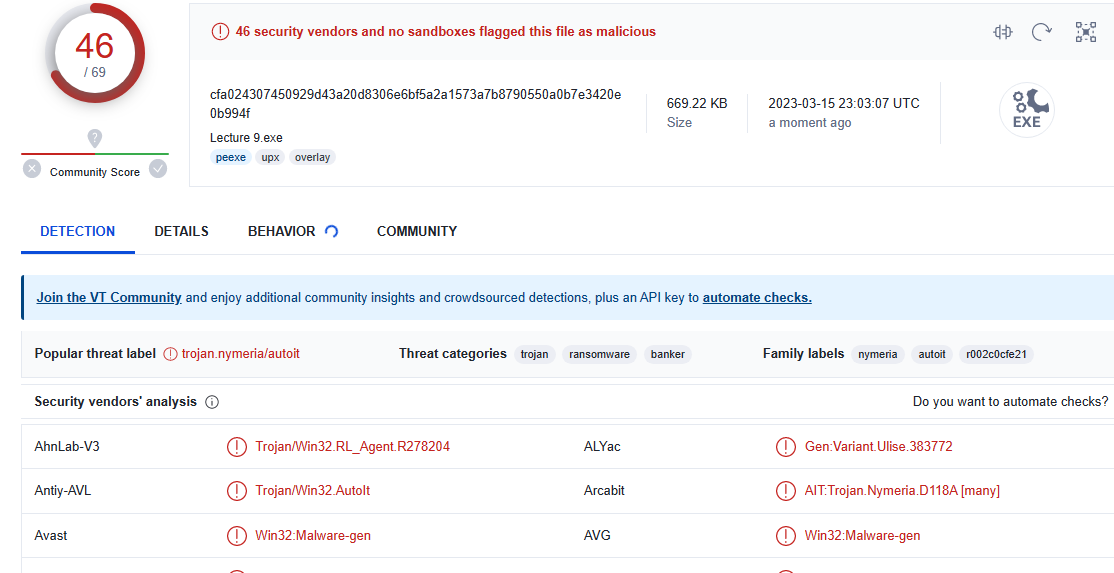
Answer the following questions using Lecture9.exe:

Answer the following questions:

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

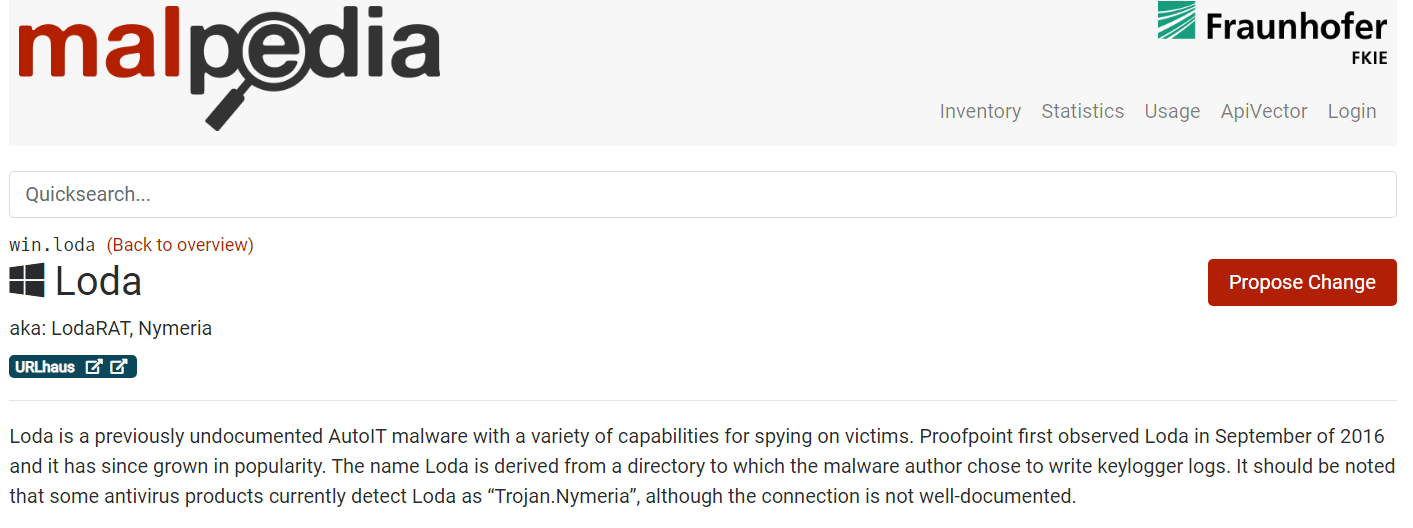
Yes. The file matches 46 of 69 existing security vendor signatures.



