Duck Hunting with Falcon Complete: A Fowl Banking Trojan Evolves, Part 2

By The Falcon Complete Team

Archived: 2026-04-05 17:54:05 UTC

This blog is Part 2 of a three-part blog series detailing the reemergence and evolution of QakBot in the spring and summer of 2020. In this installment we cover analysis of the QakBot ZIP-based delivery campaign, particularly one example that exhibited a tactical breakdown by the threat actor with a botched downloader. In addition, the CrowdStrike® Falcon Complete™ team will cover dynamic analysis of two experimental campaigns, one of which also includes an additional stage-two malware, Zloader.

Threat Background and Context

As discussed in Part 1, QakBot is an eCrime banking trojan that has the potential to severely impact an organization's ability to operate. QakBot has the ability to spread laterally throughout a network utilizing a worm-like functionality through brute forcing network shares, brute forcing Active Directory user group accounts or via server message block (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 this malware from completing its execution chain.

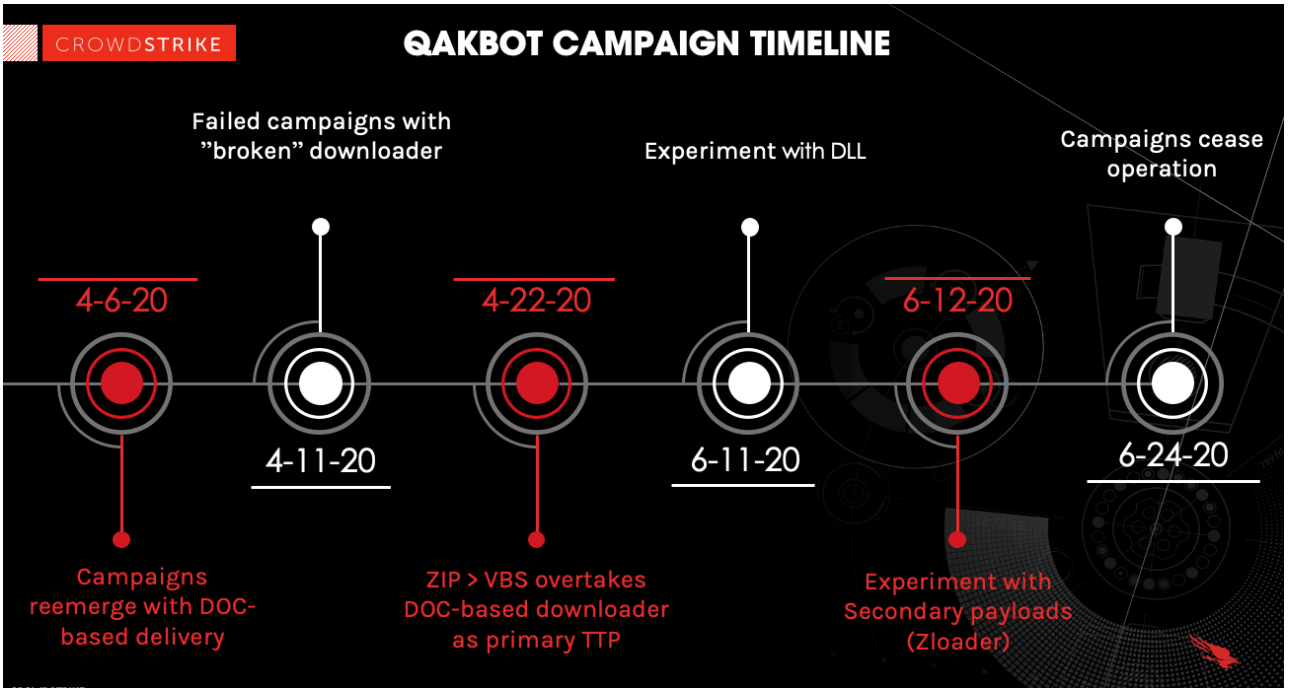

Figure 1. Timeline of QakBot Campaigns (click image to enlarge)

