NetSupport RAT hits again with new IOCs

By Ariel Davidpur

Published: 2024-01-23 · Archived: 2026-04-05 21:41:37 UTC

5 min read

Jan 23, 2024

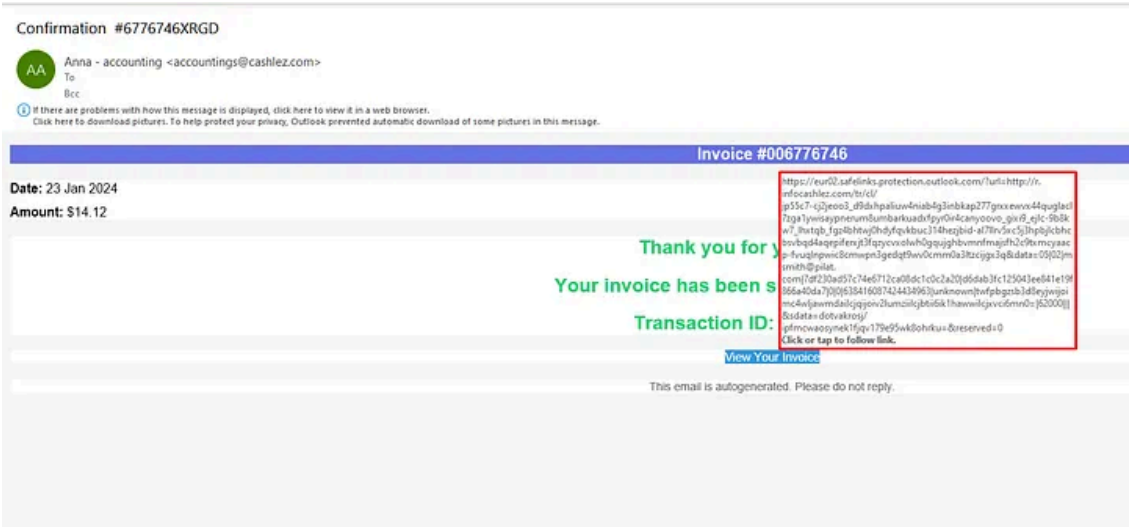
Summary

There is a new campaign from the last day that managed to bypass EDRs while using Phishing emails to distribute the malware with Safelinks protection of Microsoft to redirect to different sites that host the malware itself.

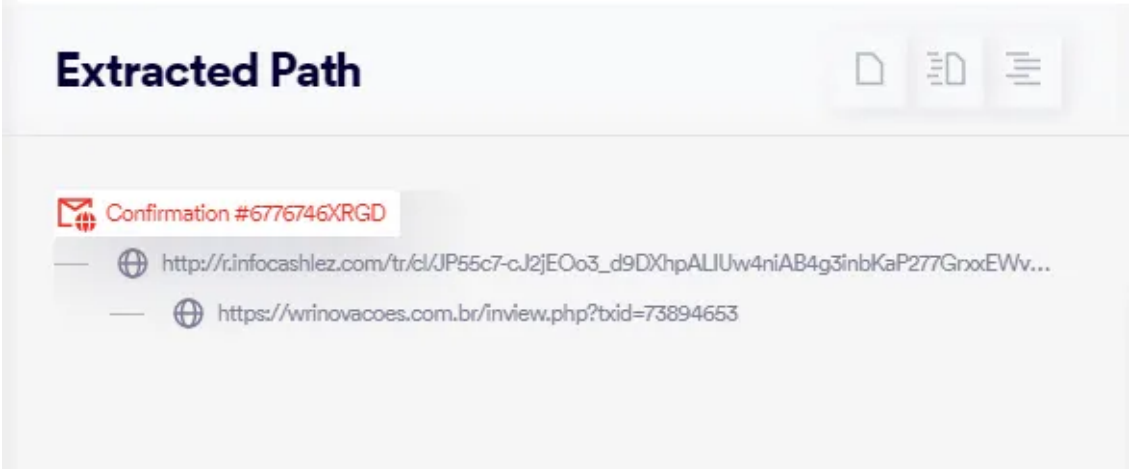
How it’s done?

The attacker mass distributes the malware via email and uses legitimate services to bypass email protection systems while targeting the Technology sector companies.

