Malware Analysis Tools & Techniques Assignment

Malware 1:

Type	32 bit Windows Executable (pr2.exe)
Filename	Wannacry.Ransomware
Md5hash	db349b97c37d22f5ea1d1841e3c89eb4
URL Download	https://www.ghidra.ninja/samples/wannacry.zip

Malware 2:

Type	Malicious word document
Filename	99eb1d90eb5f0d012f35fcc2a7dedd22293127943548436 37ebb7f40b74d0809.doc
Md5hash	16ba8f5d604b4b9a366ae2d5b2107e68
URL Download	https://github.com/InQuest/malware-samples/blob/master/2018-04-GandCrab-Swarm/99eb1d90eb5f0d012f35fcc2a7dedd22293127943548436 37ebb7f40b74d0809/99eb1d90eb5f0d012f35fcc2a7dedd22293127943548436 37ebb7f40b74d0809.doc

By Ryan Ng and Zhe Yu Chua

Table of Contents

Lab Setup

It is crucial that the malware sample is separated from the host system to prevent any damage to the host system and hence a virtual machine will be used to analyse and execute the malware in a sandbox environment.

The malware sample will be placed in the virtual machine and any damage to the virtual machine by the malware can be reverted with snapshots.

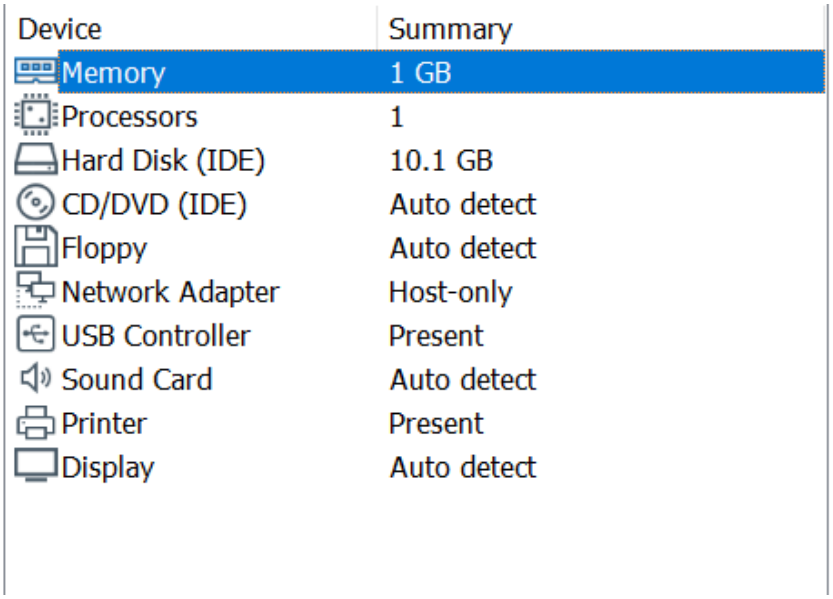
1.1 VMware Setup

Version of VMware: Workstation 15 Pro

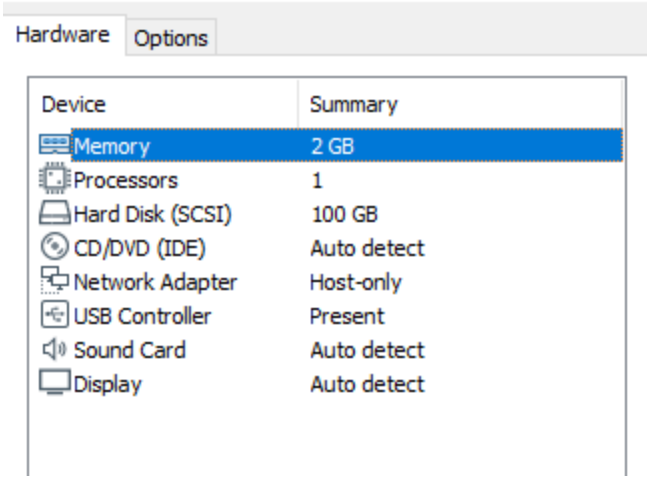
Host OS: Windows 10

Guest OS (Malicious Windows Executable): Windows XP

Guest OS (Malicious Document): Windows 8



VMware Workstation Pro 15 is installed and running on the Windows 10 Host Machine. The virtual machine operates on Windows XP with 1GB of Memory, 10GB of hard disk space and uses Host-only adapter mode.

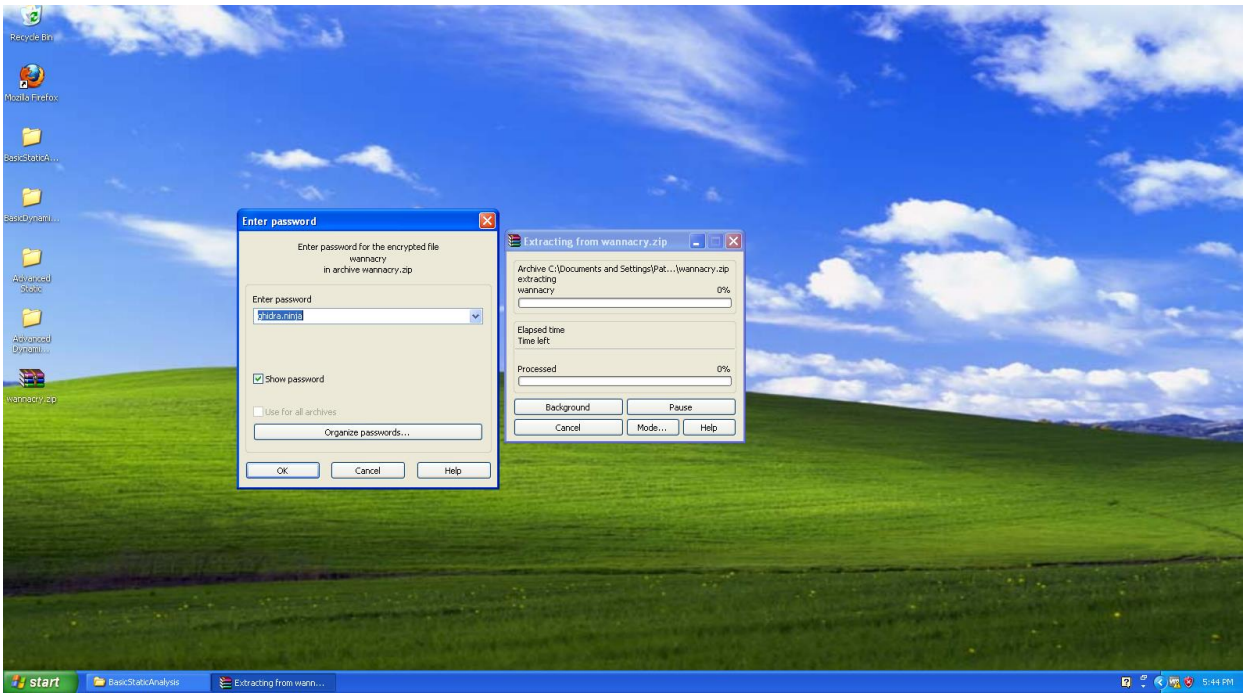


For the second virtual machine, it operates on Windows 8 with 2GB of Memory, 100GB of hard disk space and uses Host-only adapter mode.

1.1.1 Specific Setup for pr2.exe

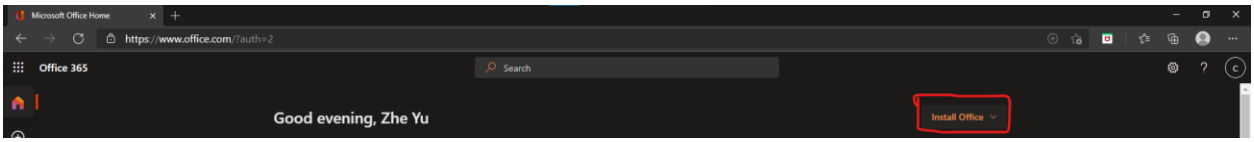
The malware is first downloaded from the link:
https://www.ghidra.ninja/samples/wannacry.zip

Thereafter, it is placed into the Windows XP machine and the executable is extracted from the zip file using the password “ghidra.ninja”. Thereafter, the extension should be added (i.e. “.exe” for the file to be run).

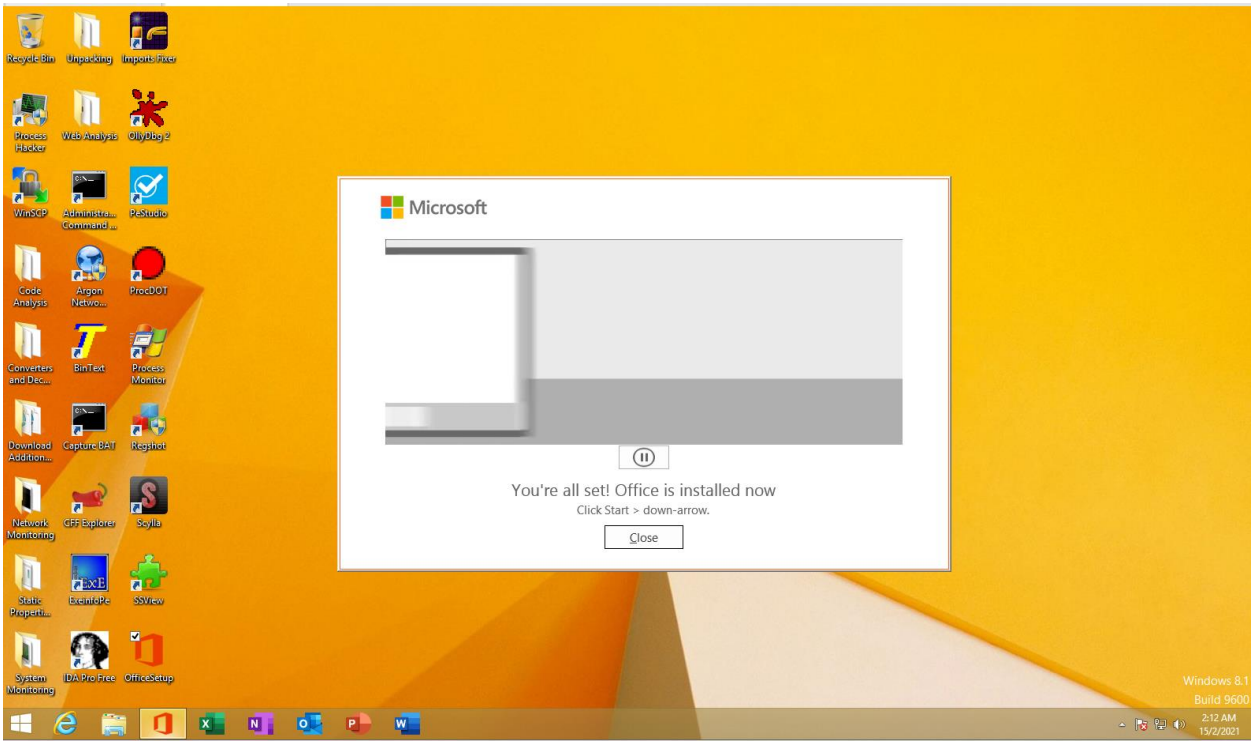


1.1.2 **Specific Setup for GandCrab-Swarm**

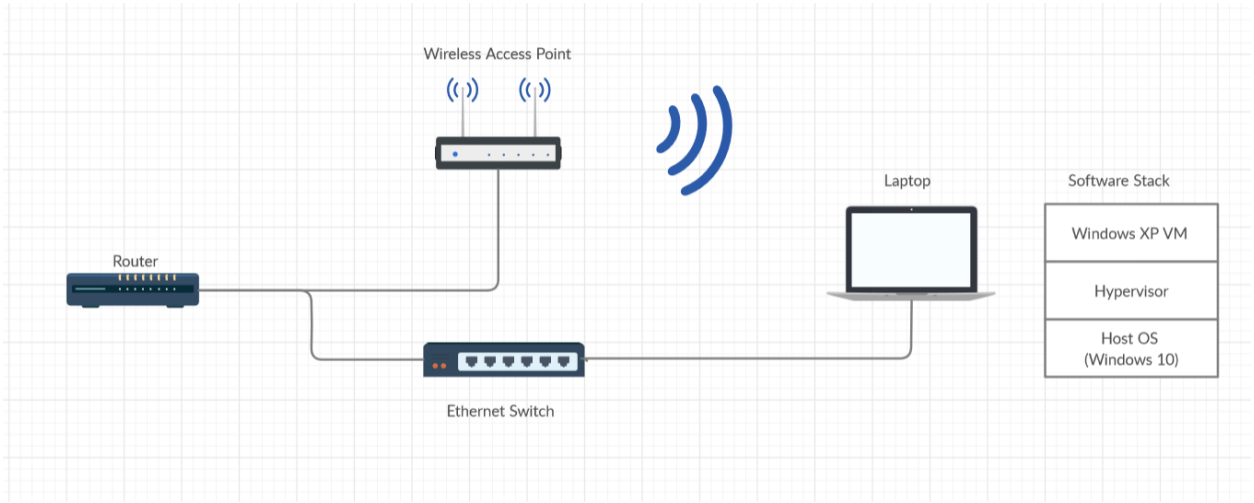
Install office from the office 365 website



Copy the Office Setup file into the Windows 8 VM and set the network setting to NAT for it to download



1.2 Network Diagram

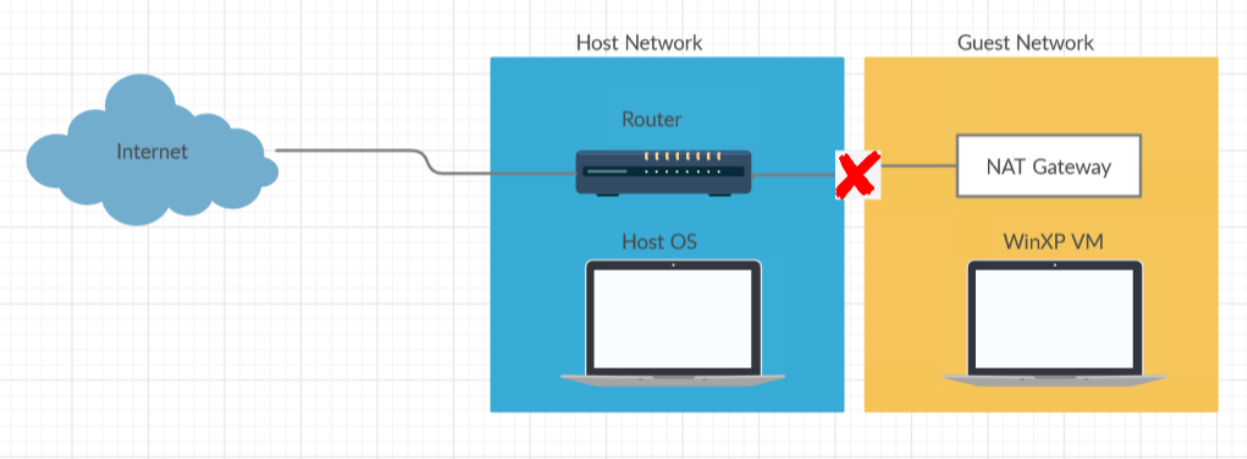


Above shows the physical setup. The laptop is either connected to the switch, which is in turn connected to the router via a switch physical ethernet cable or through a wireless access point connected to (or sometimes built into) the router. There is a Windows XP Virtual Machine as depicted above running on the host computer by using the

hypervisor will allow for virtualization software to be run. The malware needs to be executed for dynamic analysis in an isolated environment like the virtual machine so it does not pose any dangers to the Host OS potentially getting infected. Using the VMWare Workstation Pro VM Manager also allows us to take snapshots, which is critical so that we can revert back to the original state before dynamic analysis and rectify potential issues caused by the malware (i.e. encrypting the files).

For the analysis of the malicious document, the same concept is used. As opening the malicious document for analysis in the host machine can be dangerous and the document may potentially install some malware once it has been executed. As such, a virtual machine is used to ensure that the malware is isolated in a safe environment and that the Host OS will not be infected. VMware Workstation Pro is also used because of its ability to take snapshots, which we can revert to the original state of the VM after dynamic analysis of the malware is done.

1.3 Network Configuration



The host machine acts as a router for the Guest Network and this is done through a NAT Gateway on the Guest Network, which allows the Guest Network to use a private IP leased by the Host Network. In this manner, both Host and Guest can communicate with the Internet. However, due to safety precautions, we were advised to disable Internet connection on the Guest (Windows XP/Windows 8) machine.

Malware Analysis Tools

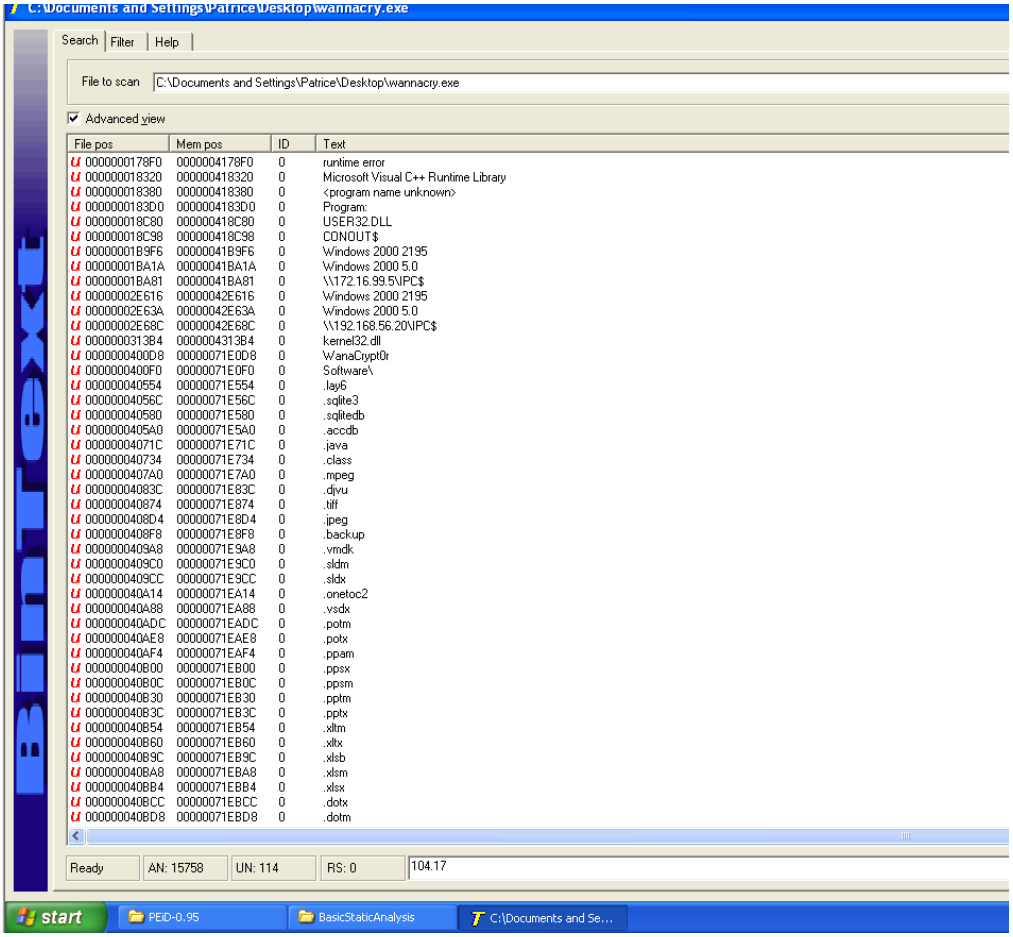
We will use various tool in the assignment to analyse the malware and malicious document. This section will discuss about the tools used for static and dynamic analysis.

2.1 Basic Static Analysis

Basic static analysis refers to analysing the malware without executing it. This can mean looking through the strings and functions imported and exported by malware. We can also use basic static analysis to look for any host base or network based indicators, for us to get a sense of what the malware may be doing to the victim machine. Host based indicators refers to files, processes that a malware may create and changes to the registry keys. Network based indicates refers to URLs, IP addresses that the malware may be using to establish an external connection.

Basic Static analysis may not reveal the entire functionality of the malware and Basic Dynamic analysis needs to be done to reveal the functionalities of the malware.

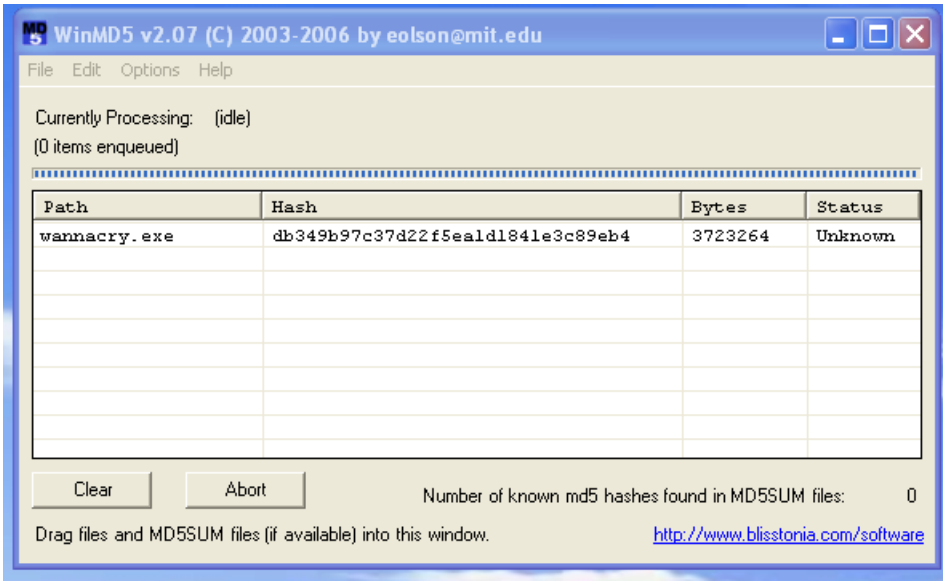
2.1.1 BinText



BinText is a powerful tool that is able to find strings in files, which includes ASCII text, Unicode and resource strings. By using BinText, we are able to find host based indicators and network based indicators through the API calls and this will allow us to have an understanding of what the malware is trying to do. The strings can also be compared with the results from other tools to ensure that the results are consistent.

BinText can be downloaded from the website [http://b2b-download.mcafee.com/products/tools/foundstone/bintext303.zip.](http://b2b-download.mcafee.com/products/tools/foundstone/bintext303.zip)

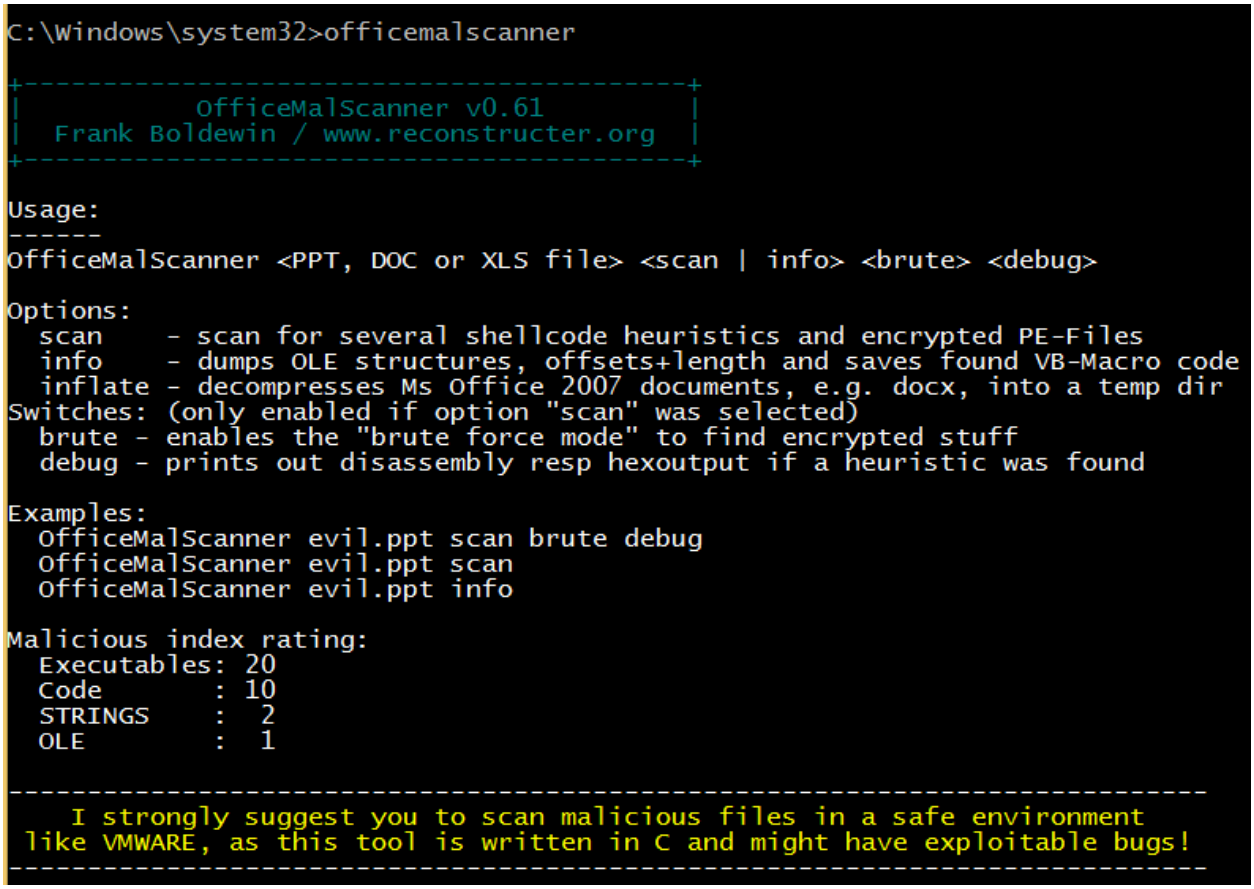
2.1.2 WinMD5



WinMD5 is a tool that is used to compute MD5 hash value for files. For this assignment, it is used to make sure that the malware does not change itself after execution or analysis and ensure that the malware we are analysing is still the same one.

WinMD5 can be downloaded from its website https://www.winmd5.com/

2.1.3 Officemalscanner



Oficemalscanner is used to scan office documents and extract macros from the document if the document has any. For this assignment, it is used to extract the macro from the malicious document so that it can be further analysed.

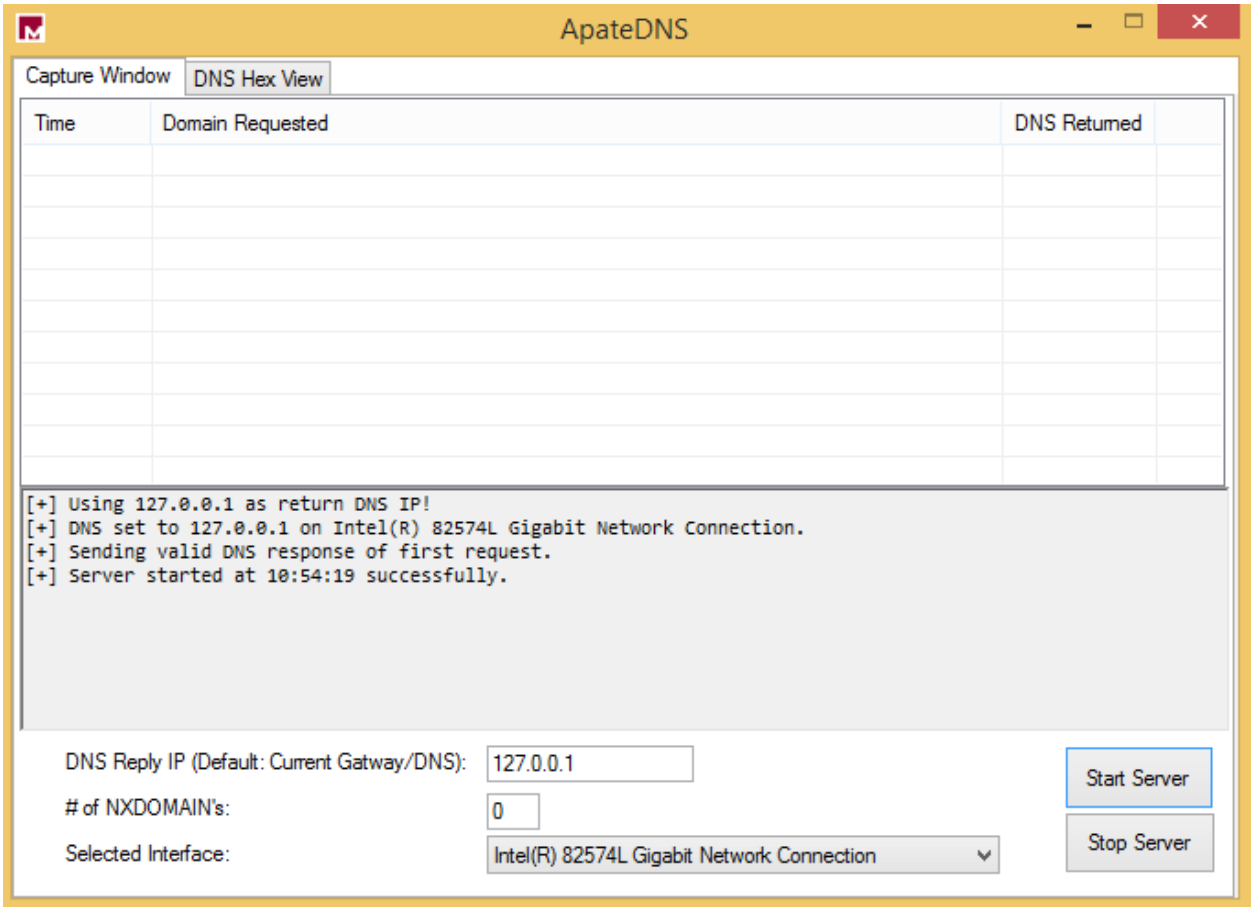
Officemalscanner can be downloaded from http://www.reconstructer.org/

2.2 Basic Dynamic Analysis

Basic Dynamic Analysis is able to give us a clearer insight as that what the malware is trying to do and check if anything has been left out by Basic Static Analysis. In Basic Dynamic Analysis, we are able to look at running processes, any DNS request being made by the malware to establish and registry changes.

This will build on the information gathered through static analysis as some of the malware functionality may be have been missed or that during static analysis, the strings are heavily obfuscated which makes determining the functionality of malware challenging. With Dynamic Analysis, it can show the functionality of malware as there may be visible changes and we can also view the processes and DNS requests made by the malware in the background and the registry changes to find if any modification were made to the keys.

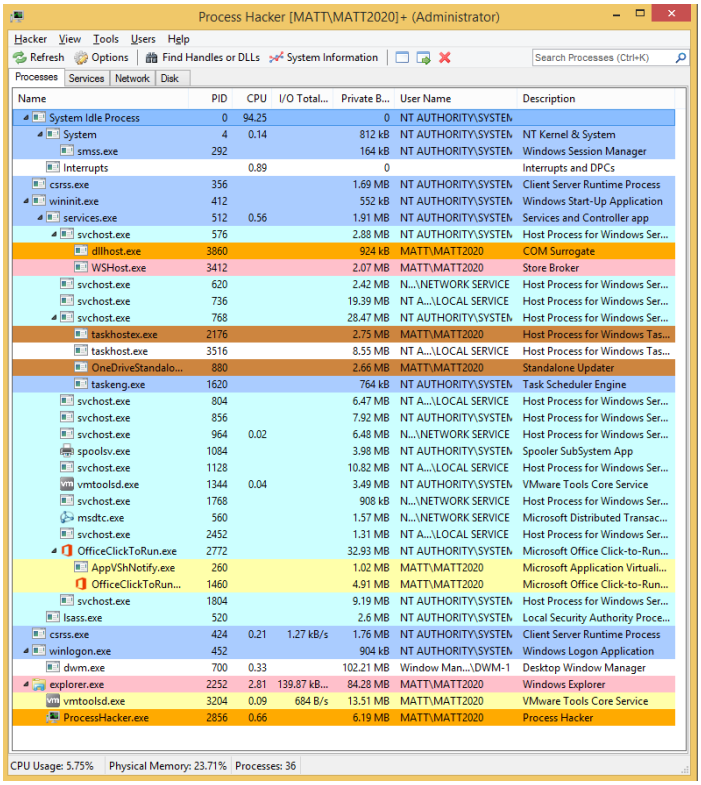
2.2.1 ApateDNS



ApateDNS is used to control DNS requests. It will reply to DNS queries made by the system and is very useful for dynamic analysis as it can discover IP addresses and host names that were not found during static analysis. This will prove very useful for the network analysis.

ApateDNS can be downloaded at
https://www.fireeye.com/services/freeware/apatedns.html

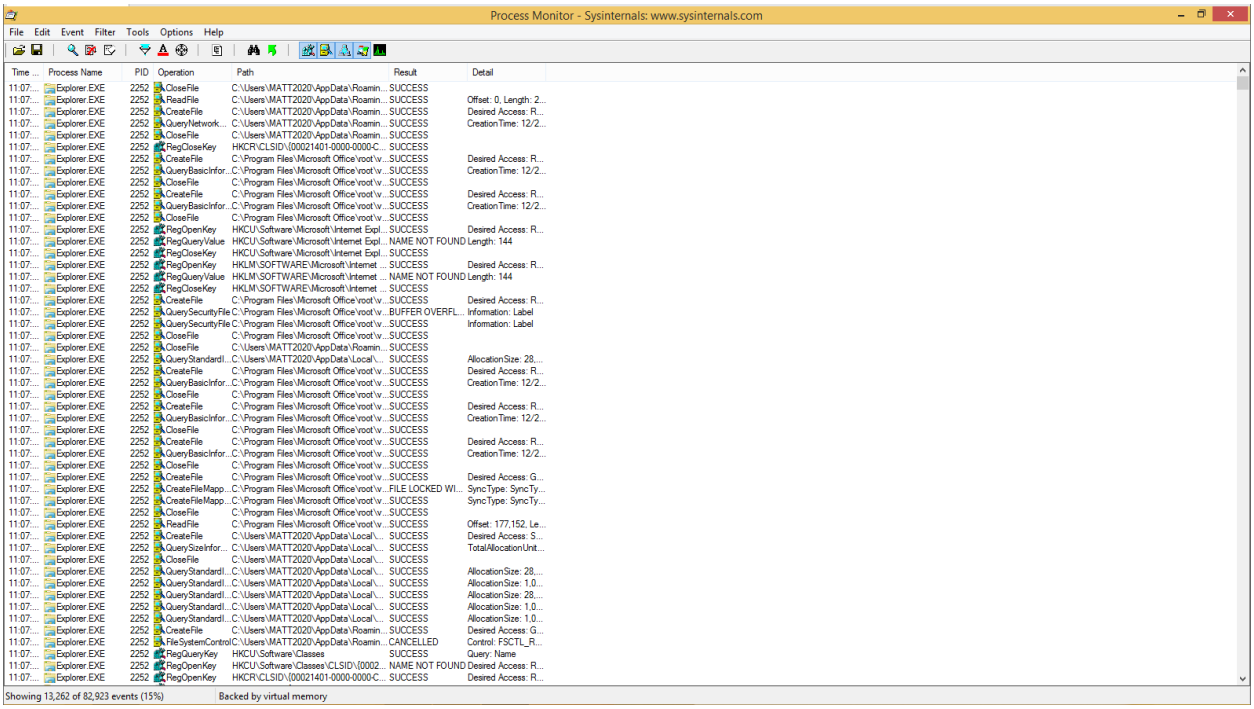
2.2.2 Process Hacker



Similar to process explorer, process hacker allows us to gain an insight into the running processes of the system. For Dynamic analysis it allows us to gain an insight into the running processes of the system when the malware is executed. It has useful information of the such as the PID of the process which we can use for the next tool.

Process Hacker can be downloaded at
https://processhacker.sourceforge.io/downloads.php

2.2.3 Process Monitor

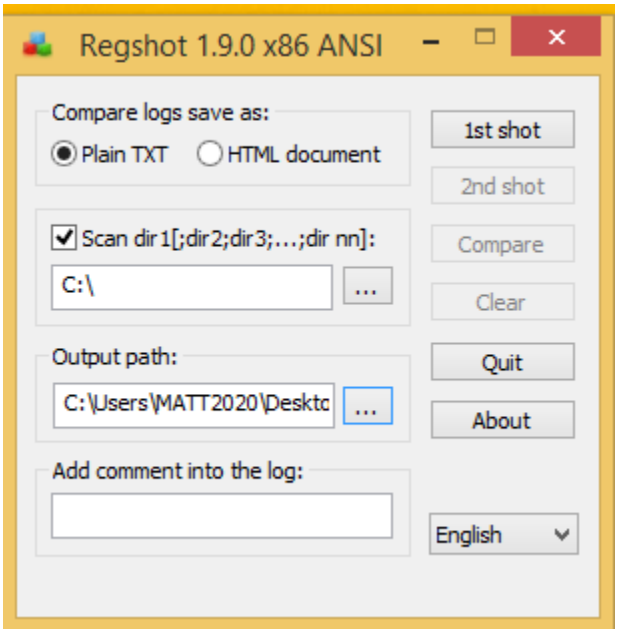


Process Monitor is a by Windows Sysinternals to monitor and display all real-time file activity from a Windows or Unix-like operation system. It is a combination of two legacy tools, FileMon and RegMon.

Process Monitor has some uses such as the filtering of Processes by name or PID, which allows us to look at the processes we are interested in rather than looking through all the processes in the machine.

Process Monitor can be downloaded at https://docs.microsoft.com/en-us/sysinternals/downloads/procmon

2.2.4 Regshot



Regshot is a tool that allows us to see what registry changes were made by the malware. It can that snapshots of the machine before and after running the malware and allows us to compare the results of both in a notepad. This is very useful as malware usually make Registry Changes after running and from the registry changes we are able to get an insight of the malware functionality.

Regshot can be downloaded at https://sourceforge.net/projects/regshot/

Malicious Windows Executable Analysis—pr2.exe (Wanancryptor)

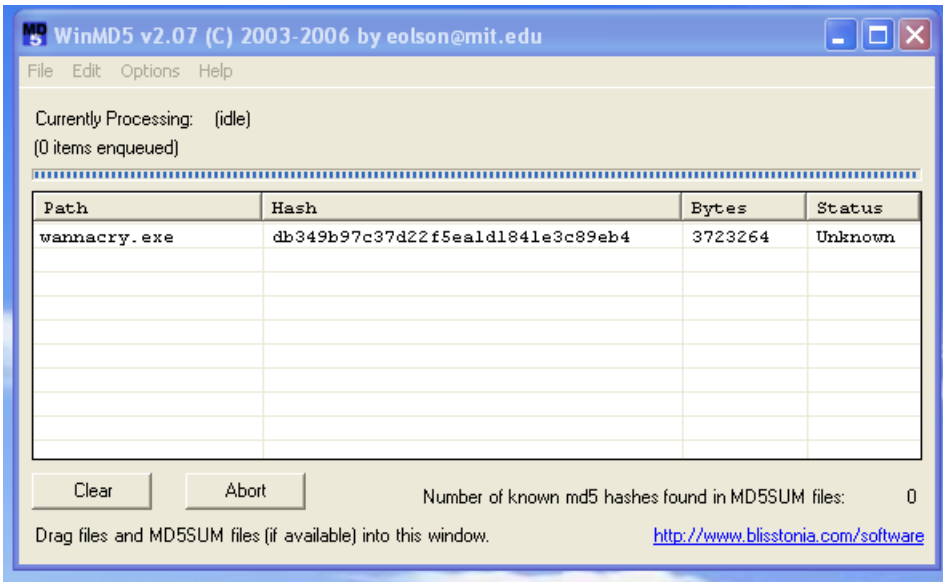
3.1 Basic Static Analysis

Our first step in analyzing the malware is to conduct basic static analysis, which use various freely available for open-source tools to help us understand the malware behaviour without the need for dynamic execution to pick up properties like the type of packer (if any), imports, strings—which may include host or network-based indicators of compromise. In this case, the Windows XP and Windows 8 machines will be used.

