Quick-Analysis/SmokeLoader/SmokeLoader.md at main · vc0RExor/Quick-Analysis

By vc0RExor

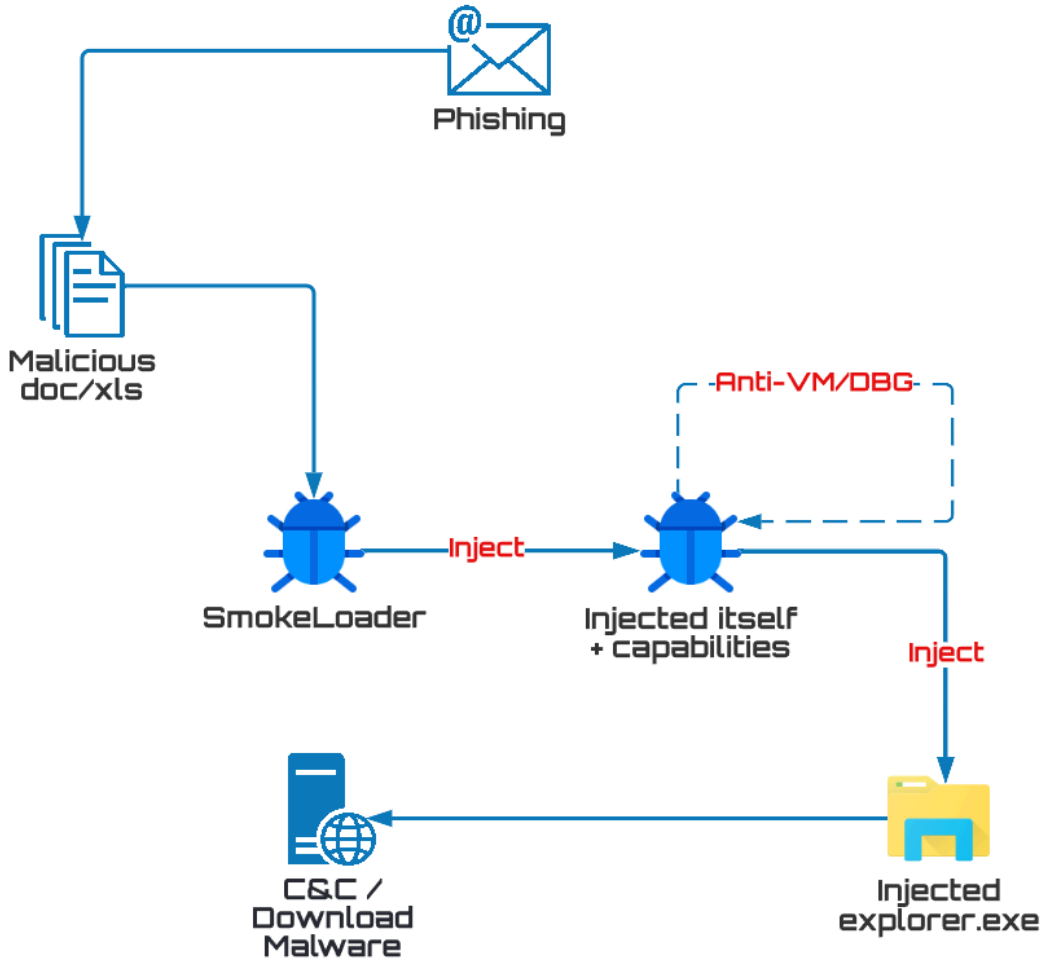
Archived: 2026-04-05 21:22:45 UTC

_Overview

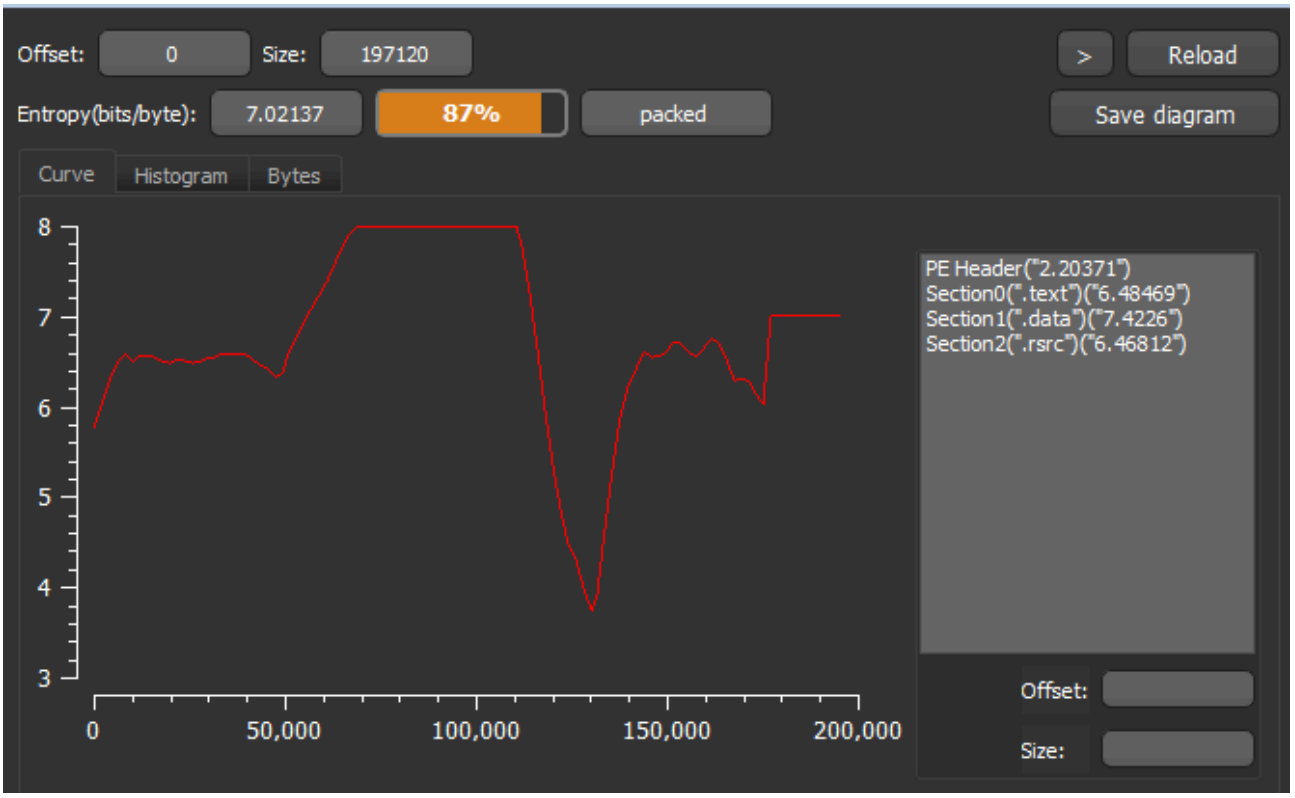
SmokeLoader is a malware that generally acts as a backdoor and is commonly used as a loader for other malware. Attributed to the criminal group Smoky Spider, a group that uses SmokeLoader and Sasfis, loader and downloader respectively. SmokeLoader has been used as a bot in infrastructures and contains strong evasion capabilities as well as Anti-Analysis, Anti-VM and Anti-DBG techniques.

