A · P · U

ASIA PACIFIC UNIVERSITY
OF TECHNOLOGY & INNOVATION

INDIVIDUAL ASSIGNMENT

NAME (TP NUMBER)	**:**	Koo Wai Kit (TP081761)
INTAKE CODE	**:**	APUMF2406CYS
MODULE TITLE	**:**	Cyber Security and Threats (072024-NOR)
MODULE LECTURER	**:**	Nor Azlina Abdul Rahman
PROJECT TITLE		CaaS Assignment 1
DATE ASSIGNED	**:**	12 July 2024
DATE COMPLETED	**:**	25 July 2024

Table of Contents

1. Abstract

The increasing reliance on computers and the internet has turned cyberspace into a new battleground for criminal activities, especially with the rise of cybercrime-as-a-service (CaaS). This report delves into a specific aspect of CaaS: Living Off the Land Binaries (LOLBins) attacks, using QBot (Qakbot) as a case study. QBot is a well-known banking Trojan with advanced techniques, as it utilizes PowerShell for script execution and BITSAdmin to download malicious components during email phishing campaigns. The research investigates the vulnerabilities that enable such attacks, supported by the MITRE Attack framework and Common Vulnerabilities and Exposures (CVE) database. Additionally, the tactics, techniques, and procedures (TTPs) used by attackers in LOLBins attacks are examined in this report. A comprehensive cybersecurity plan for organizations is provided, covering threat modeling, operational security (OPSEC), and both general and technical security measures. Through this analysis, the report aims to equip cybersecurity professionals with the knowledge and strategies needed to counteract these sophisticated threats.

2. Introduction

As digital transformation accelerates across all sectors, the threat landscape in cybersecurity has grown increasingly complex. With more individuals and organizations connected to the internet than ever before, the risk of cyber-attacks has escalated. Cybercriminals have leveraged these developments to devise new methods of attack that can inflict significant damage without requiring any physical force. Among these emerging threats is cybercrime-as-a-service (CaaS), a business model that enables malicious actors to purchase or rent cyber-attack tools and services from the dark web (Field Effect, 2023). By leveraging this model, attackers are able to launch sophisticated cyber-attacks with minimal effort and almost total anonymity (Field Effect, 2023).

One particularly stealthy method facilitated by CaaS is the use of Living Off the Land Binaries (LOLBins). LOLBins attacks exploit legitimate system tools and binaries that are already present on the victim's system, making detection and mitigation extremely challenging (Trabelsi, 2024). These attacks allow cybercriminals to operate stealthily, avoiding traditional security measures that typically focus on identifying and blocking external threats.

This report focuses on a case study of QBot (Qakbot), a well-known banking Trojan that has adapted LOLBins techniques in its operations (Maharjan, 2022). QBot has become a formidable threat due to its ability to use PowerShell for script execution and BITSAdmin to download malicious components during email phishing campaigns. The vulnerabilities that enable QBot attacks are well-documented in the Mitre Attack framework and the Common Vulnerabilities and Exposures (CVE) database. These resources provide a comprehensive overview of the tactics, techniques, and procedures (TTPs) used by attackers to exploit system weaknesses and execute QBot attacks. Moreover, the report outlines a strategic cybersecurity plan tailored for organizations to defend against QBot and similar threats. This plan includes threat modeling, operational security (OPSEC), and both general and technical security measures, offering a robust framework for mitigating the risks associated with CaaS attacks.

In conclusion, the increasing sophistication of cyber threats like QBot, which leverages LOLBins techniques, underscores the need for organizations to adopt advanced cybersecurity measures. This report aims to provide the necessary insights and strategies to mitigate the risks posed by CaaS malware, which includes QBot and other similar malware in the evolving landscape of cybercrime.

3. Overview of Cybercrime-as-a-Service (CaaS)

3.1 Definition and business model

Cybercrime-as-a-Service (CaaS) is a structured crime model in which cybercriminals provide their tools, skills, and services to others, usually for monetary profit (Chebac, 2023). This model has turned cybercriminal activities into commodities, making them accessible even to those with limited technical knowledge (Field Effect, 2023). Chebac states that CaaS transforms hacking into a subscription service available to anyone willing to pay for it on the dark web (2023).

CaaS encompasses a wide range of cybercriminal activities, including financial fraud, malware attacks, phishing, ransomware, distributed denial of service (DDoS) attacks, and social engineering (Chebac, 2023). This business model includes a wide range of offerings, from malware kits and exploit tools to distributed denial-of-service (DDoS) attacks and phishing campaigns (Unni, 2022). For instance, a DDoS booter can be rented for as little as

$60 per day, while more sophisticated ransomware kits can cost thousands of dollars (Chebac, 2023).

The evolution of CaaS has been driven by the increasing professionalization of cybercrime. As the demand for cyber-attacks has grown, the supply of CaaS providers who continuously innovate and refine their offerings has also increased. Chebac mentions that the CaaS model operates much like a legitimate business, with roles such as engineers, developers, and tech support representatives (2023). The CaaS organizations often employ money mules to launder the proceeds of their illegal activities.

3.2 Impact on the cyber threat landscape

The rise of CaaS has significantly transformed the cyber threat landscape by lowering the entry barriers for cybercriminals. Previously, conducting a sophisticated cyber-attack required substantial technical knowledge and resources. However, with CaaS, even novice attackers can launch advanced and targeted attacks (Field Effect, 2023). This has led to an increase in the volume, variety, and complexity of cyber-attacks globally. Field Effect (2023) states that since its relatively easy to get involved in cybercrime, the number of cyberattacks has increased significantly, with a 38% rise in 2022 alone.

CaaS has also contributed to the rapid spread of new attack techniques and malware variants. As service providers compete for customers, they frequently introduce new tools and features, accelerating the innovation cycle within the cybercriminal community (Moore, 2023). This results in a constantly evolving threat environment, where defenders must adapt quickly to new and emerging threats.

Furthermore, the anonymity afforded by CaaS makes it difficult for law enforcement to track and prosecute the perpetrators. Transactions in the CaaS market are often conducted using cryptocurrencies, which further obscures the identities of those involved (Field Effect, 2023). This anonymity, coupled with the global reach of the internet, complicates the efforts to combat cybercrime.

3.3 Main services offered

CaaS encompasses a wide array of services that cater to different types of cyber-attacks. According to Chebac (2023), there are four most common types of CaaS services, shown in Table 1 below:

Table 1: Common types of CaaS services *(Chebac, 2023)*

CaaS Service	Description
Ransomware-as-a-Service (RaaS)	This service allows cybercriminals to buy or rent ransomware tools to encrypt victims' data and demand ransom, making it easy for attackers with minimal technical skills to launch attacks.
DDoS-as-a-Service	Provides access to botnets for launching distributed denial-of-service (DDoS) attacks that overwhelm servers with traffic, causing operational disruptions and potential financial losses for businesses.
Phishing-as-a-Service (PhaaS)	Offers comprehensive phishing kits, including pre-made phishing pages and messages, enabling cybercriminals to conduct targeted phishing attacks with minimal effort.
Malware-as-a-Service (MaaS)	Sells ready-to-use malware kits and platforms for spreading viruses and executing trojan attacks, allowing customers to run malicious campaigns through a subscription-based service.

