Hibernating Qakbot | ThreatLabz

By Meghraj Nandanwar, Satyam Singh, Pradeep Mahato

Published: 2023-07-25 · Archived: 2026-04-05 15:24:03 UTC

Analysis Of Qakbot Attack Chains

This section presents distinct variations of the Qakbot banking trojan attack chain, examined across samples discovered between March and May of 2023. The case studies below specifically concentrate on how diverse file formats and techniques execute the Qakbot end payload on the victim's machine, instead of directly dropping and executing the malware.

Case Study 1: March 2023 - Evolving Qakbot Tactics: Exploiting File Formats for Deceptive Payload Delivery

At the outset of the year, Qakbot began spreading through OneNote files. Subsequently, in March, a shift was observed, as Qakbot transitioned to using PDF and HTML files as the initial attacking vectors to download further stage files, leading to the delivery of the final payload. These file formats are commonly utilized by numerous threat actors to infect users.

Multiple attack chains were observed, wherein Qakbot utilizes PDF files as the initial vector to download the next stage file, which contains an obfuscated JS (Javascript) file bearing names like "Invoice," "Attach," "Report," or "Attachments" to deceive users into executing the file. Upon running the JS file, Qakbot initially creates a registry key and adds the base64 encoded Powershell command into the registry key using the reg.exe command line tool, enabling the download and execution of the Qakbot DLL.

Attack Chain: MalSpam -> PDF -> URL -> JS -> PS -> Qakbot Payload

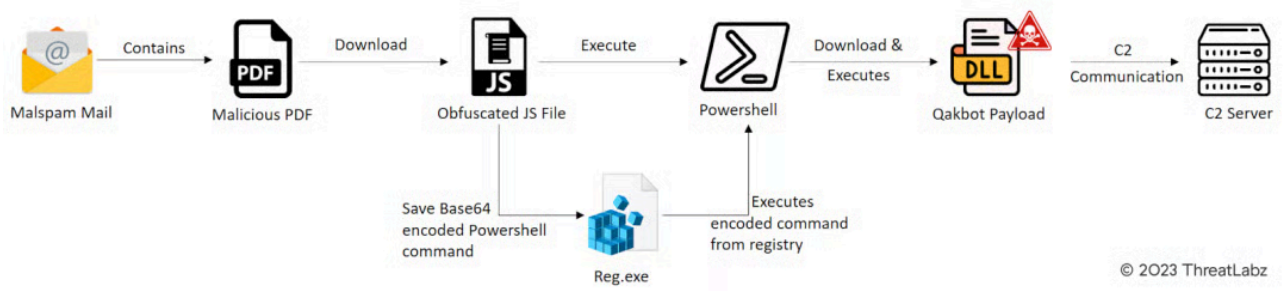


Figure 1 - Illustrates the attack chain involving a Malicious PDF as the initial attack vector.

Qakbot recently reverted to utilizing HTML smuggling as a means of delivering its initial attack payload. This technique was observed across numerous campaigns during the previous year. In March, the identification of several new malspam emails indicated that threat actors were leveraging Latin-themed HTML files to facilitate the download of zip archives. These archives contained an obfuscated JS file, initiating a sequence similar to the one depicted in Fig.1, ultimately leading to the delivery of the Qakbot payload.

The attack chain discovered in March follows the following progression: Malspam -> HTML -> URL -> ZIP -> JS -> PS -> Qakbot Payload

In this chain, malspam serves as the initial delivery method, targeting unsuspecting victims through deceptive emails. The HTML files play a pivotal role in exploiting HTML smuggling techniques, concealing malicious activities within seemingly innocuous web content.

Upon accessing the HTML files, URLs are triggered, initiating the download of zip archives containing the obfuscated JS file. The use of obfuscation ensures that the malicious code remains hidden from casual detection and analysis, enhancing the threat actors' ability to evade detection.

Subsequently, the JS file is executed, setting off a series of actions that culminate in the execution of a Powershell command (PS). The Powershell command is instrumental in obtaining and executing the final payload, which, in this case, is the notorious Qakbot banking trojan. During our campaign follow up we found this sample from Twitter handle @Pr0xylife and @Cryptolaemus1.

This resurgence of HTML smuggling by Qakbot highlights the significance of continuous monitoring and awareness of evolving malware tactics and shifting attack chains for detecting and countering such threats.

