Hunting for Unsigned DLLs to Find APTs

By Daniela Shalev, Itay Gamliel

Published: 2022-09-26 · Archived: 2026-04-05 13:17:40 UTC

Executive Summary

Malware authors regularly evolve their techniques to evade detection and execute more sophisticated attacks. We've commonly observed one method over the past few years: unsigned DLL loading.

Assuming that this method might be used by advanced persistent threats (APTs), we hunted for it. The hunt revealed sophisticated payloads and APT groups in the wild, including the Chinese cyberespionage group Stately Taurus (formerly known as PKPLUG, aka Mustang Panda) and the North Korean Selective Pisces (aka Lazarus Group).

Below, we show how hunting for the loading of unsigned DLLs can help you identify attacks and threat actors in your environment.

Palo Alto Networks customers receive protections and detections against malicious DLL loading through the Cortex XDR agent.

Threat Actor Groups Discussed	
Unit 42 tracks group as…	**Group also known as…**
Stately Taurus	Mustang Panda, PKPLUG, BRONZE PRESIDENT, HoneyMyte, Red Lich, Baijiu
Selective Pisces	Lazarus Group, ZINC, APT - C - 26

Malicious DLLs: A Common Method Attackers Use for Executing Malicious Payloads on Infected Systems

Based on our observations over years of proactive threat-hunting experience, we hypothesize that one of the main methods for executing malicious payloads on infected systems is loading a malicious DLL. As both individual hackers and APT groups use this method, we decided to conduct research based on this hypothesis.

Most of the malicious DLLs we observe in the wild share three common characteristics:

- The DLLs are mostly written to unprivileged paths.
- The DLLs are unsigned.

- To evade detection, the DLLs are loaded by a signed process, whether a utility dedicated to loading DLLs (such as rundll32.exe) or an executable that loads DLLs as part of its activity.

With that in mind, we found that the most common techniques that are being used by threat actors in the wild are the following:

1. DLL loading by rundll32.exe/regsvr32.exe – While those processes are signed and known binaries, threat actors abuse them to achieve code execution in an attempt to evade detection.
2. DLL order hijacking – This refers to loading a malicious DLL by abusing the search order of a legitimate process. This way, a benign application will load a malicious payload with the name of a known DLL.

Reviewing the results of the above techniques in the wild revealed that the most common unprivileged paths to load malicious unsigned DLLs are the folders and sub-folders of ProgramData, AppData and the users' home directories.

The next section will introduce several findings based on the above hypothesis.

Attack Trends in the Wild Related to Unsigned DLLs

To start hunting based on the hypothesis we described, we created two XQL queries. The first one looks for unsigned DLLs that were loaded by rundll32.exe/regsvr32.exe, while the other looks for signed software that loads an unsigned DLL.

The hunting activity revealed various malware families that used unsigned DLL loading. Figure 1 presents the malware we detected using these methods over the past six months (February-August 2022).