Failed Campaigns with "Broken" Downloader — Late April

In early-to-mid-April 2020, Falcon Complete identified an attempted, unsuccessful QakBot campaign. This was possibly an example of a failed development cycle during the threat actor's retooling efforts. This campaign diverged from the tactics, techniques and procedures (TTPs) observed in the prior, DOC-based campaign. Instead, the delivery tactic includes a .ZIP

attachment containing a malicious Visual Basic Script (VBS) dropper; however, due to a failure in error handling within the .VBS, the actor was unable to download and write the payload to disk successfully.

```
1036    jaJMu.open
1037    Scoinm = 459 - 19 - 16 + 3 + 243 + 337 - 8 - 337 + 88
1038    pejUb = jkkcSf - eJLQb
1039    EgHaP = pejUb - eJLQb
1040
1041    jaJMu.write PIjyiet
```

Figure 2. The portion of the dropper responsible for writing QakBot to disk (click image to enlarge)

In this instance, the VBS dropper would reach out to an array of sites, beginning with hxxp<:>//millionsawesomeproducts<.>com, in an attempt to download *444444.png* from the distribution server — which is actually a Windows PE file — and would subsequently be written to disk with the name *PaintHelper.exe*.

12:16:46	millionsawesomeproducts.com	FOUND
12:16:46	common-factor.nl	FOUND
12:16:46	funpartyrent.com	FOUND
12:16:46	leukkado.be	FOUND
12:16:46	d.teamworx.ph	FOUND
12:16:46	d.teamworx.ph	FOUND

Figure 3. The malware distribution sites associated with the failed campaign (click image to enlarge)

If a 404 error was returned, the script would move on to the next site in the array, and attempt to download the malware and place it into the user's *%TEMP%* directory. The script would then proceed along the kill chain with a persistence mechanism in the form of a scheduled task with a GUID-based naming convention. The site in question did not have a DNS record published — and thus, no HTTP response was returned because no connection could be made. Due to an apparent failure in error handling in the QakBot dropper, the script would fail to infect the host and write a zero-byte, innocuous *PaintHelper.exe to %TEMP%*. It was discovered that via spoofing a DNS response and the subsequent 404 from the site, the QakBot loader would move on to the next site and successfully write its payload — *PaintHelper.exe*. Additionally, the dropper would encode the antivirus product in use, the current OS version and other system information into a Base64 string, and utilize GET strings to inform the malware distribution servers of this information, as shown in Figure 4.

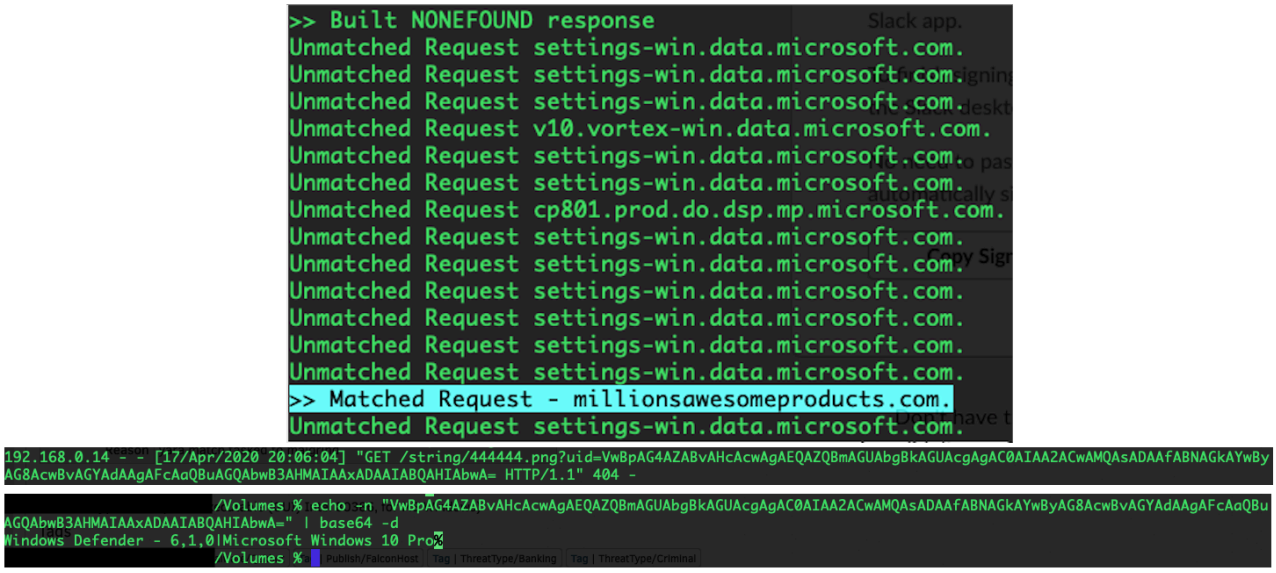

Figure 4. Spoofing the DNS response and subsequent 404 to un-break the dropper; decoded Base64 sent as a GET parameter to the server (click image to enlarge)