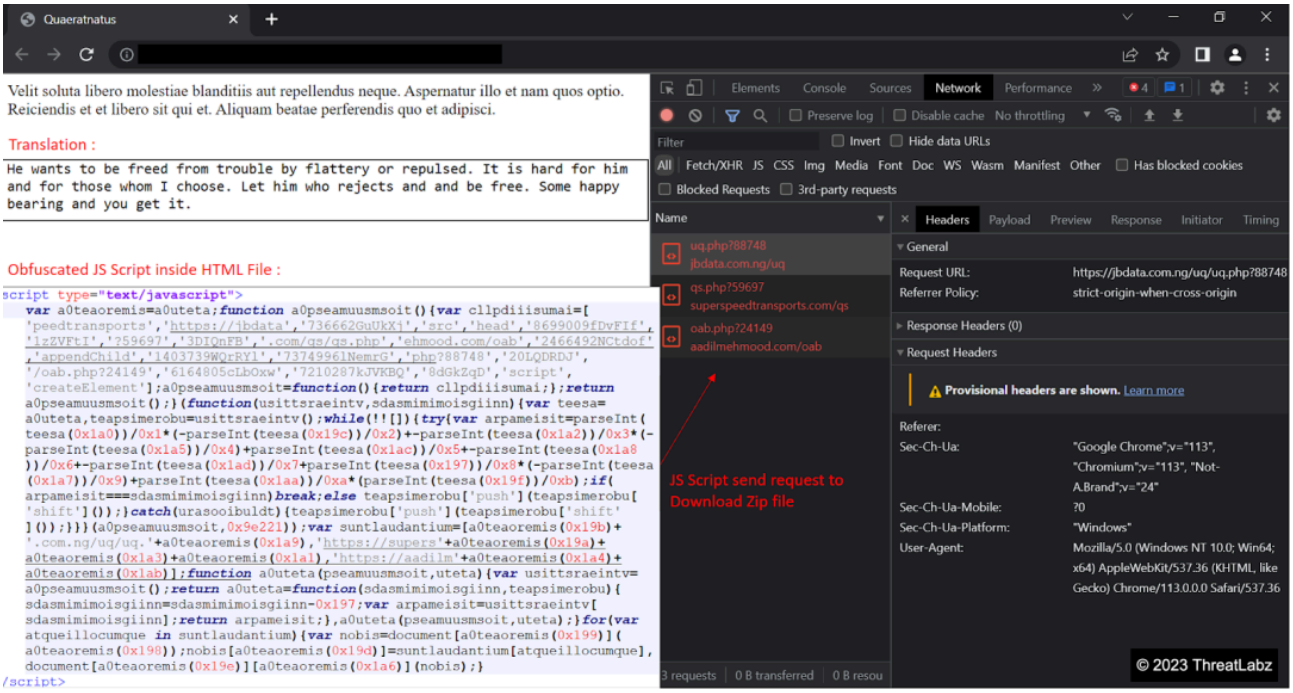


Figure 2 - Shows the attack chain with a Malicious HTML file as the initial attack vector.

Later, a similar attack chain was identified, where the initial attack vector involved a PDF file. This PDF file was designed to download a zip archive, which, in turn, contained an obfuscated WSF/HTA file. Upon execution, the WSF/HTA file ran a base64 encoded Powershell command, leading to the download and execution of the final Qakbot payload.

The observed attack chain follows the following progression: Malspam -> PDF -> ZIP -> WSF/HTA -> PS -> Qakbot Payload

In this scenario, malspam continues to serve as the initial method of propagation, disseminating malicious content through email campaigns. The PDF file, acting as the attack vector, entices users to access its contents, ultimately triggering the download of a zip archive.

Inside the zip archive, an obfuscated WSF/HTA file is concealed, obscuring its malicious intent and complicating detection efforts. Once executed, the WSF/HTA file initiates a base64 encoded Powershell command, a common technique used by threat actors to download and execute further payloads without leaving a conspicuous trail.

The culmination of this attack chain results in the delivery and execution of the Qakbot banking trojan against the targeted system and its users.

Figure 3 - Features a Malicious PDF as the initial attack vector in the attack chain, accompanied by WSF and HTA files.