Press enter or click to view image in full size



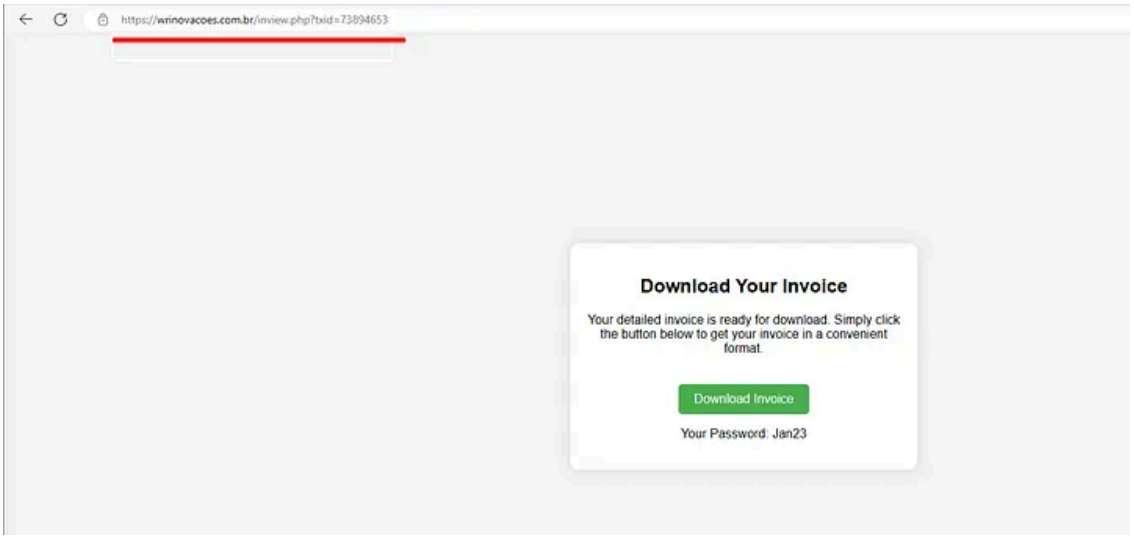
The email with the Safelink protection



The URL redirection tree after the safelinks

As soon as the victim gets the email and clicks on *"View Your Invoice"* the victim will be redirected to another site that leads to click on the *"Download Invoice"* button.

Press enter or click to view image in full size



the download page

As soon as the victim clicks on the button he will be redirected to *"googleuserconetnt.com"* which uses open-redirect to the actual website that it gets the malware from

Press enter or click to view image in full size

Google user content redirect

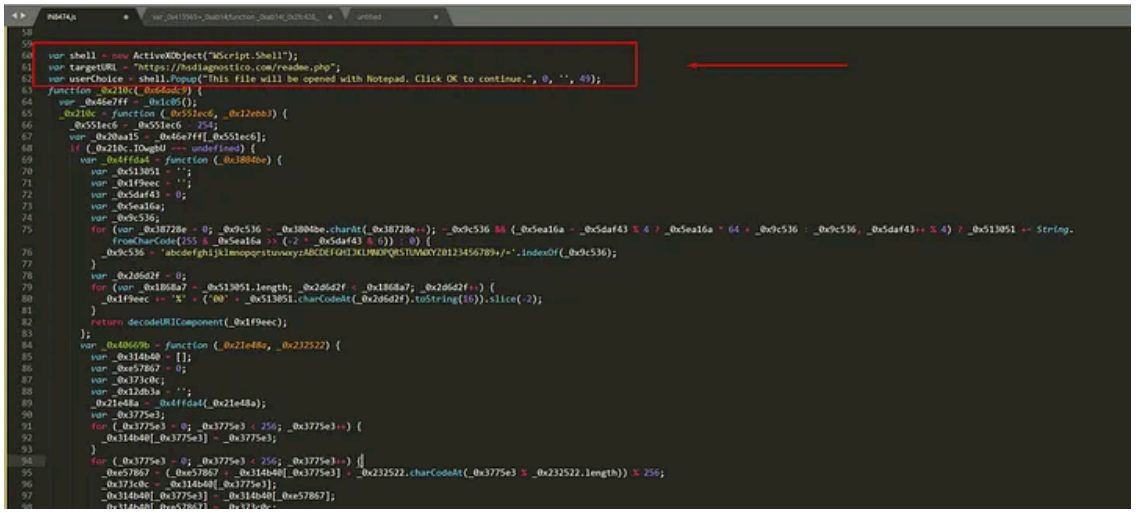
While observing the JS file that was downloaded, we noticed that the code is obfuscated by a generic JS obfuscator (Obfuscate.io).

Press enter or click to view image in full size

obfuscated code

After making the code clear enough to understand and debug, we can see the actual steps that are being made by the JS code, and what are the C2s (Target URL).