4. Overview of Living Off the Land Binaries (LOLBins)

4.1 Definition and concept

Living Off the Land Binaries (LOLBins) refer to legitimate system binaries and tools that are present on a computer but are exploited by attackers for malicious purposes (Mohanlal & Barlow, 2021). Unlike traditional malware that requires the introduction of new software, LOLBins use existing system tools and utilities to carry out attacks, making detection and prevention more challenging. Their legitimate nature allows them to bypass conventional security measures that typically focus on identifying malicious software (Trabelsi, 2024).

Originally, LOLBins were mostly employed in post-exploitation phases to ensure persistence or elevate privileges on a compromised system (Mohanlal & Barlow, 2021). Over time, their role has broadened to encompass bypassing security defenses and facilitating malware delivery. By exploiting these trusted system binaries, attackers can run their malicious payloads without needing custom code or external files, which helps them evade detection.

4.2 Common LOLBins in Windows environment

In Windows environments, several system binaries are frequently targeted for abuse in LOLBins attacks. Some key examples can be summarized in Table 2 below:

Table 2: Most common LOLBins and their usages *(Goss, 2024)*

LOLBins	**Usage**
PowerShell	A versatile scripting language in Windows used for automation and administration that can be exploited to download files, execute scripts, and perform reconnaissance, though it is often restricted in enterprise environments.
BITSAdmin	A command-line tool for managing Background Intelligent Transfer Service (BITS) that facilitates file uploads and downloads between machines and can be misused to hide malware execution as BITS jobs, offering an alternative to PowerShell for file transfer.
Windows Management Instrumentation (WMIC)	A command-line tool for system management that can be used to execute commands and scripts on remote systems, gather information, and dump credentials.
MSHTA (Microsoft HTML Application Host)	A tool for running Microsoft HTML Applications (HTA) that allows execution of malicious scripts embedded in HTAs, providing a way to run code similarly to macros or web files.
Certutil	A command-line utility for managing certificates that can be exploited to download files and encode/decode data, helping attackers evade detection and serving as an alternative to PowerShell download methods.

4.3 Mechanisms of LOLBins exploitation

LOLBins can be exploited through various malicious activities that leverage their legitimate functions for malicious purposes. Table 3 below outlines the key mechanisms through which LOLBins are used to conduct attacks. Each mechanism is described to highlight its role in leveraging LOLBins for malicious purposes.

Table 3: LOLBins exploitation *(Rumiantseva, 2023)*

Mechanism	**Description**
Bypassing security measures	Help attackers evade traditional security controls by using scripts deemed legitimate by the system, such as running arbitrary code with mshta.exe while bypassing application whitelisting and other defenses.
Covert execution	Attackers use LOLBins to conduct malicious activities within normal system processes, making it hard to distinguish between benign and harmful actions.
Privilege escalation	Enable attackers to gain higher access levels on compromised systems by exploiting vulnerabilities or misconfigurations, often bypassing User Account Control (UAC).
Lateral movement	Attackers leverage LOLBins to navigate through networks and systems, using basic utilities to identify and exploit additional targets.
Data exfiltration	Facilitate the extraction of sensitive data from compromised systems by using existing tools to encrypt, compress, and transfer information, avoiding standard Data Loss Prevention (DLP) measures.
Remote command execution	Allow attackers to remotely execute commands on compromised systems, controlling and manipulating them to perform further malicious activities.

4.4 Advantages of LOLBins for attackers

LOLBins allow threat actors to perform covert operations within compromised systems, avoiding detection by blending their actions with legitimate system processes (Trabelsi, 2024). This stealthy approach, often involving fileless execution where malicious activities occur directly in memory without leaving traces on disk, complicates detection efforts. By exploiting trusted system utilities, LOLBins evade traditional security measures, allowing attackers to maintain a persistent presence and conduct malicious activities undetected. Trabelsi (2024) further explains that evasion techniques used with LOLBins include fileless malware, which resides in memory rather than on disk, and obfuscated command line arguments, which mask malicious actions within complex or legitimate-seeming commands.

5. Overview of QBot (Qakbot)

5.1 Background

QBot, also known as Qakbot or Pinkslipbot, is a malware trojan that has been used by cybercriminals for over 15 years (Vicente, 2024). QBot was first discovered in 2007 as a tool for stealing banking information and committing financial frauds, but has evolved significantly since its initial emergence, becoming a more versatile malware. In recent years, it has evolved into an initial access broker, often deploying Cobalt Strike for lateral movement, leading to secondary infections like the BlackBasta ransomware (Vicente, 2024).

QBot has been involved in several high-profile attacks, including those on JBS, the world's largest meat producer, and Fujifilm, a Japanese multinational conglomerate, both occurring in early June 2021 (Maharjan, 2022). Both incidents were linked to the REvil gang, which likely used QBot for initial infection. Vicente (2024) states that despite being shut down by authorities in 2023, the creators of QBot quickly released a new version, showing their determination and ability to adapt. The group behind Qakbot has released five distinct versions of the malware, with the most recent update occurring in December 2023.

5.2 Capabilities

According to Maharjan (2022) Qbot's core capabilities include reconnaissance through process injection to run discovery commands. It also enables lateral movement via Windows Management Instrumentation (WMI) and facilitates privilege escalation and persistence by creating scheduled tasks and manipulating the registry. Qbot is adept at credential harvesting, targeting multiple locations including browser data like cookies and history, and focuses on data exfiltration, particularly emails. Additionally, it delivers other payloads, like deploying Cobalt Strike to deliver further malicious payloads or to sell access to other threat actors.

Additionally, QBot has enhanced anti-analysis capabilities, allowing it to bypass malware sandboxes, antivirus programs, and other security defenses (Vicente, 2024). The malware's has a modular architecture, which enables it to download plugins that add new functionalities as needed (BlackBerry, n.d.). Its various modules enable automated targeting of financial data, locally stored emails, system passwords or hashes, website credentials, and browser cache cookies. It can also log keystrokes to capture typed credentials.

6. Vulnerabilities that Enable QBot Attacks

QBot has leveraged various vulnerabilities in widely used software and operating systems to infiltrate networks, escalate privileges, and evade detection. These vulnerabilities, often found in critical components of systems, provide entry points or enable deeper penetration within an organization's infrastructure. Exploiting these weaknesses, QBot can bypass traditional security measures, making it a persistent and dangerous threat. Below, we highlight some of the key vulnerabilities that have been exploited by QBot in recent attacks, emphasizing the importance of timely patch management and robust vulnerability assessment practices. In addition, the Common Weakness Enumeration (CWE), Common Vulnerability Scoring System (CVSS) score and severity level will be provided for each vulnerability.

6.1 CVE-2021-34527

CVE-2021-34527, widely known as the PrintNightmare vulnerability, impacts the Windows Print Spooler service, allowing attackers to execute arbitrary code remotely with system-level privileges (CVE Program, 2024). No CWE has been provided for this vulnerability (CVE Program, 2024). A CVSS score of 8.8 and a high severity level has been assigned to this vulnerability.

QBot exploits this vulnerability by leveraging it to perform privileged file operations, allowing it to gain elevated access on compromised systems (Maharjan, 2022). By exploiting this flaw, QBot can execute actions with higher system privileges, which facilitates its ability to move laterally within the network, maintain persistence, and carry out various malicious activities more effectively.

6.2 CVE-2022-30190

CVE-2022-30190, commonly referred to as the Follina vulnerability, affects the Microsoft Windows Support Diagnostic Tool (MSDT). According to CVE Program (2024), a remote code execution vulnerability in MSDT allows attackers to execute arbitrary code with the same privileges as the calling application, such as Microsoft Word. By exploiting this flaw, attackers can install programs, modify or delete data, and create new accounts based on the user’s permissions. No CWE has been provided for this vulnerability (CVE Program, 2024). A CVSS score of 7.8 and a high severity level has been assigned to this vulnerability.