In another discovery made by ThreatLabz researchers, a variant of the Qakbot malware was observed employing a stealthy attack chain with the use of Microsoft Excel add-ins (XLL) as the initial vector. Microsoft Office add-ins are DLL files with distinct extensions based on the application they are designed for. While Microsoft Word add-ins use the '.wll' extension, Excel add-ins utilize the '.xll' extension.

The choice of using XLL files as the initial attacking vector is strategic for threat actors due to their ease of use. Unlike Word add-ins that must be placed in specific trusted locations depending on the Office version, XLL files are automatically loaded and opened by the Excel application, simplifying the delivery process for the attackers.

Moreover, XLL files possess unique characteristics that differentiate them from regular DLLs. They can have export functions that are invoked by the Excel Add-In manager when triggered by Excel. Upon launching an XLL file, Excel activates the export functions defined by the XLL interface, such as **xlAutoOpen** and **xlAutoClose**,

similar to **Auto_Open** and **Auto_Close** in VBA macros. This mechanism is exploited by the attackers to load the malicious payload seamlessly, evading security measures and detection.

The attack chain follows a sequence where the threat actor utilizes a .xll file in the initial phase. When a user opens this .xll file, it proceeds to drop two files, "**1.dat**" and "**2.dat**," into the '**\Users\User\AppData\Roaming**' directory. The "**1.dat**" file contains a 400-byte header of the PE file, while the "**2.dat**" file holds the remaining data of the PE file. These two files are then combined to create the "**3.dat**" file, which contains the actual Qakbot payload. Additionally, the attackers establish scheduled tasks to execute the Qakbot payload every 10 minutes, ensuring its persistence on the victim's machine.

The observed attack chain follows the following progression: Malspam -> ZIP -> XLL > Qakbot Payload

This attack chain sample underscores the ever-evolving nature of Qakbot, which continuously adapts its tactics and techniques to avoid detection and infiltrate systems. By utilizing XLL files and implementing sophisticated techniques to hide and deliver its payload, Qakbot continues to pose a significant threat to users and organizations.

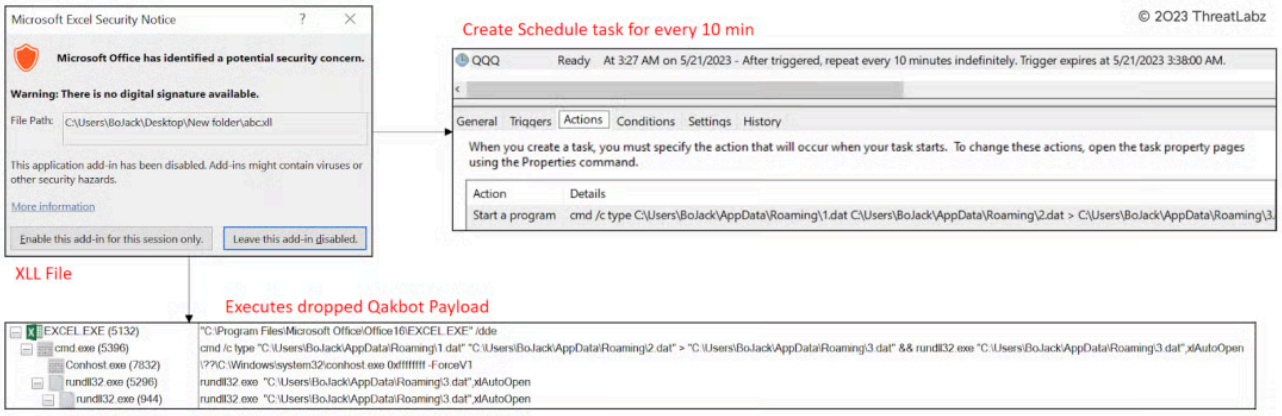


Figure 4 - Shows the attack chain involving Malicious XLL files as the initial attack vector.

Case Study 2: April 2023 - Adapting Qakbot: Unraveling the XMLHTTP Experiment in the Attack Chain

In April, researchers noted more significant changes in the Qakbot attack chain, as the samples revealed the malware continued to experiment with different file formats to infect users.

In this evolved attack chain, the WSF (Windows Script File) contains a hex-encoded **XMLHTTP** request to download the Qakbot payload, replacing the previous base64 encoded PowerShell command.

