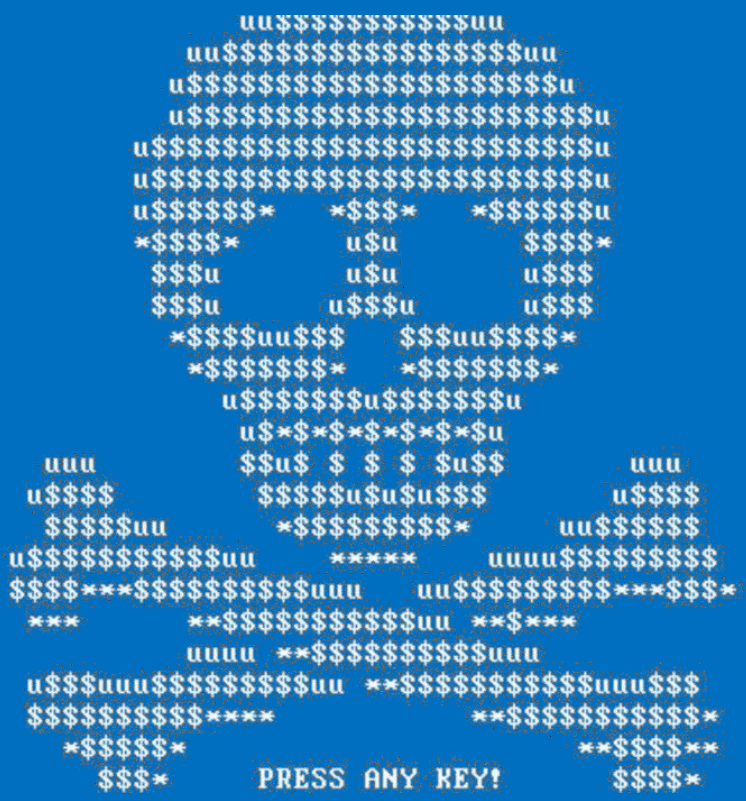

Wannacry ? Nah Dearcry !

Malware Analysis Case Study

Dakshin Tharusha
Sri Lanka Institute of Information Technology
Malabe, Sri Lanka

TABLE OF CONTENTS

1 IDENTFYING DIFFERENT MALWARE TYPES

1.1 VIRUSES

Viruses, as opposed to worms, need an infected and running operating system or application in order to function. Word documents and executable files are the most common targets for viruses. The majority of individuals are undoubtedly aware that downloading an.exe file from anywhere other than a reputable source may result in problems. Extensible files may have a wide variety of file extensions.

When a virus enters a computer, it remains dormant until the infected host file or program is run. This may happen through malicious websites, file sharing, or email attachment downloads. By the time it replicates, the infection has spread throughout your computer system.

1.2 BOTNET

A bot is a hacker-controlled computer that has been compromised with malware. After then, the bot (also known as a zombie computer) may be used to launch further assaults or join a bot army (aka a botnet).

As a hacker show-off (the more bots you gather, the stronger you are) and ransomware propagator, botnets are quite popular. As they expand unnoticed, botnets may encompass tens of millions of devices.

1.3 TROJAN

A Trojan Horse, as the name suggests, is a harmful software that masquerades as a harmless file. Users download it because it seems to be reliable, and then they discover that it turns out to be the enemy when it storms.

1.4 RANSOMWARE

Ransomware obstructs or deletes your own data in exchange for a fee. You're then asked to make a payment (often in crypto currencies) before you're let back in. In just one day in May of last year, a ransomware assault infected almost 200k machines in 150 countries. According to reports, the ransomware assault, dubbed WannaCry, inflicted damage estimated at between $100 million and $1 billion. A vulnerability in Microsoft Operating Systems that had not yet been patched was exploited by WannaCry.

1.5 ADWARE AND SCAMS

One of the most well-known forms of malware is adware. It bombards you with intrusive pop-ups and banner advertisements that have no bearing on your browsing experience. To get free software, some consumers will put up with specific kinds of adware (games for example). Adware, on the other hand, is not created equal. It's obnoxious at best, and it slows down your

computer. The worst-case scenario is that the advertisements direct visitors to sites where harmful downloads are waiting for them. In addition to delivering spyware, adware is often vulnerable to hacking, making machines that have it installed a prime target for cybercriminals including hackers, phishers, and scammers

1.6 SPAM AND PHISHING

Phishing is more of a kind of social engineering assault than a malicious software attack. Phishing is both. This type of cyber assault, however, is very prevalent. Phishing is effective because the emails sent, text messages exchanged, and web links generated seem to be from trustworthy sources. Phishing is successful because of this. In order to commit identity theft and other crimes, thieves use them to get sensitive personal and financial information. Some of them are very clever and are capable of deceiving even your most knowledgeable customers. Especially if the email account of a well-known contact has been hacked and it seems that you are receiving an order from your boss or colleagues in the IT department.

2 OBFUSCATION TECHNIQUES

2.1 DEAD CODE INSERTION

Dead code insertion is a simple method that adds unnecessary activities to a program, such NOP instructions that don't alter its behavior. Antivirus may detect and destroy NOP operations, thus Other dead code may be added to tweak the program, such incrementing and decrementing a variable. The dead code may never be run, or if executed, has no impact on the malware's original functioning.

2.2 REGISTER REASSIGNMENT

The method of renaming or reassigning a register from one virus generation to another. The program's behavior and functionality remain unchanged. It is used to reassign a register that is never utilized in the application.

2.3 SUBROUTINE PERMUTATION

Subroutine Permutation randomly shuffles the malware routine's order. This method may generate possible malware combinations using n subroutines. Each sample's subroutine order may vary.

2.4 INSTRUCTION SUBSTITUTION

Instruction Substitution replaces code with comparable operations that provide the same effects but are performed differently. To identify this obfuscation method, a dictionary of all potential replacements for each operation is required.

2.5 CODE TRANSPOSITION

Code transposition alters the sequence of the instructions in the malware. Non-dependent code blocks are reorganized to alter the malware's code without affecting its behavior. It is difficult to identify separate chunks of code. Alternately, you may reorder the instructions and reassemble the code using conditional and unconditional jumps.

3 INTRODUCTION TO DEARCRY RANSOMWARE

Microsoft exchange server is one of the leading email servers around the globe which is used to manage emails of organization that runs on windows operating systems. In December 2020 zero-day vulnerability was found related to these exchange servers followed by three other severe vulnerabilities (CVE-2021-26857, CVE-2021-26858, CVE-2021-27065), According to the tech giants' large amount of abnormal data transmission from those Microsoft exchange servers to suspicious IP addresses were initially noticed and after investigating the logs it had been revealed that a set of inbound POST requests were associated with resources utilized by outlook web access service such as images, fonts, CSS files and JS files. Even though it was suspected that those exchange servers were exploited using backdoors deep forensics investigation performed based on disk and memory analysis exposed that the incident happened due to new server-side request forgery attack, and it was later called as proxy logon vulnerability (CVE-2021-26855). That vulnerability allowed adversaries to impersonate as system admins and bypass the authentication processes which resulted to execute malicious arbitrary commands via port 443. Microsoft released an initial security update on 2nd or March 2021 and recommended every customer to apply the patches as soon as possible that caused to spotlight the zero-day attack and got the attention of the adversaries.

Due to this reason active exploitation has been detected on workplaces where the Microsoft exchange servers did not get updated to the latest security patch by the time. Although harmful cyberattacks were initially took place mostly in the United States, malicious events quickly spread throughout the globe, including to the European Union, by a growing number of hacker organizations. Analysis conducted by the group of researchers proved that there were around 250000 vulnerable systems soon after the releasement of security updates but with the organization which had strong security postures rushed to update their systems and that caused to reduction of the vulnerable servers in a considerable amount within short period of time.

As usual adversaries leveraged this vulnerability to perform post exploitations in their own way. After the successful exploitation like obtaining essential information, those adversaries placed a new ransomware which caused to encrypt all the important resources of the victim and make them completely inaccessible by the legitimate parties to obtain more financial benefits from the compromised victims. Victims who were affected by this new ransomware identified in countries like Indonesia, India the United States, and some parts of European Union. This new ransomware called itself as Dearcry also known as DoejoCrypt ransomware that left a read me note in a text file asking to contact two email addresses in order to get the decryption instructions after encrypting the files on the compromised system.

3.1 STEP BY STEP APPROACH TAKEN BY ADVERSARIES TO DEPLOY THE DEARCRY RANSOMWARE.

1. The adversary sends a malicious HTTP request to the vulnerable server on port 443 to initiate an untrusted connection with the server.

2. Use the Proxy Logon (CVE-2021-26855) vulnerability to carry out server request forgery SSRF attacks.

3. As the next step authenticate through Microsoft Exchange Server.

4. Utilize CVE-2021-26858 and CVE-2021-27065 vulnerabilities to construct malicious request and write arbitrary files on the system and other attacks.

5. Then use the CVE-2021-26857 vulnerability which exploited unsecured deserialization flaw in the unified messaging service which allowed to obtain highest privilege on the vulnerable server to execute arbitrary codes as the system user by constructing malicious requests.

6. Create malicious files which enables write files to any path on the compromised server.

7. Promote authority to execute any command.

8. Inject malicious Web Shell scripts.

9. Run the Web Shell script to upload the ransomware DearCry to the compromised Microsoft Exchange Server as a payload.

10. Run the ransomware DearCry to encrypt files in compromised Exchange Server.

4 WHAT IS BASIC STATIC ANALYSIS

As the initial step all security experts begin the malware analysis process with basic static analysis which comprise the steps that are taken to figure out the characteristics of the malware by observing its overview without actually running it in any environment. There are several common approaches like using different anti-virus programs to obtain the results to determine the maliciousness of the suspicious file, using hash values to check whether it belongs to a known malware or not and reading the strings that are contained within the suspicious file to observe hidden notes and malicious URLs, checking whether there are any obfuscation techniques involved and so on.

5 GETTING STARTED WITH BASIC STATIC ANALYSIS

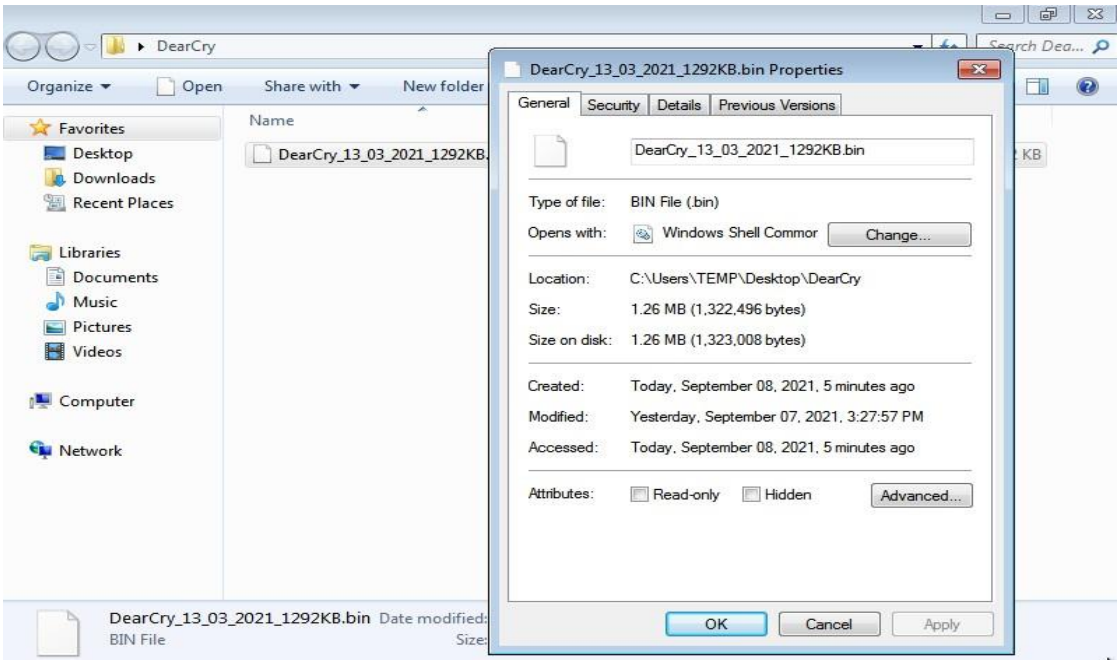

Figure1

As the first step the hash value of the suspected file was generated and to obtain the hash values HashCalc software was utilized as shown in the following figure.

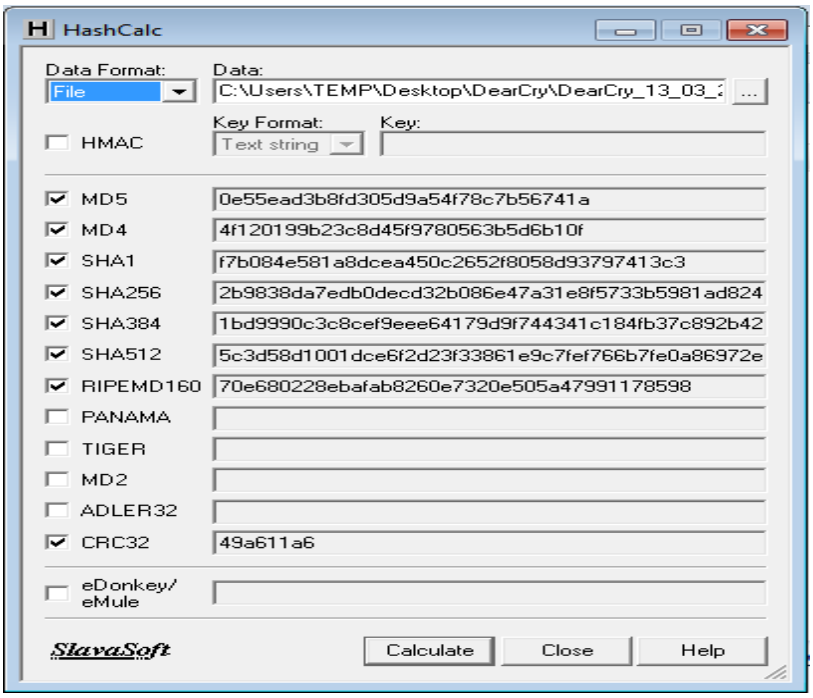

Figure2

Then the generated MD5 hash value was submitted to the virus total instead of sharing the original file to virus total since if it is shared there is a slight probability of getting accessed to that by the malware writers. Moreover it is always considered as the best practice to submit the hash by the security experts.

