# Duck Hunting w/Falcon Complete Pt. 3: QakBot Countermeasures

crowdstrike.com/blog/duck-hunting-with-falcon-complete-qakbot-countermeasures/

The Falcon Complete Team October 14, 2020

This blog is the last in a three-part series presenting the CrowdStrike® Falcon Complete™ team's analysis of the recent QakBot campaigns observed in the wild and outlining a strategy for the remote identification of a QakBot-infected host. While Part 1 and Part 2 provided an analysis of techniques used by the threat actor to gain successful infections, Part 3 provides recommendations for countermeasures that can be deployed via the CrowdStrike Falcon® platform to prevent and contain infections before a widespread incident occurs. We also outline a strategy for how the team helps organizations recover from incidents via our remote remediation capabilities.

QakBot is an eCrime banking trojan that is capable of spreading laterally throughout a network. Utilizing a worm-like functionality, it spreads through brute forcing network shares, brute forcing Active Directory user group accounts or via SMB exploitation.

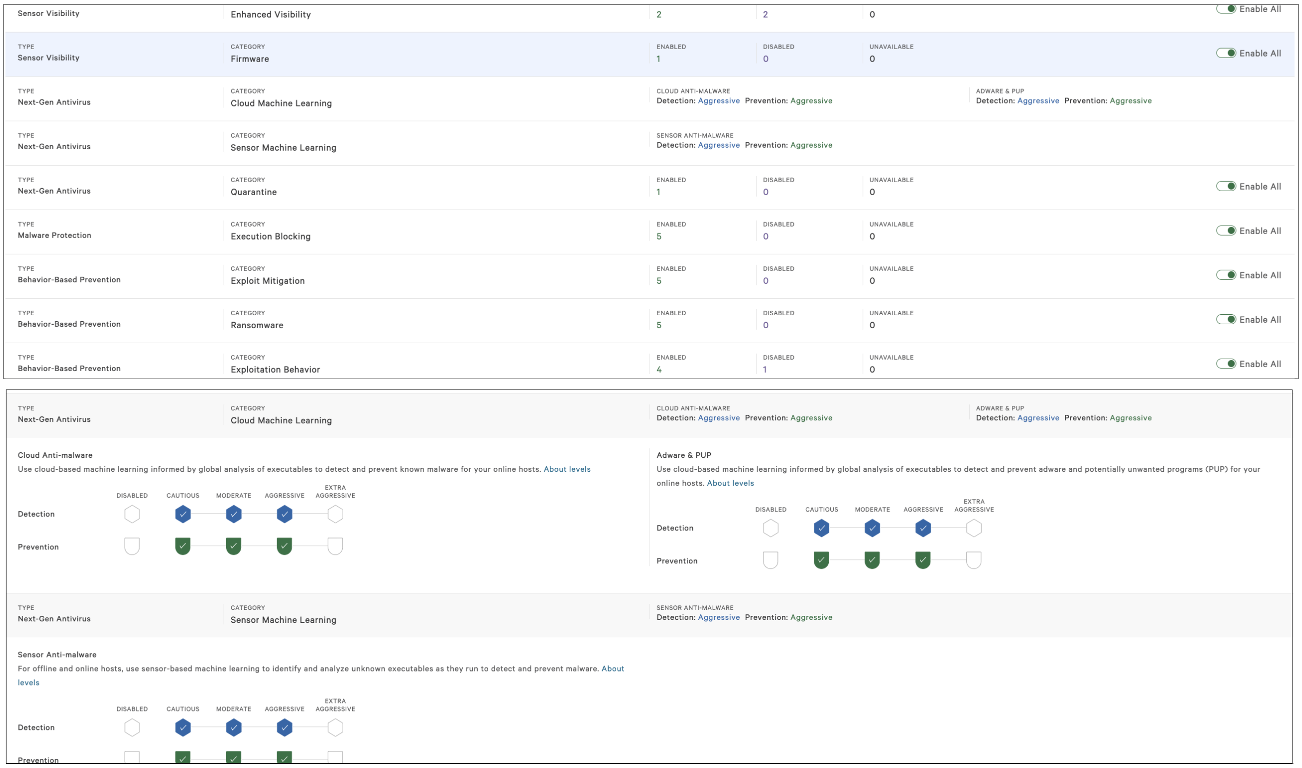
QakBot also employs a robust set of anti-analysis features to evade detection and frustrate analysis. Despite these protections, the CrowdStrike Falcon® platform detects and prevents the malware from completing its execution chain.

## Solution: Prevention, Containment and Remediation

The Falcon Complete team's approach to defeating the QakBot threat can be defined in several ways, depending on the tailored approach created for the customer. The best way to defend against QakBot is never allowing it to get a foothold in the first place, and the Falcon platform offers the prevention policies that are effective in stopping QakBot in its tracks. For situations where prevention is not enabled, a strategy of containment and fast remote response allows our analysts to quickly remediate the infection via the Falcon Real Time Response (RTR) console, resulting in less downtime, less interference and more productivity.