Press enter or click to view image in full size



deobfuscated code

Diving deeper…

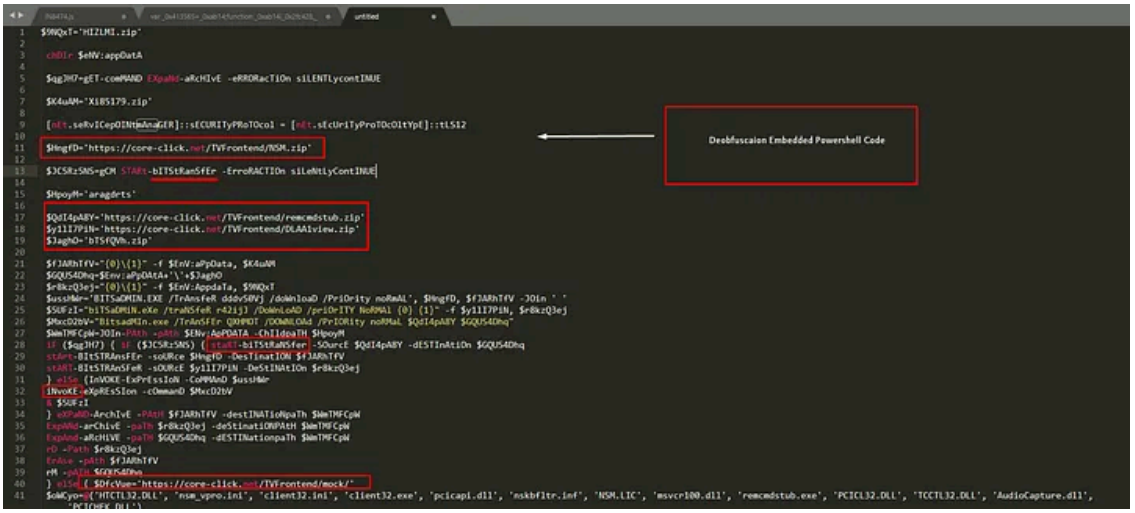
While the malware is being executed, it's using the "targetURL" to create an object that runs in "Wscript.shell", and runs a hidden encoded command.

Press enter or click to view image in full size

encoded PS code

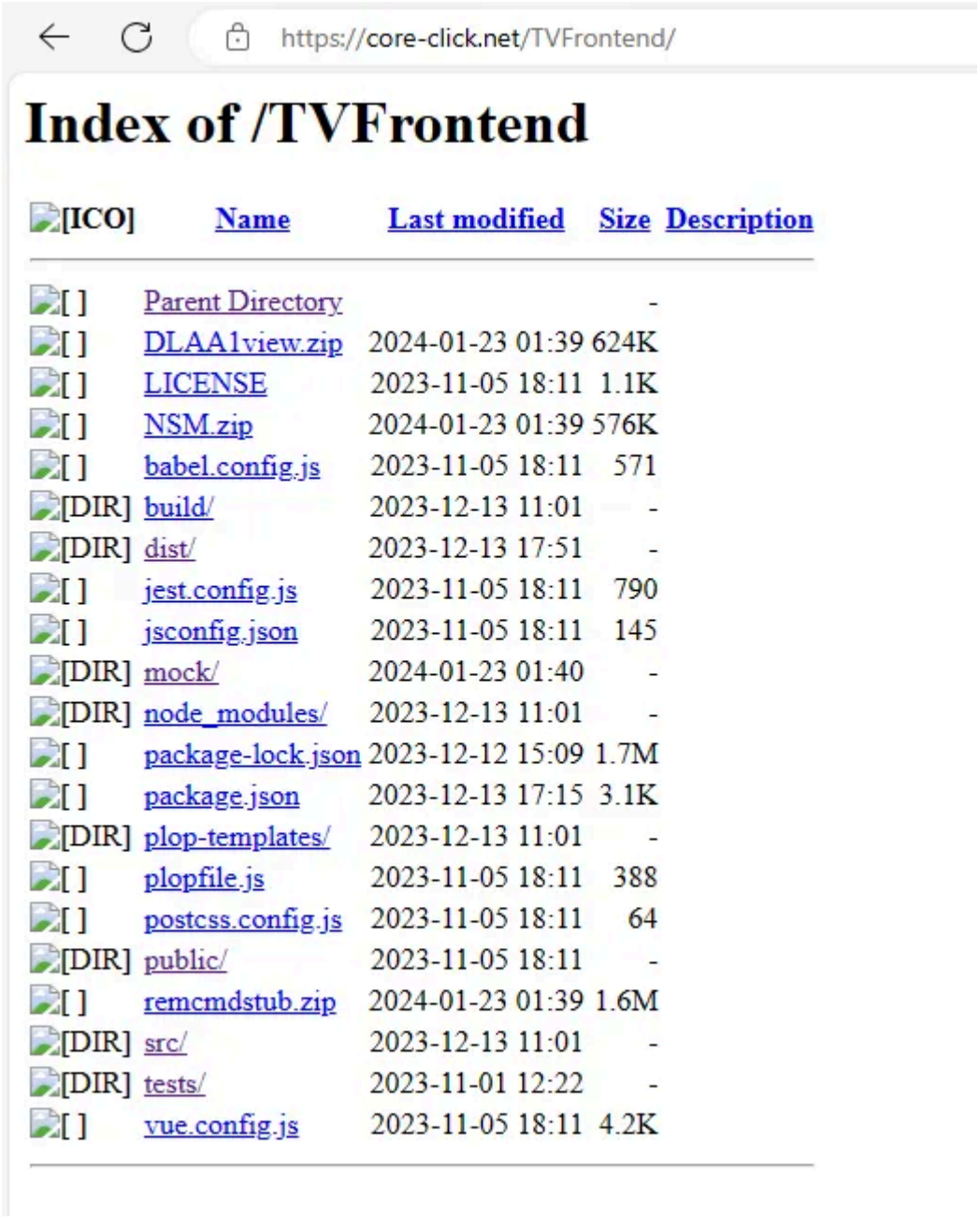
We have decoded and cleared the PS code that was in the URL, and now we can see more files that were being downloaded, where they were stored, and under what names.