The observed attack chain follows the following progression: Malspam -> PDF -> ZIP -> WSF -> XMLHTTP -> Qakbot Payload

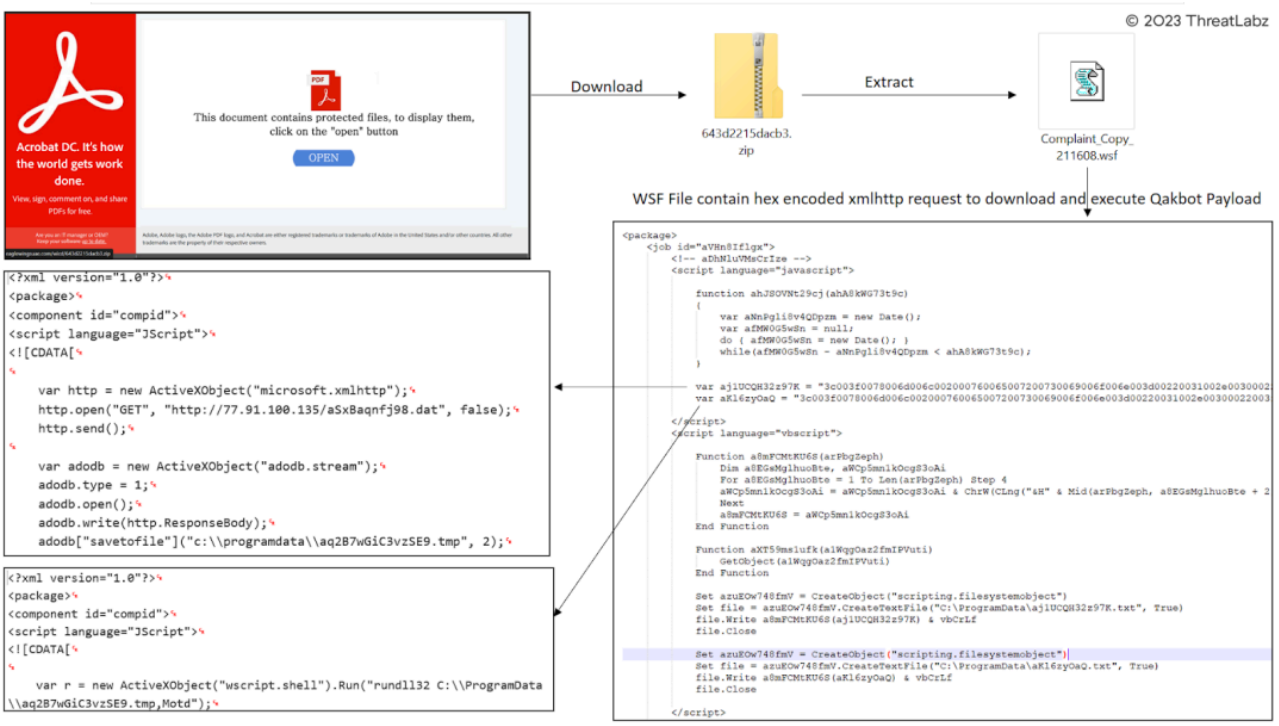


Figure 5 - Depicts the attack chain utilizing the XMLHTTP file.

Towards the end of April, Qakbot's persistent use of OneNote files as the initial attack vector was still evident in its latest campaign. OneNote files served as an effective disguise, luring unsuspecting users into opening and executing the embedded contents. The attackers leveraged the familiarity and widespread use of OneNote files to increase the chances of successful infections.

Within this attack chain, the OneNote file contains an embedded MSI (Microsoft Installer) file. This MSI file was designed to trick users by posing as a legitimate Microsoft Azure installer, exploiting victims' trust in these familiar software installations and delivering the Qakbot payload.

The MSI file was purposely crafted to include several components, enhancing its evasive capabilities and making it difficult for security systems to detect its true intent. Among these components, a self-deletion PowerShell script was incorporated, allowing the malware to erase its tracks after execution, reducing the chances of detection and analysis.

Furthermore, the MSI file contained a configuration file that held essential information, including the path to execute a WSF (Windows Script File) script. This WSF script served as a critical link in the attack chain, acting as an intermediary to facilitate the download and execution of the Qakbot payload.