3.1.1 WinMD5

The first tool we will use for static analysis is WinMD5. This serves as the GUI equivalent of the md5sum Linux command. Checking the hash value of the malware to be analysed is important since we want to make sure it is the correct version that we are

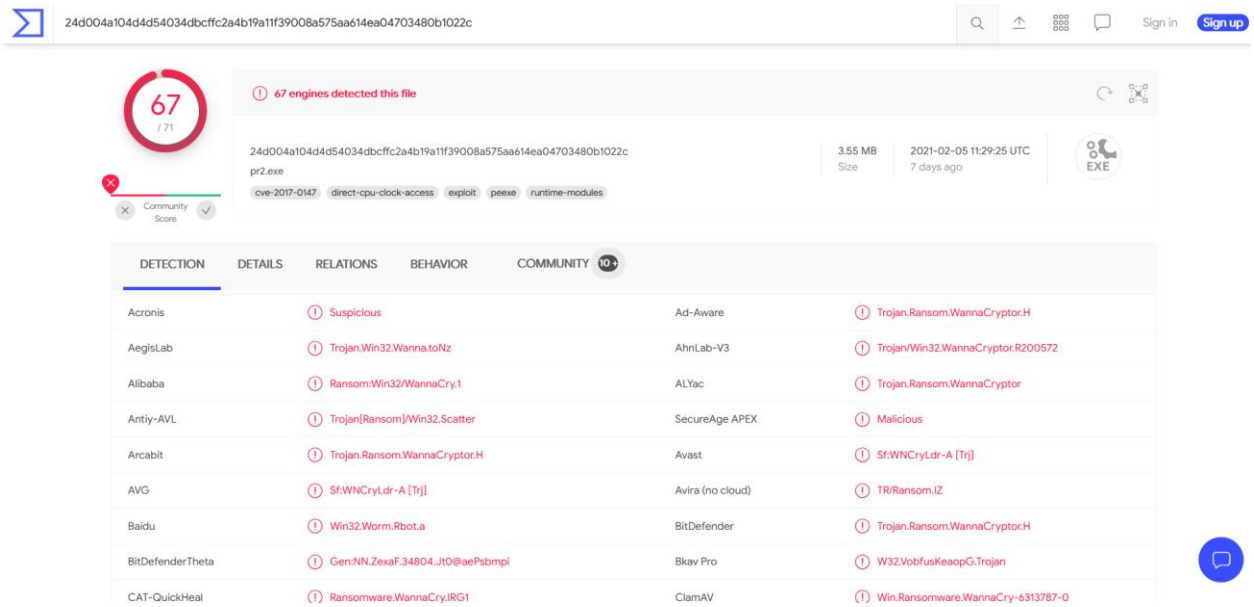
analysing (for instance WannaCry has several variants, some which include the worm, which will connect to the network and perform further infection on other devices and the non-worm versions).



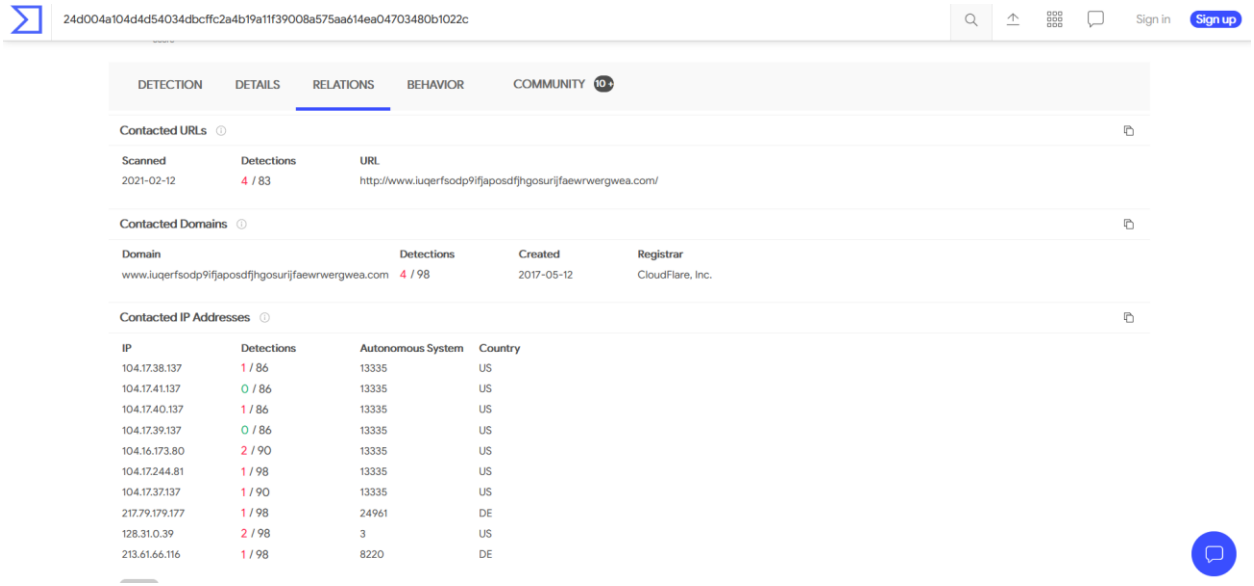
In this case the hash is db349b97c37d22f5ea1d1841e3c89eb4, which is identical to the one downloaded from the website, thus we can be sure we downloaded the correct malicious executable to analyse.

3.1.2 VirusTotal Information Gathering

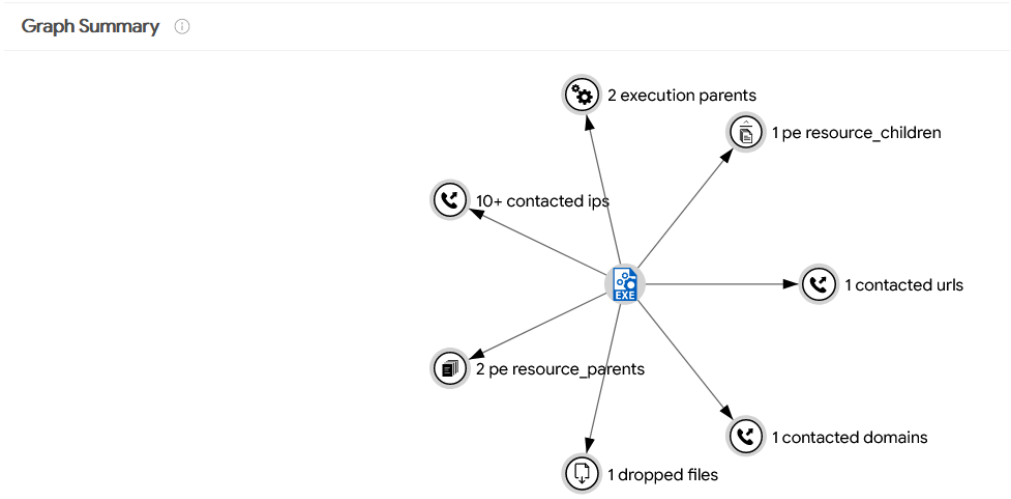
Next, we will check if this malware is registered in the signature database of various antivirus (AV) companies. Out of 71 registered AV engines on virus total, 67 picked it as as the wannacryptor program, thus there is a high likelihood that this is indeed a real sample used in the wild during the 2017 attacks that caused widespread disruption to services.



On the relations tab, we can view the killswitch url as well as the connected IP address, which means that there is a high probability of the malware connecting to a C2 server for instructions and likely network-based indicators that we need to find later during the analysis of the malware using various tools.

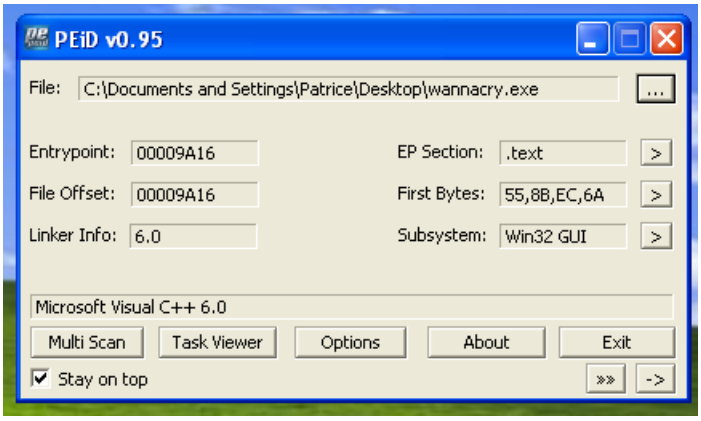


VirusTotal also provides a graph to summarize actions that the malware will perform, including its parent and children PEs, contacted IP addresses, urls and domains, which maybe useful later during dynamic analysis to observer the files created and urls and domains it connected to.



3.1.3 PEiD

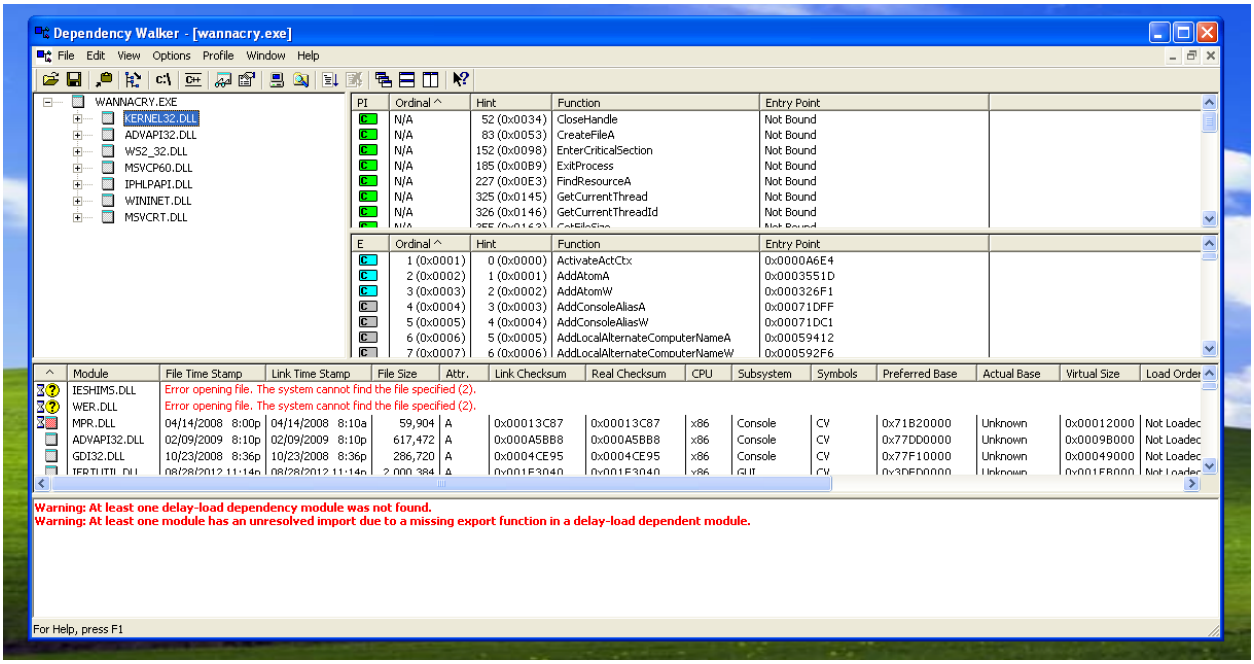
The next tool that we will use is PEiD, which allows us to analyze the compiler used to compile the malicious executable by the malware author, as well as the presence of any packer used for code obfuscation to make advanced dynamic analysis difficult.



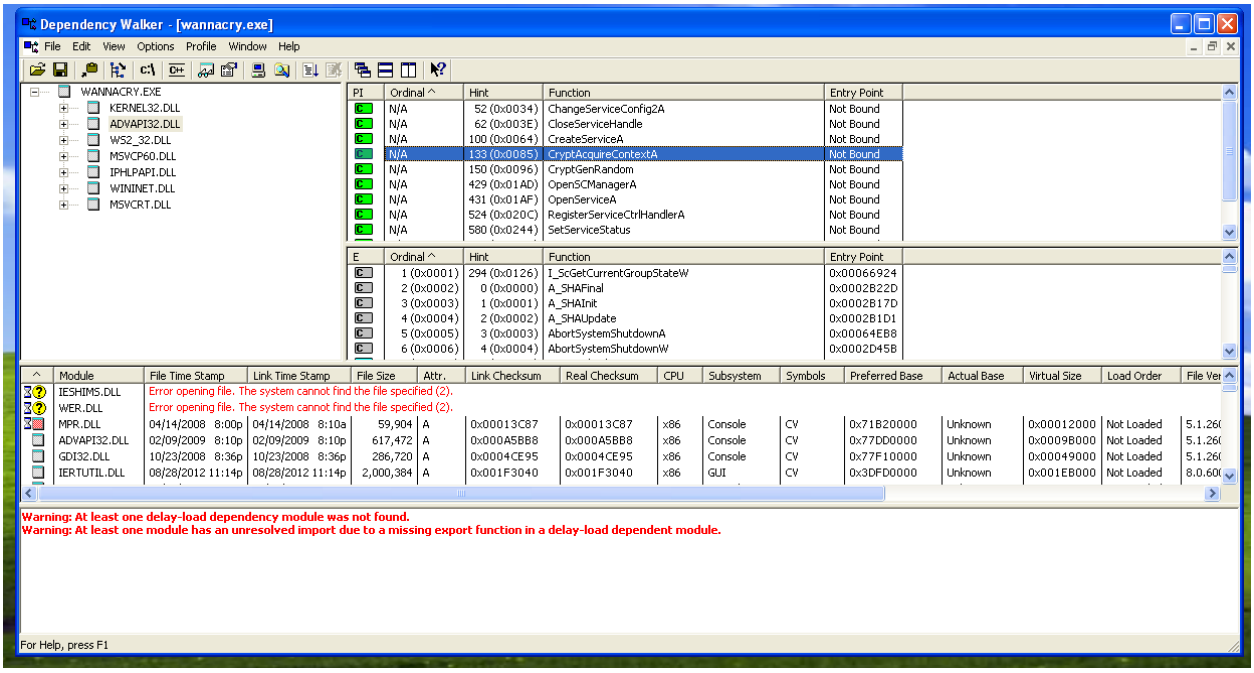
The results of making PEiD identify wannacry is as follows: The malware was compiled using the Microsoft Visual C++ compiler, suggesting it was written in C++. Since PEiD was able to identify the compiler used and there not being any packer name or there not being any information found.

3.1.4 Dependency Walker

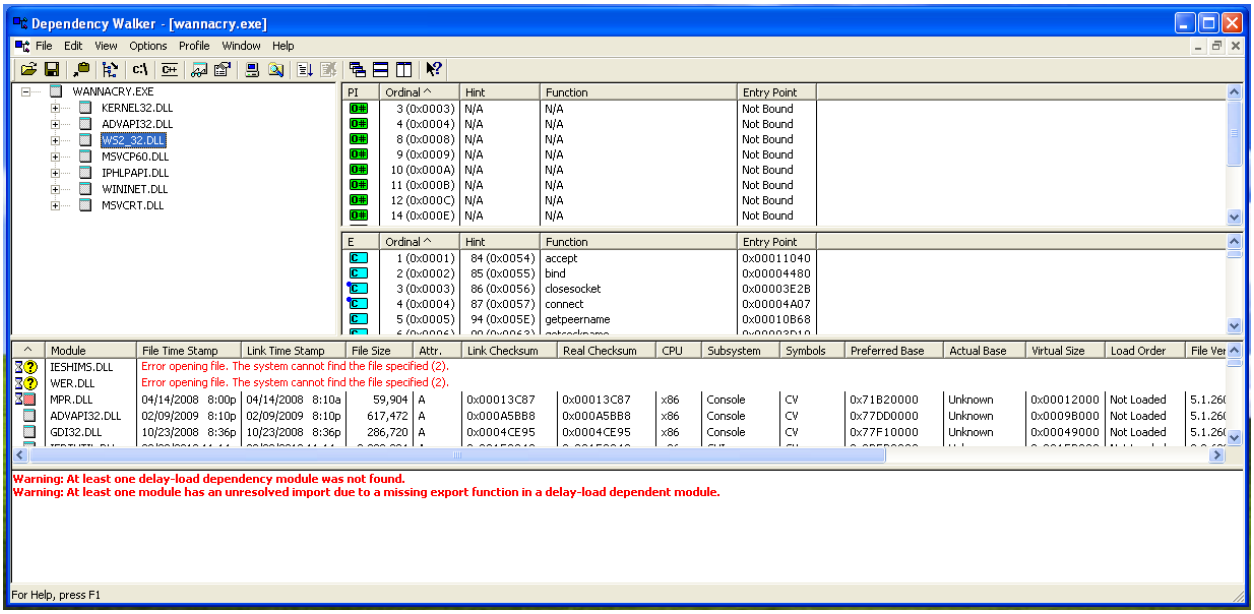
The next tool is dependency walker, which helped us understand the various Windows-specific API calls that the malware uses to gain access to various function imports. In general, wannacry.exe imports 6 APIs and some of the more significant imports are listed below:

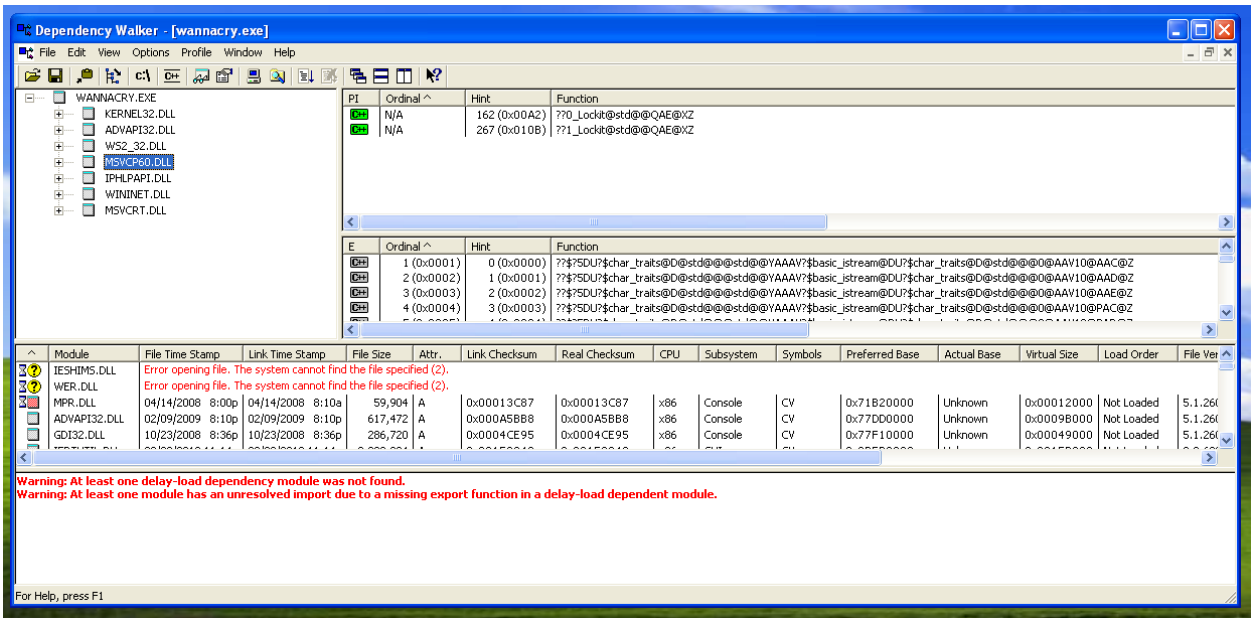


Kernel32.dll: FindResourceA—which discovers a resource that the malware may use, LockResourceA—which restricts access to the resource and Sleep—which allows a timeout increase the chances of the malware of slipping under AV's radar.

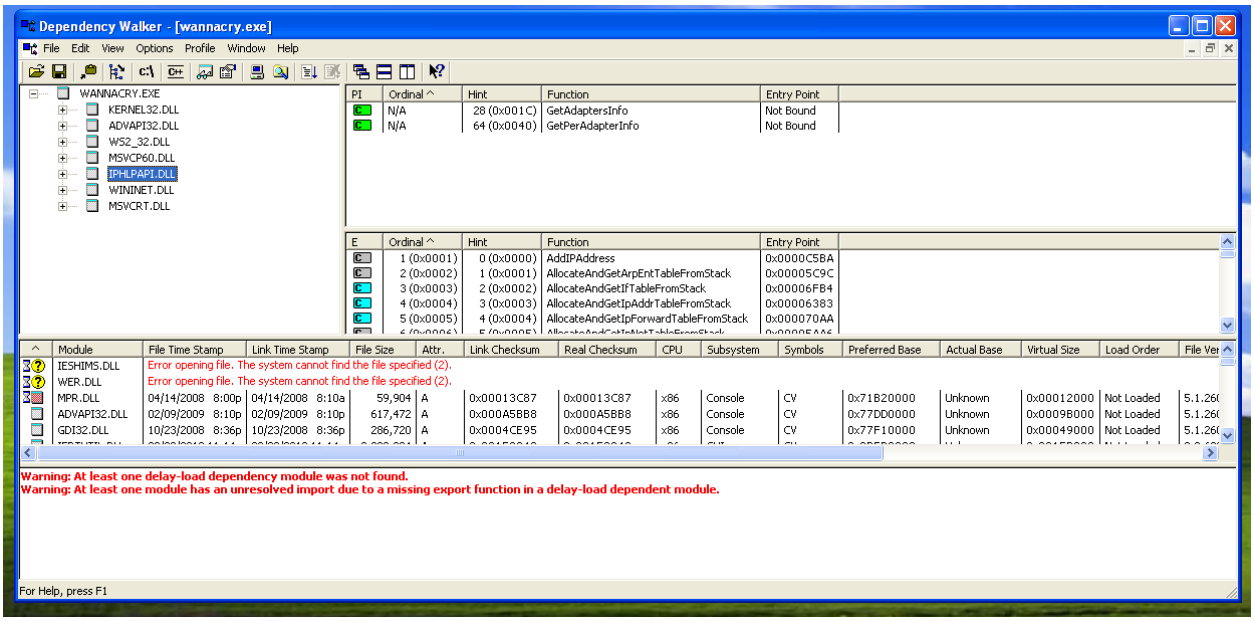


Advapi32.dll: CreateServiceA—which is suspected to enable services which the malware uses, CryptGenRandom—which allows the ransomware functionality to commence and encrypt the user's data. OpenScManagerA—which opens the Microsoft Security Center Service Version 2 or mssecsvc2.0 on the specified computer as mssecsvc.exe, which is an executable file. This is important because there is a need for the malware to call the function before manipulating the services by calling other functions.

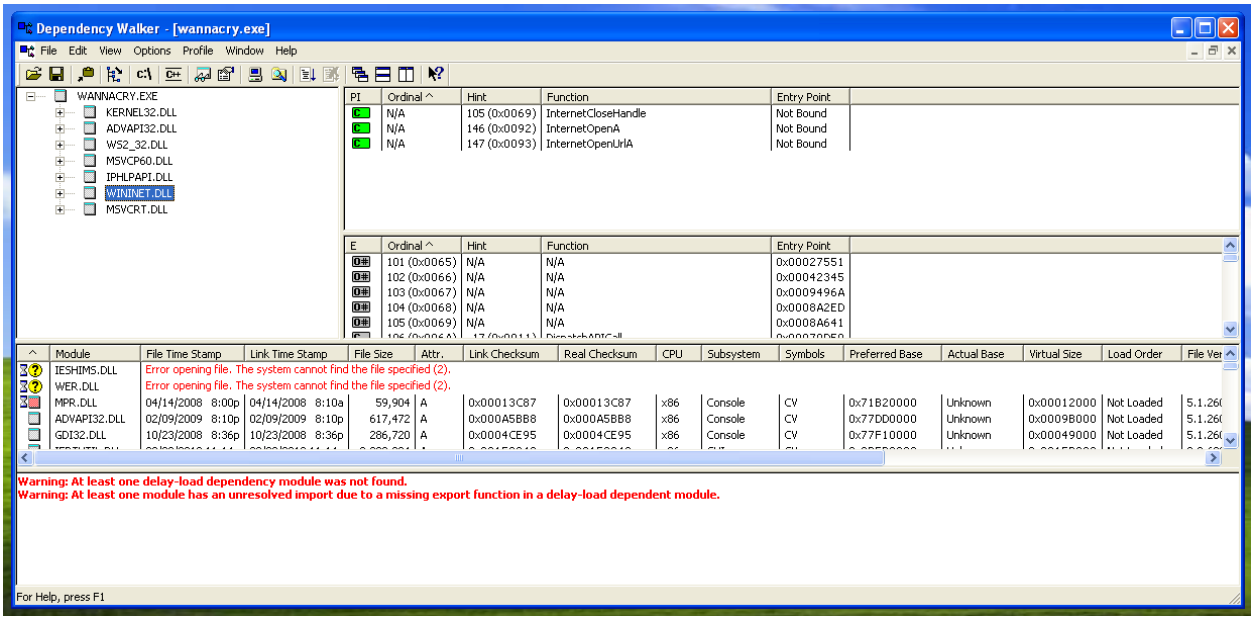




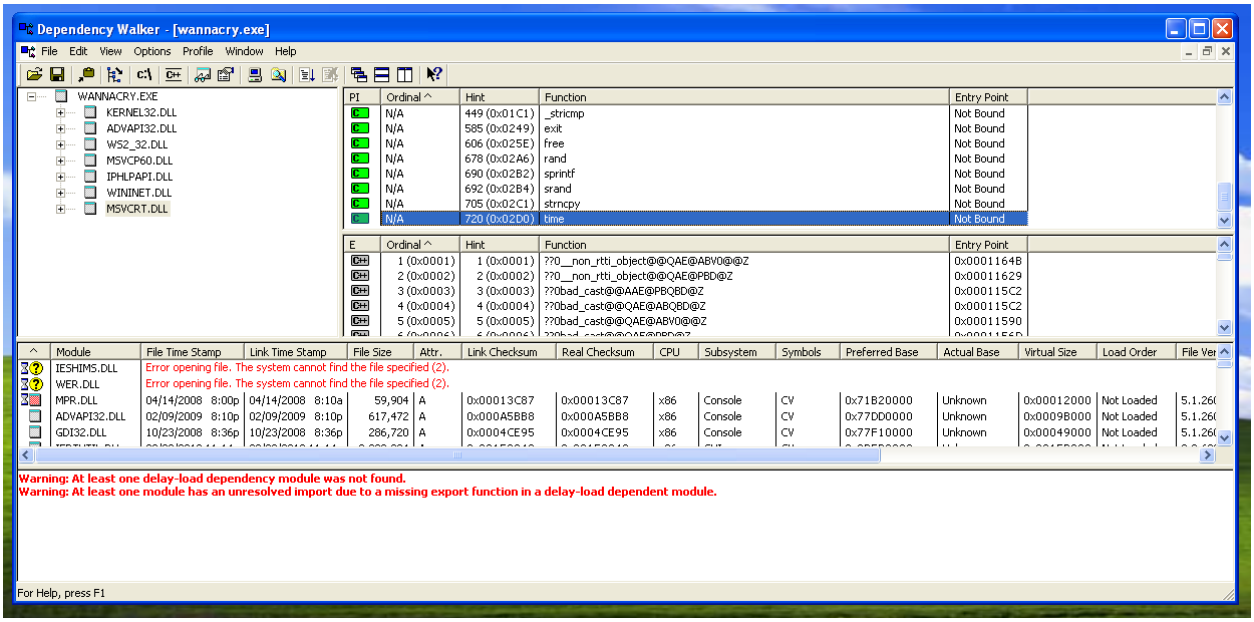
WS2_32.dll and MSVC60.dll: No imports



IPHLPAPI.dll; Has 2 imports that are most likely related to the worm functionality, since it gathers information about the wired/wireless internet adapters on the machine, which is probably sent back to the C2 server for analysis and depending on the type of adapter and its vulnerabilities, further attack vectors can be established using other malwares.

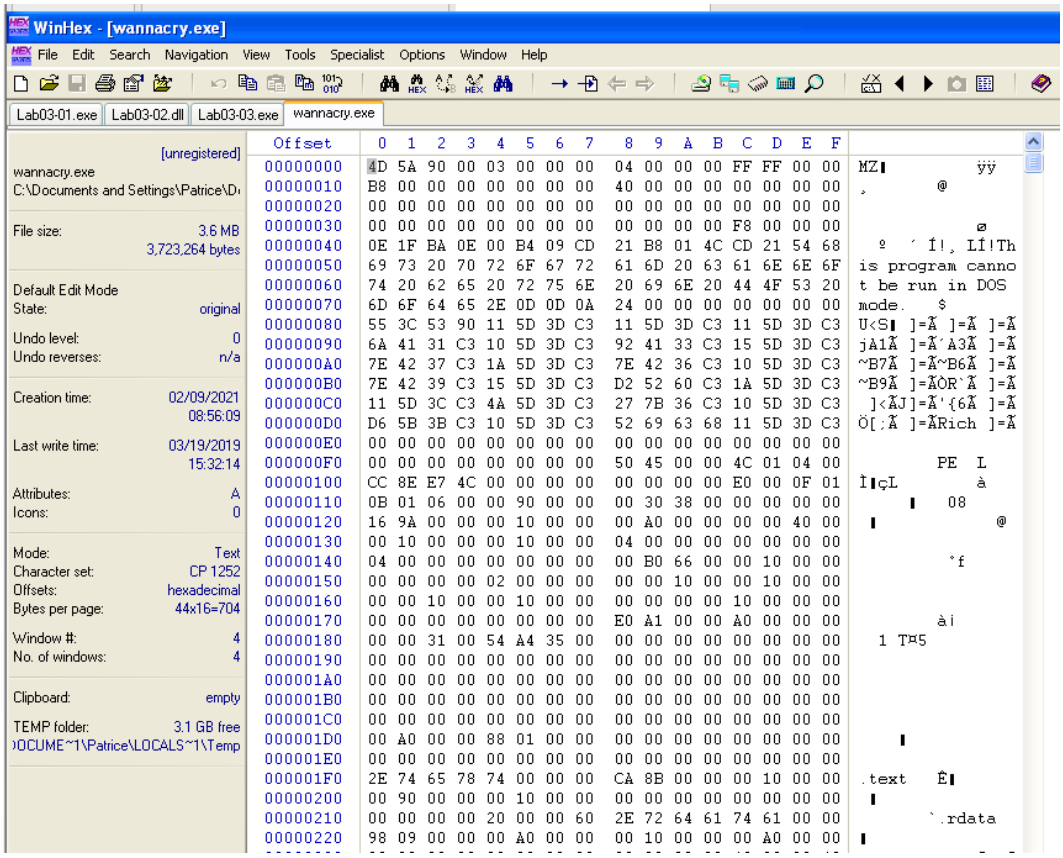


Wininet.dll: All imports in this dll have to do with internet related processes, from establishing a connection to the C2 Server/killswitch to communication to and from the infected machine to the C2 server, as well as closing the connection (via InternetClose handle).

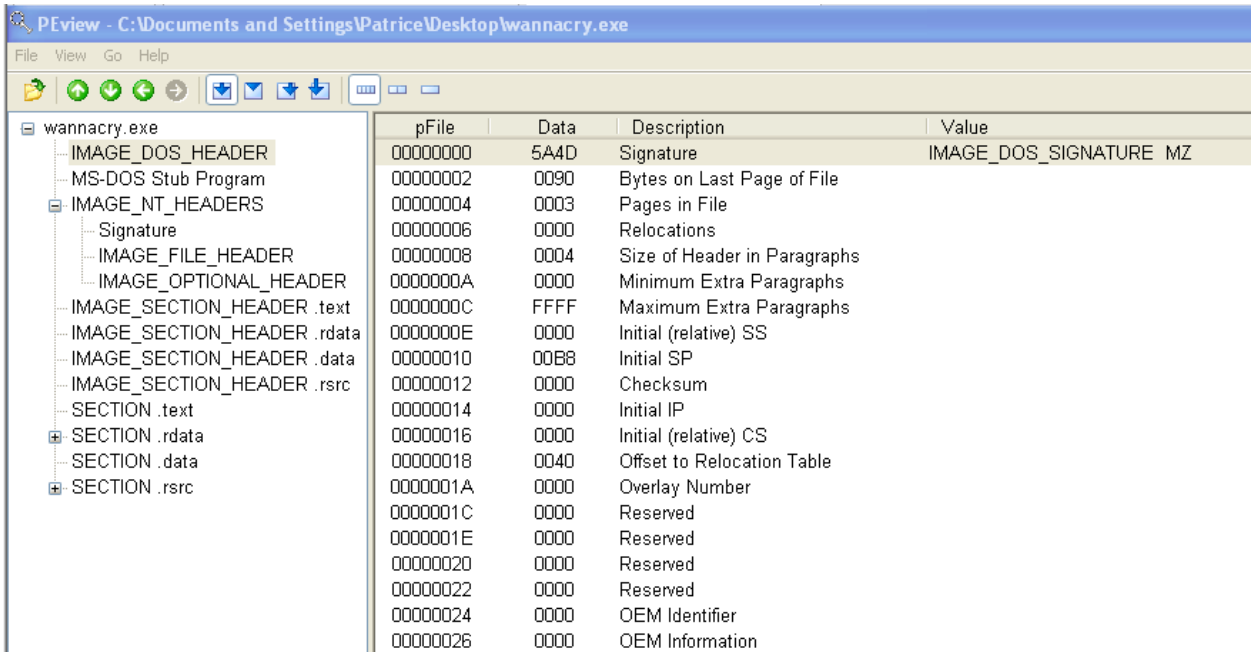


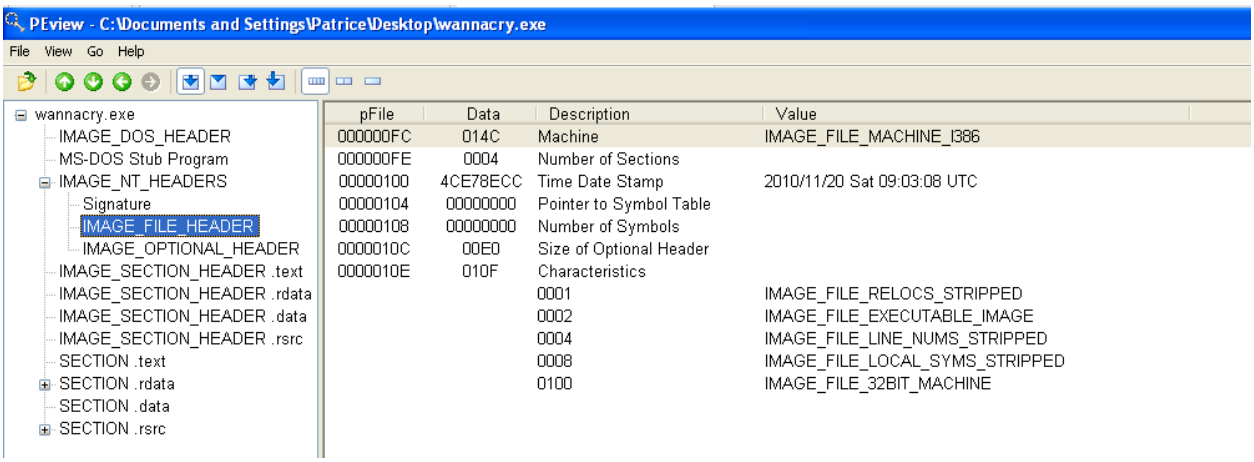
Msvcrt.dll: this probably deals with the windows specific OS functions since it has C like functions like sprint, stricpy etc.

3.1.5 WinHex.exe, PEView (XP) and PEStudio (Win8)

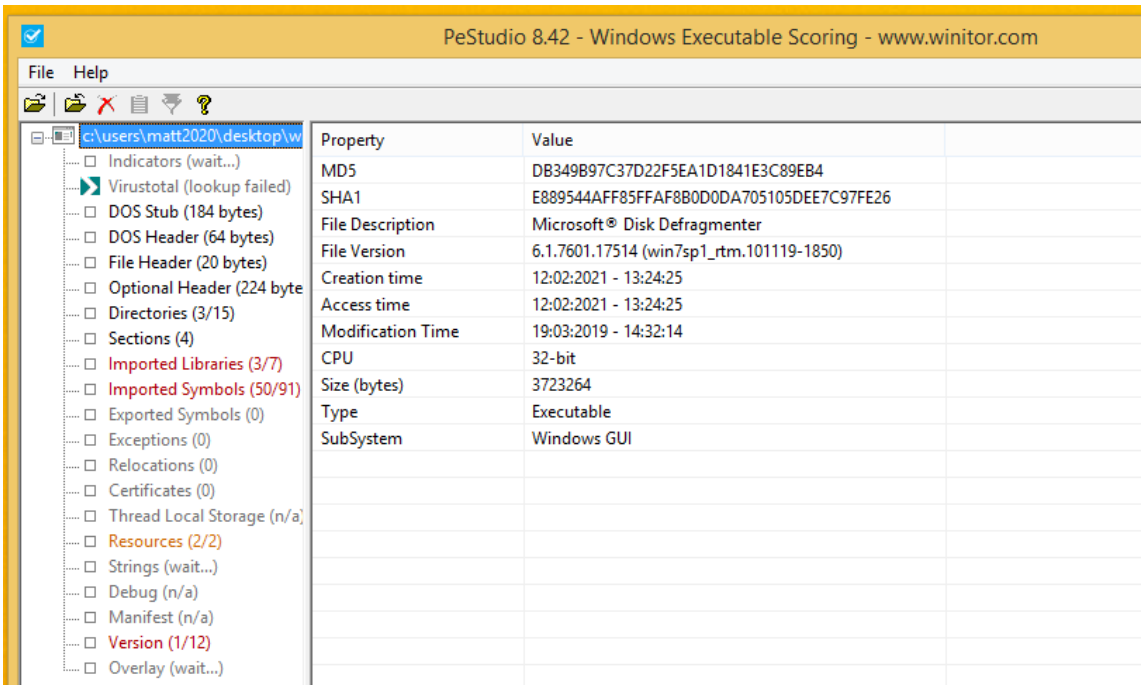


In both Winhex and PEView in the Windows XP machine, the file signature of the file is MZ, indicating that the file is a portable executable.