### Configuring Proper Prevention Policies

Prevention policies are policies configured in the Falcon UI that allow organizations to customize how aggressive the Falcon sensor is with detections and preventions. Organizations can choose to configure these policies themselves or take advantage of the Falcon Complete team's expertise to configure and offer guidance for best practices regarding these configurations.



| | | | | | |
|---|---|---|---|---|---|
| Sensor Visibility | Enhanced Visibility | 2 | 2 | 0 | Enable All |
| TYPE<br>Sensor Visibility | CATEGORY<br>Firmware | ENABLED<br>1 | DISABLED<br>0 | UNAVAILABLE<br>0 | Enable All |
| TYPE<br>Next-Gen Antivirus | CATEGORY<br>Cloud Machine Learning | CLOUD ANTI-MALWARE<br>Detection: Aggressive Prevention: Aggressive | | ADWARE & PUP<br>Detection: Aggressive Prevention: Aggressive | |
| TYPE<br>Next-Gen Antivirus | CATEGORY<br>Sensor Machine Learning | SENSOR ANTI-MALWARE<br>Detection: Aggressive Prevention: Aggressive | | | |
| TYPE<br>Next-Gen Antivirus | CATEGORY<br>Quarantine | ENABLED<br>1 | DISABLED<br>0 | UNAVAILABLE<br>0 | Enable All |
| TYPE<br>Malware Protection | CATEGORY<br>Execution Blocking | ENABLED<br>5 | DISABLED<br>0 | UNAVAILABLE<br>0 | Enable All |
| TYPE<br>Behavior-Based Prevention | CATEGORY<br>Exploit Mitigation | ENABLED<br>5 | DISABLED<br>0 | UNAVAILABLE<br>0 | Enable All |
| TYPE<br>Behavior-Based Prevention | CATEGORY<br>Ransomware | ENABLED<br>5 | DISABLED<br>0 | UNAVAILABLE<br>0 | Enable All |
| TYPE<br>Behavior-Based Prevention | CATEGORY<br>Exploitation Behavior | ENABLED<br>4 | DISABLED<br>1 | UNAVAILABLE<br>0 | Enable All |

| | | | |
|---|---|---|---|
| TYPE<br>Next-Gen Antivirus | CATEGORY<br>Cloud Machine Learning | CLOUD ANTI-MALWARE<br>Detection: Aggressive Prevention: Aggressive | ADWARE & PUP<br>Detection: Aggressive Prevention: Aggressive |

Cloud Anti-malware

Use cloud-based machine learning informed by global analysis of executables to detect and prevent known malware for your online hosts. About levels

| | DISABLED | CAUTIOUS | MODERATE | AGGRESSIVE | EXTRA AGGRESSIVE |
|---|---|---|---|---|---|
| Detection | ○ | ✓ | ✓ | ✓ | ○ |
| Prevention | ○ | ✓ | ✓ | ✓ | ○ |

Adware & PUP

Use cloud-based machine learning informed by global analysis of executables to detect and prevent adware and potentially unwanted programs (PUP) for your online hosts. About levels

| | DISABLED | CAUTIOUS | MODERATE | AGGRESSIVE | EXTRA AGGRESSIVE |
|---|---|---|---|---|---|
| Detection | ○ | ✓ | ✓ | ✓ | ○ |
| Prevention | ○ | ✓ | ✓ | ✓ | ○ |

| | | |
|---|---|---|
| TYPE<br>Next-Gen Antivirus | CATEGORY<br>Sensor Machine Learning | SENSOR ANTI-MALWARE<br>Detection: Aggressive Prevention: Aggressive |

Sensor Anti-malware

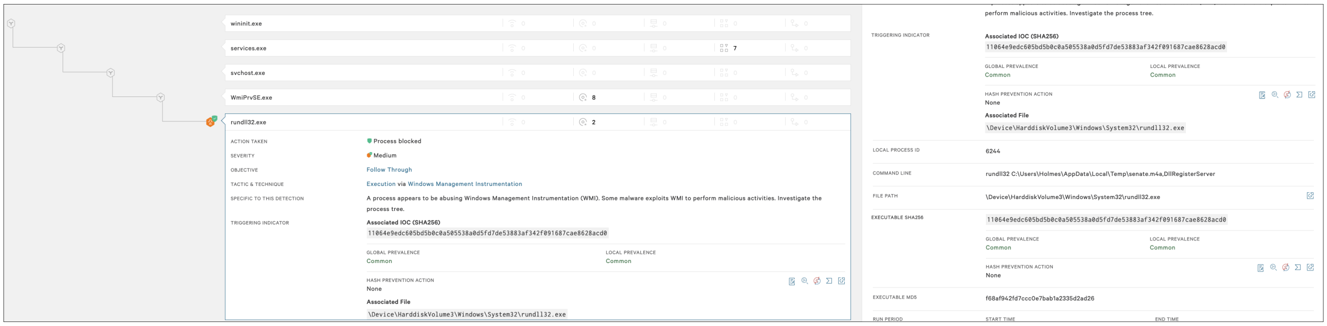
For offline and online hosts, use sensor-based machine learning to identify and analyze unknown executables as they run to detect and prevent malware. About levels

| | DISABLED | CAUTIOUS | MODERATE | AGGRESSIVE | EXTRA AGGRESSIVE |
|---|---|---|---|---|---|
| Detection | ○ | ✓ | ✓ | ✓ | ○ |
| Prevention | ○ | ✓ | ✓ | ✓ | ○ |

Figure 1. Prevention policy settings as shown in the Falcon UI (click image to enlarge)

Falcon's next-gen antivirus has the ability to block malware based on machine learning and behavioral pattern analysis. Being a non-signature-reliant platform with the proper prevention policies in place, Falcon reliably prevents known QakBot infections before any second-stage payloads are executed.

