**Stage0.exe**

**Static**

Goal: Discovery as much as possible regarding the functionality of this malware.

Known reverse shell

Sha256: fca62097b364b2f0338c5e4c5bac86134cedffa4f8ddf27ee9901734128952e3

Strings / Imports:

* CreateRemoteThread Import

file-size: 391987 bytes

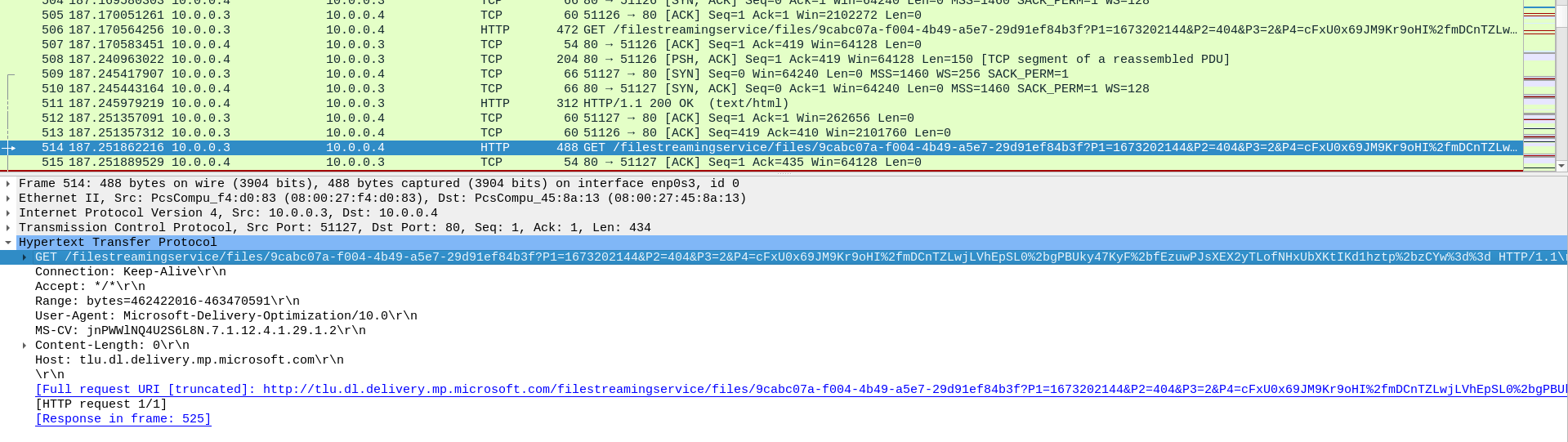
Does Not Appear to be packed (similar virtual and raw size) but I also didn’t find many useful strings

32-bit executable

Likely written in C

**Dynamic**

GET /filestreamingservice/files/9cabc07a-f004-4b49-a5e7-29d91ef84b3f?P1=1673202144&P2=404&P3=2&P4=cFxU0x69JM9Kr9oHI%2fmDCnTZLwjLVhEpSL0%2bgPBUky47KyF%2bfEzuwPJsXEX2yTLofNHxUbXKtIKd1hztp%2bzCYw%3d%3d



Files Created – many other files created too



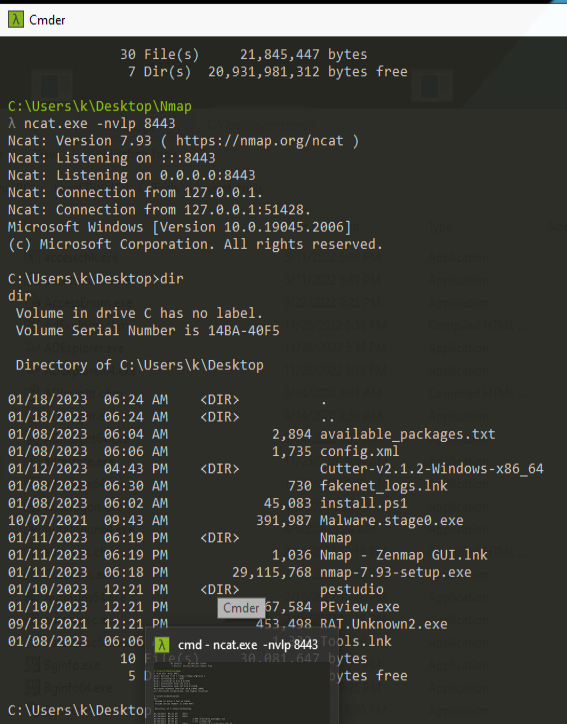
Werlift.exe while similar to the later referenced werfault.exe is different. Wefault.exe is a windows program tats prompts a user for more information on a bug they experienced. This werlift.exe file is created upon running the malware and we can use our advanced static analysis to get a better idea of what it does.