Press enter or click to view image in full size



decoded PS

Now we can see that it has multiple ZIP files that contain multiple files and how it uses “start-bitstransfer” to retrieve them from the C2 server to the client(victim’s machine).
We can also see that all of the files were retrieved from the same domain, and while browsing into it we could find an open directory.

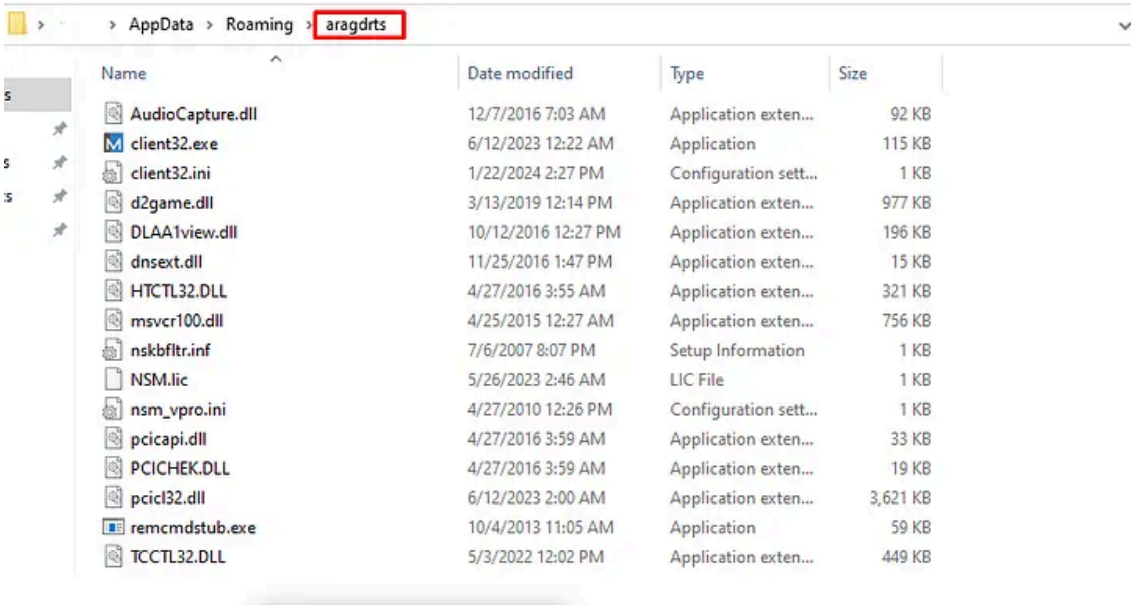


C2 open dir

Execution Flow

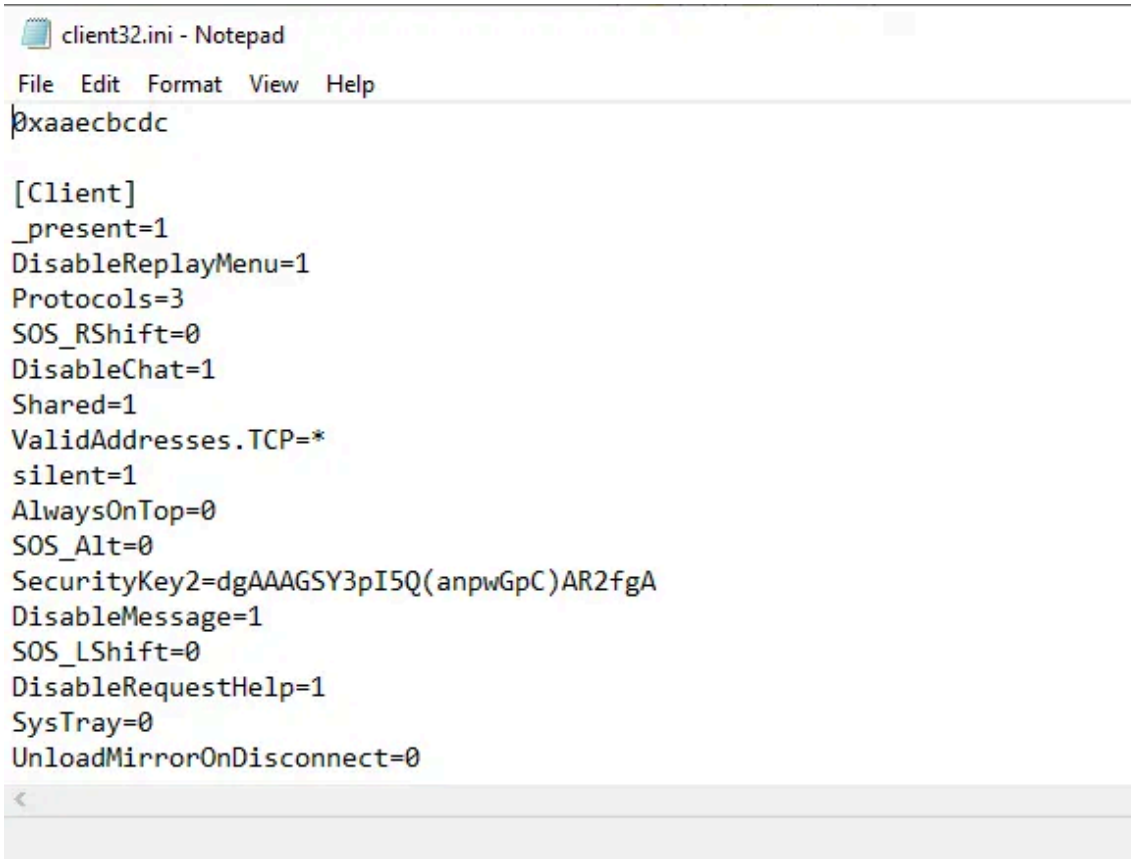
We took the decoded PS command and executed it in the CMD.
The PS creates a directory in the %AppData% called “aragdrts” and stores it inside all the files.