With prevention policies configured like those depicted in Figure 1, we can see that the execution of a senate.m4a Zloader payload was blocked, and the detection details appear in the Falcon UI for analyst review (see Figure 2).



wininit.exe 0 0 0 0 0
services.exe 0 0 0 7 0
svchost.exe 0 0 0 0 0
WmiPrvSE.exe 0 8 0 0 0
rundll32.exe 0 2 0 0 0

ACTION TAKEN: Process blocked
SEVERITY: Medium
OBJECTIVE: Follow Through
TACTIC & TECHNIQUE: Execution via Windows Management Instrumentation
SPECIFIC TO THIS DETECTION: A process appears to be abusing Windows Management Instrumentation (WMI). Some malware exploits WMI to perform malicious activities. Investigate the process tree.
TRIGGERING INDICATOR: Associated IOC (SHA256)
11064e9edc605bd5b0c0a505538a0d5fd7de53883af342f091687cae8628acd0
GLOBAL PREVALENCE: Common
LOCAL PREVALENCE: Common
HASH PREVENTION ACTION: None
Associated File
\Device\HarddiskVolume3\Windows\System32\rundll32.exe

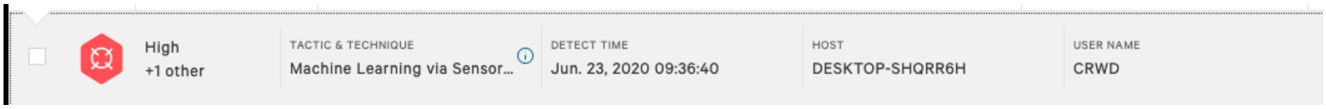
perform malicious activities. Investigate the process tree.
TRIGGERING INDICATOR: Associated IOC (SHA256)
11064e9edc605bd5b0c0a505538a0d5fd7de53883af342f091687cae8628acd0
GLOBAL PREVALENCE: Common
LOCAL PREVALENCE: Common
HASH PREVENTION ACTION: None
Associated File
\Device\HarddiskVolume3\Windows\System32\rundll32.exe
LOCAL PROCESS ID: 6244
COMMAND LINE: rundll32 C:\Users\Holmes\AppData\Local\Temp\senate.m4a,DllRegisterServer
FILE PATH: \Device\HarddiskVolume3\Windows\System32\rundll32.exe
EXECUTABLE SHA256: 11064e9edc605bd5b0c0a505538a0d5fd7de53883af342f091687cae8628acd0
GLOBAL PREVALENCE: Common
LOCAL PREVALENCE: Common
HASH PREVENTION ACTION: None
EXECUTABLE MD5: f68af942fd7ccc0e7bab1a2335d2ad26
RUN PERIOD: START TIME END TIME

Figure 2. Falcon blocking second-stage payload execution of senate.m4a (click image to enlarge)

## Host Network Containment via the Falcon UI

In the event that prevention policies are not set to actively block the QakBot threat — as is sometimes the case in more sensitive environments — customers can take advantage of the 24/7/365 virtual security operations center (SOC) offered by Falcon Complete. In these cases, Falcon Complete will triage, stop lateral spread and remediate the QakBot banking trojan threat.

