fake interfaces to steal sensitive information, or even taking full control of the user's browser.

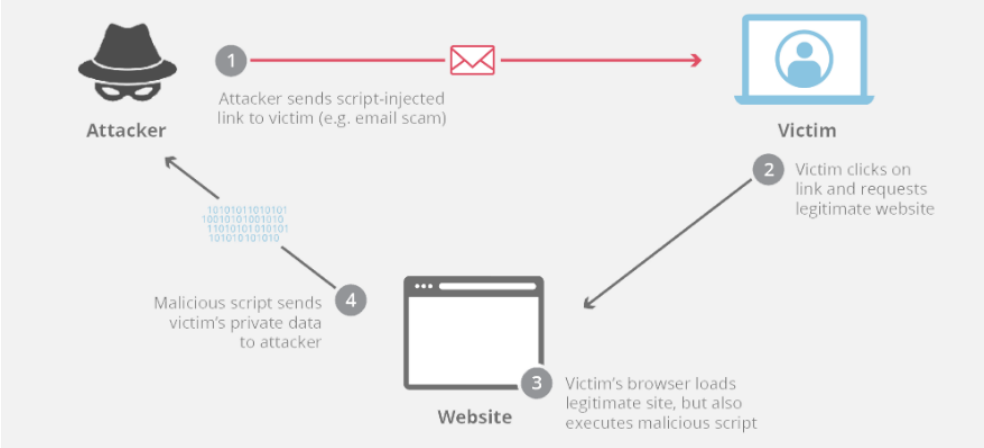

Figure 2.26: Typical steps in an XSS exploit.

Clickjacking

Clickjacking is a malicious technique used by attackers to deceive users into unknowingly clicking on elements of a web page. This attack exploits a vulnerability in how web browsers handle transparency layers in web pages.

How it work?

The attacker creates a malicious web page containing a transparent or invisible element, often in the form of a button, link, or input field. Then he overlays this transparent or invisible malicious page with a legitimate web page they want to target. This legitimate page could be anything, such as a login page, a payment button, etc. Users, upon visiting the page with both legitimate and malicious elements, interact with what appears to be legitimate content, unaware of the hidden malicious element. The attacker may position the login or payment button in the transparent or invisible layer. Upon clicking what seems to be the legitimate login or payment button, users inadvertently interact with the hidden malicious element. This enables the attacker to initiate undesirable actions, such as stealing credentials, processing payments, liking or sharing content on social media, and more.

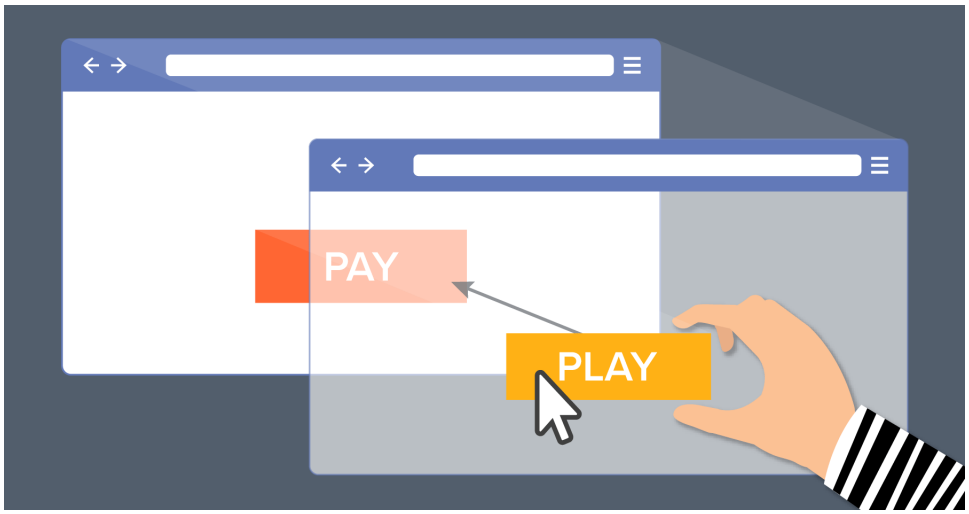

Figure 2.27: Clickjacking illustration

The Document Object Model (DOM)

The Document Object Model (DOM) is a representation of a web page's structure created by the browser when a web page is opened. It serves as a sort of blueprint or map of the page, with all its elements like headings, paragraphs, images, links, etc., represented as objects in a tree-like structure.

How it can be exploited?

the DOM, while essential for web page functionality, can be a target for exploitation. Attackers exploit vulnerabilities within the web page's DOM representation to carry out malicious activities. By identifying weaknesses in the construction of the page, they can inject malicious JavaScript code into accessible elements. This injected code can then execute within the user's browser, leading to a range of harmful actions.

For example, attackers may alter the functionality of buttons or links, redirect users to fraudulent websites, steal sensitive data such as login credentials or personal information, or create deceptive visual changes to deceive users.

Essentially, the manipulation of the DOM allows attackers to clandestinely modify the behavior and content of the web page, often without the user's awareness.

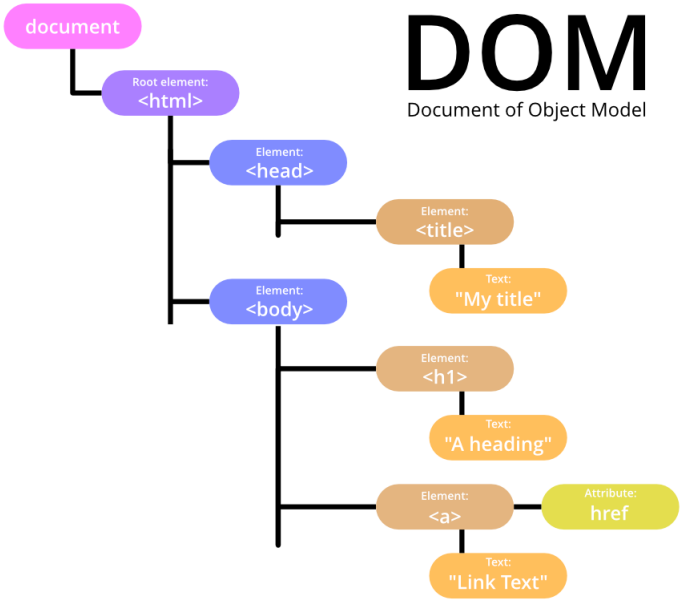

Figure 2.28: Structure of The Document Object Mode

2.4.6 WinRAR Exploits

What is Winrar?

WinRAR is a popular file compression software for Windows. Compressed files are typically used to reduce their size and make it easier to transfer or store them. However, there have been exploits or vulnerabilities discovered in WinRAR, allowing attackers to manipulate compressed files in a way that enables them to execute malicious code on a targeted system.

Figure 2.29: Winrar illustration

WinRAR exploit?

The CVE-2018-20250

The WinRAR exploit targeted a decompression vulnerability called CVE-2018-20250 within the WinRAR software. This security flaw allowed a remote attacker to execute malicious code on a vulnerable system. This vulnerability affected WinRAR versions before the release of version 5.70 beta 1.

The WinRAR exploit functioned by utilizing a specially crafted compressed file, often in the ACE (Archiver Compression for Executables) format.

The Archiver Compression for Executables format(ACE)

ACE files are compressed files using a specific format called ACE. This compression format is an older format created by Marcel Lemke in 1993 for his archiving software WinACE. ACE files were popular for a time due to their ability to compress executable files and other file types.

Exploitation Mechanism

Malicious ACE files were used to exploit the decompression vulnerability (CVE-2018-20250). The ACE files were prepared in such a way that when decompressed by WinRAR, the extracted files could be placed in sensitive locations of the system such as system directories or startup folders, thus allowing the execution of malicious code.

Attackers could use this method to deploy and execute various forms of malware on the targeted system. This included the installation of backdoors, keyloggers, ransomware, or other malicious software. Additionally, sensitive data could be compromised, and complete control of the compromised system could be obtained.

To address this vulnerability, WinRAR users were advised to update their software to version 5.70 beta 1 or later. Security updates are crucial to ensure that systems are protected against known exploits and potential attacks.

3 Malware Specifically Designed to Target Windows Computers

In our initial chapter, we provided a brief overview of malware concepts. Now, we're set to venture deeper into the realm of malicious software, particularly focusing on the prominent threats targeting Windows operating systems.

3.1 Exploring the Anatomy of Malicious Code

Before we dive into the specific types of attacks targeting Windows computers, it's essential to understand the various components of malicious code. These individual elements, though they may appear innocuous, can combine to create a potent and complex threat when manipulated by cybercriminals.

3.1.1 Payload

A payload refers to the part of malicious code that performs a specific action once the malware is executed on a target system. This action could range from stealing data, deleting files, installing backdoors, or any number of harmful activities. Payloads can take

different forms. For instance, they can be categorized into stage payloads and stageless payloads.

3.1.1.1 Stage Payload

A stage payload is a method of deploying malicious code that is divided into multiple parts, or "stages". Each stage has a specific role in the overall attack.

First Stage

The first stage, also known as a "stager", is often a small piece of code initially executed on the target system. Its primary goal is to prepare the ground for the rest of the payload by establishing a connection, bypassing security defenses, or gaining necessary privileges. This first stage can be relatively simple and compact to remain unnoticed and bypass security checks.

Second Stage

Once the first stage has fulfilled its function, it downloads and executes the second stage of the payload. The second stage, often larger and more complex, is the actual payload of the attack. It may include more advanced features such as installing backdoors, stealing sensitive information, remote system control, etc.

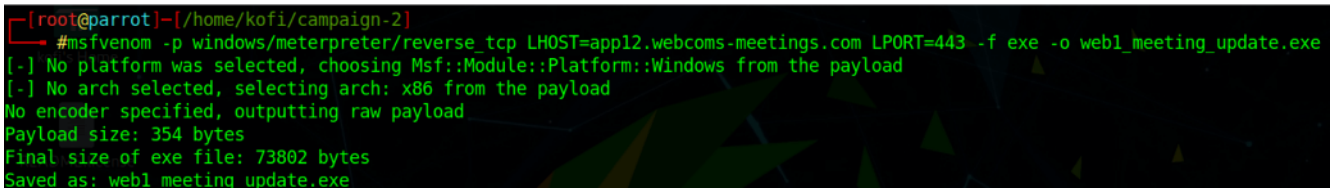

Figure 2.30: An example of a staged payload as a Windows executable

Advantages

Payload stages are frequently employed to circumvent security defenses by distributing the malicious components of an attack. They can operate more covertly since the initial

stage may be crafted to avoid detection by antivirus software or threat detection tools.

Disadvantages

The staged process can present a more intricate challenge for attackers, demanding meticulous planning and coordination. Additionally, it may result in a slower attack progression, as each stage must be executed before the final payload becomes active.

3.1.1.2 Stageless Payload

A stageless payload refers to an attack where all the malicious code is executed simultaneously, without the requirement for multiple distinct stages. Here are some key characteristics of these payloads:

Direct Execution

In a stageless payload, all the necessary code for the attack is contained within a single piece. When this payload is executed on the target system, it promptly initiates all the planned actions without the need to download additional components.

Simplicity

Stageless payloads can be simpler to create and implement since they do not necessitate coordination among multiple code segments. Additionally, they can operate more swiftly, as the entire attack process is carried out in a single execution.

Detection

However, stageless payloads are more susceptible to detection by antivirus software and threat detection tools, as all the malicious code is present in one instance.

Uses

Stageless payloads are frequently employed when speed and deployment simplicity are crucial, or when security defenses are not as robust.

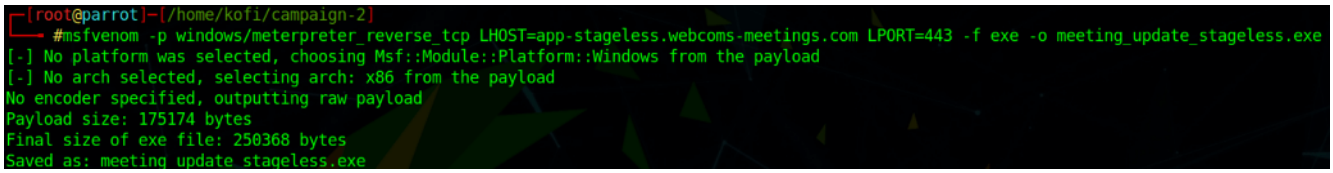

Figure 2.31: A stageless meterpreter reverse TCP payload as a Windows executable

In summary, the main difference between a staged payload and a stageless payload lies in how the malicious code is structured and executed. Stage payloads are divided into multiple parts, while stageless payloads are a single entity, affecting their complexity, execution speed, and ability to bypass security defenses.

3.1.1.3 Staged vs. stageless payloads

Metasploit staged payloads have the forward-slash symbol after the word Meterpreter. The screenshot below shows examples of Windows staged Meterpreter payloads.