_Technical Analysis

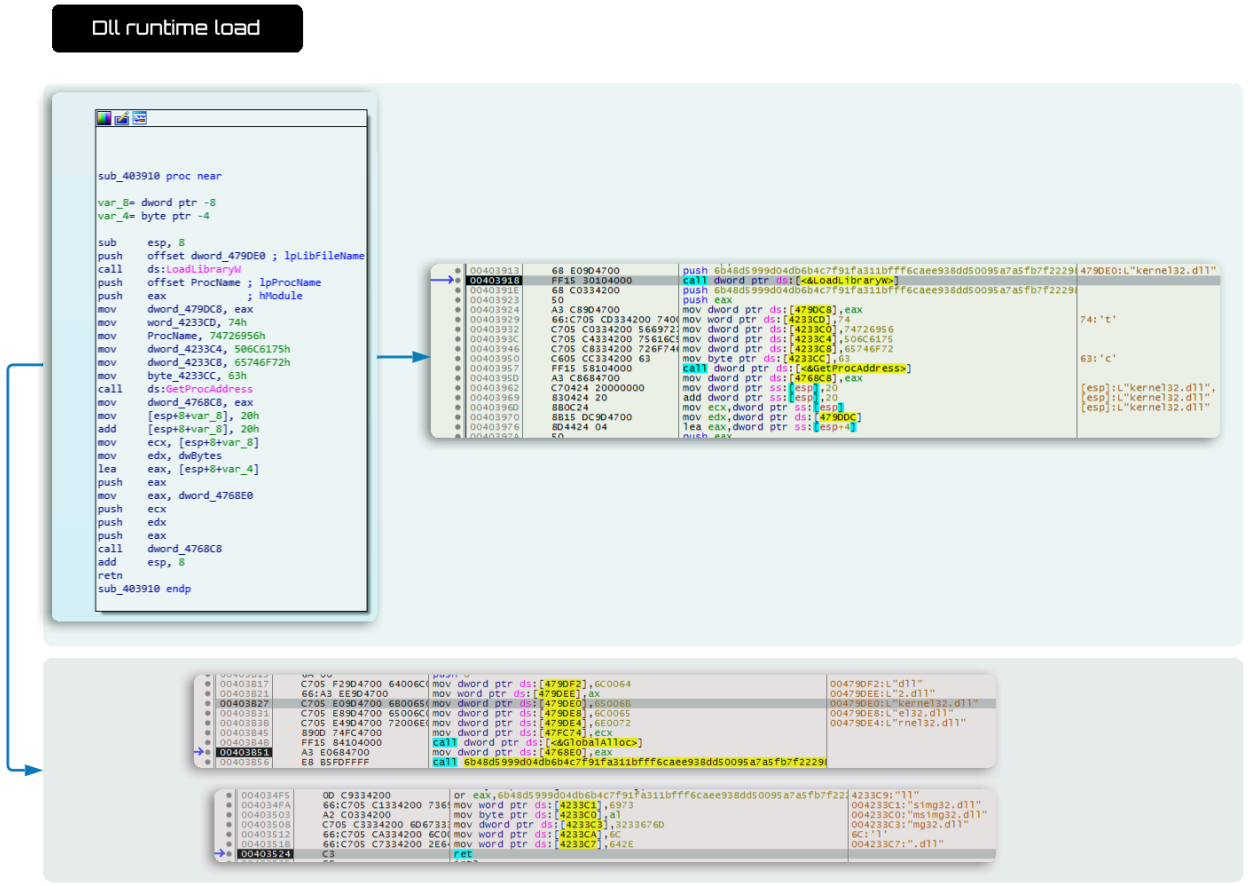
SmokeLoader appears on systems usually through phishing, although it can be loaded by other PUP/PUA or malware. The main execution will revolve around a document that will spawn the SmokeLoader which will run, in most of its versions, a version of itself in a suspended state to inject code, after which it will execute an *explorer.exe* that it will inject again in order to perform the malicious C&C actions or download other files using legitimate software.