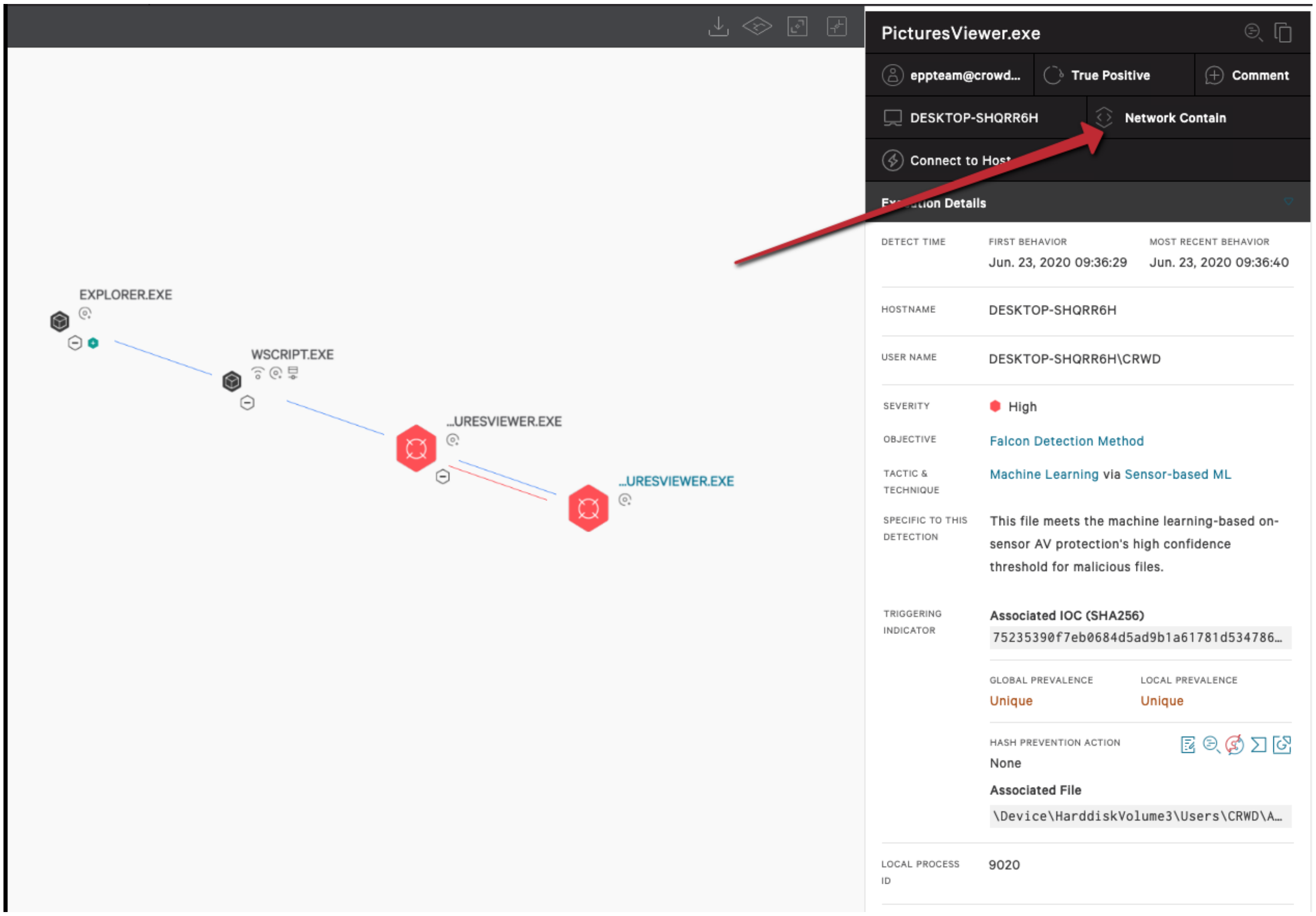
In the instance of an unprevented QakBot infection, the Falcon Complete team receives a high-confidence alert for malicious files, triggered by Falcon's machine-learning algorithms.



□ High +1 other | TACTIC & TECHNIQUE: Machine Learning via Sensor... | DETECT TIME: Jun. 23, 2020 09:36:40 | HOST: DESKTOP-SHQRR6H | USER NAME: CRWD

Figure 3. Detection in the UI as shown by Falcon (click image to enlarge)

Upon expanding the alert, the analyst can recognize the QakBot threat by the tactics employed — many discovered during CrowdStrike's tracking of this threat.



DETECT TIME
FIRST BEHAVIOR Jun. 23, 2020 09:36:29
MOST RECENT BEHAVIOR Jun. 23, 2020 09:36:40

HOSTNAME DESKTOP-SHQRR6H

USER NAME DESKTOP-SHQRR6H\CRWD

SEVERITY ● High

OBJECTIVE Falcon Detection Method

TACTIC & TECHNIQUE Machine Learning via Sensor-based ML

SPECIFIC TO THIS DETECTION This file meets the machine learning-based on-sensor AV protection's high confidence threshold for malicious files.

TRIGGERING INDICATOR Associated IOC (SHA256)
75235390f7eb0684d5ad9b1a61781d534786…

GLOBAL PREVALENCE Unique
LOCAL PREVALENCE Unique

HASH PREVENTION ACTION
None

Associated File
\Device\HarddiskVolume3\Users\CRWD\A…

LOCAL PROCESS ID 9020

Figure 4. Ability to network-contain hosts in emergency situations (click image to enlarge)

If among the pre-approved countermeasures, the analyst can network-contain the host to prevent the lateral spread of the info-stealer within the environment. This limits business impact and productivity loss for users, and saves time and cleanup for your internal SOC team.

## Remediation with Falcon RTR

QakBot malware is fairly simple in its functionality, but its capability to move laterally can be potentially devastating in enterprise networks. Because of this worm-like spreading and its costly repercussions, the Falcon Complete team has classified the difficulty of the remediation process as “Hard.” The following is a brief illustration of the remote remediation process via Falcon Real Time Response (RTR).