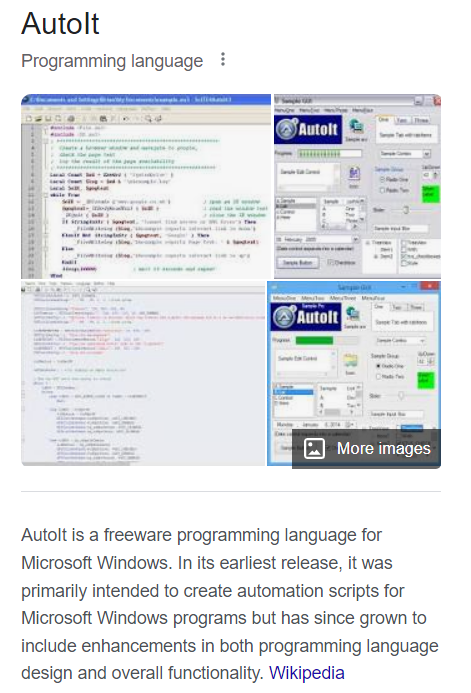
**What is this file known for?**

In the screenshot above in more security vendor signatures, the malware is identified as the Nymeria Trojan. There are other vendor descriptions that allude to a type of bank fraud malware. The name “AutoIt” is mentioned a few times as well.

On Malpedia, Nymeria is also known as LodaRAT and is a type of AutoIT malware. It states that it has spyware capabilities and has keylogger capabilities.

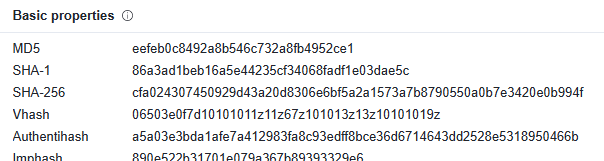


A quick search for AutoIt shows that AutoIt is a programming language, the language in which this file was most likely written in.



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

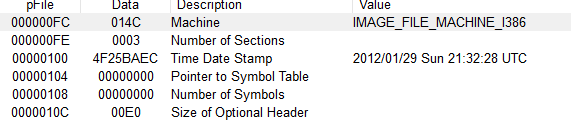
The MD5 hash is: eefeb0c8492a8b546c732a8fb4952ce1



**Using the tools, we discussed so far, answer the below.**

**When was this file compiled?**

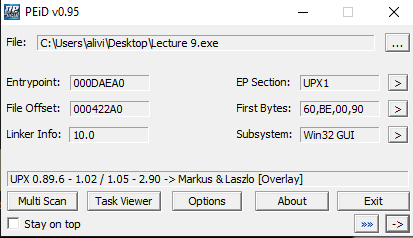
PEview shows the file was compiled on 29 Jan 2012 at 12:32:28 UTC.



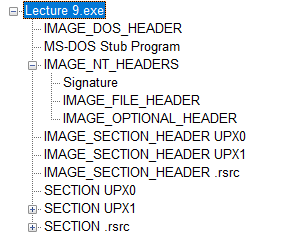
**Are there any indications that this file is packed or obfuscated? If so, what are the indicators?**

Packing:

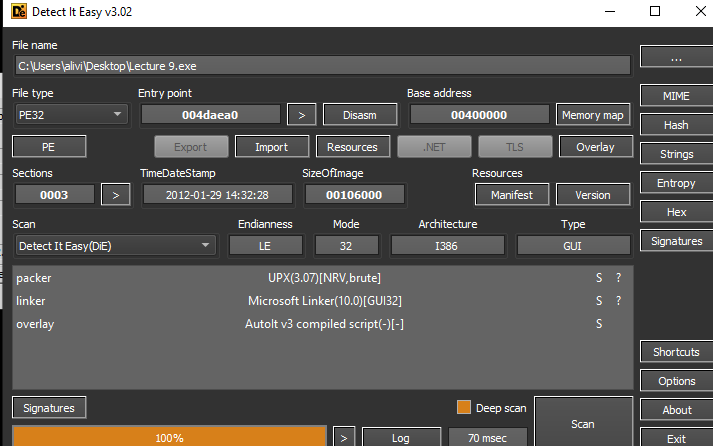
PEiD shows the file was packed with UPX.