Figure 2.32: examples of Windows staged Meterpreter payloads

Stageless payloads employ the underscore symbol after the word Meterpreter. The screenshot below shows examples of Windows Meterpreter stageless payloads.

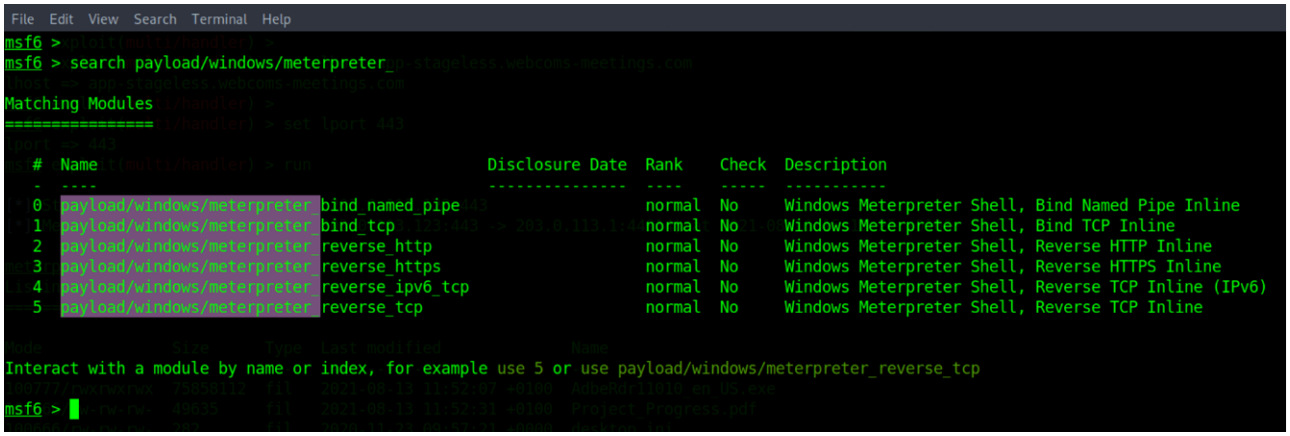

Figure 2.33: examples of Windows Meterpreter stageless payloads

The example below shows both categories of payloads.

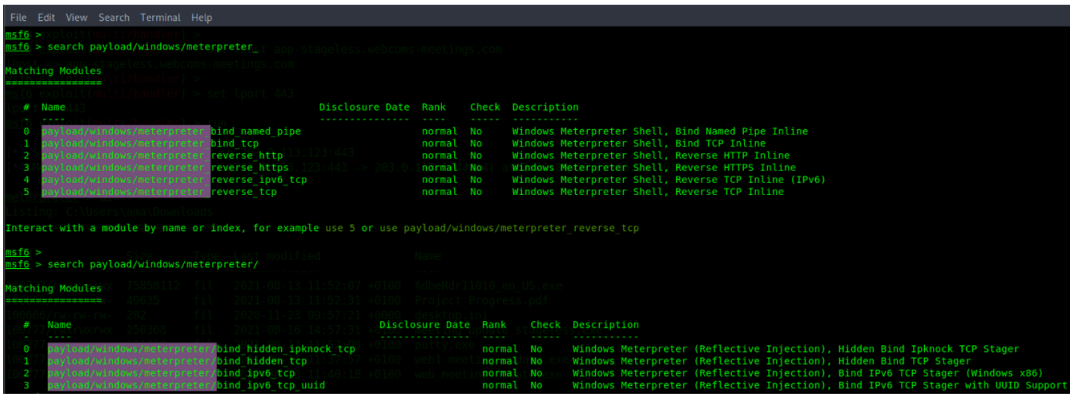

Figure 2.34: Payloads categories

Stageless payloads are self-contained and do not require the extra step of sending a stage to the victim machine once the malware makes a callback to the attacker's infrastructure. Notice in the screenshot below that after the reverse TCP handler starts, the next step is to open a hands-on Meterpreter remote shell session to the victim machine straight away without the need to send any further payloads such as a stage.

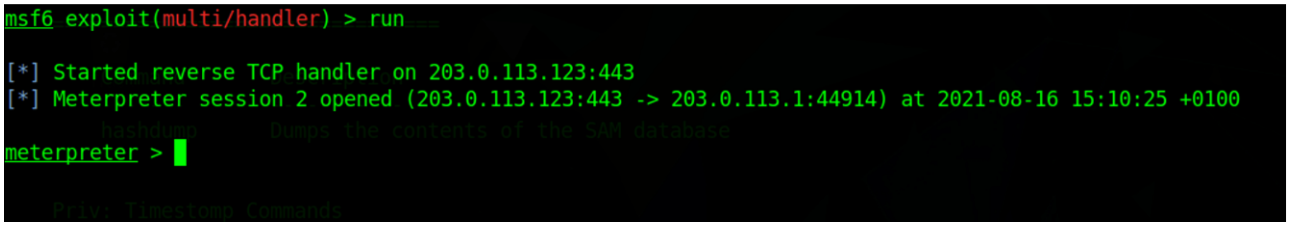

Figure 2.35: Reverse TCP handler

From the screenshot below, we can see that there is the extra step of sending the stage after the stager makes a callback to the attacker infrastructure: "sending stage (175174) to 203.0.113.1."

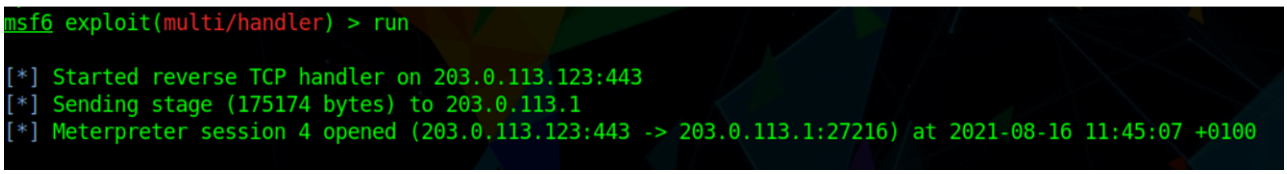

Figure 2.36: A callback to the attacker infrastructure:

Another difference between staged and stageless is the size of the payloads. In the screenshot below, the stageless payload (meeting update stageless.exe) is much larger at 245KB compared to the staged initial payload (web1 meeting update.exe) at 73KB.

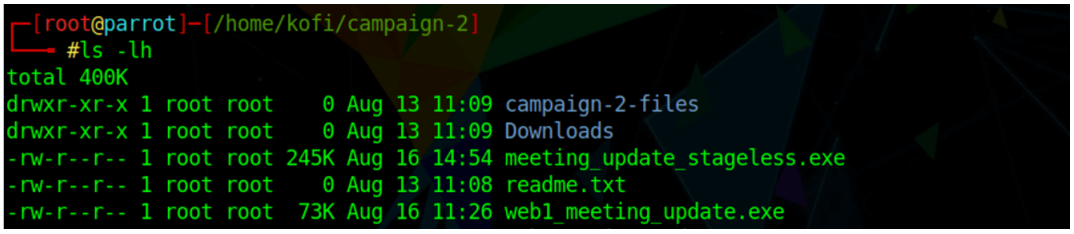

Figure 2.37: Size difference between different types of payloads [16]

3.1.2 shell code

A shellcode is a small piece of code typically written in assembly language that is injected into the memory of a running program. It is often a crucial component of a payload, serving as the direct executable portion of the malicious code. Its main purpose is to provide the attacker with direct access to the target system by replacing the normal flow of the program with malicious code. The shellcode can be used to open a command-line interface (shell) on the target system, allowing the attacker to execute commands remotely. The example below shows the Powershell shell code (ps1).

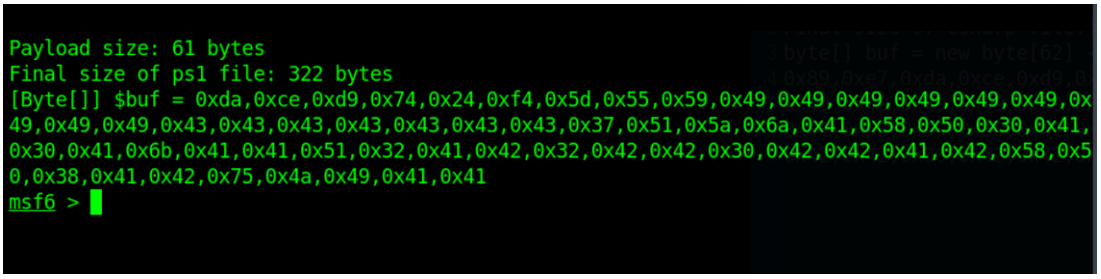

Figure 2.38: Powershell shellcode

This particular example utilizes an in-memory injected Windows Dynamic Link Library (DLL) via a reflective loader. The shellcode is generated in alphanumeric form. Upon successful execution, it can reconnect back to the attacker via a reverse DNS TCP session generated from the Metasploit Framework.

Metasploit framework

Metasploit is an open-source penetration testing framework. It provides tools, resources, and infrastructure to test the security of computer systems. It includes a wide variety of modules, exploits, payloads, and tools to conduct penetration tests on different types of systems and software. It is used to search for vulnerabilities, develop exploits, execute attacks, test network security, and more. Additionally, there are other frameworks available such as Cobalt Strike, Covenant, PowerShell Empire, and Armitage.

Meterpreter

Meterpreter is an advanced and powerful command interpreter included in the Metasploit framework. It is one of the types of payloads available in Metasploit. Meterpreter is designed to be implanted on compromised systems after a successful exploit. It offers a wide range of post-exploitation functionalities, including the ability to remotely control the target system, steal data, install additional malware, pivot through networks, erase activity traces, etc. Meterpreter is popular due to its versatility, robustness, and stealthiness.

3.1.3 Propagation Mechanism

Malware often includes mechanisms to spread itself to other systems. This could be through email attachments, network shares, USB drives, or exploiting vulnerabilities in software to spread across a network.

3.1.4 Evasion Techniques

Malicious code often employs various evasion techniques to avoid detection by antivirus programs and other security measures. This might involve encryption to hide its true purpose, polymorphic code that changes its appearance each time it runs, or rootkit functionality to hide its presence on the system.

3.1.5 Backdoor Access

Many malware strains create a backdoor into the infected system. This allows the attacker to access the compromised system remotely for further exploitation, data exfiltration, or to use the system as part of a botnet.

3.1.6 Persistence Mechanism

Malware often tries to maintain a presence on the system even after a reboot. This could involve adding entries to startup scripts, creating hidden files or registry entries, or installing itself as a service or driver.

3.1.7 Command and Control (C2) Communication

To receive instructions and exfiltrate data, the malicious code often communicates with a remote server controlled by the attacker. The c2 is essentially the communication channel used by attackers to remotely control malware on compromised systems. This communication is usually encrypted to avoid detection.

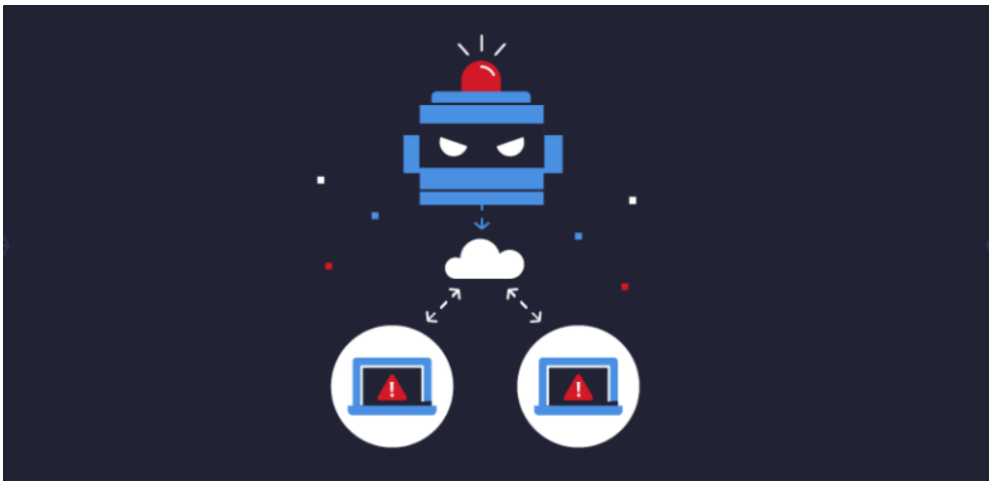

Figure 2.39: Command and control illustration
[17]

3.1.8 Trigger

The trigger is the event or condition that causes the malicious code to execute its payload. This could be a specific date, time, system event, user action, or any other predefined condition set by the attacker.