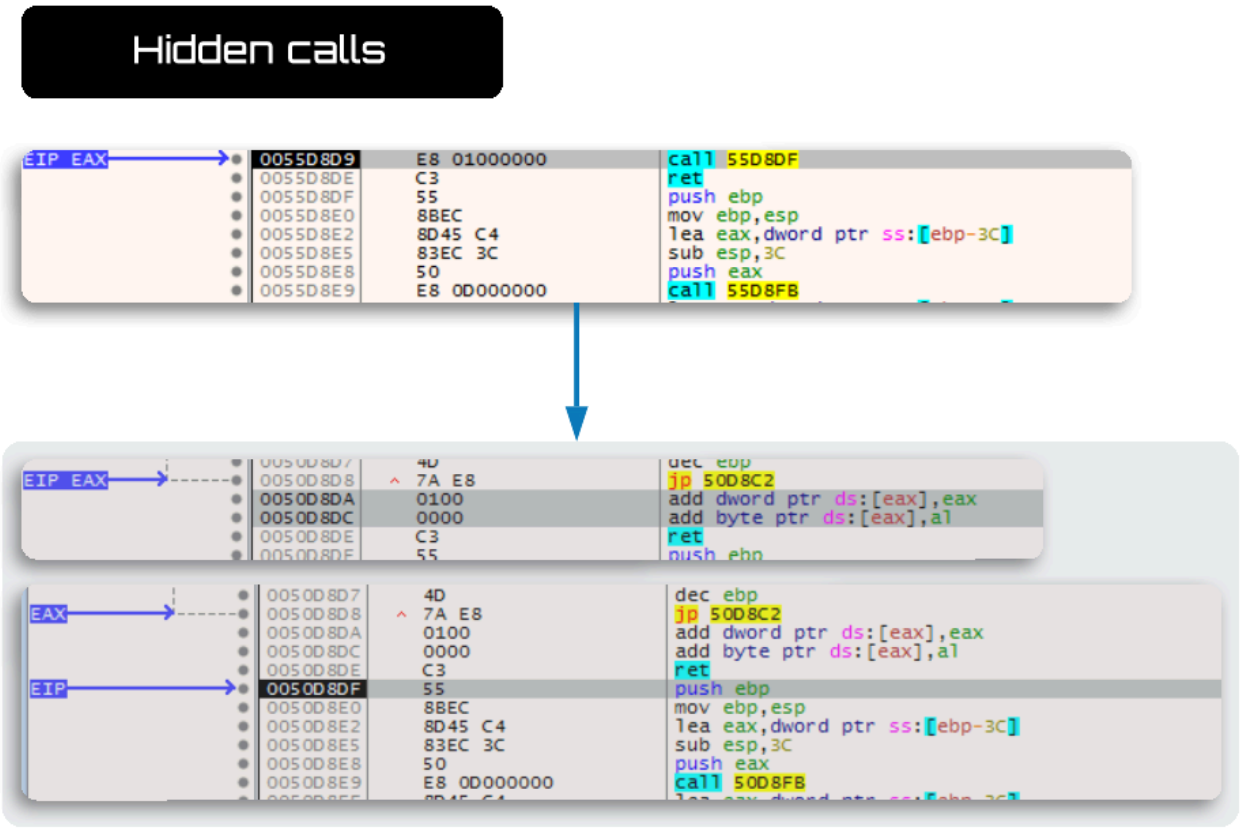
The samples that have been found have in most cases been detected as packed, due to the high level of entropy contained in their sections.



At the initial point, we see how it tries to load libraries in RunTime, something really useful since it prevents us from being able to discern its intentions if we perform a basic static analysis, so it will obtain new functionalities during its execution.



