Analysis of Stealer Malware

Authored by: Garrett Blaylock and Will Kittredge — student analysts

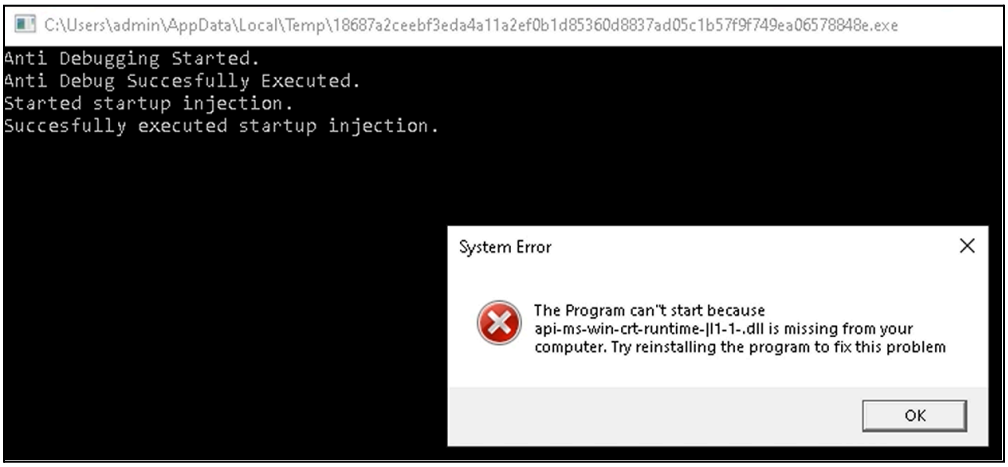
In this post, we analyze stealer malware using ANY.RUN to try to understand its capabilities. Based on our findings, we discuss the potential impact of the malware and suggest some mitigation strategies.

If you want to follow along, you can find more details about the malware and download a sample from the MalwareBazaar database entry linked at the end. Be forewarned, though. Understand the potential dangers and remember to take proper precautions with real malware samples!

Behavior during execution

During execution, the malware seems to keep out of the way. The average user who just opened their laptop for some web browsing would probably be unaware of its presence unless they went looking for it. This would be expected for stealer malware - the less attention it draws to itself, the better. By remaining hidden, it can persist on the system for longer and steal more data.

We do get some indication that something might be amiss when the malware first installs and executes, though.



Popup and `cmd` windows visible at the desktop on startup

In the background, information-gathering child processes are being created. We will examine these later in the Child process section.

Network activity

We can see that the `stub.exe` process made a request to `ip-api.com` to perform an IP lookup. Based on the `User-Agent` field of the request, we infer that the malware is probably using Python. This is consistent with information listed on MalwareBazaar.

Timeshift	Class	PID	Process name	Message
4019 ms	Device Retrieving External IP Address De...	2192	svchost.exe	INFO [ANY.RUN] External IP Check (ip-api .com)
4036 ms	Device Retrieving External IP Address De...	2192	svchost.exe	ET INFO External IP Lookup Domain in DNS Lookup (ip-api .com)
4530 ms	Device Retrieving External IP Address De...	4684	stub.exe	ET POLICY External IP Lookup ip-api.com
5557 ms	Not Suspicious Traffic	2192	svchost.exe	INFO [ANY.RUN] Attempting to access raw user content on GitHub

Request

URL	/json
Method	GET
Host	ip-api.com
Accept	*/*
Accept-Encoding	gzip, deflate
User-Agent	Python/3.10 aiohttp/3.10.5