Graphical user interface, text, email

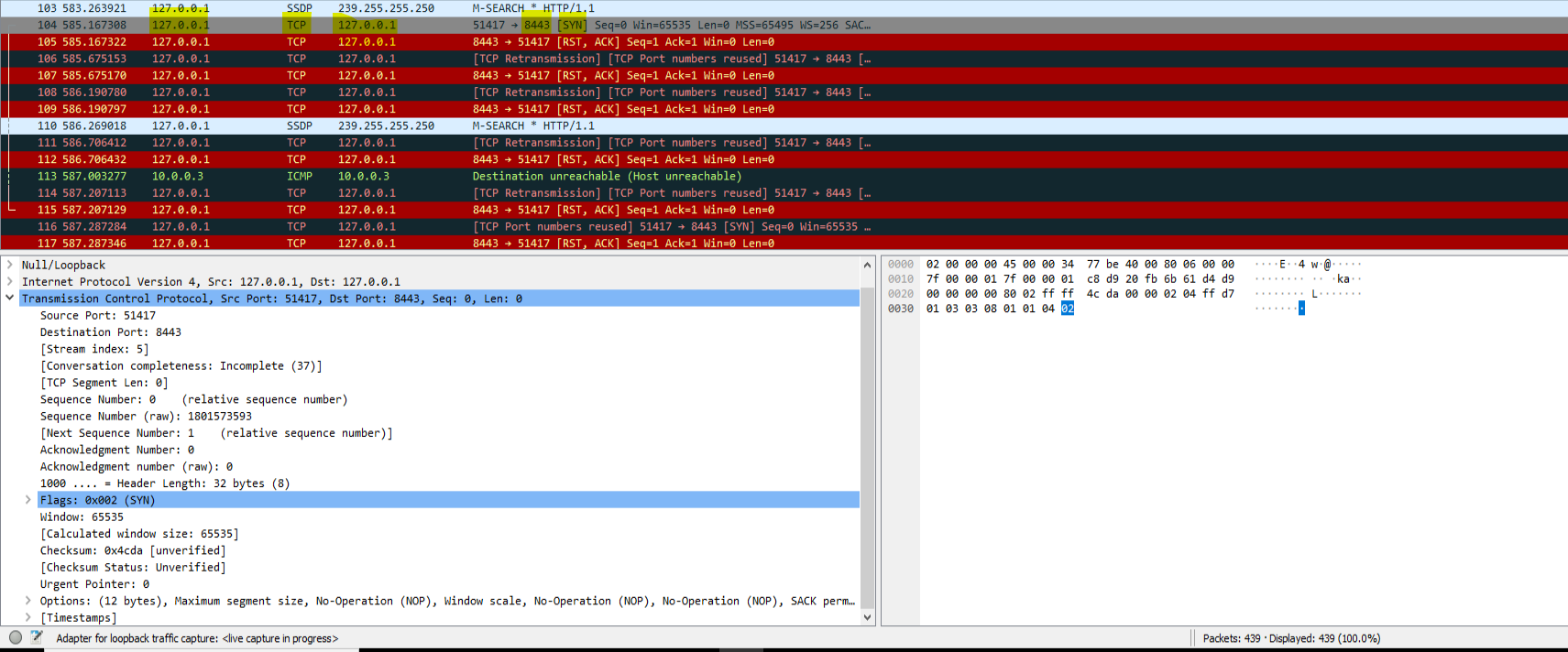
Description automatically generated

WerFault.exe,1864,TCP,Syn Sent,127.0.0.1,51418,127.0.0.1,8443,1/18/2023 6:49:39 AM,WerFault.exe,,,,

* TCP request from “WerFault.exe”



Definitely is a remote shell, seems to be a good one too.



Here we can see the SYN packet being sent from the malware to our local host on port 8443

**Advanced Static Analysis**

We can see this is a classic example of process injection. This is an evasion technique where a API call in this case CreateRemoteThread is injected into an existing process running on this pc. This can help to hide evidence of the malicious api call as it is now sharing the same metadata with an existing process.

How we know this:

* We seed a CALL dword [OpenProcess] this is of course an API call. The 3 values pushed onto the stack before this call are the parameters used in this call. This call is basically gibing the malware full control of the process being passed in to the call, at this point this process is a variable arg\_ch which is passed into main
* So we can see that this malware is injected the createremotethread api call into an existing process

We can then visualize and verify this with a tool called Process Hacker 2 which shows processes, and allows us to visualize processes in a cool and useful way making it easy to see the “hierarchy” of processes