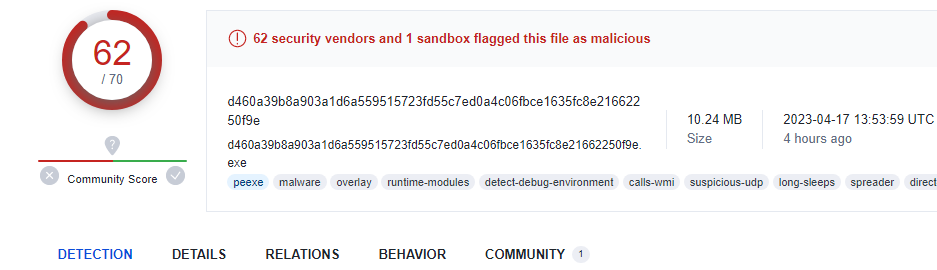
In the past lectures, we discussed many different tools and the last two discussion posts have introduced you to a few different ways malicious actors could "trick" a user to opening it or hide from analysis. Both examples were somewhat of an exaggeration. This malware attempts to mimic those as well but has some built in evasion techniques.

Answer the following questions using the Lecture13 zip file:

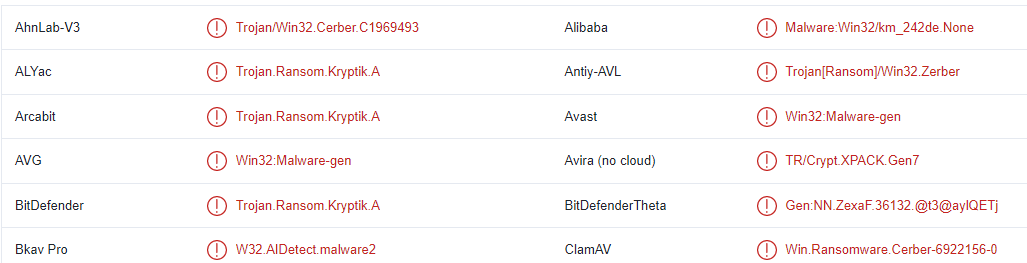
**Upload the file to**[**http://www.VirusTotal.com/**](http://www.virustotal.com/)**. Does the file match any existing antivirus signatures?**

Yes. It matches 62 of 70 security vendor signatures.



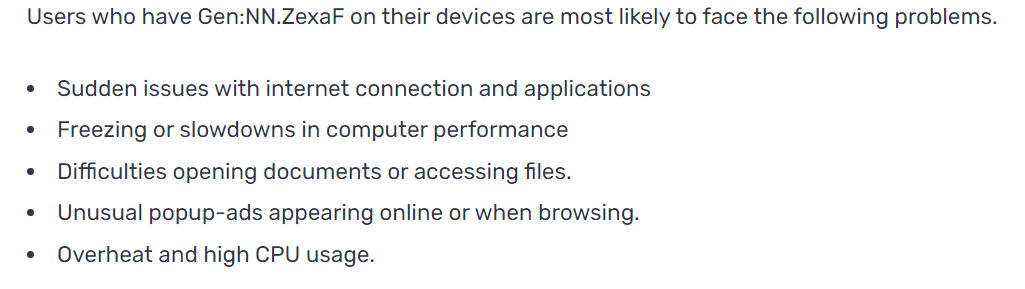
**What is this file known as?**

VirusTotal showed multiple names for this malware, such as Kryptik, ZexaF, and Cerber. More generically, multiple security vendors identified this malware as being ransomware.