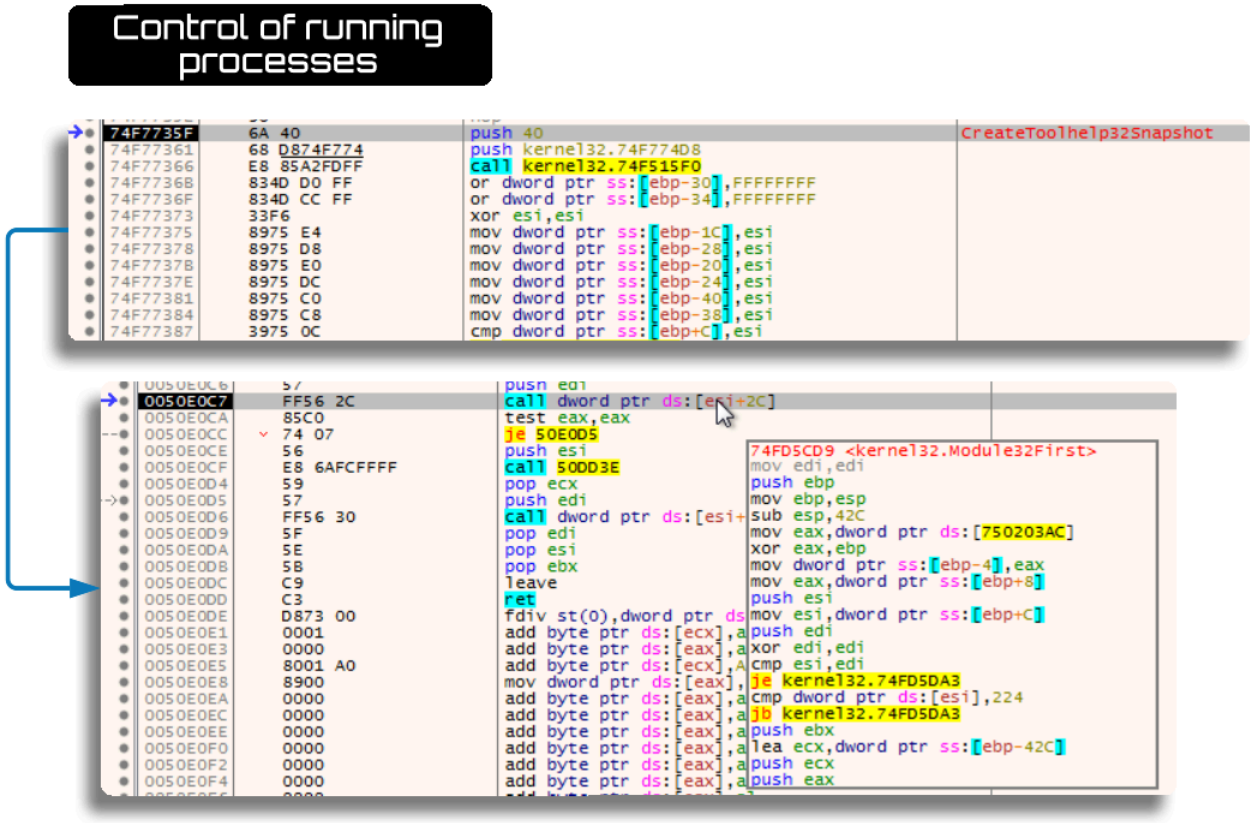
In some of the techniques used to hinder the analysis, such as code obfuscation, we find different hidden calls, as well as abuses of RET to reach calls that we will not see statically.