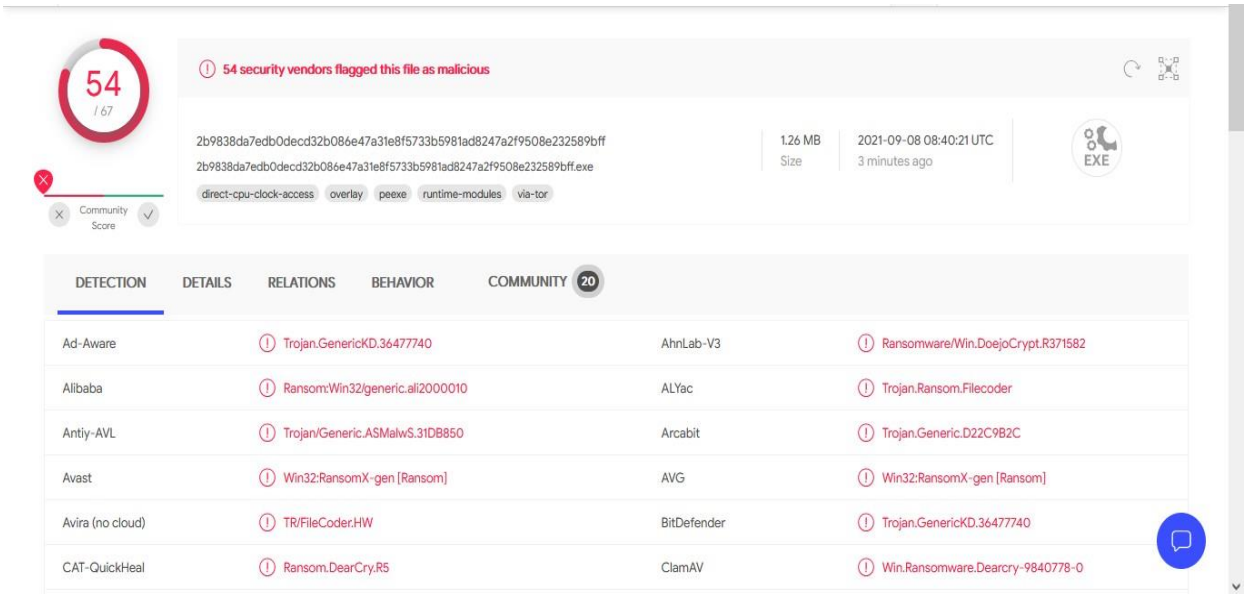

Figure3

NOTE: when you submit particular hash value to check whether the file is malicious or not you should always reanalyze the content to get the latest updates of the security vendors because with the time security vendors update their threat databases, file signatures and machine learning algorithms.

As shown in the above figure it was flagged as malicious file by 54 security vendors. Also the results were differed from each other as some of them identified that hash value related to a trojan whereas some vendors identified it as a ransomware. Moreover details tab provided some interesting information about the suspicious file hash. They are listed as follows,

Basic Properties

MD5	0e55ead3b8fd305d9a54f78c7b56741a
SHA-1	f7b084e581a8dcea450c2652f8058d93797413c3
SHA-256	2b9838da7edb0decd32b086e47a31e8f5733b5981ad8247a2f9508e232589bff
Vhash	016056756d55555173z72z67nz35z67z
Authentihash	a19ed23411cd99c0427370728563d3dc309130c8d75b124f569155bfb928fa18
Imphash	f8b8e20e844ccd50a8eb73c2fca3626d
Rich PE header hash	be78f27ebdeabcb0f8dec8f6b28e0a6b
SSDEEP	24576:LU5NX2yJOiUXmElCxu2WAP0NIzkQM+KpPRQ9StlUDpl1fpxkHVZgMCS+:L7XP7P9o5QzUtl1fpxkHVZgMC3
TLSH	T11B55CFC3FAC248F2D846097951B3973B5F3ADA10832AC5C3DFA159A59C21AD1673E3C9
File type	Win32 EXE
Magic	PE32 executable for MS Windows (console) Intel 80386 32-bit
TrID	Win32 Executable MS Visual C++ (generic) (48.8%)
TrID	Win64 Executable (generic) (16.4%)
TrID	Win32 Dynamic Link Library (generic) (10.2%)
TrID	Win16 NE executable (generic) (7.8%)
TrID	Win32 Executable (generic) (7%)
File size	1.26 MB (1322496 bytes)

Figure4

Following image depicts that this file was initially created on 09th of March in 2021 and other names have been seen in the world time to time.

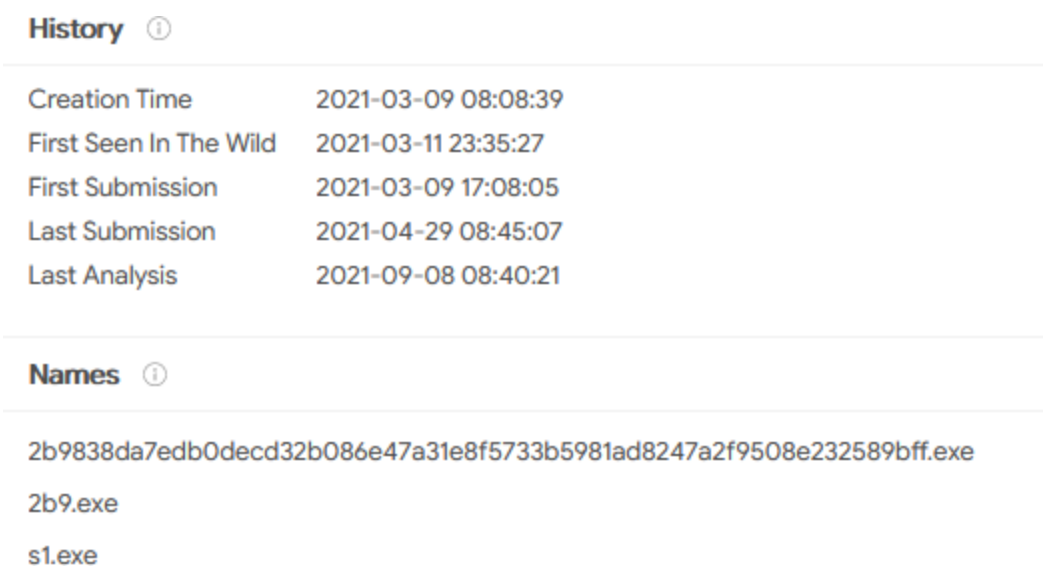

Figure5

More importantly this file was identified as an executable file even though file extension was bin as shown in the figure 1.

Apart from that observing community tab you can get more clues which would be beneficial for the analysis as it provisions the individual reports of various researchers related to the suspicious file that the submitted hash belongs to, and it can contain essential clues for us to use in our own analysis.

Then that malicious file is analyzed with Exeinfo PE tool to check whether it is packed or not. Surprisingly the results proved that it was not packed as expected and had no defensive mechanism against detection though it is very common approach to pack or obfuscate malwares to make them stealthy or in other words make them undetectable. This turns out the meaning that the malware writer could be an amateur or this version could be a testing prototype released intentionally to observe the impact.

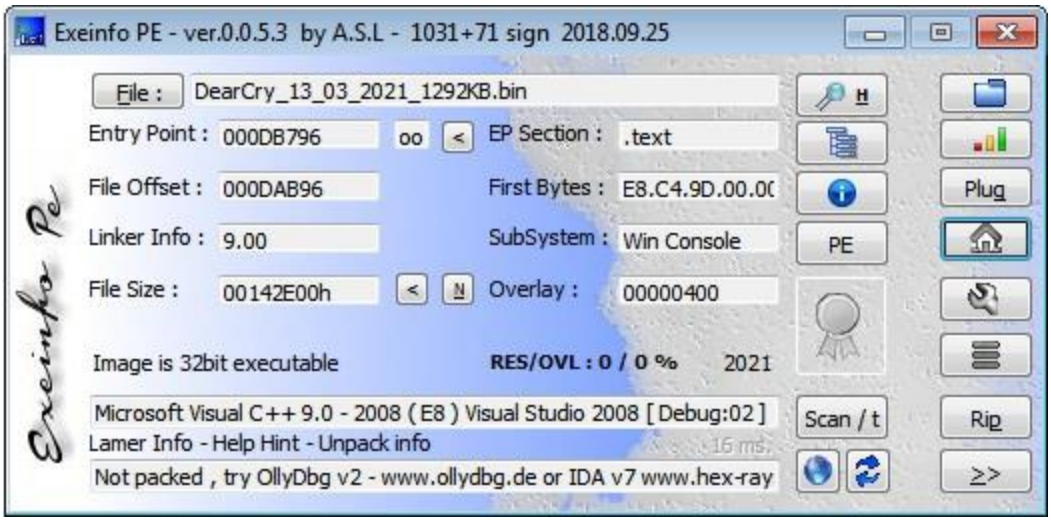

Figure6

Since we were able to discover it was not a packed malware then we analyzed it using a hex editor called HxD to determine the actual file type since by analyzing file signatures we can determine the correct file type even though the file extension was modified to mislead by the malware writers.

As it is shown below from the hex dump of this file, there are several indicators that this file is a Portable Executable (PE). Portable Executable is a file format which includes executables, object code, DLLs, etc. which are used in Windows Operating Systems.

- The first 2 bytes of the file are 4D and 5A decoded as MZ. This is a clear indicator of a PE Header.
- We can see the text “This program cannot be run in DOS mode” this is another indicator.

These indicators again proved that what we are dealing with is a file of the PE file format and it is an executable even though the sample file itself has the .bin extension.

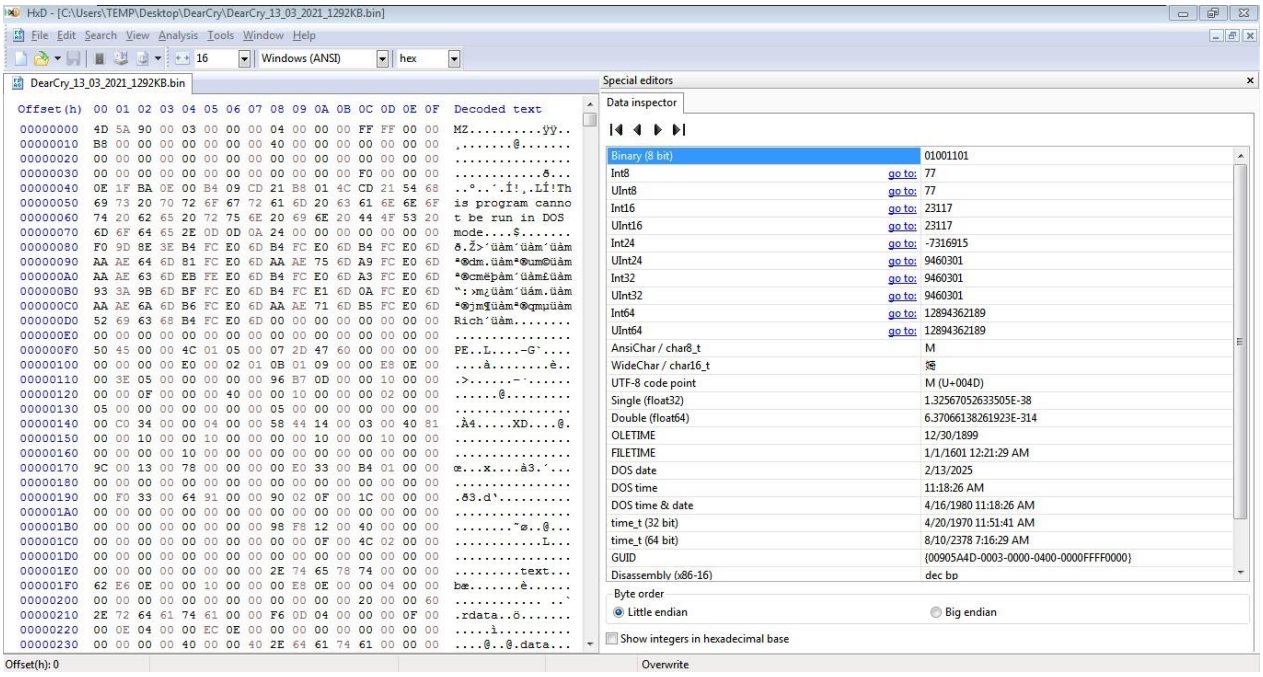

Figure7

Then we utilized pestudio to further analyze and obtain more details about the malware. As we can see the file-type is identified as a 32-bit executable and under signature microsoft visual c++ 8 is identified as the compiler which was used to compile the source code. Additionally compiled time can be seen along with the date, which was previously identified through virus total report.

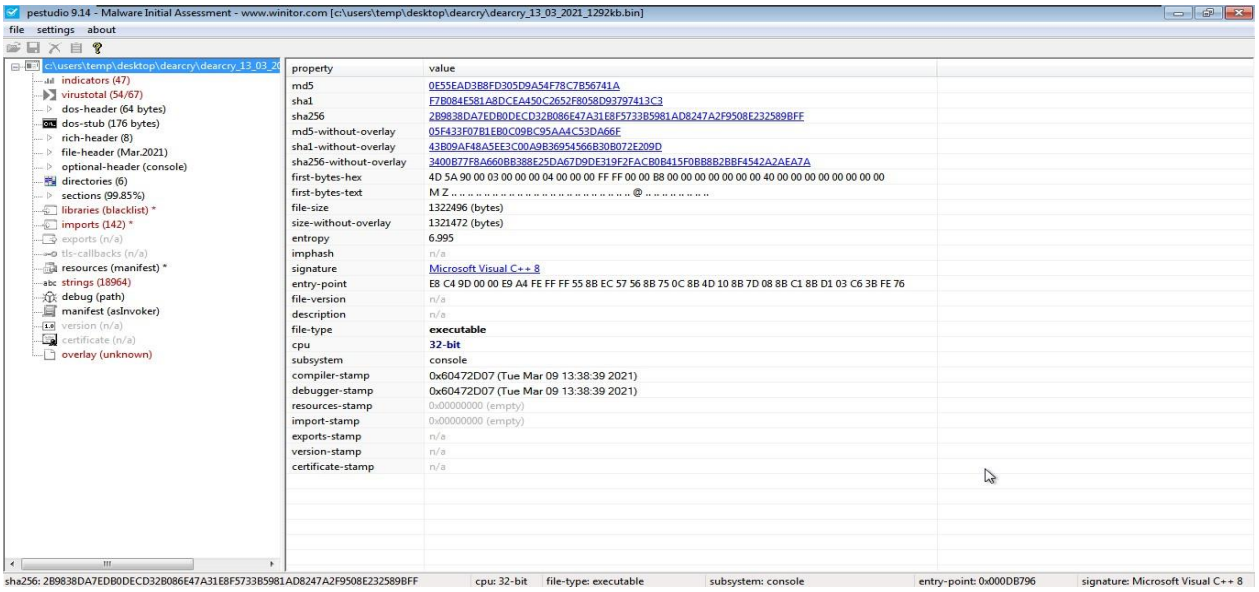

Figure8

Under the indicators tab another interesting artifact that has been found was the path of the PDB file present inside the binary, which shows the string “svc,” a common acronym often used for indicating the term service also the name “john” could probably be the computer name where the binary was compiled at the very first time by the malware author.