**The remediation of QakBot can be broken into three distinct steps:**

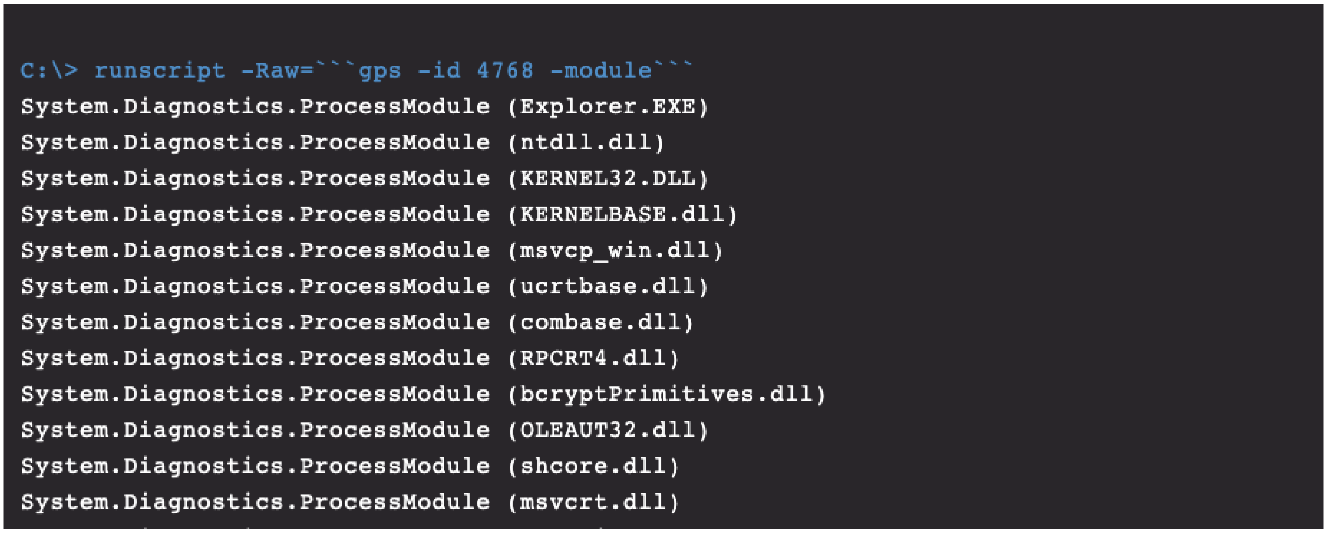
1. Killing the malicious processes (e.g., injected explorer)

2. Removing the persistence mechanism (e.g., scheduled task, registry run key)
3. Removing disk artifacts (e.g., binaries and directories).

***Please note: Some of the examples in the following scenario have CrowdStrike Falcon configured with DETECTIONS ONLY and PREVENTIONS off for illustrative purposes. A properly configured Falcon instance, as noted previously, would prevent the activity presented here.***

## STEP 1. Finding and Killing the Malicious `explorer.exe` Process

QakBot will create a new instance of the `explorer.exe` process and inject itself into the new process. It is possible for multiple instances of this process to occur, but responders must determine the malicious instance. It may not be immediately clear which `explorer.exe` process is legitimate, but this can be determined by querying the process with the “-Module” parameter, as shown in Figure 5.



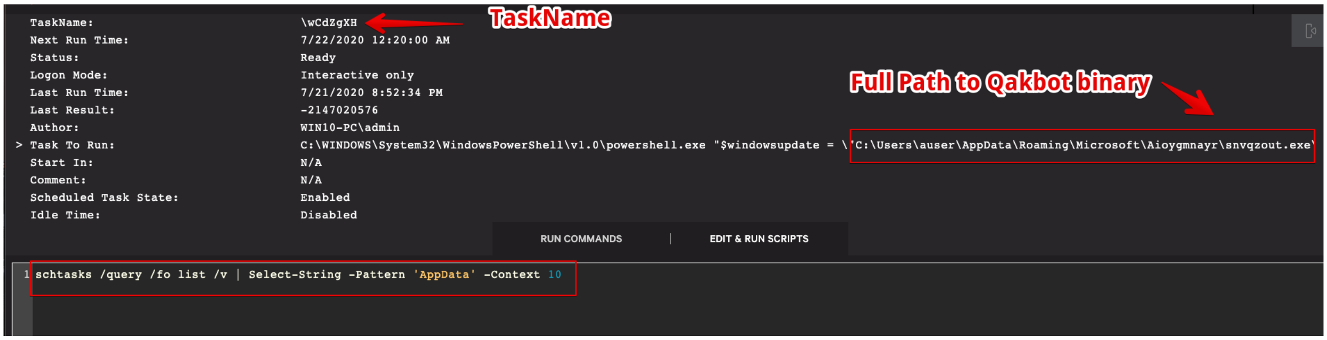
```
C:\> runscript -Raw=```gps -id 4768 -module```
System.Diagnostics.ProcessModule (Explorer.EXE)
System.Diagnostics.ProcessModule (ntdll.dll)
System.Diagnostics.ProcessModule (KERNEL32.DLL)
System.Diagnostics.ProcessModule (KERNELBASE.dll)
System.Diagnostics.ProcessModule (msvcp_win.dll)
System.Diagnostics.ProcessModule (ucrtbase.dll)
System.Diagnostics.ProcessModule (combase.dll)
System.Diagnostics.ProcessModule (RPCRT4.dll)
System.Diagnostics.ProcessModule (bcryptPrimitives.dll)
System.Diagnostics.ProcessModule (OLEAUT32.dll)
System.Diagnostics.ProcessModule (shcore.dll)
System.Diagnostics.ProcessModule (msvcrt.dll)
```

Figure 5. Query for explorer.exe process injection (click image to enlarge)

In the example in Figure 5, “gps” is a built-in alias for the “Get-Process” cmdlet, and the process id was determined by an early query (not shown). The legitimate explorer process will have several modules associated with it, but the malicious instance will have very few — typically less than ten. In this case, Falcon had already prevented the injected process, but if there were a malicious instance present, it could be killed with the following command: “kill ‘<PID>.’”