QBot exploits this vulnerability by using it to download and execute malicious payloads (Maharjan, 2022). The attack begins with a compromised file that contains an Office document, a shortcut file (.lnk), and a DLL file within a disk image (.img). When the document is opened, the shortcut file launches the DLL, which then executes a PowerShell script exploiting CVE-2022-30190. Maharjan (2022) also mentions that despite the vulnerability being patched, attackers have adapted by using native binaries and LOLBins (Living Off the Land Binaries) to circumvent security measures and ensure the success of their attack.

6.3 CVE-2023-36033

CVE-2023-36033 is a vulnerability in the Windows Desktop Window Manager (DWM) core library that allows attackers to gain system privileges on a compromised system (Microsoft Security Response Center, 2023). This vulnerability is associated with CWE-822, which refers to Untrusted Pointer Dereference (CVE Program, 2024). A CVSS score of 7.8 and a high severity level has been assigned to this vulnerability.

QBot has been known to exploit this zero-day vulnerability for privilege escalation (Bagwe, 2024). By exploiting this flaw, QBot can gain elevated access rights, enabling it to execute high-privilege operations, install additional malware, and maintain control over the system. This exploitation method enhances QBot's ability to perform its malicious activities more effectively.

6.4 CVE-2024-30040

According to Microsoft Security Response Center (2024), CVE-2024-30040 is a vulnerability in the Windows MSHTML platform that bypasses security features protecting against vulnerable COM/OLE controls in Microsoft 365 and Office. Exploiting this flaw requires convincing a user to load a malicious document, often through email or messaging (Microsoft Security Response Center, 2024). Once the document is interacted with, an attacker can execute arbitrary code with the user's privileges, leading to potential system compromise. This vulnerability is associated with CWE-20, which refers to Improper Input Validation (CVE Program, 2024). A CVSS score of 8.8 and a high severity level has been assigned to this vulnerability.

QBot is able to exploit this vulnerability by leveraging a security feature bypass in the Windows MSHTML platform (Alder, 2024). To exploit this flaw, QBot tricks users into loading a malicious file, typically delivered through email or messaging platforms. While users do not need to click or open the file, they must interact with it in some manner. Successfully exploiting this vulnerability allows QBot to execute arbitrary code on the user's system, potentially leading to remote code execution and further compromise.

6.5 CVE-2024-30051

CVE-2024-30051 is a privilege escalation vulnerability in the Windows Desktop Window Manager (DWM) core library, caused by a heap-based buffer overflow (Microsoft Security Response Center, 2024). Successfully exploiting this flaw allows attackers to gain system-level privileges on affected systems. This vulnerability is associated with CWE-122, which refers to Heap-based Buffer Overflow (CVE Program, 2024). A CVSS score of 7.8 and a high severity level has been assigned to this vulnerability.

QBot exploits this vulnerability by leveraging the privilege escalation flaw in the Windows DWM core library caused by a heap-based buffer overflow (Gatlan, 2024). By exploiting this vulnerability, QBot can gain system-level privileges on the compromised system. This elevated access allows QBot to execute high-privilege operations, including installing additional malware and manipulating critical system components, thus enhancing its control and persistence on the affected system.

7. Tactics, Techniques, and Procedures of QBot Attacks

7.1 Intrusion tactics

Qakbot often enters a target system as a second-stage exploit, following initial breaches that occur through various techniques, such as malspam, email phishing with trojanized documents, or exploiting public-facing vulnerabilities (BlackBerry, n.d.). Microsoft Threat Intelligence (2021) mentions that QBot is spread through emails containing malicious links, attachments, or embedded images.

Maharjan (2022) states that QBot's intrusion tactics have evolved to leverage highly targeted malspam email campaigns. These campaigns often involve email thread hijacking, making the messages appear more legitimate and increasing the likelihood of user interaction.

The email messages use brevity, fake-reply techniques, and impersonation of trusted entities to increase the likelihood of target users interacting with the malicious content (Microsoft Threat Intelligence, 2021). The attackers typically include HTML attachments that, when opened, lead to the download of malicious files.

7.2 Attack techniques

Initially, QBot relied heavily on malicious Excel attachments containing macros that automate malicious commands when opened by users (Small, 2022). As security measures evolved, QBot operators shifted to using HTML attachments that facilitate stealthy downloads of additional malicious files. These files, often ZIPs containing ISOs, LNKs, and DLLs, are designed to circumvent newer security protections like the Mark of the Web (MotW) protections and execute QBot. Small (2022) explains that the malware also exploits vulnerabilities like the "Follina" exploit, embedded in LNK files, to run QBot DLLs. Even with patches in place for the “Follina” vulnerability, attackers have adapted by using native binaries through LOLBins to maintain persistence (Maharjan, 2022). In post-exploitation stage, QBot uses techniques such as Regsvr32 for proxying execution of malicious code, process injection, and scheduled tasks to maintain control and avoid detection.

QBot is designed to ensure persistence and avoid detection (Maharjan, 2022). Before an infected device shuts down, QBot activates its persistence mechanism, deleting all traces of itself upon system restart or awakening from sleep. This rapid engagement makes it difficult for security software to identify and remove the threat. Additionally, QBot employs legitimate process injection methods to obfuscate its presence, making it difficult for security software to detect (Maharjan, 2022). Once inside a network, QBot uses its lateral movement capabilities to spread to other machines. The malware also incorporates a Universal Plug-and-Play (UPnP) module, which allows it to transform infected hosts into intermediate command and control (C2) servers, furthering its reach within the network.

Another technique that QBot utilized to maintain persistence and evade detection involves evaluating the local system environment to avoid execution in virtualized environments or where specific security products or registry keys are detected (BlackBerry, n.d.). This strategy helps QBot conceal its payload from security researchers. Additionally, QBot injects itself into legitimate application processes, making it harder to detect. It also uses unauthorized run keys in the Windows Registry to auto-start itself upon user login, ensuring persistence. QBot is frequently updated to obscure its known indicators of compromise (IOCs), further complicating detection efforts (BlackBerry, n.d.).

Furthermore, QBot is highly modular, downloading and installing additional components to extend its capabilities. These include a Password Grabber Module for credential theft, an hVNC Plugin for remote control of the infected device, a Cookie Grabber Module to steal browser cookies, a Web-Inject Module to inject malicious JavaScript into financial websites, and an Email Collection Module to extract emails from Outlook clients for further phishing campaigns.

A complete list of techniques used by QBot are summarized in Table 4 below. This information is obtained from the MITRE ATT&CK framework.