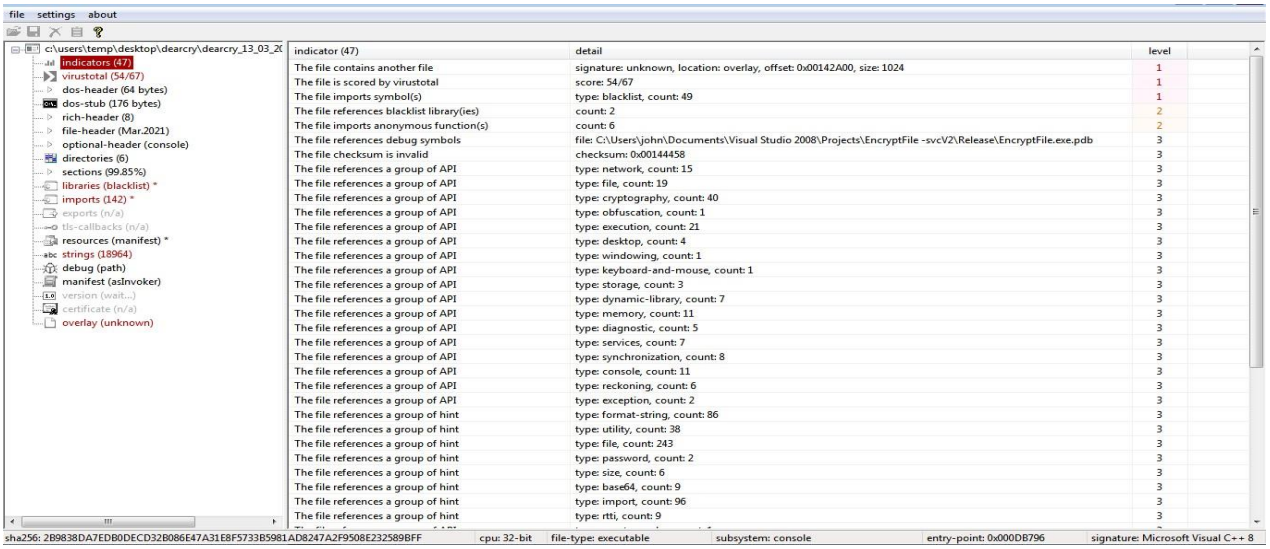

Figure9

Pestudio also has a built in virustotal tab which identifies detections of malicious indicators of the loaded file, and it was similar to the results which we identified in the first place.The Rich header is an undocumented component that can be found in most PE files created by Microsoft's LINK.EXE utility. This header is effectively present in any binary created with the standard Microsoft Visual Studio tool. Therefore following image turns out the meaning that the malware author had used that Microsoft Visual Studio 2008 tool to write this malicious code.

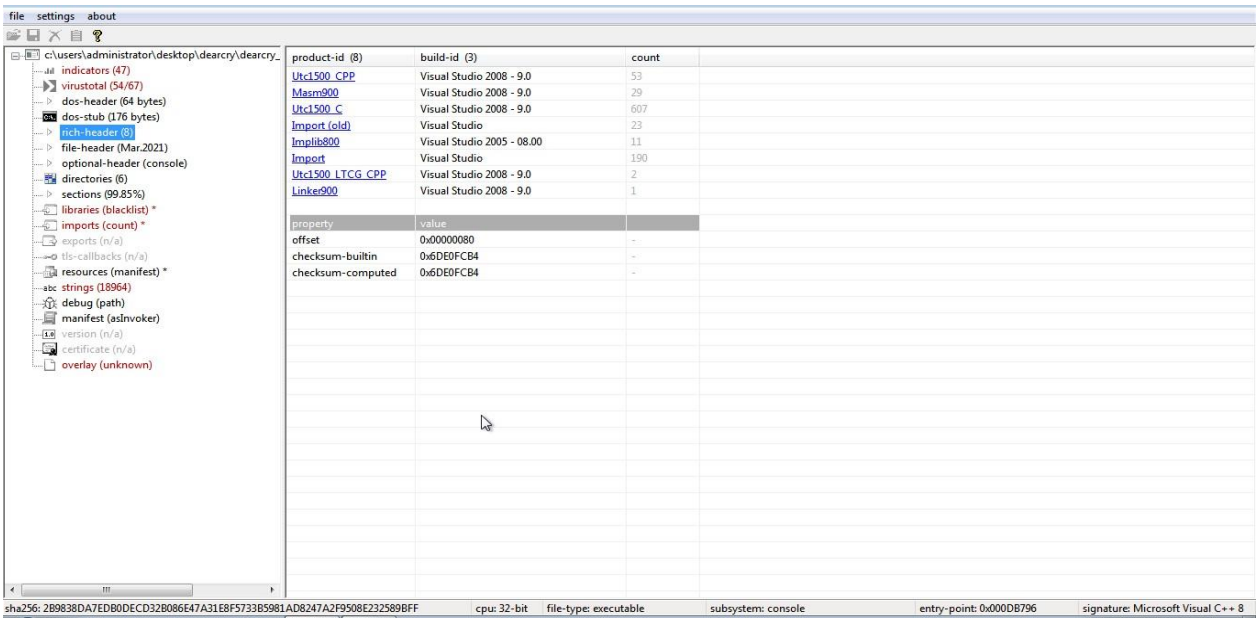

Figure10

Under the sections tab we can find out typical information related to this malicious file like the entropy values, byte size of each section and more importantly the entry point can be found and it is where the operating system or the loader passes the control to start the program execution.

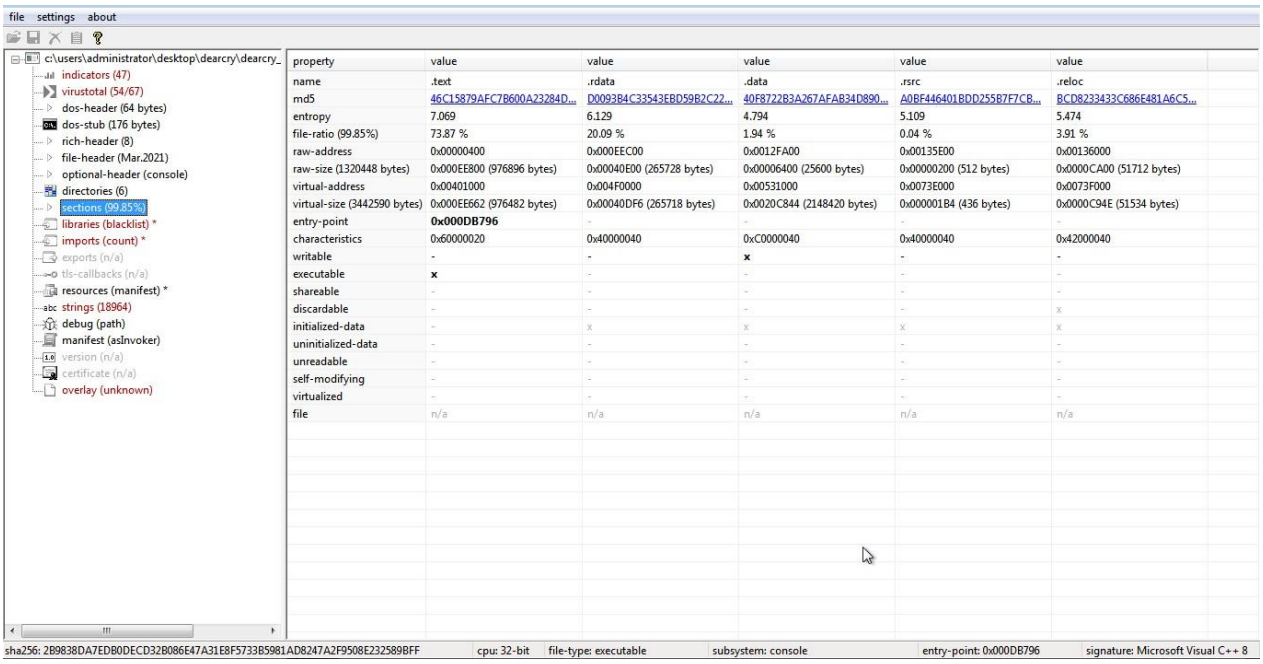

Figure11

Under the libraries tab we can find out four common libraries.

- Kernel32.dll - This is a fairly popular DLL that includes fundamental functions which allows to manipulate the hardware, files and memory of the system.
- Advapi32.dll – This gives the access to comparatively advanced core components of windows like registry editor and service manager.
- User32.dll – This is responsible for dealing with user interactions like responding and controlling user interfaces.
- Ws2_32.dll – Typically network related tasks are performed by this.

The interesting fact is pestudio itself blacklists this ws2_32.dll which turns out the meaning that it could cause to abnormal network activities in the process execution time.

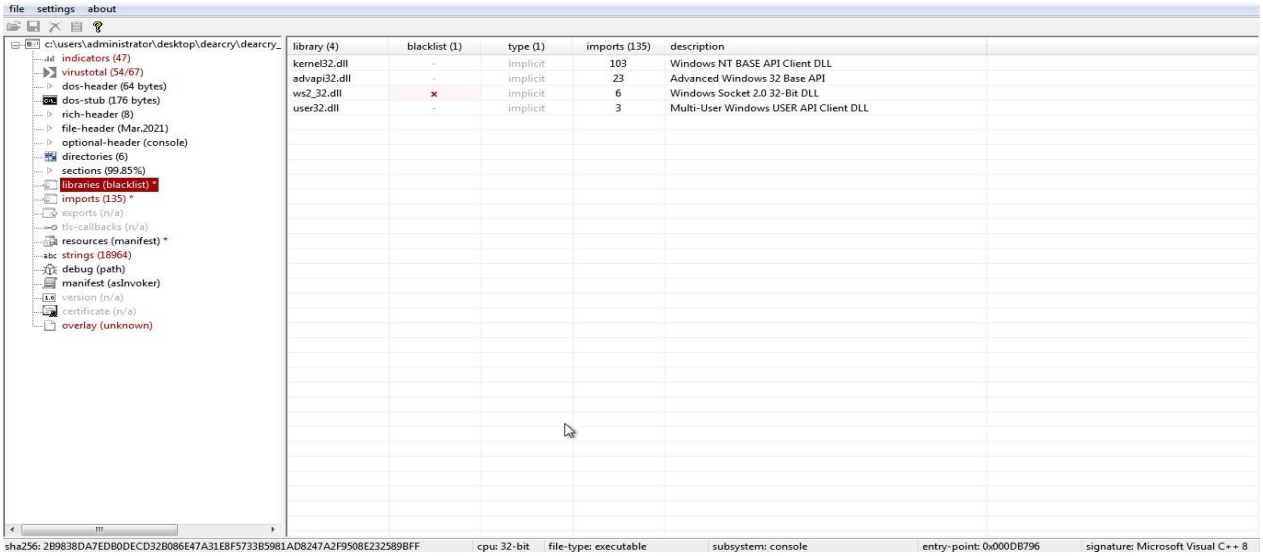

Figure12

The list of functions that an executable imports, is one of the most helpful pieces of information we can use to determine the hidden characteristics about it. Imports are functions that are utilized by one program but are really stored in another. These imports are connected to their programs by the programmers so that they do not have to re-implement certain functionality in numerous applications. Another important fact is they themselves are not malicious but can be utilized to execute malicious activities by linking them together to perform chain of tasks.

By observing imports tab we can find out 135 imports are used by this suspicious file and some of them are blacklisted like we have seen in the libraries tab. As we assumed in the previous step, here we can also find out some clues that this malware is going to create some malicious connections during the time of execution by looking at blacklisted functions' names since they have self-explanatory names.

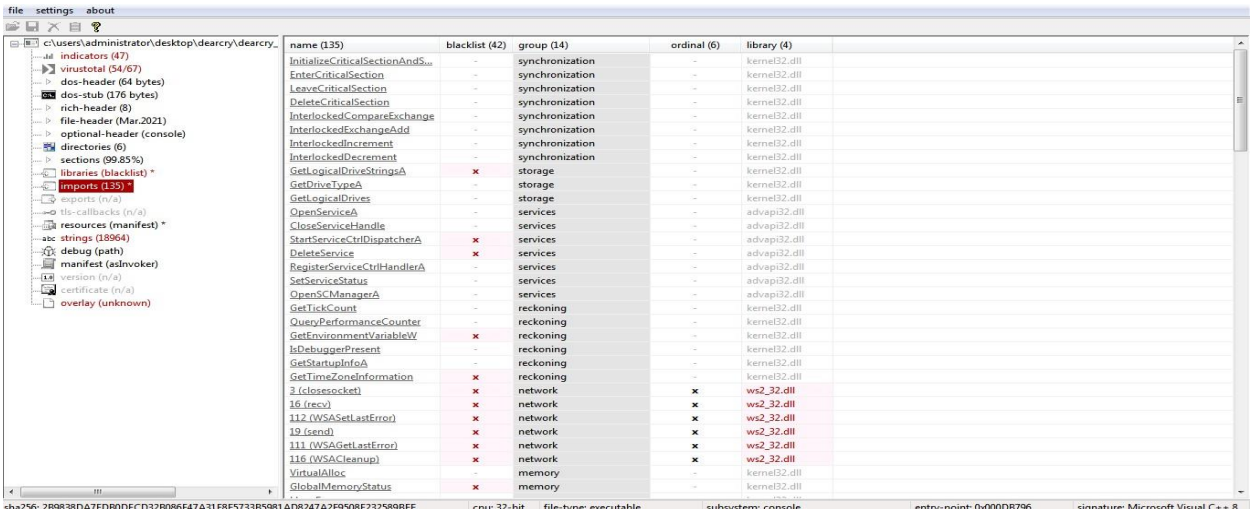

Figure13

Apart from that DeleteFileA, FindFirtsFileA, FindNextFileA imports proves that this is going to find some files from the system and delete them later.