## STEP 2. Removing Persistence

QakBot typically employs a scheduled task and a registry run key for persistence mechanisms on compromised hosts. Figure 6 shows a query for scheduled tasks on the affected host. The search can be assisted by the historical knowledge that will point to a binary in %AppData%, so this is used as a search string to quickly identify not only the malicious TaskName but also the full path to the main binary.

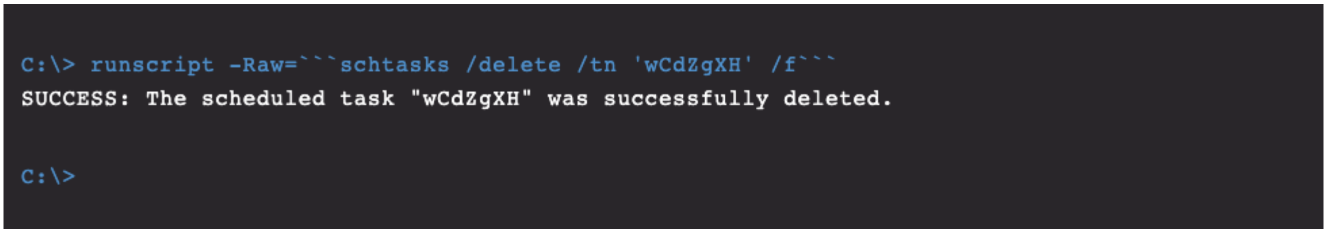


```
  TaskName:                       \wCdZgXH
  Next Run Time:                  7/22/2020 12:20:00 AM
  Status:                         Ready
  Logon Mode:                     Interactive only
  Last Run Time:                  7/21/2020 8:52:34 PM
  Last Result:                    -2147020576
  Author:                         WIN10-PC\admin
> Task To Run:                    C:\WINDOWS\System32\WindowsPowerShell\v1.0\powershell.exe "$windowsupdate = \'C:\Users\auser\AppData\Roaming\Microsoft\Aioygmnayr\snvqzout.exe'
  Start In:                       N/A
  Comment:                        N/A
  Scheduled Task State:           Enabled
  Idle Time:                      Disabled
```

```
schtasks /query /fo list /v | Select-String -Pattern 'AppData' -Context 10
```

Figure 6. Query to find the path for scheduled task persistence mechanism (click image to enlarge)

The parent directory of the QakBot binary is a key indicator of compromise (IOC) and can be used later in the manual remediation process and as an integral search string. In the query shown in Figure 6, the parameter “-context 10” shows the surrounding ten lines and will reveal the actual name of the task that is required for removal. This can simply be deleted with the built-in CMD program schtasks.exe, like so:

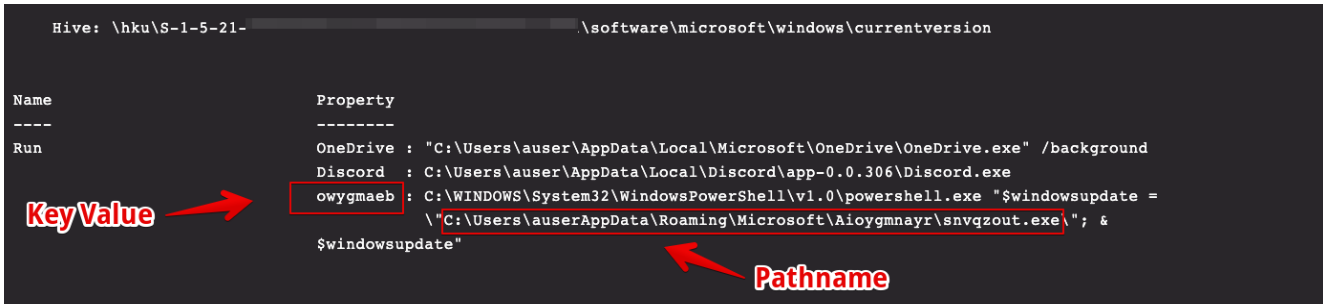


```
C:\> runscript -Raw=```schtasks /delete /tn 'wCdZgXH' /f```
SUCCESS: The scheduled task "wCdZgXH" was successfully deleted.

C:\>
```

Figure 7. Removing the scheduled task (click image to enlarge)

In addition to scheduled tasks, QakBot will often create a registry run key to establish persistence. The modified value is placed under this key name: HKEY_CURRENT_USER\SOFTWARE\Microsoft\Windows\CurrentVersion\Run. Figure 8 shows the output of the query for this registry key.