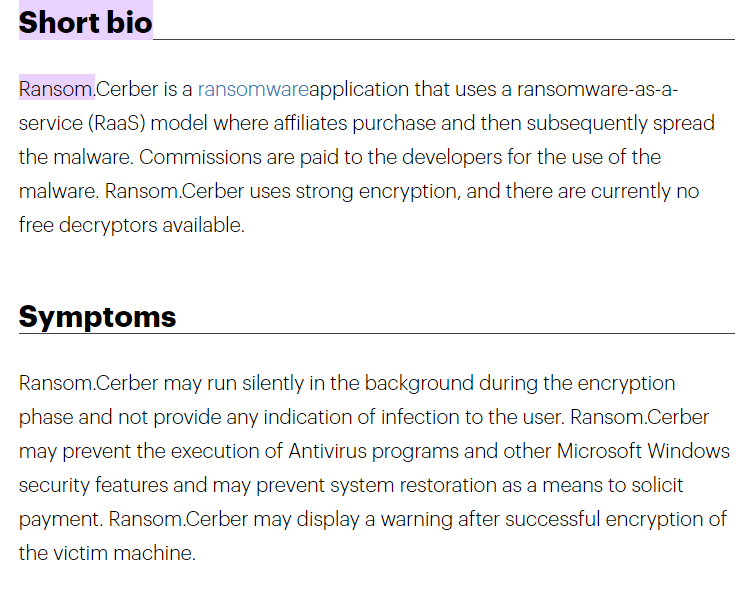


In a previous discussion post, I analyzed the Kryptik malware as a backdoor trojan.

One website identified the ZexaF signature as malware that can has network-based capabilities, can cause physical hardware damage to the PC through overheating, and other computer performance issues.

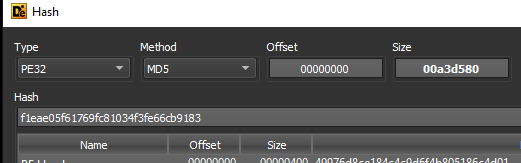


[Malwarebytes](https://www.malwarebytes.com/blog/detections/ransom-cerber#:~:text=Cerber-,Short%20bio,Ransom.) stated that “Cerber” is a type of Ransomware as a Service (RaaS) malware where there aren’t any free decryptors available. They state that it will run silently while it is encrypting files and can prevent the use of antivirus programs to remove it. Any attempt to do so will prompt a user for payment, but will eventually show the ransom message after the encryption is complete.



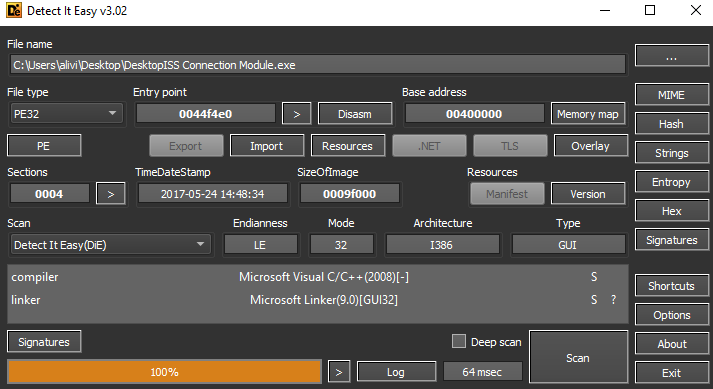
**What is the hash of the file?**

MD5: f1eae05f61769fc81034f3fe66cb9183

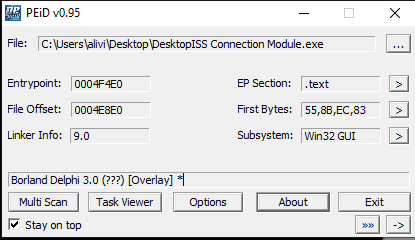


**Using the tools we have discussed so far, conduct your static analysis and present your findings.**

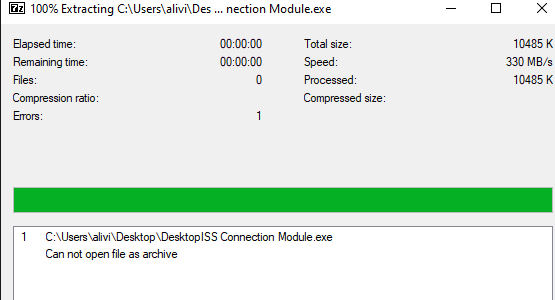
The file was compiled nearly 6 years ago on 2017-05-24 at 14:48:34 MST. It appears to be written in C/C++ and uses a linker.



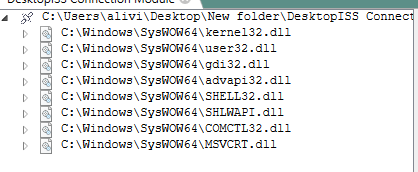
It also appears have used Borland Delphi 3.0, an IDE, to write the code. This IDE also contains functionality to pack files.



Interestingly, the file could not have its component parts extracted with UniExtract nor with 7zip.

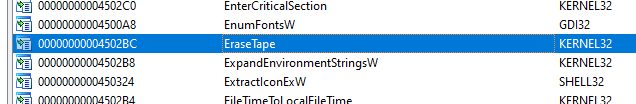


Dependencies showed common imports discussed throughout this course. Most concerning are the imports of shell32.dll and shlwapi.dll. This indicates that the program has some shell functionality and potentially could be used to establish a backdoor or download additional files from another host/server (or upload data to it).