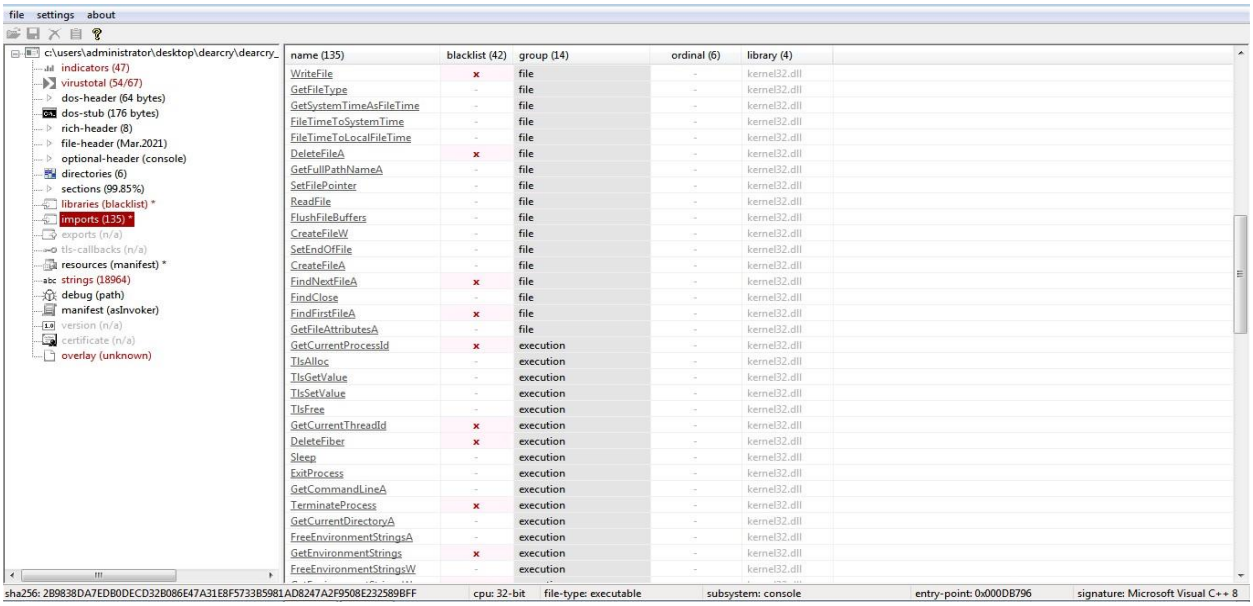

Figure14

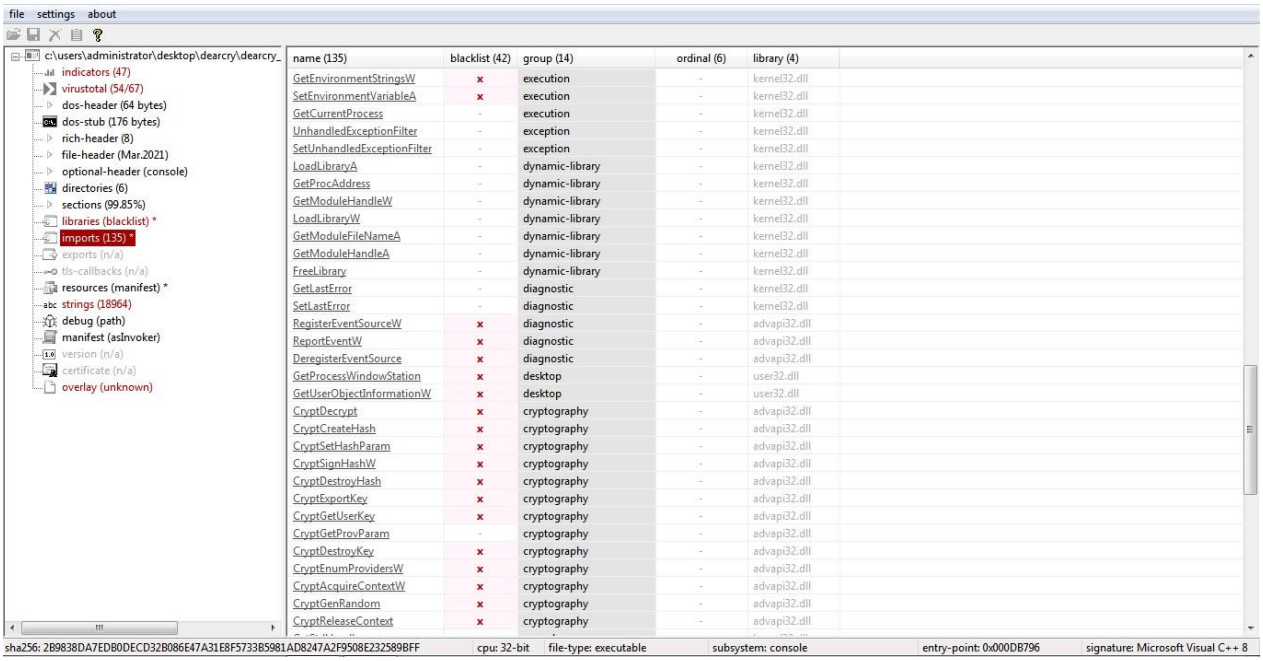

Figure15

Above image witnesses that this executable is going to use some cryptographic related functions and because of that we can assume that we are dealing with some kind of a ransomware.

String analysis is the investigation of readable text contained within malware. This can aid in the discovery of useful information. Therefore to retrieve the embedded strings we used a tool installed in the Flare VM called strings. Here -a represents the ASCII only which means it is going to filter ASCII text and the -n 6 represents the minimum string length.

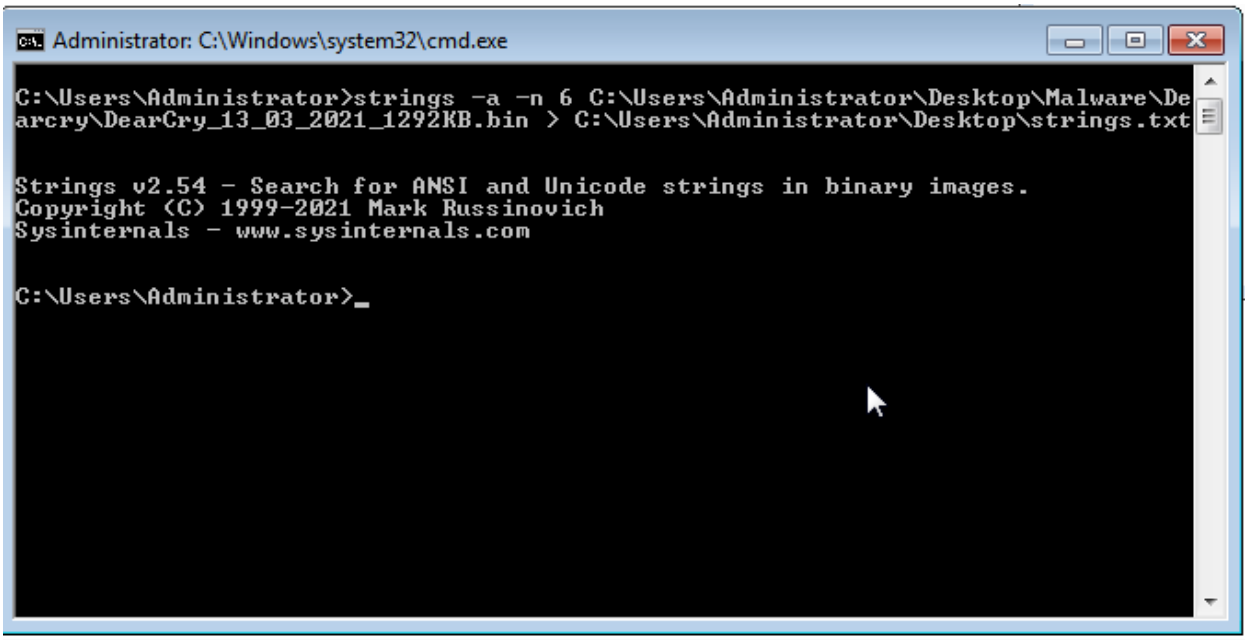

Figure 16

By observing the following text we can assume that this ransomware is going to encrypt the files using AES algorithm and it is considered as a symmetric encryption mechanism which uses the same key for encrypting and decrypting process.

```
cryptocom
cryptopro
whirlpool
dsa_with_SHA256
dsa_with_SHA224
hmacWithSHA512
hmacWithSHA384
hmacWithSHA256
hmacWithSHA224
hmacWithMD5
ecdsa-with-SHA512
ecdsa-with-SHA384
ecdsa-with-SHA256
ecdsa-with-SHA224
ecdsa-with-Specified
ecdsa-with-Recommended
id-aes256-wrap
```

Figure 17

Interestingly after observing more we were able to find out a hard coded public key of RSA and it is an asymmetric encryption algorithm which uses one key for the encryption purpose called considered as public key and another key for decryption process called as private key. Therefore we can assume that it uses both encryption algorithms for encryption purposes. Moreover, in order to make this work there is only one method and that is initially encrypt the content of the victim machine with symmetric algorithm first and then encrypt those content with the public key of the asymmetric algorithm. By performing the encryption process that way make it harder to decrypt using decryption tools.

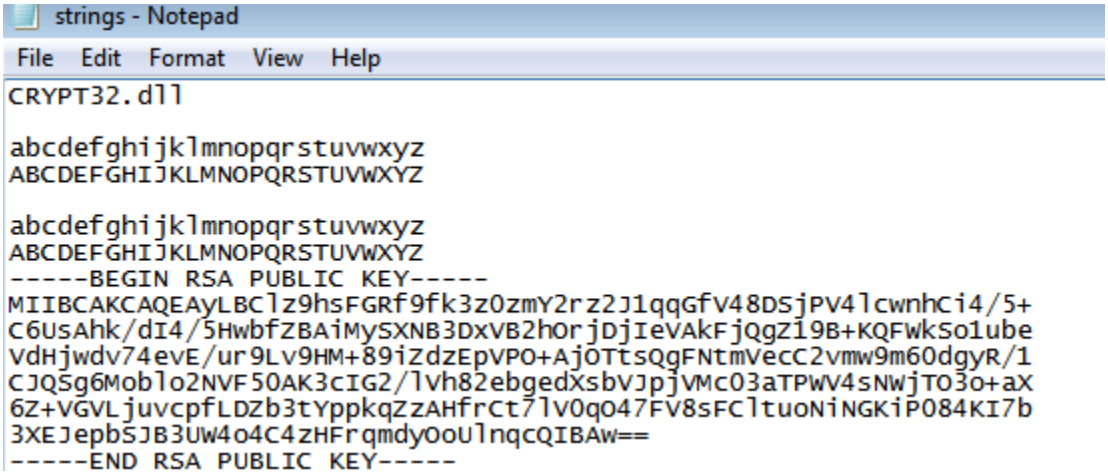

Figure 18

6 WHAT IS ADVANCED STATIC ANALYSIS

As explained previously basic static includes steps that helps to get basic understanding and overview of the suspicious file. On the other hand advanced static analysis phase is used to deep dive into the functionalities of the suspicious file to detect more evidence than preliminary once which helps to get the whole idea. Most of the times this is done utilizing tools called disassemblers. This phase involves inspecting the infected file's assembly code in terms of understanding the malware's operation and behavior. To perform this a deeper understanding of x86 assembly, the ability to recognize various C code structures in assembly and ability of identifying malicious functions is required.

7 GETTING STARTED WITH ADVANCED STATIC ANALYSIS

As the next step the malware sample was loaded into IDA tool in order to disassemble the executable and analyze the assembly code. In general, a disassembler is a tool that reconstructs the malware's assembly code from the machine instructions. This allows programmers to examine all the potential instructions that the malware may execute on the victim machine as assembly code

is human readable. Furthermore, a specific function inside the code which has a predefined implementation, or a specific digital signature can be identified.