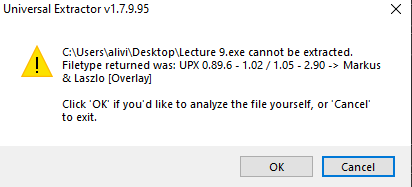
PEview also doesn’t show a .text section and instead shows UPX0 and UPX1 sections. This is similar to Week 8 malware which was packed with MPRESS.



DIE also showed the same packing with UPX and identified AutoIt as the overlay like what was identified in the VirusTotal analysis. It also shows Microsoft Linker being the linker. Note: I really like this tool and was included in the Flare VM install. Just right-click on the file in question and click “Detect it Easy”. It aggregates a lot of the tools we were taught into one GUI.

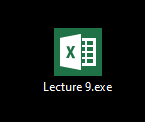


When attempting to unpack the file with UniExtract, I received an error stating that it could not be unpacked due to the UPX packing.

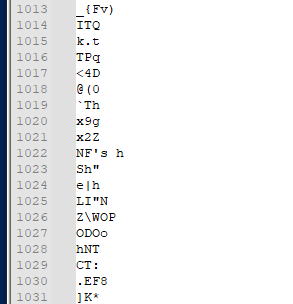


Obfuscation:

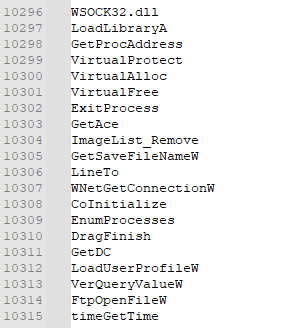
Simply looking at the icon of the file shows that it is trying to disguise itself as an Excel file.



Running strings on the file showed thousands of lines of gibberish, further pointing towards obfuscation.



Over 10,000 lines into the strings extract, there were some .dll imports as well as some external functions that were called. These functions show that this file will run some process enumeration, file opening, and potential network-related tasks, but nothing concrete.



Packing/Obfuscation Conclusion: The file is packed with UPX and is obfuscated to hide its purpose.

**Do any imports hint at what this malware does? If so, which imports are they?**

There are no .exe imports that were identified from the strings analysis. Dependencies shows multiple imports.

Kernel32.dll: Will give the file access to hardware functions and the kernel

User32.dll: Will give it access to the user interface.

Gdi32.dll: Access to functions to manipulate and display graphics.

Advapi32.dll: Indicates that core Windows components will be altered, such as the Service Manager and Registry

WSock32.dll: Access to functions to perform network-related tasks.

Wininet.dll: Access to functions to implements networking protocols.

Comdlg32.dll: Common Dialog Box Library. Implements a variety of Windows dialog boxes intended to perform common application tasks.

MPR.dll: Multiple Provider Router. Contains functions to handle comms between the OS and installed network providers.

Ole32.dll: Object linking and embedding.

Oleaut32.dll: Used by the setup program.

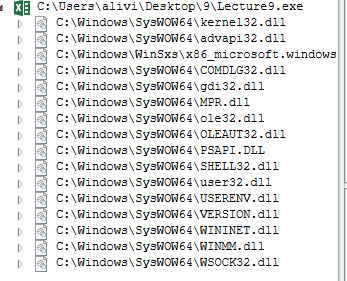
PSAPI.dll: Obtains info about processes and device drivers.

Shell32.dll: Windows shell API functions used to open web pages and files.

Userenv.dll: Creates and manages user profiles.

Version.dll: Functions to perform version checking on other libraries and supporting files.