To ensure further obfuscation and evasion, the WSF script was hex-encoded, making it challenging for traditional security measures to interpret its true purpose. This encoded script was responsible for executing an **XMLHTTP** request, a technique used to download the actual Qakbot payload from a remote server.

Through this intricate sequence of deception and evasion, attackers aim to successfully deliver the Qakbot payload onto victim machines. By continuously adapting their attack techniques and leveraging familiar file formats, the threat actors behind Qakbot seek to stay one step ahead of security defenses and professionals.

The observed attack chain follows the following progression: Malspam -> OneNote -> MSI -> WSF -> XMLHTTP -> Qakbot Payload

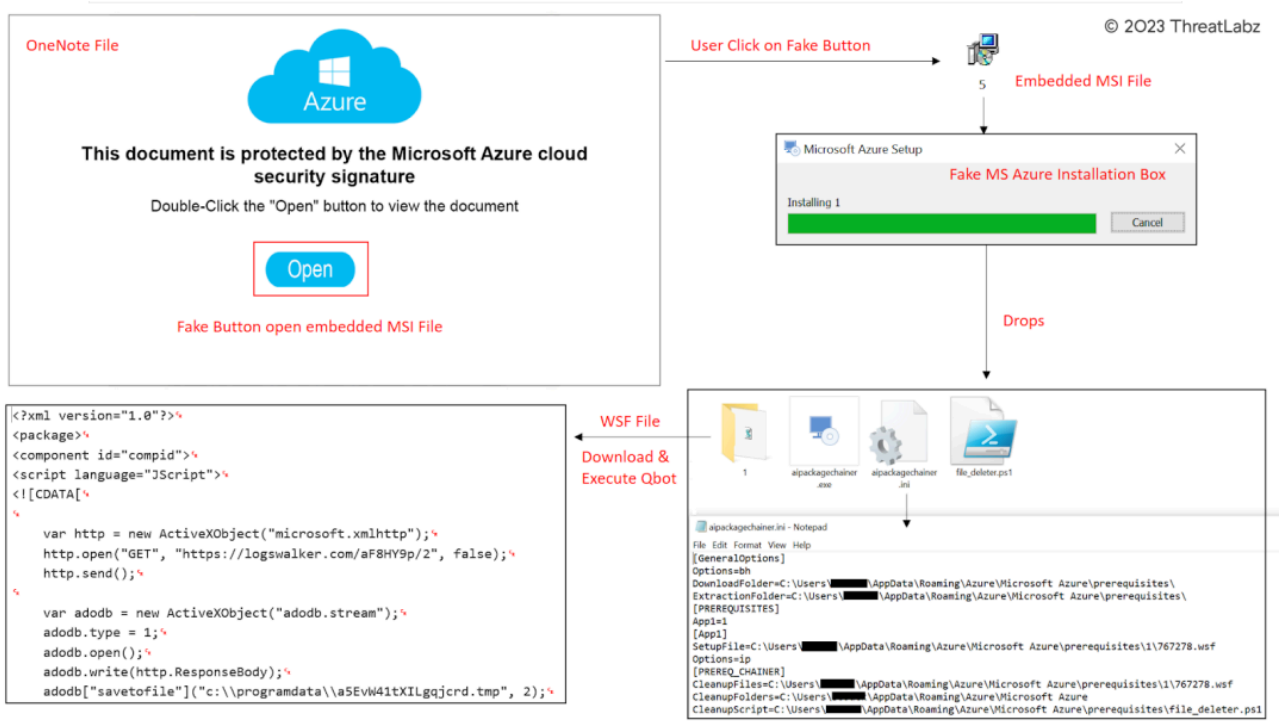


Figure 6 - Evolving Attack Chain: Leveraging Malicious OneNote and MSI Files as Initial Attack Vector.

Case Study 3: May 2023: Qakbot Explores Advanced Defense Evasion Tactics

Throughout the month of May, researchers closely monitored Qakbot's activities and observed the threat actor's efforts to experiment with innovative Defense Evasion Tactics aimed at infecting users and evading detection. Alongside changes in the attack chain, Qakbot introduced sophisticated techniques, including Indirect Command Execution using conhost.exe and DLL Side-Loading, further complicating its detection and removal.