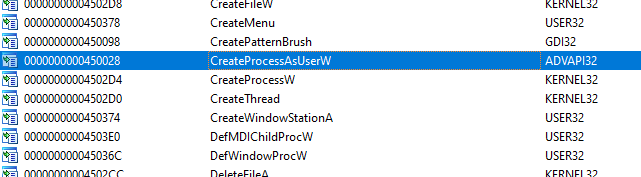


A strings analysis of the file yielded strings that corresponded to functions identified within the Dependencies analysis of the file imports. However, most of the data within the strings were random hex data. Therefore, it is reasonable to conclude at this point that the file is both packed and obfuscated.

IDA Pro was not extensively helpful in determining what this program does specifically, but there were some imports that caused suspicion. The “EraseTape” function is invoked which is used to erase contents of a tape within a tape drive (such as a CD-ROM).

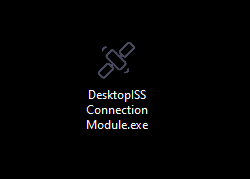


There are also calls to process-related functions, potentially indicative of persistence-generating behavior.

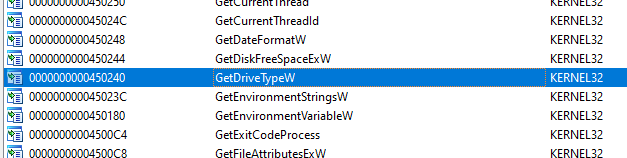


**What techniques were used to make this file dangerous in an organization**

The first technique that the file did was to disguise itself as some sort of program that insinuates it connects to a live feed of the International Space Station. This makes it attractive for users to readily click on it. This is “masquerading” according to the MITRE ATT&CK framework.



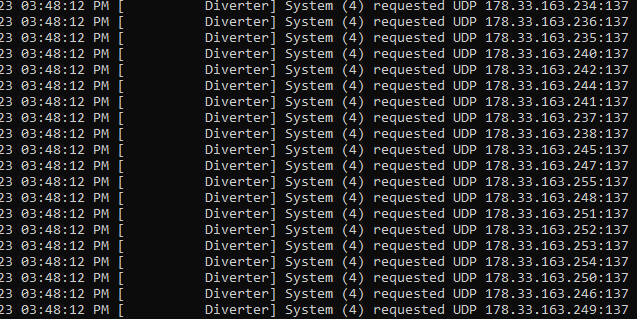
Other techniques, as identified above, were to disguise its true purpose through obfuscation. However, some critical functions that have no business being in a legitimate program were identified. Firstly, the EraseTape function could potentially erase critical pieces of data from the organization and cause a lot of harm. This function is must likely invoked after the GetDriveTypeW function is called (GetDriveTypeW returns information about the drive).



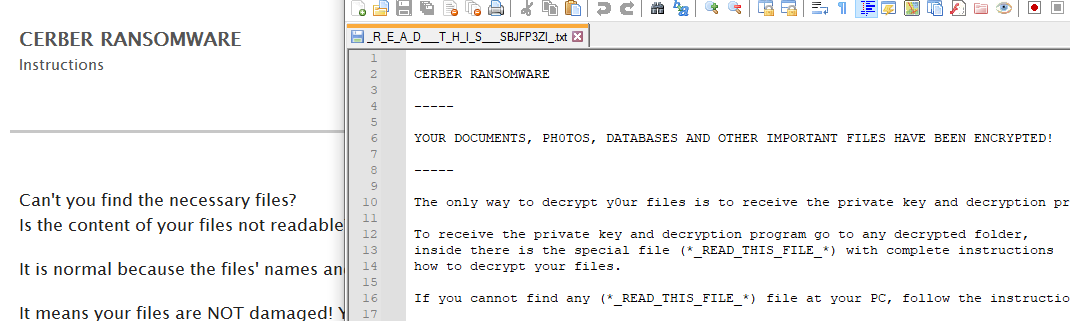
Secondly, although some legitimate programs do use the shell API, it is unlikely that one with good intentions is. Creating a shell can allow this file to download additional malicious payloads as well as used to exfiltrate stolen data.