Table 4: Techniques used by QBot *(Millington & Danilevich, 2023)*

No.	Purpose	Techniques
1	Initial access	a) **Phishing**: Spearphishing Link, Spearphishing Attachment b) **Replication Through Removable Media** c) **User Execution**: Malicious Link, Malicious File d) **Exploit Vulnerabilities**: Exploitation of Remote Services
2	Execution	a) **Command and Scripting Interpreter**: PowerShell, Windows Command Shell, JavaScript, Visual Basic b) **System Binary Proxy Execution**: Regsvr32, Msiexec, Rundll32 c) **Create or Modify System Process**: Windows Service d) **Hijack Execution Flow**: DLL Side-Loading
3	Persistence	a) **Boot or Logon Autostart Execution**: Registry Run Keys / Startup Folder b) **Modify Registry**
4	Privilege escalation	a) **Brute Force** b) **Process Injection**: Process Hollowing, Process Injection
5	Defense evasion	a) **Obfuscated Files or Information**: Binary Padding, Fileless Storage, HTML Smuggling, Command Obfuscation, Indicator Removal from Tools, Software Packing b) **Impair Defenses**: Disable or Modify Tools

		c) **Masquerading**: Masquerade File Type d) **Hide Artifacts**: Hidden Files and Directories, File Deletion e) **Subvert Trust Controls**: Code Signing, Mark-of-the-Web Bypass f) **Virtualization/Sandbox Evasion**: System Checks, Time Based Evasion
6	Credential access	a) **Credentials from Password Stores**: Credentials from Web Browsers b) **Input Capture**: Keylogging
7	Discovery	a) **Application Window Discovery** b) **Browser Session Hijacking** c) **Domain Trust Discovery** d) **File and Directory Discovery** e) **Network Share Discovery** f) **Remote System Discovery** g) **Peripheral Device Discovery** h) **Permission Groups Discovery:** Local Groups i) **Process Discovery** j) **System Information Discovery** k) **System Network Configuration Discovery:** Internet Connection Discovery l) **System Network Connections Discovery** m) **System Owner/User Discovery** n) **System Time Discovery** o) **Software Discovery:** Security Software Discovery
8	Lateral movement	a) **Protocol Tunnelling** b) **Proxy**: External Proxy c) **Ingress Tool Transfer**
9	Collection	a) **Email Collection**: Local Email Collection b) **Data from Local System** c) **Data Staged**: Local Data Staging, d) **Steal Web Session Cookie**

10	Exfiltration	a) **Data Encoding**: Standard Encoding b) **Dynamic Resolution**: Domain Generation Algorithms c) **Encrypted Channel**: Symmetric Cryptography d) **Exfiltration Over C2 Channel**
11	Command and control	a) **Application Layer Protocol**: Web Protocols b) **Non-Application Layer Protocol** c) **Obfuscated Files or Information** d) **Deobfuscate/Decode Files or Information**

7.3 Attack procedures

QBot's attack procedures demonstrate a high level of adaptability and sophistication. After gaining initial access through phishing emails, the malware employs multiple layers of execution to ensure successful infection. An overview of a typical QBot attack can be observed in the flowchart under Figure 1 below. This flowchart provides a clear view of how QBot infects a system and maintains control, using a combination of social engineering, legitimate Windows tools, and advanced evasion techniques. Explanations for each component of the flowchart are provided under Table 5 below.

Figure 1: Process of a QBot attack *(Maharjan, 2022)*

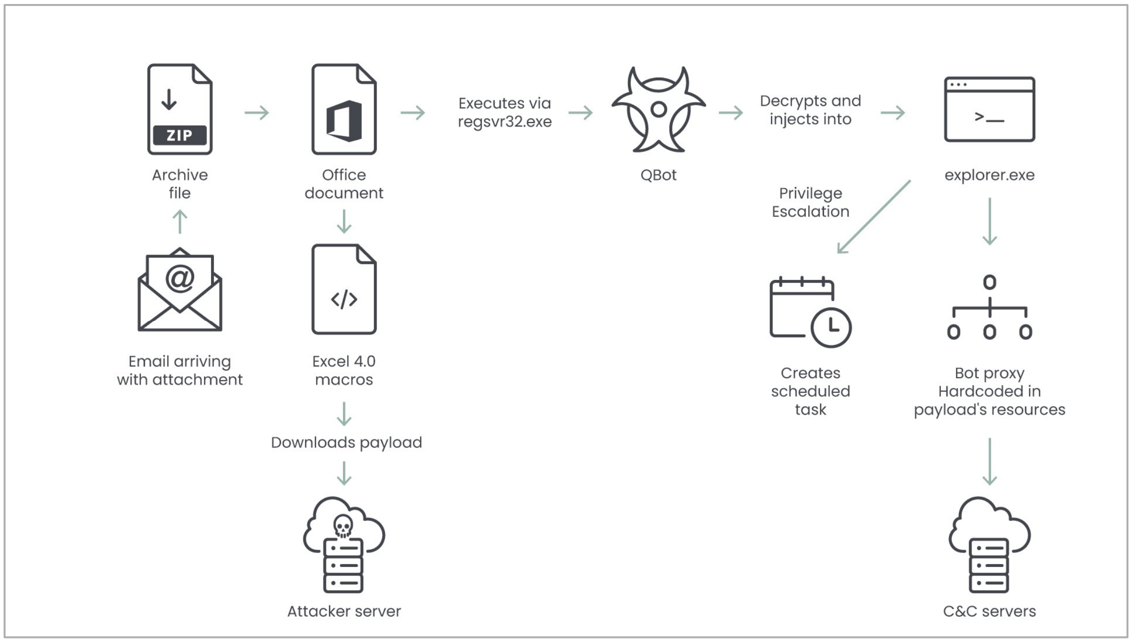
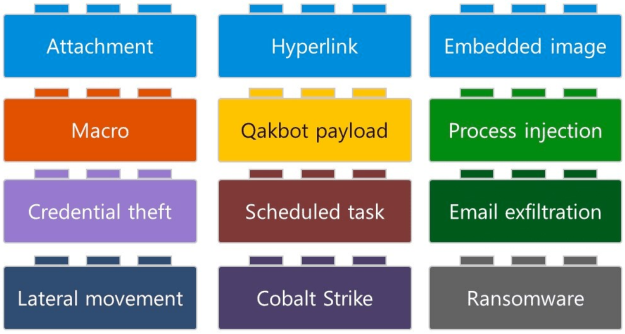


Table 5: Explanations for the QBot attack process flowchart

Component	Explanation
Email arriving with attachment	The process begins with the victim receiving an email that contains a malicious attachment. This is typically a phishing email designed to trick the recipient into opening the attachment.
Archive file	The attachment is an archive file, such as a ZIP file, which contains the malicious payload. The victim is convinced to download and open this file.
Office document	Inside the archive, there is an Office document, often a Microsoft Word or Excel file. This document typically contains malicious macros that are designed to execute when the document is opened.
Excel 4.0 macros	The Office document utilizes Excel 4.0 macros, an older type of macro that can still be used in modern Excel files. These macros are used to download the QBot payload from the attacker's server.
Downloads payload	The macro initiates a connection to an attacker-controlled server, from which it downloads the QBot malware.
Executes via regsvr32.exe	The downloaded QBot malware is then executed using regsvr32.exe, a legitimate Windows utility that is often abused by malware to run scripts and executables.
QBot	Once executed, QBot begins its malicious operations on the infected system.
Decrypts and injects into explorer.exe	QBot decrypts its payload and injects itself into a legitimate process, such as explorer.exe, to evade detection and maintain persistence.
Privilege escalation	QBot may attempt to escalate its privileges on the system to gain higher-level access, allowing it to perform more dangerous actions.
Creates scheduled task	To ensure it runs automatically even after a system reboot, QBot creates a scheduled task that triggers its execution.
Bot proxy hardcoded in payload's resources	The QBot malware includes a hardcoded bot proxy within its resources, which it uses to communicate with the command and control (C&C) servers.
C&C servers	Finally, QBot connects to the C&C servers to receive instructions, exfiltrate stolen data, and possibly download additional malware or updates.