In this attack chain, Qakbot takes advantage of conhost.exe as a proxy binary to bypass defensive measures. By employing conhost.exe, Qakbot attempts to outwit security counter-measures that restrict the use of typical command-line interpreters. This enables the threat actor to execute commands using various Windows utilities, creating a clever diversion and making it more challenging for security tools to identify and mitigate the threat effectively.

The attack sequence starts with malspam, where malicious emails are distributed to unsuspecting victims. These emails often contain malicious attachments disguised as innocent files, luring users into opening them. The threat actors use PDF files packed within ZIP archives, which, when accessed, lead to the execution of WSF files via XMLHTTP.

To further obscure its activities, Qakbot then leverages conhost.exe, employing it as an intermediary to carry out specific commands. This tactic is part of Qakbot's strategy to operate stealthily within the compromised system, remaining undetected by conventional security mechanisms that may primarily focus on detecting direct malicious code execution.

The ultimate goal of this attack chain is to deliver the Qakbot payload, allowing the malware to infiltrate the victim's system, steal sensitive information, and potentially carry out other malicious activities, including espionage and financial theft.

The observed attack chain follows the following progression: Malspam -> PDF -> ZIP -> WSF -> XMLHTTP -> conhost.exe -> Qakbot Payload

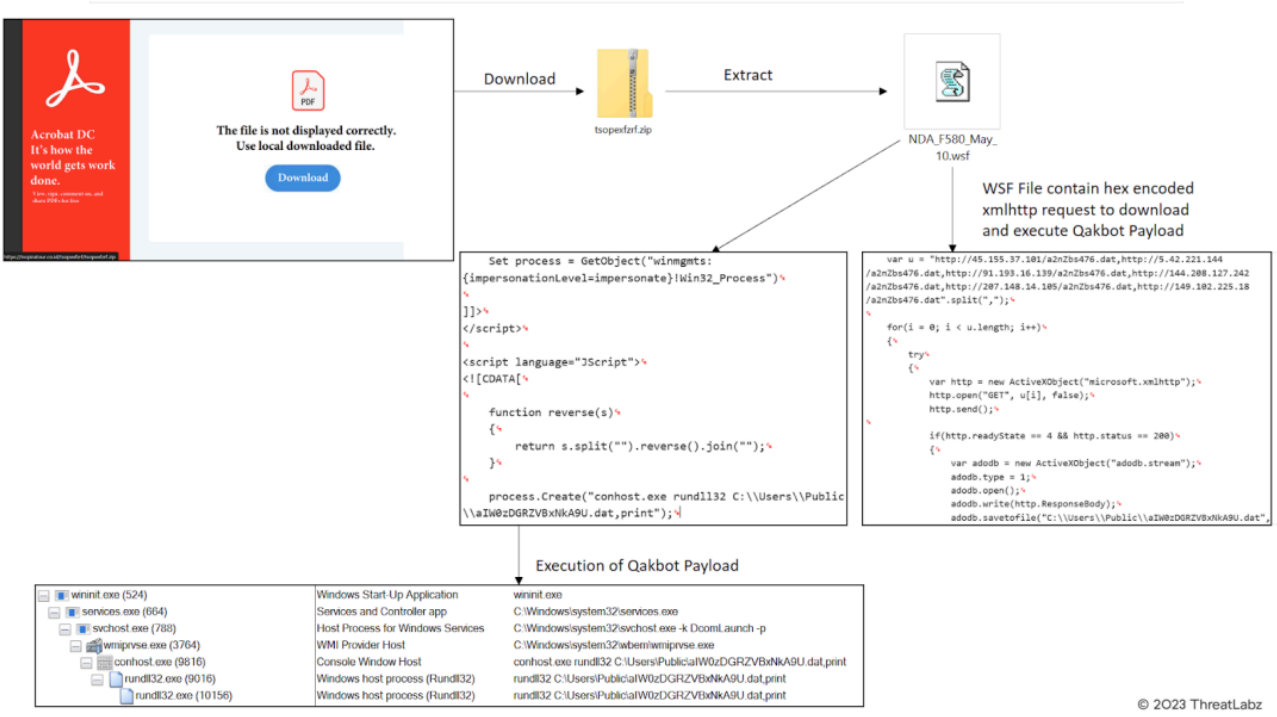


Figure 7 - Demonstrates Qakbot's utilization of Indirect Command Execution with conhost.exe.