Response

Status code	200: OK
Date	Sat, 02 Nov 2024 23:33:56 GMT
Content-Type	application/json; charset=utf-8
Content-Length	287
Access-Control-Allow-Origin	*
X-Ttl	60
X-Rl	44

```
00b0  41 63 63 65 73 73 2d 43  6f 6e 74 72 6f 6c 2d 41   Access-C ontrol-A
00c0  6c 6c 6f 77 2d 4f 72 69  67 69 6e 3a 20 2a 0d 0a   llow-Ori gin: *··
00d0  58 2d 54 74 6c 3a 20 36  30 0d 0a 58 2d 52 6c 3a   X-Ttl: 6 0··X-Rl:
00e0  20 34 34 0d 0a 0d 0a 7b  22 73 74 61 74 75 73 22    44····{ "status"
00f0  3a 22 73 75 63 63 65 73  73 22 2c 22 63 6f 75 6e   :"succes s","coun
0100  74 72 79 22 3a 22 47 65  72 6d 61 6e 79 22 2c 22   try":"Ge rmany","
0110  63 6f 75 6e 74 72 79 43  6f 64 65 22 3a 22 44 45   countryC ode":"DE
0120  22 2c 22 72 65 67 69 6f  6e 22 3a 22 42 45 22 2c   ","regio n":"BE",
0130  22 72 65 67 69 6f 6e 4e  61 6d 65 22 3a 22 4c 61   "regionN ame":"La
0140  6e 64 20 42 65 72 6c 69  6e 22 2c 22 63 69 74 79   nd Berli n","city
0150  22 3a 22 42 65 72 6c 69  6e 22 2c 22 7a 69 70 22   ":"Berli n","zip"
0160  3a 22 31 30 31 37 38 22  2c 22 6c 61 74 22 3a 35   :"10178" ,"lat":5
0170  32 2e 35 32 32 32 2c 22  6c 6f 6e 22 3a 31 33 2e   2.5222," lon":13.
0180  34 30 39 33 2c 22 74 69  6d 65 7a 6f 6e 65 22 3a   4093,"ti mezone":
0190  22 45 75 72 6f 70 65 2f  42 65 72 6c 69 6e 22 2c   "Europe/ Berlin",
01a0  22 69 73 70 22 3a 22 4d  32 34 37 20 45 75 72 6f   "isp":"M 247 Euro
01b0  70 65 20 53 52 4c 22 2c  22 6f 72 67 22 3a 22 4d   pe SRL", "org":"M
01c0  32 34 37 20 4c 74 64 20  42 65 72 6c 69 6e 22 2c   247 Ltd  Berlin",
01d0  22 61 73 22 3a 22 41 53  39 30 30 39 20 4d 32 34   "as":"AS 9009 M24
01e0  37 20 45 75 72 6f 70 65  20 53 52 4c 22 2c 22 71   7 Europe  SRL","q
01f0  75 65 72 79 22 3a 22 31  39 33 2e 31 37 36 2e 38   uery":"1 93.176.8
0200  36 2e 34 30 22 7d                                  6.40"}
```

ANY.RUN detected threats (top) and `stub.exe` request/response information (left and right)

(ip.addr == 208.95.112.1 && tcp.port == 80) && (ip.addr == 192.168.100.143 && tcp.port == 49690)

No.	Time	Source	Destination	Protocol	Length	Host	Info
604	6.865425	192.168.100.143	208.95.112.1	TCP	66		49690 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
605	6.965301	208.95.112.1	192.168.100.143	TCP	66		80 → 49690 [SYN, ACK] Seq=0 Ack=1 Win=8190 Len=0 MSS=1330 WS=16 SACK_PERM=1
606	6.965469	192.168.100.143	208.95.112.1	TCP	54		49690 → 80 [ACK] Seq=1 Ack=1 Win=263168 Len=0
607	6.966194	192.168.100.143	208.95.112.1	HTTP	179	ip-api.com	GET /json HTTP/1.1
608	7.066733	208.95.112.1	192.168.100.143	HTTP/J…	518		HTTP/1.1 200 OK , JavaScript Object Notation (application/json)
609	7.068404	192.168.100.143	208.95.112.1	TCP	54		49690 → 80 [FIN, ACK] Seq=126 Ack=465 Win=262656 Len=0
610	7.167292	208.95.112.1	192.168.100.143	TCP	54		80 → 49690 [FIN, ACK] Seq=465 Ack=127 Win=8192 Len=0
611	7.167429	192.168.100.143	208.95.112.1	TCP	54		49690 → 80 [ACK] Seq=127 Ack=466 Win=262656 Len=0

We can download the PCAP containing all of the network traffic and use Wireshark to confirm that a connection was established. The site responded with JSON data.

5240 ms | GET | 404: Not Found | ? | https://raw.githubusercontent.com/justforMonster/injection/main/injection.js | 14 b | text

The malware also attempted to access raw user content on GitHub. The request gets a `404: Not Found` response back, but the URL was suspicious. Based on the URL value, we assumed that this was Javascript to perform some type of malicious injection.

Child processes

The malware spawns child processes that gather information about the system, among other things. In this case `cmd.exe` is being invoked with the `/c` option, which tells the command prompt to exit after the specified command is executed.