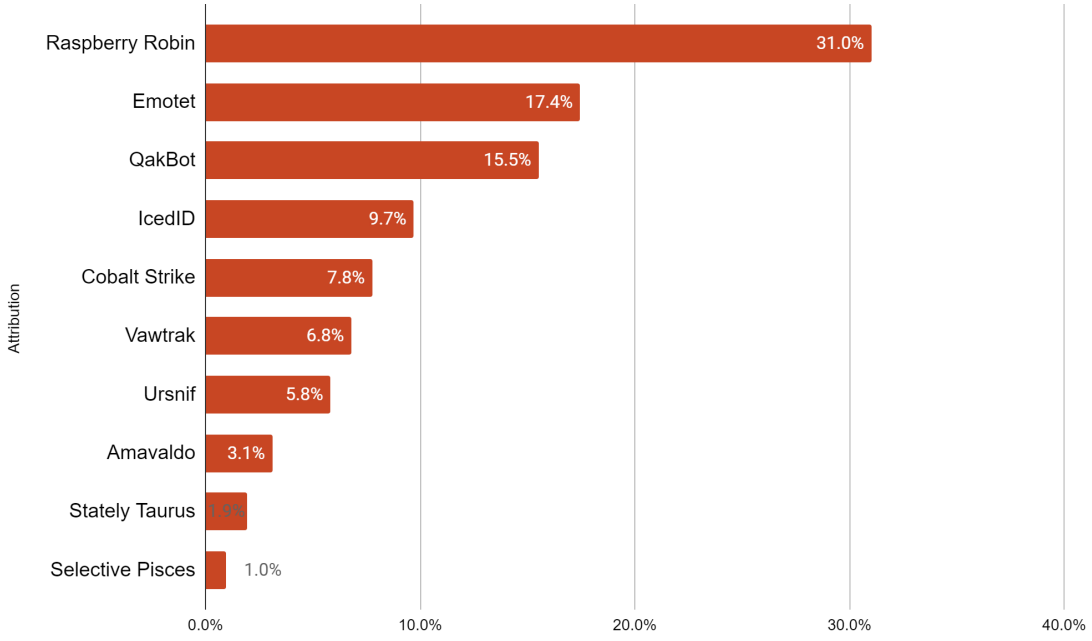

Figure 1. Malware observed using DLL loading.

Analyzing the execution techniques used by the above threats showed that banking trojans and individual threat actors typically used rundll32.exe or regsvr32.exe to load a malicious DLL, while APT groups used the DLL side-loading technique most of the time.

Diving Into Selected Payloads

Stately Taurus

We decided to highlight an investigation around Stately Taurus activity that we detected in the environment of one organization. Stately Taurus is a Chinese APT group that usually targets non-governmental organizations and is known for abusing legitimate software to load payloads.

In this case, we observed the usage of the DLL search order hijacking technique that enabled the attacker's malicious DLL to load into the memory space of a legitimate process. The threat actor used multiple pieces of third party software for the DLL side-loading, such as antivirus software and a PDF reader.

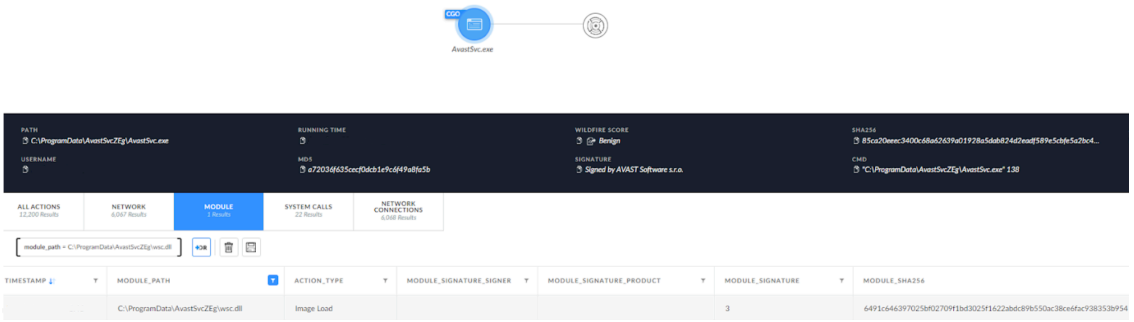

Figure 2. AvastSvc.exe uses side-loading to load a malicious DLL.

To achieve DLL side-loading, the group dropped the payload into the ProgramData folder, which contained three files – a benign EXE file for DLL hijacking (AvastSvc.exe), a DLL file (wsc.dll) and an encrypted payload (AvastAuth.dat). The loaded DLL appeared to be the PlugX RAT, which loads the encrypted payload from the .dat file.

File Create	AvastAuth.dat		C:\ProgramData\AvastSvcZEg\AvastAuth.dat
File Write	wsc.dll		C:\ProgramData\AvastSvcZEg\wsc.dll
File Write	wsc.dll		C:\ProgramData\AvastSvcZEg\wsc.dll
File Create	wsc.dll		C:\ProgramData\AvastSvcZEg\wsc.dll
File Write	AvastSvc.exe		C:\ProgramData\AvastSvcZEg\AvastSvc.exe

Figure 3. PlugX files: benign executable, DLL loader and encrypted .dat file.