This appears to be an implementation of hashbusting — a method of obfuscation in which a malware sample is subtly changed on the fly so each sample has a different checksum. As a result, the SHA256 hash of each payload downloaded from the sites in question appeared to be unique. However, the SSDEEP fuzzy hash of this sample was as follows: *6144:y2la96gEZbXtD/uY/HmJV8cc0em/wnXPKYGvZxYney3brNLFDPMTJYhr64Fgw:y2JvZbJYRwnXPKvZxYn7hLFPMdV4Fgw*
Unfortunately, the actors behind QakBot quickly recovered, and a new campaign was launched the week of May 25, 2020, with no such mistakes and far more success. Despite this success, the operators continued development and experimentation with additional delivery tactics as their campaigns persisted.

QakBot Experiments with PicturesViewer.dll and Secondary Payloads

The following dynamic analysis was conducted on two iterations of QakBot observed and blocked within client environments. During mid-June, the actors behind the QakBot malware experimented with two disparate tactics that each only lasted one day.

- The first anomalous TTP payload was on June 11, 2020. QakBot would drop a Windows PE named *PicturesViewer.dll*, which differs from the *PicturesViewer.exe* seen in prior weeks and noted in the previous section.
- The following day on June 12, QakBot downloaded and executed an additional Windows PE named senate.m4a. Once executed, this binary would install additional malware from the ZeuS family known as Zloader, or Zbot.

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 above would prevent the activity presented here.

Dynamic Analysis of QakBot — June 11

On June 11, QakBot reemerged with new tactics. Beginning at approximately 13:47 UTC, Falcon Complete observed QakBot threat actors using a new .VBS payload. Once a user invokes this script, the process tree is as follows.

1. Wscript makes subsequent DNS requests for a Stage Two payload
2. ‘Rundll32.exe -> PicturesViewer.dll, DllRegisterServer,’ allowing for C2 communication
3. MSIExec, spawning multiple Cmd.exe processes and Explorer injection

4. Commands run by the C2:

- - Cmd.exe /c net view /all /domain
  - Cmd.exe /c net view /all
  - Cmd.exe /c net config workstation
  - Cmd.exe /c ipconfig /all