ANY.RUN process list, cmd.exe process spawned by stub.exe are visible

Some of these processes establish persistence methods for the malware. For example, we can see that process 2600 ran a command that creates a scheduled task to automatically run the malware. Process 396 added a key to the registry to make the malware a startup program. Other processes could reveal potentially sensitive information. Process 2356, for example, executes the `Get-Clipboard` PowerShell cmdlet to get the contents of the clipboard. Passwords and personal information might be stored here depending on the victim user's prior activity that day.

A de-obfuscated PowerShell command leads us to believe that this malware is also able to take screenshots of the infected system.

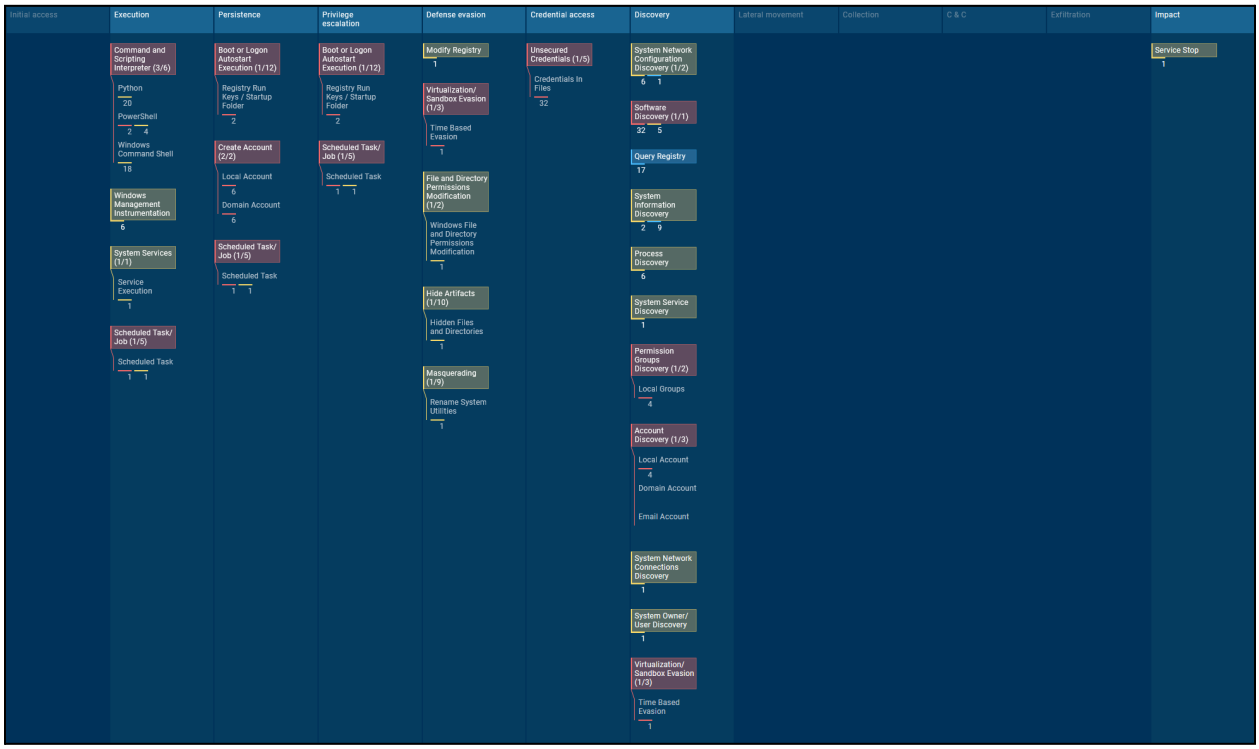
```
powershell.exe -NoProfile -ExecutionPolicy Bypass -EncodedCommand WwBS
AGUAZgBsAGUAYwB0AGkAbwBuAC4AQQBzAHMAZQBtAGIAbAB5AF0AOgA6
AEwAbwBhAGQAVwBpAHQAaABQAGEAcgB0AGkAYQBsAE4AYQBtAGUAKAAi
AFMAeQBzAHQAZQBtAC4ARAByAGEAdwBpAG4AZwAiACkADQAKAGYAdQBu
AGMAdABpAG8AbgAgAHMAYwByAGUAZQBuAHMAaABvAHQAKABbAEQAcg
BhAHcAaQBuAGcALgBSAGUAYwB0AGEAbgBnAGwAZQBdACQAYgBvAHUAbg
BkAHMALAAgACQAcABhAHQAaAApACAAewANAAoAIAAgACAAJABiAG0AcA
AgAD0AIABOAGUAdwAtAE8AYgBqAGUAYwB0ACAARAByAGEAdwBpAG4AZw
AuAEIAaQB0AG0AYQBwACAAJABiAG8AdQBuAGQAcwAuAHcAaQBkAHQAaA
AsACAAJABiAG8AdQBuAGQAcwAuAGgAZQBpAGcAaAB0AA0ACgAgACAAIAA
kAGcAcgBhAHAAaABpAGMAcwAgAD0AIABbAEQAcgBhAHcAaQBuAGcALgB
HAHIAYQBwAGgAaQBjAHMAXQA6ADoARgByAG8AbQBJAG0AYQBnAGUAKAA
kAGIAbQBwACkADQAKAA0ACgAgACAAIAAkAGcAcgBhAHAAaABpAGMAcwA
uAEMAbwBwAHkARgByAG8AbQBTAGMAcgBlAGUAbgAoACQAYgBvAHUAbgB
kAHMALgBMAG8AYwBhAHQAaQBvAG4ALAAgAFsARAByAGEAdwBpAG4AZw
AuAFAAbwBpAG4AdABdADoAOgBFAG0AcAB0AHkALAAgACQAYgBvAHUAbg
BkAHMALgBzAGkAegBlACkADQAKAA0ACgAgACAAIAAkAGIAbQBwAC4AUwB
hAHYAZQAoACQAcABhAHQAaAApAA0ACgANAAoAIAAgACAAJABnAHIAYQB
wAGgAaQBjAHMALgBEAGkAcwBwAG8AcwBlACgAKQANAAoAIAAgACAAJABi
AG0AcAAuAEQAaQBzAHAAbwBzAGUAKApAA0ACgB9AA0ACgANAAoAJABi
AG8AdQBuAGQAcwAgAD0AIABbAEQAcgBhAHcAaQBuAGcALgBSAGUAYwB0
AGEAbgBnAGwAZQBdADoAOgBGAHIAbwBtAEwAVABSAEIAKAAwACwAIAAw
ACwAIAAxADkAMgAwACwAIAAxADAAOAAwACkADQAKACQAcABhAHQAaAA
gAD0AIAAoAEcAZQB0AC0ASQB0AGUAbQAgAC4AKQAuAEYAdQBsAGwATgBh
AG0AZQArACIAXABzAGMAcgBlAGUAbgBzAGgAbwB0AC4AcABuAGcAIgANA
AoAcwBjAHIAZQBlAG4AcwBoAG8AdAAgACQAYgBvAHUAbgBkAHMAIAAkAH
AAYQB0AGgA
```