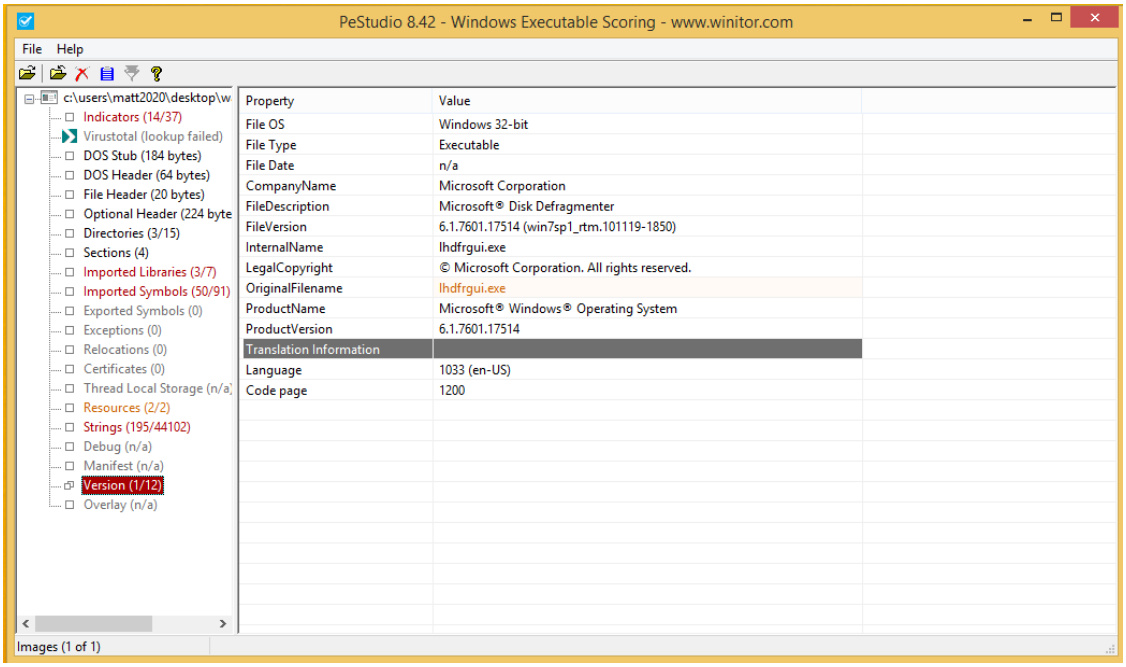




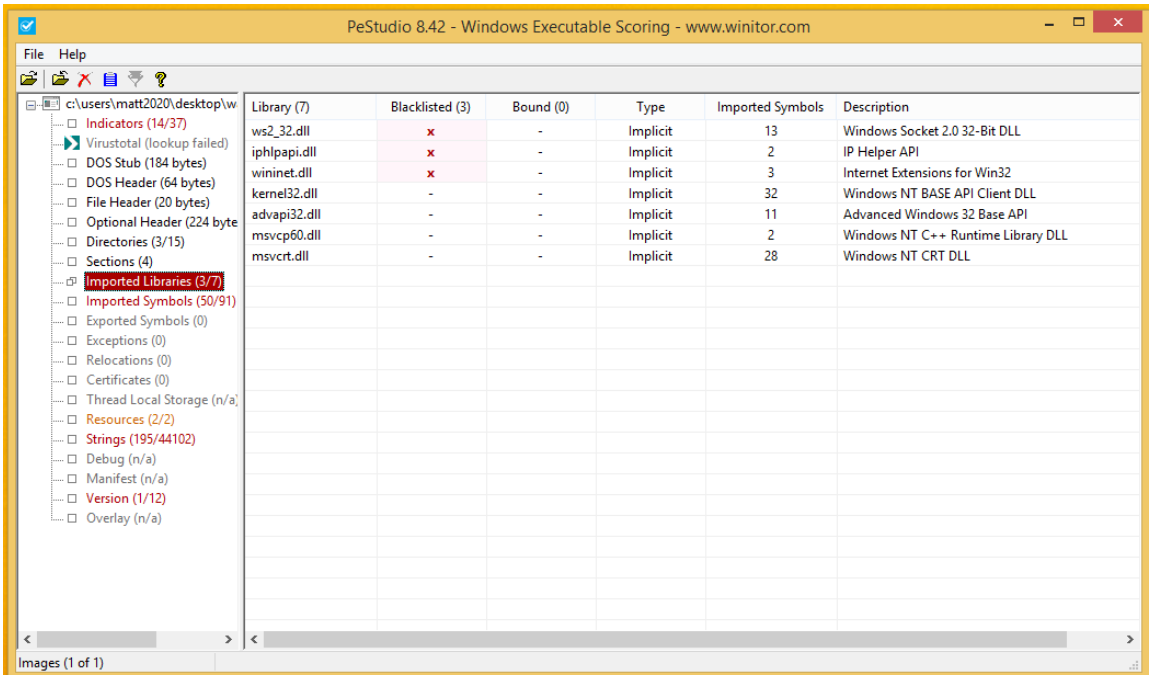
In addition, PEView also provides the date when the file was compiled—Saturday 20 November 2010 at 9:30am UTC.



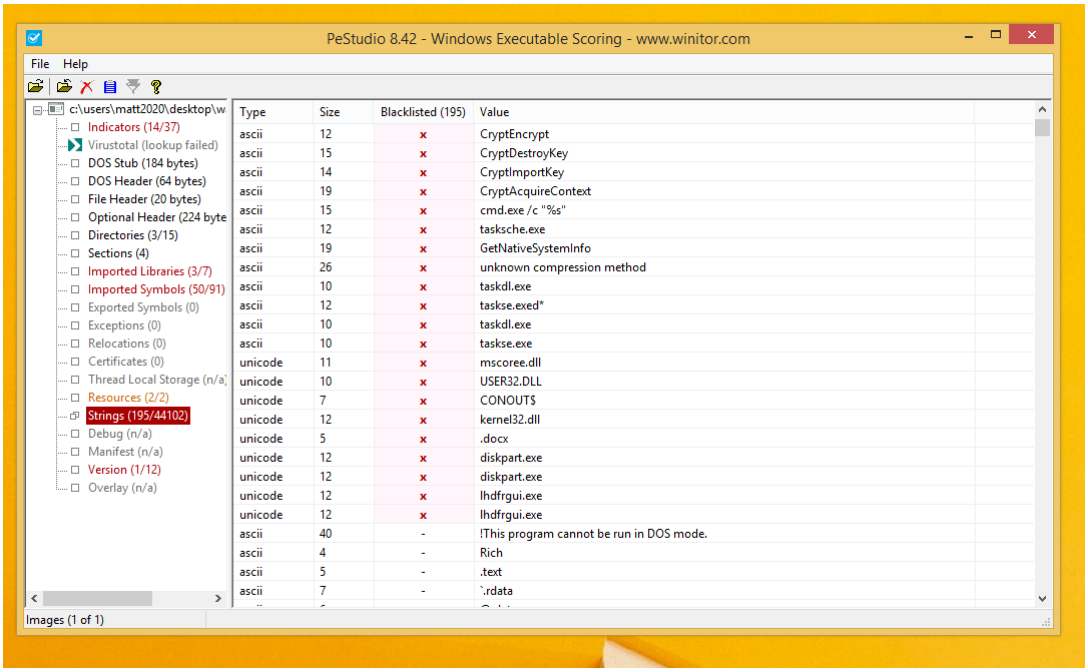
Some additional information that PE Studio provides that both PEView and WinHex includes the MAC times, which are when likely the malware was uploaded and downloaded to and from the ghidra website. It also provides the disguised name that the malicious executable uses, which is MS Disk Defragmenter, a common program used for system maintenance in the Windows Operating System (OS).



More information is also available in the version tab.



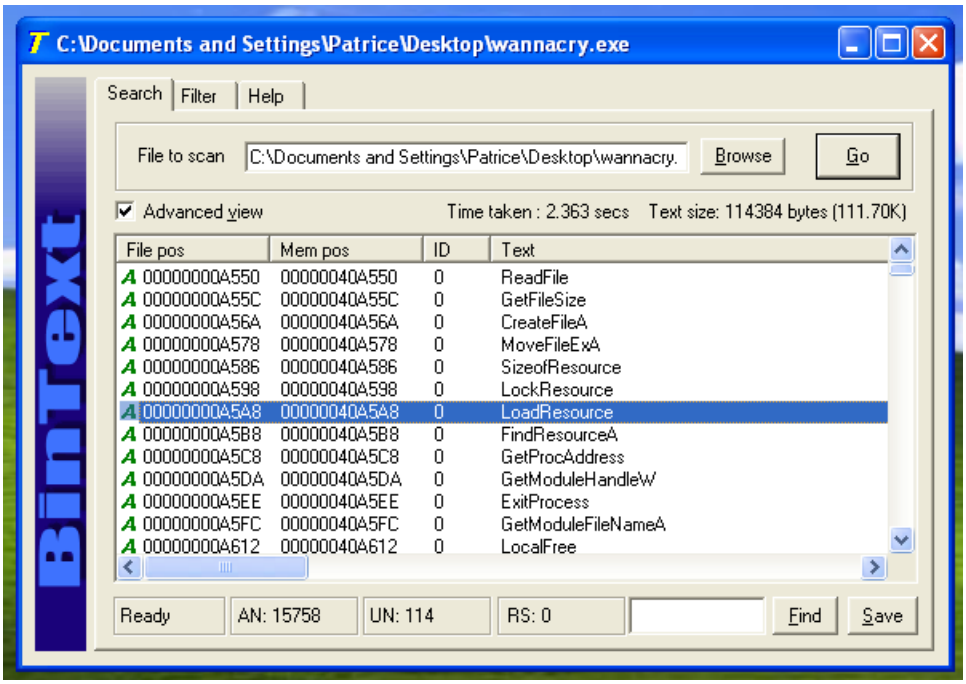
It also provides more verbose descriptions of what we see in dependency walker (the functions of legitimate processes invoked when the dll is used and the imported symbols (number of imports per dll)) and whether the dll is blacklisted (likely to be malicious if used by the malware).



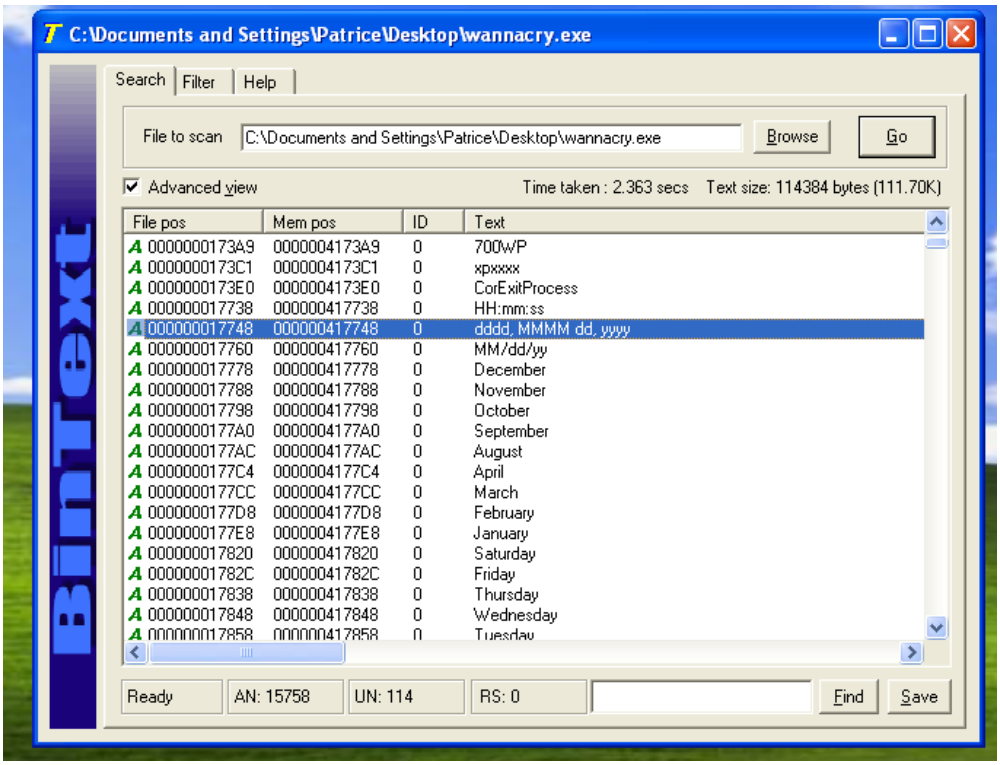
In addition, the strings tab also reveals the type and size of the various strings, which are not present in Bintext. This maybe useful when tied together with Advanced Static Analysis using the IDA Pro disassembler since the size of the various strings and imports are important in the assembly language.

3.1.6 Bintext

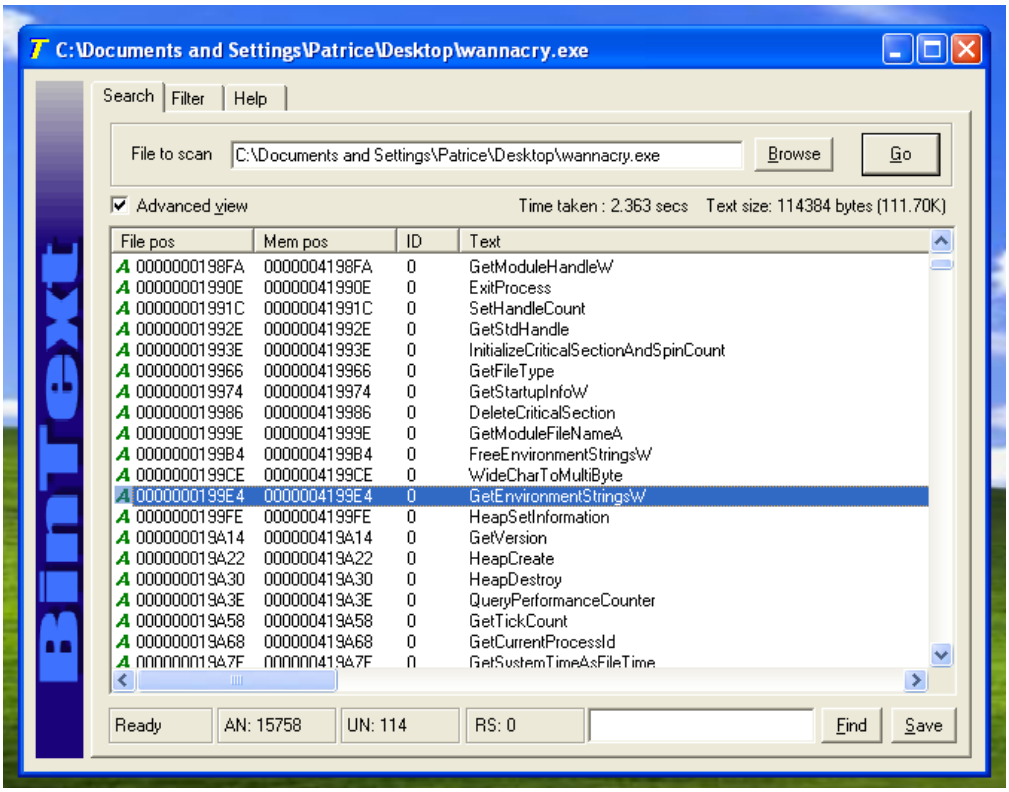
Using Bintext allows us to analyze the strings withing the executable program itself, which may contain information about various network and host-based indicators.



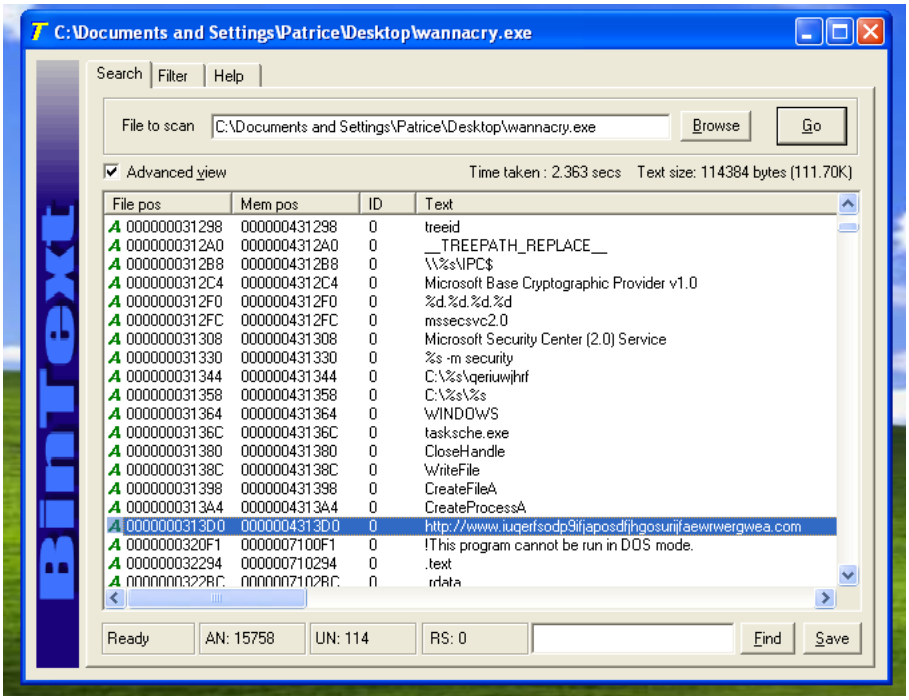
Based on the strings in Bintext, there were some indicators of significant API calls as what was discovered previously using the Dependency walker application.



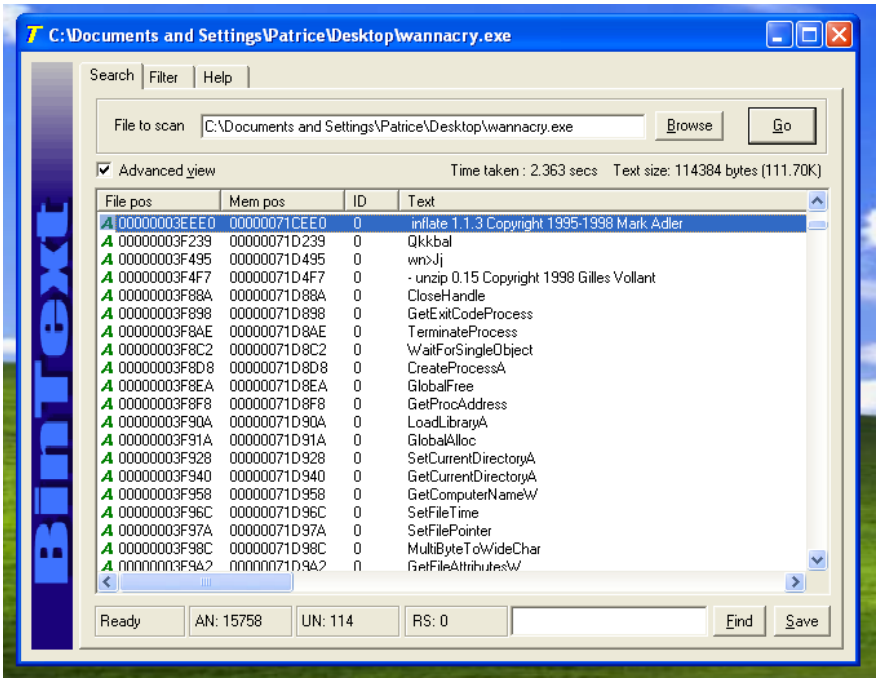
In addition, there are also some strings regarding date, month, year and names of days. This is probably used by the malware to feedback to the C2 Server when the machine was infected based on the date and time on the local machine.



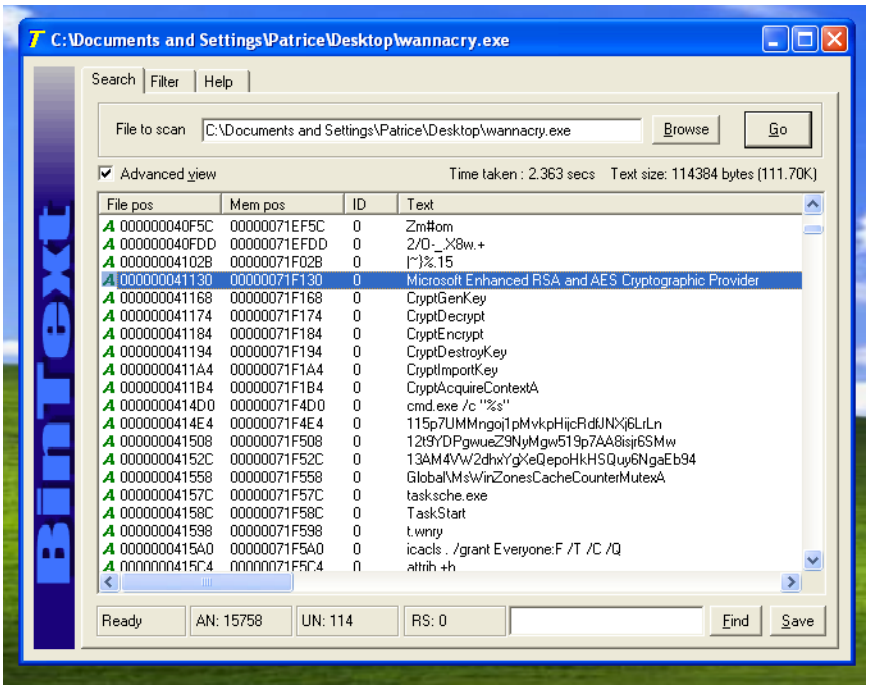
There are also some possible setting of environment variables based on some of the import strings as seen above that were not previously detected in the dependency walker application thus using more than 1 application to statically analyze the malware since the software used to analyse them are not perfect and may miss out something that other static analysis tools might pick up.



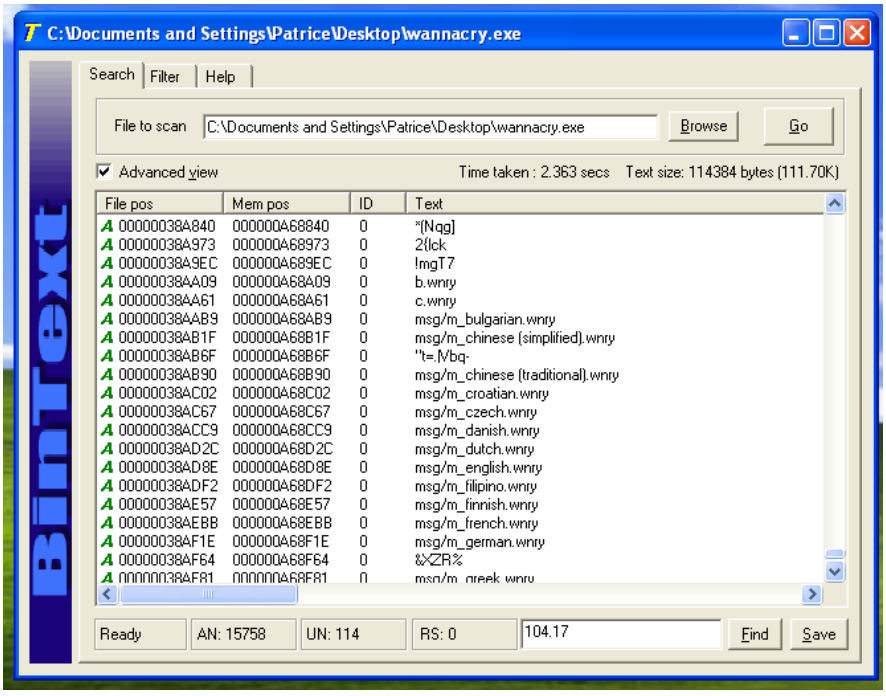
Several host and network-based indicators are seen as above, including the killswitch url, path place holders (%s referencing directory or files within the C:\ drive which is used to boot the computer), as well as refences to the Microsoft Base Cryptographic Service and Security Center (probably to disable security protections in the OS).



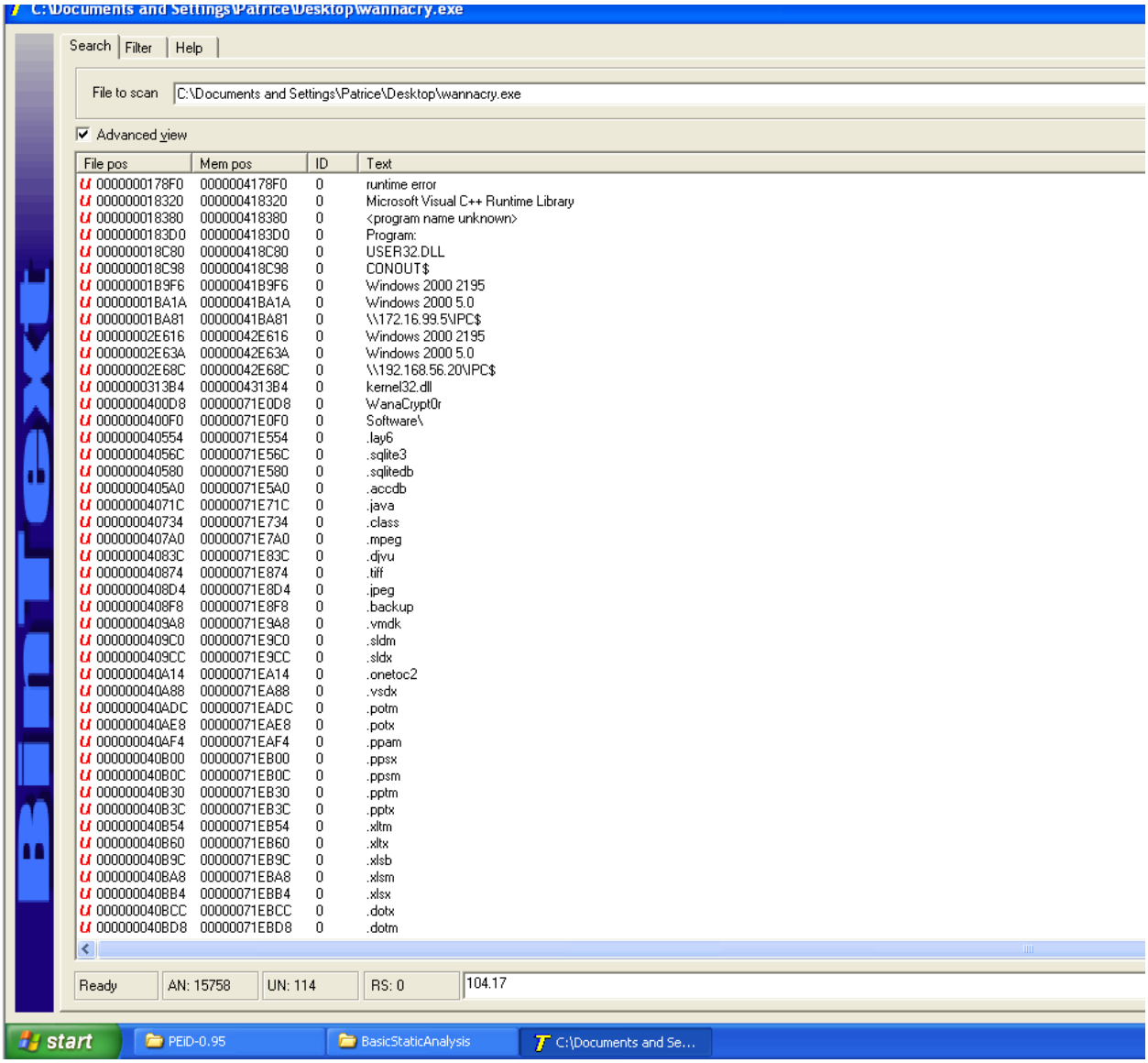
Some other programs that maybe used include inflate and unzip 0.15, which may be used to inflate and unzip the malware payloads assuming that it was previously compressed, to access the imports and functionality during runtime.



Here, cryptographic APIs and imports are refernced together with a mutex to ensure that the malware only run once and does not reinfect the machine inadvertently if run more than once.



The files which contain the ransom note in various languages are also available in the strings.

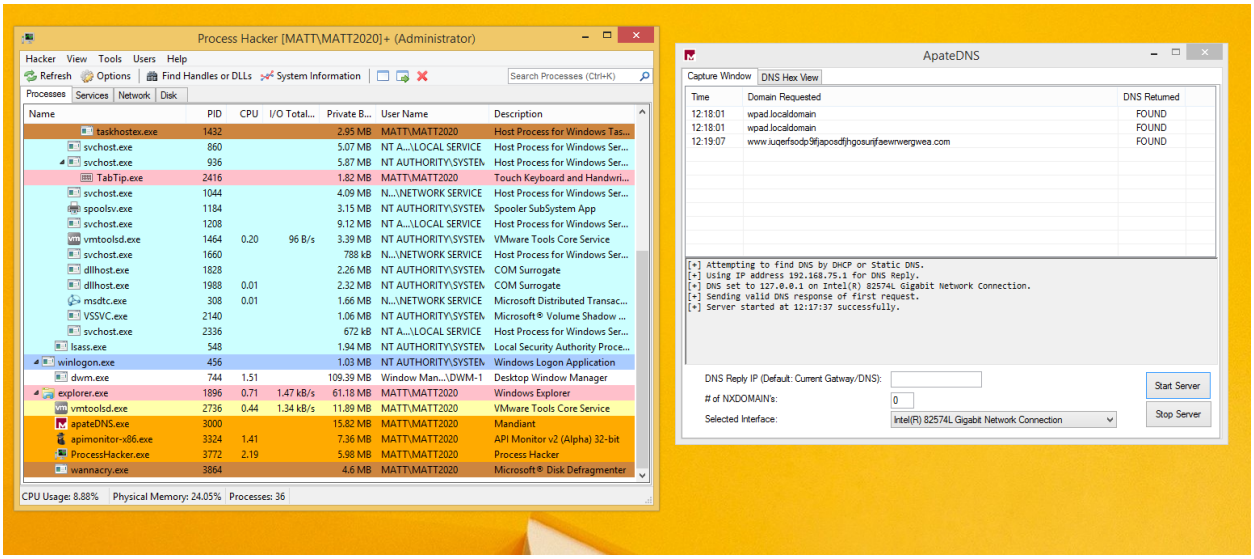


There are also some more windows specific imports and files near the bottom, with various file extensions, probably related to the encryption process by the ransomware and OS versionings as well as possible file shares for the worm to infect (using \\<ip address>\PC$) to display hidden files and folders in network shares too.

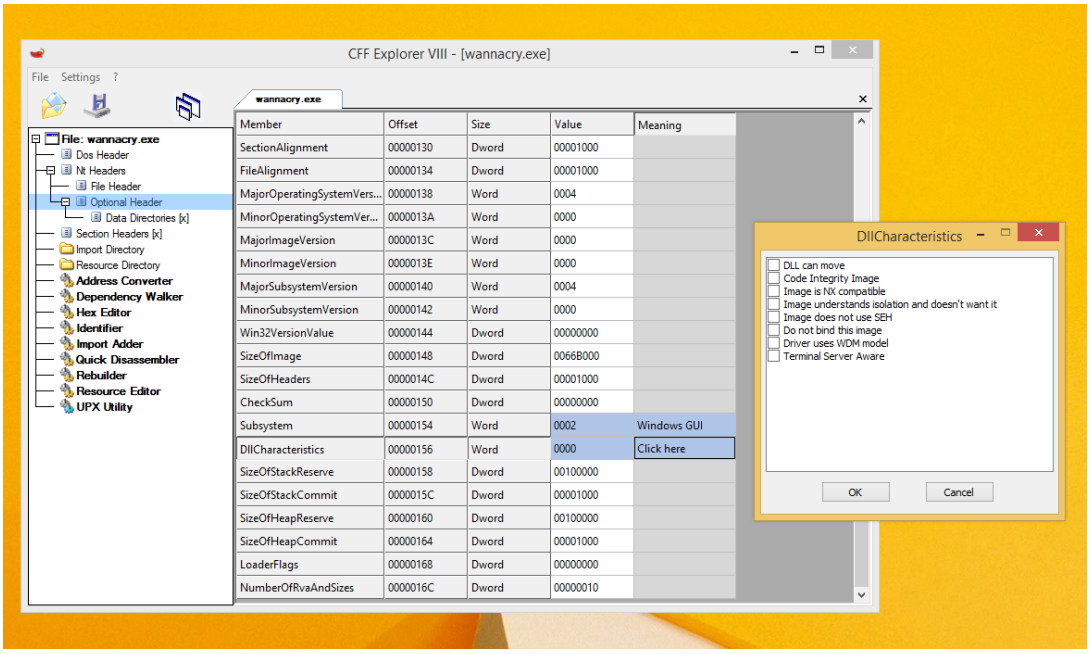
3.2 Basic Dynamic Analysis

Dynamic Analysis should be conducted to make up for the limitations of static analysis. It allows the malware to behave as if it were running on an infected computer in the wild. This allows us as malware analysts to analyze behaviour of the malware upon execution—various host-based and network indicators which will not be able to be detected unless the malware is run on the target system, which may include file deletion and creation, network communications, registry updates for persistence and API imports.

Dynamic Analysis is done solely on the Windows XP machines because there were likely to be compatibility issues with pr2.exe on the Windows 8 platform (Build 9600). This was not due to the ASLR issue but likely already patched in a security update.



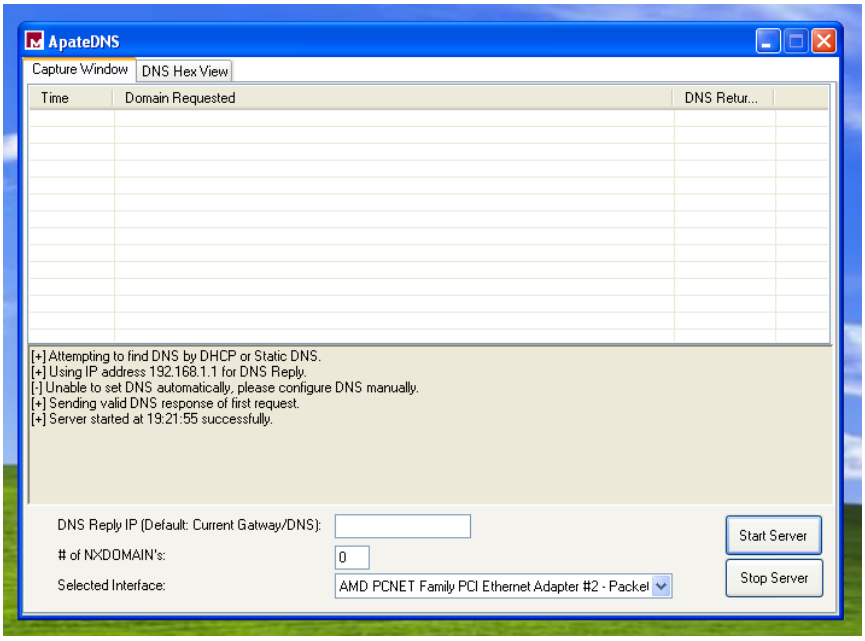
As seen in the screenshot above, Process Hacker captures the process running but it terminates shortly after. ApateDNS, which simulates the malware being active in an actual network also has the killswitch url displayed only once, suggesting that the process was terminated prematurely.



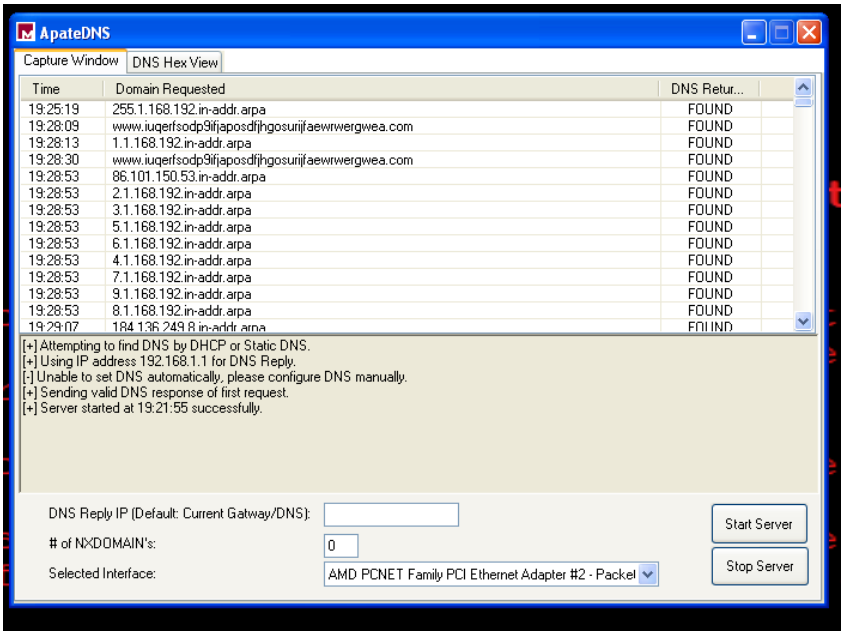
As seen above, the checkmark for DLL can move was already unchecked before execution, suggesting that it is not the ASLR problem which does not let the malware infect the machine. Thus dynamic analysis can only be run on the Windows XP machine since the Wannacryptor window is displayed and other signs of compromise are detected (which we will talk about later below.

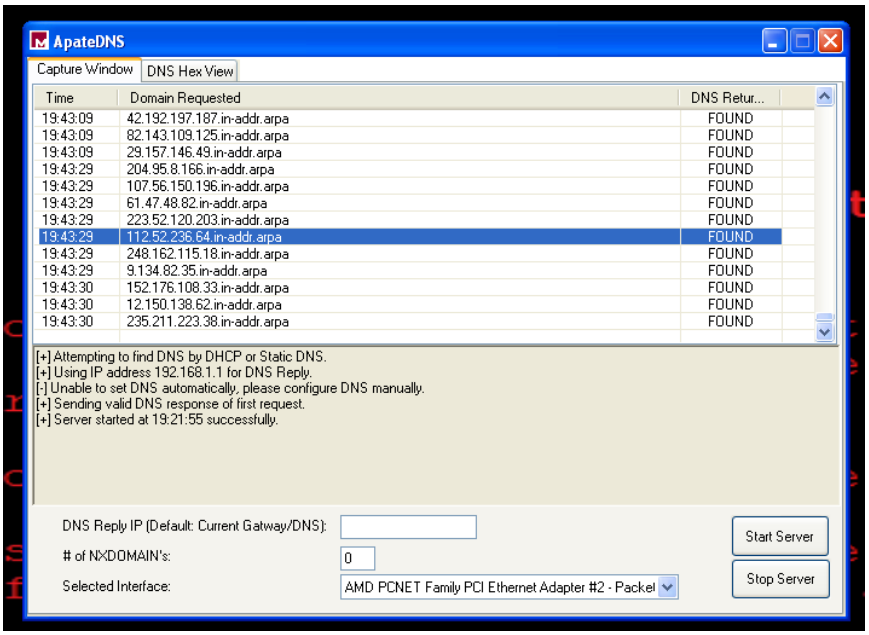
3.2.1 ApateDNS

To monitor outbound network traffic and simulate an actual DNS server, we will use apateDNS. Since the malware is a worm, we expect it to connect to the network to infect other machines, as well as to the remote C2 server for further instructions. We start the server before executing the malware.