Selective Pisces

Among the results of our hunting queries, we also identified several high-entropy malicious modules within the ProgramData directories shown in Figure 4.

ACTION_MODULE_PATH	ACTOR_PROCESS_IMAGE_PATH	MODULE_ENTROPY	COUNTER
C:\ProgramData\USOShared\uso.dat	C:\ProgramData\USOShared\uso.dat	0.999545	1
C:\ProgramData\Microsoft\Windows\LfSvc\mi.dll	C:\ProgramData\Microsoft\Windows\LfSvc\wsmprovhost.exe	0.999396	1
C:\ProgramData\MarkAny\mi.dll	C:\ProgramData\MarkAny\wsmprovhost.exe	0.999393	1
C:\ProgramData\MarkAny\PrtDAT\SXSHARED.DLL	C:\ProgramData\MarkAny\PrtDAT\dfrgui.exe	0.999354	1
C:\ProgramData\USOShared\USOShared.tmp	C:\Windows\System32\rundll32.exe	0.999317	1

Figure 4. DLL side-loading by Selective Pisces.

Investigating the execution chain of the unsigned modules shown in Figure 4 revealed that they were dropped to the disk by the signed DreamSecurity MagicLine4NX process (MagicLine4NX.exe).

MagicLine4NX.exe executed a second-stage payload that we observed utilizing DLL side-loading in order to evade detection. The second-stage payload wrote a new DLL named mi.dll, and copied wsmprovhost.exe (host process for WinRM) to a random directory in ProgramData. Wsmprovhost.exe is a native Windows binary that attempts to load mi.dll from the same directory. The attackers abused this mechanism in order to achieve DLL side-loading (T1574.002) with this process.

The mi.dll payload was observed dropping a new payload named ualapi.dll to the System32 directory (C:\Windows\System32\ualapi.dll). As ualapi.dll is in this case a missing DLL on the System32 directory, the attackers used this fact to achieve persistence by giving their malicious payload the name ualapi.dll. That way, spoolsv.exe will load it upon startup.

After analyzing the payloads above, we attributed them to the North Korean APT group that Unit 42 tracks as Selective Pisces. This group's utilization of legitimate third party-software such as MagicLine4NX was described earlier this year in a blog post by Symantec.

Raspberry Robin

The last attack we would like to elaborate on is the most common one we observed in the wild.
Some of the results that our query yields share several common characteristics:

- DLLs with scrambled names reside in random sub-folders of the ProgramData or AppData folders.
- Those DLLs have a similar range of entropy (~0.66).
- All of them were loaded by rundll32.exe or regsvr32.exe

For example: RUNDLL32.EXE C:\ProgramData\<random_folder>\fhcplow_Tudjdm.dll,iarws_sbv

ACTION_MODULE_PATH	MODULE_ENTROPY	COUNTER
C:\ProgramData\FunctionApplication\LikenseProcess\npwmse_Nemcttdid.dll	0.662609	26
C:\ProgramData\DirectoryStarted\ProteteiinDirect\iyclBaiic_fig.dll	0.665728	8
C:\Users\ [redacted] \AppData\Local\MediumInput\SenuorJuest\Portsoft_Gdfv.dll	0.665709	2
C:\ProgramData\PagesSetup\SsndZets\adlsoft_w_ni.dll	0.667337	12
C:\ProgramData\FeatureAnimation\TrqzeIbternet\dfjssoft_002.dll	0.663264	6
C:\Users\ [redacted] \AppData\Local\VeryRegister\FiltewsSchedule\fhcplow_Tudjdm.dll	0.669195	8
C:\Users\ [redacted] \AppData\Local\AccountsAssets\BasicKpph\CNBERngs_De4_3.dll	0.667706	2

Figure 5. DLLs loaded by Raspberry Robin.

The DLL loading activities that take place in those attacks were attributed to a campaign called Raspberry Robin, which was recently described by Red Canary.