Winmm.dll: Communicating and/or controlling multimedia devices.



That was a lot. But based on these imports and if this file is malware (ha ha), then these imports would allow the file to gather input from a user’s keyboard with winmm.dll. It also has the capability to create/modify user profiles and establish a shell connection with another machine. This gives the file the ability to manipulate the system, potentially lock it, and/or create a backdoor for the file’s creator.

**Are there other files or host-based indicators you could look for on infected systems?**

There wasn’t anything that was gathered during the static analysis that indicate any host-based indicators. The only thing that comes to mind is the fact that the excel icon the file has in conjunction with the .exe extension is indicative of a malicious piece of software attempting to disguise itself as a legitimate document.

**BEFORE you run this malware, would you consider this file malware based on your findings? "Exclude your virus total findings."**

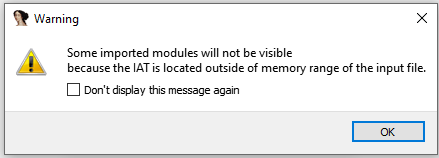
Yes. The fact that the icon shows it being an excel document but having the .exe file extension is a telltale sign of malware to have an unwitting user click on it. Plus the fact that it is packed with UPX and contains a lot of suspicious .dll imports are giveaways as well.

**\**ONLY IN THE VM\** TURN OFF NETWORKING!!!!!**

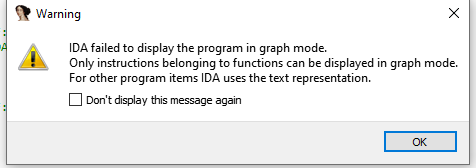
**Using the tools, we discussed in lecture 6, answer the questions below and provide screenshots from IDA Pro.**

**Provide a screenshot of any error messages this malware produces.**

A warning message was displayed upon initial loading of the malware into IDA.

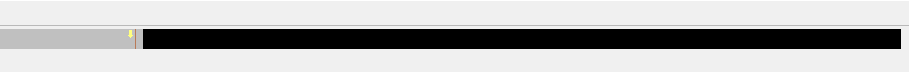


Trying to view the code in graph mode produces a warning.



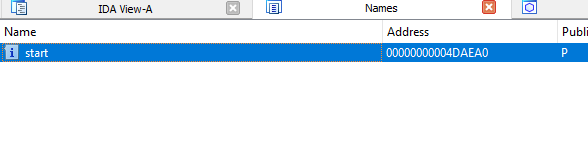
**Provide a screenshot of the navigator bar (the color-coded bar showing library functions, regular functions, code, data, etc.)**

The navigator bar was completely gray up until the last portion.



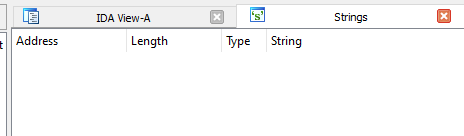
**Open the Names Window and view every function, Library, Code, String, Dara, and Linked Function. Provide an analysis of which of these are "interesting."**

The only item in the Names Window was “start”. When trying to view the function in graph view, a warning message appears and it cannot be viewed in such a way (the warning message screenshot was given above).



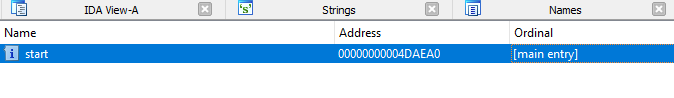
**Open the Strings window and provide an analysis of which ones are interesting and why.**

There was nothing in the Strings window. This is interesting and most likely has to do with the UPX packing. It could also be from the AutoIt language the file was written in and doesn’t translate well to IDA.



**Open the Imports and Exports windows and provide an analysis of your findings.**

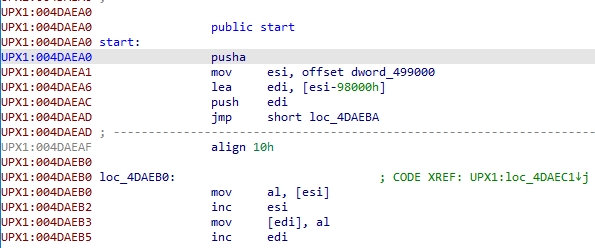
The only item within either of these windows was “start” and was located within the Export window. The Imports window was completely empty despite the functions that were identified during the static analysis. Once again, I believe this is due to the UPX packing that the file was packed into and IDA cannot sufficiently read the proper information.