**Using the tools, we have discussed so far, conduct your dynamic analysis and present your findings.**

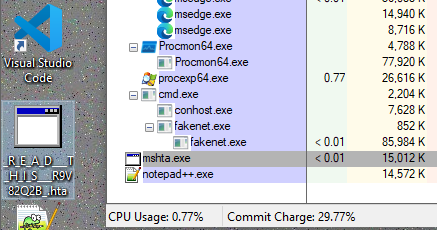
After running the file, there was a massive influx of FakeNet traffic captured over port 137, the NetBIOS Name Service (NBNS). NBNS allows the host machine to communicate with other devices on the network.



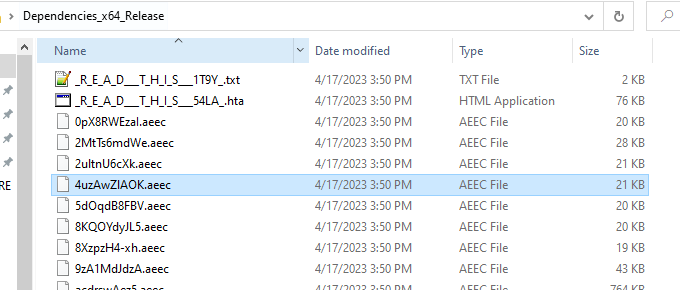
As I was writing the paragraph above, the file was confirmed as Cerber ransomware due to an application popup in addition to a .txt file displaying instructions.



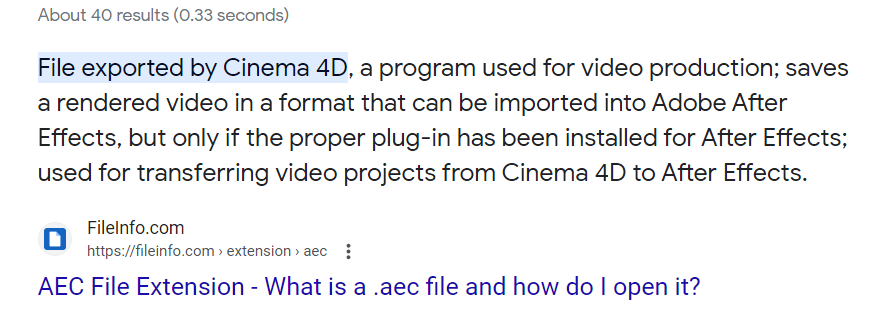
The .exe popup was identified within Process Explorer as “mshta.exe”, a windows application used to execute HTML application files identified with the file extension “.hta”. A file with the .hta extension was found on the desktop with a similar title as the .txt file.



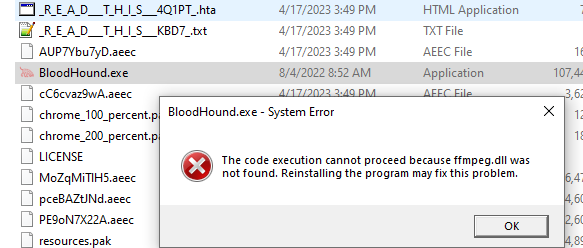
The .hta and .txt files seemed to copy itself to directories in which it performed some action. Since this was run on the desktop, other folders on the desktop (not shortcuts) contained copies of the two files. An example can be seen below where it was copied into the Dependencies x64 release folder located on my desktop.



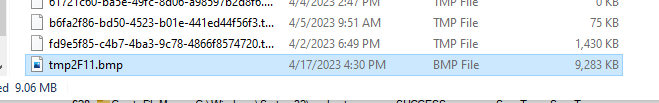
Also of note, the other files within the folder were set to the .aeec file format. This is a file extension exported by the “Cinema 4D” program for use in Adobe After Effects. This is likely the encryption method used by the malware to lock the files. These file extensions were observed on other files and documents throughout the machine.



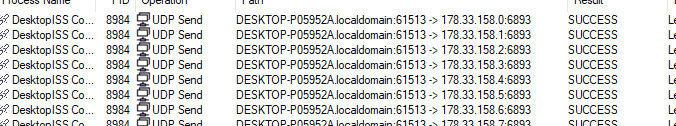
When attempting to run additional applications that had their files changed to the .aeec format, they could not be run.