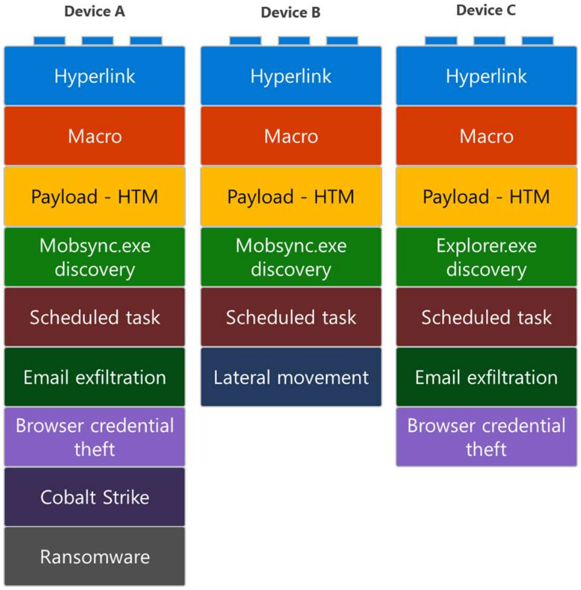
Based on Microsoft's analysis, a QBot-related incident can be deconstructed into distinct "building blocks" that assist security analysts in identifying and responding to QBot campaigns (Microsoft Threat Intelligence, 2021). Figure 2 below illustrates these building blocks. Their observations suggest that each QBot attack chain contains only one block of each color, with the first row and the macro block representing the email mechanism used to deliver QBot.

Figure 2: Building blocks of QBot attack chain *(Microsoft Threat Intelligence, 2021)*



A sample of QBot campaign constructed from the building blocks in Figure 2 are illustrated in Figure 3, where Devices A and C were identified as sending emails outside the organization, except Device B.

Figure 3: Sample of devices infected by a single QBot campaign *(Microsoft Threat Intelligence, 2021)*



8. Cybersecurity Strategies for Defending Against QBot/LOLBins Attacks

8.1 Threat Modeling for QBot

Threat modeling is an essential part of securing systems, applications, and networks, particularly when dealing with sophisticated threats like QBot and LOLBINS. The threat modeling process discussed below is based on a framework from OWASP community, and it provides a structured approach to identify, evaluate, and mitigate security threats effectively.

The process involves four main steps: Scoping the work, determining threats, determining countermeasures and mitigation, and assessing the work (Conklin, n.d.). Each step is designed to ensure that all potential vulnerabilities are identified and addressed, minimizing the risk posed by advanced persistent threats like QakBot.

8.1.1 Define scope

The first step involves understanding and defining the environment and components at risk from QBot/LOLBins attacks. This begins by drawing Data Flow Diagrams (DFDs) that visualize the flow of data within the organization's systems, focusing on entry points where QBot could infiltrate, such as email systems and endpoints. The diagrams also highlight paths where LOLBins could be exploited to execute QBot, such as through the use of regsvr32.exe or PowerShell. A sample diagram is illustrated in Figure 1 under Section 7.3. Identifying privilege or trust boundaries within the system, where different levels of access rights are granted, is crucial to understanding how these attacks could escalate privileges or gain unauthorized access.

Next, the process involves identifying specific entry points where threats might manifest. For example, phishing emails are a common entry point for QBot, where the malware is delivered through malicious attachments or links. LOLBins execution points, such as BITSAdmin or MSHTA, serve as other critical entry points where QBot might exploit legitimate system processes to run malicious scripts.

Identifying critical assets is also part of this step, including email systems vulnerable to phishing attacks, endpoints susceptible to LOLBins exploitation, and user credentials that QBot could steal for lateral movement across the network. Trust levels within the system are also analyzed, distinguishing between regular user access, which might unknowingly execute QBot, and administrative access, which QBot might target to escalate its privileges.

8.1.2 Identification of threats

The next step focuses on identifying and categorizing the threats associated with QBot/LOLBins attacks. This is achieved using threat categorization methodologies like STRIDE, which helps to systematically identify potential threats across various categories, including spoofing, tampering, repudiation, information disclosure, denial of service, and elevation of privilege (Conklin, n.d.). Based on the STRIDE methodology, the potential threats of QBot include:

a. **Spoofing**: QBot can spoof legitimate processes to avoid detection.
b. **Tampering**: Attackers might tamper with system files or settings using QBot to maintain persistence.
c. **Repudiation**: QBot's use of LOLBins can make it challenging to track the origin of the attack.
d. **Information Disclosure**: QBot could steal sensitive data, including user credentials.
e. **Denial of Service**: QBot might be used to disrupt services within the network.
f. **Elevation of Privilege**: QBot can exploit LOLBins to gain higher privileges within a system.

Additionally, phishing emails are recognized as the primary vector for QBot delivery, exploiting vulnerabilities in email filtering and user awareness (Maharjan, 2022). LOLBins exploitation involves the use of legitimate Windows utilities such as PowerShell, regsvr32.exe, and BITSAdmin, allowing stealthy execution and privilege escalation. Weak endpoint security, including outdated antivirus software and misconfigured security policies, presents another threat. Furthermore, credential theft by QBot can lead to lateral movement across the network, compounding the risk. Mapping these threats using DFDs provides a clear view of how QBot could exploit specific data sources, processes, and interactions within the system. For example, a phishing email could lead to the execution of a malicious macro, which in turn triggers QBot via regsvr32.exe.

8.1.3 Countermeasure and mitigation

After identifying the threats, the next step is to develop countermeasures and mitigation strategies to address them. Risk mitigation strategies should be prioritized based on the likelihood and impact of each threat. For example, high-impact threats like phishing attacks leading to QBot execution and lateral movement using stolen credentials should be prioritized for mitigation.

To counteract and mitigate these threats, several strategies can be implemented. Three types of controls are proposed:

a. Preventative controls:
   - **Email Security**: Deploy advanced email filtering and phishing detection solutions to block malicious emails.
   - **Endpoint Protection**: Implement Endpoint Detection and Response (EDR) solutions to monitor and respond to LOLBins usage and unusual process behaviours.
   - **User Training**: Conduct training programs to educate users on recognizing phishing emails and handling unknown attachments.

b. Detective Controls:
   - **Monitor LOLBins Usage**: Set up monitoring systems to identify abnormal usage patterns of PowerShell, regsvr32.exe, and BITSAdmin.
   - **Network Traffic Analysis**: Employ network monitoring tools to detect suspicious data exfiltration and lateral movement activities.

c. Responsive Controls:
   - **Incident Response**: Develop and refine incident response plans specifically for QBot-related scenarios, focusing on containment and eradication strategies.

8.1.5 Plan assessment

The final step in the OWASP threat modeling process involves reviewing and validating the work done. This includes verifying that all DFDs, threat lists, and countermeasure controls have been documented and are accessible. It is also essential to confirm that all identified threats have corresponding countermeasures and that no critical threats have been overlooked.

To validate the threat model, organizations can conduct red teaming exercises to simulate QBot/LOLBins attacks and evaluate the countermeasures' effectiveness (Cisco, n.d.). Regular audits of the threat model should also be performed to account for any changes in the environment or the emergence of new threats, ensuring that the organization remains resilient against evolving threats like QBot and LOLBins.

8.2 Operational Security (OPSEC) Plan

Operational Security (OPSEC) is a critical discipline in cybersecurity that focuses on safeguarding sensitive information and minimizing vulnerabilities to prevent malicious activities (Proofpoint, n.d.). According to Proofpoint (n.d.), the process of OPSEC can be divided into five main steps, where each step is crucial to identify vulnerabilities and develop strategies in protecting information effectively. Each step will be discussed in detail under the subsections below.