**Open the file in IDA pro and identify the following:**

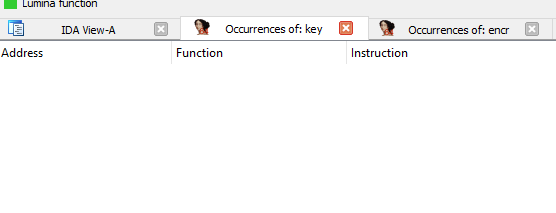
What does IDA identify as the START of the program

IDA Identifies the code beginning at 0x004DAEA0 in the UPX1 section as “Start”. This is because this is the location that IDA opened up to initially and identified as the main entry point in the Exports window.



**Can you identify any sections that appear to be handing encryption or Key generation**

I am unable to identify any sections in particular that handle encryption or Key generation due to the massive amount of data within the code that is encrypted and packed with UPX. Any references to Windows Cryptography functions in conjunction with file traversal would be a dead giveaway. However, the code appears to be sufficiently obfuscated to hide any static identification of sections that handle encryption or key generation. A search of “key” and “encr” within the code yielded no results.

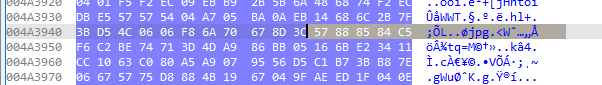


**When the file is run, a background image is visible, where is this image stored within IDA?**

After running the malware, the background changed to the image below and an error window popped up referencing the AppData\Local\Temp folder.

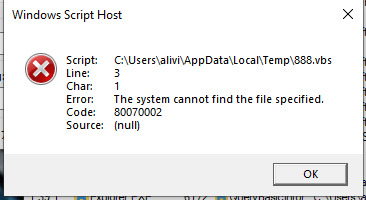


The Temp folder contained a file titled “wl.jpg” and is identical to the background image. This name, the .jpg extension, and the Temp folder are hints as to where this file are stored within IDA. Within the hex view window, a text search for “jpg” was ran and only one occurrence of it existed at around 0x004A3940. Therefore, it is reasonable to conclude that this is the memory address of the background image.

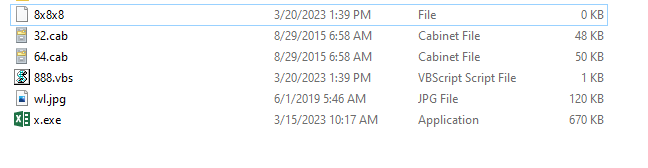


**Run the file and record your observations. This malware may require you to reset your box a few times, this is nasty malware, don't enable networking, and run fakenet.**

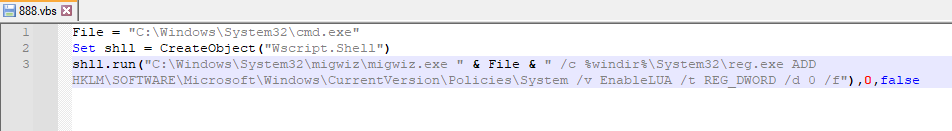
When the file is run, the background changes to the image above. An error message also popped up showing where a file was attempted to be copied into a directory.



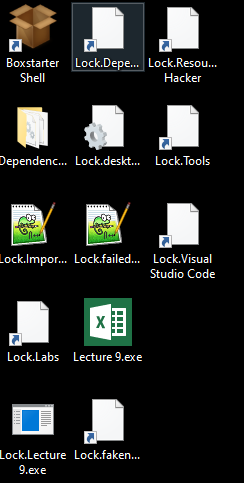
Within the Local\Temp directory, some new files were located, including another excel icon with a .exe extension, a script file, some compressed files containing cryptbase.dll, and the wl.jpg image that has made itself the desktop background. I assume the 32 and 64.cab files are for 32-bit and 64-bit windows builds.