In this intricate attack chain, the initial vector is a ZIP file that conceals an executable (EXE) file. Upon execution, the EXE file loads a hidden dynamic-link library (DLL) that employs a curl command to download the final Qakbot payload. This attack chain also involves the use of DLL side loading technique, adding another layer of complexity to the attack.

The threat actor initiates this attack through malspam, sending deceptive emails containing URLs that lead to the delivery of the ZIP file. Once the user accesses the ZIP file and executes the embedded EXE file, the attack unfolds, triggering the loading of the concealed DLL. This DLL utilizes a curl command to download the final Qakbot payload from a remote server.

By incorporating DLL side loading, the threat actor creates a diversion, making it more challenging for security measures to detect the malicious activities. This advanced technique allows the malware to execute code indirectly and evade traditional detection mechanisms, adding an extra layer of sophistication to the attack.

The attack sequence follows: Malspam -> URL -> ZIP -> EXE -> DLL -> CURL -> Qakbot Payload

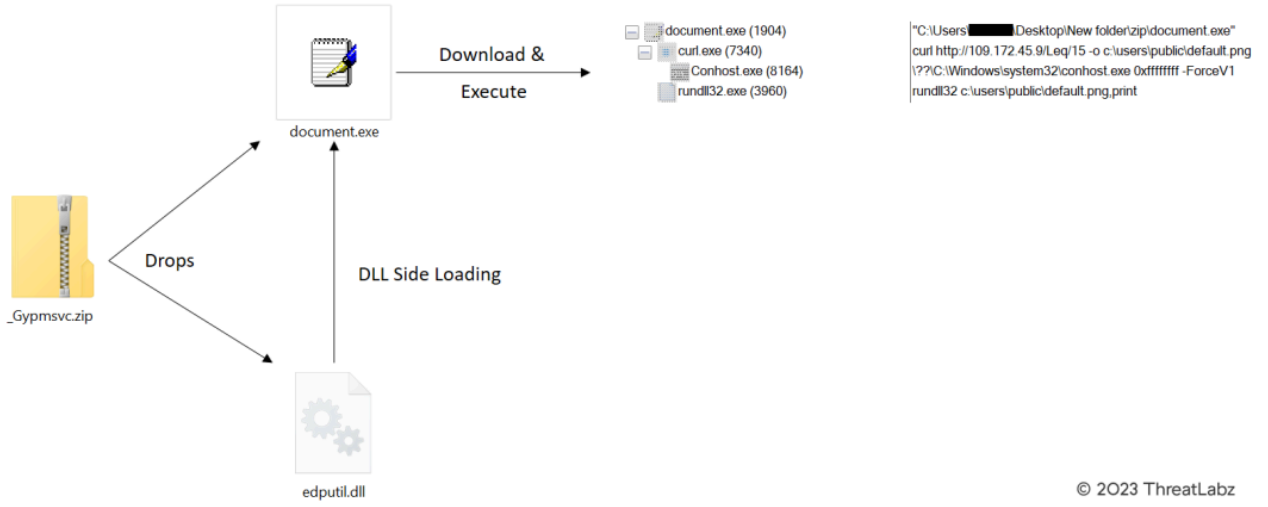
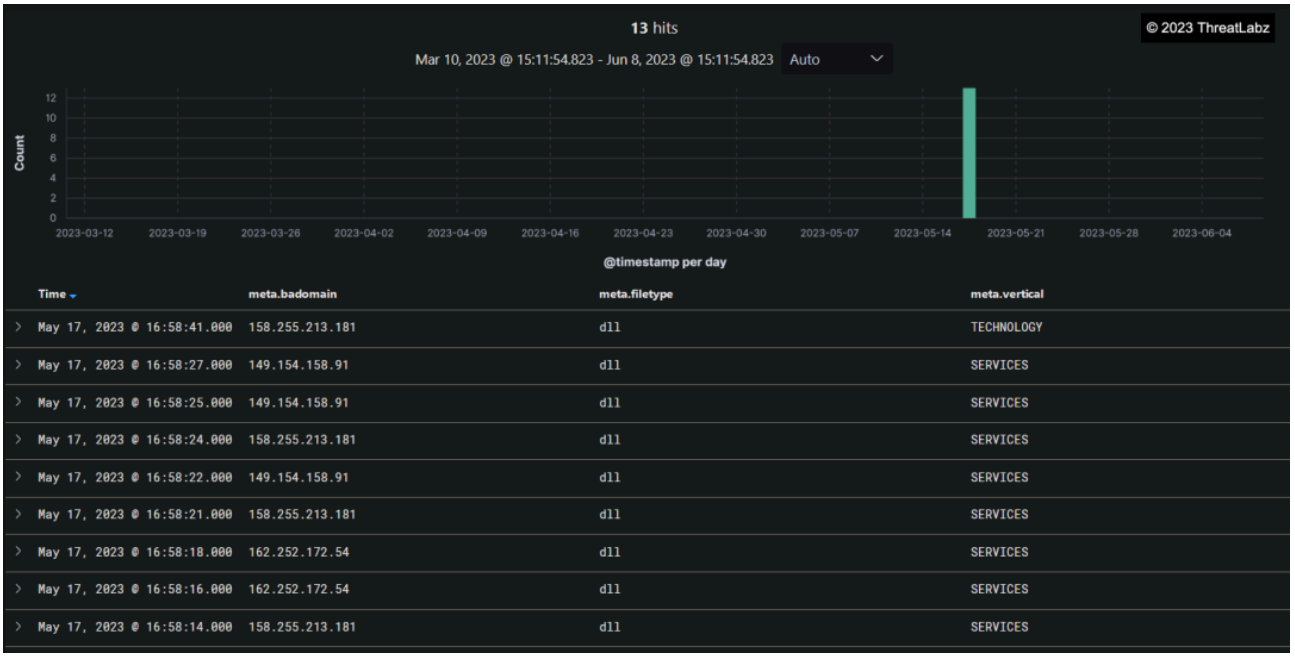