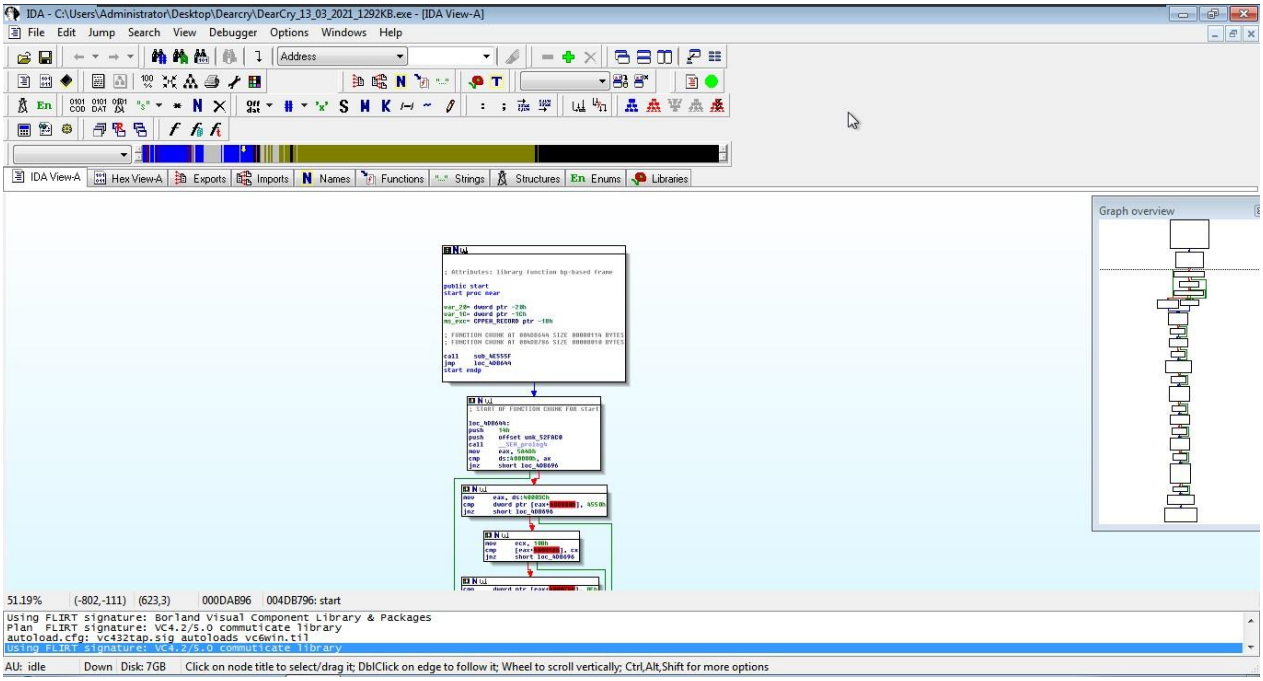

Figure 19

In the main function, it calls StartServiceDispatcherA and service in which the name called as “msupdate”. Therefore we can assume that it is trying to start a service. After that it calls an unknown function as it is shown in the below image.

```
; Attributes: bp-based frame fuzzy-sp

; int __cdecl main(int argc, const char **argv, const char **envp)
main proc near

ServiceStartTable= SERVICE_TABLE_ENTRYA ptr -10h
var_8= dword ptr -8
var_4= dword ptr -4
argc= dword ptr  8
argv= dword ptr  0Ch
envp= dword ptr  10h

push    ebp
mov     ebp, esp
and     esp, 0FFFFFFF8h
sub     esp, 10h
xor     eax, eax
mov     [esp+10h+var_8], eax
mov     [esp+10h+var_4], eax
lea     eax, [esp+10h+ServiceStartTable]
push    eax             ; lpServiceStartTable
mov     [esp+14h+ServiceStartTable.lpServiceName], offset ServiceName ; "msupdate"
mov     [esp+14h+ServiceStartTable.lpServiceProc], offset ServiceMain
call    ds:StartServiceCtrlDispatcherA
call    sub_401D10
xor     eax, eax
mov     esp, ebp
pop     ebp
retn
main endp
```

Figure 20

If we follow through along that function we can find out that suspicious email addresses and a ransome note which says that “Your file has been encrypted”. So that we can assume this function is responsible for the malicious things that this malware is intended to perfrom in the execution time. Furthemore it calls another function which is highlighted in the following image.

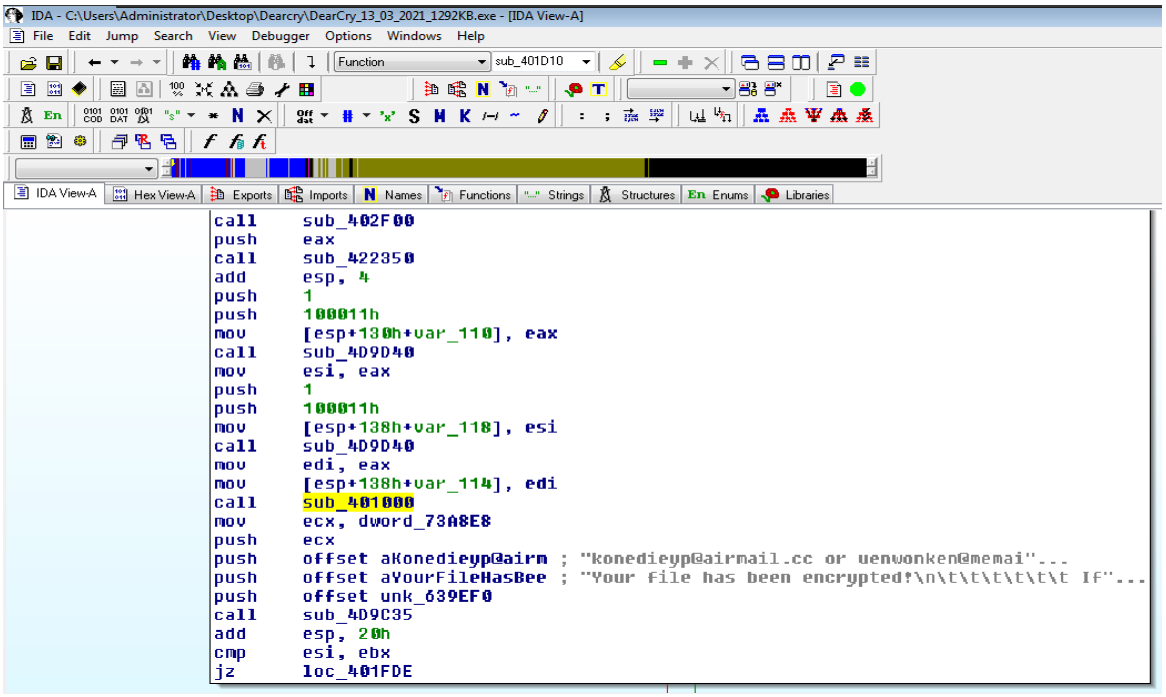

Figure 21

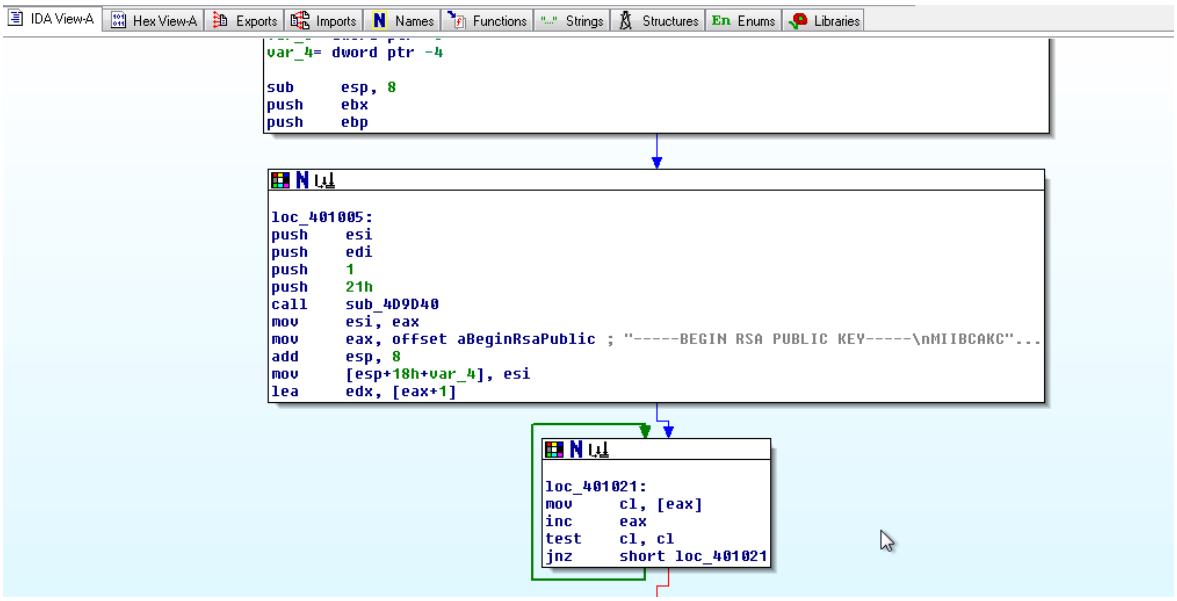

Figure 22

The second unknown function shown in the above image proves that it directly connected with the encryption process since it refers the hard coded public RSA key.

Then it obtains the bit masking representation of the logical drives of the infected machine.

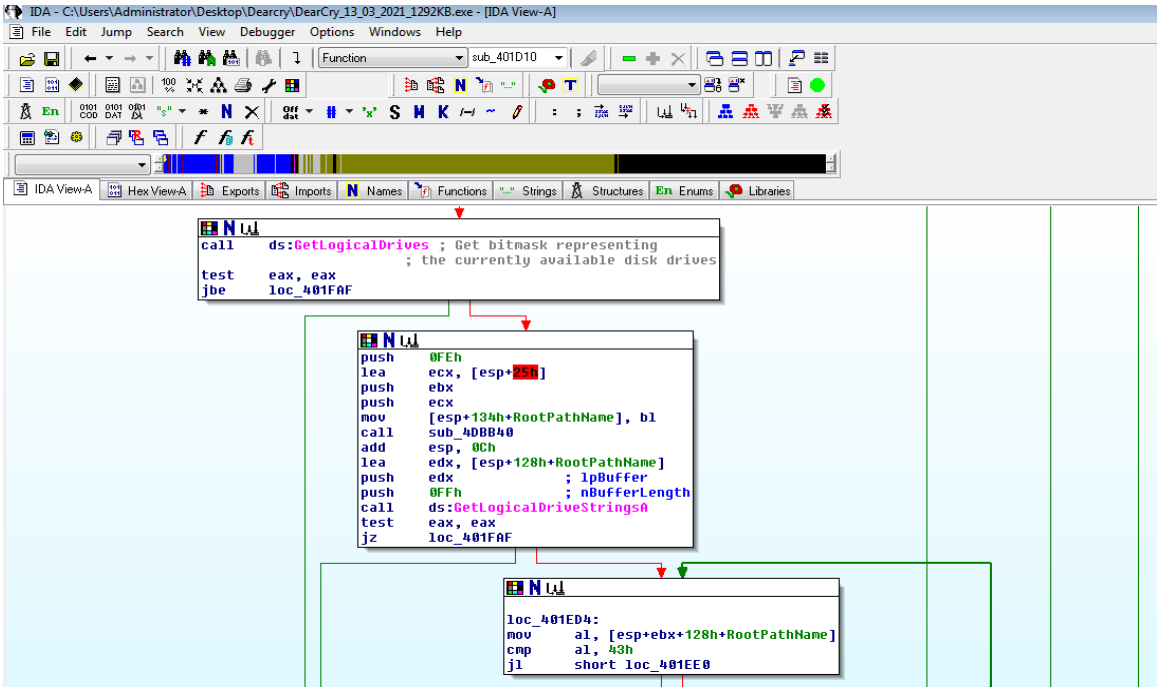

Figure 23

Then it starts finding the files of the system.

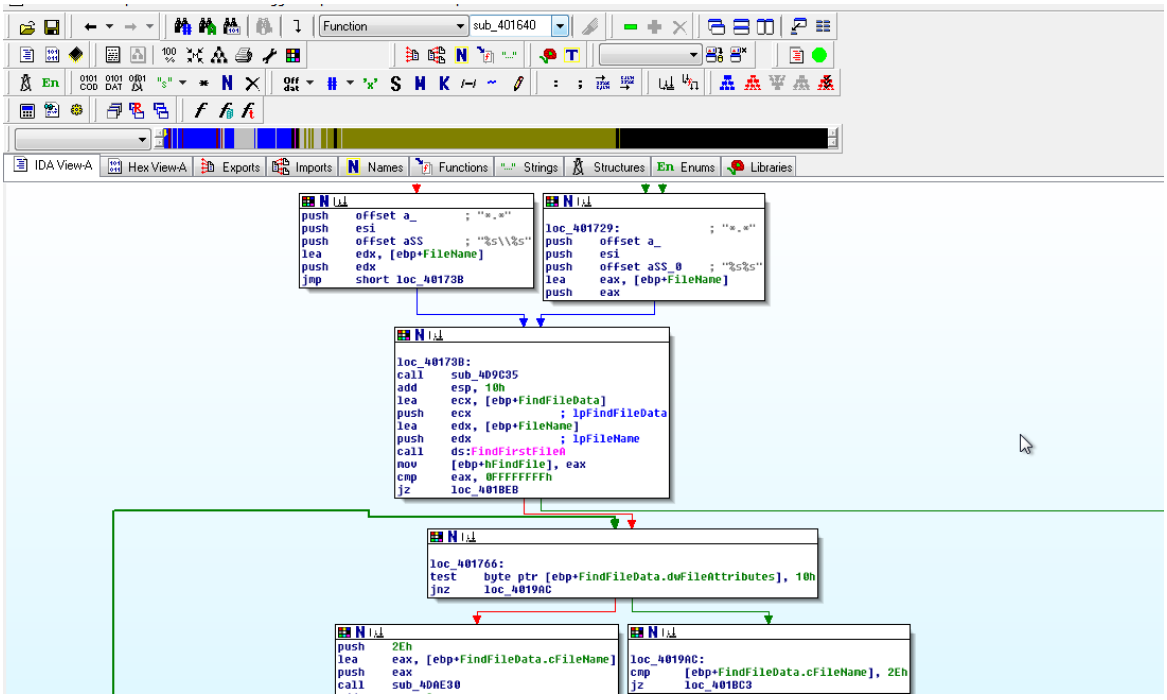

Figure 24

Under that function execution the list of file extensions also can be found out and we can assume that those are the file types of this ransomware tries to encrypt once it gets infected.

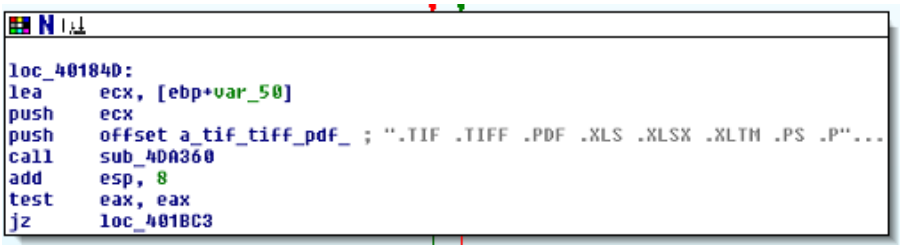

Figure 25

It continues like a loop and tries to find files available in the system.