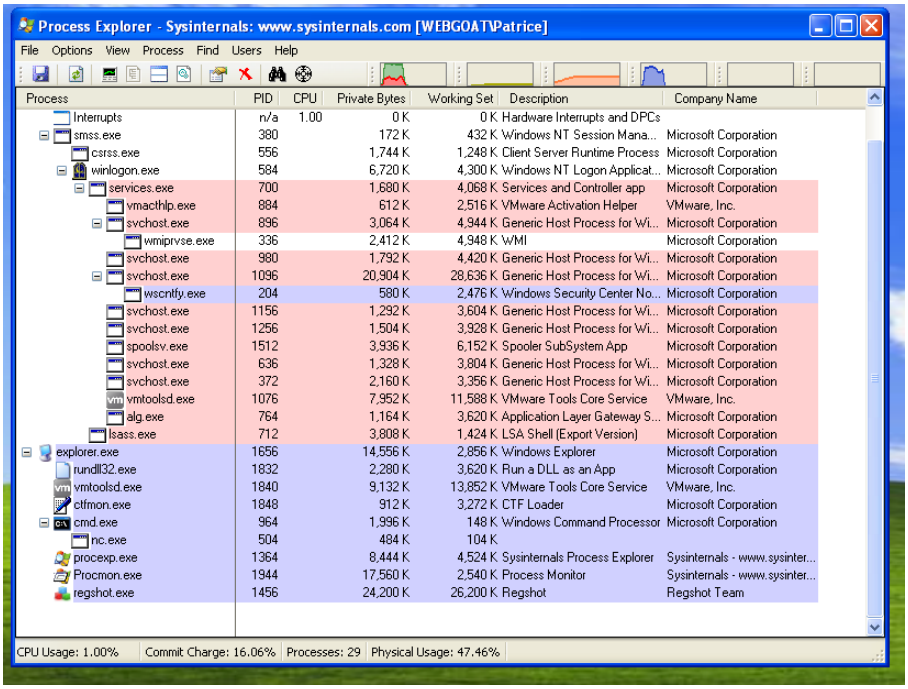
After executing the malware, we discover that it connects firstly to the killswitch url followed by random set of ip addresses ending with ".in-addr.arpa".



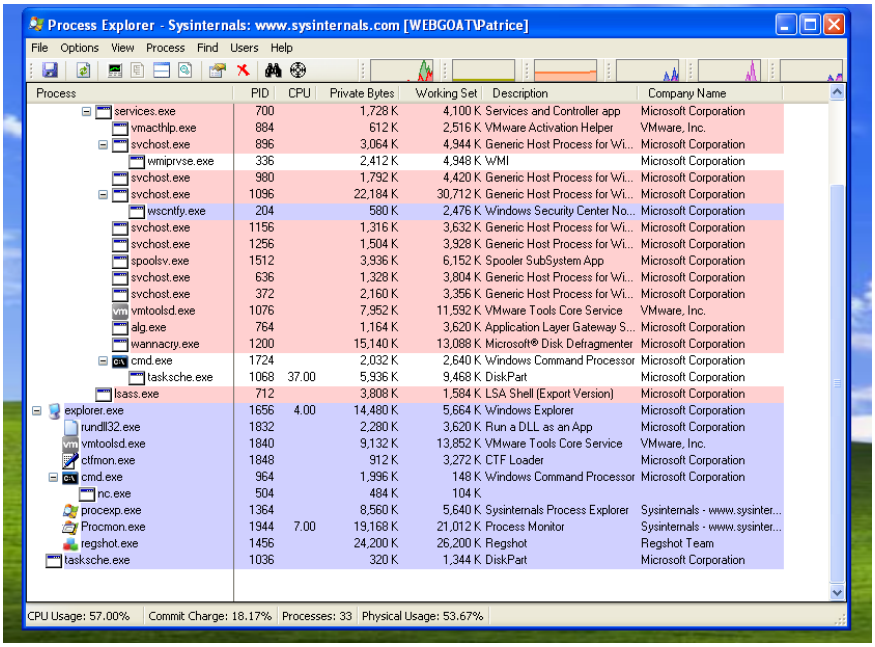


From the screenshot above, we can also notice that it is contacting the IP addresses at given intervals of about 20 seconds and does so for 4 to 7 addresses each time.

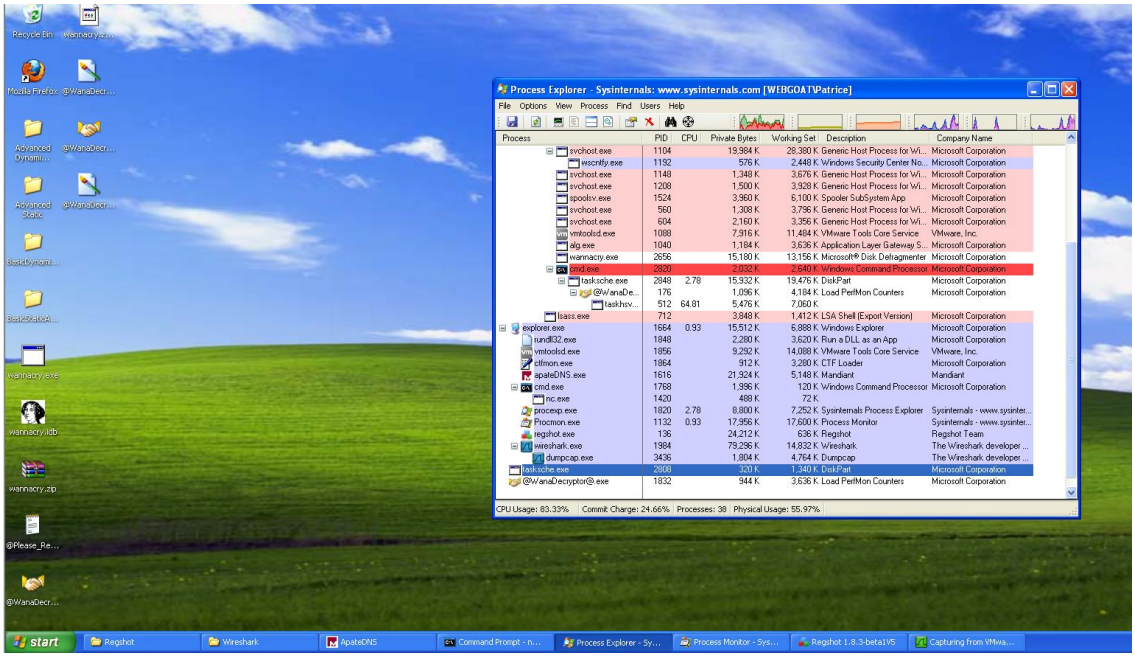
2.2.2 Process Explorer

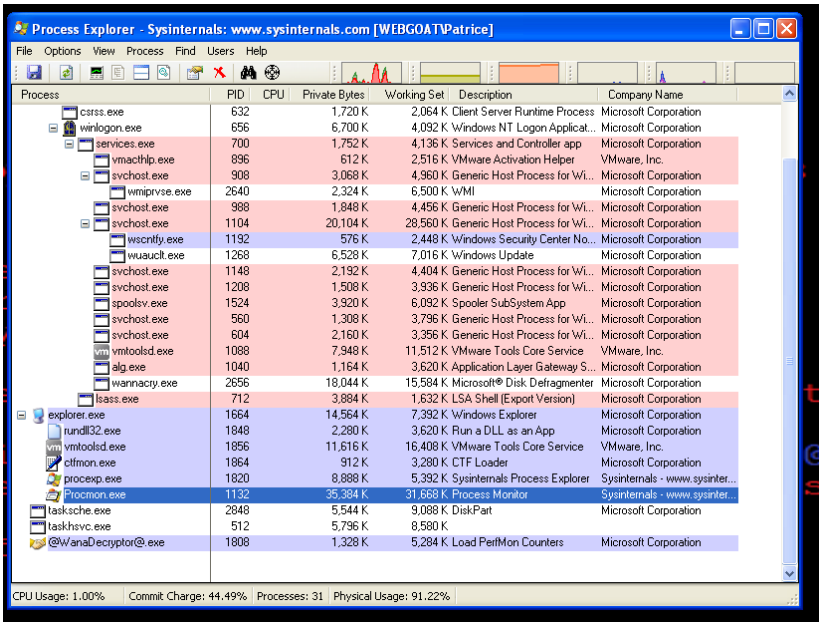


Before the malware was executed, this is a snapshot of how it looked like. We will conduct a comparison of the processes and subprocesses triggered when the malware is executed.

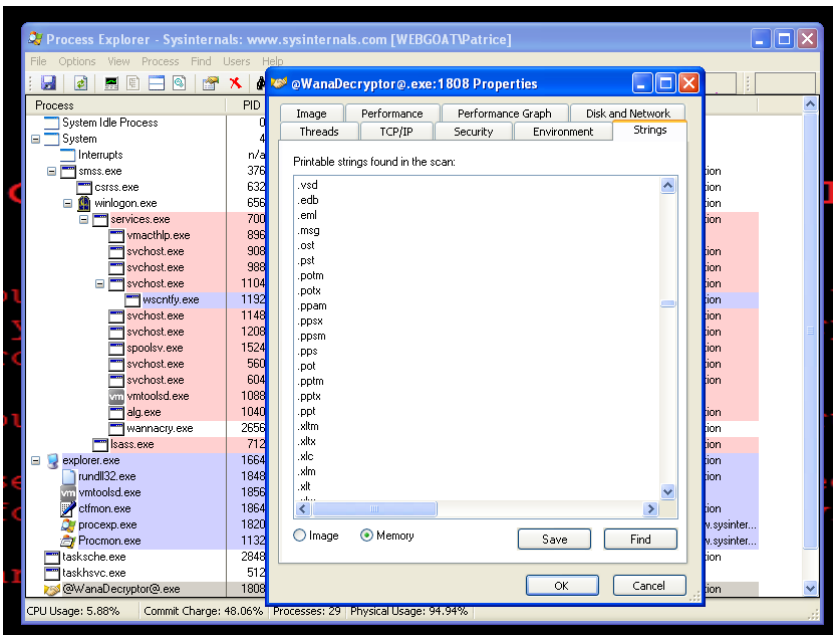


When the malware begins execution, the WannaCrypt0r and tasksche processes gets created almost instantaneously. Thereafter, the main process (tasksche.exe) creates a subprocess @WanaDecrypt0r@.exe using the command line (cmd.exe) as shown below, which has a child process of taskhse.exe

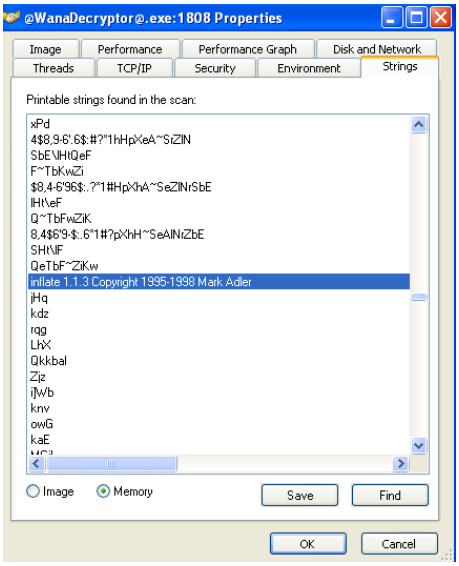




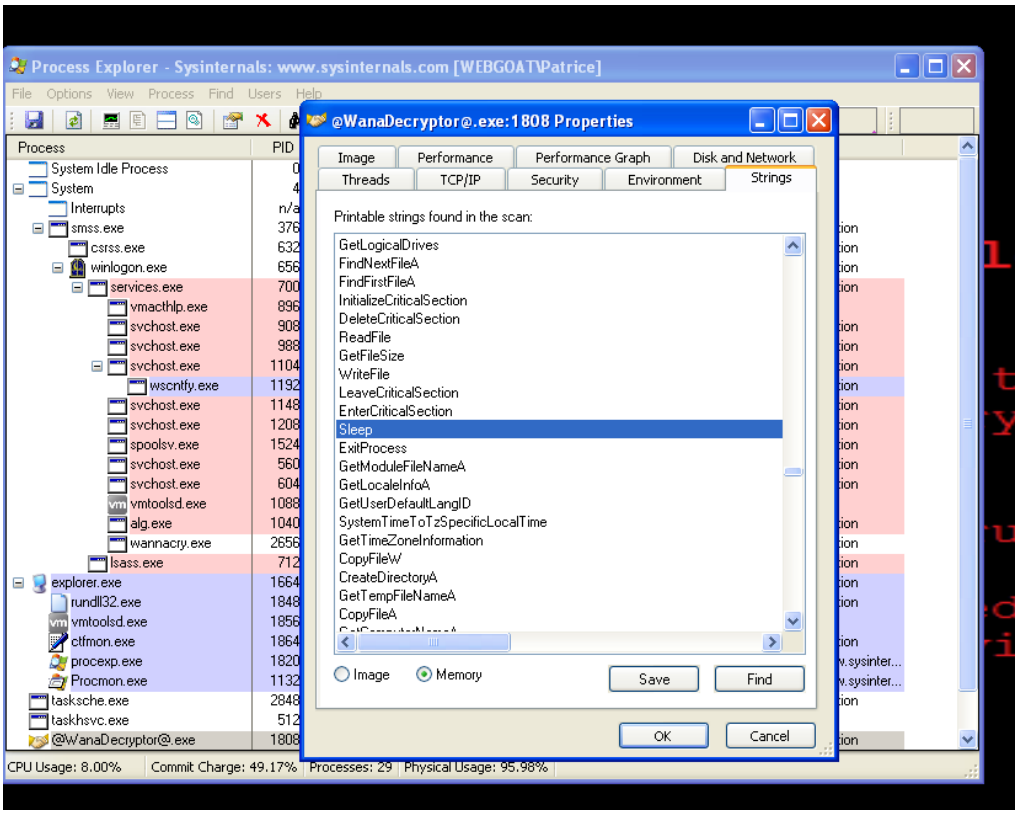
This shows the processes active after the encryption is complete and the ransom note is displayed to the user using @WanaDecrypt0r@.exe. The next step is to analyse the strings in memory for the specific subprocesses since those being analyzed using BinText in static analysis may reveal less information to when the malware is executed and runs dynamically.

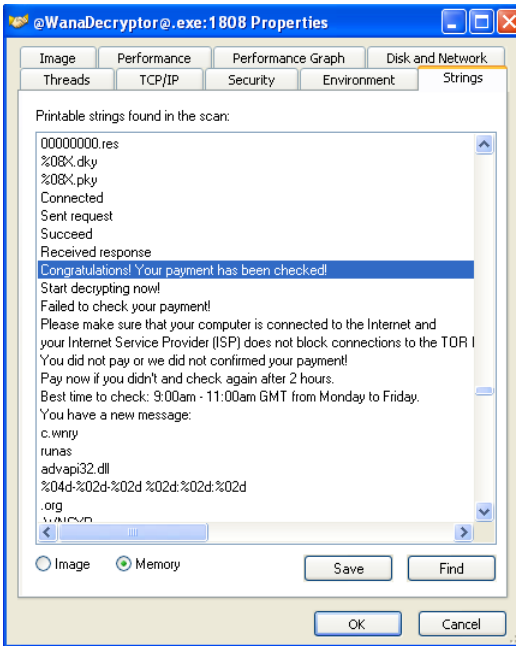


Viewing the memory of the wannadecryptor application, there seems to be multiple file formats listed, likened to what was displayed before the malware was executed in BinText, which is probably used for support in decrypting the various file formats.

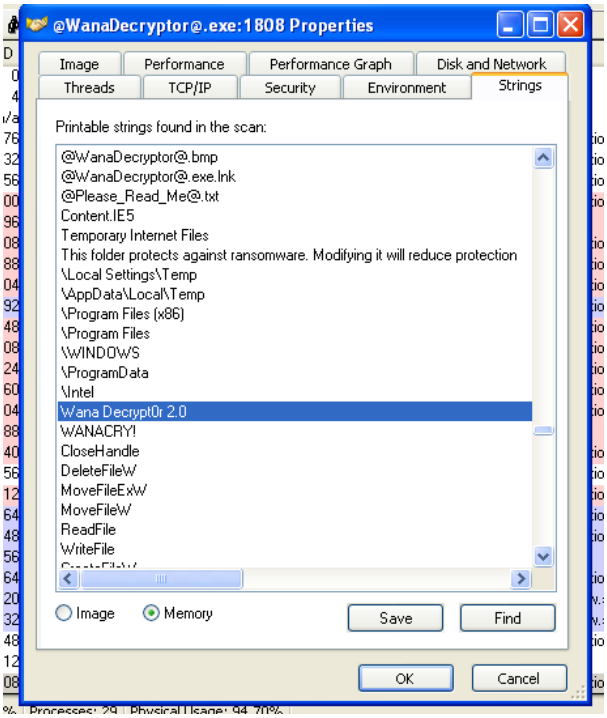


In addition, the inflate program previously in BinText was also found here in the process memory. Several API calls are also available as seen below.

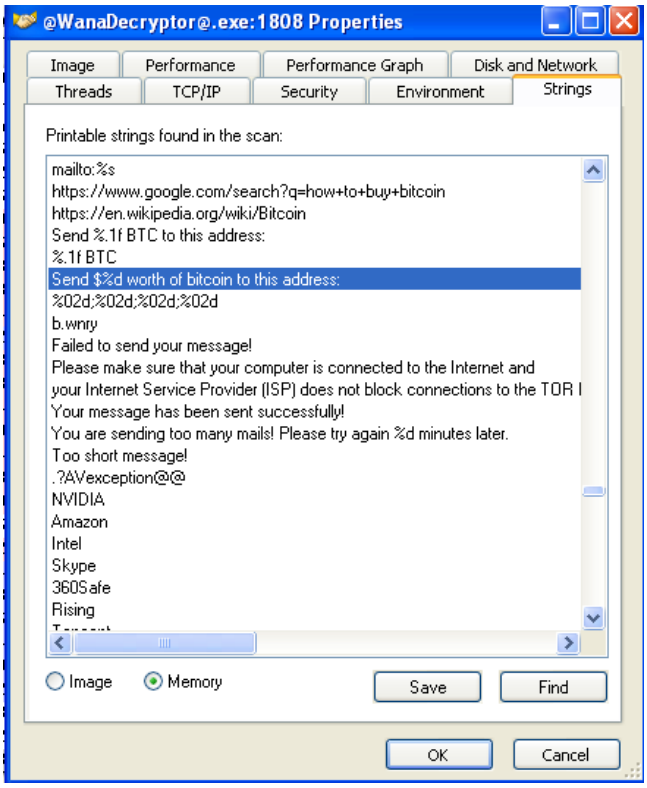




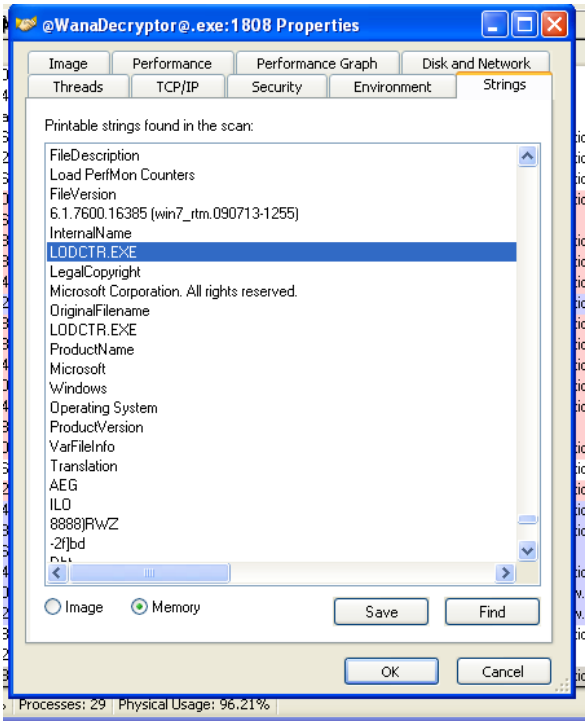
Additional strings that were not previously available include ransom and internet connectivity related messages and timezone information to let the user know how and when to pay up and whether they are connected successfully to the Tor network to make the Bitcoin ransom payment.



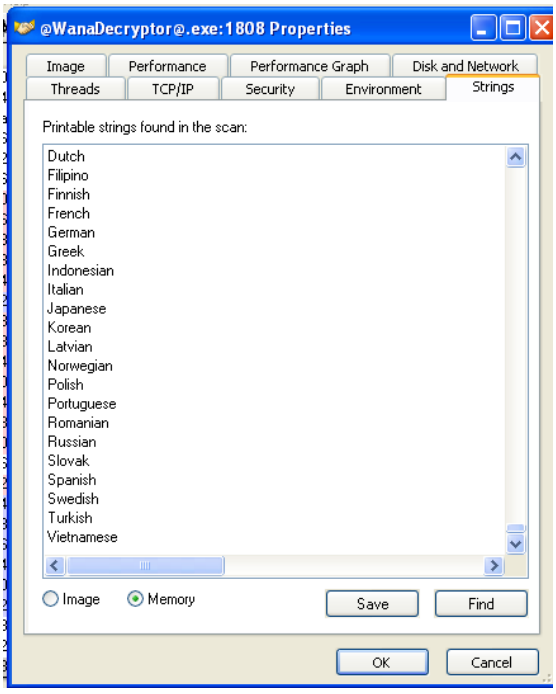
In addition, some additional information relating to the paths in the Windows OS as well as references to temporary directories where temp files are stored. It also shows the wanncryptor desktop icons filenames.



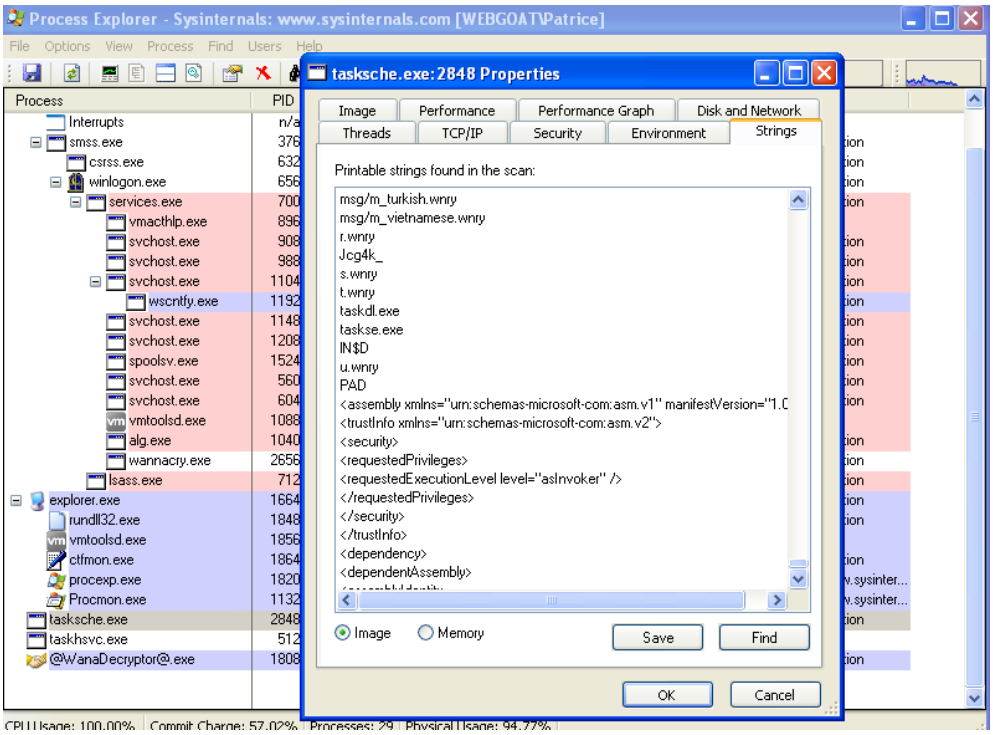
Additional ransom information, internet information about bitcoin for those who do not use them (including a Wikipedia article and a google search) are also given. In addition, in the bottom part of the screenshot, there are also names of various application manufacturers, perhaps suggesting that the malware can be used to encrypt these applications.



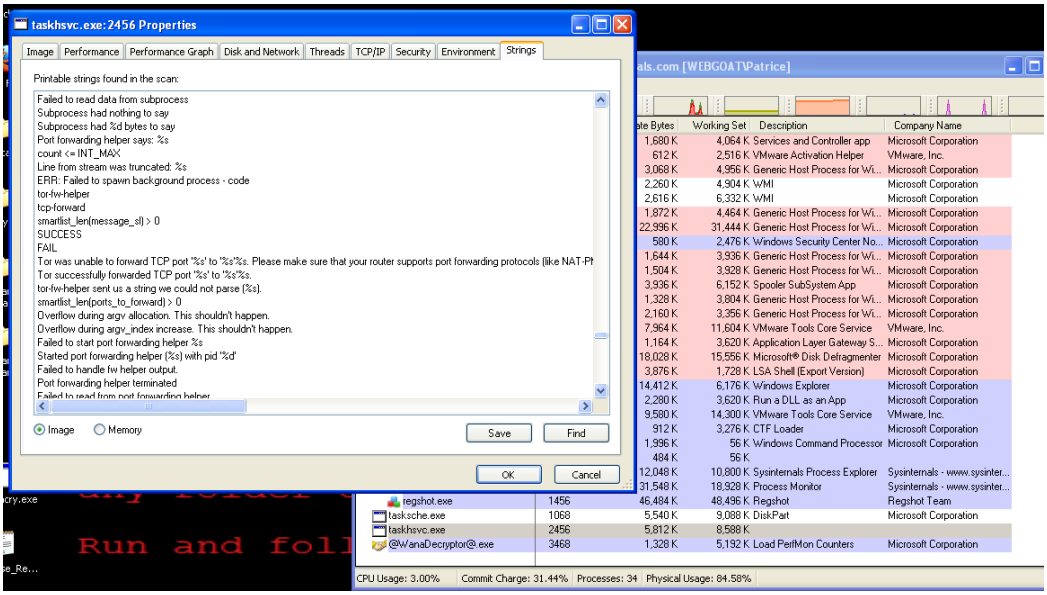
There is also information pertaining to additional applications used, such as LODCTD.exe (above), which is the disguised version of the wanadecryptor application and the supported languages for the ransom note (below).

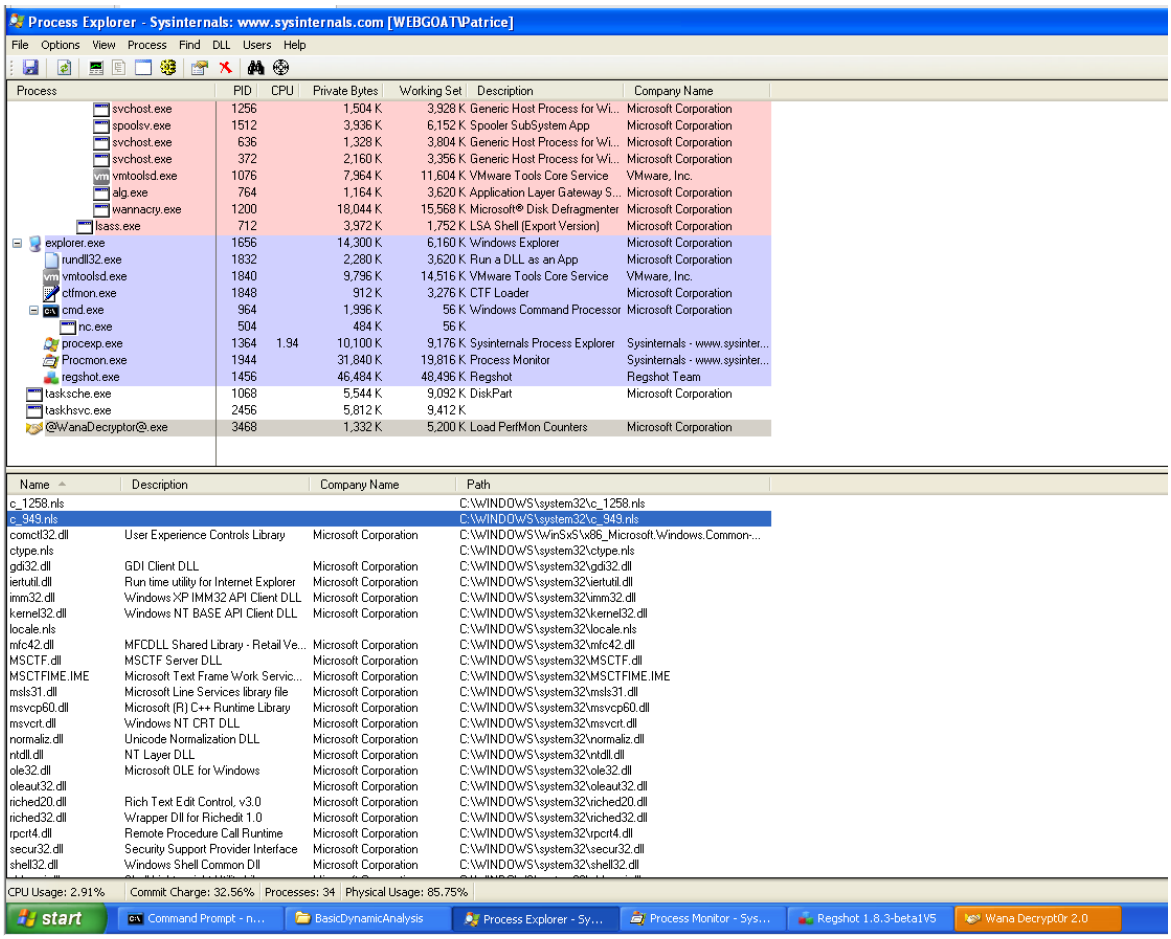


Since quite a bit of information has already been gathered about this particular application, I decided to proceed on with tasksche.exe, which is the main parent process triggered by wannacry.exe (pr2.exe) in this case.



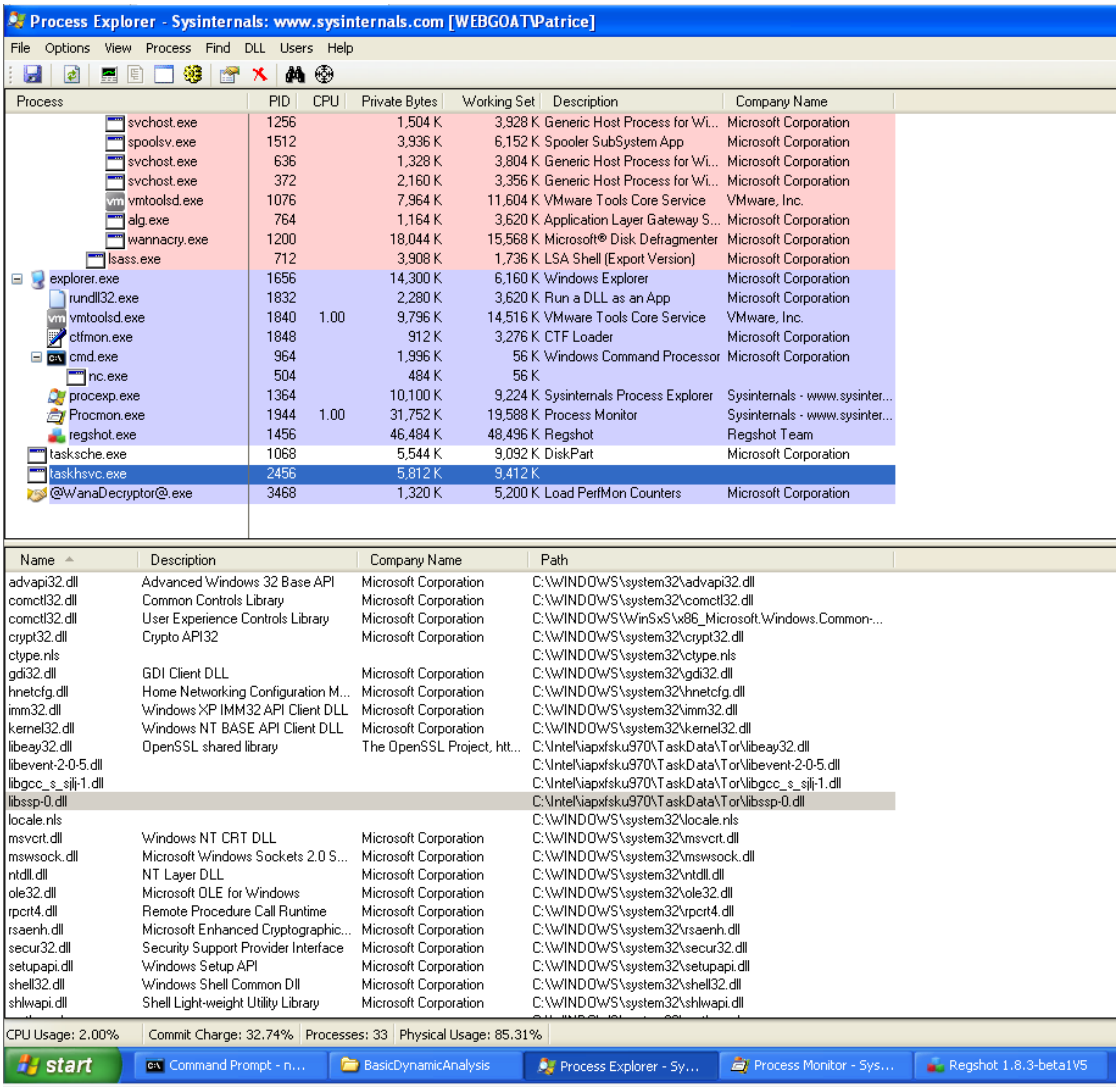
As the malware runs, various processes can be seen to be executed under legitimate windows processes like svchost.exe (particularly wsxntfy.exe). We will be moving on to explore tasksche,exe. In this case several of the dependency files (t,s,u and r.wnry) are seen in the image strings of the process. In addition, there are also some information relating to possible buffer overflows and intenet related functionalities like tcp-forward and connectivity to the tor network using tor-fw-helper. Various placeholders are also used, confirming that C++ is used in the creation and compiling the malware.



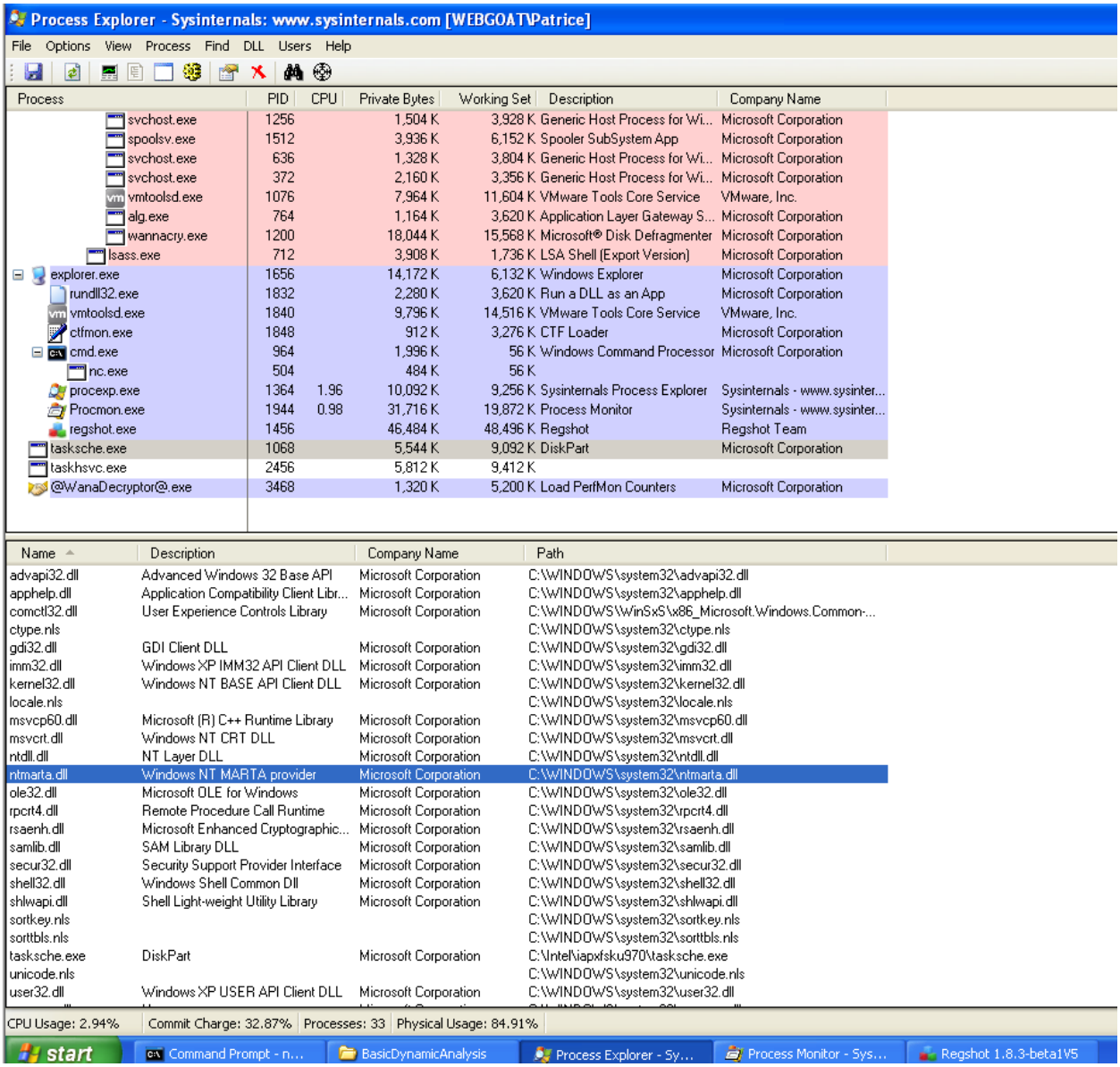


We will then explore various dll imports that the wana decryptor application employs, most of which are not seen in Dependency Walker during static analysis. A notable dll in

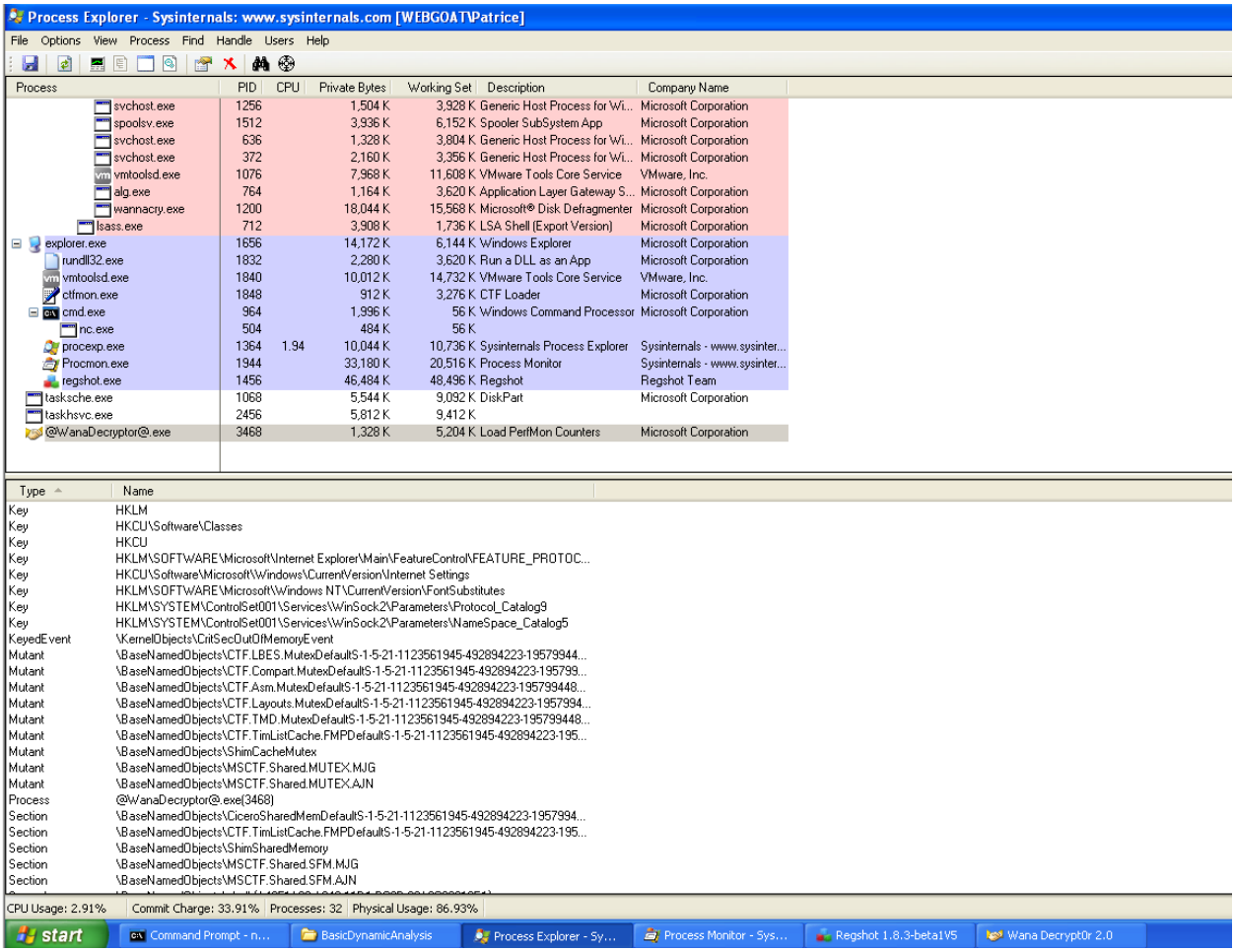
this case will be shell32.dll, which suggests that the application uses the command line to execute.