3.1.9 Obfuscation Techniques

Malware authors often obfuscate their code to make it difficult for analysts and security tools to understand its behavior. This could involve code obfuscation, packing, or using

anti-analysis techniques. Let's take this Shell Code injector as an example :

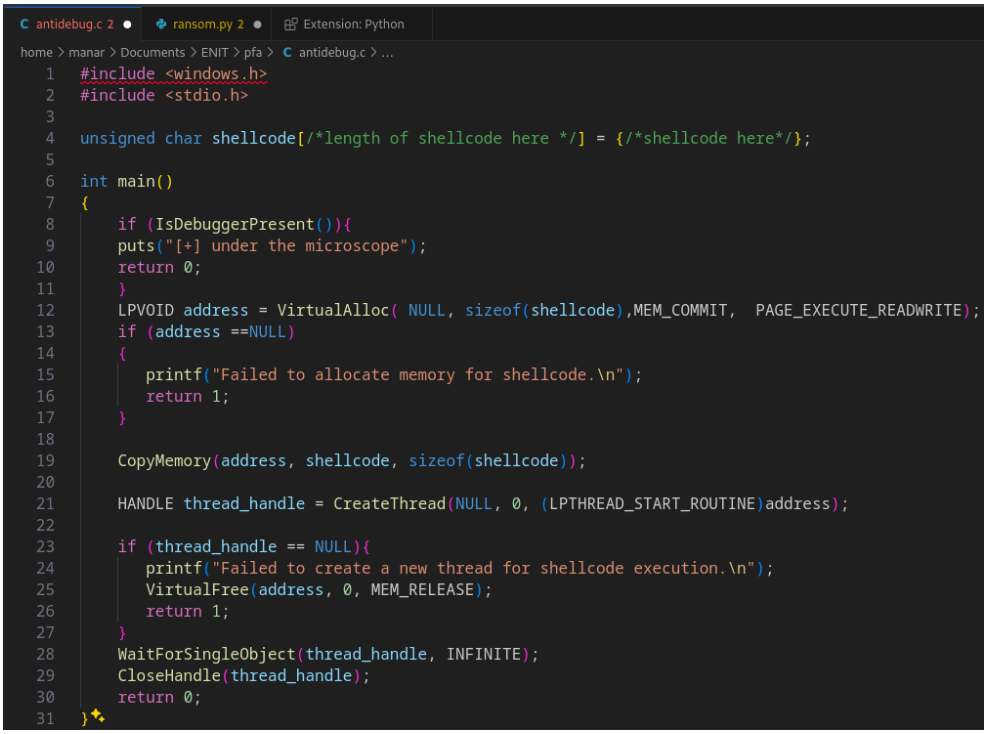

Figure 2.40: Shell code injector using an anti-debugging technique

This shell code injector program checks if it's being debugged. Otherwise, it allocates memory for shellcode execution, copies the shellcode into the allocated memory, creates a new thread to execute it, waits for the thread to finish execution, and then releases the allocated memory.

3.1.10 Self-Replication

Some types of malware, such as worms, are designed to self-replicate and spread autonomously to other systems without requiring user intervention.

3.1.11 Exfiltration Mechanism

If the goal of the malware is to steal data, it will include mechanisms to gather sensitive information from the infected system and send it back to the attacker's server.

3.2 Ransomware

3.2.1 Understanding the Mechanics of Ransomware

Ransomware operates by infiltrating a target system, encrypting its files, and then demanding a ransom from the victim. While the specific methods may vary between different ransomware variants, they all generally follow the same fundamental stages. [10]

- **Phase 1: Initial Access and Propagation**
  Ransomware employs various means to infiltrate an organization's systems, with certain infection vectors being more commonly favored by ransomware operators.

  One prevalent method is through phishing emails, where malicious links or attachments lead to the download and execution of ransomware on the victim's computer.

  As shown in the next few images, a threat actor sends a phishing email to the victim, in which he attaches an embedded Microsoft Word file. The moment it gets open a malicious code is executed and a reverse shell is being created and redirected to the attacker's machine. This attack was executed an a Windows 10 pro computer, and none of the security firewalls and applications were alerted. Once the attacker has full control over the victim's machine, he can download any kind of malware. He might as well insert the ransomware directly in the phishing email. Another tactic involves exploiting services like Remote Desktop Protocol (RDP), allowing attackers to remotely access a computer within the network using stolen or guessed login credentials. Additionally, ransomware may exploit vulnerabilities directly, as seen with the EternalBlue exploit used by WannaCry. Most ransomware variants utilize multiple infection vectors to maximize their reach.

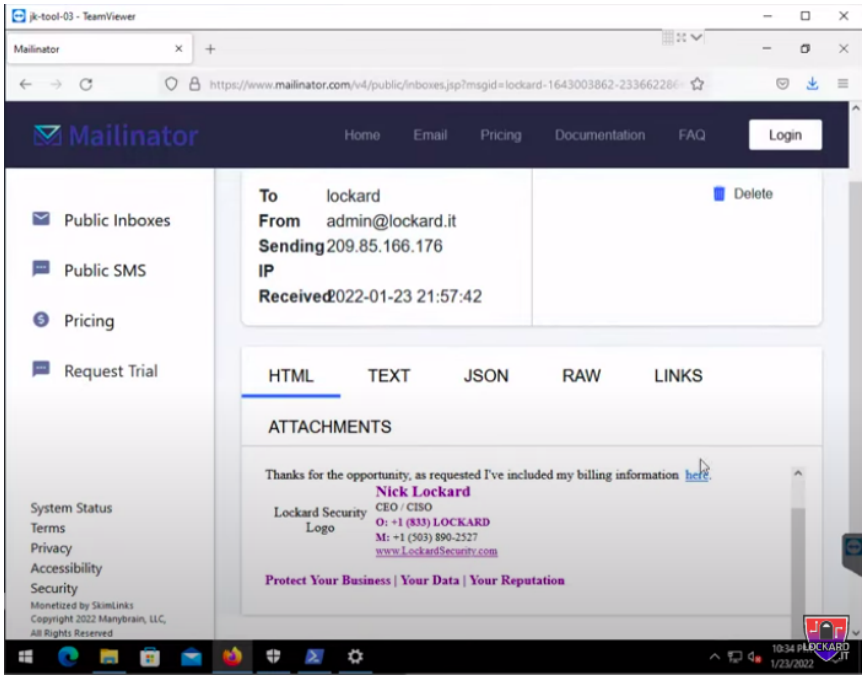

Figure 2.41: Phishing email

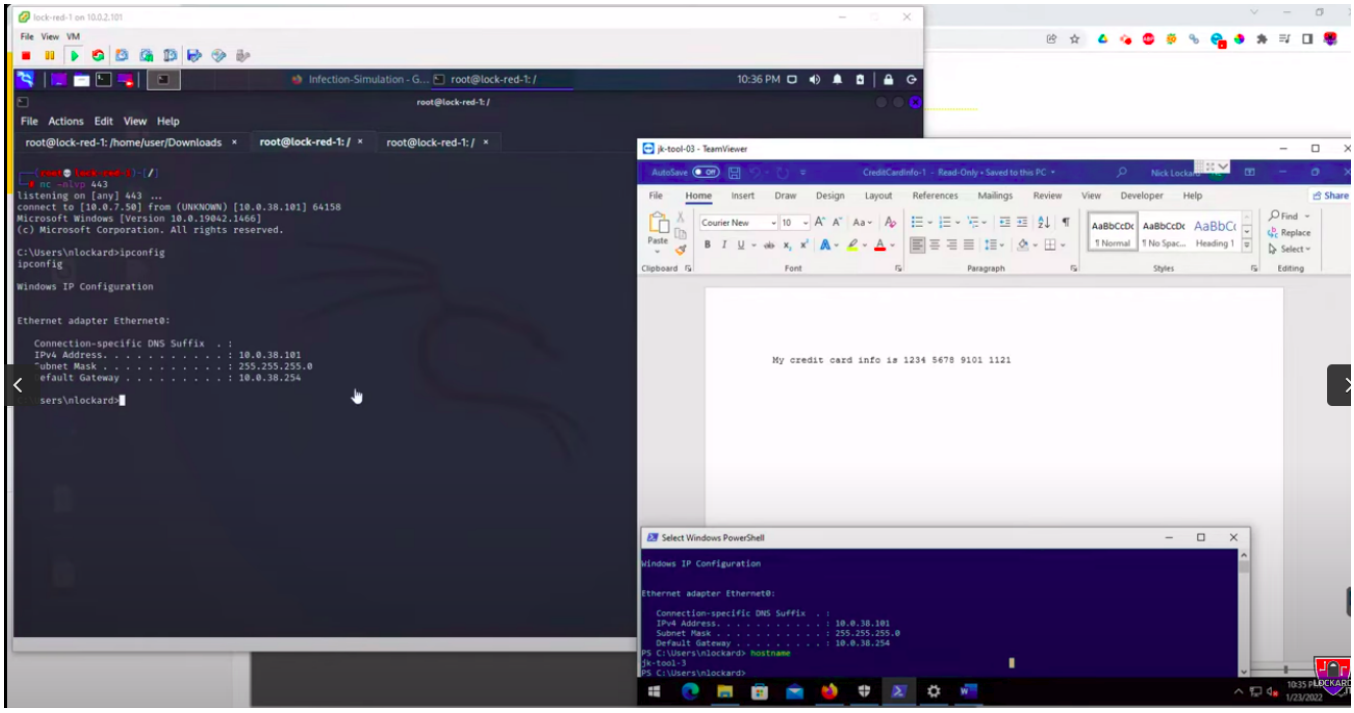

Figure 2.42: The backdoor has been created just after the file was opened

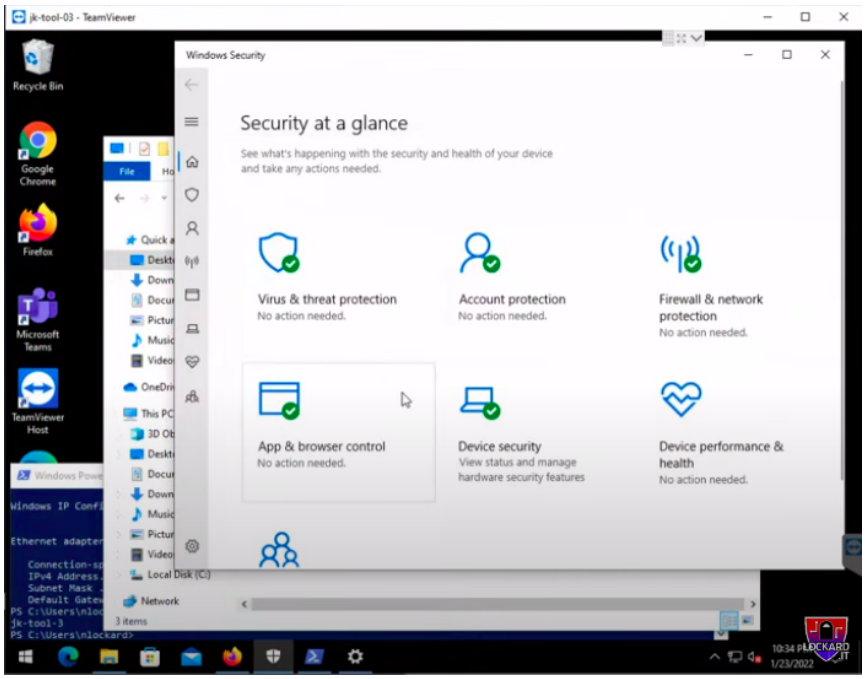

Figure 2.43: All security measures are in place, and none of them detected the malicious file

- **Phase 2: File Encryption**
  Once inside a system, ransomware initiates the encryption process on its targeted files. This typically involves accessing files, encrypting them using an attacker-controlled key, and substituting the originals with the encrypted versions. Ransomware variants exercise caution in selecting files to encrypt to avoid destabilizing the system. Some may even delete backup and shadow copies to hinder recovery without the decryption key.

  In the image that follows, we developed a simple ransomware using Python. This script encrypts all files in the current directory using the Fernet symmetric encryption algorithm and generates a unique key to decrypt them. The encrypted contents are stored in the same files, effectively rendering them unreadable without the decryption key. In real life, it's not that simple, the code must be obfuscated or encrypted to bypass all firewalls. Furthermore, the attacker might add some features to prevent security engineers from debugging their malware. The simplest way to do that is to use predefined functions such as IsDebuggerPresent() in C programming for example which aborts all the payload once it detects the presence of a debugger. But the use of such a function can easily be detected and that's why threat actors have created various techniques to replace that function.