Those attacks begin from a shortcut file on an infected USB device. This spawns msiexec.exe to retrieve the malicious DLL from a remote C2 server. Over installation, a scheduled task is created in order to achieve persistence, loading the DLL using rundll32.exe/regsvr32.exe on system start up.

Using Unsigned DLLs to Hunt for Attacks in Your Environment

You can hunt for the loading of unsigned DLLs using XQL Search in Cortex XDR.
To narrow down the results, we suggest focusing on the following:

- For DLL side-loading, we recommend paying attention to known third-party software placed in non-standard directories.
- Focus on the file's entropy – binaries that have a high value of entropy may contain a packed section that will be extracted during execution.
- Focus on the frequency of execution – high-frequency results may indicate a legitimate activity that occurs periodically, while low-frequency results may be a lead for an investigation.
- Focus on the file's path – results that contain folders or files with scrambled names are more suspicious than others.

ACTION_MODULE_SHA256	ACTION_MODULE_PATH	MODULE_ENTROPY	COUNTER
45a9b070cf4c167299766ab84ecbf6f76559592e261aa1449dcba068831f5224	C:\Users\ [redacted] \AppData\Local\Awvlulrcoqqevg\gzwdmbhfz.lqa	0.860774	1
3647c1b0d158ad58b044841737l2e13f17ea4b9c7015cb4b0c984c0c44819b2d	C:\Users\ [redacted] \AppData\Local\Awvlulrcoqqevg\gzwdmbhfz.lqa	0.802315	1
a6fbd79ce5eab32a9a33fc9e296edc78225e03f803857cc96da985df1a4c1dc9	C:\Users\ [redacted] \AppData\Roaming\Highspot\OutlookAddin\HighspotOutlookAddin64.dll	0.773056	2
ed2e9f64df54f0fa93ec3f3dc2e5c19fe5e004a61803b4c6b93d028f4985a599	C:\ProgramData\Autodesk\ApplicationPlugins\VRay3dsMax2022\bin\ChaosThumbnailHandler.dll	0.737689	3
095e49ae5611b78a76c7af5e4e2395eb0056e246ef50602fddc0053302a918d9	C:\Users\ [redacted] \AppData\Roaming\Citrix\SelfService\Chooseanapplication.exe	0.637715	1

Figure 6. Query results sorted by the module's entropy.

Figure 6 contains partial results of the queries that are mentioned in the next section, sorted by the module's entropy. While the first two rows are an example of Emotet execution, the others are benign DLLs.

Hunting Queries

```
// Rundll32.exe / Regsvr32.exe loads an unsigned module from uncommon folders over the past 30 days.

config case_sensitive = false timeframe = 30d

| dataset = xdr_data

| filter event_type = ENUM.LOAD_IMAGE and (action_module_path contains "C:\ProgramData" or action_module_path contains "\public" or action_module_path contains "\documents" or action_module_path contains "\pictures" or action_module_path contains "\videos" or action_module_path contains "appdata") and action_module_signature_status = 3 and (actor_process_image_name contains "rundll32.exe" or actor_process_image_name contains "regsvr32.exe") and (actor_process_command_line contains "programdata" or actor_process_command_line contains "\public" or actor_process_command_line contains "\documents" or actor_process_command_line contains "\pictures" or actor_process_command_line contains "\videos" or actor_process_command_line contains "appdata")

| alter module_entropy = json_extract_scalar(action_module_file_info, "$.entropy")

| fields agent_hostname , action_module_sha256  , action_module_path  , actor_process_image_name, actor_process_command_line, module_entropy

| comp count (action_module_path) as counter by action_module_path , action_module_sha256 , module_entropy
```

```
// Possible DLL side-loading - a signed process loaded an unsigned DLL from AppData\ProgramData\Public folder over the past 30 days

config case_sensitive = false timeframe = 30d

| dataset = xdr_data

| filter event_type = ENUM.LOAD_IMAGE and (action_module_path contains "C:\ProgramData" or action_module_path contains "\public" or action_module_path contains "appdata") and action_module_signature_status = 3 and (actor_process_image_name not contains "rundll32.exe" or actor_process_image_name not contains "regsvr32.exe") and actor_process_signature_status = 1 and (actor_process_image_path contains "appdata" or actor_process_image_path contains "programdata" or actor_process_image_path contains "public" )

| alter module_entropy = json_extract_scalar(action_module_file_info, "$.entropy")

| fields agent_hostname , action_module_sha256  , action_module_path  , actor_process_image_name, actor_process_command_line, module_entropy, actor_process_image_path

| comp count (action_module_path) as counter by action_module_path , action_module_sha256 , module_entropy, actor_process_image_path
```