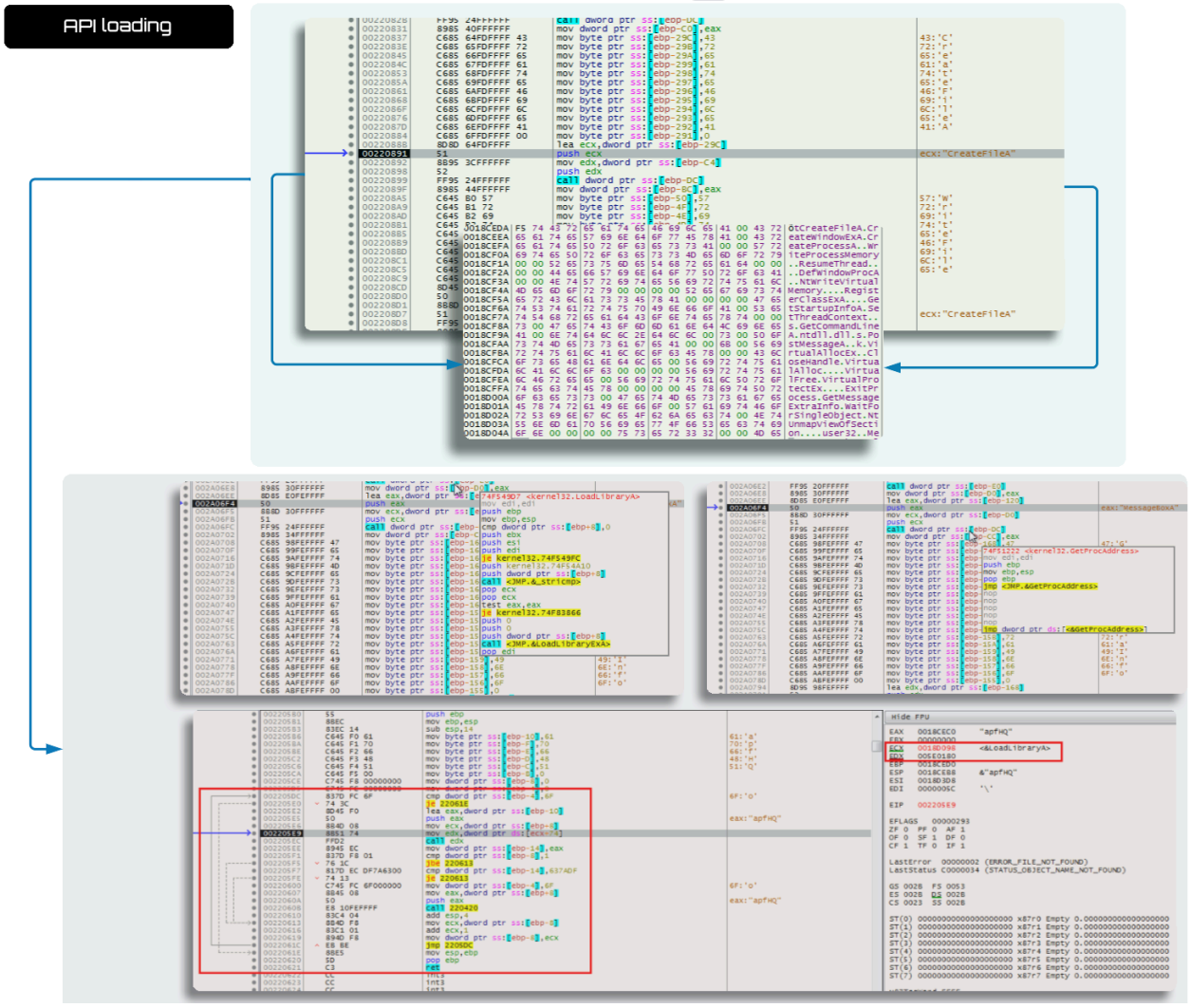
As mentioned above, it fetches libraries during runtime and is dedicated to resolving APIs that it could use later on

Resolving APIs

At all times, it has control over what is running on the machine, as it subsequently performs various Anti-Vm and Anti-dbg techniques, so having all running processes mapped is always a good technique.



After this, it starts loading APIs that will serve it moments later, in which we will see a routine that will be loading from memory and using LoadLibrary + GetProcAddress