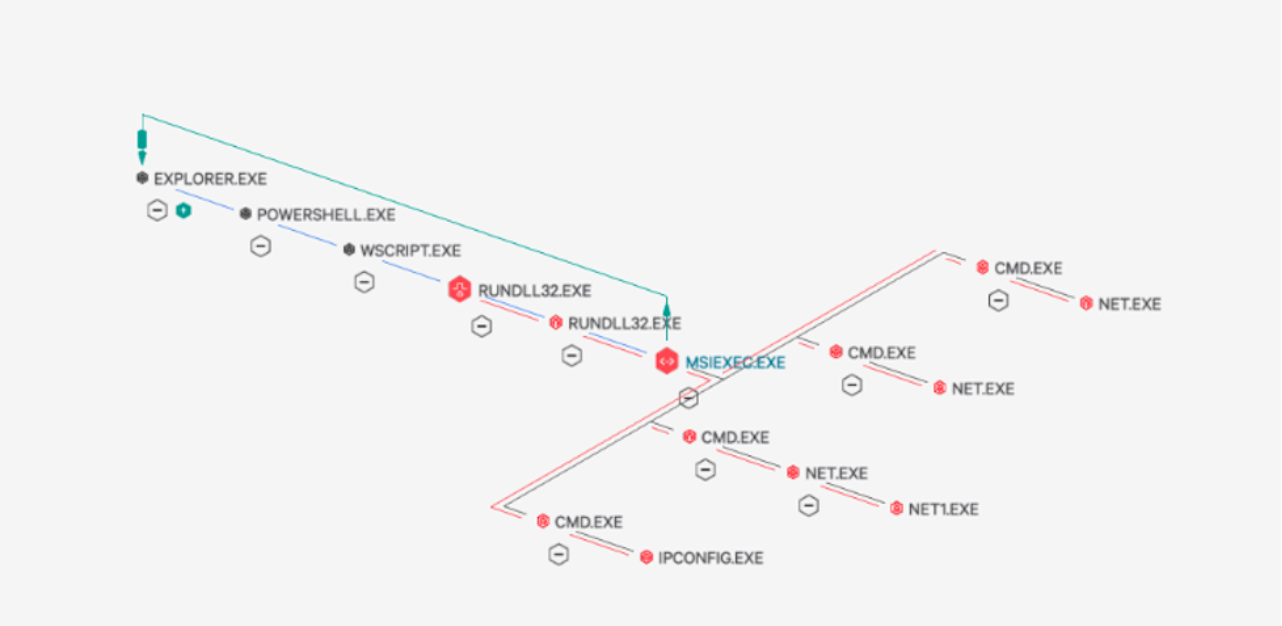

Figure 5. Process tree of QakBot as displayed in Falcon (click image to enlarge)

If we take a closer look at Wscript.exe, we can see the process reaches out to four suspicious-looking domains. Further analysis of these four domains reveals an HTTP GET request for the following payloads.
hxxp<:>ameliasmoments<.>com/wp-includes/js/thickbox/wifgyfro/8888888.jpg hxxp<:>digitalschoolfaridabad<.>in /courses/images/parallax/mjogqxakfxg/8888888.jpg hxxp<:>Uniquehindunames<.>com/ wp-content/uploads/cnesco/8888888.jpg hxxp<:>Sometechsense<.>com/ wp-includes/jtinymce/plugins/wptextpattern/tbpfdfelf/8888888.jpg

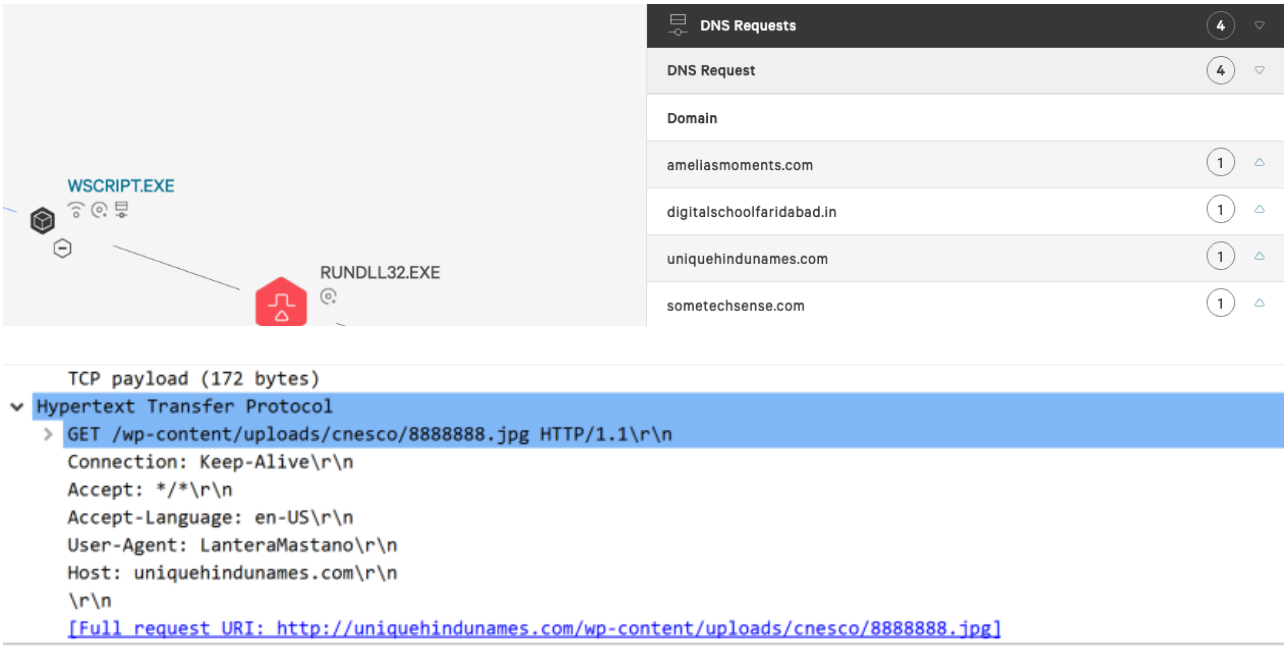

Figure 6. Suspicious DNS requests as displayed in Falcon and Wireshark-captured GET request (click images to enlarge)