Base64 encoded command

```
[Reflection.Assembly]::LoadWithPartialName("System.Drawing")
function screenshot([Drawing.Rectangle]$bounds, $path) {
   $bmp = New-Object Drawing.Bitmap $bounds.width, $bounds.height
   $graphics = [Drawing.Graphics]::FromImage($bmp)

   $graphics.CopyFromScreen($bounds.Location, [Drawing.Point]::Empty, $bounds.size)

   $bmp.Save($path)

   $graphics.Dispose()
   $bmp.Dispose()
}

$bounds = [Drawing.Rectangle]::FromLTRB(0, 0, 1920, 1080)
$path = (Get-Item .).FullName+"\screenshot.png"
screenshot $bounds $path
```

Decoded Base64

MITRE ATT&CK



MITRE ATT&CK chart

The malware's tactics align with several MITRE ATT&CK techniques, including command-line execution (T1059.001), registry key modifications for persistence (T1547.001), and standard application layer protocols for command and control communication (T1071.001). The MITRE ATT&CK chart generated by ANY.RUN helps us get a more complete picture of what the malware is doing.

Command and scripting interpreters such as Python, PowerShell, and the Windows command prompt are leveraged by the malware in execution. Using persistence methods like autostart and scheduled tasks, the malware ensures that it will be executed when the victim uses their machine. Once running, it attempts to gather information from the system and access credentials while simultaneously hiding itself. The stealer malware also appears to be virtualization-aware, as ANY.RUN detected time-based virtualization/sandbox evasion.