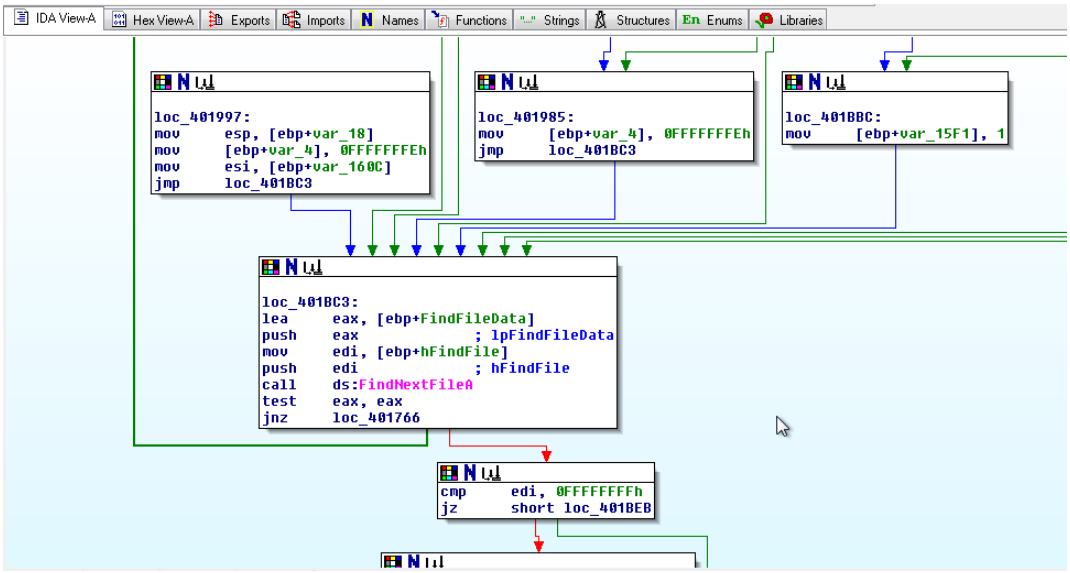

Figure 26

Then it creates a file called readme.txt as shown in the following image.

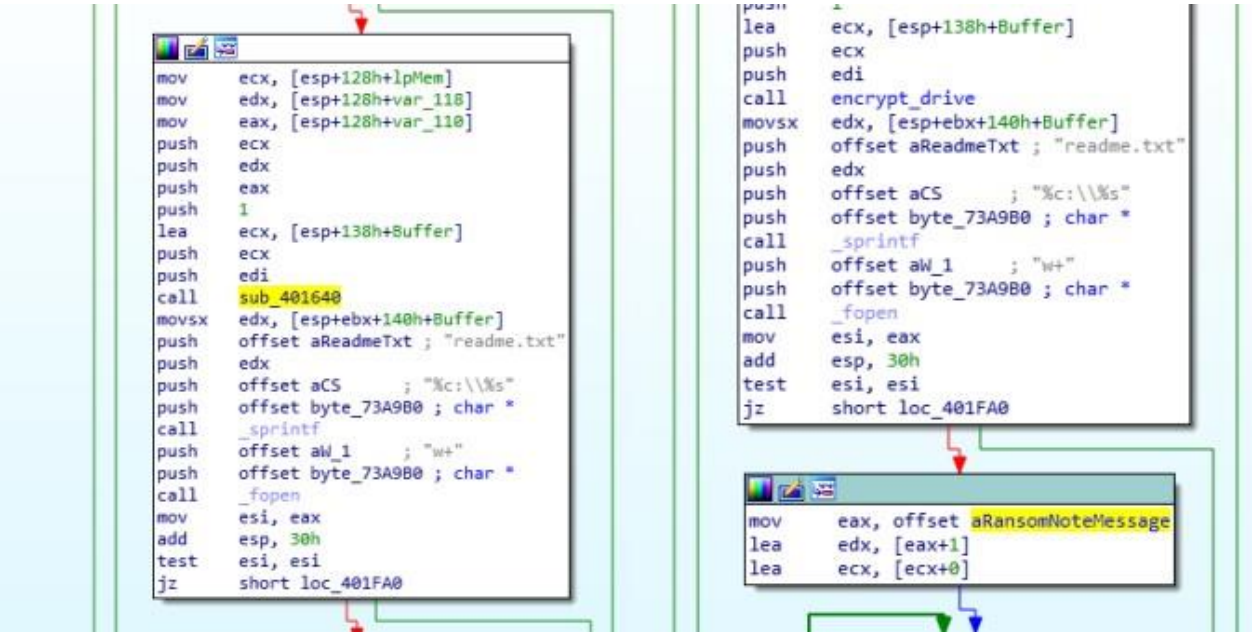

Figure 27

As the final step it deletes the service “msupdate” which was created in the first place and closes the handle.

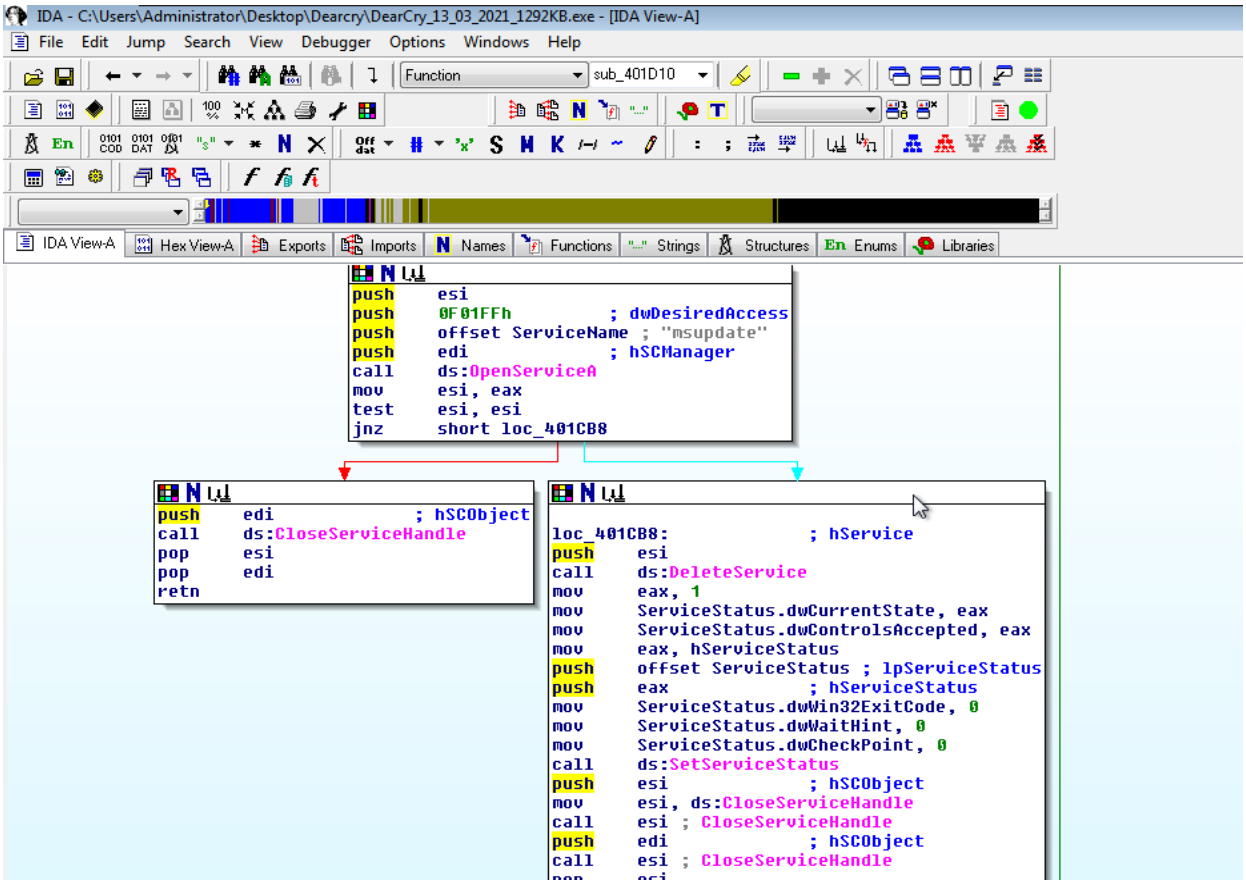

Figure 28

8 WHAT IS BASIC DYNAMIC ANALYSIS

This phase includes the process of execution of the suspicious file in a controlled isolated environment. Most of the times this is performed using special virtual box or sandbox environment because execution of a malware could harm the host machine and compromise the network. Moreover this is typically performed only after the static and advanced static analysis phase. It enables malware analysist to observe the behavior of the malware and the changes it is doing to the compromised system in the execution time. Even though this is more powerful than the static analysis methods this has its own drawbacks since some malwares were developed to identify the sandbox environments and obfuscate their functionalities if they were able to identify restrictions and act in a different way than its normal execution.

9 GETTING STARTED WITH BASIC DYNAMIC ANALYSIS

Since we have identified that this file is a 32-bit executable the file name has changed from .bin extension to .exe to run it and perform the basic dynamic analysis of this ransomware.

In this phase Sandboxie tool was used and it is a free and open-source Windows sandboxing application which enables users to use this to test untrustworthy websites and applications in a controlled virtual environment.

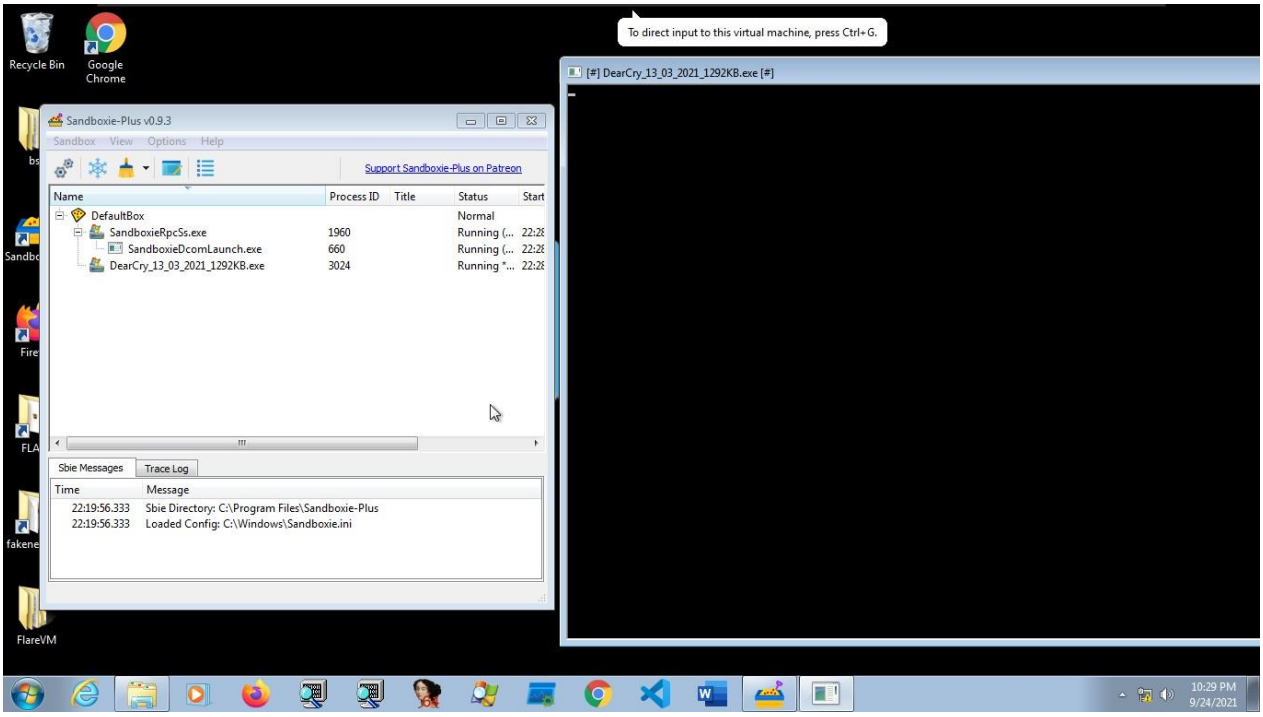

Figure 29

As shown in the above image after executing the ransomware blank console appeared and automatically closed in a while. Then the file system analysis was performed and there we were able to be found files with. CRYPT extension as shown in the following image.

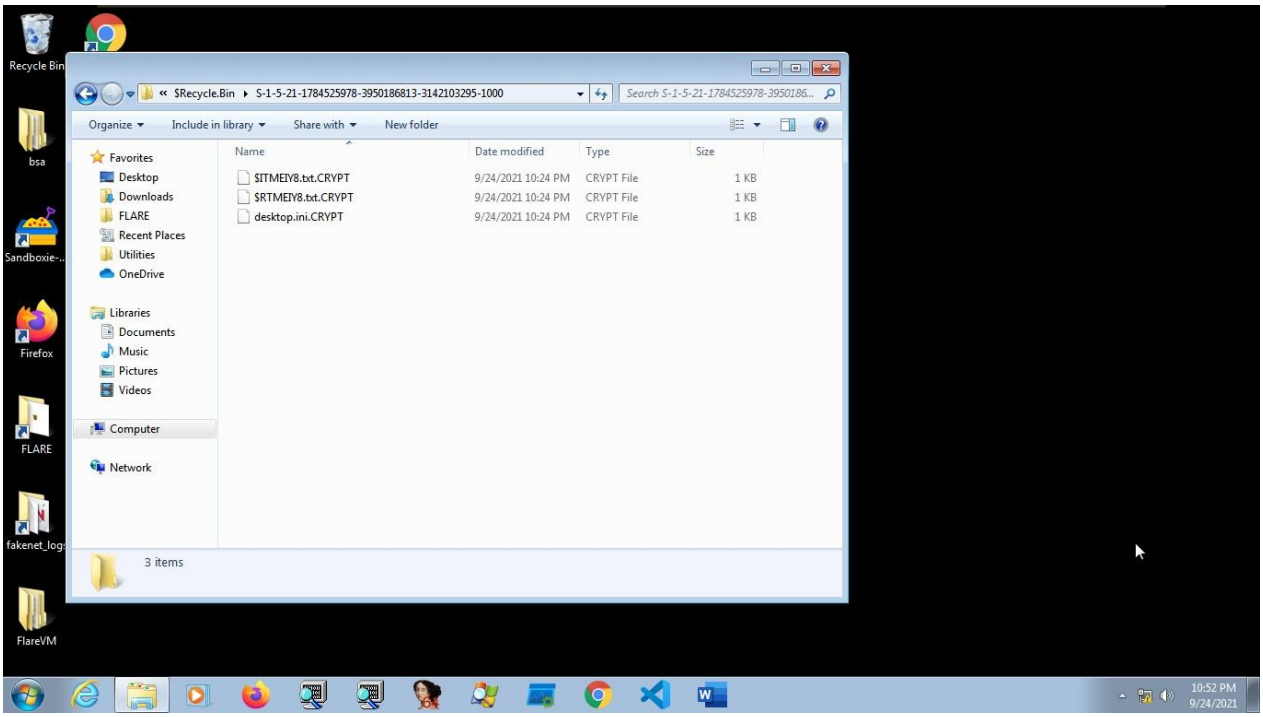

Figure 30

As the next step we used clean windows 7 machine and created one text file named “TextDoc.txt” and another with the name of “TextDoc.xyz”. The xyz extension was intentionally added to check the Dearcry ransomware’s impact. After running the ransomware we were able to find out that “TextDoc.txt” was replaced with an encrypted file called “TextDoc.txt.CRYPT” but “TextDoc.xyz” was not encrypted by the ransomware. Therefore we can come to an assumption that this Dearcry ransomware only encrypts files with predefined extensions by the malware author.