Next, we will analyse the dlls imported by taskhsvc.exe. It employs the use of several cryptography-related libraries, including crypt32.dll, rsaenh.dll.

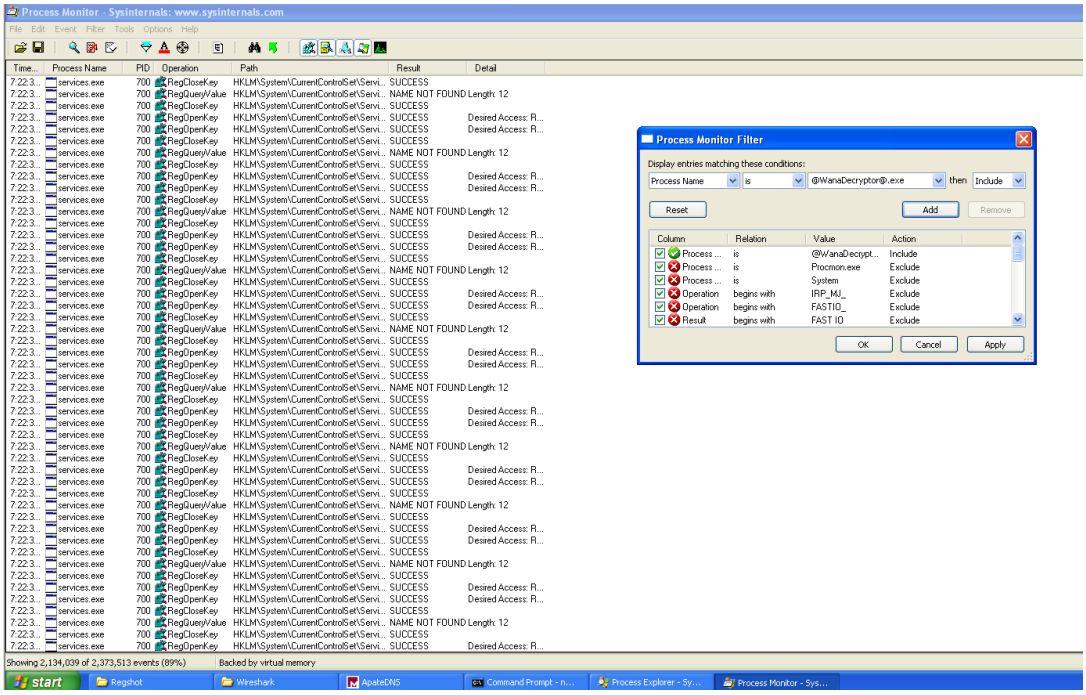


Tasksche.exe is the most interesting of the 3, with all the dlls in Dependency walker seen here, as well as itself running as Microsoft-signed Diskpart application (the resource loader). Next, we will go on to analyse the imported handles made. Out of the 3 processes, only @WanaDecrypt0r@.exe has the most interesting output as shown below.

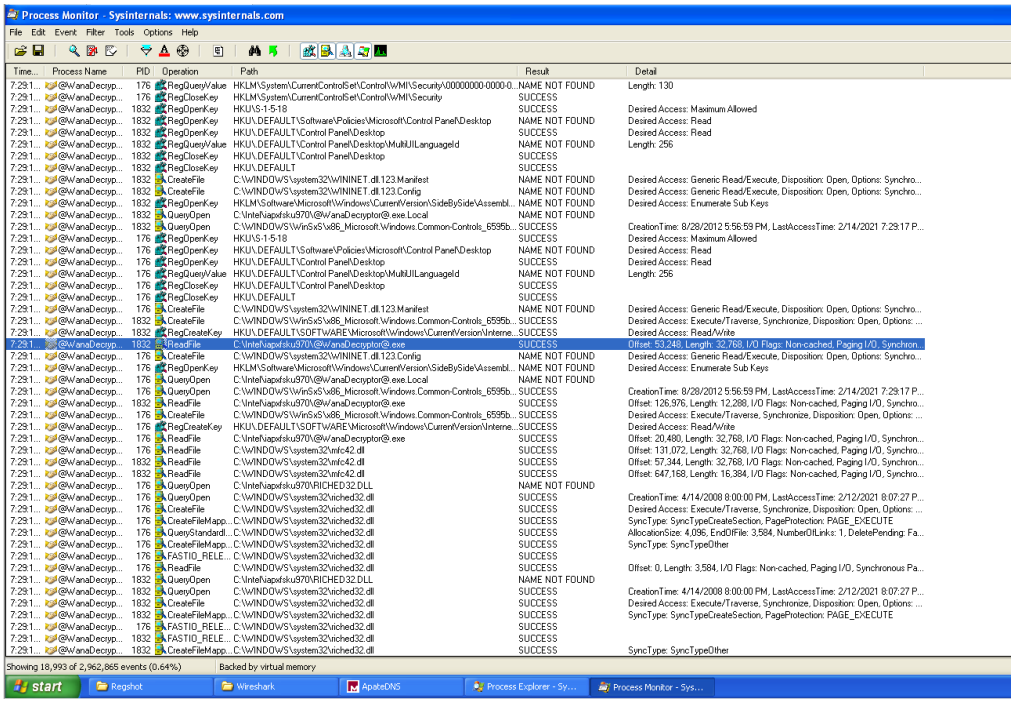


The process has multiple calls to keys (mostly registry keys related the local machine—HKLM) and has several mutexes (probably for the multiple processes that run) to ensure that they do not reinfect/redo the same action that was already previously performed.

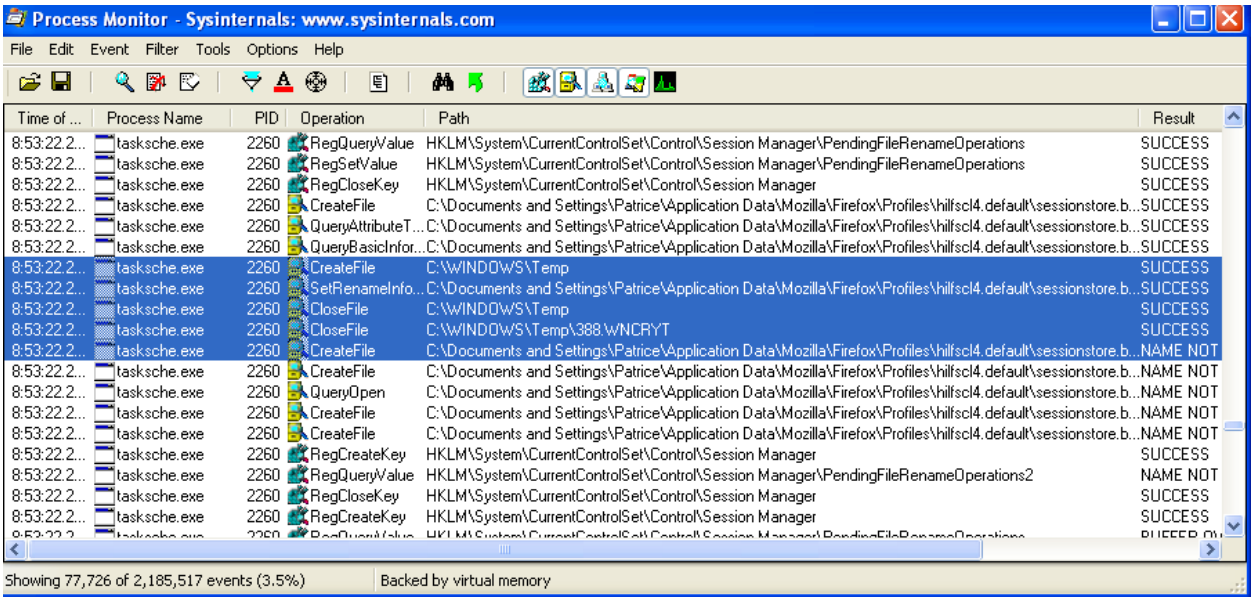
3.2.3 Process Monitor



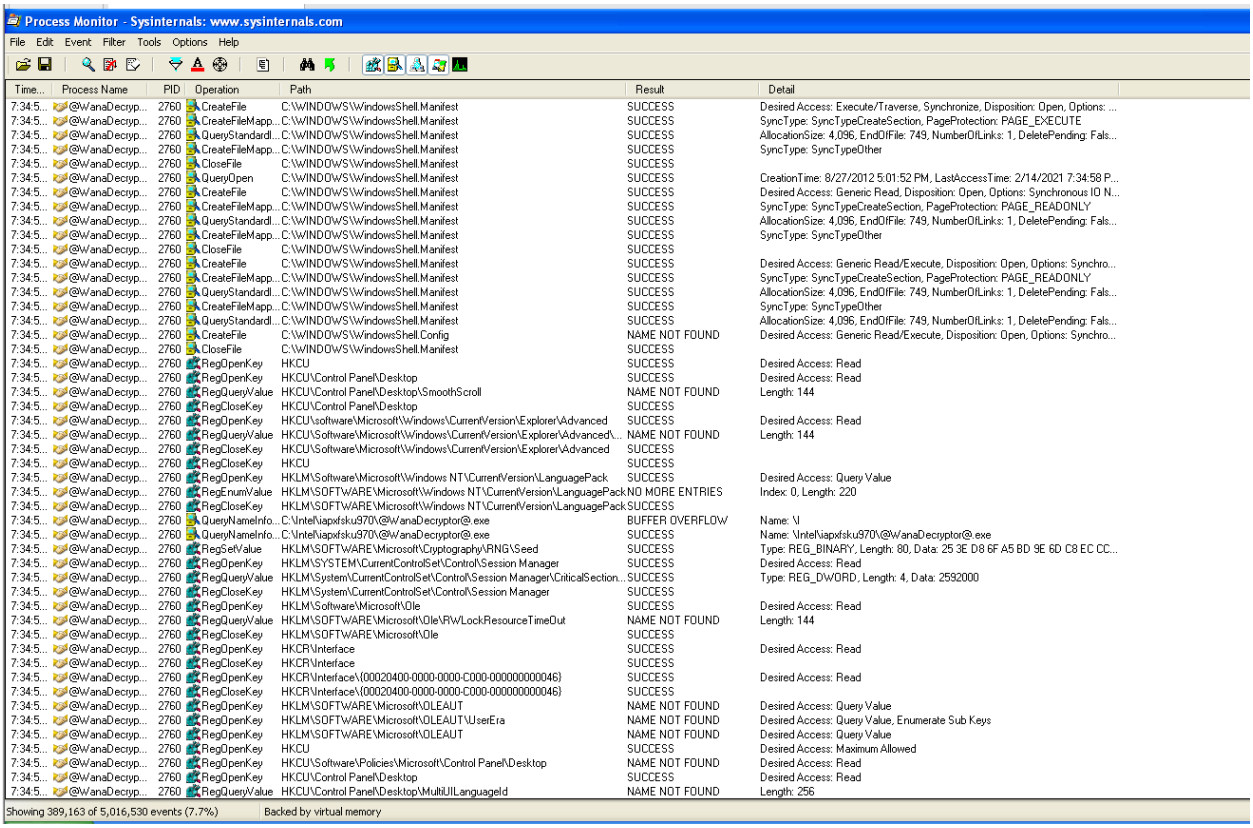
The next tool that we will be using is Process Monitor, that allows us to track and identify specific actions that were taken by specified applications (via the process name). It also includes a timestamp, which is useful in finding out the chronological sequences of events and filters to narrow down the search and get rid of the noisy data triggered by other system or user processes that are not related to the malware. Firstly, we have to use the filter to narrow down each process to analyze and we will start with wannadecryptor and thereafter tasksche.exe.



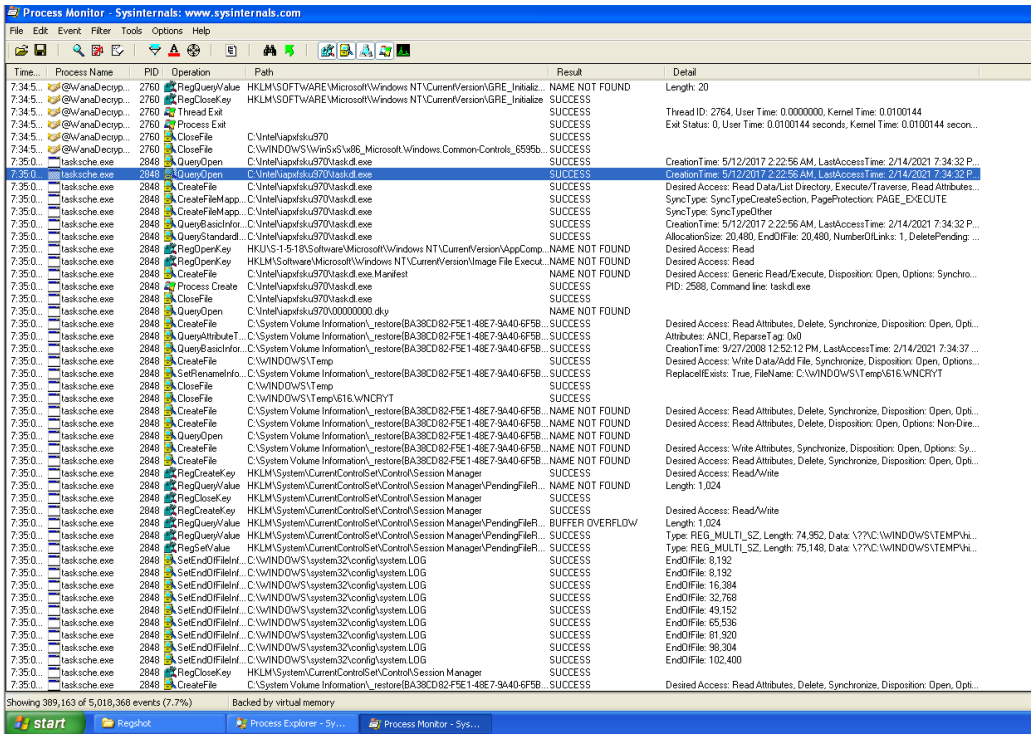
The first interesting artifact is found when the application reads itself in the C:\Intel Directory and creates a key in the registry relating to Internet Explorer. There are also several dlls open and read later on, suggesting that the malware is getting its imports.



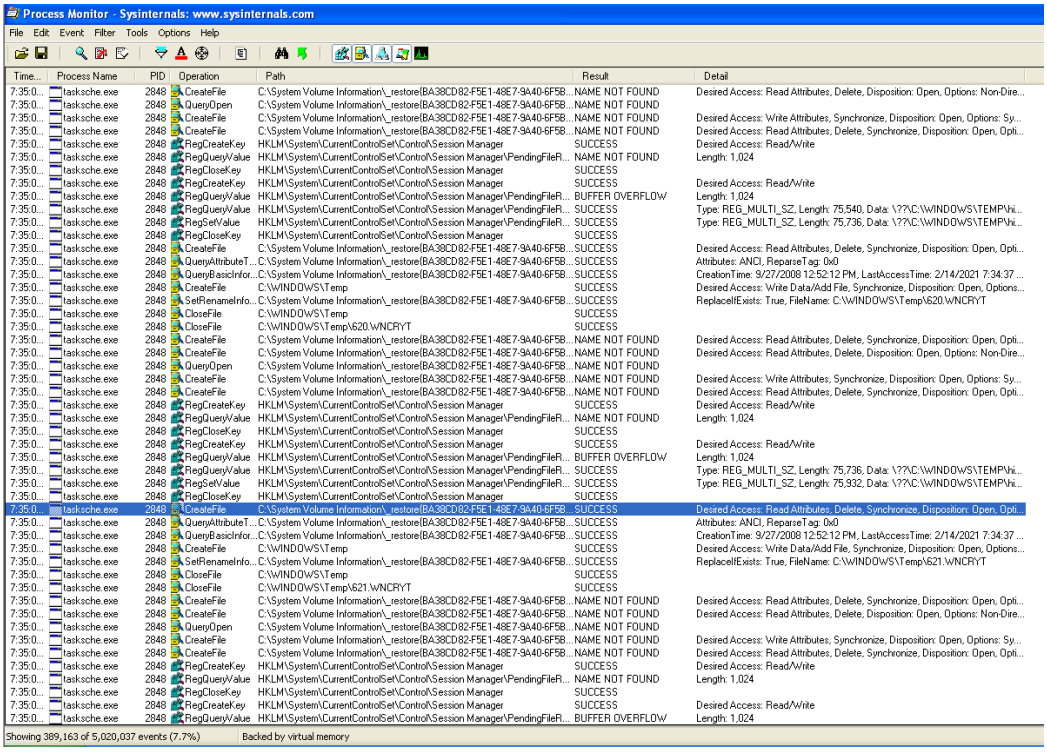
The malware also creates a temporary file in the C:\WINDOWS\Temp directory, which is usually used to store system files as WNCRYT. It also modifies and creates several Firefox profile files.



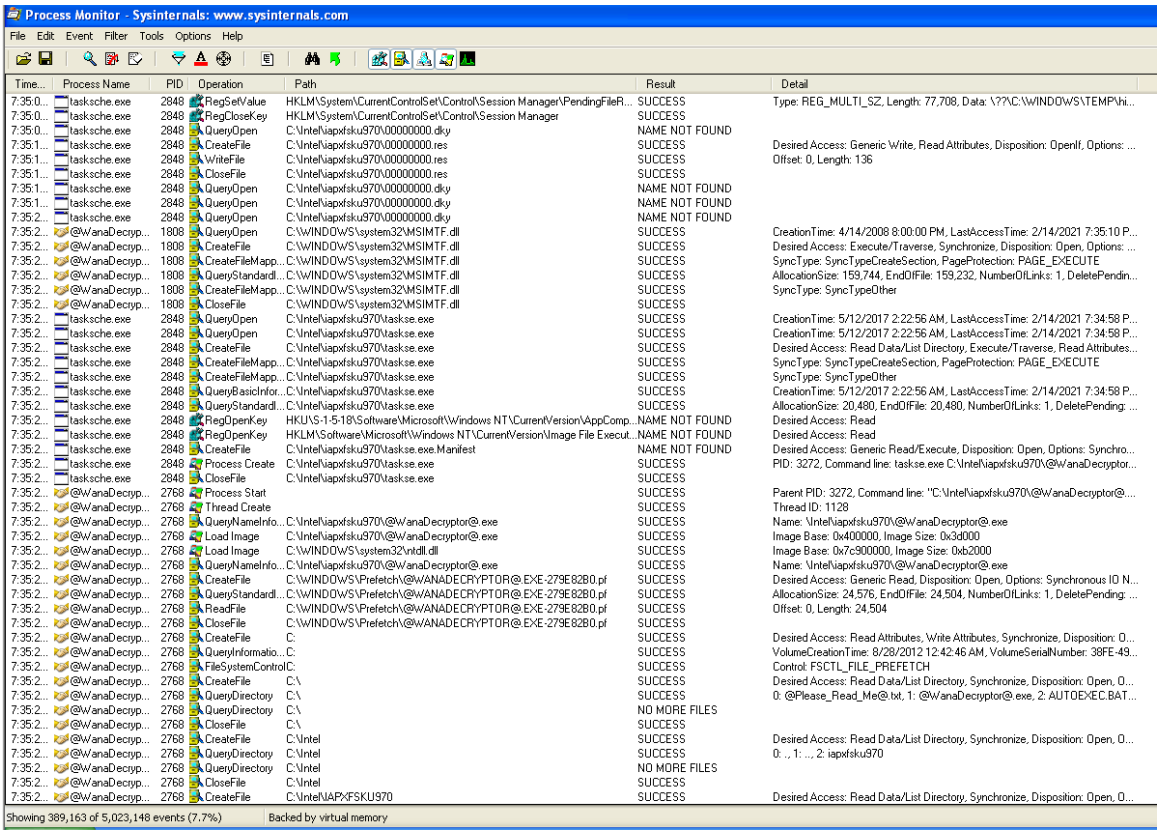
Thereafter, the malware creates a shell manifest and config files, indicating that it is quite likely to use the command line. It then reads a series of registry keys and thereafter perform RegSetValue on a seed of the RNG in the Microsoft Cryptography Registry key, indicating that most likely that will be used for encryption, since a seed is commonly used for consistency (i.e. using the same seed and algorithm will usually give the same encrypted output).



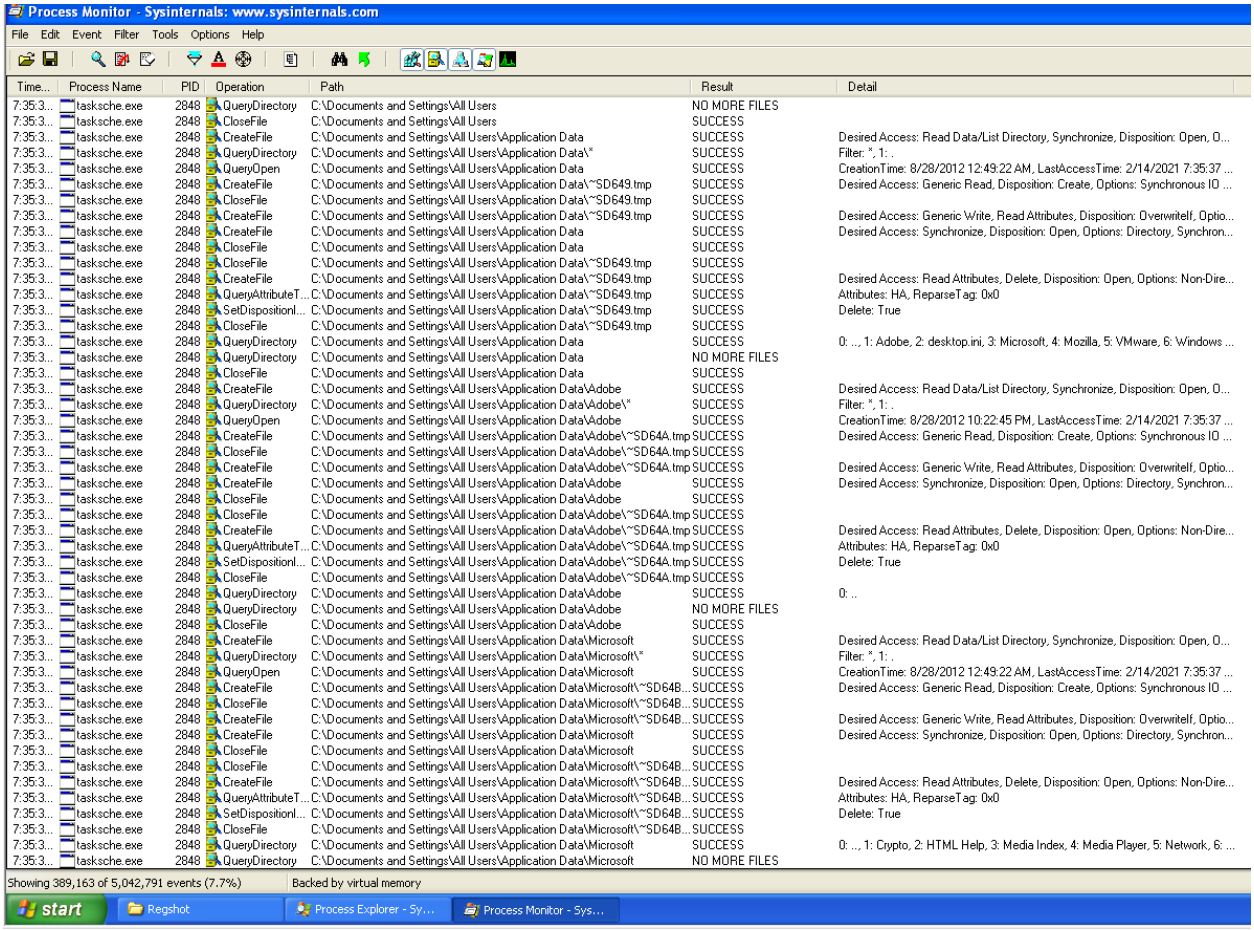
Thereafter, I also found the malware querying for a particular exe file (taskdl.exe) after it created its own folder under C:\Intel. In additional a few registry keys were read and a key was created for the Session manager with the path of the temp folder (likely where WNCRYT lives).



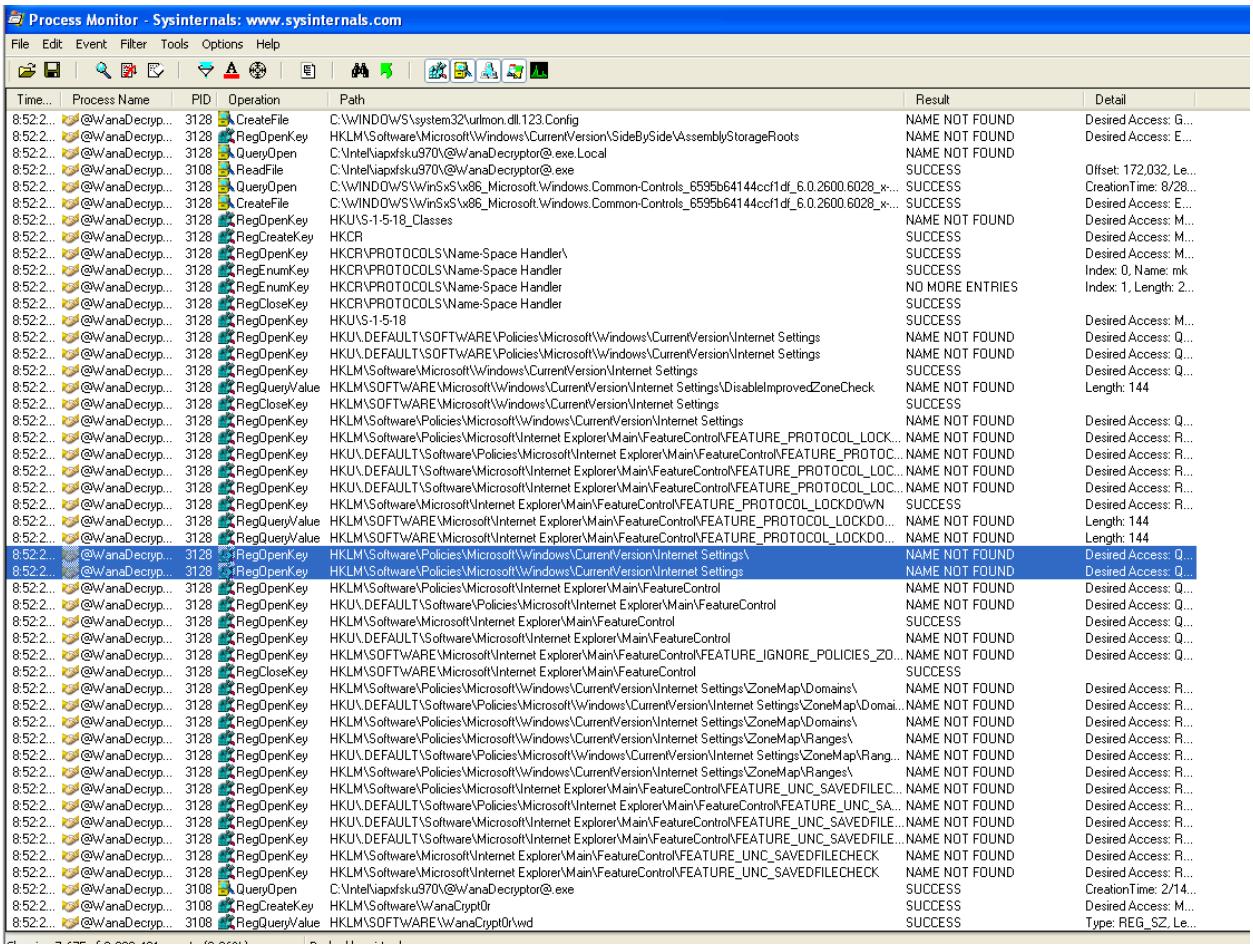
The malware was also found to have created a few restore point on the system, probably to make it persistent when it has been deleted.



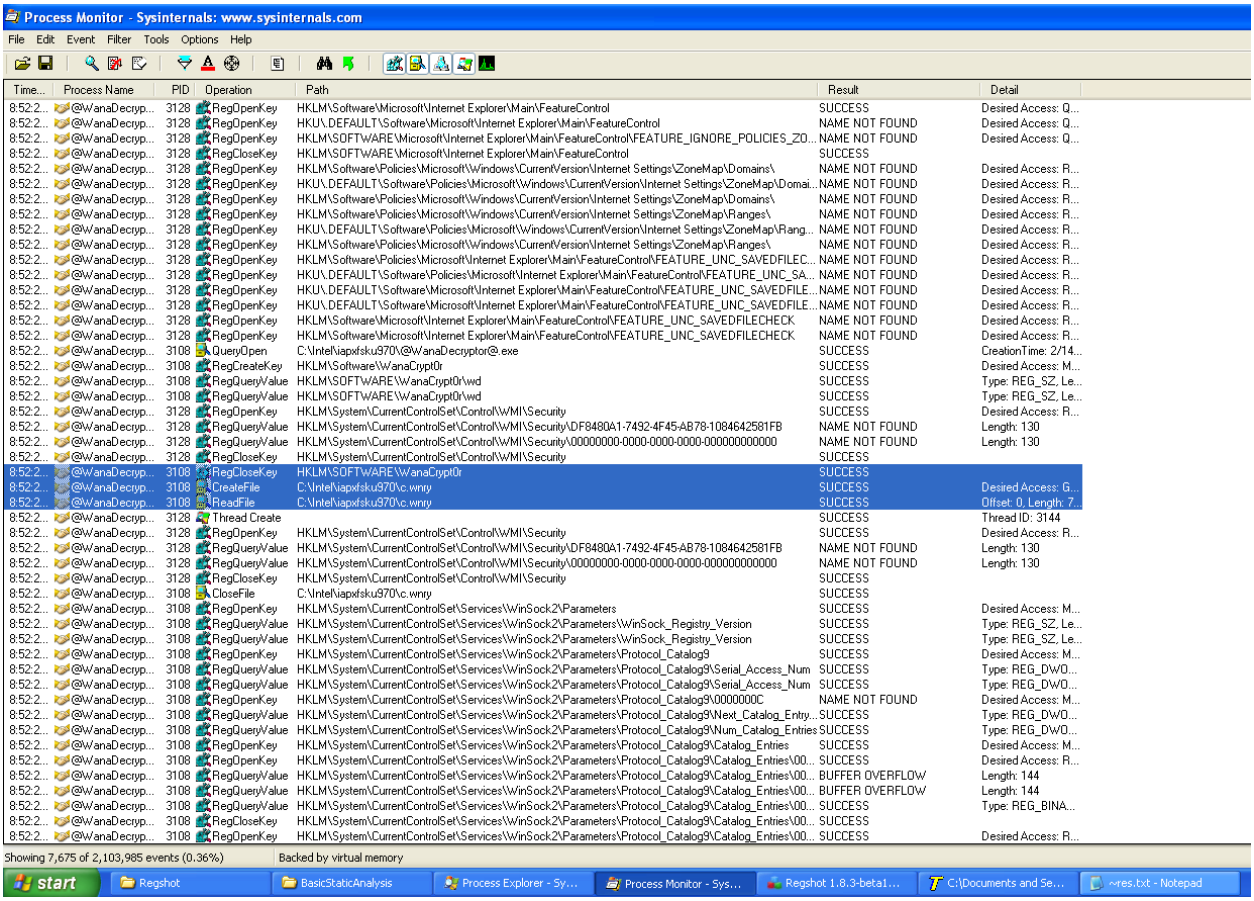
After enabling the filters for both wanadecryptor and its parent process-tasksche.exe, I observed that the malware creates a few prefetch files (running a few processes for the very first time), which will be interesting to investigate further in a forensic investigation using forensic tools.



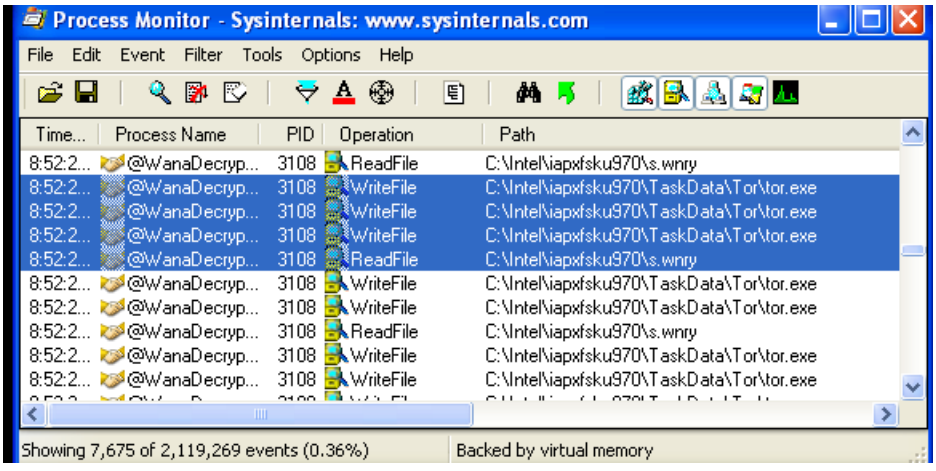
It also queries and opens various system files pertaining to various applications installed and services running on the machine, probably for the use of encryption.

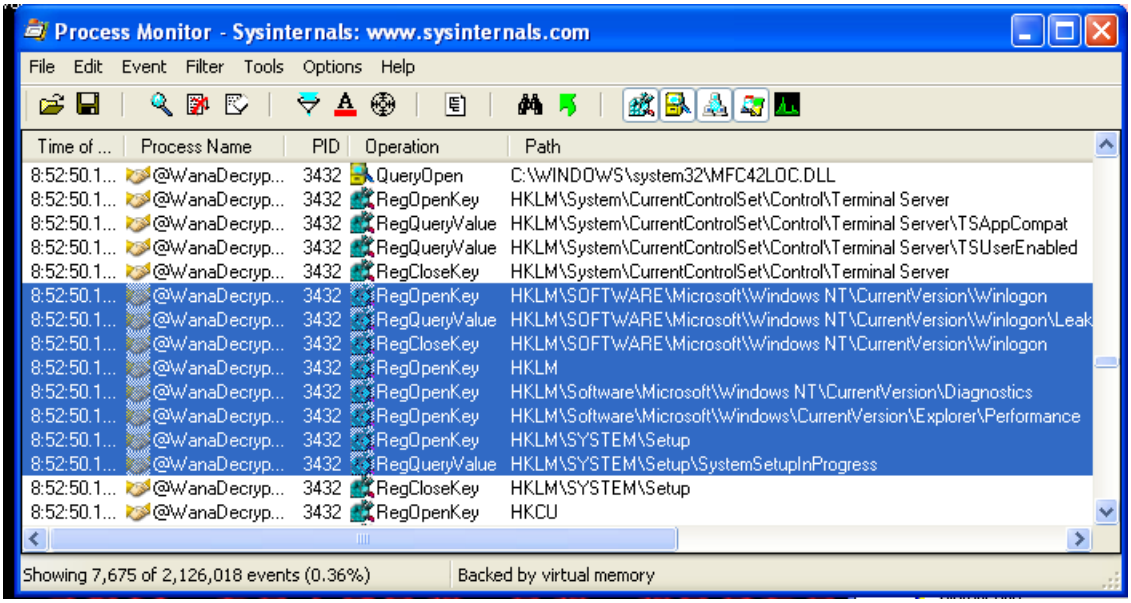


In addition, it also modifies several Internet Setting related registry keys, possibly related to Internet Explorer and its services.

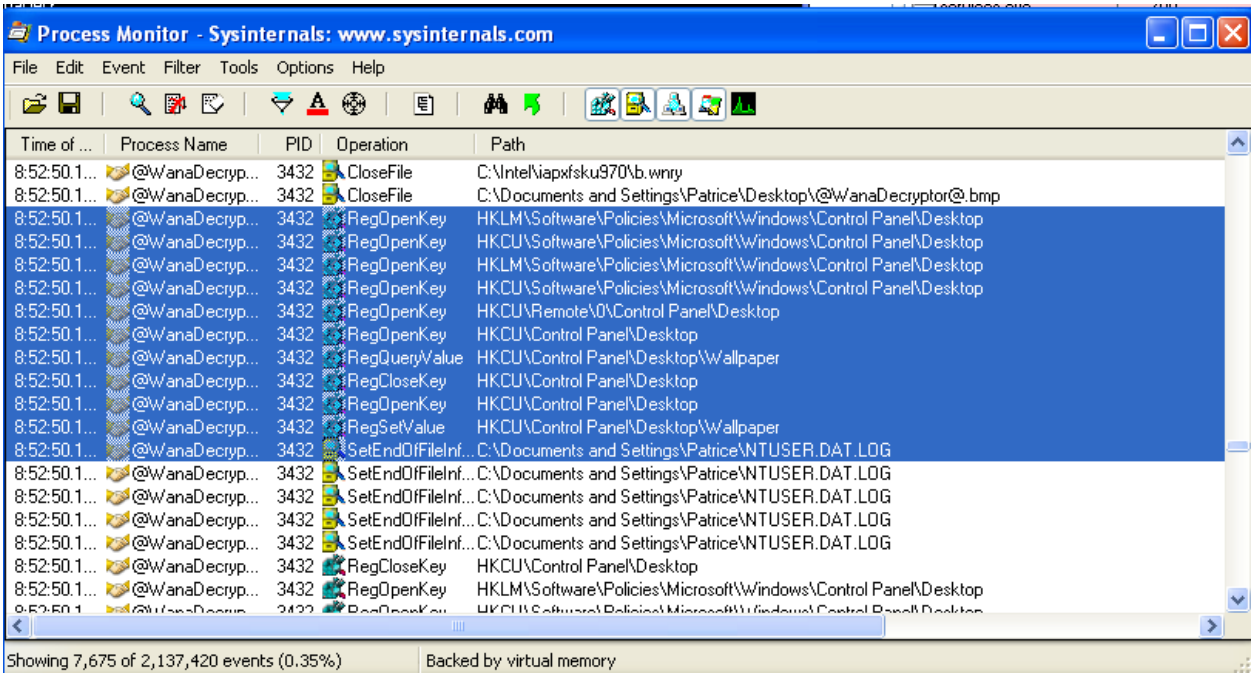


The malware at this point has started to create its dependencies, namely c.wnry which contains the target IP address and the tor information. As seen below, the process s.wnry has been invoked to execute the software and queries are made to the TaskData directory, which most of the Tor-related information are stored.