Indicators of Compromise (IOCs)

As part of the information-gathering processes started by the malware, it created network requests that ANY.RUN detected as potential indicators of compromise. These consist of a network connection that the `stub.exe` process made to the `restores.name` domain (IP `135.181.65.219`) and the failed attempt to access the `injection.js` file stored on GitHub. The network communication with `ip-api.com` was likely for geolocation and reconnaissance purposes.

Obfuscation techniques were also identified, with Base64-encoded PowerShell commands used to execute scripts, possibly for capturing screenshots of the victim's system. These IOCs provide critical insights for defenders to recognize and respond to the malware's activity effectively.

Key observations

The analyzed malware exhibits stealthy behavior typical of stealer malware, designed to operate quietly in the background to avoid drawing attention. While it generally remains unobtrusive, initial execution reveals some activity through visible popups and cmd windows. This behavior might hint at an issue to attentive users but would likely go unnoticed by most. The malware engages in extensive information gathering by creating child processes that collect system data, and its capabilities include executing PowerShell commands, potentially for capturing screenshots.

Persistence mechanisms are a notable aspect of this malware's functionality, with evidence of registry modifications and scheduled tasks ensuring automatic execution upon system startup. This allows it to maintain access and carry out operations without requiring user intervention. Additionally, the malware communicates with external servers, including ip-api.com, to gather reconnaissance data, leveraging HTTP requests to interact with these endpoints. The User-Agent field in the request suggests that it operates using Python, aligning with information from the MalwareBazaar database. Collectively, these observations indicate that the malware could be used for credential theft, espionage, or data exfiltration, making it a significant threat to affected systems.

Recommendations

To mitigate the risks posed by this malware, organizations should implement a multi-layered approach to cybersecurity. Robust endpoint protection solutions, such as Endpoint Detection and Response (EDR) tools, should be deployed to monitor and block unauthorized activities, including child process creation, registry modifications, and suspicious network communications. Complementing this, user awareness training is essential to reduce the likelihood of infection, particularly by educating users on the dangers of interacting with unknown attachments, links, or files.

Network monitoring should also be a priority, using firewalls and intrusion detection/prevention systems (IDS/IPS) to block communication with known malicious domains or IP addresses, such as ip-api.com. An incident response plan should be in place to address infections swiftly, encompassing steps for isolating affected systems, removing malware, and recovering data. Finally, keeping operating systems and software regularly updated is vital to minimize vulnerabilities that the malware could exploit. By combining these strategies, organizations can effectively reduce the risks and impact of malware attacks.

Discussion

This malware demonstrates the sophisticated yet stealthy behavior typical of modern stealer malware. By blending in with legitimate processes and establishing persistence through registry and scheduled task modifications, the malware maximizes its chances of success. The observed communication with external servers underscores its reconnaissance capabilities, likely aimed at tailoring its operations to the infected system's environment.

The use of PowerShell commands for potential screenshot capturing further highlights the flexibility of this malware. Combined with its persistence techniques, it poses a significant threat to individual users and organizations, especially those lacking advanced security measures. While its exact payload and objectives remain unclear, the potential for credential theft or sensitive data exfiltration is evident.

Conclusion

Analyzing this stealer malware provides critical insights into its techniques and impact. From initial infection to establishing persistence, its methods highlight the importance of proactive cybersecurity measures. Advanced tools like ANY.RUN make it easier to dissect such threats and understand their implications.

Organizations must prioritize defense-in-depth strategies to mitigate the risks posed by malware of this nature. A combination of user education, advanced monitoring tools, and effective response protocols can reduce exposure and improve resilience against similar threats.

For further details on this malware, including its behavior and characteristics, refer to the provided ANY.RUN task links. Understanding and preparing for these threats is essential in an ever-evolving cybersecurity landscape.

More links

MalwareBazaar database entry:
https://bazaar.abuse.ch/sample/18687a2ceebf3eda4a11a2ef0b1d85360d8837ad05c1b57f9f749ea06578848e/

Public (not ours) ANY.RUN task:
https://app.any.run/tasks/f71ce34e-fadf-4f68-8512-b132d935e46d/

This report also contains some useful information about the malware sample:
https://tria.ge/241102-3gj3aayell

Exela Stealer GitHub page (detected by ANY.RUN):
https://github.com/quicaxd/Exela-V2.0