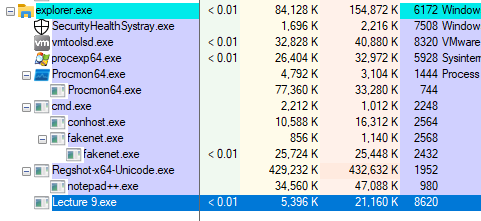
The 888.vbs script showed shell code to create a shell object. It also shows it calling migwiz.exe which is the file and settings transfer wizard. The EnableLUA is also set to false in the registry which means that the user will not be notified if programs try to install software or make changes to the computer.



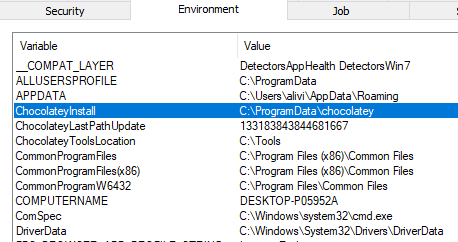
Files were also prepended with the name “Lock” and were encrypted or could not be accessed. It wasn’t just files, but also applications. The files that were “locked” were all over the system and not just relegated to the desktop.



After the file was executed and files were encrypted, it still showed itself running in Process Explorer as a child process of Explorer.exe.

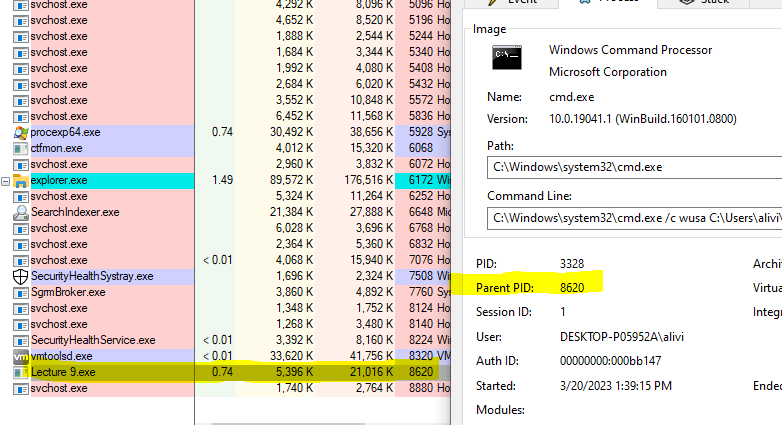


Within the properties of Lecture 9.exe in Process Explorer, the “Environment” tab shows that it is installed and is using a program called “Chocolatey”. It is an open-source package installer for windows (like apt for Linux) and has been known for its use in malware.

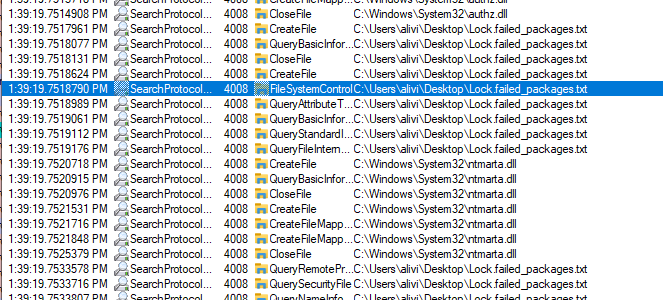


Within the \lib directory in the chocolatey folder, there were a lot of familiar-looking files that it has, such as Boxstarter.

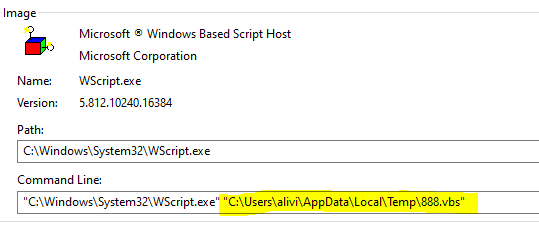
Within procmon, there were a lot of instances of cmd.exe. Inspecting this, the parent PID is Lecture 9.exe.



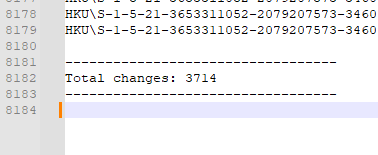
There was also a lot of file traversing behavior captured with procmon through the use of SearchProtocolHost.exe.