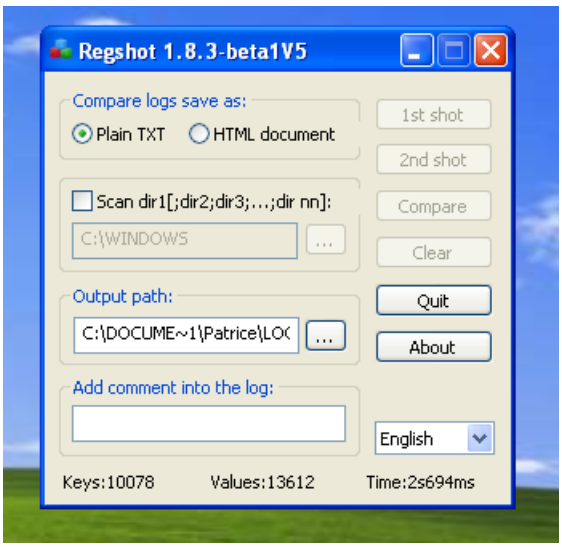


Thereafter, to make the malware persistent, wanadecryptor creates several registry keys for persistence (namely Winlogon, Setup) and also reads the keys relating to performance and diagnostics to probably monitor the activities and resource usage taken up by itself and other processes running on the computer. It also changes several registry keys for the Current user (HKCU), with regard to the desktop and wallpaper, predominantly using the Control Panel Application to accomplish this.

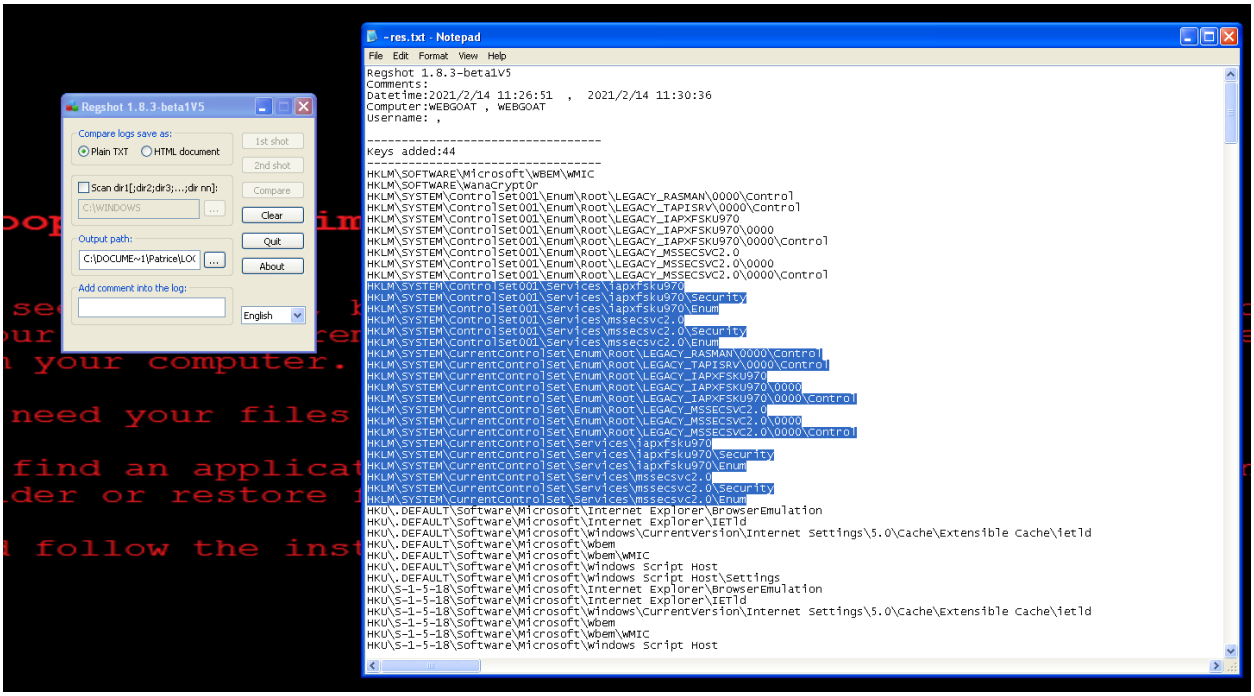


3.2.4 Regshot

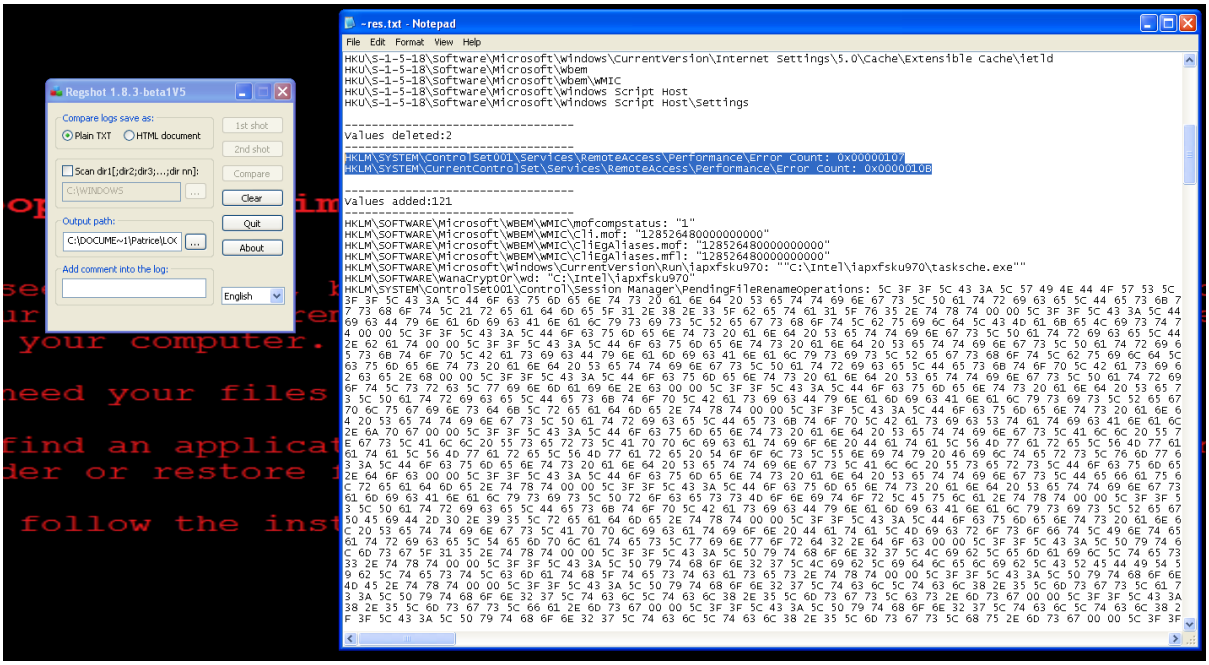
The next tool that we will use is Regshot, which will allow us to compare the changes in the specific registry keys before and after the malware is executed. Since Process Monitor only gives the registry keys that are process-specific when using the filters and it is likely the malware analyst would miss out one or 2 registry changes when looking at ProcMon for the changes (excluding the fact that it is time consuming), this tool simplifies the malware analyst's job.



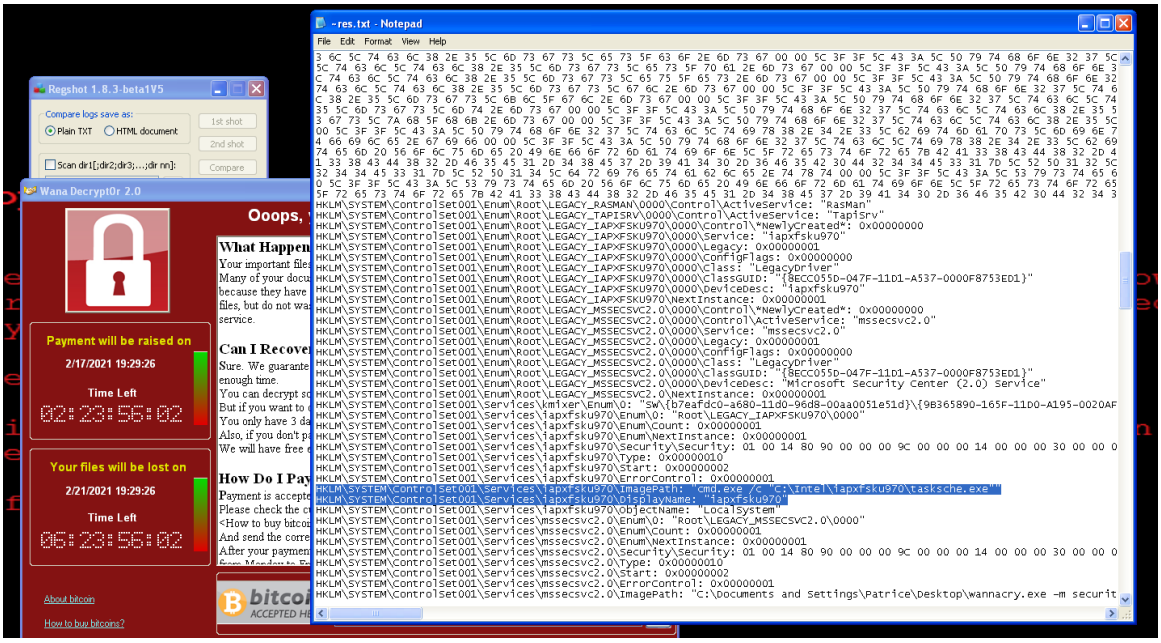
A 1st shot of state of the registry is taken before the malware is executed.



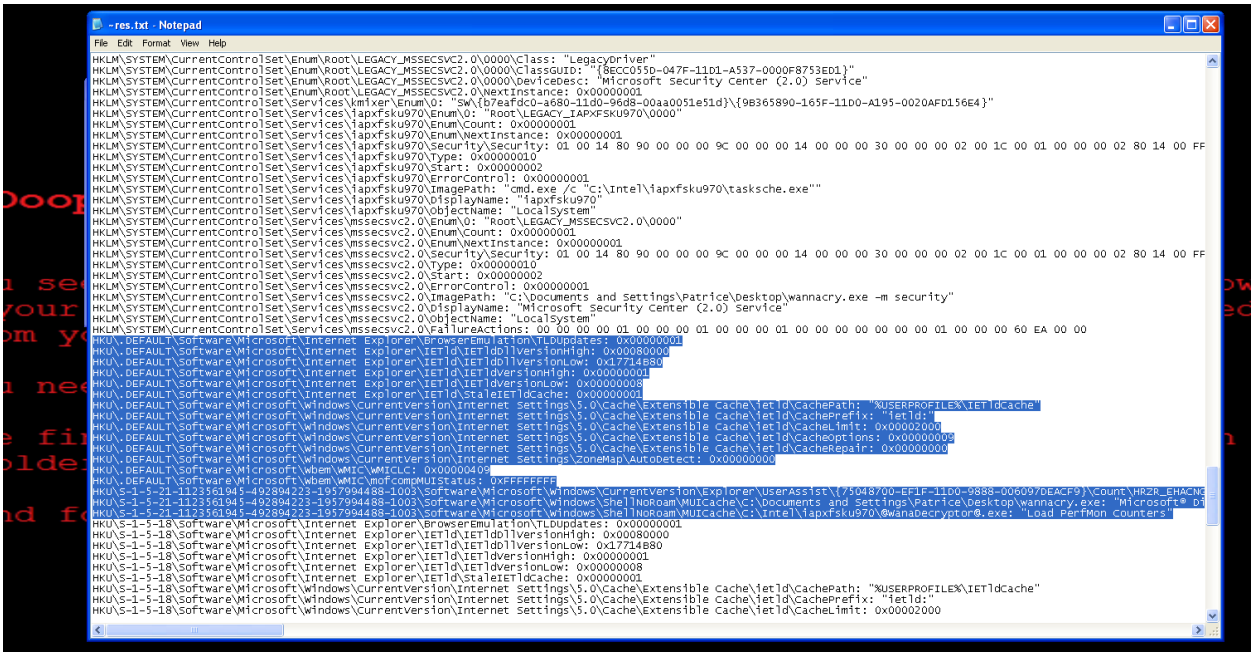
In the keys added section, several new keys relating to ControlSet001, which is the last set of that the computer boot with so it is likely that these settings will get loaded when the computer turns on from a shutdown. CurrentControlSet on the other hand has got to do if the user tries to restart the computer, so that the settings configured in the current instance will be applied after the computer restarts, making the malware persistent (since Windows overrides ControlSet001 with CurrentControlSet following a restart).

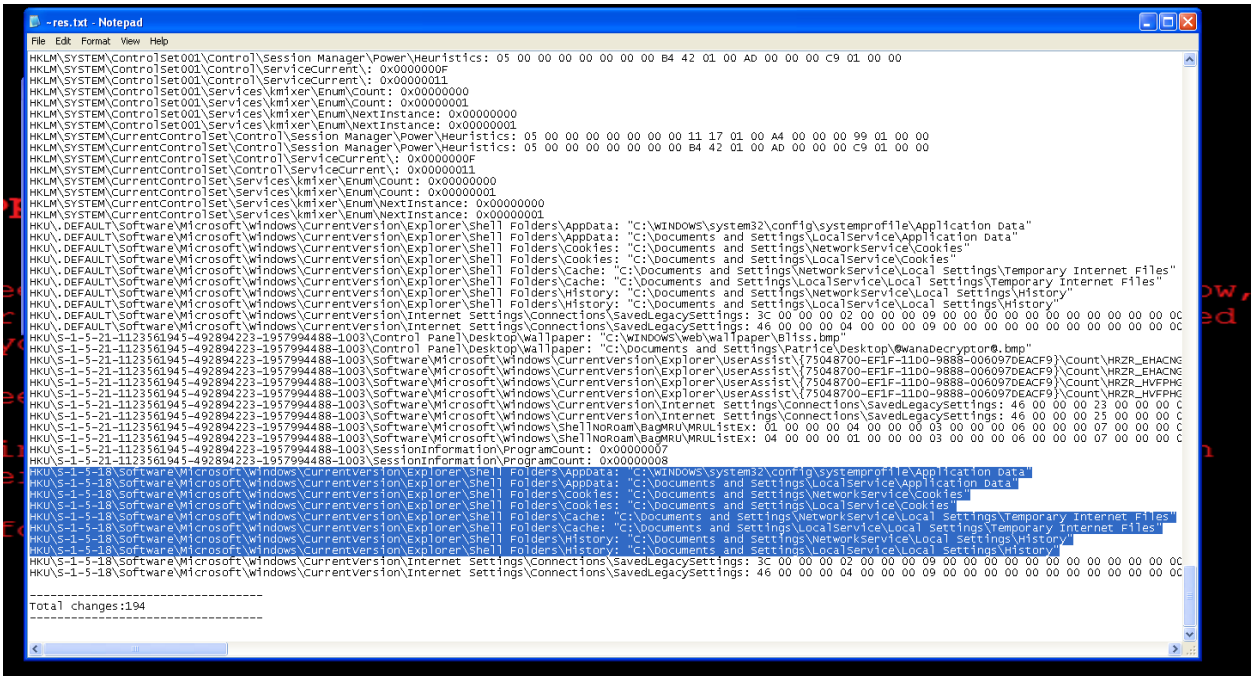


After it is executed, a second shot is taken and the “Compare” functionality allows a quick and easy way of discovering the keys that were added, deleted or modified. In this case, the keys that were added and modified have precedence over those that were deleted since these are telltale signs of additional functionalities added by the malware to create persistence and/or make detection or deletion of the malware more cumbersome.



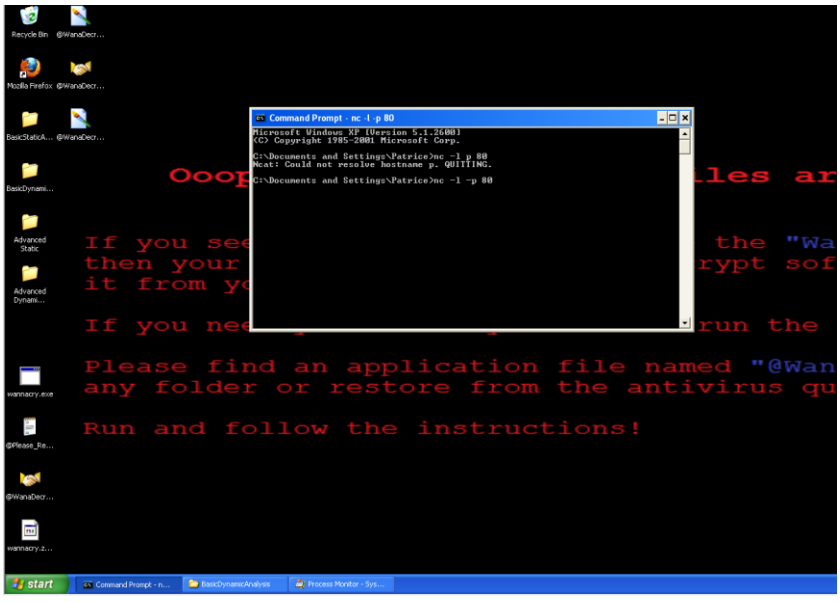
In the keys modified section, there are several changes made, including having a specific folder in services of HKLM that deals with the malware execution and display name. In addition, several Internet-based registry settings are also being modified as shown below. Shell Cache settings were also changed.



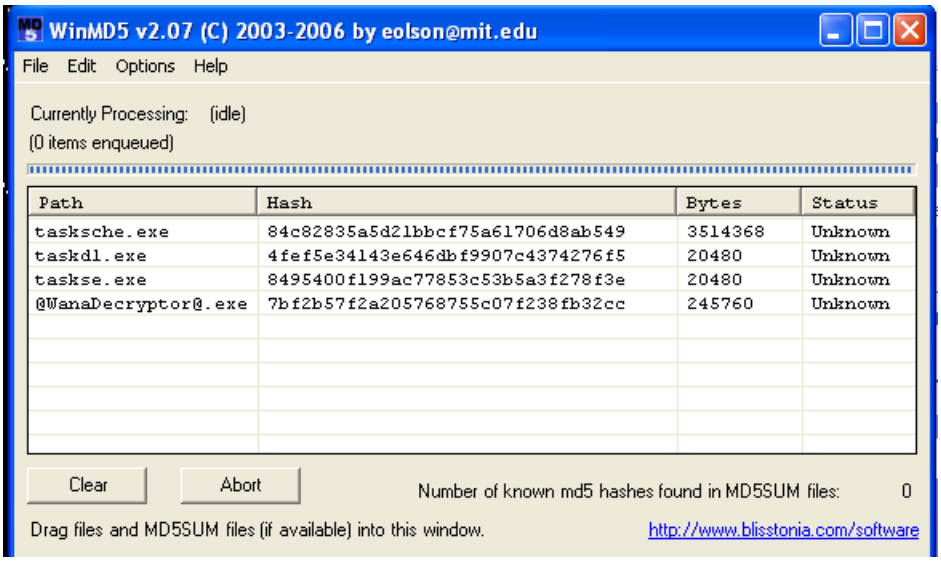


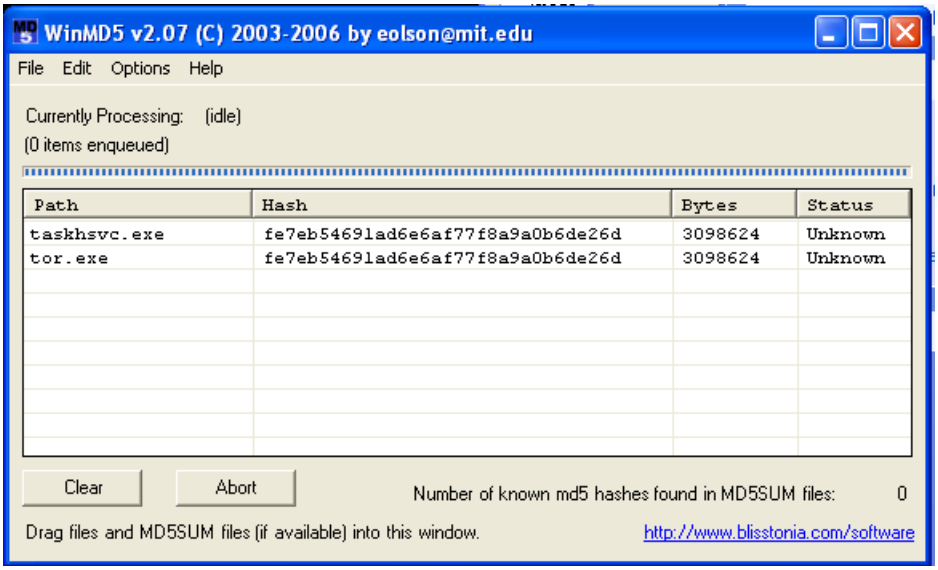
In addition, several shell folders containing appdata, cache and internet-related artifacts are also modified.

3.2.5 Netcat and WinMD5



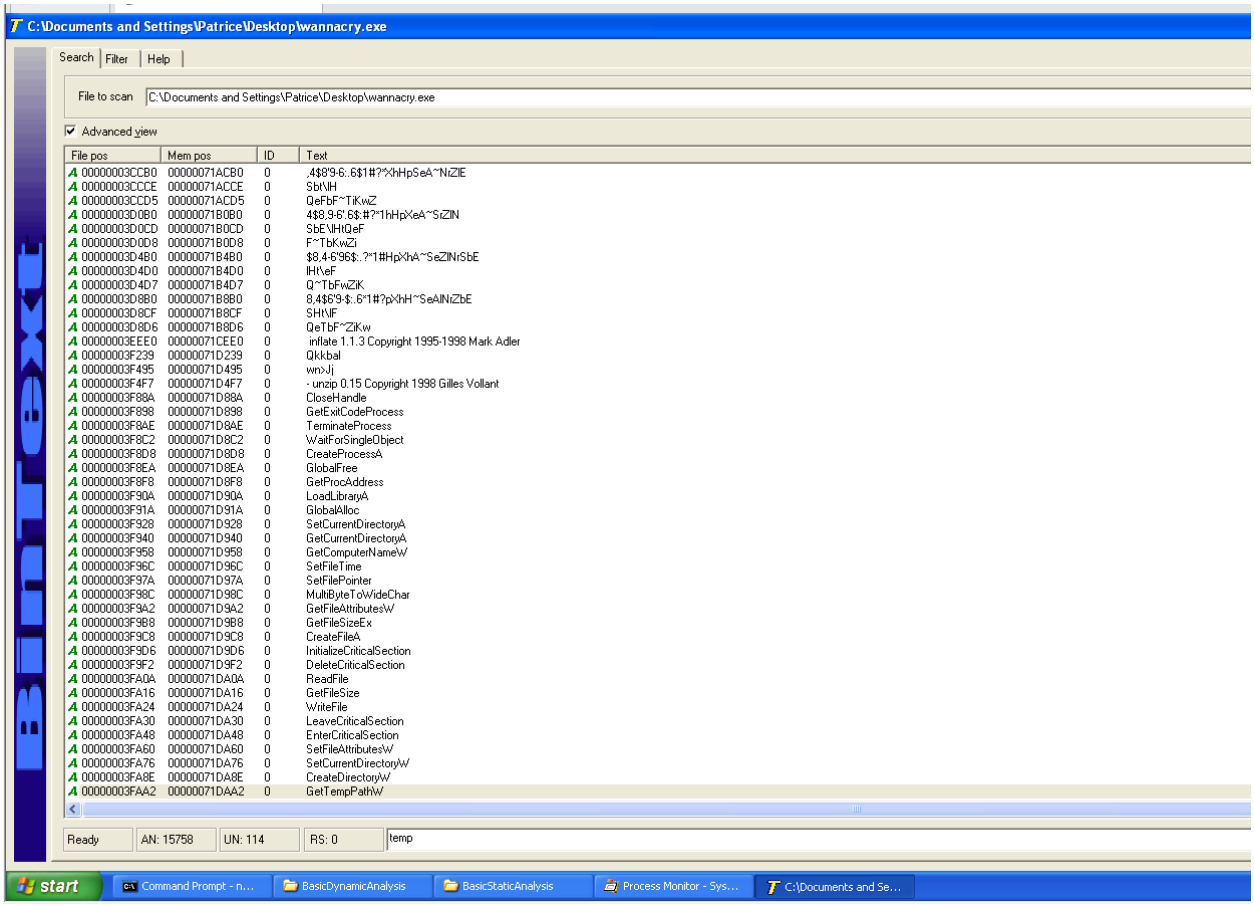
Netcat did not pick up any network activity, even though apateDNS did, probably because the Internet adapter was disabled with host only mode turned on. Port 80 is used in this case since the malware attempts to connect using http to the C2 Server.





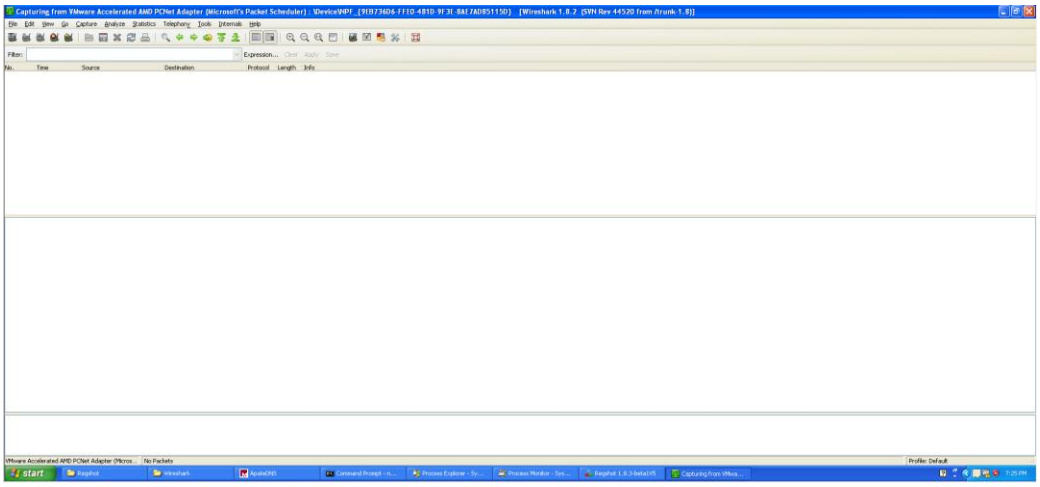
Here are the various md5 hashes of the created subprocesses and parent processes for verification. It can be noted that taskdl and taskse are children of tasksche.exe and taskhsvc and tor.exe have the same hash, meaning that it is likely that the Tor service was disguised, but both are indeed the same malicious Tor service instance.

3.2.6 Changes in strings (Bintext)

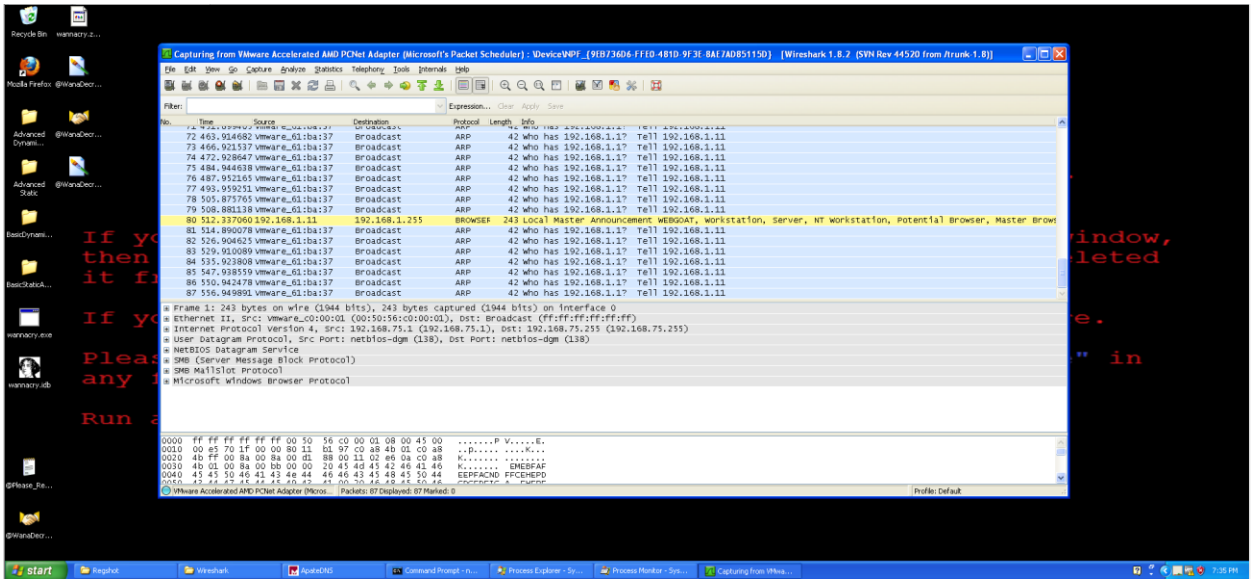


In Bintext after the malware was run, there were additional API calls to GetTempPathW, which is probably where the malware had loaded some of its files.

3.2.7 Wireshark



I also ran wireshark while the malware executed but this did not yield meaningful results as it was mainly the VM trying to communicate to the local host machine to resolve ARP requests and did not capture the killswitch url or the various IP addresses as listed in apateDNS.

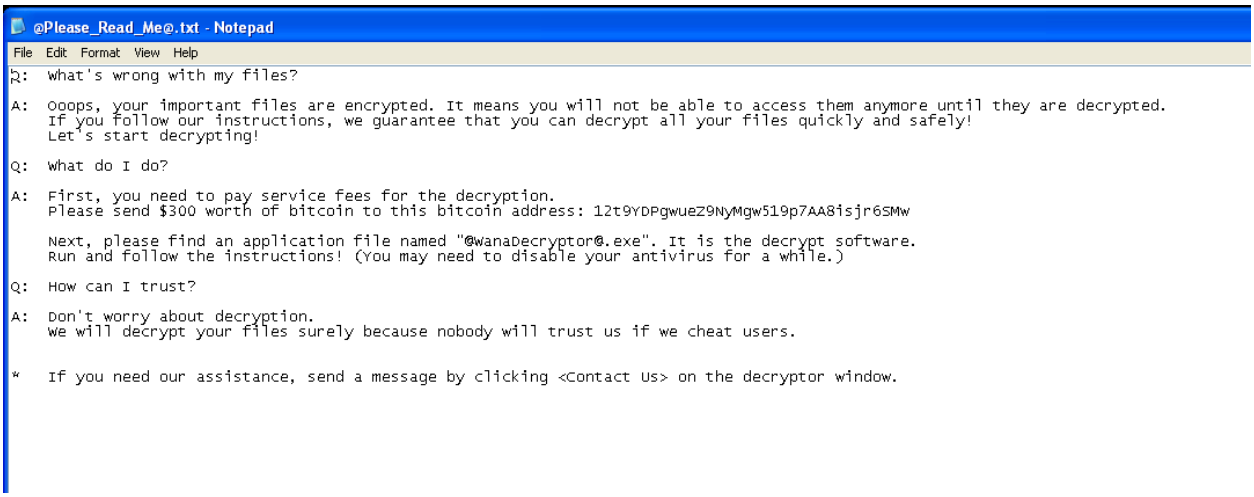


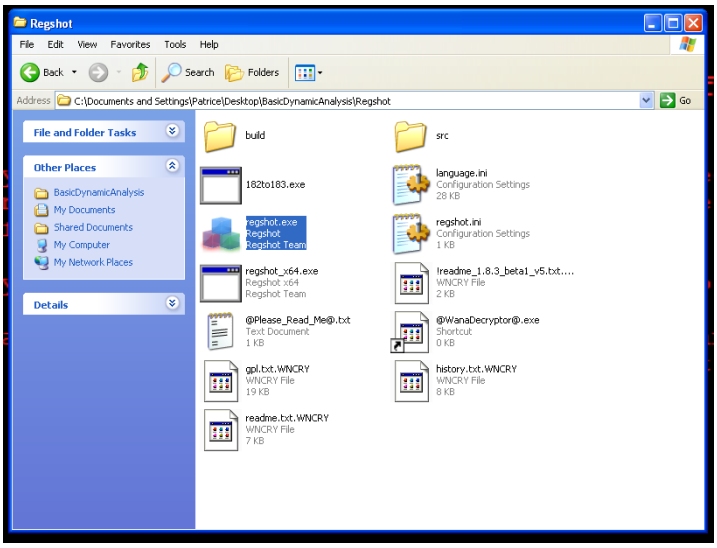
3.2.8 General Observable changes

Now I will be moving on to talk about the general observable changes in the system after the malware had executed.

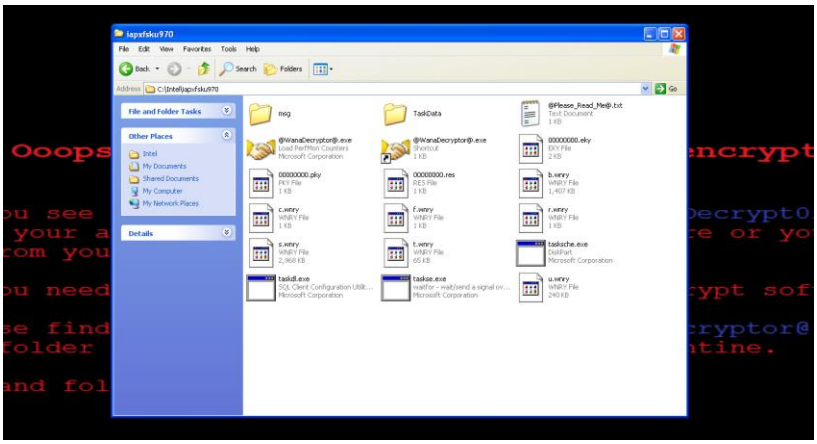
This was the state before the malware was executed and the desktop still looks fine with no indicators of compromise.

After the malware was executed, the wana decryptor application appears with the wallpaper changed to some instructions and a few other malware related files and a QnA note on the desktop. Opening the note will show instructions which the user can follow to pay the ransom. In this case since it is a simulated attack, there is no need to pay the ransom, but simply revert to the original state once the dynamic analysis is completed.

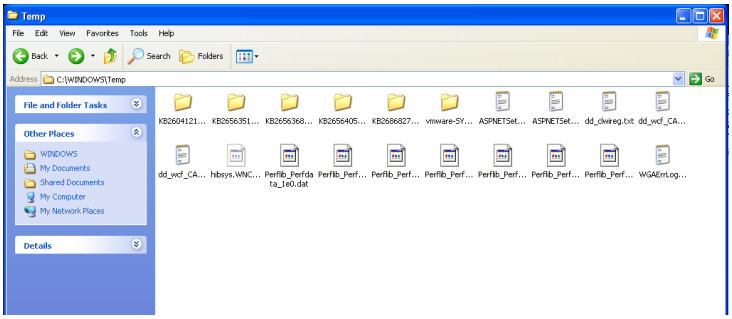




In ordinary application folders like this one for regshot, there are also individual ransom notes and readme files and other dependencies with the extension of ".WNCRY".



In addition, the ransomware also creates its dependencies in the C:\Intel\iapxssku970 folder. These are slightly different from those in user folders, as the dependencies have a different extension of ".wnry".



In addition, several prefetch files are created in the Temp directory after the malware has completed execution.

3.3 pr2.exe removal

Removal is the stage where the malware is eradicated completely or partially, depending on the tools to disinfect the target victim machine.

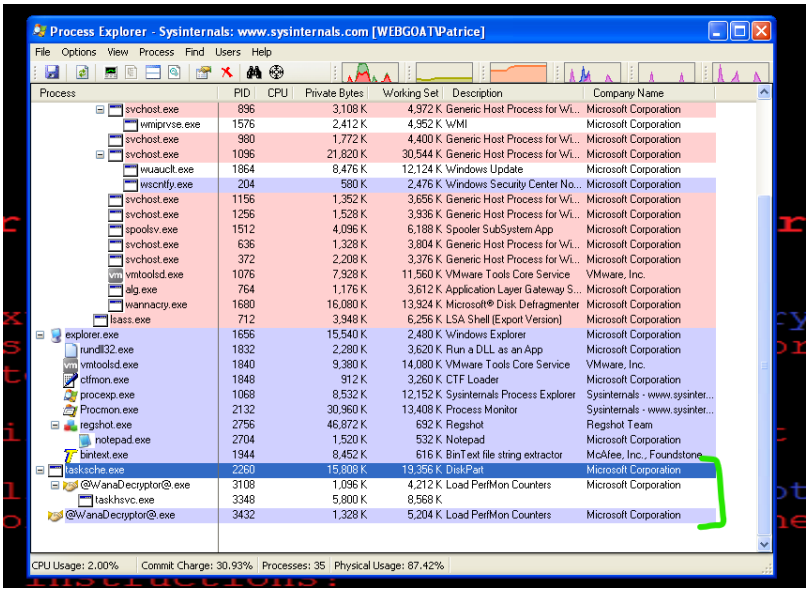
3.3.1 Automated Removal

Once the wannacry malware is detected on a computer, it is advised that the computer is disconnected and isolated from the network (through wireless and wired means) since this variant is a worm that connects to a C2 server and also may infect other computers on the network if left exposed.

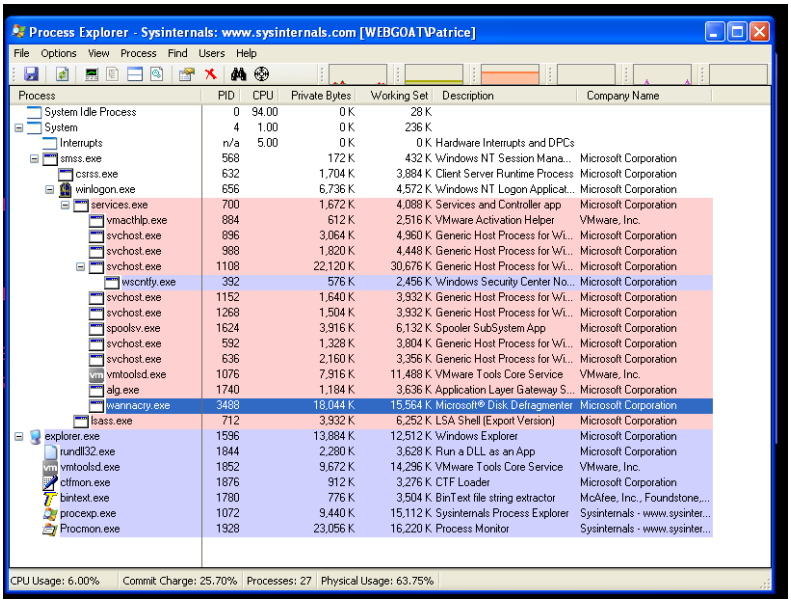
This way of removal is recommended to remove all traces of the malware. Since the malware is detectable by most AV engines today, it is recommended that the user of a computer infected with pr2.exe installs a reputable antivirus software such as Avast, Avira or Malwarebytes with the latest patches and virus definitions before performing a full/deep scan to rid the malware and its dependencies appropriately or quarantine them to be sent for analysis by a malwarse analyst.