Further review of the dynamic behavior of MSIExec.exe shows process injection into Explorer, multiple files written to disk, and multiple DNS requests, as shown below. Files written:

- %AppData%\lwob\esexydry.dll
- %AppData%\PicturesViewer.dll

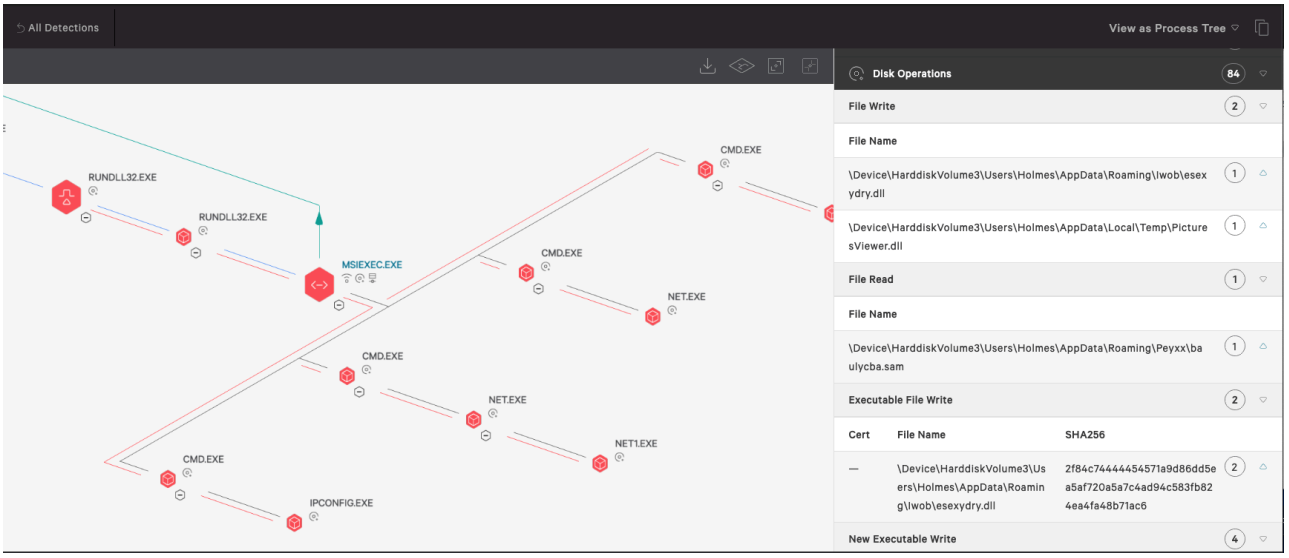

Figure 7. Disk operations as displayed in Falcon (click image to enlarge)