8.2.1 Identify critical information

The first step in developing an OPSEC plan for a QBot/LOLBINS attack is to identify the critical information within the organization. Critical information includes any data, systems, or processes that, if compromised, could significantly harm the organization. For a QBot/LOLBINS attack, critical information might include:

a. **User Credentials:** QBot is known for stealing credentials, which can be used for lateral movement within a network. Protecting login information, especially those with administrative privileges, is crucial.

b. **Endpoint Devices:** Devices such as workstations and servers that can be exploited by LOLBins (Living Off the Land Binaries) to execute QBot.

c. **Email Systems:** As phishing emails are a common initial vector for QBot infections, email systems must be considered critical.

d. **Network Infrastructure:** The communication channels within the network that QBot might exploit for command and control (C2) purposes.

e. **Security Software Configurations:** The settings and deployment of security tools that, if identified and disabled by QBot, could leave the organization vulnerable.

Recognizing these pieces of critical information is vital as it allows the organization to focus its protective measures on the most important assets.

8.2.2 Analyze threats

In the second step, the organization needs to analyze the threats that could target the identified critical information. Threat analysis involves understanding who might want to attack the organization and how they might do it. For QBot/LOLBINS, the following threats are particularly relevant:

a. **External Hackers**: Cybercriminals aiming to deploy QBot through phishing campaigns, exploiting LOLBins to bypass traditional security measures.
b. **Insider Threats**: Employees or contractors who might unintentionally introduce QBot by clicking on phishing links or running malicious scripts embedded in legitimate processes.
c. **Advanced Persistent Threats (APTs)**: State-sponsored actors or organized cybercrime groups using sophisticated methods to deploy QBot and evade detection.
d. **Malware Developers**: Those continuously updating QBot to evade detection by security systems and exploit new vulnerabilities within LOLBins.

Understanding these threats helps in tailoring defenses against specific adversaries that are most likely to target the organization.

8.2.3 Assess vulnerabilities

Assessing vulnerabilities involves a detailed examination of the organization's current defenses to identify where they may be lacking. This step scrutinizes both technological and human factors to uncover potential weak points that could be exploited in a QBot/LOLBINS attack. Vulnerabilities to consider include:

a. **Outdated Software**: Systems or applications that have not been updated and are vulnerable to exploitation by LOLBins or QBot.
b. **Weak Email Security**: Insufficient filtering or detection mechanisms in email systems that allow phishing emails to reach end-users.
c. **Lack of Endpoint Protection**: Insufficient monitoring of endpoint devices that might allow unauthorized execution of LOLBins.
d. **Unsecured Network Configurations**: Network settings that do not adequately segment or isolate critical systems, allowing QBot to spread laterally.
e. **Insufficient User Training**: Employees who are not trained to recognize phishing attempts or understand the risks associated with executing unverified files.

This step is crucial for identifying the gaps in the organization's defenses that need to be addressed to prevent or mitigate QBot/LOLBINS attacks.

8.2.4 Evaluate risk

Once vulnerabilities are identified, the next step is to evaluate the risk they pose to the organization. Risk evaluation involves analyzing both the likelihood of a vulnerability being exploited and the potential impact it would have. For QBot/LOLBINS attacks, consider the following risk factors:

a. **Likelihood of Phishing Success**: The probability that a phishing email will bypass filters and deceive an employee into executing QBot.
b. **Impact of Credential Theft**: The potential consequences if QBot successfully harvests credentials, such as unauthorized access to sensitive systems or data breaches.
c. **Severity of Lateral Movement**: The impact on the organization if QBot uses LOLBins to move laterally within the network, compromising multiple systems.
d. **Effectiveness of Current Defenses**: How well the organization's existing security measures can detect and respond to a QBot/LOLBINS attack.

Risks are then prioritized based on their likelihood and potential impact, guiding the organization on where to focus its resources for the greatest effect.

8.2.5 Apply Countermeasures

The final step is to apply countermeasures to address the identified risks. Countermeasures are specific actions or tools implemented to reduce the likelihood or impact of a QBot/LOLBINS attack. Examples of countermeasures include:

a. **Email Filtering and Security**: Implement advanced email security solutions that detect and block phishing attempts, reducing the chance of QBot delivery (Microsoft Threat Intelligence, 2021).
b. **Endpoint Detection and Response (EDR)**: Deploy EDR solutions that monitor and detect the use of LOLBins, providing alerts or automatically responding to suspicious activity (Microsoft Threat Intelligence, 2021).
c. **Software Updates and Patching**: Ensure all systems and applications are up-to-date, particularly those associated with LOLBins like PowerShell or regsvr32.exe (Maharjan, 2022).
d. **Network Segmentation**: Isolate critical systems to prevent QBot from moving laterally if it does manage to infiltrate the network (Darktrace, n.d.).
e. **User Training Programs**: Regularly educate employees about the dangers of phishing, the importance of verifying email sources, and how to report suspicious activity (Small, 2022).

In summary, applying these countermeasures significantly reduces the organization's exposure to QBot/LOLBINS attacks, enhancing its overall security posture.

8.3 General Security Plan

This general security plan is based on the general security concepts provided in the Red Hat Documentation website (Red Hat Documentation, n.d.). There are a total of 6 concepts retrieved from the documentation, and each concept serves as a component in this plan.

8.3.1 Authentication

The primary objective of authentication is to ensure that only authorized individuals can access systems and sensitive data, thereby preventing unauthorized access by QBot or LOLBins. To achieve this, it is essential to implement Multi-Factor Authentication (MFA), which adds an extra layer of security beyond passwords. MFA should combine something the user knows, like a password, something they have, like a smartphone app for OTPs, and something they are, which refers to biometrics.

Additionally, enforcing strong password policies is critical. Passwords should be at least 12 characters long, combining upper and lower case letters, numbers, and special characters, and should be updated regularly. Implementing account lockout mechanisms after a set number of failed login attempts can protect against brute force attacks. Biometric authentication methods, such as fingerprint or facial recognition, should be used wherever feasible, as it provides an additional layer of security.

For instance, MFA is implemented on all employee accounts to require both a password and a code from an authentication app.

8.3.2 Authorization

Authorization ensures that users only have access to resources necessary for their roles, minimizing the risk of exploitation by QBot or LOLBins. Role-Based Access Control (RBAC) should be employed to assign permissions based on user roles, limiting access to information and systems required for job functions. Applying the least privilege principle ensures that users have only the access necessary to perform their tasks, with regular reviews and updates to permissions. Segregation of duties is another important strategy, ensuring that critical tasks require more than one person to reduce the risk of fraud or unauthorized changes.

For instance, an IT administrator has access to server management tools but not to financial systems. A finance employee has access to accounting software but not to HR systems.

8.3.3 Encryption

Encryption is crucial for protecting sensitive data from unauthorized access and exfiltration. Encrypting sensitive data at rest on servers, databases, and backup tapes using strong algorithms like AES-256 helps prevent QBot from accessing or exfiltrating critical information. Implementing end-to-end encryption ensures that data remains secure during transfer between users or systems, protecting against interception and eavesdropping. Additionally, full disk encryption should be applied to all company devices, including laptops and mobile devices, to safeguard data in the event of loss or theft.

For instance, customer data stored in databases is encrypted with AES-256 and end-to-end encryption is implemented in internal communication tools.