Applying a Parent PID filter to procmon for Lecture 9.exe, it only had 2 children: cmd.exe and WScript.exe. WScript.exe executed the 888.vbs script found in the \Temp folder.

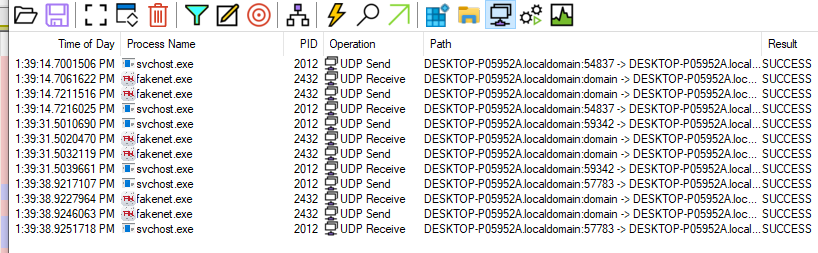


There were 3714 registry changes noted when using Reg Shot.



**What network-based indicators could be used to find this malware on infected machines?**

Fakenet did not detect any network-based indicators that deviated from the baseline nor did procmon.

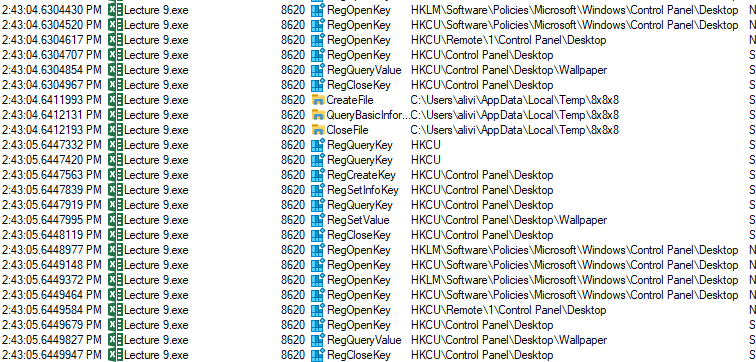


**What would you guess is the purpose of this file?**

This is very clearly a type of ransomware. It encrypts files but does provide the user a method for which to unlock them via payment. It appears to simply be destructive without the purpose of scamming money.

I believe that the malware attempts to disguise itself as an excel document and is packed/obfuscated enough to where it is difficult to determine the true purpose without a decryption key. The malware installs numerous packages and uses native Windows processes to inflict its destructive payload. It installs packages using Chocolatey. I suspect that since it uses a Shell, it potentially has the ability to exfiltrate data or do a remote logon. I also suspect there might be a keylogging aspect and stores the data in the 8x8x8 file. I could not find evidence of this, though.

Some kind of data is stored in the 8x8x8 file because Lecture 9.exe continues to run in the background. It goes through a cycle of accessing the registry and the querying the file. Based off of the paths that it is accessing, it appears to want to ensure that the desktop wallpaper remains the same.



**What programs do you see open, what is the malware trying to do?**

See above.

**Now, how often would you be able to run this VM, excluding snapshots?**

I wouldn’t be able to run it for the purposes of analyzing malware due to that all of my tools were locked.

**What would your recommendation be to Management? Do we need to stop generating revenue and cleaning, or can we go on and clean as we go?**

We need to stop generating revenue and clean the machine. We need to check if any other machines were infected by this malware, but it appears to be unlikely due to the fact that no network activity was observed. The staff needs to be warned of downloading and clicking on random files, especially if the icon LOOKS like a familiar document but is appended with a .exe file extension. This kind of training is important for employees to recognize how malware can disguise itself.

**Can you clean the system, and if so, how would you do it?**

No. I don’t know how to decrypt the locked files.

* Go to two other student's posts and observe their findings. Post if you agree or disagree with the results.
  + Suppose you were working on this malware to see if this could be allowed in your organization. Did your analysis provide enough detail to make this determination?
  + Did they find something using a new technique, and if so, would you use this next time?
  + From a business perspective, if you were both being paid at the same rate. Would your analysis be more cost-productive and achieve the same results?