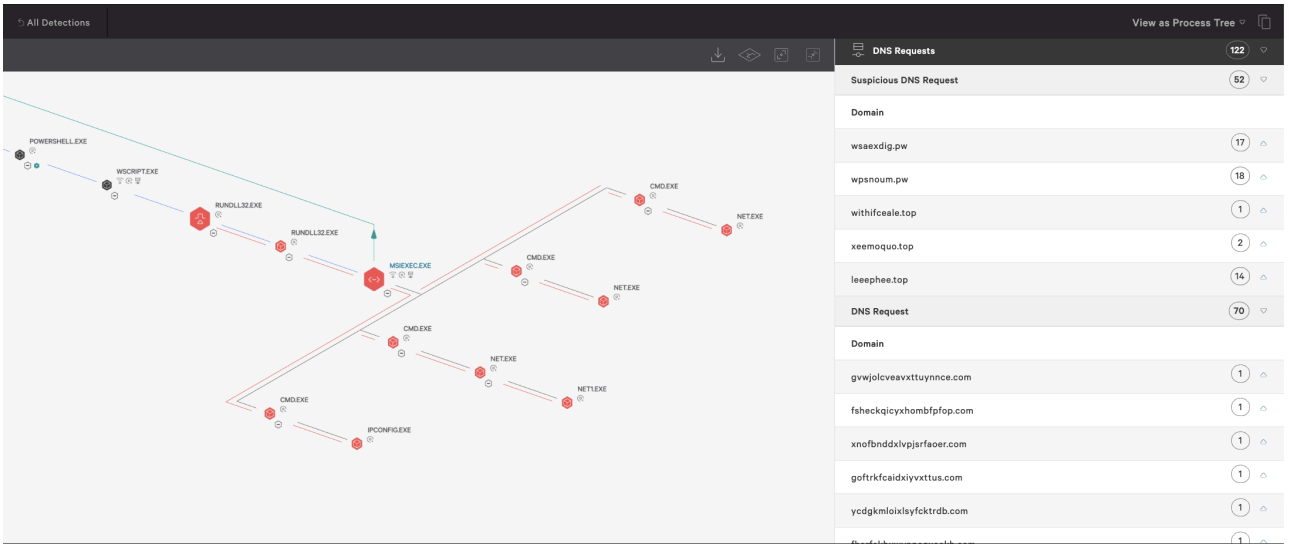

Figure 8. Further DNS requests as displayed in Falcon (click image to enlarge)

Dynamic Analysis of QakBot — June 12

On June 12, QakBot continued its evolution. The delivery method of a .ZIP file to malicious .VBS was the same, but this time QakBot also dropped a Zloader payload on its victim. Beginning around 14:24 UTC, Falcon Complete observed QakBot threat actors using a new .VBS payload. Once the user invoked this script, the process tree is as follows.

1. Wscript.exe
2. WmiPrivSe.exe
3. Rundll32 -> %AppData%\senate.m4a, DllRegisterServer
4. MSIExec, spawning multiple Cmd.exe processes, DNS requests, and writing a few interesting files to disk
5. Commands run by the C2:

-
  -
    - Cmd.exe /c net view /all /domain
    - Cmd.exe /c net view /all
    - Cmd.exe /c net config workstation
    - Cmd.exe /c ipconfig /all

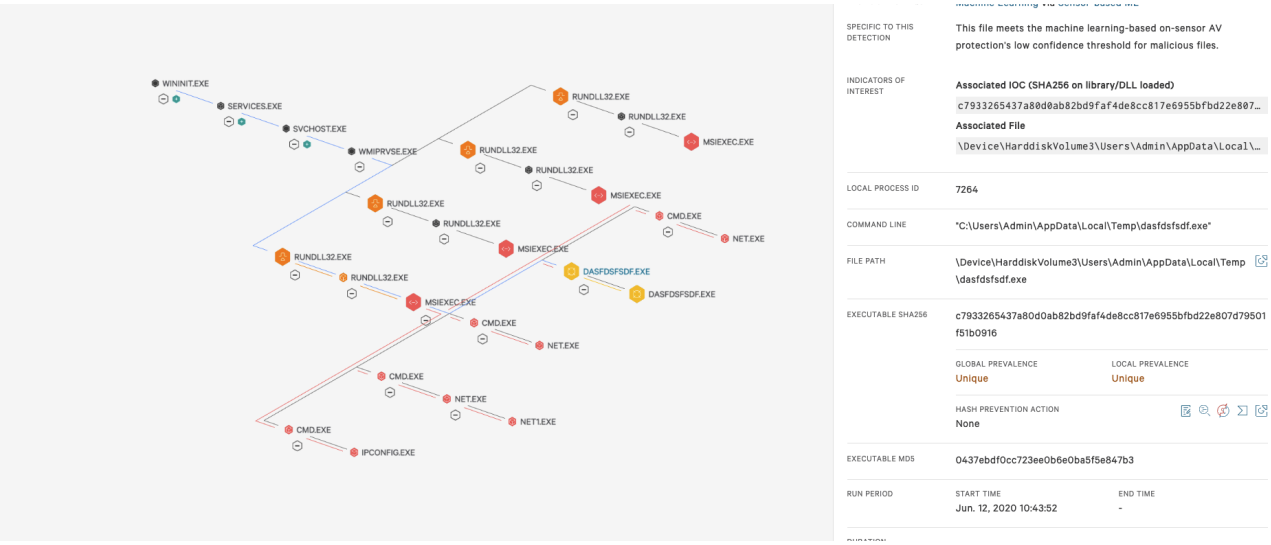

Figure 9. Process tree as displayed in Falcon (click image to enlarge)