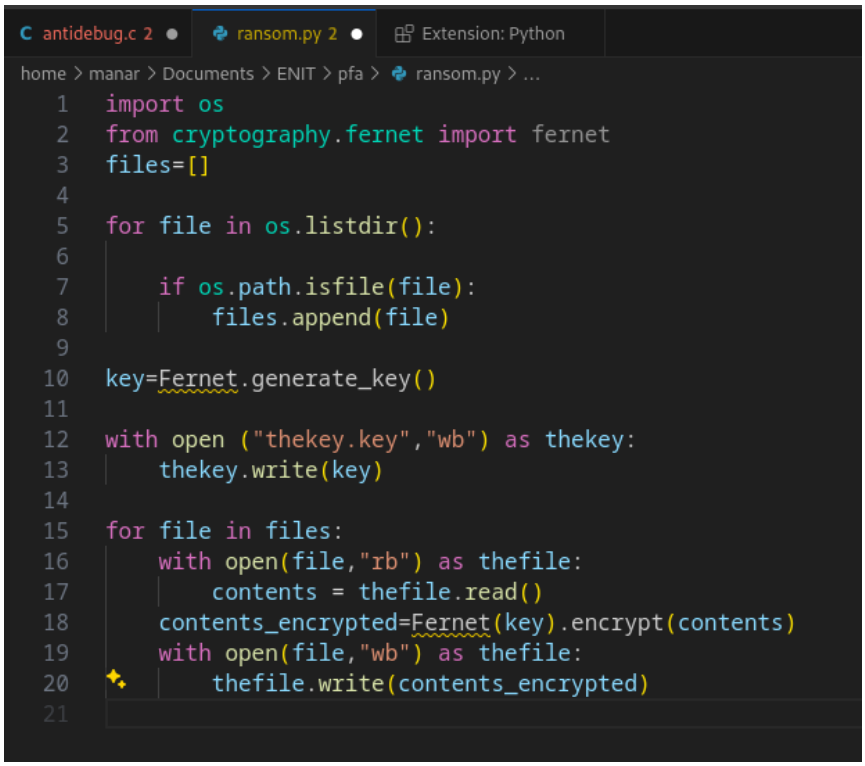

Figure 2.44: An example of ransomware written using Python

- **Phase 3: Ransom Demand**
  Upon completing the encryption process, ransomware presents the victim with a ransom demand. This can manifest in various ways, such as altering the display background to show a ransom note or placing text files containing ransom instructions in encrypted directories. Typically, these demands involve a specified sum of cryptocurrency in exchange for file access. Upon payment, the ransomware operator provides either the decryption key or decryption instructions, enabling the victim to restore their files.

While these core stages are common to all ransomware variants, variations and additional steps may exist. For example, Maze ransomware incorporates data scanning and theft before encryption, while WannaCry seeks out vulnerable devices for infection.

3.2.2 Types of Ransomware Incidents

Ransomware has undergone significant evolution over the years, leading to the emergence of various types and related threats:

Figure 2.45: The ransom prompt of the infamous WannaCry ransomware attack

- **Double Extortion:** This variant combines file encryption with data theft, allowing cybercriminals to threaten data exposure alongside encryption to pressure victims into paying.
- **Triple Extortion:** Triple extortion adds a third element, often targeting the victim's customers or partners with ransom demands or conducting distributed denial-of-service (DDoS) attacks against the company.
- **Locker Ransomware:** Unlike traditional ransomware, locker ransomware locks the victim's computer, rendering it unusable until the ransom is paid.
- **Crypto Ransomware:** This term emphasizes the use of cryptocurrency for ransom payments due to its difficulty to trace.
- **Wiper:** While similar to ransomware, wipers aim to permanently deny access to files by deleting encryption keys.
- **Ransomware as a Service (RaaS):** Ransomware as a service (RaaS) is a cybercrime business model where ransomware operators write software and affiliates pay to launch attacks using said software. Affiliates do not need to have technical skills of their own but rely on the technical skills of the operators.
- **Data-Stealing Ransomware:** Some variants focus on data theft rather than encryption, exploiting stolen data for extortion purposes.

3.2.3 Prominent Ransomware Variants

Numerous ransomware variants exist, each with distinct characteristics. However, certain groups have gained notoriety for their prolific and impactful operations:

- **Ryuk**: Ryuk ransomware is characterized by its targeted approach, often infiltrating the networks of large enterprises. It employs sophisticated encryption algorithms to encrypt critical files and demands substantial ransom payments, typically exceeding $1 million, in exchange for decryption keys. Ryuk operators meticulously select their targets, aiming for organizations with significant financial resources to maximize ransom potential.

- **Maze**: Maze ransomware is notable for pioneering the technique of "double extortion." In addition to encrypting files, Maze operators exfiltrate sensitive data from compromised systems. They then leverage the threat of public data exposure to coerce victims into paying ransom. This dual-threat strategy adds a layer of complexity to the attack and increases the pressure on victims to comply with ransom demands.

- **REvil** (Sodinokibi): REvil ransomware, also known as Sodinokibi, is a prominent threat to large organizations. It employs sophisticated encryption algorithms to encrypt files on compromised systems rapidly. However, what sets REvil apart is its double extortion technique, where it not only demands ransom for file decryption but also threatens to leak sensitive data if ransom demands are not met. This approach significantly increases the stakes for victims, compelling many to pay the ransom to prevent data exposure.

- **WannaCry**: WannaCry ransomware gained infamy for its widespread propagation in 2017, exploiting the EternalBlue exploit to target vulnerable Windows systems. WannaCry encrypts files on infected machines and demands ransom payments in cryptocurrency for decryption keys. Its rapid spread across networks, coupled with its use of a potent exploit, resulted in widespread disruption and financial losses for organizations globally.

- **LockBit**: LockBit ransomware is a relatively recent entrant in the Ransomware-as-a-Service (RaaS) landscape. It employs sophisticated encryption techniques to swiftly encrypt files on compromised systems. LockBit operators distribute the ransomware to affiliates through underground forums, allowing them to carry out attacks independently. This decentralized approach enhances the scalability and adaptability of LockBit operations, making it a significant threat to organizations worldwide.

- **DearCry**: DearCry ransomware exploits vulnerabilities in Microsoft Exchange servers to infiltrate networks and encrypt files on compromised systems. It typically targets organizations that use vulnerable Exchange servers, exploiting known security flaws to gain unauthorized access. DearCry operators demand ransom payments for decryption keys, leveraging the urgency of data recovery to compel victims to comply with their demands.

- **Lapsus$**: Lapsus$ is a South American ransomware group known for its targeted attacks on high-profile entities. It employs a variety of tactics, including phishing campaigns and software vulnerabilities, to gain initial access to target networks. Once inside, Lapsus$ ransomware encrypts files and demands ransom payments in cryptocurrency. Additionally, it may threaten to release sensitive information to further pressure victims into paying the ransom.

3.3 Spyware

Spyware is malicious software designed to infiltrate a user's device covertly and monitor their activities without their knowledge or consent. Unlike other types of malware that aim to disrupt or damage systems, spyware operates stealthily, often remaining undetected while gathering sensitive information.

3.3.1 Functionality of Spyware

Spyware performs various functions, typically focused on espionage and data theft. Some common functionalities of spyware include:

- **Keylogging**: Spyware may capture keystrokes entered by the user, allowing attackers to record sensitive information such as login credentials, credit card numbers, and personal messages.

- **Screen Capture**: Certain spyware variants can take screenshots of the user's screen at regular intervals or in response to specific triggers. This enables attackers to monitor the user's activities and gather visual information.

- **Data Theft**: Spyware may exfiltrate sensitive data stored on the infected device, including documents, photos, and browsing history. Attackers can use this stolen information for identity theft, financial fraud, or blackmail.

- **Remote Access**: Advanced spyware tools may provide attackers with remote access to the compromised device, allowing them to execute commands, install additional malware, or manipulate files and settings.
- **Tracking and Monitoring**: Spyware can track the user's online activities, including websites visited, search queries, and online purchases. This information may be used for targeted advertising, profiling, or surveillance purposes.

3.3.2 Infection Vectors

Spyware can infect devices through various vectors, including:

- **Malicious Websites**: Users may inadvertently download spyware by visiting compromised or malicious websites that host exploit kits capable of exploiting software vulnerabilities.
- **Email Attachments**: Spyware may be distributed via email attachments, disguised as legitimate files or documents. Unsuspecting users who open these attachments inadvertently install the spyware on their devices.
- **Software Bundling**: Some legitimate software applications may include spyware components bundled with their installation packages. Users who install such software unknowingly also install the accompanying spyware.
- **Drive-by Downloads**: Spyware can be delivered through drive-by downloads, where malicious code is automatically downloaded and executed when users visit compromised or malicious websites.
- **Removable Media**: Spyware can spread through infected removable media, such as USB drives or external hard disks when connected to compromised devices.

3.3.3 Impact of Spyware

The presence of spyware on a device can result in severe consequences for both individuals and organizations. Here are some notable impacts of spyware:

- **Privacy Breaches**: Spyware compromises user privacy by monitoring their activities, gathering sensitive information, and transmitting it to unauthorized third parties without consent.

- **Financial Losses**: Spyware can lead to financial losses through identity theft, unauthorized access to banking or payment credentials, fraudulent transactions, and extortion attempts.
- **Reputation Damage**: Organizations affected by spyware incidents may suffer reputational harm due to breaches of customer confidentiality, loss of intellectual property, or disclosure of sensitive information.
- **Legal and Regulatory Consequences**: Organizations targeted by spyware attacks may face legal and regulatory repercussions, such as fines, penalties, and lawsuits for inadequate protection of sensitive data.
- **System Instability**: Certain spyware variants consume substantial system resources, resulting in degraded performance, system crashes, or software malfunctions. This can disrupt normal operations and productivity.

3.4 Trojans

3.4.1 What is trojan malware?

Trojans, a form of malicious software, disguise themselves as legitimate files or programs, tricking users into downloading and executing them. Unlike viruses, Trojans cannot spread on their own and require users to unwittingly install their server-side components. Once installed, Trojans possess a variety of harmful capabilities, from stealing sensitive data to creating hidden access points for future attacks.
Trojan viruses typically spread through seemingly genuine emails and their attachments. These emails are designed to reach as many inboxes as possible, luring users into downloading the malicious attachments, which unknowingly initiates the installation of the Trojan server. This server then operates covertly each time the infected device starts up.
Furthermore, Trojans frequently exploit social engineering tactics, a common tool in the cybercriminal's arsenal. Users may be coerced into downloading malicious applications hidden within banner ads, pop-ups, or links on websites.

3.4.2 Types of Trojan malware

Trojans come in various forms, each with its own insidious purpose:

- **Backdoor Trojans**: A backdoor Trojan enables an attacker to gain remote access to a computer and take control of it using a backdoor. This enables the malicious

actor to do whatever they want on the device, such as deleting files, rebooting the computer, stealing data, or uploading malware. A backdoor Trojan is frequently used to create a botnet through a network of zombie computers.

- **Banker Trojans**: A banker Trojan is designed to target users' banking accounts and financial information. It attempts to steal account data for credit and debit cards, e-payment systems, and online banking systems.
- **DDoS Trojans**: These Trojan programs carry out attacks that overload a network with traffic. It will send multiple requests from a computer or a group of computers to overwhelm a target web address and cause a denial of service.
- **Dropper or downloader Trojans**: A downloader Trojan targets a computer that has already been infected by malware, then downloads and installs more malicious programs to it. This could be additional Trojans or other types of malware like adware.
- **Fake antivirus Trojans**: A fake antivirus Trojan simulates the actions of legitimate antivirus software. The Trojan is designed to detect and remove threats like a regular antivirus program, then extort money from users for removing threats that may be nonexistent.
- **Gaming Trojans**: A game-thief Trojan is specifically designed to steal user account information from people playing online games.
- **Instant message Trojans**: This type of Trojan targets IM services to steal users' logins and passwords. It targets popular messaging platforms such as AOL Instant Messenger, ICQ, MSN Messenger, Skype, and Yahoo Pager.
- **Infostealer Trojans**: This malware can either be used to install Trojans or prevent the user from detecting the existence of a malicious program. The components of infostealer Trojans can make it difficult for antivirus systems to discover them in scans.
- **Mailfinder Trojans**: A mailfinder Trojan aims to harvest and steal email addresses that have been stored on a computer.

Examples of Trojan horse virus attacks

Trojan attacks have been responsible for causing major damage by infecting computers and stealing user data. Well-known examples of Trojans include:

- **Rakhni Trojan**: The Rakhni Trojan delivers ransomware or a cryptojacker tool—which enables an attacker to use a device to mine cryptocurrency—to infect devices.

- **Tiny Banker**: Tiny Banker enables hackers to steal users' financial details. It was discovered when it infected at least 20 U.S. banks.
- **Zeus or Zbot**: Zeus is a toolkit that targets financial services and enables hackers to build their own Trojan malware. The source code uses techniques like form grabbing and keystroke logging to steal user credentials and financial details[11][12].

3.5 Rootkits

3.5.1 What is rootkit malware?

A rootkit is a malicious software package designed to allow an intruder unauthorized access to a computer or network. These programs are engineered to conceal their presence within an infected system, making them extremely challenging to detect due in part to their ability to block some antivirus software and malware scanner software. When a rootkit takes hold, the malicious program sets up a backdoor exploit and may deliver additional malware, such as ransomware, bots, keyloggers or trojans.The computer may act as a "zombie" device, granting the hacker full and remote control over the machine. It is this capability that gives rootkits their potential for harm.