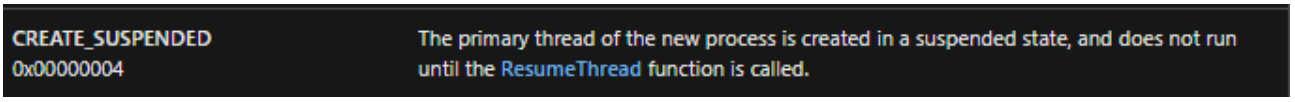


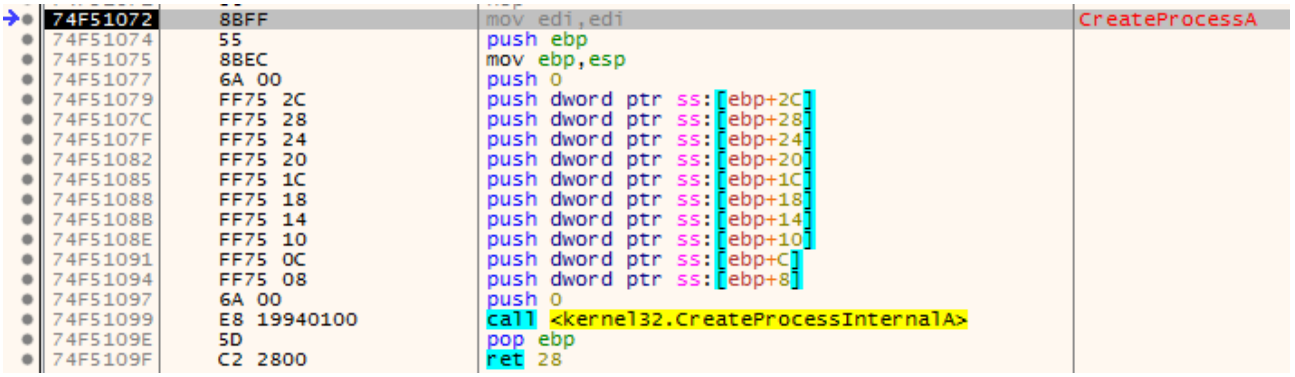
APIs:

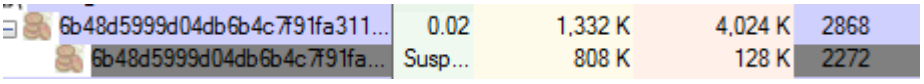
```
 CreateFileA
CreateWindowExA
CreateProcessA
WriteProcessMemory
ResumeThread
DefWindowProcA
NtWriteVirtualMemory
RegisterClassExA
GetStartupInfoA
SetThreadContext
GetCommandLineA
PostMessageA
VirtualAllocEx
CloseHandle
VirtualAlloc
VirtualFree
VirtualProtectEx
```

```
ExitProcess
GetMessageExtraInfo
WaitForSingleObject
NtUnmapViewOfSection
MessageBoxA
ReadProcessMemory
GetThreadContext
WriteFile
GetModuleFileNameA
GetFileAttributesA
WinExec
GetMessageA
```