Wscript.exe does a number of things: It deletes the original QakBot.vbs and writes four files to disk in %APPDATA% induce.flac, pep.csv, rhythm.tex and senate.m4a. Senate.m4a is deleted after full process execution. Looking closer at Rundll32.exe, we can see that it's executing a senate.m4a, DllRegisterServer. Further analysis shows senate.m4a is actually a Zloader.dll. At the time of this writing, we are no longer seeing the senate.m4a Zloader being distributed by the QakBot .VBS delivery method. Continuing with the dynamic behavior of MSIExec.exe shows multiple files written to disk and multiple DNS requests, as shown below. DNS requests:

- withifceale<.>top/treusparq.php
- xeemoquo<.>top/treusparq.php
- leeephee<.>top/treusparq.php
- cccommercialcleaning<.>com<.>au/wp-content/themes/twentyfifteen/1/spx139/dasfdsfsdf.exe

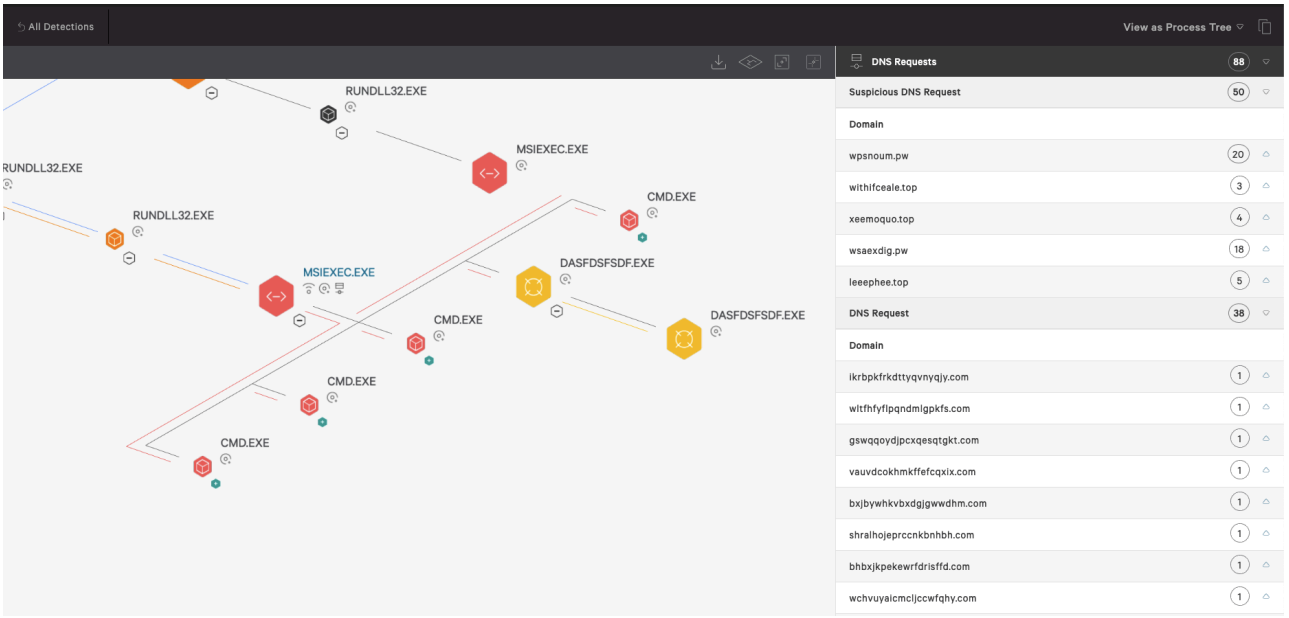

Figure 10. DNS requests as displayed in Falcon (click image to enlarge)