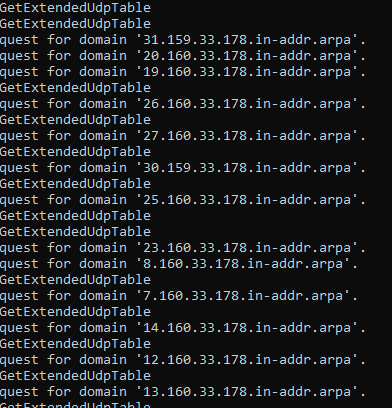
The desktop background image was found in AppData\Local\Temp and labeled as tmp2f11.bmp.



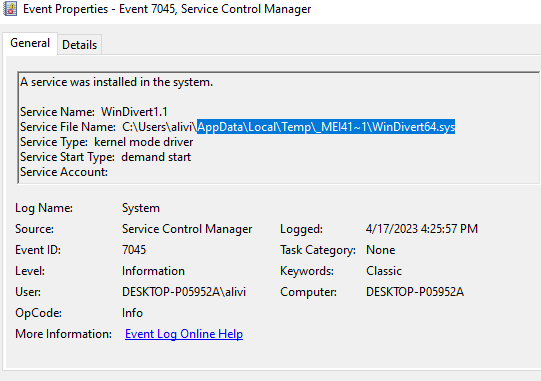
UDP network traffic was captured in procmon from the malware, attempting to send information to multiple IP addresses from unreserved ports.



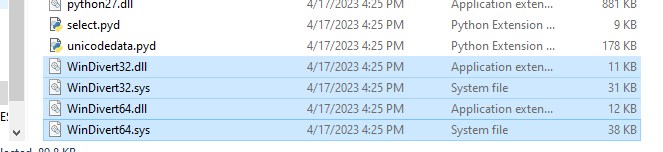
FakeNet captured a lot of different IP addresses within its capture relating for domain requests.



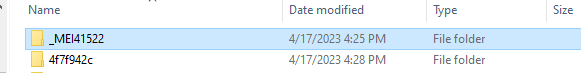
Using Windows Event Viewer, one service was found to be created (Event ID 7045) and the file was located in AppData\Local\Temp\\_MEI41~1\ and entitled, “WinDivert64.sys” with a service type being a kernel mode driver. This is most likely where the malware established persistence: establishing a rootkit.



There were four total files created. Two .sys and two .dll files with 32 and 64 naming conventions, most likely referring to x32 and x64 architectures.



In fact, two folders within Local\Temp were created: “\_MEI41522” and “4f7f942c”. Each file within the folders was last modified at the time the malware ran. These folders and associated files did not exist when the VM was restored from last snapshot. These files are suspected to be the instructions to allow the newly-established rootkit to run.



**What techniques were deployed to make this file more dangerous compared to previous malware samples we have seen.**

This is the first malware that potentially deployed a rootkit by creating a kernel mode driver service type. Rootkits are known to be exceptionally difficult to remove from a system and allow a malicious actor unfettered access to a machine.

**Over the last 13 weeks, you have been able to tackle multiple types of malware. Would you consider this malware to be more difficult compared to the previous few weeks’ samples based of your experiences so far?**

I would consider the static analysis portion more difficult than the previous versions. Determining the techniques it used that make it dangerous to an organization during the static analysis was difficult to answer and can only be comprehensively analyzed during the dynamic analysis. There was also an enormous amount of activity that the malware performed that had to be sifted through, but process of elimination allowed me to identify new services created. To identify the new service, I used Event Viewer, a tool I have not extensively used throughout this course. But it certainly helps to narrow down newly-created services.

**This malware was very destructive when it was first released and was one of the first to deploy the RaaS model. Using your last 13 weeks of experience, give your opinion on what type of malware development advances you expect in the next five to ten years. This does not need to focus on ransomware but any of the various malware families.**

I expect malware development to incorporate extensive use of obfuscation and packing techniques, each more creative than the last. The more computing power they are able to leverage in development will be directly correlated to the complexity and sophistication of the malware.

It is also hard to underestimate AI technology advancements (such as ChatGPT) that will be used to not only develop malware, but also to fight against it. I used ChatGPT throughout this course to get quick answers to function and .dll purposes rather than scouring the internet for related documentation.

I also believe that more malware will be tailored to mobile devices. An acquaintance of mine had a family member fall victim to bluesnarfing, resulting his password manager credentials being stolen and having thousands of dollars stolen from his bank account. I think that whatever malware is developed in the future will leverage unique deployment techniques, such as over Bluetooth, to achieve their goals.

**Go to two other students' posts and observe their findings. Post if you agree or disagree with their analysis.**