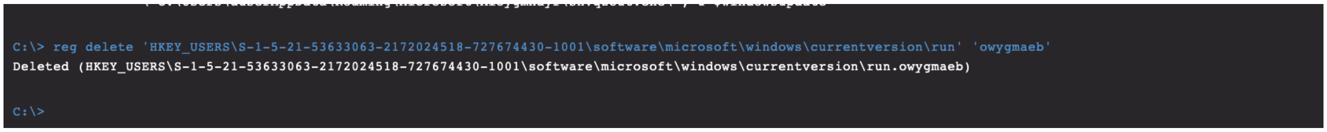


```
    Hive: \hku\S-1-5-21-[redacted]\software\microsoft\windows\currentversion


Name                          Property
----                          --------
Run                           OneDrive : "C:\Users\auser\AppData\Local\Microsoft\OneDrive\OneDrive.exe" /background
                              Discord  : C:\Users\auser\AppData\Local\Discord\app-0.0.306\Discord.exe
                              owygmaeb : C:\WINDOWS\System32\WindowsPowerShell\v1.0\powershell.exe "$windowsupdate =
                                         \"C:\Users\auserAppData\Roaming\Microsoft\Aioygmnayr\snvqzout.exe\"; &
                              $windowsupdate"
```

Figure 8. Registry query (click image to enlarge)

In Figure 8, we obtain the value of the malicious run key and again encounter the path to the main QakBot binary location. Removal of the key is performed with the command shown in Figure 9.



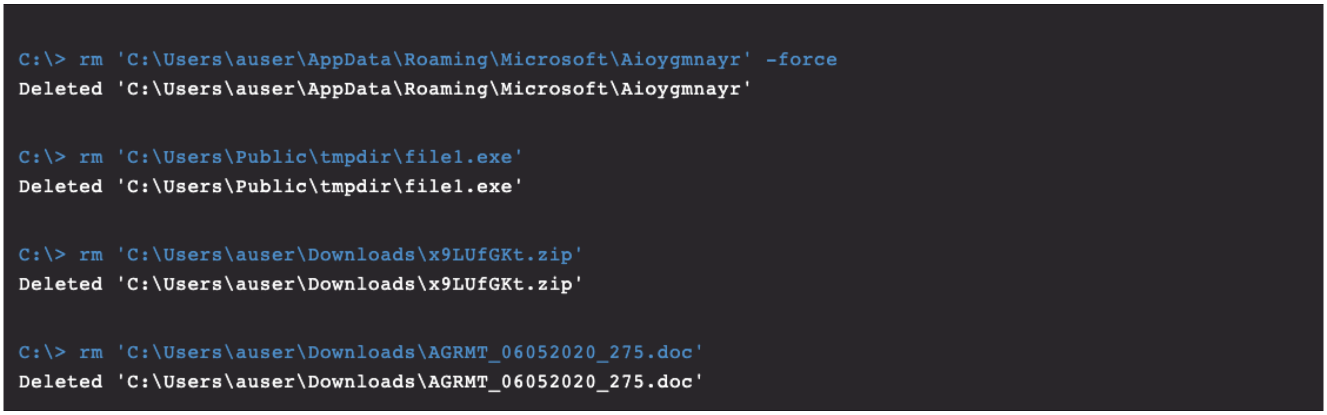
```
C:\> reg delete 'HKEY_USERS\S-1-5-21-53633063-2172024518-727674430-1001\software\microsoft\windows\currentversion\run' 'owygmaeb'
Deleted (HKEY_USERS\S-1-5-21-53633063-2172024518-727674430-1001\software\microsoft\windows\currentversion\run.owygmaeb)

C:\>
```

Figure 9. Removing the run key (click image to enlarge)

### STEP 3. Removing Remaining Artifacts

QakBot leaves file system residue in a few specific locations on infected hosts. The first one is the random alphabetically named folder located in `C:\Users\*\AppData\Roaming\Microsoft`, identified in the remediation steps above. The name of this directory is dynamic, but it will reliably contain QakBot's core binary, .dat files and other resources. This initial download along with the temporary location of the loader's initial execution may also require removal, as shown in Figure 10.



```
C:\> rm 'C:\Users\auser\AppData\Roaming\Microsoft\Aioygmnayr' -force
Deleted 'C:\Users\auser\AppData\Roaming\Microsoft\Aioygmnayr'

C:\> rm 'C:\Users\Public\tmpdir\file1.exe'
Deleted 'C:\Users\Public\tmpdir\file1.exe'

C:\> rm 'C:\Users\auser\Downloads\x9LUfGKt.zip'
Deleted 'C:\Users\auser\Downloads\x9LUfGKt.zip'

C:\> rm 'C:\Users\auser\Downloads\AGRMT_06052020_275.doc'
Deleted 'C:\Users\auser\Downloads\AGRMT_06052020_275.doc'
```

Figure 10. Removal of residual file system artifacts (click image to enlarge)

The steps outlined above are the general process for identifying QakBot artifacts and successfully remediating a host for the QakBot malware. This process is quite effective in singular instances, but in reality this is rarely the case due to QakBot's lateral movement capabilities. A properly configured Falcon platform is a critical component of a successful defense strategy to defeat this threat.