Figure 31

Also we were able to find the ransom note created by Dearcry ransomware after the encryption process as shown in the image below.

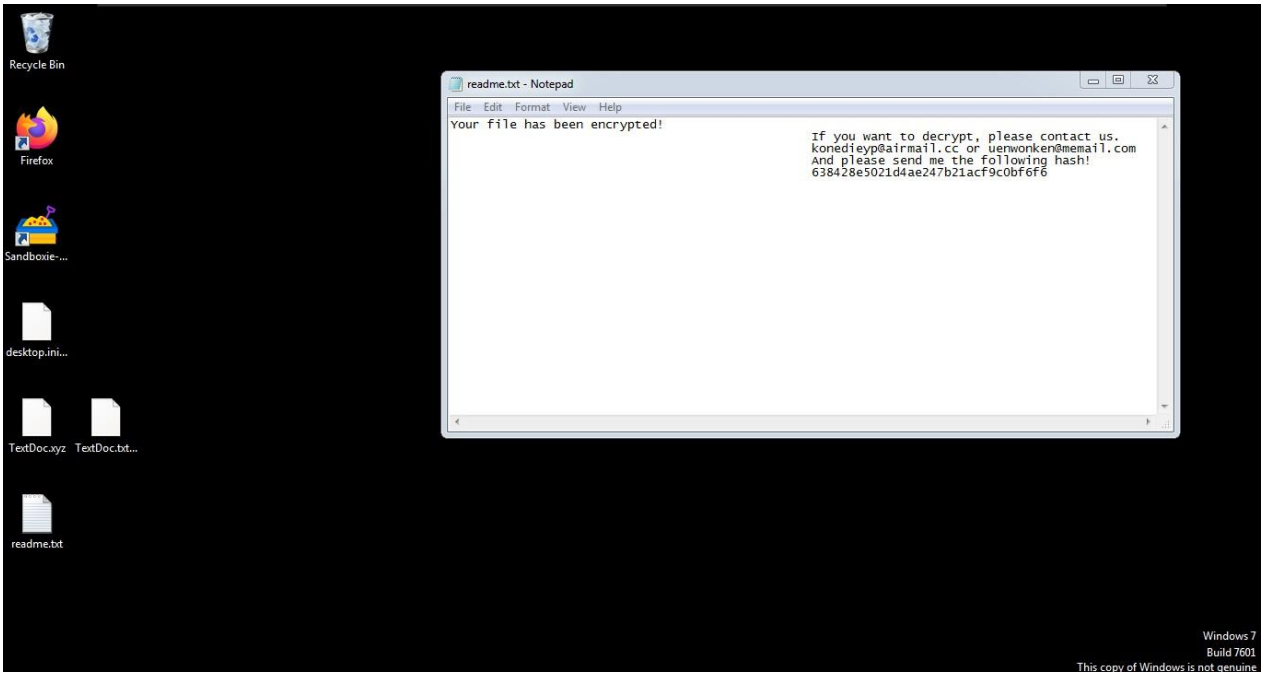

Figure 32

10 WHAT IS ADVANCED DYNAMIC ANALYSIS

In this phase a debugging tools are utilized in most of the times to identify the functionalities of the malware executables, that otherwise could have been impossible to acquire utilizing other approaches explained before. Because it is thoroughly examined rather than static analysis, it is difficult to miss significant behaviors performed by the malwares. One of the key differences in advanced static analysis and advanced dynamic analysis is, advanced dynamic enables to change the opcodes, change the program execution, and observe the results stored in different registers and memory locations in real time. Here also sound knowledge in low level languages like assembly is highly required and considered as a specialist skill that might be challenging to beginner level programmers.

11 GETTING STARTED WITH ADVANCED DYNAMIC ANALYSIS

To start with the advanced dynamic analysis a clean snapshot of the system was utilized and x32dbg which is a 32-bit version of x64dbg debugger (because we were able to identify this malware sample is a 32-bit executable), is used to examine what actually is happening under the hood since it allows us to step through the code when the malware executes. Apart from that Process Hacker was run simultaneously to examine the resource utilization.

As it's shown in the following figure when you initially load the malware it does all the necessary things in the background that is needed by our malware to function properly.

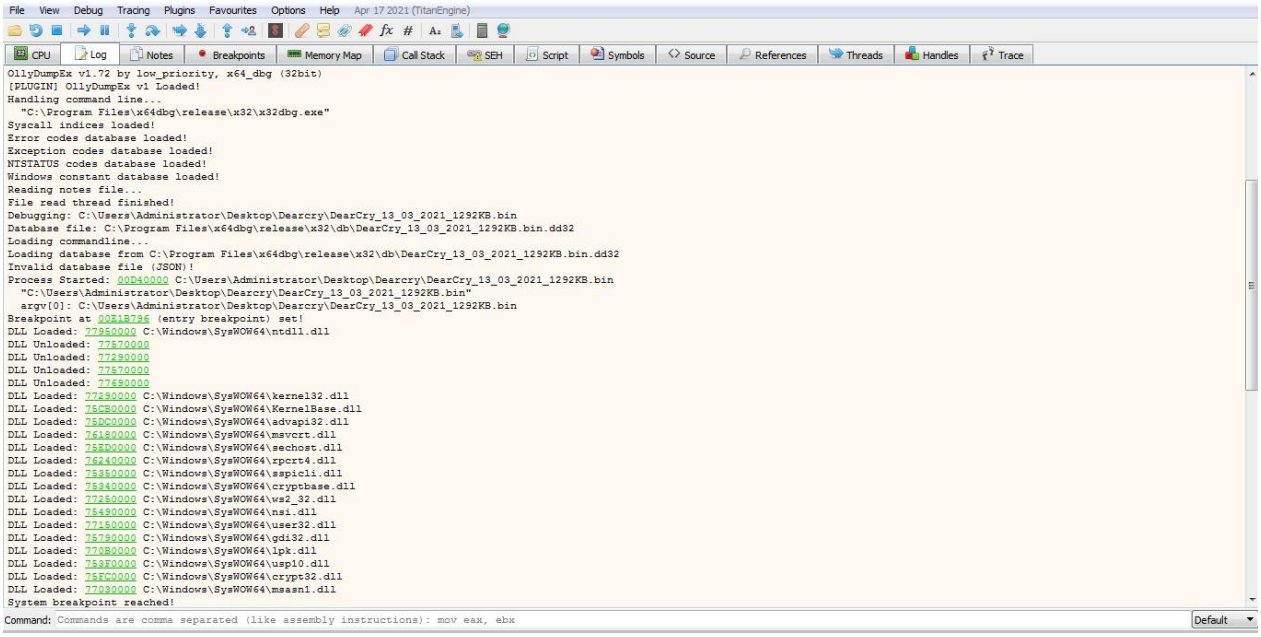

Figure 33

According to the previous findings that have been done in the static analysis phase breakpoints have been set and in addition to the blacklisted imports we intentionally set up a break point on IsDebuggerPresent to check whether we hit that while debugging because if we hit that we can make sure that this ransomware is going to take different approaches than the usual one to complexify the analyzing process because malware authors often use that technique because they do not want people to reverse engineer their malwares so easily.

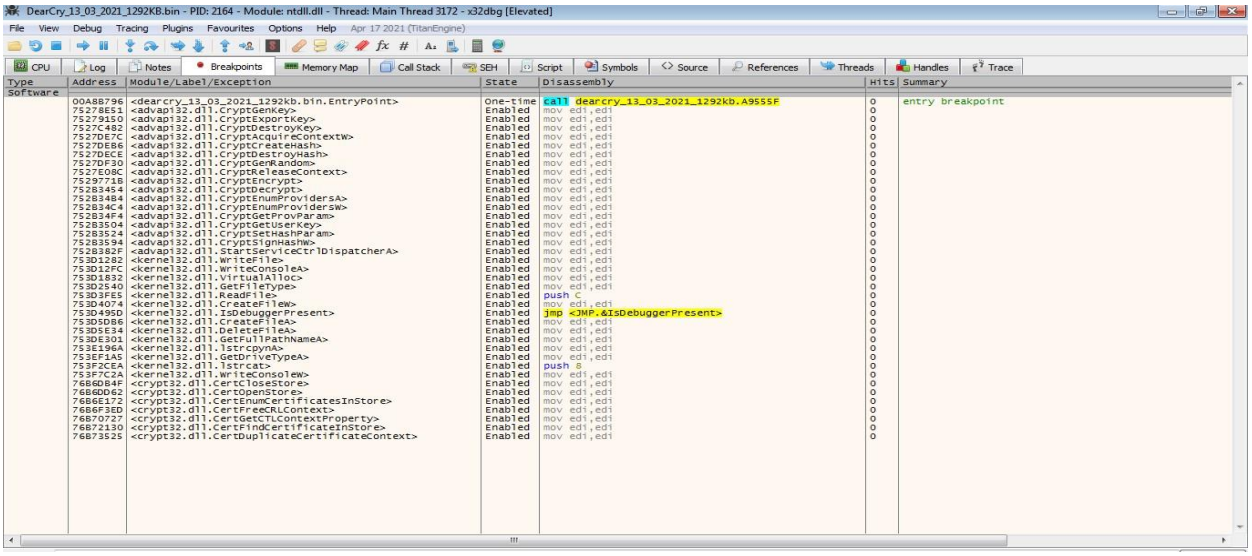

Figure 34

When we start executing it initially hit its first break point and it is the exact entry point address that we have been identified via pestudio while analyzing the sections information.

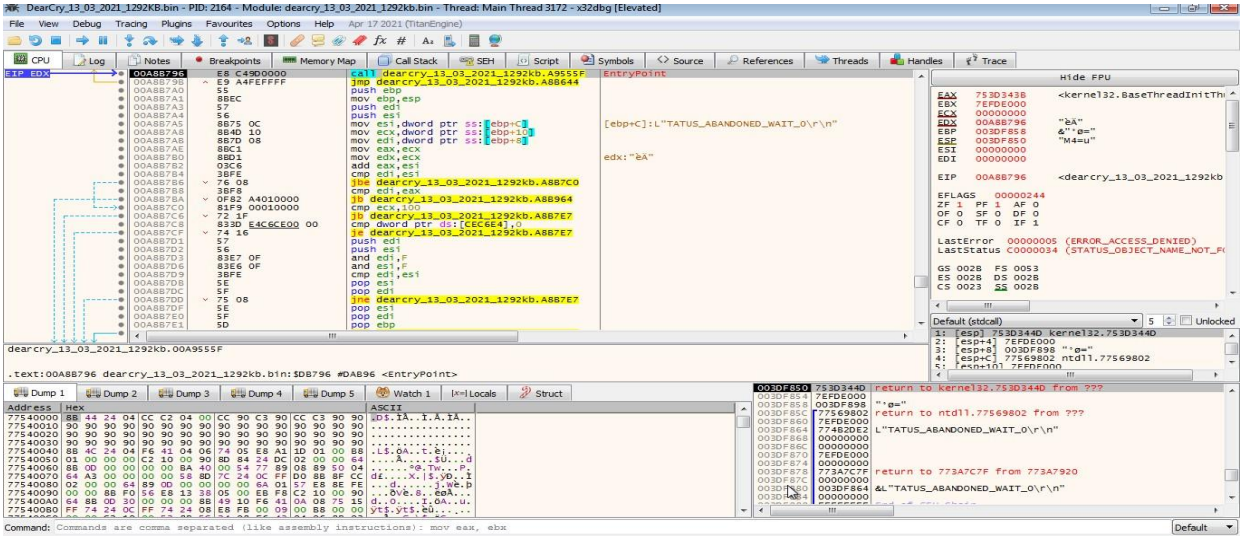

Figure 35

Then we hit our next break point and it is GetFileType and it is one of the Microsoft APIs and typically used to interact with the operating system. What that basically does here is passing a handle and return different codes depending on the file type of the system.

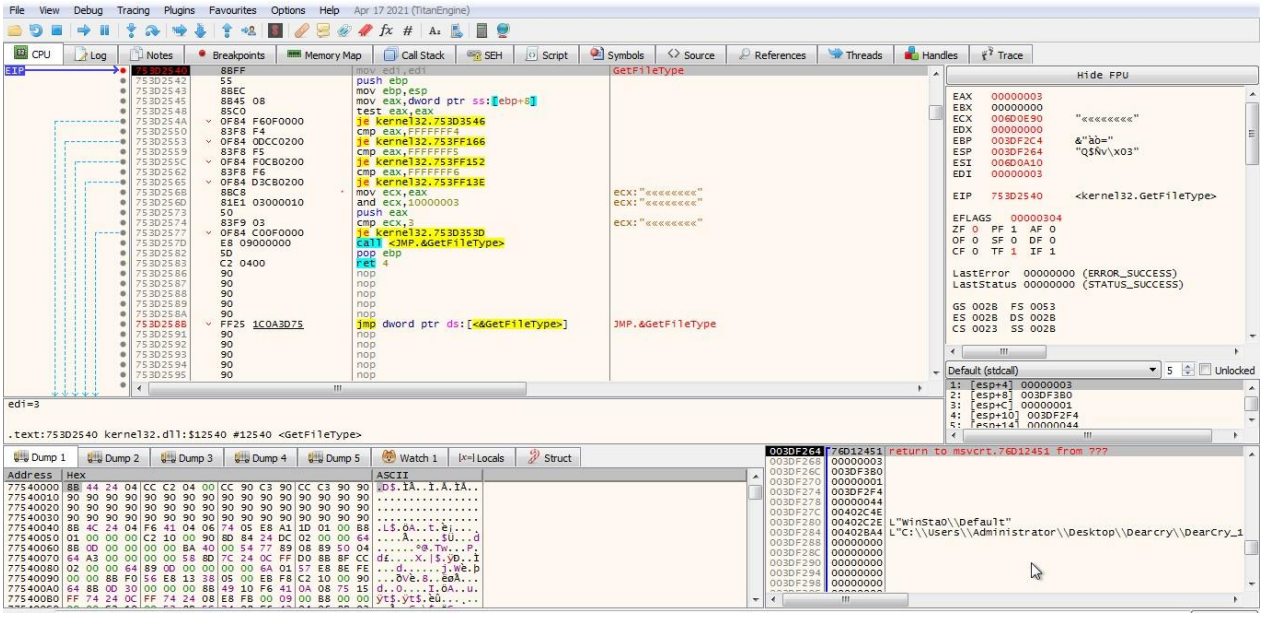

Figure 36

By observing the EAX register we can find out it returned value 2 which stands for a console or an LPT device. But in this case, we can determine it indicates the console of the ransomware and if we continue, we can notice it is returning different handles for that console.