Conclusion

Most detection techniques for blocking malicious DLLs rely on the module's behavior after it has been loaded into memory. This can limit the ability to block all malicious modules.

That said, you can proactively hunt for malicious unsigned DLLs using hunting approaches such as the ones presented in this blog.

Knowing the baseline of your network in terms of legitimate software or behavior can reduce the number of results generated by the above queries, allowing you to focus on results that might be suspicious.

Cortex XDR alerts on and blocks malicious DLLs loaded by known hijacking techniques, and can also prevent post-exploitation activities, through the Behavioral Threat Protection and Analytics modules.

Indicators of compromise and TTPs associated with Stately Taurus can be found in the Stately Taurus ATOM.

If you think you may have been compromised or have an urgent matter, get in touch with the Unit 42 Incident Response team or call North America Toll-Free: 866.486.4842 (866.4.UNIT42), EMEA: +31.20.299.3130, APAC: +65.6983.8730, or Japan: +81.50.1790.0200.

Indicators of Compromise

Threat Actor	SHA256
Selective Pisces	779a6772d4d35e1b0018a03b75cc6f992d79511321def35956f485debedf1493
Selective Pisces	d9b1ad70c0a043d034f8eecd55a8290160227ea66780ccc65d0ffb2ebc2fb787
Selective Pisces	3131985fa7394fa9dbd9c9b26e15ac478a438a57617f1567dc32c35b388c2f60
Selective Pisces	5be717dc9eda4df099e090f2a59c25372d6775e7d6551b21f385cf372247c2fd
Selective Pisces	18cc18d02742da3fa88fc8c45fe915d58abb52d3183b270c0f84ae5ff68cf8a2
Selective Pisces	7aa62af5a55022fd89b3f0c025ea508128a03aab5bc7f92787b30a3e9bc5c6e4
Selective Pisces	79b7964bde948b70a7c3869d34fe5d5205e6259d77d9ac7451727d68a751aa7d
Selective Pisces	cf9ccba037f807c5be523528ed25cee7fbe4733ec19189e393d17f92e76ffccc
Selective Pisces	32449fd81cc4f85213ed791478ec941075ff95bb544ba64fa08550dd8af77b69
Selective Pisces	5a8b1f003ae566a8e443623a18c1f1027ec46463c5c5b413c48d91ca1181dbf7
Selective Pisces	5bb4950a05a46f7d377a3a8483484222a8ff59eafdf34460c4b1186984354cf9
Stately Taurus	352fb4985fdd150d251ff9e20ca14023eab4f2888e481cbd8370c4ed40cfbb9a
Stately Taurus	6491c646397025bf02709f1bd3025f1622abdc89b550ac38ce6fac938353b954

Stately Taurus	e8f55d0f327fd1d5f26428b890ef7fe878e135d494acda24ef01c695a2e9136d
Raspberry Robin	06f11ea2d7d566e33ed414993da00ac205793af6851a2d6f809ff845a2b39f57
Raspberry Robin	202dab603585f600dbd884cb5bd5bf010d66cab9133b323c50b050cc1d6a1795
Raspberry Robin	f9e4627733e034cfc1c589afd2f6558a158a349290c9ea772d338c38d5a02f0e
Raspberry Robin	9fad2f59737721c26fc2a125e18dd67b92493a1220a8bbda91e073c0441437a9
Raspberry Robin	9973045c0489a0382db84aef6356414ef29814334ecbf6639f55c3bec4f8738f

Table 1. Hashes of samples.