8.3.4 SSL/TLS and Certificates

To secure data transmission and prevent eavesdropping or tampering, SSL/TLS should be enforced for all web-based applications and services. It is essential to use the latest version of TLS to avoid vulnerabilities that QBot could exploit. A robust certificate management system should be implemented to regularly update, renew, and revoke SSL/TLS certificates. Extended Validation (EV) certificates can be used where appropriate to enhance security and user trust. Additionally, certificate pinning in mobile apps and sensitive web applications helps prevent man-in-the-middle attacks by ensuring that only trusted certificates are accepted.

For instance, HTTPS is enforced across all public-facing web applications and SSL certificates are regularly renewed for internal and external systems.

8.3.5 Single Sign-On (SSO)

SSO simplifies user access management while maintaining robust security. By implementing a centralized identity management system such as Okta, all applications requiring authentication can be integrated into a single login system. Secure tokens like SAML or OAuth should be used for authentication across applications, ensuring that user credentials are not exposed. Integrating SSO with MFA enhances security, reducing the need for multiple logins while providing an additional layer of protection against QBot attacks.

For instance, using Okta as an SSO provider to enable single login for email, CRM, and other applications, and implementing MFA with SSO for added security.

8.3.6 Lightweight Directory Access Protocol (LDAP)

LDAP is used to centralize and streamline authentication and authorization processes. Integrating LDAP with existing systems provides a centralized directory service for managing user information, simplifying authentication and access control. Access rights, group memberships, and permissions should be managed via LDAP directories to ensure appropriate access levels and reduce the risk of QBot exploiting excessive permissions. To secure LDAP communications, it is crucial to encrypt connections using SSL/TLS and to monitor LDAP logs for any suspicious activity that could indicate an attempt to exploit vulnerabilities.

For instance, Microsoft Active Directory is implemented using LDAP for user account and permission management and LDAP queries are encrypted with TLS to secure communication between client applications and the directory service.

8.4 Technical Security Plan

This technical security plan integrates preventative, detective, and corrective measures to provide comprehensive protection against QBot and LOLBins (Swanagan, 2023). By implementing these controls, organizations can enhance their ability to prevent, detect, and respond to potential security threats effectively.

8.4.1 Preventative

Preventative measures are crucial in proactively defending against QBot and LOLBins by stopping these threats before they can infiltrate or cause damage. By employing controls designed to block, filter, or limit potential attack vectors, organizations can mitigate risks and reduce the likelihood of successful breaches. Effective preventative measures include deploying firewalls, intrusion prevention systems, multi-factor authentication, and antivirus solutions (Swanagan, 2023). These controls work together to fortify the organization's defenses and create barriers that are difficult for attackers to breach. Each measure is described in further detail as follows:

A. **Firewall**: The deployment of firewalls is critical in controlling and filtering network traffic to prevent unauthorized access and malicious activity associated with QBot and LOLBins. Firewalls should be configured to block incoming and outgoing traffic from known malicious IP addresses and to enforce strict rules on what traffic is allowed based on the organization's security policies. Regular updates to firewall rules and configurations are necessary to adapt to evolving threats and to protect against new attack vectors.

B. **Intrusion Prevention System (IPS)**: An IPS should be implemented to detect and prevent malicious activities and intrusions before they can exploit vulnerabilities. The IPS will monitor network traffic for signs of QBot-related behaviours, such as unusual patterns or known attack signatures associated with LOLBins. It is essential to regularly update the IPS signatures and rules to keep pace with new QBot variants and LOLBins techniques.

C. **Multi-Factor Authentication (MFA)**: Enforcing MFA adds an additional layer of security to user accounts and systems, making it significantly harder for attackers to gain unauthorized access even if passwords are compromised. MFA should be deployed across all critical systems, including remote access points and administrative interfaces, to mitigate the risk of QBot and LOLBins gaining access through stolen credentials.

D. **Antivirus**: Antivirus software is a fundamental preventative measure against QBot and LOLBins. It should be installed and maintained on all endpoints and servers, with regular updates to ensure it can detect and block the latest threats. Real-time scanning and periodic system scans are necessary to identify and neutralize any malicious software before it can execute or spread within the network.

8.4.2 Detective

Detective measures play a vital role in identifying and responding to threats that have managed to bypass initial defenses. By continuously monitoring and analyzing system activities, organizations can detect suspicious behaviours and potential intrusions early. Key detective measures for addressing QBot and LOLBins include intrusion detection systems and honeypots (Swanagan, 2023). These tools provide valuable insights into attack patterns and allow for timely responses to emerging threats, enhancing the organization's ability to identify and address security incidents effectively. Each measure is described in further detail as follows:

A. **Intrusion Detection System (IDS)**: An IDS is crucial for identifying and alerting on suspicious activities and potential threats. It should be configured to detect anomalous behavior and known attack patterns related to QBot and LOLBins. The IDS will help in recognizing attempts to exploit vulnerabilities or unauthorized access attempts, providing valuable insights for incident response.

B. **Honeypots**: Deploying honeypots can be an effective technique to detect and analyze QBot and LOLBins activities. By creating decoy systems that mimic real vulnerabilities, honeypots can attract and capture interactions from these threats, providing detailed information about their behavior and methods. This data can be used to improve defensive strategies and to better understand the tactics used by QBot and LOLBins.

8.4.3 Corrective

Corrective measures are essential for responding to and mitigating the impact of security incidents that have already occurred. These actions focus on addressing and rectifying issues caused by QBot and LOLBins, ensuring that systems are restored to a secure state and vulnerabilities are remediated. Key corrective measures include vulnerability patching, rebooting systems, and quarantining viruses (Swanagan, 2023). Implementing these steps helps organizations recover from incidents, strengthen their defenses, and prevent future occurrences of similar threats. Each measure is described in further detail as follows:

A. **Vulnerability Patching**: Regular and timely vulnerability patching is essential to protect against exploits used by QBot and LOLBins. It involves applying patches and updates to software, operating systems, and applications to fix known vulnerabilities. A systematic approach to vulnerability management should be established to ensure that patches are evaluated, tested, and deployed promptly.

B. **Reboot a System**: Rebooting systems can be a corrective measure to resolve issues caused by QBot or LOLBins infections. It is particularly useful when a system needs to be reset to a clean state after a malware removal process. However, it should be part of a broader incident response strategy, ensuring that all remnants of malware are addressed before rebooting.

C. **Quarantine a Virus**: When an antivirus or endpoint protection solution detects QBot or LOLBins, the infected files should be quarantined to prevent them from causing further harm. Quarantine involves isolating the malicious files so they cannot execute or spread while allowing for further analysis and remediation. This process helps in preventing the immediate impact of the infection and facilitates subsequent removal or clean-up actions.

9. Conclusion

The rise of Cybercrime-as-a-Service (CaaS) and the sophisticated use of Living Off the Land Binaries (LOLBins) have introduced new challenges for organizations striving to protect their digital assets. This report has provided an in-depth exploration of these threats, with a particular focus on QBot (Qakbot), a pervasive malware that leverages vulnerabilities and LOLBins to execute its malicious activities.

The detailed overview of CaaS has highlighted its disruptive impact on the cyber threat landscape, revealing how these criminal services have democratized access to advanced attack techniques and tools. The report has examined LOLBins and their exploitation mechanisms, emphasizing the advantages they offer attackers in evading detection and enhancing persistence within compromised environments.