Due to it being ransomware, lodging of a police report and reporting of this incident to a local Computer Emergency Response team as well as not paying the ransom is advised, since it is critical to get professional help for complete malware removal and preventing a reoccurrence. If the user still wishes to have his/her files back, he/she can go to the No more ransom project at: https://www.nomoreransom.org/ to decrypt the files for free since the decryption keys are made freely available before running the antivirus software.

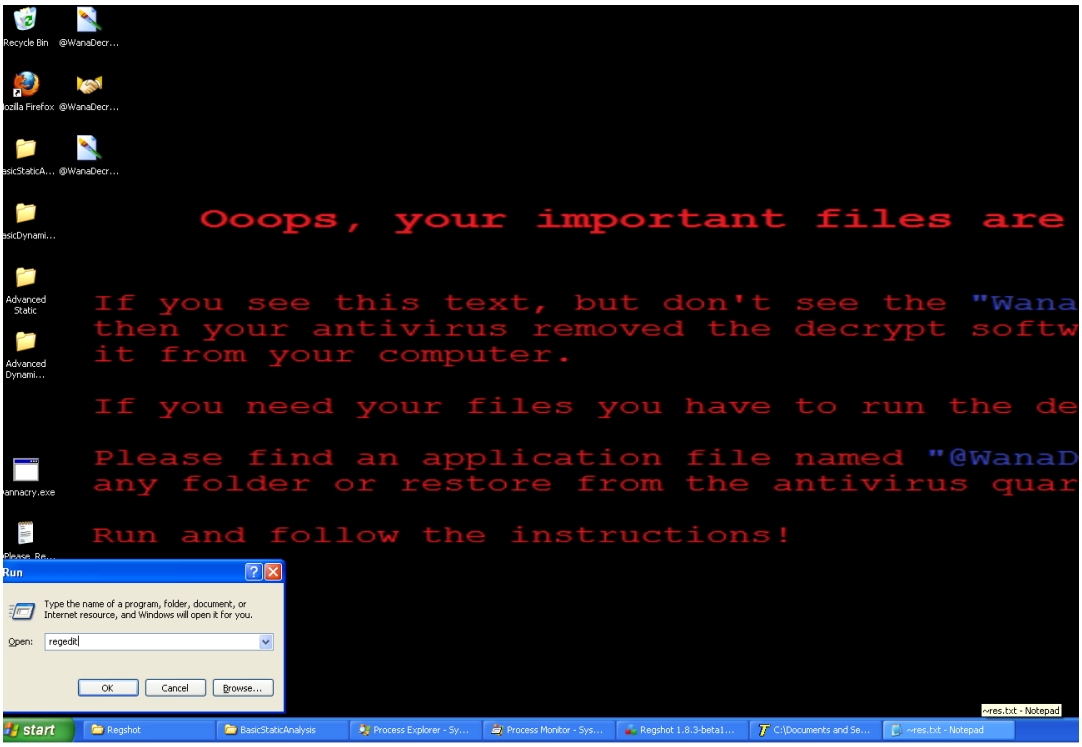
3.3.2 Manual Removal with limited success

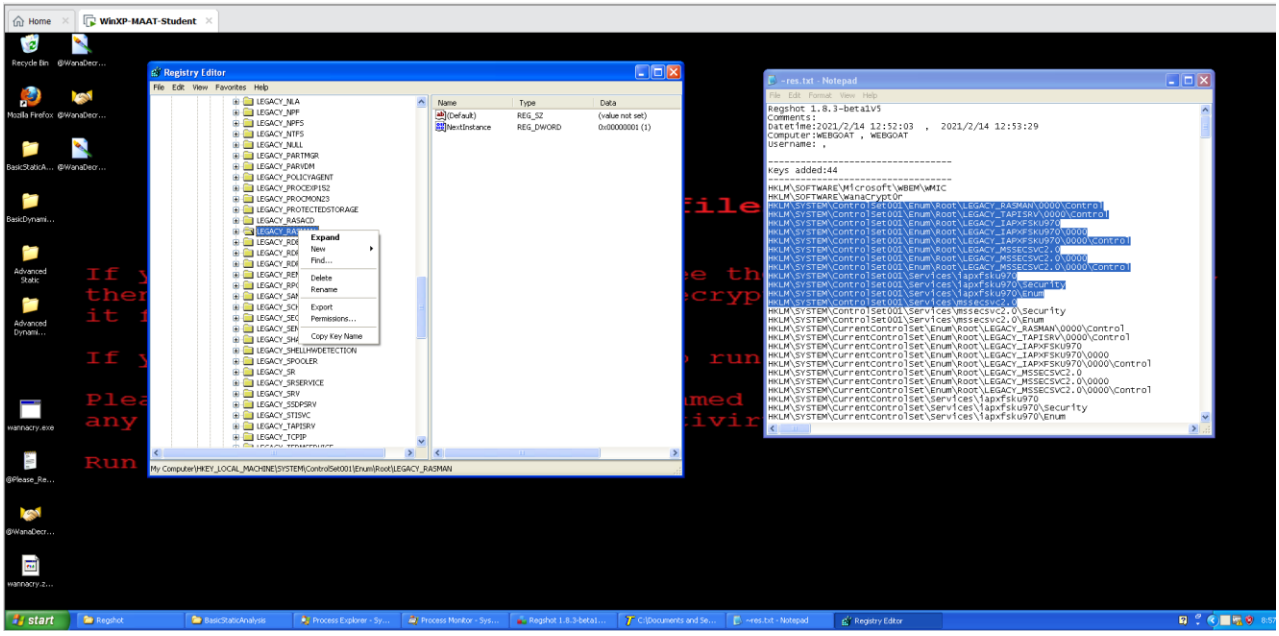


To remove the malware manually, kill the 3 processes highlighted in green as shown above.

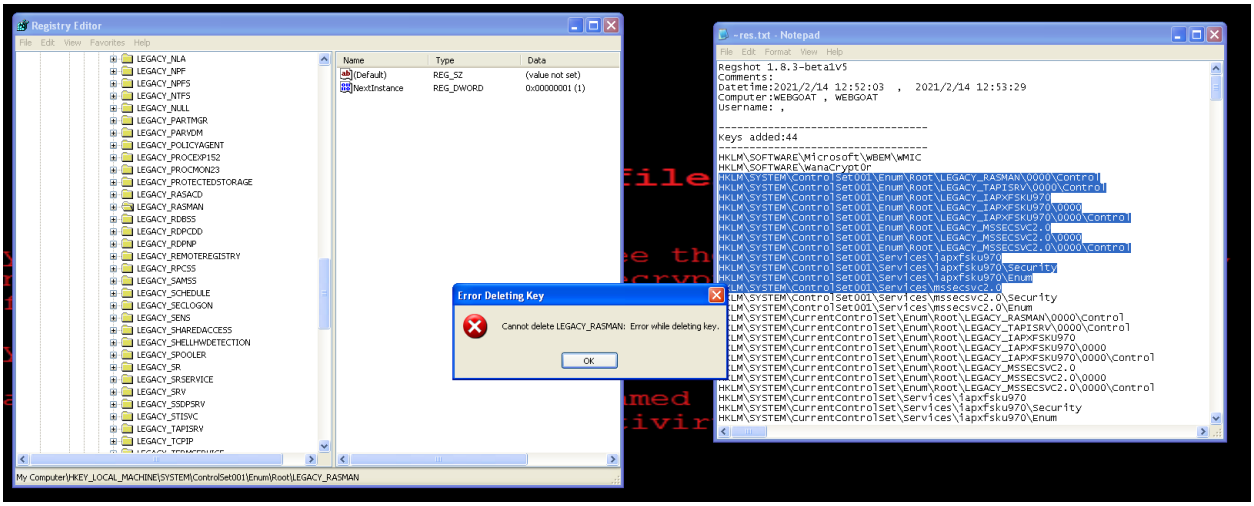


In addition, all traces of malicious processes (wannacry.exe and svchost.exe containing wscnify.exe) should be terminated. Thereafter, the user can proceed on to attempt to remove the registry values created by the malware using the regedit tool.

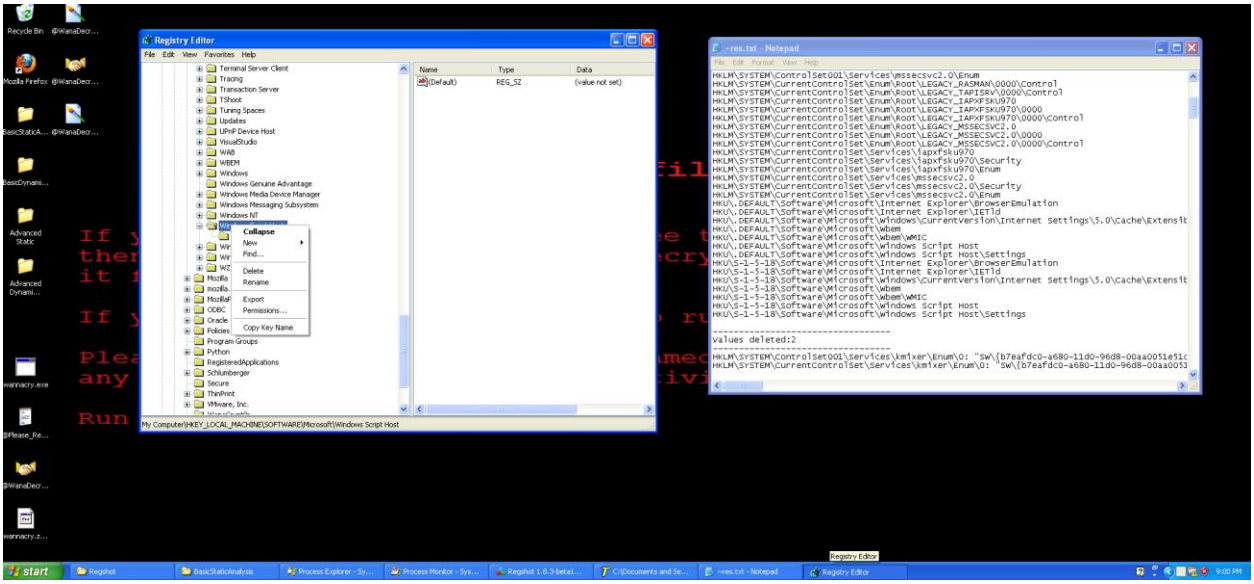




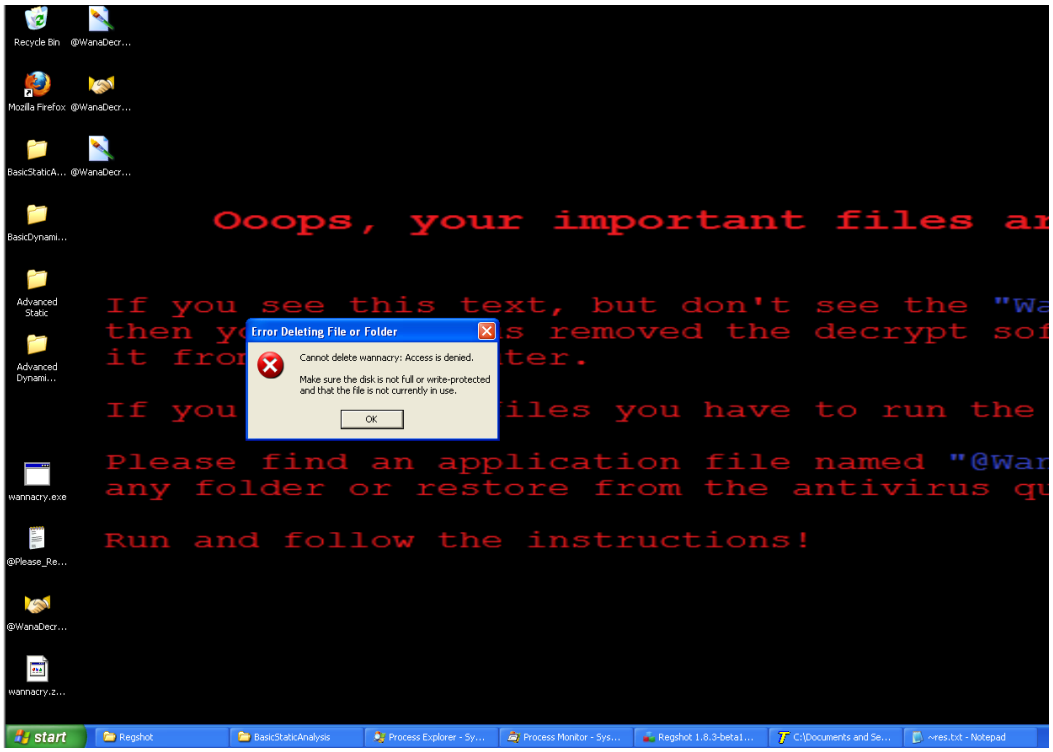
Based on the keys created or modified as reflected in regshot, the user can manually try to delete or modify the corresponding keys and values. Note that this is a rather tedious process since there are over 100 keys that were added and/or modified.



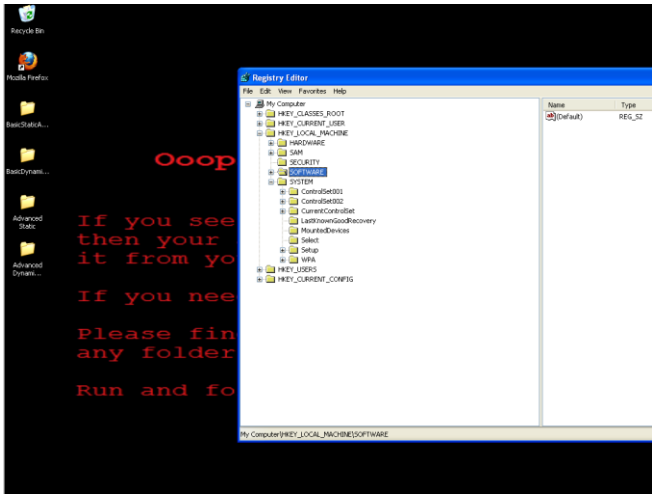
The screenshot above shows that doing this can have limited success since the malware has most likely gained ownership of the key(s) and made deletion of the keys impossible when the user tries to perform this action.



Some keys like the one selected could be successfully deleted but then again, this is a rather tedious process and can be automated using an Antivirus software.



After deleting and modifying some keys, I tried to remove some of the artifacts on the desktop that the malware created to no avail so the best option would be the Automated way, as well as seeking expert help. To remove the artifacts on the Desktop, delete the wannacryptor folder in regedit under HKLM\SOFTWARE



3.4 General Analysis of pr2.exe

We will be summarizing our findings of the malware that we have discovered using various static and dynamic analysis methods.

The malware is a form of ransomware that contains several components, each that has its own functionality, making the malware modular and dependent on other components.

Throughout our analysis we have seen the Dropper and Infection (using the description of MS Disk Defragmenter), the resource loader (which takes the form of the tasksche “diskpart” utility, which loads the various resources to be used such as the various .wnry files. We have also seen the C2 phase where wanadecryptor through LODCTR.exe (found in Process Monitor) executes the ransomware functionality and traces the infections and payments and sends these back to the C2 server. In this case, we did not see the application used in the deployment and destruction phase so that may have been present in another variant.

TABLE I MAIN COMPONENTS OF WANNACRY

Phase	Execution Component	WannaCry		
		File (Internal) Name	*File Description*	*SHA256*
Deployment	Export PlayGame	launcher.dll	Inject through Doublepulsar backdoor	9411c59a83c8c32a925d53a902bef168ebe5b403a88ab4d8dfe807fd7435dd9e
Installation	Dropper and Infection	mssecsvc.exe (lhdfrgui.exe)	Microsoft® Disk Defragmenter	24d004a104d4d54034dbcffc2a4b19a11f39008a575aa614ea04703480b1022c
	Resource Loader	tasksche.exe (diskpart.exe)	DiskPart	ed01ebfbc9eb5bbea545af4d01bf5f1071661840480439c6e5babe8e080e41aa
Destruction	Encryption DLL	kbdlv (3.13)	Latvia Keyboard Layout	1be0b96d502c268cb40da97a16952d89674a9329cb60bac81a96e01cf7356830
Command-and-control	Trace Infection and Payments	@WanaDecryptor@.exe (LODCTR.EXE)	Load PerfMon Counters	b9c5d4339809e0ad9a00d4d3dd26fdf44a32819a54abf846bb9b560d81391c25

Source: https://www.coursehero.com/file/33565476/20180369-finalpaperpdf/

3.5 Malware Defenses

The best way to prevent this infection is to update Windows OS to the latest version with the latest security patches, because even though pr2.exe is detectable using Windows Defender, pr2.exe exploits an old SMB flaw in the Windows OS. It is also recommended to maintain an Antivirus with the latest AV database definitions. In addition, users should not click on or download suspicious files/links to prevent infection.

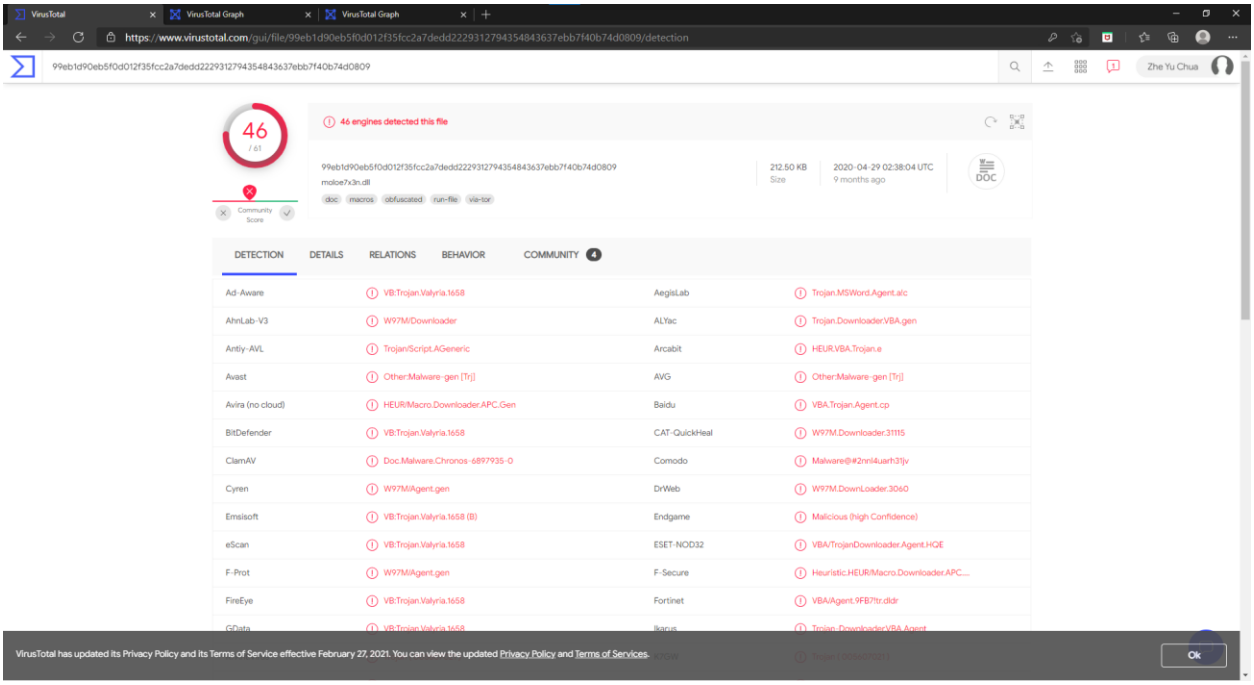
Malicious Document Analysis – 2018-04 GandCrab-Swarm (Document Carrier)

4.1 Basic Static Analysis

Basic Static Analysis is conducted to understand how the malware works without running it. This allows us to understand how the malware works through its metadata, such as the strings and macros which may have some host and network based indicators to show what is it planning to achieve.

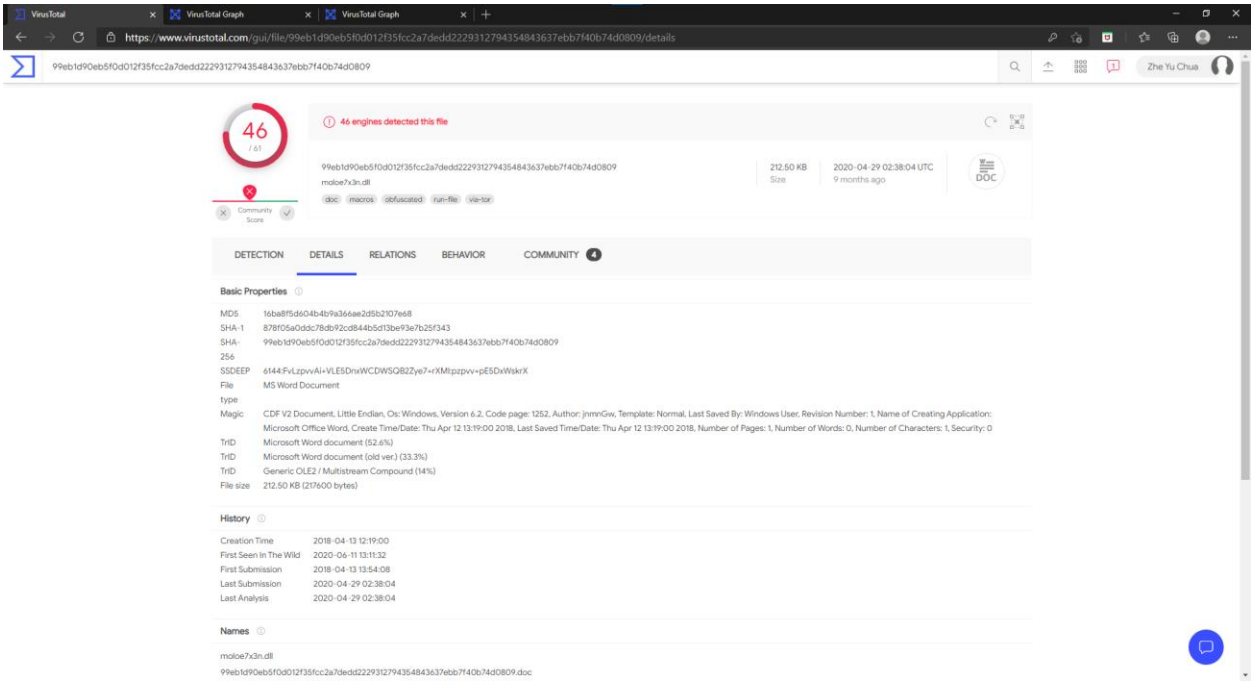
4.1.1 VirusTotal Scanning

VirusTotal is an online tool used that aggregates many antivirus products and online search engines. This allows us to see what the many anti virus engines will detect regarding the malware or malicious document and help us discover what type of malware it could be.

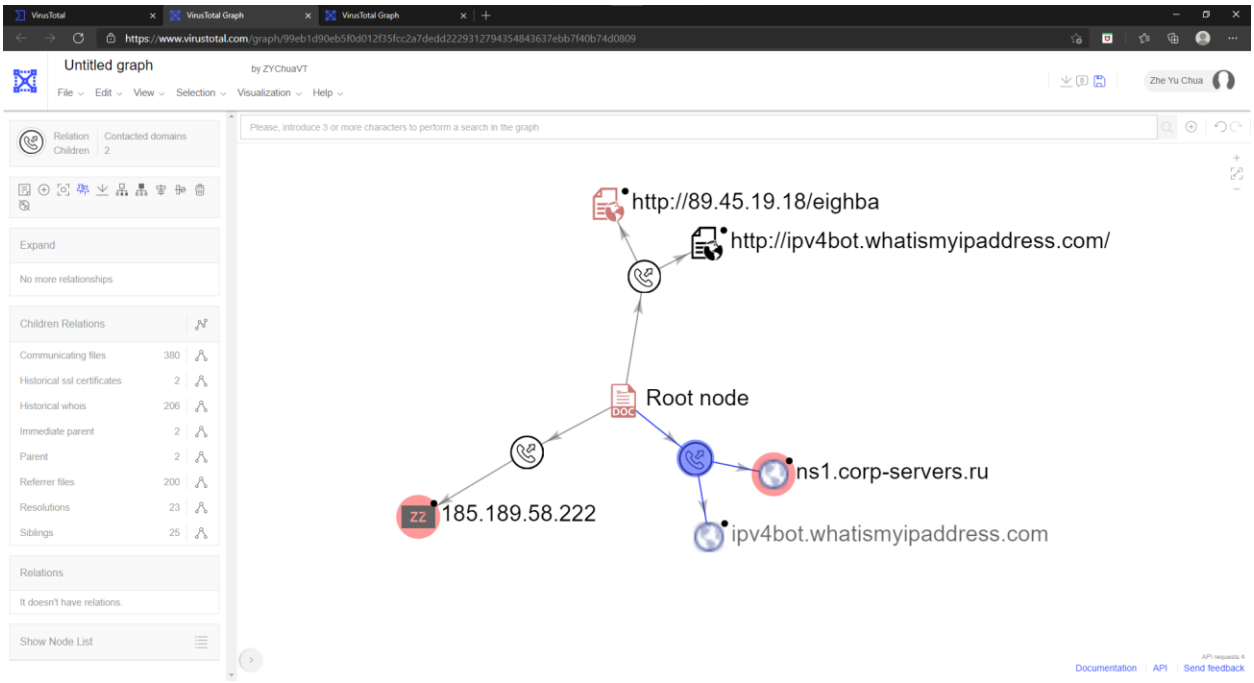


Based on the virus total screenshot uploaded above, 46 out of the 61 malware engines detected that the document is a malicious, which indicates that the document is indeed malicious and not a false positive. Most of the malware engines also detected it as a Trojan.Valyria. After doing some research on it, it means that it is a malicious Microsoft

word document used to distribute other malware, which proves the fact that it is true as this word document contains the GandCrab Ransomware.



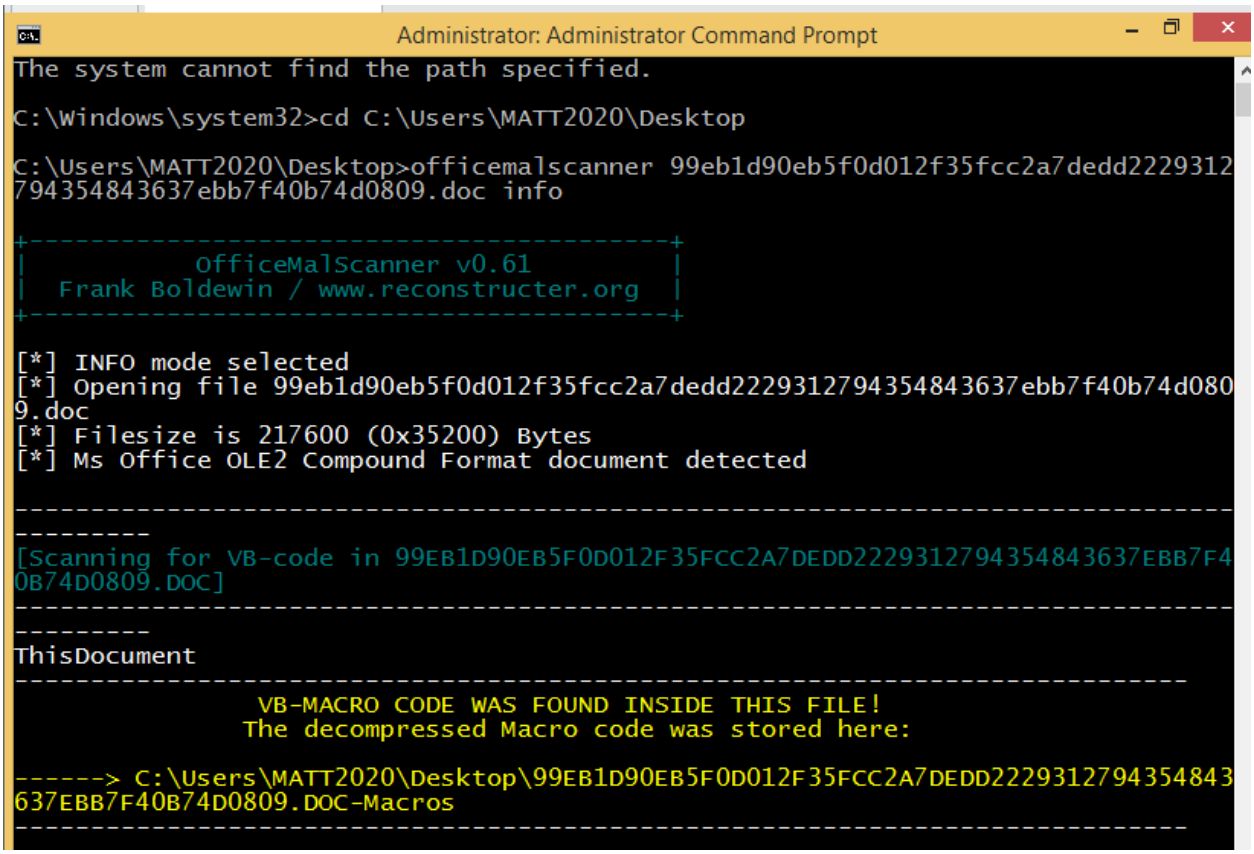
With VirusTotal, we are able to gather some more information on the malicious document, such as the hashes, what file it is, which in our case is a MS Word Document. Other information like the file names associated with the file that other users of VirusTotal have uploaded.



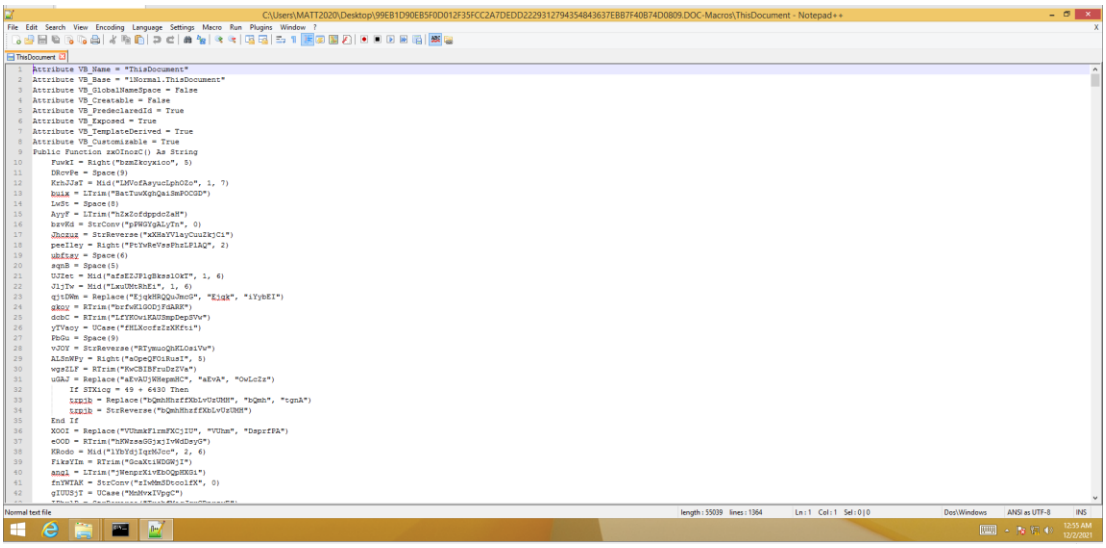
We can use VirusTotal relation to help us understand the relationship between files, URLs, domain and IP address. For the malicious document, as shown in the graph, it contacts 2 URLs, 1 to a private IP address and another to whatismyipaddress which will possibly look for the victim public IP address. Domains contacted are ns1.corp-servers.ru, which looks like the malicious document is trying to establish a connection to a server in Russia. Lastly there is an IP address 185.189.58.222, which looks like it is associated with MassiveGrid Ltd and search results show the IP address associated with the malware in the document.

4.1.2 officemalscanner

To get the macros, we need to cd into the directory that the malicious document is stored is, in this case the Desktop. After that we run the command officemalscanner 99eb1d90eb5f0d012f35fcc2a7dedd22293127943548436 37ebb7f40b74d0809.doc info.



Using the command will lead to Macro ThisDocument being found, which we will open with Notepad++ to see its contents.



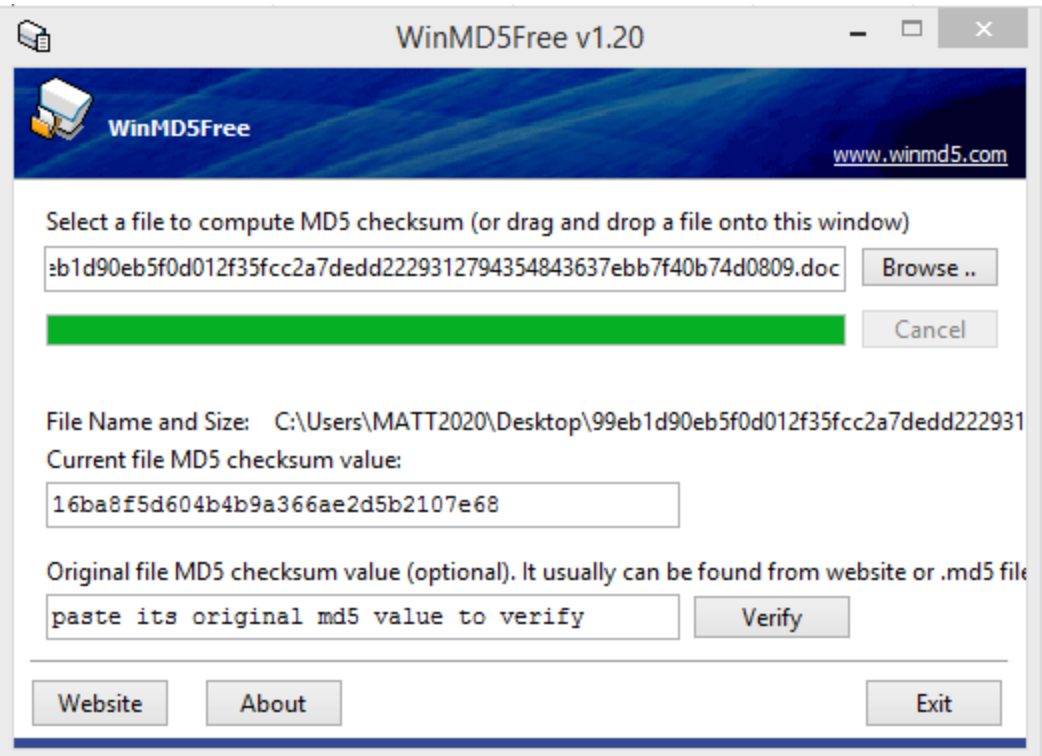
```
Private Sub Document_Open()
JEgxyx = StrReverse("QftUlOBFdtpM")
    For nBtdE = 0 To 86
        qyeCrnu = UCase("wYZXCnUtFXQdJJC")
        fbqxO = Space(7)
        djXkpSPaB = LTrim("EwsgTMFkKzaQQU")
        yiFBPee = LTrim("HlEMKxJgqU")
            If bvODwV = 256 + 2819 Then
        OzuPS = Replace("aYHjcGOjlBhjZbEpbc", "aYHj", "hstRCg")
        OzuPS = StrReverse("aYHjcGOjlBhjZbEpbc")
        nyGtX = Replace("DLqFCCPoHDKgrX", "DLq", "xKGU")
        nyGtX = StrReverse("DLqFCCPoHDKgrX")
    End If
        kZZzLm = Replace("GHjKEvKJIoKG", "GHjK", "uHzUqV")
        KzALZ = Left("JZHZMBAPOgyeSArwq", 3)
        iZVWp = Space(6)
            If iPLaWn = 99 + 8039 Then
        yuLCz = Replace("gltOzPvRxGHEEmZmrmU", "glt", "jOIk")
        yuLCz = StrReverse("gltOzPvRxGHEEmZmrmU")
```

After seeing the content, it seems to be obfuscated with random strings, and makes use of functions such as strreverse and replace to manipulate the strings to the correct ones for the malware to execute. However after trying to reverse the strings using an online tool, we are still unsure of what the malware is trying to achieve as after reversing it is still obfuscated and difficult to make sense of.

4.1.3 WinMD5

Hashing the malicious document ensures that the malware is unchanged after doing some analysis on the malicious document like using officemalscanner to extract its macro.

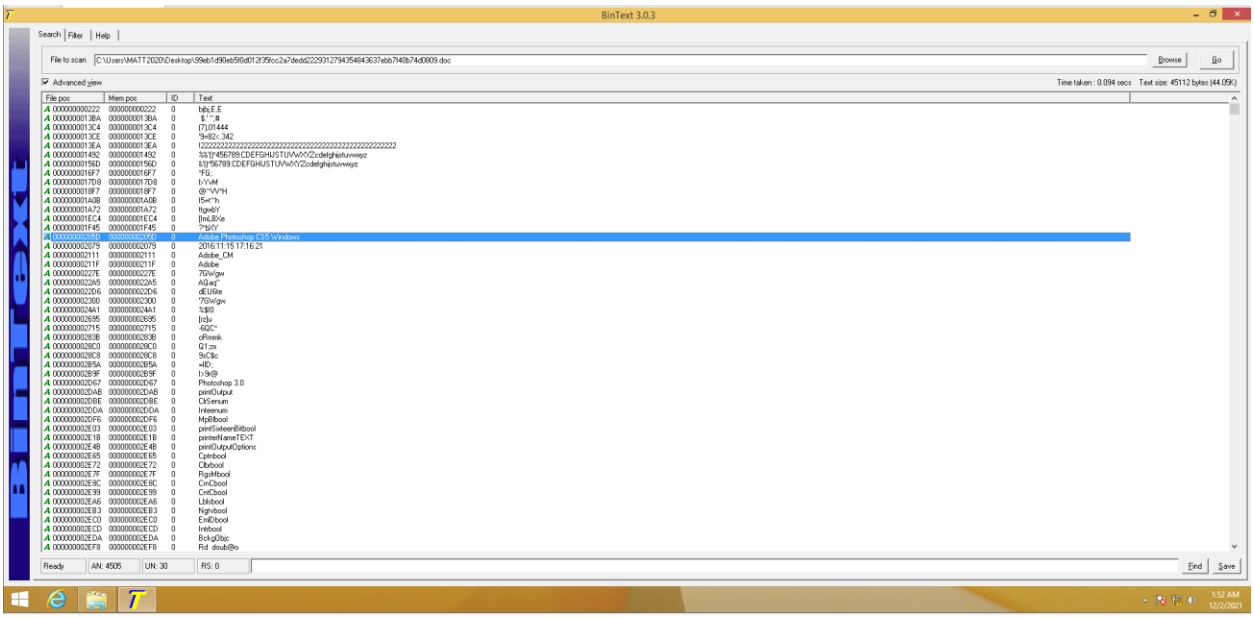
To find out the hash of the malicious document, I will be using WinMD5 which will calculate the malicious document hash.