3.5.2 Types of rootkits

Various types of rootkit viruses provide attackers with diverse entry points into computers, allowing them to pilfer data from unsuspecting users.

3.5.2.1 Firmware rootkits

Firmware serves as the embedded software within hardware components, providing fundamental instructions for their operation. Notably, BIOS/UEFI stands as a prime example of firmware, governing the startup process of computers and furnishing essential hardware directives.

A firmware rootkit represents a sophisticated form of malware targeting the firmware of hardware devices. By altering the firmware's code, these rootkits aim to remain hidden and maintain persistent access to the computer system. They accomplish this by embedding themselves within the software that executes during the boot process, preceding the startup of the operating system. This stealthiness poses a significant challenge for detection, as firmware rootkits can endure even through the reinstallation of the operating system.

The prevalence of firmware rootkits has increased in tandem with advancements in technology, particularly the transition from hard-coded BIOS software to remotely updatable BIOS/UEFI software. Additionally, the vulnerability of cloud computing systems, which host multiple virtual machines on a single physical system, has further highlighted the potential risks posed by firmware rootkits.

Examples of firmware rootkits include the UEFI rootkit, Cloaker, and VGA rootkit. These insidious threats underscore the importance of robust cybersecurity measures to safeguard against unauthorized access and data breaches.

3.5.2.2 Kernel mode rootkits

A kernel mode rootkit is a sophisticated form of malware that operates at the deepest level of the operating system. It can inject new code into the kernel or modify existing code, allowing it to hide its presence and activities from detection. These rootkits are complex to develop, and if a kernel rootkit contains bugs, it can severely impact the performance of the target computer. However, the silver lining is that a faulty kernel rootkit often leaves traces that antivirus programs can detect.
Examples of kernel mode rootkits include Spicy Hot Pot, FU, and Knark.

3.5.2.3 Bootloader rootkits

A bootloader is a small program that resides in a computer's firmware and is responsible for initiating the boot process of the operating system. It is the first software program that runs when a computer starts up, loading the necessary files and settings to launch the operating system.

Bootloader rootkits operate in synchronization with the operating system and target critical areas such as the Master Boot Record (MBR) or the Volume Boot Record (VBR). The MBR is the initial code executed upon computer startup, while the VBR contains essential instructions for booting the system or loading an OS/application. By attaching itself to one of these records, a bootloader rootkit evades detection within a standard file system view, posing a challenge for antivirus or rootkit removal tools.
Notable examples of bootloader rootkits include the Stoned Bootkit, Olmasco, and Rovnix.

3.5.2.4 User Mode rootkits

User mode rootkits are a type of malware that operates within the application layer of an operating system. Unlike kernel mode rootkits that delve deep into the core of the

OS, user mode rootkits modify the behavior of application programming interfaces (APIs). These insidious programs can deceive administrators by displaying false information, intercept system calls, filter process output, and execute other actions to mask their presence effectively.

Although user mode rootkits target applications rather than critical system processes, they do leave behind traces or "breadcrumbs" that can trigger alerts from antivirus and rootkit removal tools. This makes them somewhat easier to detect and remove compared to their kernel mode counterparts.
Noteworthy examples of user mode rootkits include Vanquish, Hacker Defender, and Aphex.

3.5.3 The Role of Rootkits

Since the primary goal of a rootkit is to achieve administrator-level access,essentially privileged access to the computer,it is capable of modifying various elements that an administrator is authorized to change. Here's a brief list of what rootkits can do or modify:

- **Concealing Other Malware**: Rootkits mask other types of malware on a device, complicating their removal.
- **Remote Access Provision**: Rootkits provide remote access to the operating system while evading detection. The installation of rootkits is increasingly associated with scams involving remote access.
- **Altering or Disabling Security Programs**: Some rootkits can evade the computer's security programs or even disable them, further complicating the detection and removal of the malware.
- **Data Theft**: Cybercriminals often employ rootkits to steal data. Some hackers target individuals, acquiring their data for identity theft or fraud. Others focus on businesses, engaging in espionage or financial crimes.
- **Creating Persistent Backdoors**: Certain rootkits can establish hidden cybersecurity backdoors in a system, leaving them open for the hacker to return later.
- **Surveillance Purposes**: Rootkits can be utilized as surveillance tools, enabling hackers to spy on activities.
- **Intrusion into Privacy**: With a rootkit in place, the hacker can intercept internet traffic, record keystrokes, and even read emails.

The versatility and stealthiness of rootkits make them potent tools for cyber attackers, emphasizing the critical need for robust cybersecurity practices to defend against their insidious actions[13][14][15].

4 Conlusion

In conclusion, this chapter has provided a comprehensive overview of the various attacks and malware targeting Windows computers. From the disruptive power of DOS/DDOS attacks to the stealthy infiltration of spyware and Trojans, we have explored the diverse array of threats that can compromise the security of Windows systems.

Understanding these attack vectors is crucial for implementing effective defensive strategies. As we move forward, the next chapter will shift focus to the practical side of cybersecurity. We will set up a demonstration laboratory to delve deeper into the execution of these attacks in real-world scenarios.

CHAPTER 3

PREPARATION OF A DEMONSTRATION LABORATORY AND A PRESENTATION FOR THE KEY SOLUTIONS TO OVERCOME CYBER THREATS

Contents

1 Introduction

In this chapter, we delve into the practical aspects of the cyber attacks discussed in the preceding sections. We will explore how these attacks are executed in real-world scenarios within a controlled environment. Additionally, we will present comprehensive solutions aimed at mitigating and overcoming these pervasive cyber threats.

2 Setting up the Demonstration Laboratory

In this section we embarked on setting up our Demonstration Laboratory, a crucial step in our exploration of cybersecurity concepts and practices. Here's how we accomplished it:

2.1 Setting up the virtual machine

Firstly, we initiated the process by configuring the attacking machine, which we named 'Attacking_vm,' on VirtualBox running on our Debian 12 machine. We allocated 6GB of RAM, assigned 3 CPU cores, and provided 157GB of storage space to ensure optimal performance for our Attacking_vm. We opted to install a 64-bit version of Windows 11 Family edition on this VM to replicate real-world scenarios. We also deployed another attacking machine running Kali Linux with 4Go of RAM, 2 cores, and 150Go of storage. All the attacks we executed were targeting a Windows 10 virtual machine.

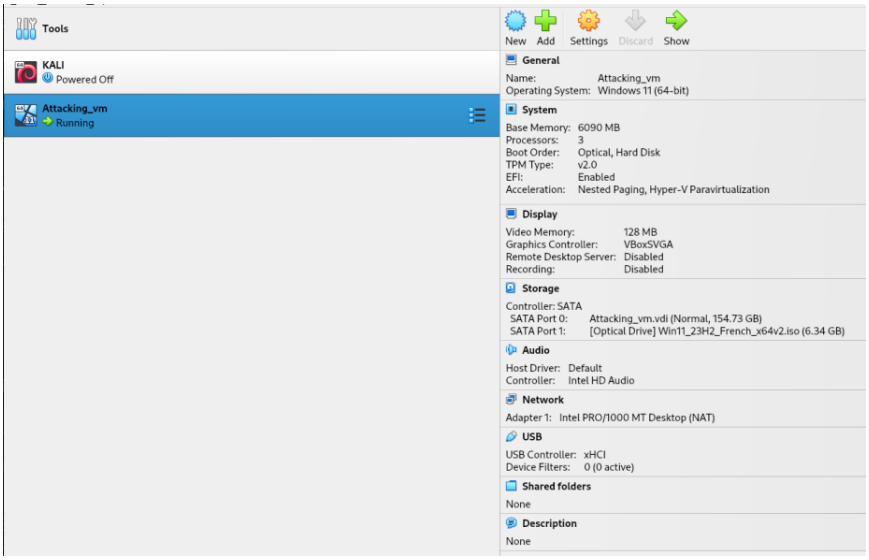

Figure 3.1: The attacking virtual machine configuration

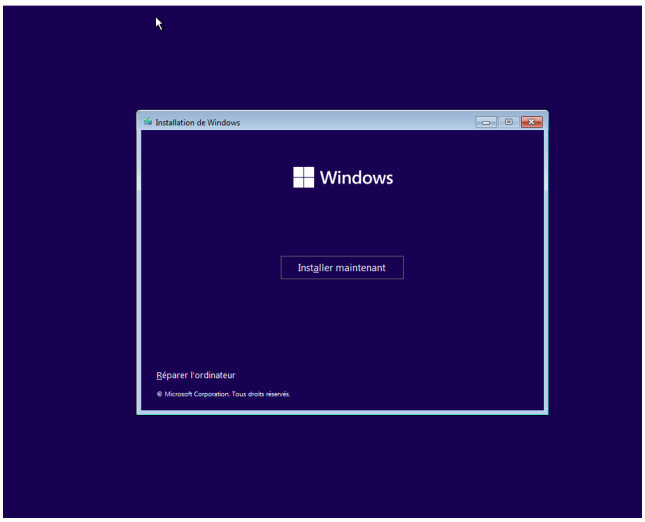

Figure 3.2: Booting the virtual machine and installing Windows

2.2 Creating a backdoor using NJRAT

Next, on this newly created Attacking_vm, we installed and configured Njrat, a well-known remote access Trojan (RAT) tool. This tool allows for the creation of backdoors, providing remote access and control over compromised systems. Configuring Njrat involved setting up communication parameters and defining backdoor behavior.

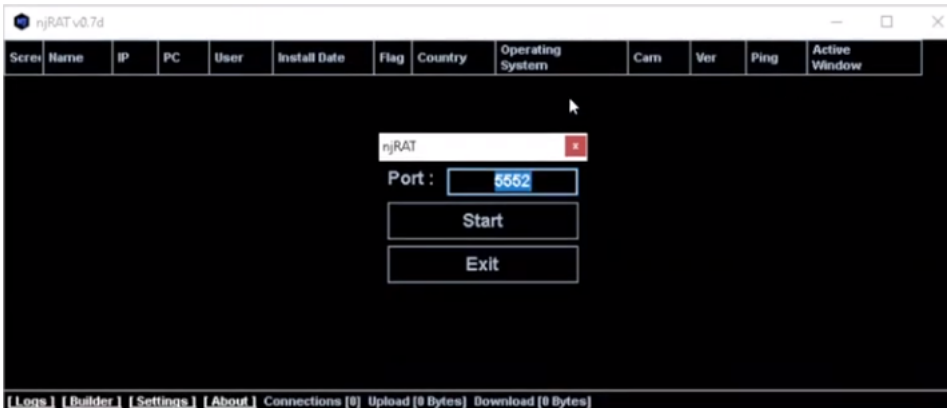

Figure 3.3: Using the 5552 port for the backdoor

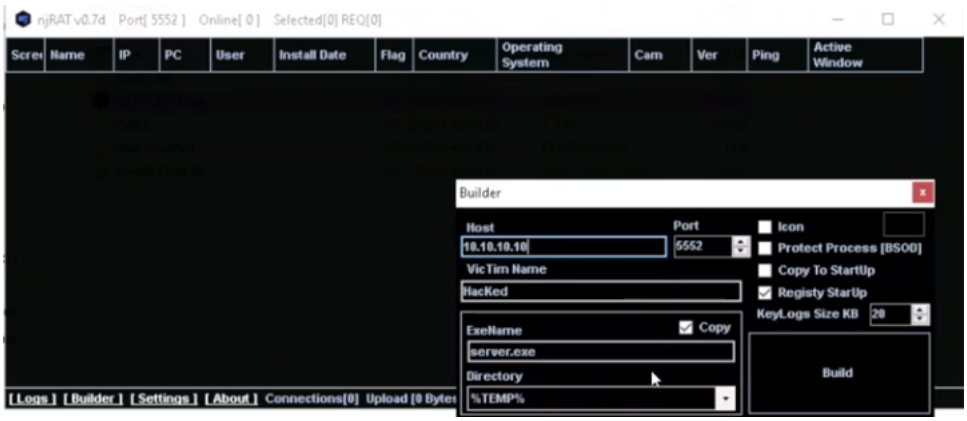

Figure 3.4: Changing the default IP address to the host's IP address

Figure 3.5: Saving the created backdoor as test.exe

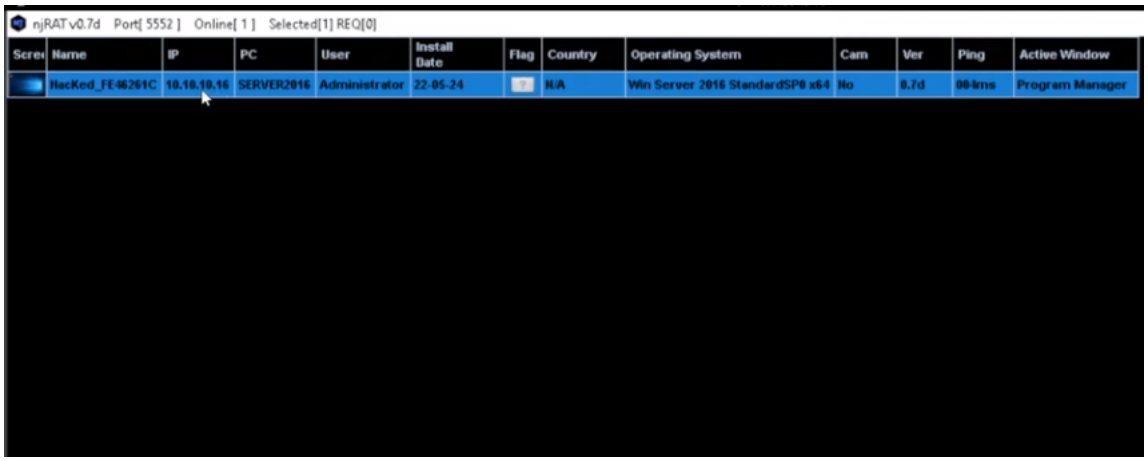

Figure 3.6: After placing the executable on the victim's machine and running it, now we have full access to their device