Figure 8 - Depicts Qakbot's utilization of DLL Side Loading in its attack chain.

On May 17th, several Pikabot samples were distributed using tactics, techniques, and procedures (TTPs) similar to those of Qakbot within the Zscaler Cloud. This discovery is valuable as it highlights a potential link or copycat scenario and provides insights into Pikabot malware behavior and distribution methods. The resemblance between Pikabot and Qakbot, including similarities in their behavior and internal campaign identifiers, suggests a possible connection between the two. However, there is not yet sufficient evidence to definitively link these malware families to the same threat actor.

Understanding the similarities and differences between Pikabot and Qakbot is critical for cybersecurity professionals to effectively respond to these threats. The identification of new malware variants helps security teams stay ahead of evolving attack trends, enabling them to adjust their defense strategies accordingly. By closely monitoring the behavior and distribution patterns of these malware families, security experts can enhance their threat intelligence and improve their ability to detect and mitigate such attacks in the future.

Threatlabz's ongoing technical analysis of Pikabot will provide further insights into its capabilities and potential impact on organizations. Keeping abreast of such developments and conducting thorough examinations of new malware variants is crucial for safeguarding networks, systems, and sensitive data from cyber threats. As the investigation progresses, security professionals can better assess the potential risks posed by Pikabot and formulate effective mitigation measures to protect against its infiltration and harmful activities.



Time	meta.badomain	meta.filetype	meta.vertical
May 17, 2023 @ 16:58:41.000	158.255.213.181	dll	TECHNOLOGY
May 17, 2023 @ 16:58:27.000	149.154.158.91	dll	SERVICES
May 17, 2023 @ 16:58:25.000	149.154.158.91	dll	SERVICES
May 17, 2023 @ 16:58:24.000	158.255.213.181	dll	SERVICES
May 17, 2023 @ 16:58:22.000	149.154.158.91	dll	SERVICES
May 17, 2023 @ 16:58:21.000	158.255.213.181	dll	SERVICES
May 17, 2023 @ 16:58:18.000	162.252.172.54	dll	SERVICES
May 17, 2023 @ 16:58:16.000	162.252.172.54	dll	SERVICES
May 17, 2023 @ 16:58:14.000	158.255.213.181	dll	SERVICES

Figure 9 - Shows the distribution of Pikabot, discovered in Zscaler Cloud.

Explore more Zscaler blogs