Press enter or click to view image in full size



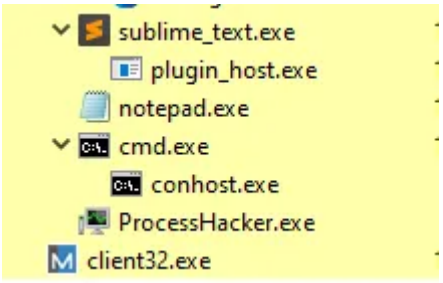
malware directory

After storing the files, it executes automatically the “client32.exe” that sets the persistence that takes the configuration of the connection from the “client32.ini” file.



persistence config

Now we can see in the execution tree that the “client32.exe” is always running and has an active connection to the attacker.



process tree

The exe file uses a few techniques for persistence:
Scheduled tasks
Startup Menu file saving
Registry Key

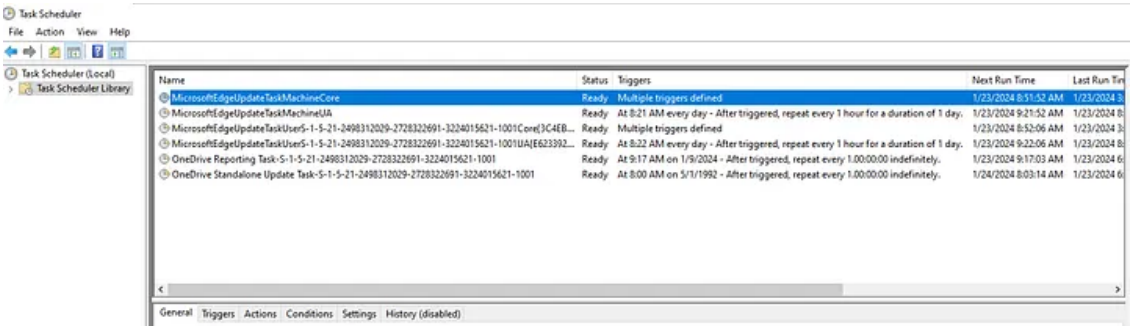
Get Ariel Davidpur's stories in your inbox

Join Medium for free to get updates from this writer.

Remember me for faster sign in

The scheduled task:

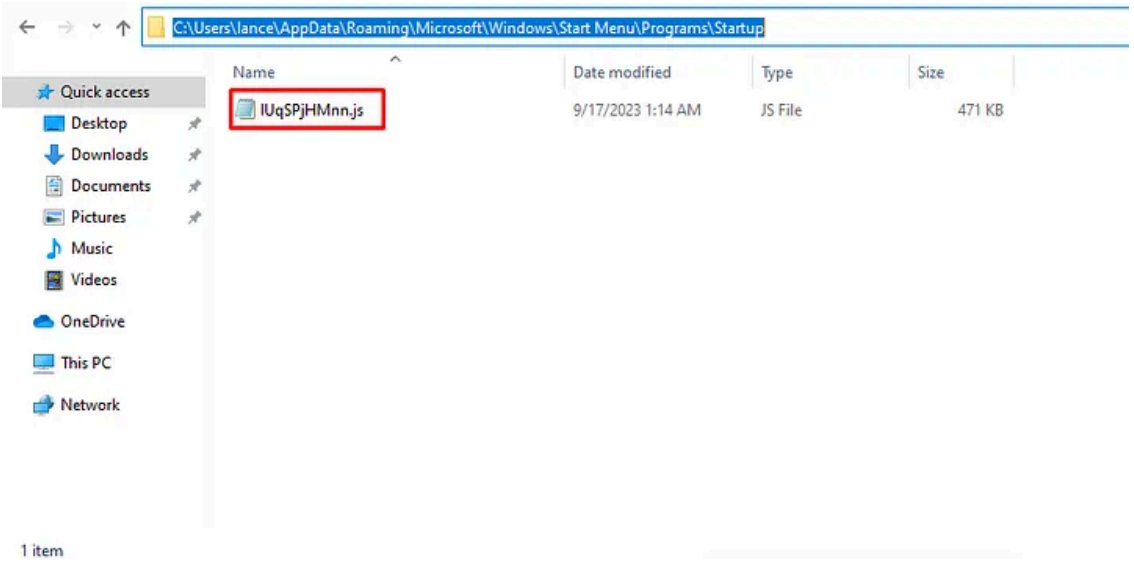
Press enter or click to view image in full size