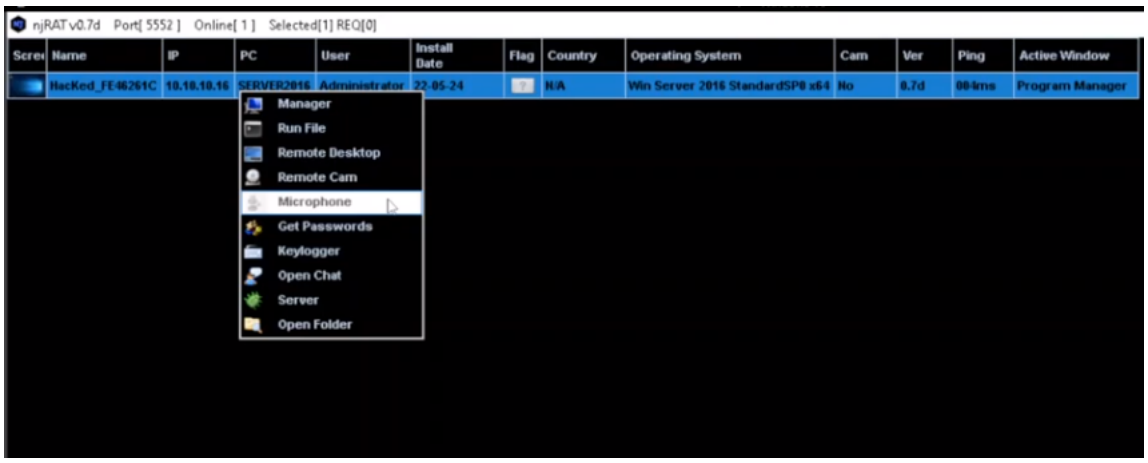

Figure 3.7: This image shows the various actions we can do on the victim's machine

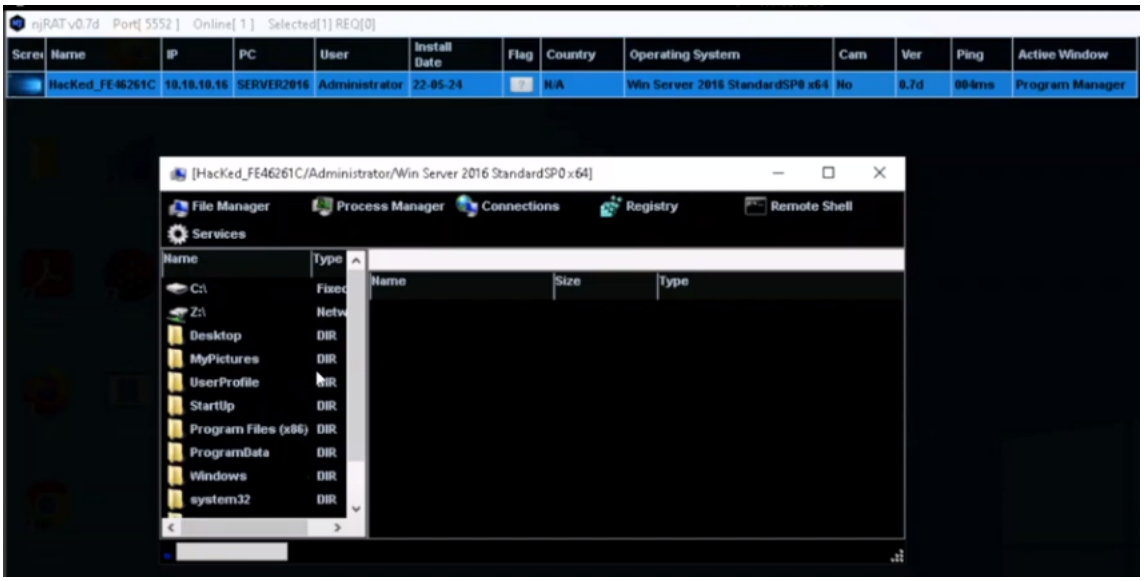

Figure 3.8: We access the victim's file manager and add/delete files as we wish

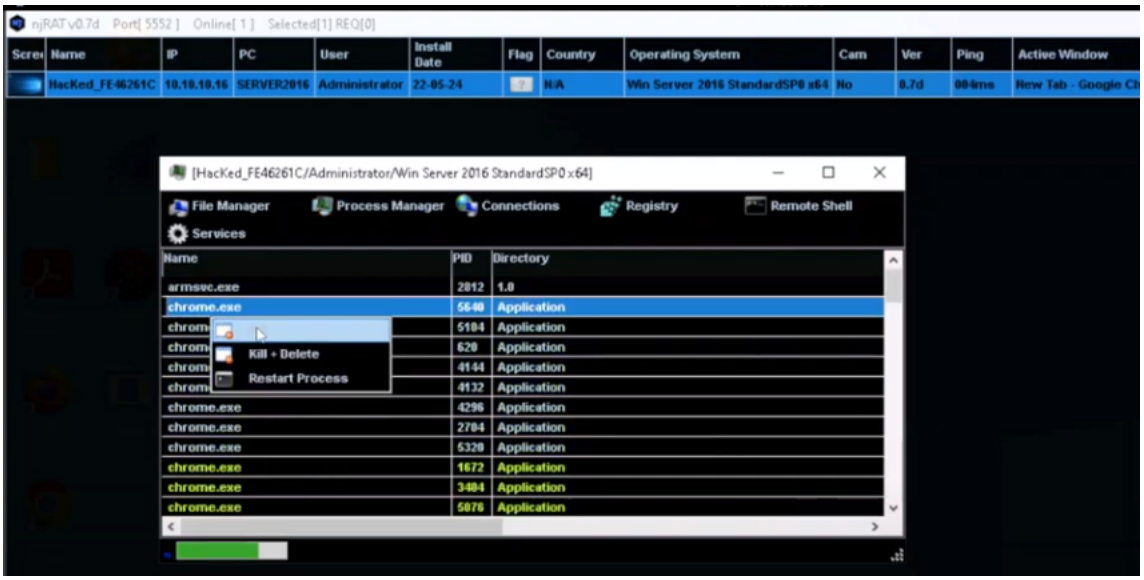

Figure 3.9: We can access running processes on the victim's device and kill any of them as shown in this image

2.3 Demonstration of macro exploitation attack

In this demo, we will showcase the creation of a malicious MS Office file utilizing Metasploit on a Kali Linux system. Additionally, we'll highlight other notable tools such as Empire, Evil Clipper, LuckyStrike, and Magic Unicorn.

Our approach involves leveraging Metasploit to produce malevolent macro-code. This code, upon activation, will trigger the execution of a reverse_https payload. Once operational on the target system, this payload will establish an HTTPS connection with our attacking machine, managed through a distinct Metasploit terminal. This connection enables subsequent exploitation opportunities. **??**

- Step 1: Generating VBA Payload To kick off the process, we access the Metasploit console by entering `msfconsole` and then transition to the `reverse_https` sub-console using the command `use windows/meterpreter/reverse_https`.

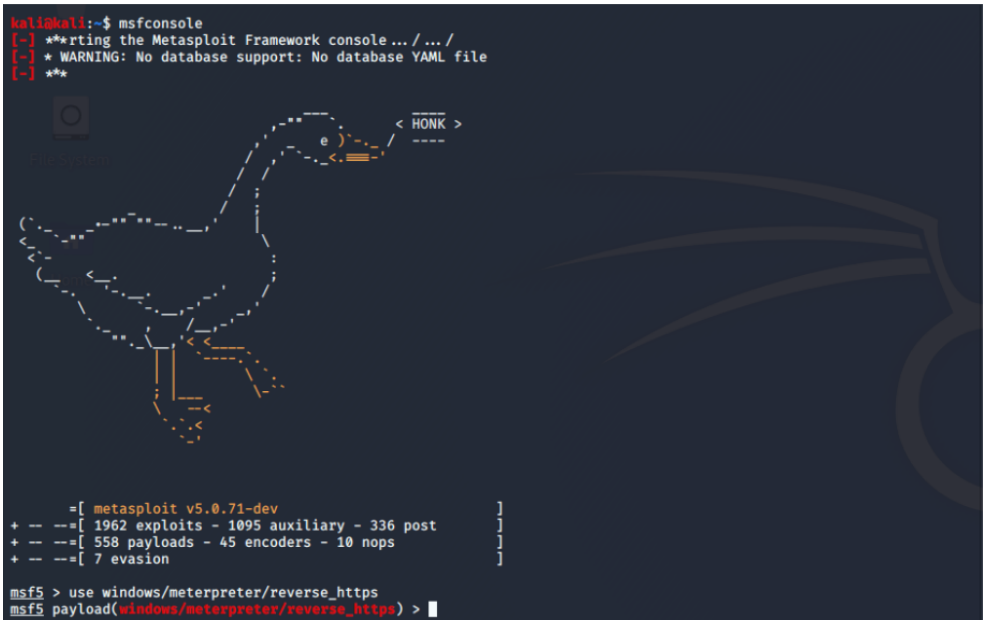

Figure 3.10: "reverse_https" payload sub-console in Metasploit console

The console options become visible when we input `show options`.

Figure 3.11: "reverse_https" sub-console options

Subsequently, we define the host and port the payload will attempt to connect with upon execution. This involves setting our Linux machine's local IP address and a random port. Additionally, we configure the payload to migrate to a new process upon initiating a session by implementing a migration script to run automatically post-execution.

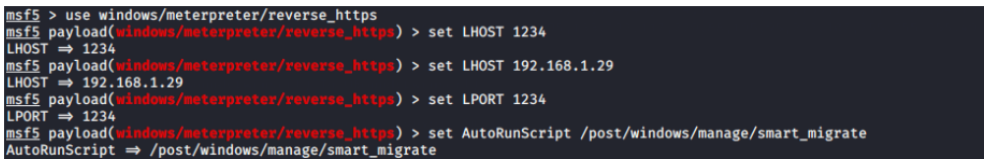

Figure 3.12: Configuring our payload's host, port, and other options

We then proceed to generate the payload in VBA format using the `generate -f vba -o <filename>` command.

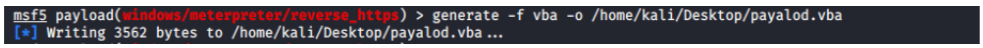

Figure 3.13: Generating the payload code in VBA format and outputting it to a file

- Step 2: Setting Up a Connection Handler On a different terminal, we set up a receiver to capture HTTPS requests from the target's device. This establishes a CNC interface with the victim. As in previous steps, we define the host, port, and payload to listen to.

Figure 3.14: Configuring a handler on a separate Metasploit terminal

The handler is initiated to commence listening for HTTPS requests.

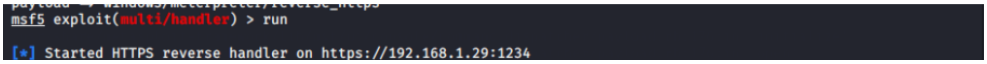

Figure 3.15: The handler is now running and listening for HTTPS requests for 192.168.1.29

- Step 3: Embedding VBA Payload Moving on to a Windows machine, we open our chosen MS Office application, MS Word in this instance. We access the Macros window by selecting View -¿ Macros.

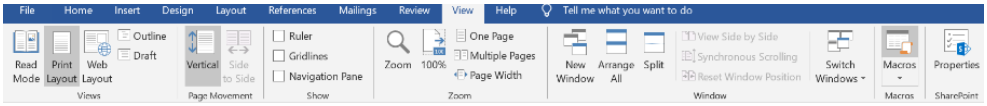

Figure 3.16: MS Office toolbar on a Windows 10 machine, displaying the "Macros" button on the right

Within the Macro Window, we create a new macro and paste the contents of `payload.vba` into the VBA editor.