Files written:

- %APPDATA%\dasfdsfsdf.exe
- %APPDATA%\Iwhoq\pozypua.dll
- %APPDATA%\IE\GGYJG27Z\dasfdsfs.df<1>.exe

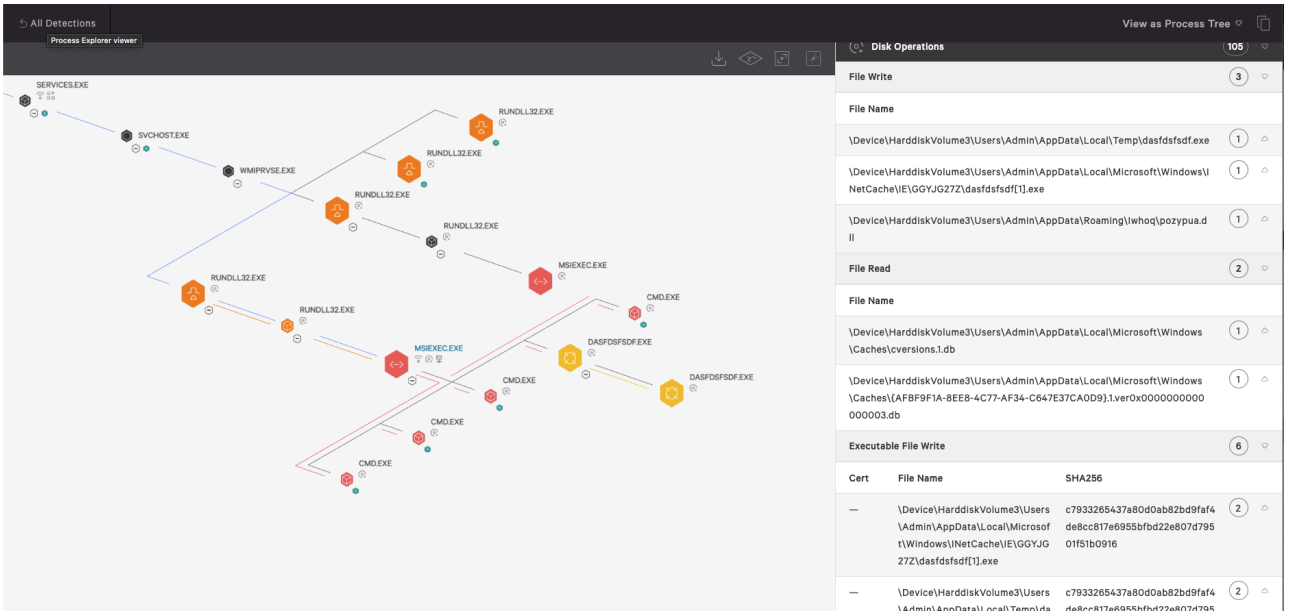

Figure 11. QakBot disk operations as displayed in Falcon (click image to enlarge)

Once the full QakBot and Zloader process execution is complete, persistence may be established through the following registry key: HKEY_CURRENT_USER\SOFTWARE\Microsoft\Windows\CurrentVersion\Run. The name of the value will either be randomly chosen or a GUID, targeting a PE in the %APPDATA%\Roaming\Microsoft\ folder.

Conclusion

As we have seen, QakBot employs a robust set of anti-analysis features and has recently surged in its operational volume within the threat landscape. This blog provided an in-depth analysis of a botched QakBot downloader along with dynamic

analysis of experimental payloads that include a secondary malware stage: Zloader. The threat actors behind QakBot, tracked as MALLARD SPIDER, have demonstrated the ability to rapidly retool, implement anti-analysis techniques and develop methods of advanced obfuscation in a short period. Part 3 of this series will outline the Falcon Complete team's strategy for the remote remediation of a QakBot-infected host.

Additional Resources

- *Read Part 1 of this blog 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 CROWDSTRIKE FALCON® INTELLIGENCE™ threat intelligence by visiting the webpage.*
- *Learn about CrowdStrike's comprehensive next-generation endpoint protection platform by visiting the Falcon products webpage.*
- *Test CrowdStrike next-gen AV for yourself: Start your free trial of Falcon Prevent™.*