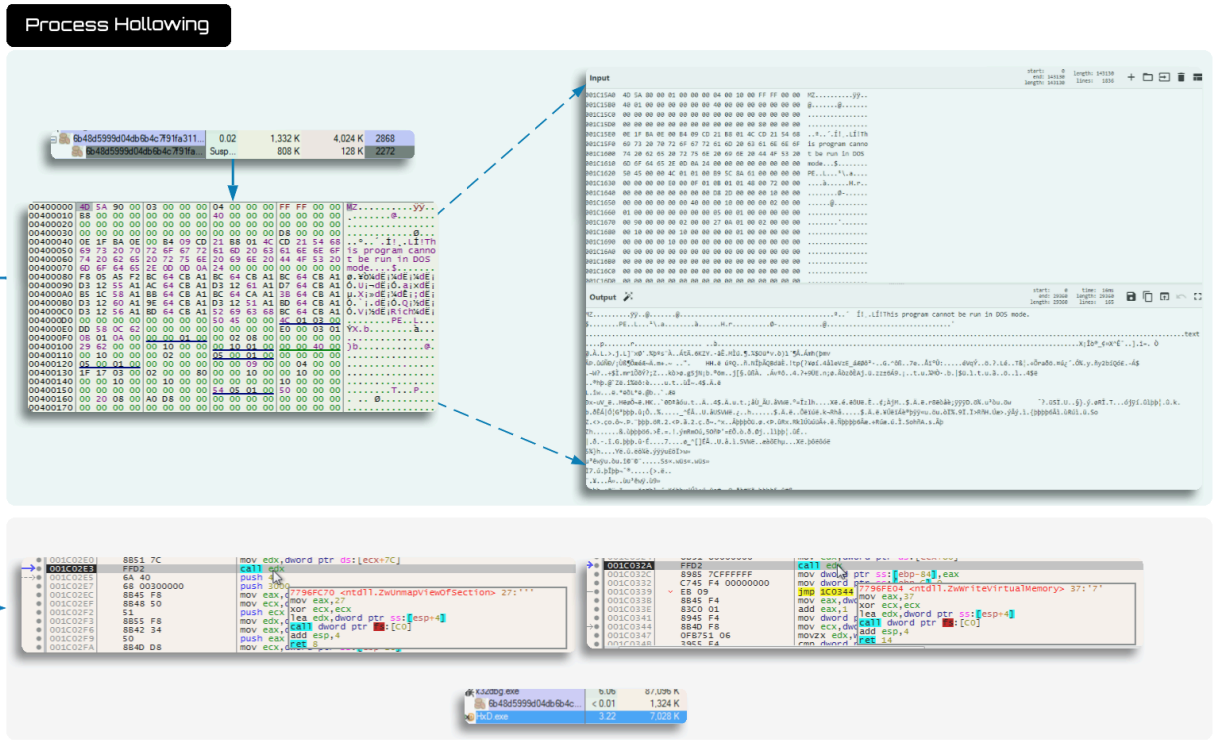
Once it has the libraries, APIs and processes controlled, it creates a process in suspended state, for this it uses CreateProcessInternalA that will call CreateProcessInternalW entering 0x04 in dwCreationflags to create the process in suspended state.







Once the process is created in a suspended state, it proceeds to introduce the binary inside the previously spawned process, which, through ProcessHollowing, will unmap data from itself, to write the binary inside, this is usually done through ZwUnmapViewOfSection + VirtualAlloc + ZwWriteVirtualMemory, once introduced into the memory of the process in suspension, it will stop being suspended and will execute it, so the memory file will be detonated.



[The binary extracted from memory, which will inject explorer.exe, is very interesting, we will follow soon :) 🕵️]

_IOC

_SHA256

```
 Ebdebba349aba676e9739df18c503ab8c16c7fa1b853fd183f0a005c0e4f68ae
D618d086cdfc61b69e6d93a13cea06e98ac2ad7d846f044990f2ce8305fe8d1b
Ee8f0ff6b0ee6072a30d45c135228108d4c032807810006ec77f2bf72856e04a
6b48d5999d04db6b4c7f91fa311bfff6caee938dd50095a7a5fb7f222987efa3
B961d6795d7ceb3ea3cd00e037460958776a39747c8f03783d458b38daec8025
02083f46860f1ad11e62b2b5f601a86406f7ee3c456e6699ee2912c5d1d89cb9
059d615ce6dee655959d7feae7b70f3b7c806f3986deb1826d01a07aec5a39cf
5318751b75d8c6152d90bbbf2864558626783f497443d4be1a003b64bc2acbc2
79ae89733257378139cf3bdce3a30802818ca1a12bb2343e0b9d0f51f8af1f10
F92523fa104575e0605f90ce4a75a95204bc8af656c27a04aa26782cb64d938d
```

_IP

```
 216.128.137.31
8.209.71.53
```

_Domains

```
 host-file-host6[.]com
host-host-file8[.]com
fiskahlilian16[.]top
paishancho17[.]top
ydiannetter18[.]top
azarehanelle19[.]top
quericeriant20[.]top
xpowebs[.]ga
venis[.]ml
tootoo[.]ga
eyecosl[.]ga
bullions[.]tk
mizangs[.]tw
mbologwuholing[.]co[.]ug
quadoil[.]ru
```

> 🦖 vc0=Rexor 🕵️

Source: https://github.com/vc0RExor/Quick-Analysis/blob/main/SmokeLoader/SmokeLoader.md