scheduled task

Startup Menu file:

Press enter or click to view image in full size



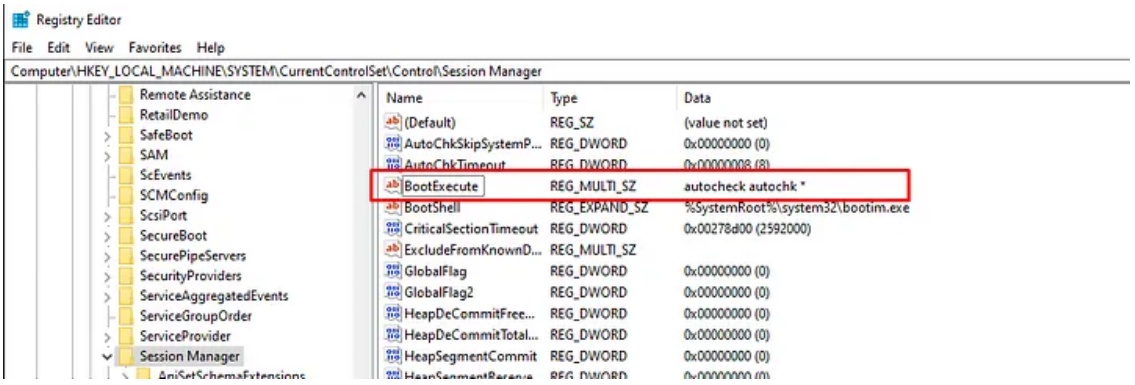
startup menu file

Registry Key:

Press enter or click to view image in full size

Procmon where it saved

Press enter or click to view image in full size

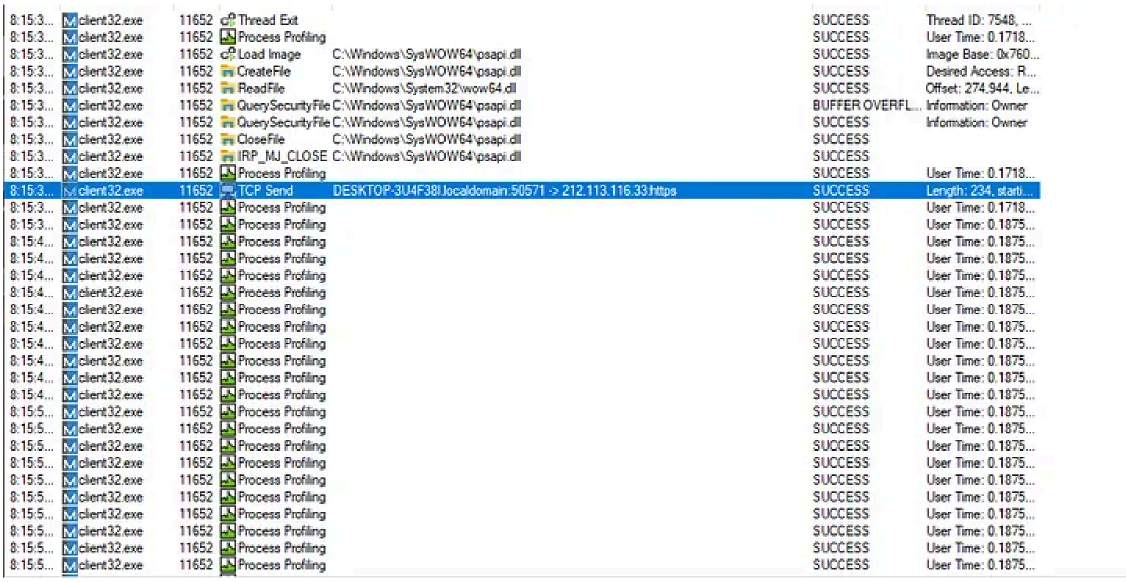


the registry key

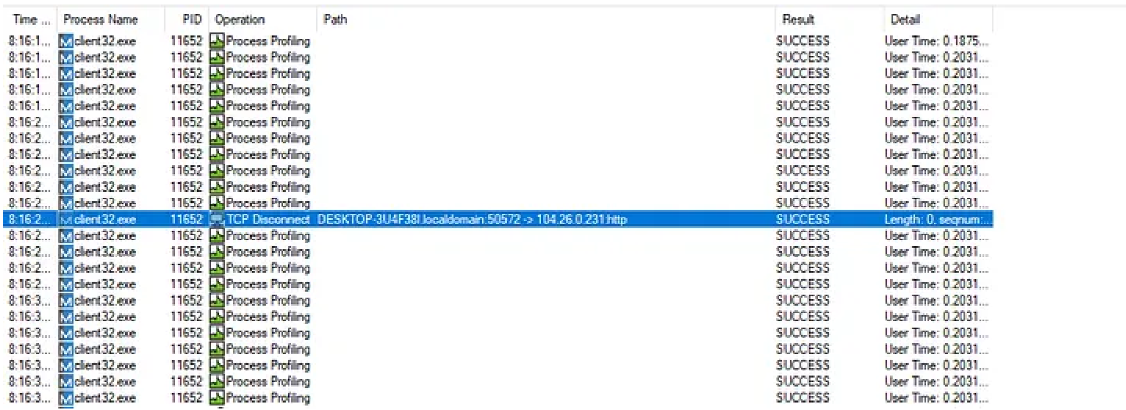
Press enter or click to view image in full size

Now that the attacker has persistence, let's analyze his connection.
Using the Process Monitor, we can see the TCP packets used by the exe file and what are the IPs used for the session and the remote access, using a dedicated port only for the session itself.

Press enter or click to view image in full size



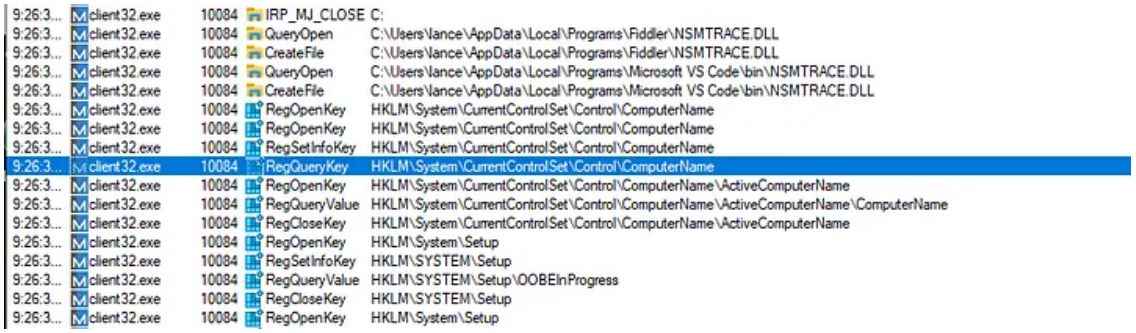
Press enter or click to view image in full size



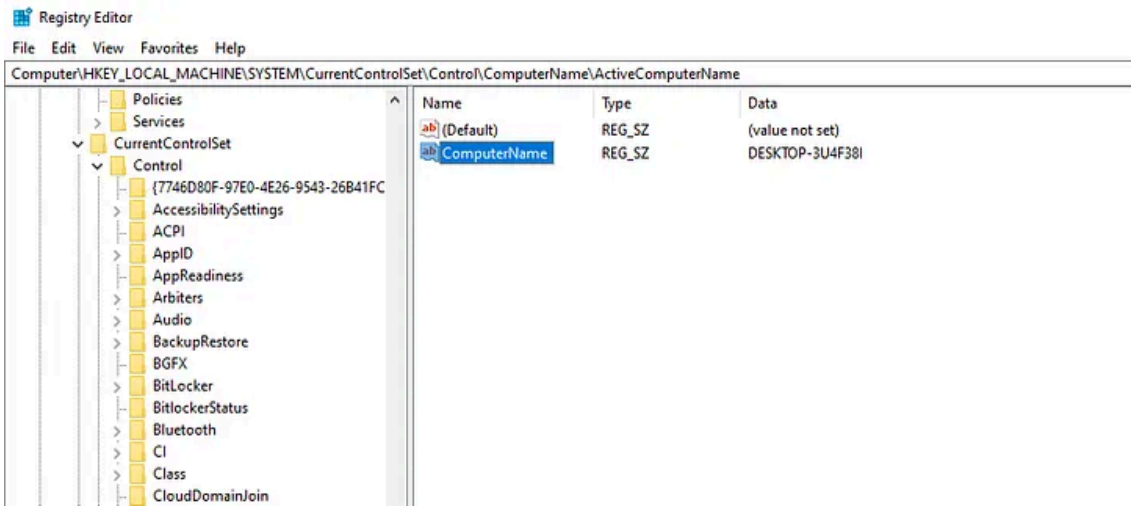
Analyzing the processes, we can now find all the data that we need in addition to the session of the RAT.