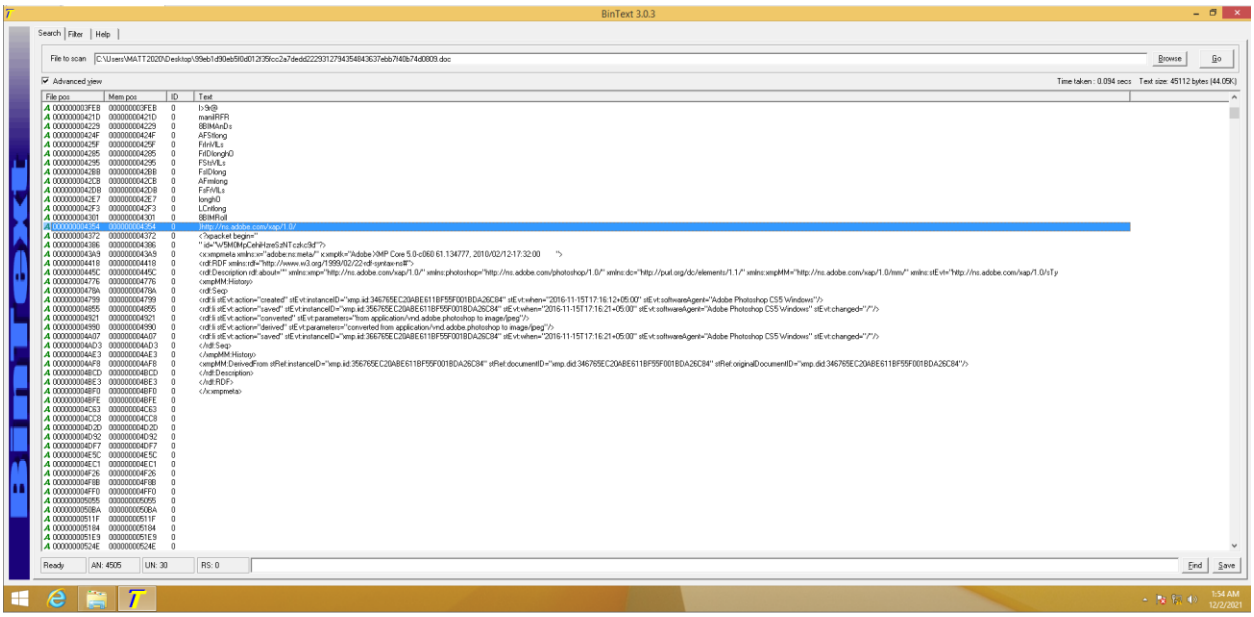
As shown by the results, the hash remains the same as the one identified by VirusTotal. This shows the malware did not change after extracting the macro from it and hence is still the same to the one that was initially uploaded to VirusTotal for the analysis in the previous section. We can continue to use the results from VirusTotal for the analysis of the document.

4.1.4 Analyse strings using Bintext

We will use Bintext to analyse the strings from the malicious document to ensure that we did not miss out anything during the static analysis.



After putting the malicious document into BinText, there are many strings identified which are obfuscated and not readable. There remains some suspicious strings, such as “Adobe Photoshop CS5 Windows” and a timestamp below it saying “2016:11:15 17:16:21”.



There are some URLs found under the strings as well such as “http://ns.adobe.com/xap/10” and “http://www.w3.org/1999/02/22-rdf-syntax-ns#”. After

doing a quick search on the URLs, it seems like they are metadata of an image that is in the word document. With the many references to Adobe Photoshop in the strings, the image could have also been edited using Adobe Photoshop before it was pasted or exported in the malicious document.

4.1.5 Summary of Static Analysis

From Static Analysis, we found out a few things about the malicious document, such as the fact that from VirusTotal that is a malicious word document used to distribute other malware, which is the GandCrab Ransomware. However, from the macros that were extracted from the document, it is heavily obfuscated which makes it difficult to analyse or find out what it is trying to do the victim.

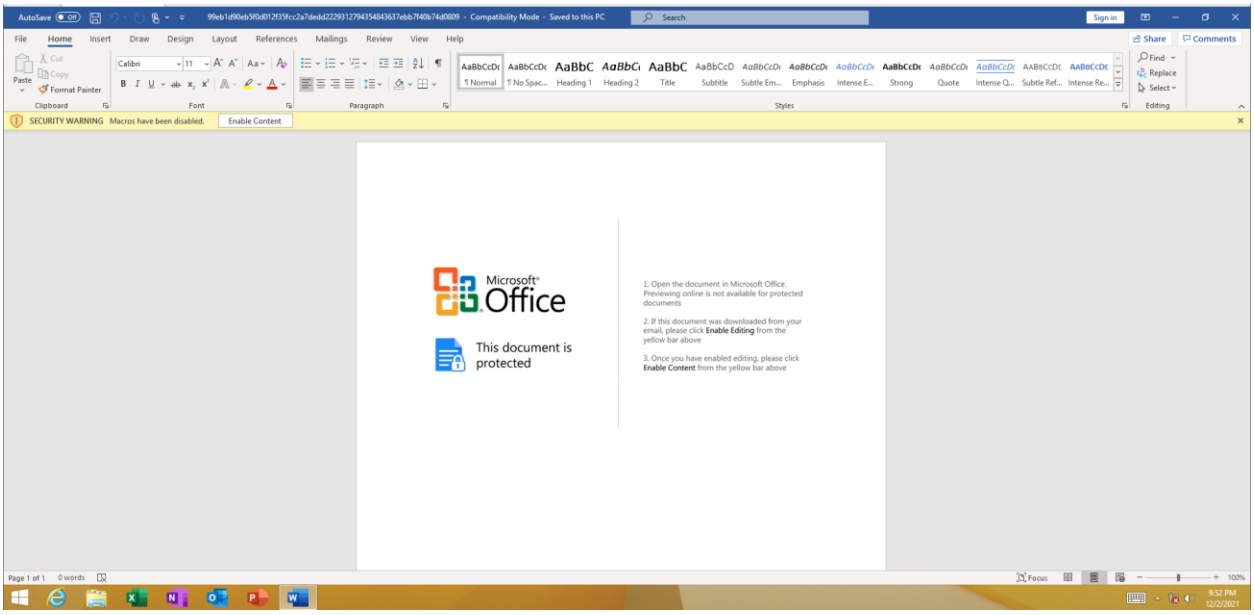
Through the analysis of the strings, we found that there is an image in the document, possibly edited with photoshop. This image could be used by the attacker to get the victim to execute the macros in the document that was obfuscated and cause the victim to be infected by the malware after enabling the execution of the macros.

There is not enough information of how the malware works with static analysis and basic dynamic analysis will need to be conducted to determine the purpose of the malware that is in the document.

4.2 Basic Dynamic Analysis

Static Analysis was able to provide us with some information of the malicious document. The document contains some malware but with the macros heavily obfuscated, we are unable to really see what the malware is trying to do the victim. There is an image in the document and through static analysis we are unable to see the contents of it. We will need to open the document to find out what the image is and run other tools alongside it to examine the malicious document when it is executed. This refers to the processes in the background, changes to the registry keys and the domains that it may be contacting like we have seen in VirusTotal. To analyse all this, we will conducting basic dynamic analysis on the malicious document.

4.2.1 General Analysis of malicious document



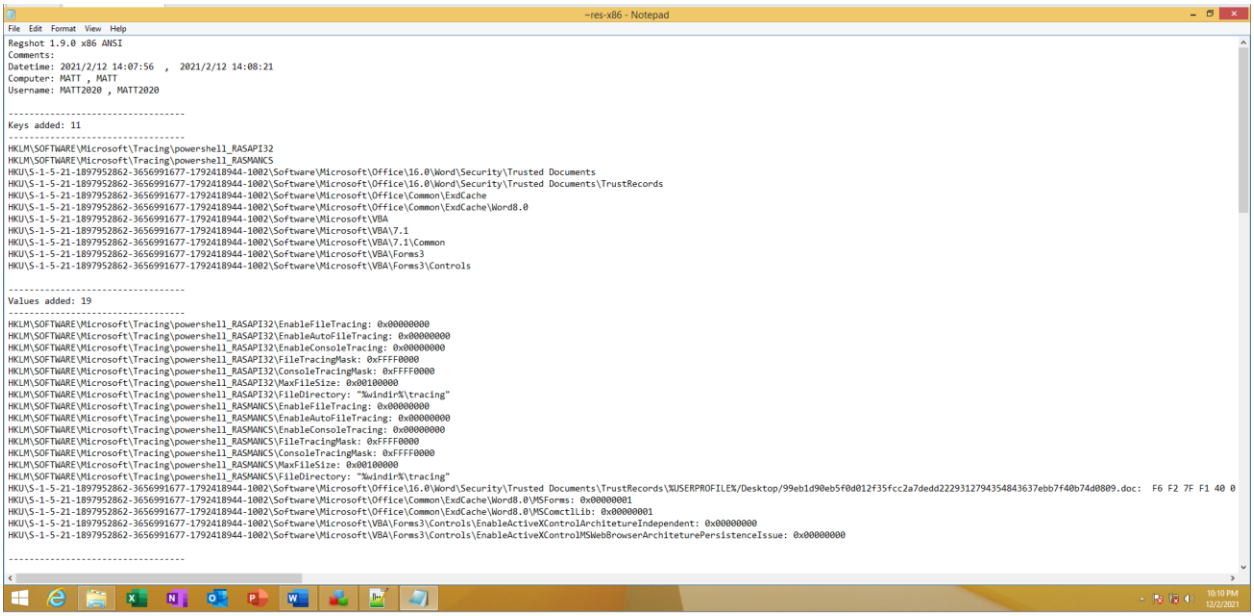
Upon opening the document with Microsoft Word, there is indeed an image in the document which proves what we have analysed in BinText through the metadata of the image. The image seems to be used for social engineering as it tells the victim to click enable editing from the yellow bar above, something we usually see when we download a MS word document online or from an attachment which is how malicious document usually spreads. Furthermore, the third step is asking the victim to click enable content on the yellow bar which is shown in the screenshot, which is a red flag as this will allow the macro embedded in the malicious document to run. Once the macro is executed, the malware will be able to infect the victim machine.

After clicking on enabling content by the following the instructions and waiting for a few minutes however, there seems no noticeable damage done to the victim machine even after enabling the macros. Since after enabling the macros did not allow us to see what the malware was doing, dynamic analysis tools will be used to discover about the malware.

4.2.2 Registry Analysis

Since there are no visible changes to the machine, registry analysis is conducted to examine if by opening the document and enabling content following the image, what the macro embedded in the file may be doing the victim machine since during static

analysis, we are unable to find out what the macros due to it being heavily obfuscated and the functions used made it difficult for analysis.

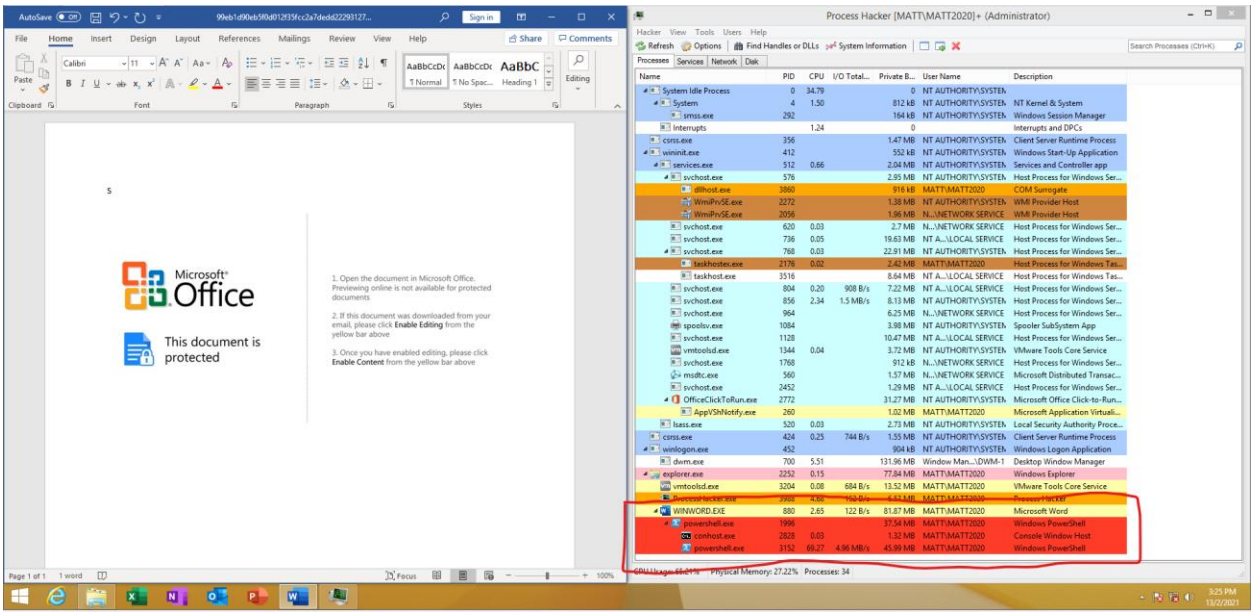


As shown with regshot, there are keys added pertaining to ‘HKLM\SOFTWARE\Microsoft\Tracing\powershell_RASAPI32’ and ‘powershell_RASMCS’. This means the malware tried to establish an external connection using powershell, possibly to the domains in the VirusTotal graph above and to the server. It can also be used to download the malware by establishing a connection to the IP address that was shown in VirusTotal.

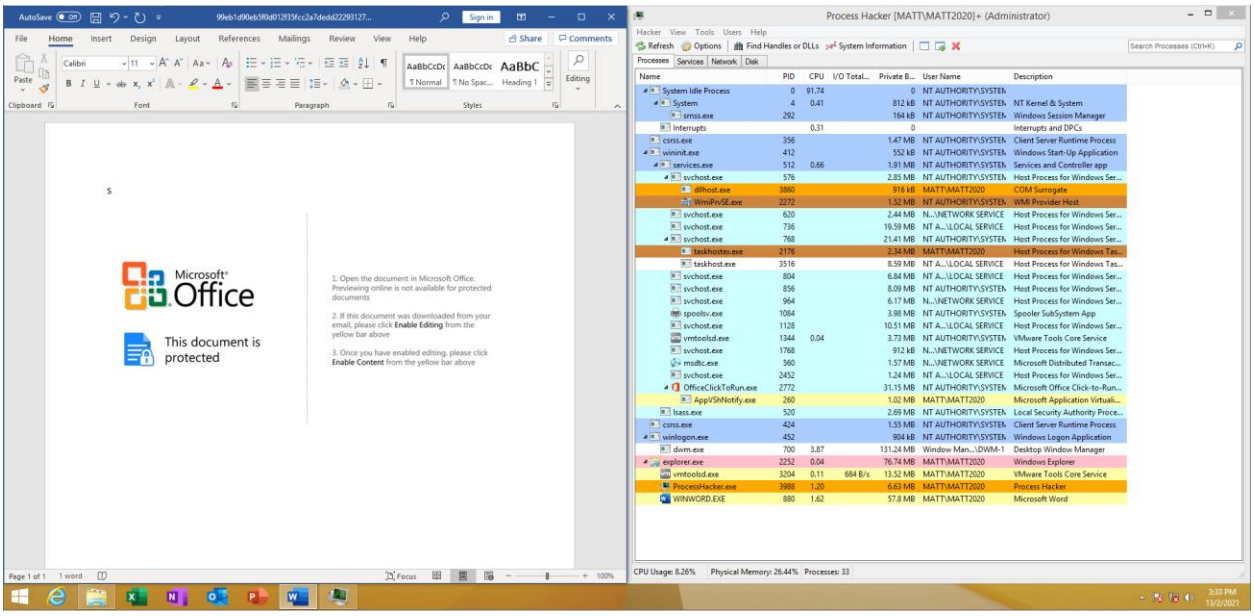
4.2.3 Process Analysis

Since from Regshot there has been some keys added pertaining to RAS (Remote Access Service, just like the malware in the sample report. This could be crucial since we will have some changes to the registry keys relating to Powershell, indicating that powershell may be running in the background after enabling content in MS word.

To do this, we will using Process Hacker to monitor the processes running in the background.

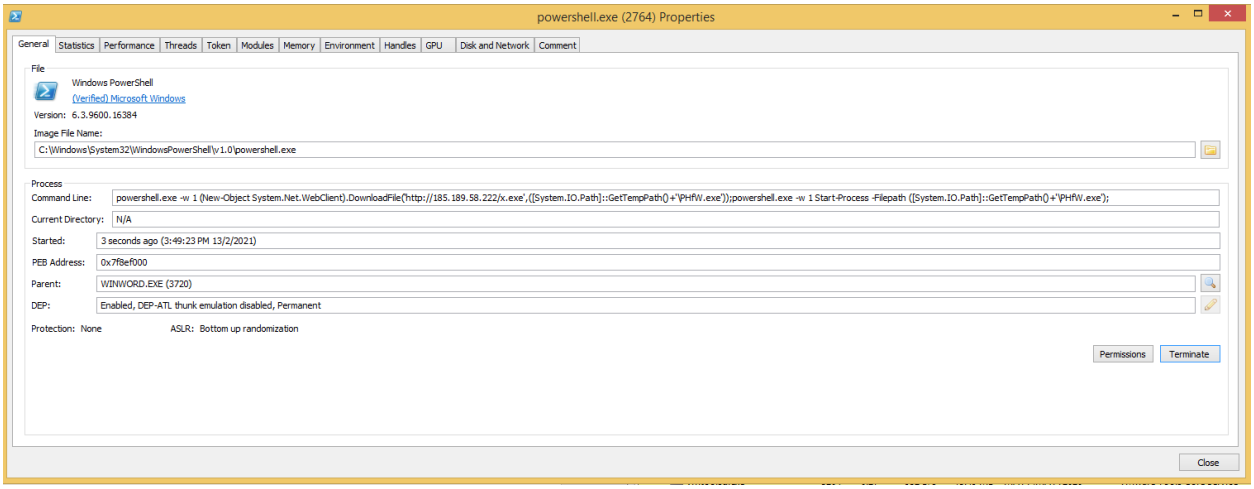


As expected after running the macros, 2 processes of powershell was actually launched which concurs with what was found in registry analysis with the keys pertaining to powershell being added. However, the process only lasts for a few seconds and gets removed from Process Hacker as shown below.

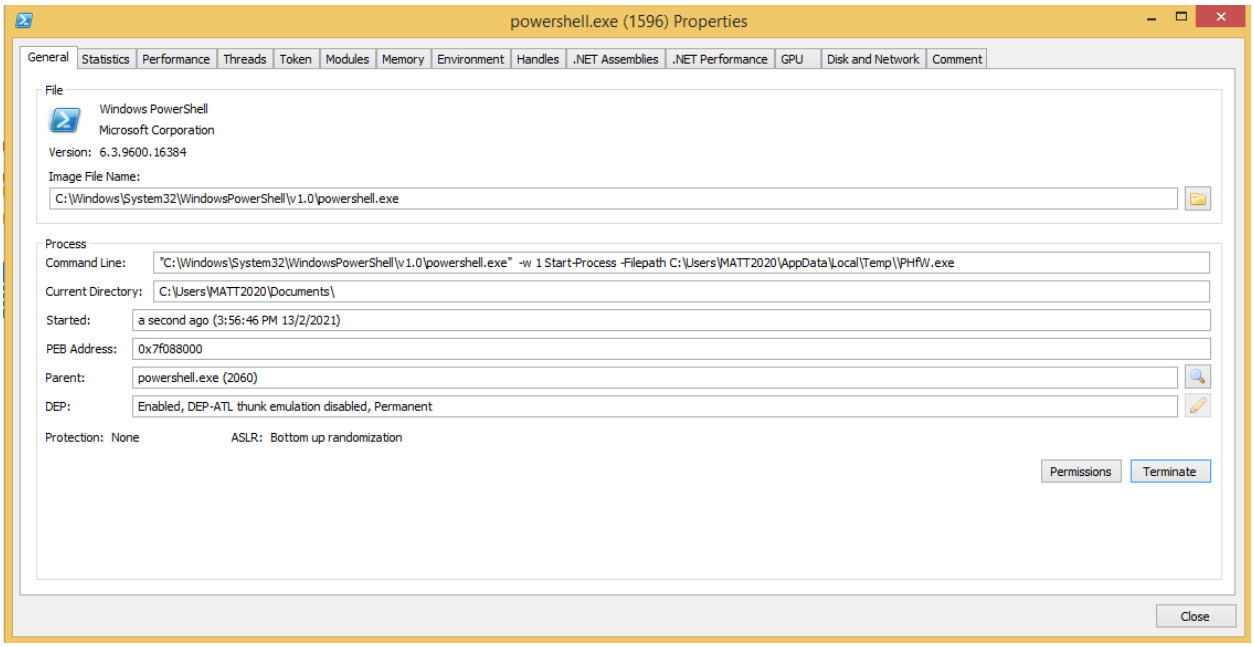


However I was able to click on the powershell.exe processes before they were killed, which allowed me to see what the macros was trying to achieve.

For the first powershell process. The malware runs powershell to download the file from “http://185.189.58.222/x.exe”. This is highly likely to be the malware and corresponds with the IP address that was discovered by VirusTotal. It downloads it as a temp file to avoid the victim detection. It stores the malware as PHfw.exe in a temp directory. For the second powershell command, the malware will start go to the same directory to find the executable file that it downloaded, before running the executable file that it has downloaded.

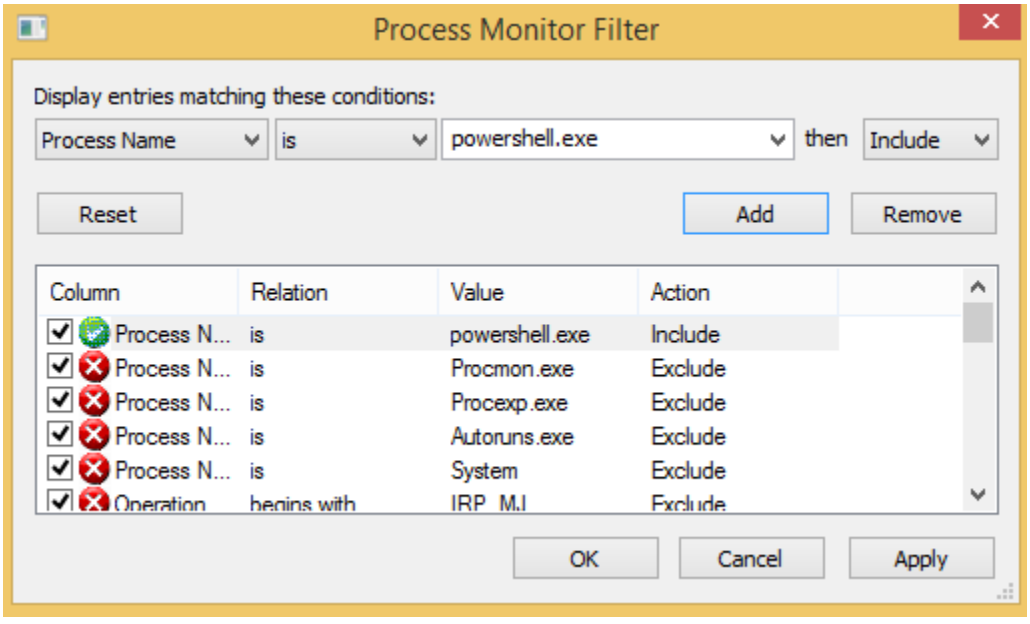


Looking at the second powershell process, the directory where the file is located is shown clearly it starts the process of the file that is located in that directory, which is highly likely the malware that was downloaded from the URL.

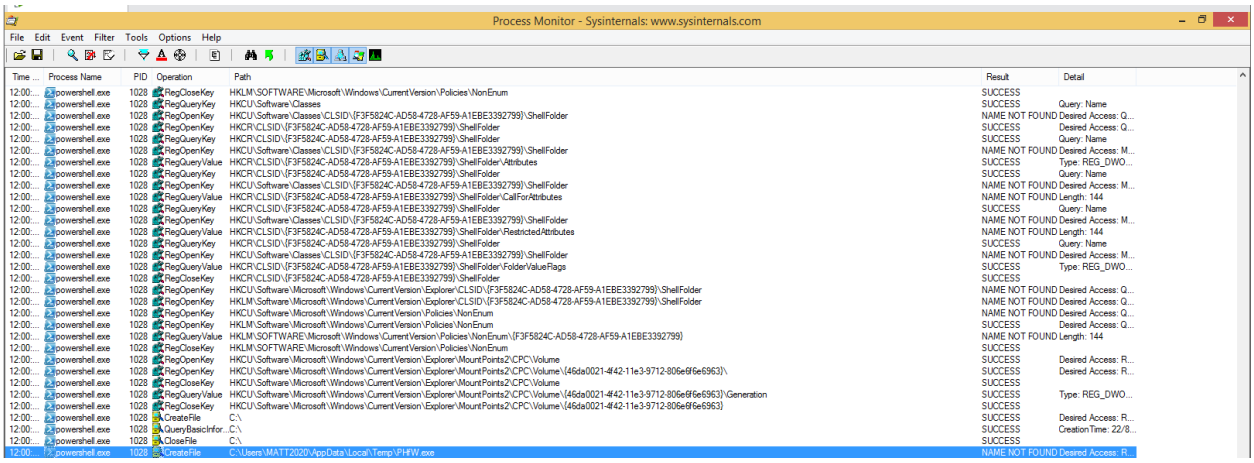


4.2.4 Monitoring running processes using procmon

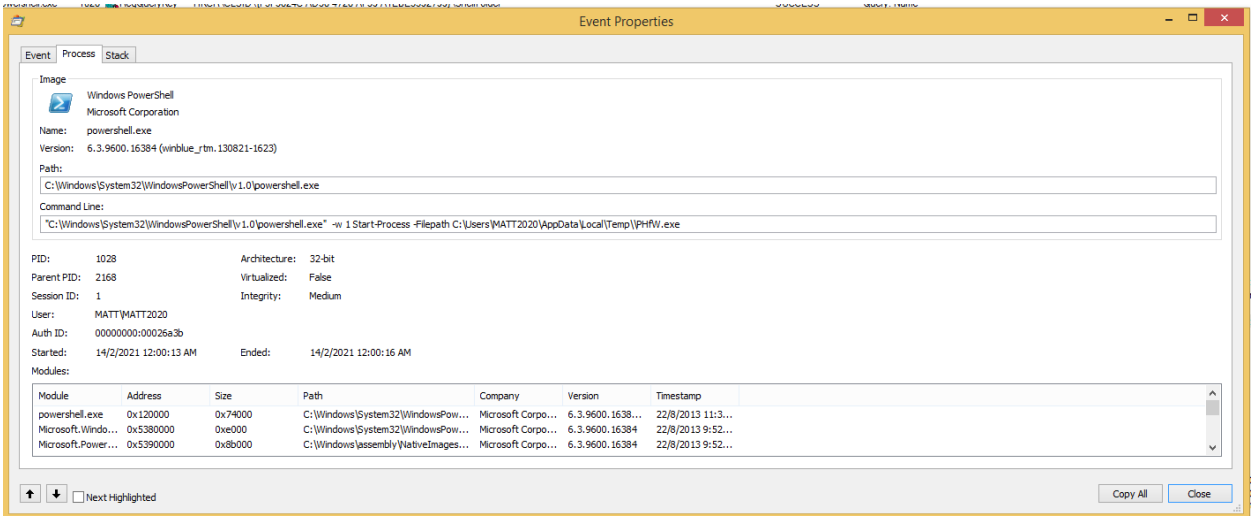
Since we know from Process Hacker that powershell.exe is launched, we should use procmon to filter the powershell processes so that we are able to analyse why the malware decided to use powershell.



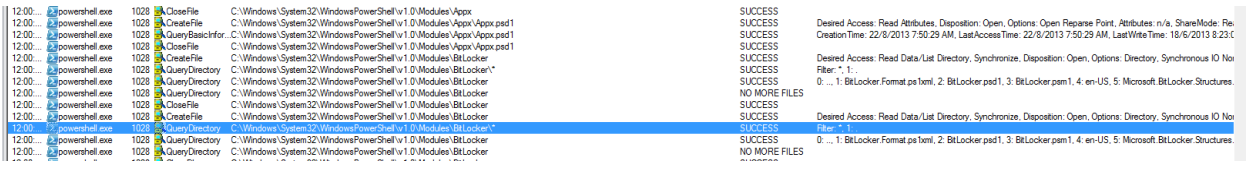
After only including powershell processes, I managed to find some interesting processes, such as when the the malware tries to find the Phfw.exe file that it downloaded in a Temp Directory, which proves what was discovered on Process Hacker Correct. However since the malware did not manage to download itself and store it into the directory, the outcome was not found.



Looking at the properties of the process, we are able to find the same command that we found in Process Hacker

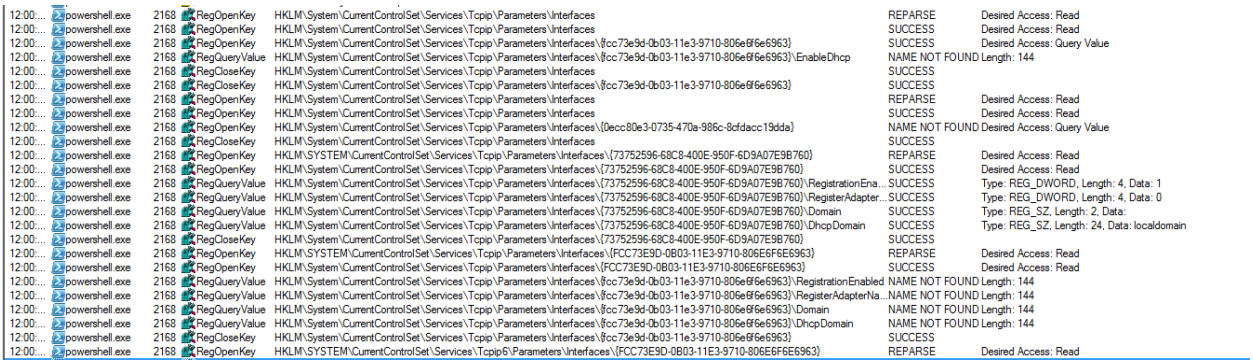


There were some references to BitLocker as well, which means that the malware could be checking whether certain disks were encrypted.

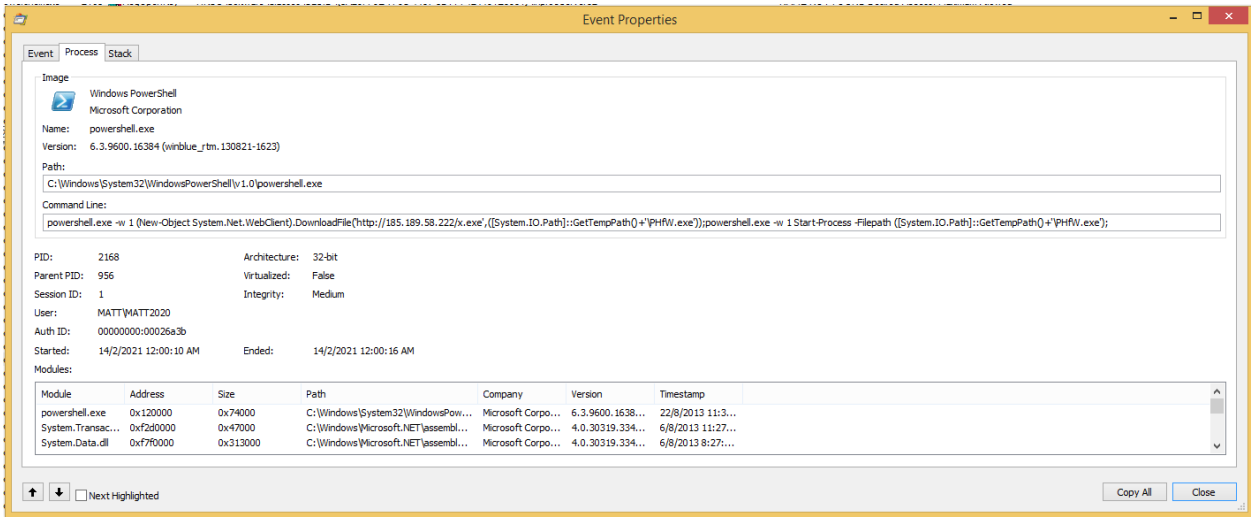


There were references to Winsock2, which is a network based indicator and indicates that the malware perhaps trying to establish a connection through the internet.

There were references to tcpip parameters, which indicated that the malware was trying to read the network configuration of the device

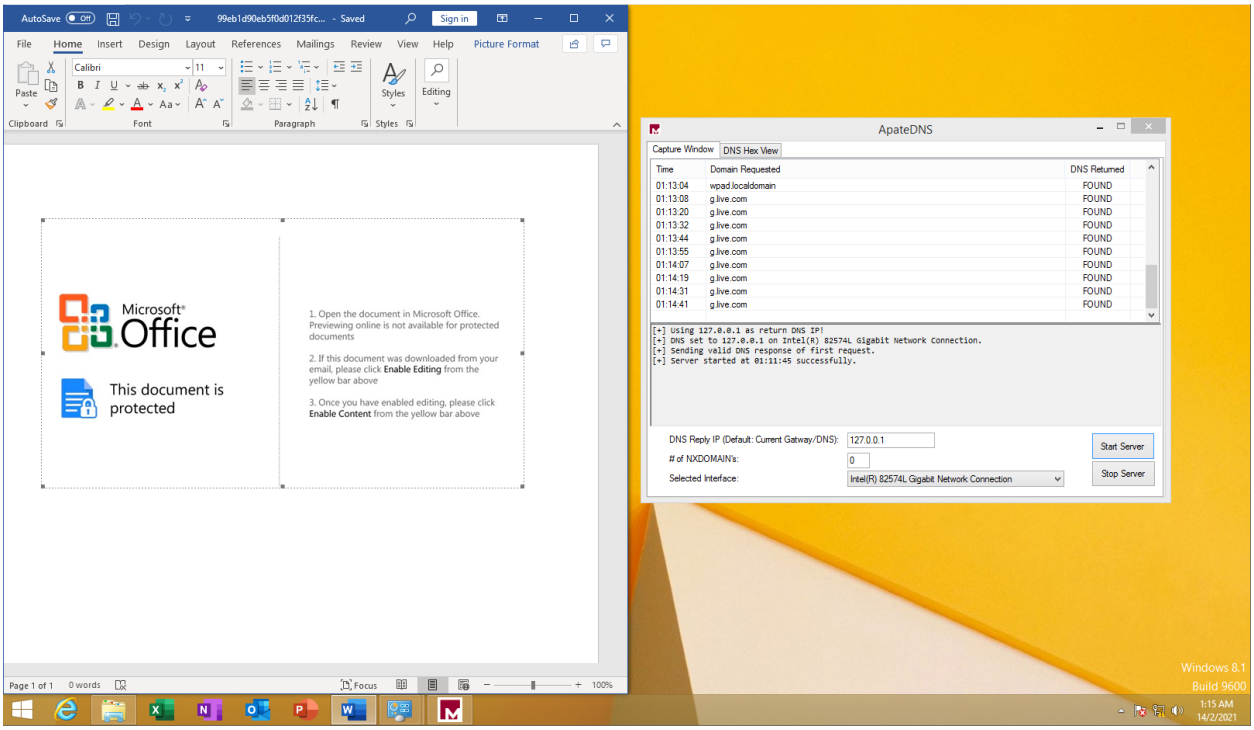


By clicking on the process, it shows that the process was the same as the one captured on Process hacker with the same commands being shown.



4.2.5 Network Analysis using ApateDNS

We can analyse the network traffic of the victim machine to see if any external connections were made by the malware apart from the powershell command that was seen earlier where downloads an exe for the IP address that was seen in VirusTotal.



After enabling content like what was shown on the image and allowing the macro to run, no suspicious external connections were shown on ApateDNS, which means that the malware only runs powershell to try to get the exe file from the IP address 185.189.58.222 and executes the file after it has been downloaded into the Temp Directory.

4.2.6 Summary of Dynamic Analysis

Although there was no visible damage done to the machine even after the macros was run, the use of tools in dynamic analysis gave us a clearer view of what the malware was doing. After seeing the registry changes relating to powershell, we were able to monitor the process and find out the command that the malware was trying to run and its intention which is going to the IP address that was shown in VirusTotal and installing the file in the Temp directory so as to avoid detection by the victim. The malware then runs the exe file that it has downloaded from the directory using powershell. Since the processes only last for a few seconds, the victim will not have noticed that powershell was running to install the malware into the victim machine.

4.3 General Analysis

We will be summarising what we found about the malicious document through static and dynamic analysis, and discussing about what type of malware it is, along with what functionalities it has.

4.3.1 Type of malicious document

After analysing the malicious document, it is shown that is a Microsoft Word document dropper. The image in the malicious document, like most malicious documents, uses social engineering to get the victim to execute the macros. It claims that the document is protected and to view its contents, the victim needs to enable editing and enable content in Microsoft Word. Once the victim clicks enable content, the macro will be executed. Upon execution of the macro, the malware will open powershell to download itself from “http://185.189.58.222/x.exe” into a temp directory and executing the file once it has been downloaded.

4.3.2 Execution of malicious document

As with most malicious documents, the malicious document works by getting the victim to downloading it. One example is through email attachments, which is the reason why the second step was shown to the victims. It requires social engineering for the victim to execute the macros and start the download of the malware from the IP address it contacts.

4.3.3 Functionalities of malicious document

Just downloading the document is harmless. Once the document is opened, if the victim follows the instructions in the image, the macro will be executed. This leads to the 2 powershell commands seen in Process Hacker and Process Monitor being run, which is to download the malware from the IP address and store it as PHfw.exe in the Temp directory. The second command will determine the path where the malware is downloaded and start another powershell process to execute the file.

4.3.4 Malicious document defenses

Since it is a malicious document, the first form of defense is to append itself into the malicious document as a macro so that the victim will not suspect that it is actually malicious and just a regular word document. Upon opening the document, it will use social engineering to convince the victim that it is protected and enabling content will show its contents, but in actual fact it will cause the malware to download itself and execute itself. Since all this happens in a matter of a few seconds and the executable file that the malware downloads will be in a temp directory, the victim will not realise that the malware is already in the machine.

4.3.5 Malicious Document Removal

The easiest way to remove the malicious document is to prevent it from executing. This can be done through antivirus such as McAfee Antivirus or Malwarebytes. It is important to check the document on whether is there a need to enable content, as documents that have macros tend to be malicious.

The malware can be removed manually easily since it is unable to download itself into the victim machine. Although the malware was not able to download itself into the victim machine, this does not mean that the malicious document does not need to be removed as there may be other processes that the malware may be running.

To remove the malicious document, delete the malicious document ” 99eb1d90eb5f0d012f35fcc2a7dedd222931279435484363 7ebb7f40b74d0809.doc”.

Go to registry editor and delete the two keys “HKLM\SOFTWARE\Microsoft\Tracing\powershell_RASAPI32” and “HKLM\SOFTWARE\Microsoft\Tracing\powershell_RASMCS”

If the malware managed to install itself, go to the directory “C:\Users\<Username>\AppData\Local\Temp” and check if the executable PHfw.exe is present. If the executable is present, delete the executable.

These steps should remove the malware and all of its traces, but antivirus is strongly recommended to protect against malware, since anti-malware is able to prevent the download of the malware, hence the malware will not get a chance to execute itself.

Youtube Video Links

Introduction: https://www.youtube.com/watch?v=uPqdefGEO80

Basic Static Analysis: https://youtu.be/-a4scFqSZQs

Basic Dynamic Analysis: https://youtu.be/LJfyPii6cbo

Malicious Document: https://www.youtube.com/watch?v=b3WCgOy-SX8

Conclusion: https://www.youtube.com/watch?v=napPmYjYi-M