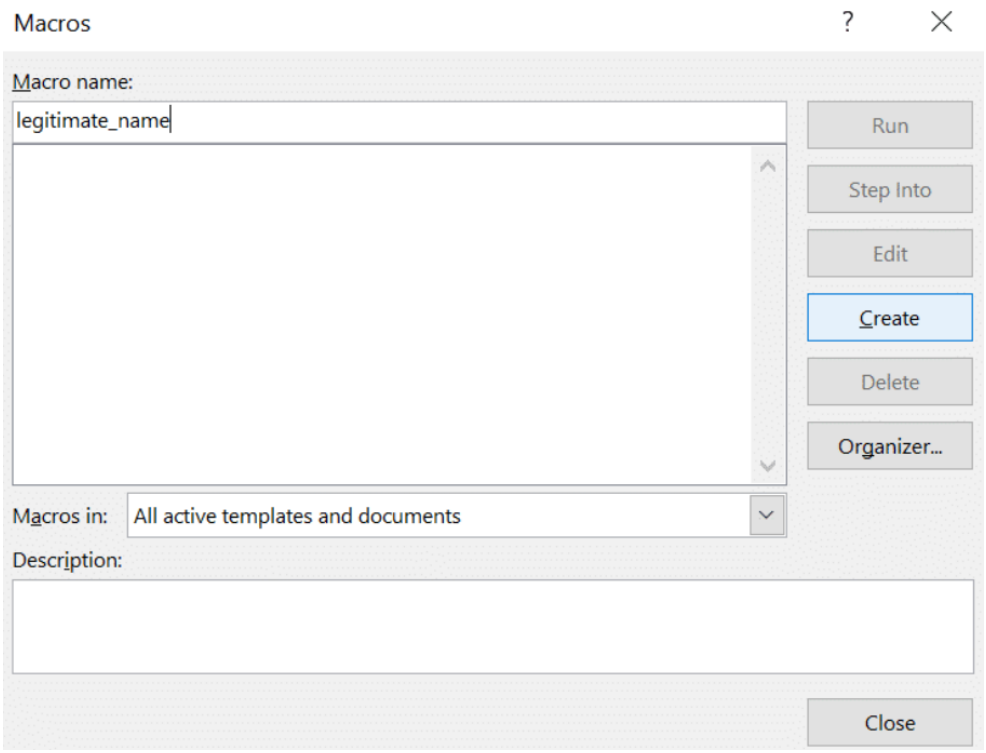

Figure 3.17: Naming and creating a new macro

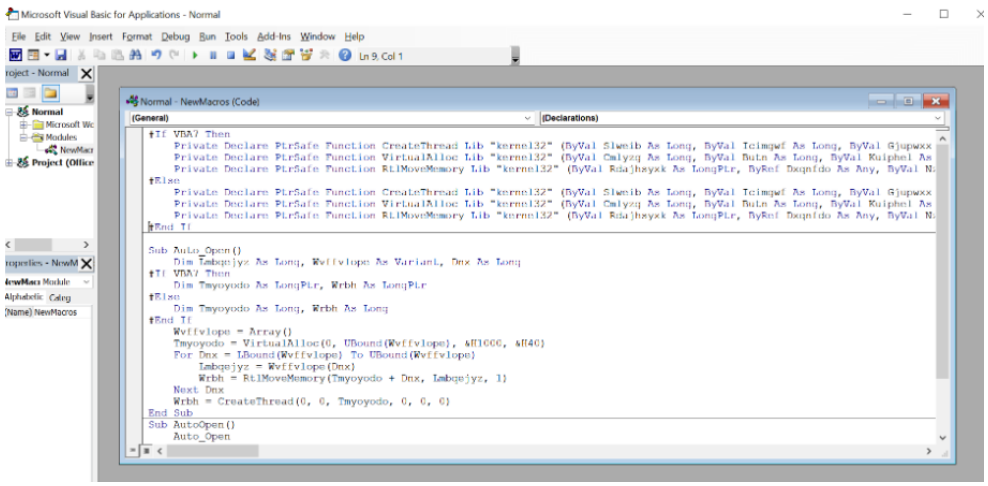

Figure 3.18: Inserting our malicious code into the VBA editor

With all preparations complete (aside from tailoring the document to entice a specific victim), opening the file prompts the victim to either enable macros automatically or via the "Enable Content" Button.

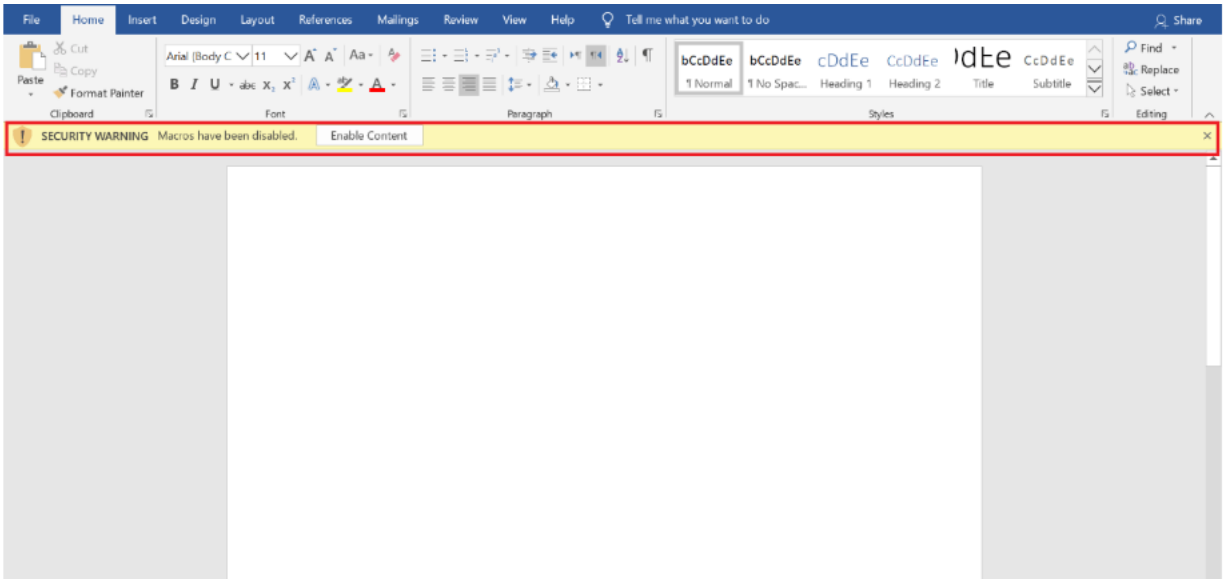

Figure 3.19: When our malicious file is opened, the victim is prompted to enable macros.

2.4 Demonstration of a phishing attack

In this demo, we will try to create a fake website using a tool called Blackeye preinstalled on Kali Linux.

Next, and after sending the website to the targeted victim, we wait until they type anything and it will be displayed on our blackeye tool.