A comprehensive analysis of QBot's background and capabilities has been presented, alongside an examination of the vulnerabilities that facilitate its attacks. By understanding the tactics, techniques, and procedures (TTPs) employed by QBot, organizations can better anticipate and defend against its malicious operations.

The cybersecurity strategies outlined in this report provide a robust framework for defending against QBot and LOLBins. The threat modeling approach, operational security (OPSEC) plan, general security plan, and technical security plan collectively offer a structured methodology for identifying, assessing, and mitigating risks associated with these threats.

In conclusion, addressing the challenges posed by QBot and LOLBins requires a comprehensive and proactive approach to cybersecurity. By implementing the strategies and controls detailed in this report, organizations can enhance their ability to prevent, detect, and respond to these sophisticated threats, ultimately strengthening their overall security posture and safeguarding their critical assets against emerging cyber threats.

10. References

Alder, S. (2024, May 16). *Microsoft Patches Zero-Day Vulnerability Exploited to Deliver QakBot and Other Malware*. Retrieved from The HIPAA Journal: https://www.hipaajournal.com/windows-dwm-zero-day-vulnerability-qakbot-malware/

Bagwe, M. (2024, May 15). *Microsoft Addresses Zero-Day Vulnerability Exploited by QakBot Malware*. Retrieved from The Cyber Express: https://thecyberexpress.com/zero-day-exploited-by-qakbot-malware/

BlackBerry. (n.d.). *Qakbot Malware*. Retrieved from BlackBerry: https://www.blackberry.com/us/en/solutions/endpoint-security/ransomware-protection/qakbot

Chebac, A. (2023, September 13). *What Is Cybercrime-as-a-Service (CaaS)?* Retrieved from Heimdal: https://heimdalsecurity.com/blog/what-is-cybercrime-as-a-service-caas/

Cisco. (n.d.). *What Is Threat Modeling?* Retrieved from Cisco: https://www.cisco.com/c/en/us/products/security/what-is-threat-modeling.html#~how-threat-modeling-works

Conklin, L. (n.d.). *Threat Modeling Process*. Retrieved from OWASP: https://owasp.org/www-community/Threat_Modeling_Process

CVE Program. (2024, August 4). *CVE-2021-34527*. Retrieved from CVE Website: https://www.cve.org/CVERecord?id=CVE-2021-34527

CVE Program. (2024, August 3). *CVE-2022-30190*. Retrieved from CVE Website: https://www.cve.org/CVERecord?id=CVE-2022-30190

CVE Program. (2024, August 2). *CVE-2023-36033*. Retrieved from CVE Website: https://www.cve.org/CVERecord?id=CVE-2023-36033

CVE Program. (2024, August 2). *CVE-2024-30040*. Retrieved from CVE Website: https://www.cve.org/CVERecord?id=CVE-2024-30040

CVE Program. (2024, August 2). *CVE-2024-30051*. Retrieved from CVE Website: https://www.cve.org/CVERecord?id=CVE-2024-30051

Darktrace. (n.d.). *What is Qakbot?* Retrieved from Darktrace: https://darktrace.com/cyber-ai-glossary/qakbot

Field Effect. (2023, April 19). *The rise of cybercrime-as-a-service*. Retrieved from Field Effect: https://fieldeffect.com/blog/cybercrime-as-a-service

Gatlan, S. (2024, May 14). *Microsoft fixes Windows zero-day exploited in QakBot malware attacks*. Retrieved from BleepingComputer: https://www.bleepingcomputer.com/news/microsoft/microsoft-fixes-windows-zero-day-exploited-in-qakbot-malware-attacks/

Goss, A. (2024, June 25). *LOLBins (Complete Guide to Living Off the Land Binaries)*. Retrieved from StationX: https://www.stationx.net/lolbins-living-off-the-land-binaries/

Maharjan, N. (2022, September 21). *What the Quack: Hunt for the QBOT with Logpoint*. Retrieved from Logpoint: https://www.logpoint.com/en/blog/what-the-quack-hunt-for-the-qbot-with-logpoint/

Microsoft Security Response Center. (2023, November 14). *CVE 2023 36033*. Retrieved from Microsoft Security Response Center: https://msrc.microsoft.com/update-guide/vulnerability/CVE-2023-36033

Microsoft Security Response Center. (2024, May 14). *CVE-2024-30040*. Retrieved from Microsoft Security Response Center: https://msrc.microsoft.com/update-guide/vulnerability/CVE-2024-30040

Microsoft Security Response Center. (2024, May 14). *CVE-2024-30051*. Retrieved from Microsoft Security Response Center: https://msrc.microsoft.com/update-guide/vulnerability/CVE-2024-30051

Microsoft Threat Intelligence. (2021, December 9). *A closer look at Qakbot's latest building blocks (and how to knock them down)*. Retrieved from Microsoft: https://www.microsoft.com/en-us/security/blog/2021/12/09/a-closer-look-at-qakbots-latest-building-blocks-and-how-to-knock-them-down/

Millington, E., & Danilevich, I. (2023, December 5). *QakBot*. Retrieved from MITRE ATT&CK®: https://attack.mitre.org/software/S0650/

Mohanlal, N., & Barlow, E. (2021, August). *Security 101: What are LOLBins and How Can They be Used Maliciously?* Retrieved from SecurityHQ: https://www.securityhq.com/blog/security-101-lolbins-malware-exploitation/

Moore, T. (2023, October 12). *Protect Your Organization from Cybercrime-as-a-Service Attacks*. Retrieved from Thales: https://cpl.thalesgroup.com/blog/encryption/cybercrime-as-a-service-caas-explaned

Proofpoint. (n.d.). *OPSEC (Operational Security)*. Retrieved from Proofpoint: https://www.proofpoint.com/us/threat-reference/operational-security-opsec

Red Hat Documentation. (n.d.). *Chapter 1. Overview of General Security Concepts*. Retrieved from Red Hat Documentation: https://docs.redhat.com/en/documentation/red_hat_jboss_enterprise_application_platform/6.4/html/security_architecture/overview-of-general-security-concepts

Rumiantseva, O. (2023, July 18). *What Are LOLBins?* Retrieved from SOC Prime: https://socprime.com/blog/what-are-lolbins/#How_Can_LOLBins_be_Used_for_Cyber_Attacks

Small, S. (2022, November 30). *Identifying and Defending Against QakBot's Evolving TTPs*. Retrieved from Tidal Cyber: https://www.tidalcyber.com/blog/identifying-and-defending-against-qakbots-evolving-ttps

Swanagan, M. (2023, December 7). *The 3 Types Of Security Controls (Expert Explains)*. Retrieved from PurpleSec: https://purplesec.us/learn/security-controls/

Trabelsi, A. (2024, April 8). *Unveiling LOLBins: A data-driven exploration*. Retrieved from Cyberdefense: https://www.orangecyberdefense.com/global/blog/cybersecurity/unveiling-lolbins-a-data-driven

Unni, A. (2022, January 11). *What You Need To Know About Crime As A Service (CSaaS)*. Retrieved from StickmanCyber: https://www.stickmancyber.com/cybersecurity-blog/what-you-need-to-know-about-crime-as-a-service-csaas#:~:text=What%20is%20CaaS%3F,out%20attacks%20with%20relative%20ease.

Vicente, J. (2024, January 31). *Tracking 15 Years of Qakbot Development*. Retrieved from Zscaler: https://www.zscaler.com/blogs/security-research/tracking-15-years-qakbot-development

11. Appendices

None

Word count: 8158