Computer name:

Press enter or click to view image in full size

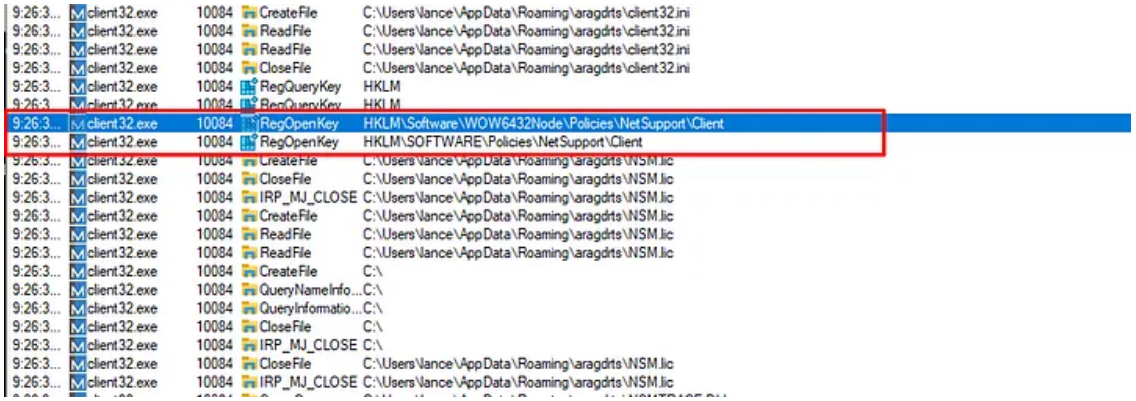


Press enter or click to view image in full size



The name of the **RAT itself!**:

Press enter or click to view image in full size



Conclusion

The NetSupport RAT we found was delivered via a phishing email with a URL that has multiple redirects that eventually download JS malware.

The malware was obfuscated and pulled malicious encoded PS script that was stored on a public URL.

The PS command has the C2s servers and uses BITS transfer to get the malicious files from the attacker's server and execute them.

After executing the "client32.exe" file it uses the other files as configuration for the remote session for the NetSupport.

The malware uses multiple techniques for persistence to make sure that the session won't terminate.

TTPs

Remote Access Software (T1219) — https://attack.mitre.org/techniques/T1219/
Scheduled Task/Job: Scheduled Task(T1053/005) — https://attack.mitre.org/techniques/T1053/005/
Windows Management Instrumentation(T1047) — https://attack.mitre.org/techniques/T1047/
Hide Artifacts: Hidden Files and Directories(T1564/003) — https://attack.mitre.org/techniques/T1564/003/
Boot or Logon Autostart Execution: Registry Run Keys / Startup Folder(T1547/001) — https://attack.mitre.org/techniques/T1547/001/
Hide Artifacts: Hidden Window(T1564/003) — https://attack.mitre.org/techniques/T1564/003/
Modify Registry(T1112) — https://attack.mitre.org/techniques/T1112/
Obfuscated Files or Information: Software Packing(T1406/002) — https://attack.mitre.org/techniques/T1406/002/
System Network Connections Discovery(T1049) — https://attack.mitre.org/techniques/T1049/
File and Directory Discovery(T1083) — https://attack.mitre.org/techniques/T1083/
Process Discovery(T1057) — https://attack.mitre.org/techniques/T1057/
Query Registry(T1012) — https://attack.mitre.org/techniques/T1012/
Non-Standard Port(T1571) — https://attack.mitre.org/techniques/T1571/

IOCs

> ps1.dropper
> hxxps://hsdiagnostico[.]com/readme.php
>
> URLs
> exe.dropper
> hxxps://core-click[.]net/TVFrontend/NSM.zip
> exe.dropper
> hxxps://core-click[.]net/TVFrontend/remcmdstub.zip
> exe.dropper
> hxxps://core-click[.]net/TVFrontend/DLAA1view.zip
> exe.dropper
> hxxps://core-click[.]net/TVFrontend/mock/
>
> Domains:
> helasirasi[.]com [Client32]
> geo[.]netsupportsoftware[.]com [Client32]

> hsdiagnostico[.]com [PowerShell]
> core-click[.]net [tls,http2] (edited)
>
> IPs:
>
> 74.50.81.180
> 98.143.147.253
> 212.113.116.33
> 104.26.1.231:80
>
> SHA256:
>
> [IN5632.js]
> 5657AEA8AFD1E0C0BDC4D3ACDBDF4C8C02ABDF4C75D4687083A6F26BAB09610D
> [Client32]
> 42C2D35457ABCE2FEA3897BA5E569F51B74B40302FF15B782E3B20B0AA00B34E
> StartUp Folder JS:
> 3689DDD7D45EA04F13E073F993AFB1B52D576D455D9317F446A31CC282324213
>
> OpenDir:
> hxxps://core-click[.]net/TVFrontend/ / hxxps://core-click[.]net/TVFrontend/mock/
>
> filename pattern(RegEx):IN[0–9]{4}.js

Credits: Idan Tarab, Ariel Davidpur

Blog References:
https://www.sentinelone.com/blog/gotta-catch-em-all-understanding-the-netsupport-rat-campaigns-hiding-behind-pokemon-lures/