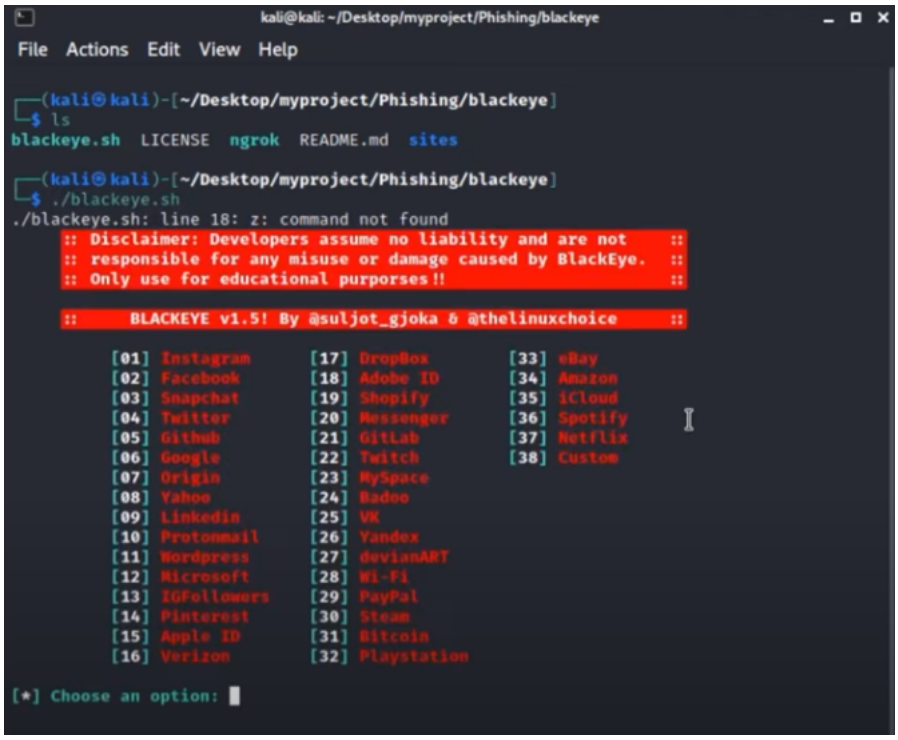

Figure 3.20: Blackeye tool

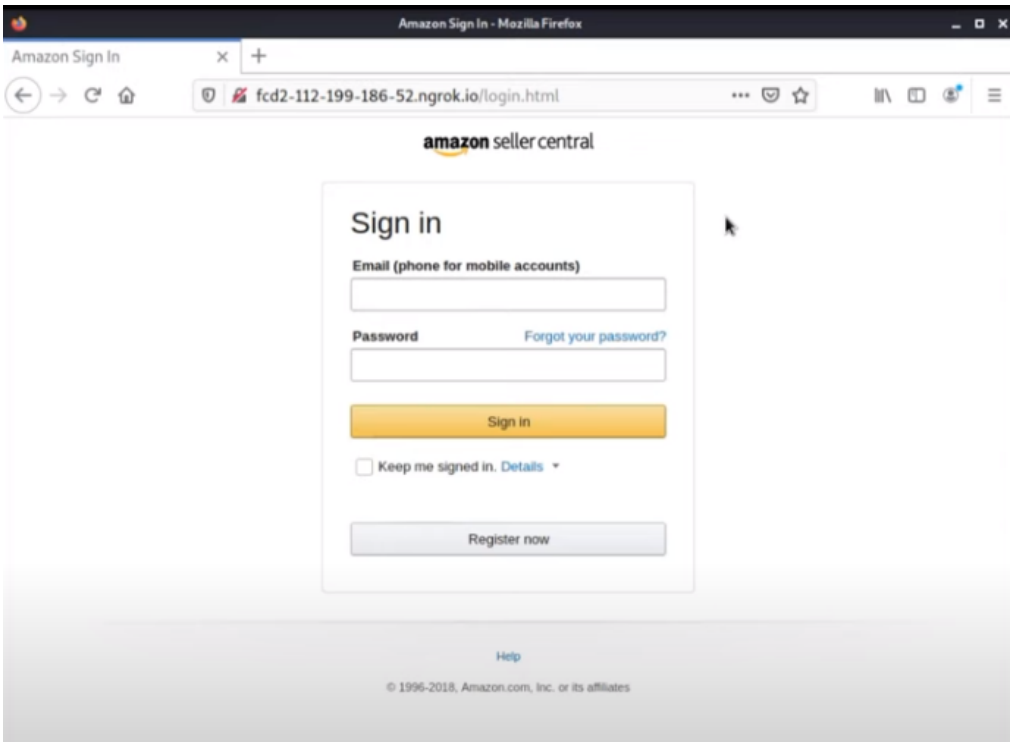

Figure 3.21: Creating a fake site using blackeye

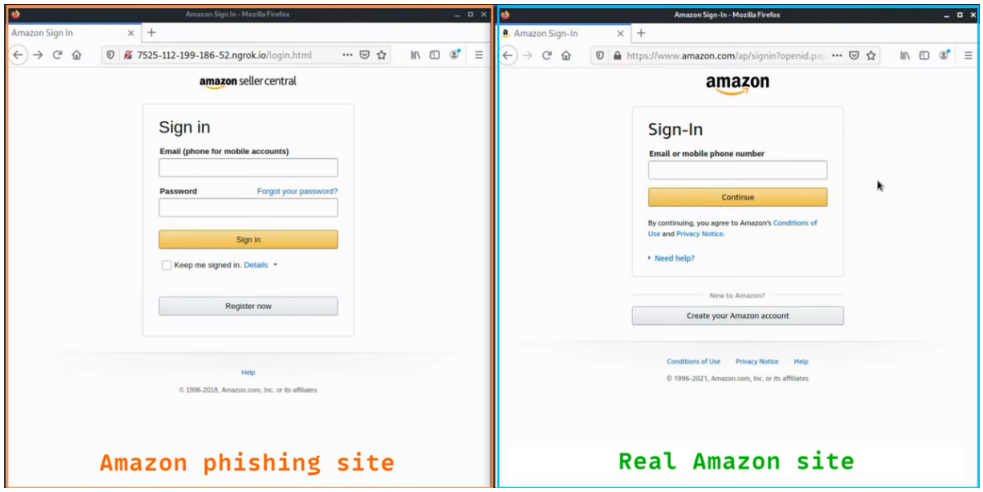

Figure 3.22: This image shows how similar the fake and the real sites look

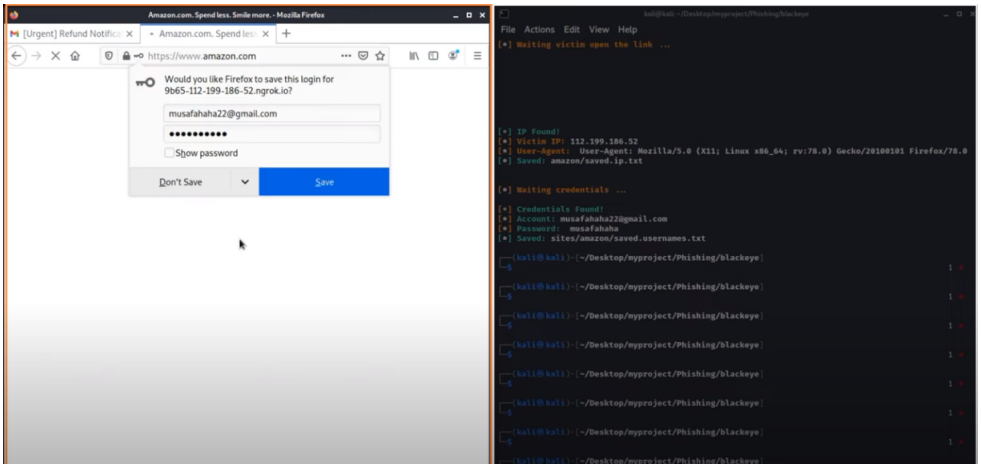

Figure 3.23: This image shows that everything typed by the victim on the login page is displayed on the attacker's machine

3 Techniques for Protection Against Attacks on Windows Computers

Protecting Windows computers from cyber attacks requires a comprehensive approach that combines best security practices with the use of appropriate security tools. This section delves into various strategies to mitigate the risk of attacks on Windows systems

3.1 Best Security Practices

Implementing the following best practices can significantly enhance the security posture of Windows computers:

- **Regular Software Updates:** Maintaining the latest software versions is vital for fixing identified weaknesses. To guarantee prompt security updates for Windows, Microsoft Office, and other Microsoft programs, it is recommended to enable automatic updates in Windows Update. Furthermore, activating automatic updates for non-Microsoft software, such as web browsers and Adobe Acrobat Reader, is essential to stay safeguarded from new and evolving risks.
- **Use of Strong Passwords:** Ensuring robust password security is crucial in preventing unauthorized access attempts. Encourage users to form resilient, distinct passwords and activate multifactor authentication whenever feasible. Educate users

on password cleanliness, stressing the significance of avoiding ordinary phrases and frequently refreshing passwords.

- **Firewall Protection:** Enable the built-in Windows Firewall to regulate incoming and outgoing network traffic. Firewalls act as a barrier against unauthorized access and help prevent malware infections by monitoring and controlling network communications.
- **User Vigilance:** Educate users about common cyber threats such as phishing emails and malicious attachments. Advise them to exercise caution when opening email attachments or clicking on unfamiliar links. Remind users to verify the authenticity of websites before entering sensitive information and to report suspicious activities promptly.
- **Safe Browsing Habits:** Instruct users to browse the web safely by avoiding potentially malicious websites and refraining from downloading pirated content. Encourage the use of modern, secure web browsers like Microsoft Edge, which offer built-in protection against phishing attempts and malicious downloads.
- **External Device Security:** Caution users against using unauthorized external devices such as USB drives or external hard drives. Remind them to scan external devices for malware before accessing any files and to only use devices from trusted sources.

3.2 Recent security tools used by Windows operating system

Security and privacy depend on an operating system that guards your system and information from the moment it starts up, providing fundamental chip-to-cloud protection. Windows 11 is the most secure Windows yet with extensive security measures designed to help keep our privacy. These measures include built-in advanced encryption and data protection, robust network and system security, and intelligent safeguards against ever-evolving threats. In this subsection, we will present the most important security features of the latest Windows operating system (Windows 11) [6]

3.2.1 System Security

Feature	Description
Secure Boot and Trusted Boot	Secure Boot prevents malware and corrupted components from loading during device startup. Trusted Boot extends this protection by verifying the boot process, ensuring a safe and secure boot-up sequence.
Measured Boot	Measures important code and configuration settings during Windows boot-up, storing the measurements securely in the TPM and providing anti-malware software with a trusted log of boot components.
Device Health Attestation Service	Confirms the integrity of the device, firmware, and boot process before granting access to corporate resources, using data stored in the TPM and verified by an attestation service.
Windows Security Policy Settings and Auditing	Provides a robust set of security policies for IT administrators to protect Windows devices and resources in their organization.
Assigned Access	Offers locked-down experiences for specialized devices, such as single-app or multi-app kiosks, managed through Assigned Access to limit user access to specific applications.
Windows Firewall	Host-based, two-way network traffic filtering that blocks unauthorized traffic based on network properties, reducing the attack surface and complementing existing security solutions.

Table 3.1: System Security Features in Windows 11

3.2.2 Virus and Threat Protection

Feature	Description
Microsoft Defender Antivirus	Provides continuous monitoring for malware, viruses, and security threats from the moment Windows boots up. Includes real-time, behavior-based, and heuristic antivirus protection.
Local Security Authority (LSA) Protection	Enhances protection against credential theft by allowing only trusted, signed code to load, significantly reducing the risk of unauthorized access.

Feature	Description
Attack Surface Reduction (ASR)	Helps prevent software behaviors commonly exploited to compromise devices or networks, thereby reducing overall vulnerability.
Tamper Protection Settings for MDE	Safeguards against attempts to disable or change critical security settings, preventing bad actors from exploiting vulnerabilities.
Controlled Folder Access	Protects specific folders by managing app access, allowing only trusted apps to modify files within protected folders, thus mitigating the risk of ransomware attacks.
Microsoft Defender SmartScreen	Defends against phishing, malware websites, and potentially malicious file downloads, with customizable notifications for IT administrators.
Microsoft Defender for Endpoint	Offers advanced threat detection and response capabilities, allowing organizations to detect, investigate, and respond to sophisticated threats effectively.

Table 3.2: Virus and Threat Protection Features in Windows 11

3.2.3 Network Security

Feature	Description
Transport Layer Security (TLS)	Implements the latest TLS 1.3 protocol by default, enhancing communication security over networks with improved encryption and performance.
Domain Name System (DNS) Security	Supports DNS over HTTPS (DoH), encrypting DNS queries to protect against on-path attackers and ensure secure name resolution.
Bluetooth Pairing and Connection Protection	Provides secure pairing protocols and updates to ensure Bluetooth devices maintain security standards and protect against potential vulnerabilities.
Wi-Fi Security	Supports WPA3 certification for enhanced Wi-Fi network security, providing more robust encryption and protection against unauthorized access.

Feature	Description
Opportunistic Wireless Encryption (OWE)	Enables encrypted connections to public Wi-Fi hotspots, ensuring secure communication even in untrusted network environments.
Windows Firewall	Offers host-based, two-way network traffic filtering to block unauthorized traffic and reduce the attack surface of devices.
Virtual Private Network (VPN)	Integrates built-in VPN protocols and configuration support for secure remote access to network resources, with simplified management and control options.

Table 3.3: Network Security Features in Windows 11

3.2.4 Encryption and Data Protection

Feature	Description
BitLocker Management	Allows management of BitLocker encryption features through MDM solutions, facilitating secure encryption of OS volumes and removable storage.
BitLocker Enablement	Utilizes BitLocker Drive Encryption to protect against data theft by encrypting volumes with AES algorithm, integrated with TPM for secure key storage.
Encrypted Hard Drive	Employs self-encrypting drives at the hardware level to provide full disk encryption transparently, enhancing data security and management benefits.
Personal Data Encryption (PDE)	Seamlessly encrypts user documents and files using BitLocker and Windows Hello for Business, ensuring data protection even when devices are locked.
Email Encryption (S/MIME)	Enables encryption of outgoing email messages and attachments, ensuring confidentiality and integrity of communication with digital signatures.

Table 3.4: Encryption and Data Protection Features in Windows 11

4 Conclusion

In this chapter, we've explored the practical side of cybersecurity, setting up a Demonstration Laboratory to simulate real-world cyber attacks. From creating backdoors to executing phishing and macro exploitation attacks, we've highlighted the vulnerabilities organizations face. We've also outlined best practices and security tools, like Windows 11's built-in features, to fortify defenses against such threats. Ultimately, this chapter underscores the importance of proactive cybersecurity measures in safeguarding against cybercrime.

GENERAL CONCLUSION AND PERSPECTIVES

In this extensive report, we have thoroughly examined the intricacies of safeguarding Windows computers. Our goal was not only to comprehend the dangers but also to equip ourselves with practical strategies for defense. We started by delving into the fundamental aspects of Windows operating system technology and analyzing the widespread menace of malware. From there, we explored the specific attacks that target Windows systems and the unique characteristics of malware designed for this operating environment.

Moving from theory to application, Chapter 3 saw us establishing a demonstration laboratory. Through hands-on exercises, we gained invaluable experience in identifying vulnerabilities and implementing defense measures, ranging from basic security protocols to advanced encryption techniques.

Ultimately, this journey has yielded two significant outcomes. Firstly, we have developed a deeper understanding of the complexities of Windows computer security, empowering us to effectively recognize and mitigate potential risks. Secondly, armed with practical skills and knowledge, we are now better prepared to protect Windows systems against ever-evolving cyber threats.

BIBLIOGRAPHY

[1] https://www.researchgate.net/publication/309766493_International_Journal_of_Advance_Research_in_Computer_Science_and_Management_Studies_Introduction_to_Malware_and_Malware_Analysis_A_brief_overview

Consulted on 17/03/2024

[2] https://www.crowdstrike.com/cybersecurity-101/malware/

Consulted on 17/03/2024

[3] https://www.elprocus.com/what-is-an-operating-system-and-its-components/

Consulted on 19/03/2024

[4] https://www.avg.com/fr/signal/computer-security-exploits

Consulted on 20/03/2024

[5] https://www.avg.com/en/signal/what-is-a-macro-virus

Consulted on 21/03/2024

[6] https://learn.microsoft.com/en-us/windows/security/operating-system-security/

Consulted on 125/03/2024

[7] https://www.geeksforgeeks.org/how-to-prevent-man-in-the-middle-attack/

Consulted on 25/03/2024

[8] https://www.geeksforgeeks.org/mitm-man-in-the-middle-attack-using-arp-poisoning/
Consulted on 26/03/2024

[9] https://www.geeksforgeeks.org/ip-spoofing/
Consulted on 29/03/2024

[10] https://www.checkpoint.com/cyber-hub/threat-prevention/ransomware/
Consulted on 30/03/2024

[11] https://www.fortinet.com/resources/cyberglossary/trojan-horse-virus
Consulted on 01/04/2024

[12] https://www.avast.com/c-trojan
Consulted on 01/04/2024

[13] https://www.fortinet.com/resources/cyberglossary/rootkit#:~:text=A%20common%20rootkit%20definition%20is%20a%20type%20of,remain%20hidden%20for%20a%20long%20period%20of%20time.
Consulted on 02/04/2024

[14] https://www.avast.com/fr-fr/c-rootkit
Consulted on 02/04/2024

[15] https://www.crowdstrike.com/cybersecurity-101/malware/rootkits/
Consulted on 03/04/2024

[16] https://www.illumio.com/blog/types-malicious-payloads
Consulted on 04/04/2024

[17] https://www.varonis.com/fr/blog/quest-ce-que-le-cc
Consulted on 06/03/2024

[18] https://www.cloudflare.com/learning/ddos/glossary/denial-of-service/
Consulted on 07/04/2024

[19] https://www.malekal.com/syn-flood/
Consulted on 07/04/2024

[20] https://www.cloudflare.com/learning/ddos/udp-flood-ddos-attack/
Consulted on 09/04/2024

[21] https://www.cloudflare.com/learning/access-management/phishing-attack/
Consulted on 10/04/2024

[22] https://www.avast.com/fr-fr/c-phishing
Consulted on 10/04/2024

[23] https://www.frontiersin.org/articles/10.3389/fcomp.2021.563060/full
Consulted on 11/04/2024