## Conclusion

QakBot has undergone a resurgence in both its delivery volumes and technical evolution. The potential impact from a successful QakBot infection includes (but is certainly not limited to) widespread lateral infections, theft of confidential data, deployment of secondary payloads and loss of organizational prestige.

Remediating these types of infections becomes more complicated with these variants' ability to spread laterally to many hosts. Furthermore, with a transition to a more remote workforce, the capability to remotely remediate infections on hosts that are geographically distributed will become increasingly important as rebuilding systems becomes impractical.

The Falcon Complete team will continue to track this threat and monitor our clients' environments for any notable developments. Despite QakBot's anti-analysis and evasive capabilities, the CrowdStrike Falcon platform prevents this malware from completing its execution chain when it detects the VBScript execution. The Falcon Complete team deals with threats like this every day, providing our customers with the expertise required to remediate these infections and help organizations recover from potentially devastating incidents.

## Appendix

Table 1 below contains a mapping of QakBot tactics to the MITRE ATT&CK$^{®}$ framework.

| Tactic | Technique | Sub-Technique | ID |
|---|---|---|---|
| Initial Access | Phishing | Spear-Phishing Attachment | T1566.001 |
| Execution | User Execution | Malicious Link, Malicious File | T1204.001, T1204.002 |
| Execution | Command and Scripting Interpreter | PowerShell, CMD Shell, Visual Basic | T1059.001, T1059.003, T1059.005 |
| Execution | Signed Binary Proxy Execution | Msiexec, Rundll32 | T1218.007, T1218.011 |
| Persistence | Boot or Logon Autostart Execution | Registry Run Keys / Startup Folder | T1547.001 |
| Persistence | Scheduled Task/Job | Scheduled Task | T1053.005 |
| Defense Evasion | Obfuscated Files or Information | None | T1027 |
| Defense Evasion | Process Injection | Dynamic-link Library Injection | T1055.001 |
| Defense Evasion | Virtualization/Sandbox Evasion | System Checks | T1497.001 |
| Discovery | Virtualization/Sandbox Evasion | User Activity Based Checks | T1497.002 |
| Discovery | Network Share Discovery | None | T1135 |
| Credential Access | Brute Force | Password Guessing | T1110.001 |
| Lateral Movement | Remote Services | SMB/Windows Admin Shares | T1021.002 |
| Command and Control | Application Layer Protocol | Web Protocols | T1071.001 |

Table 1. MITRE ATT&CK mapping

**IOCs associated with QakBot analyses are available in Table 2.**

| Indicator | Purpose |
|---|---|

| | |
|---|---|
| PicturesViewer.dll, PicturesViewer.exe, PaintHelper.dll, PaintHelper.exe, file1.exe | QakBot binary names |
| “[0-9]{6,9}\.zip”, “NUM_[0-9]{4,6}\.vbs” | Regular expression of observed filename convention of zip archives containing VBS that launches QakBot downloader |
| “dfPEZd”, “ezQVN”, “wCdZgXH” | Scheduled Task tasknames |
| “””C:\windows\System32\WScript.exe””<br>“”C:\Users\*\AppData\Local\Temp\Temp1_*.zip\NUM_*.vbs”” | Command line example of initial execution |
| C:\Users\*\Downloads\“[0-9]{6,9}\.zip” | Initial QakBot download path. Observed as an 8 or 9-character numeric name. |
| C:\Users\*\AppData\Local\Temp\Temp1_“[0-9]{6,9}\.zip\NUM_[0-9]{4,6}\.vbs | Execution path of VB downloader script |
| %AppData%\lwob\esexydry.dll<br>%AppData%\PicturesViewer.dll<br>%APPDATA%\dasfdsfsdf.exe<br>%APPDATA%\lwhoq\pozypua.dll<br>%APPDATA%\IE\GGYJG27Z\dasfdsfs.df[1].exe<br>C:\Users\Public\tmpdir | QakBot binary paths in home directories, observed as a alphabetical name under an alphabetical folder in %AppData%, or pre-named PicturesViewer, PaintHelper |
| HKEY_CURRENT_USER\SOFTWARE\Microsoft\Windows\CurrentVersion\Run | Registry run key persistence |

Table 2. IOCs associated with QakBot

## Additional Resources

- *Read "Duck Hunting with Falcon Complete: Analyzing a Fowl Banking Trojan, Part 1" and "Duck Hunting with Falcon Complete: A Fowl Banking Trojan Evolves, Part 2" in this series.*
- *Find out how CrowdStrike can help your organization answer its most important security questions: Visit the CrowdStrike Services webpage.*
- *Learn how any size organization can achieve optimal security with Falcon Complete by visiting the product webpage.*
- *Learn more about Falcon X™ threat intelligence by visiting the webpage.*
- *Learn about CrowdStrike's comprehensive next-gen endpoint protection platform by visiting the Falcon products webpage.*
- *Test CrowdStrike next-gen AV for yourself: Start your free trial of Falcon Prevent™.*