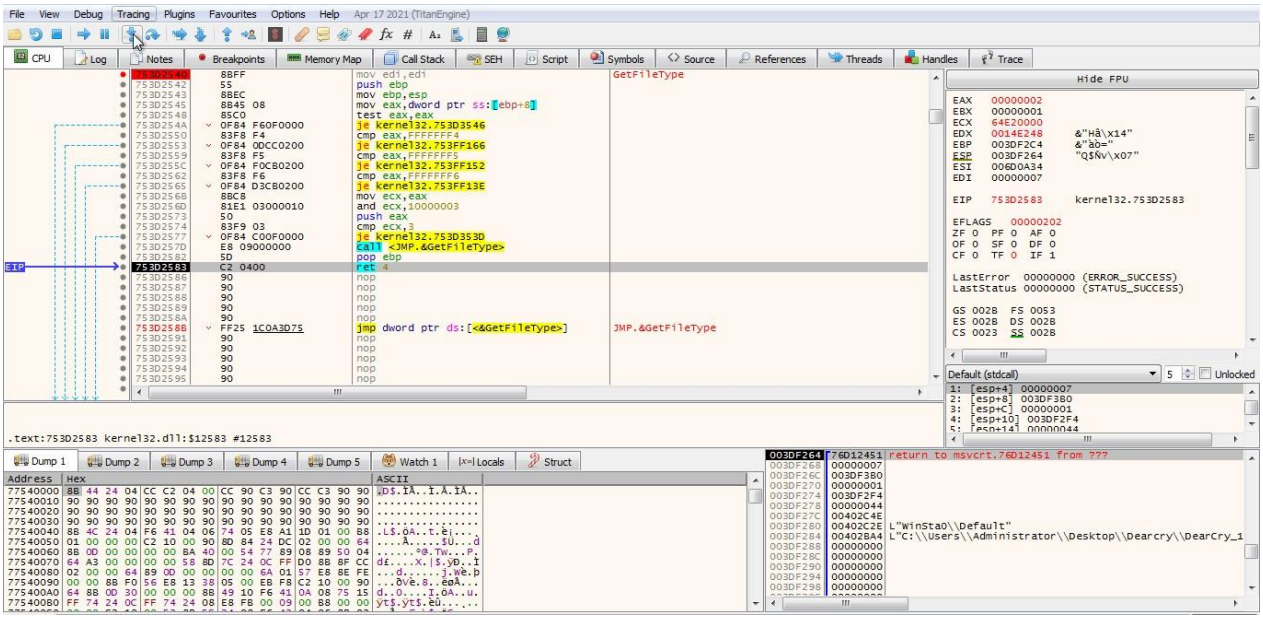

Figure 37

Then it hits the next break point, StartservicedispatcherA, which is one of the first things that ransomware's main function calls at its initial steps. Here it creates tries to create a connection with service control manager to register and start a service. By observing the value stored in the EAX register the name of the service can be found and it is named as "msupdate" which is responsible for the cryptographic related tasks like encryption and gets deleted in the end of the execution.

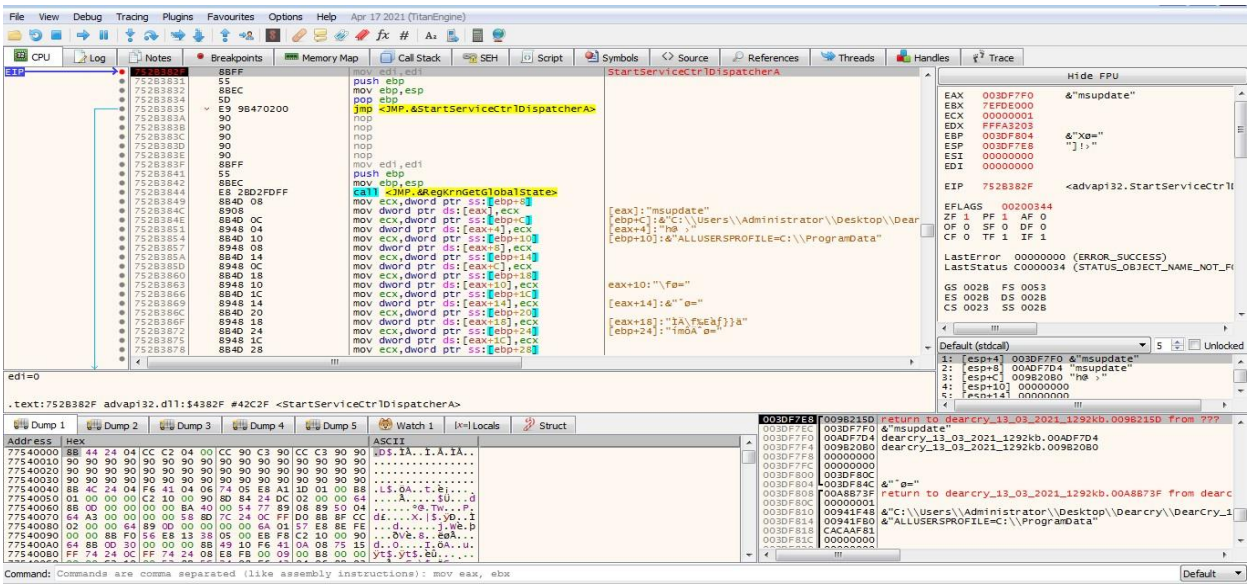

Figure 38

After passing through couple of opcodes it reached another break point called GetDriveTypeA. Which is used to determine the drive types of the victim machines which is straight forward and self-explanatory by function's name.

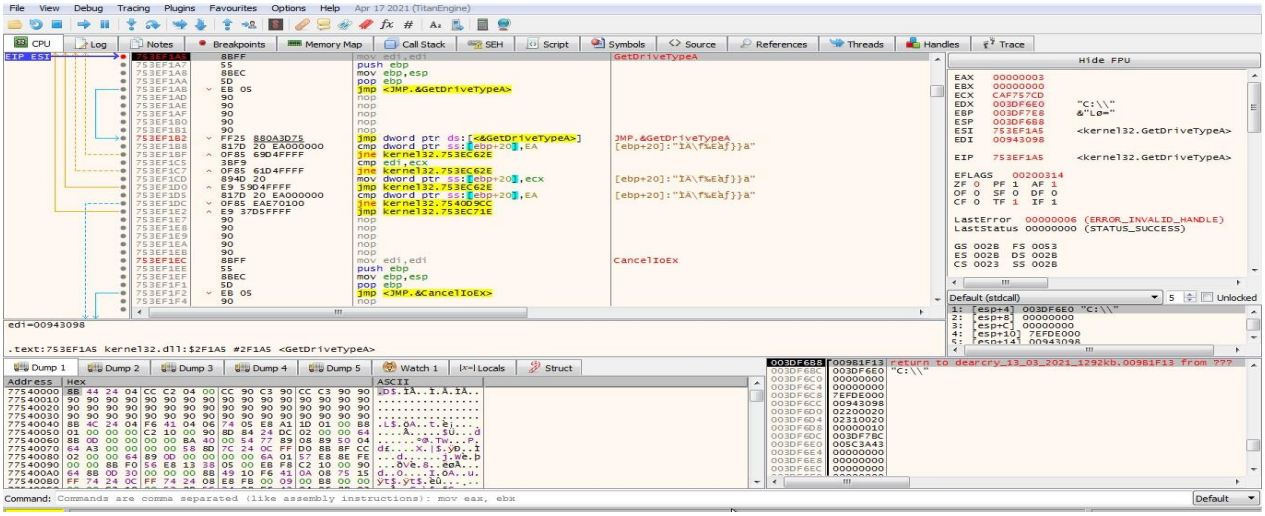

Figure 39

By stepping into that function, it can be observed that the value stored in the EAX register is compared with the value 5, which stands for CD ROM drive. So that it turns out the meaning that this function is used to enumerate and identify all the drive types one by one which are connected to the victim machine before the encryption process.

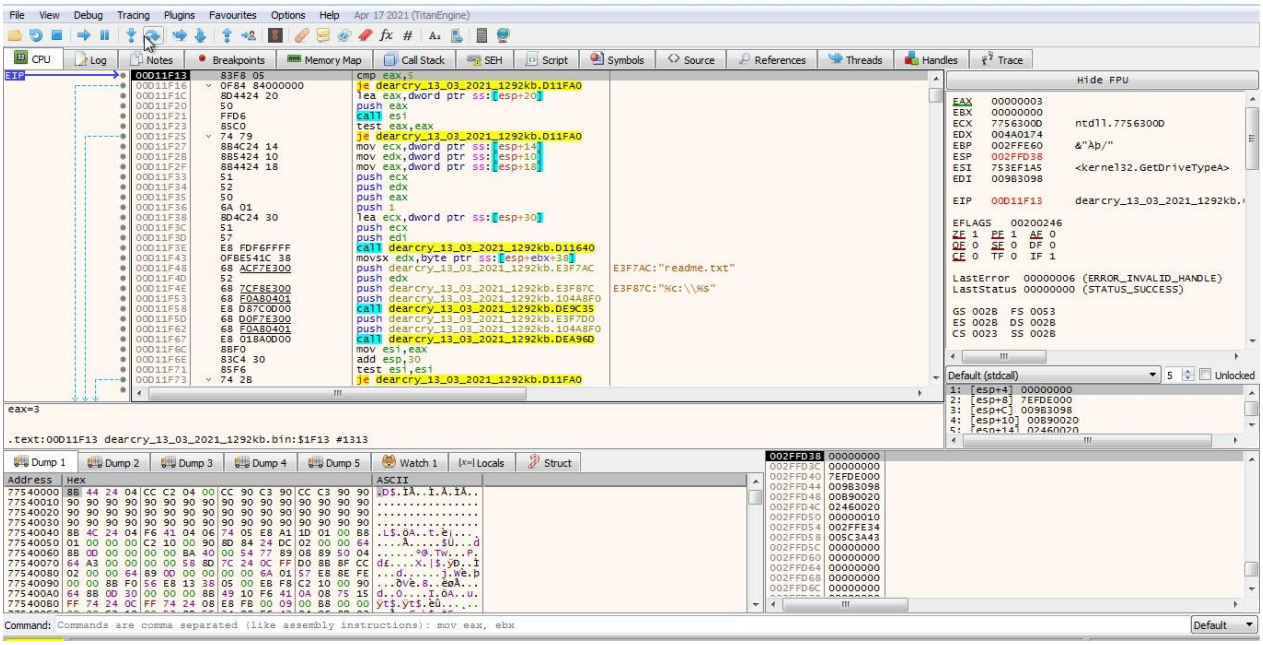

Figure 40

After executing the function it returns value 3 and stored it in the EAX register which stands for fixed hard disk drive.

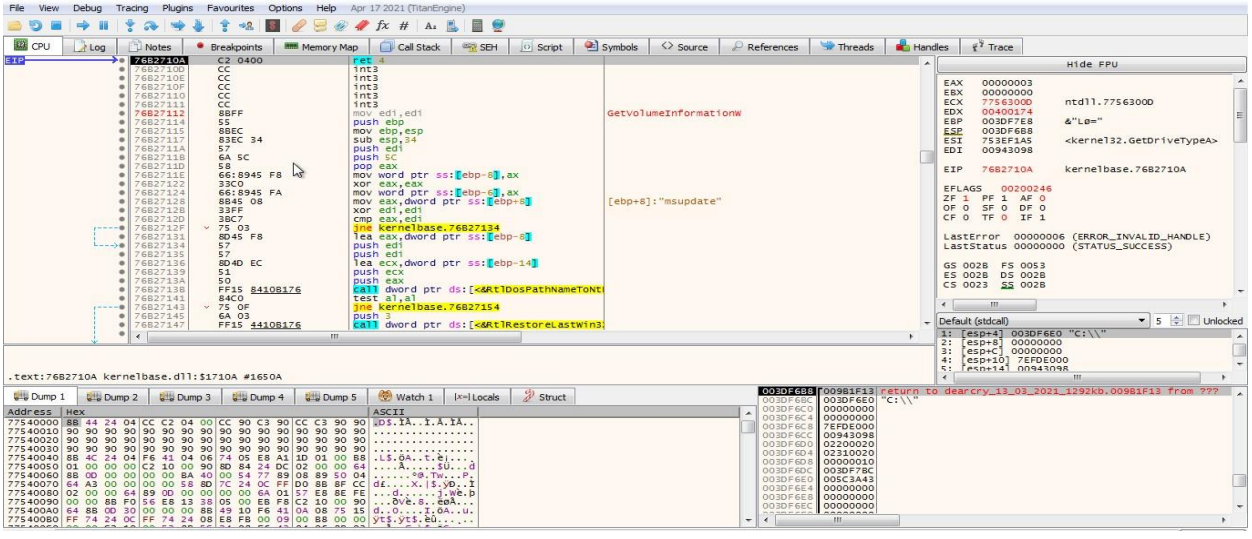

Figure 41

Right after the enumerating process it reaches to next break point CreateFileA which enables dearcry ransomware to interact with the files in the victim machine. This function often returns a handle to the file which malware is trying to access.

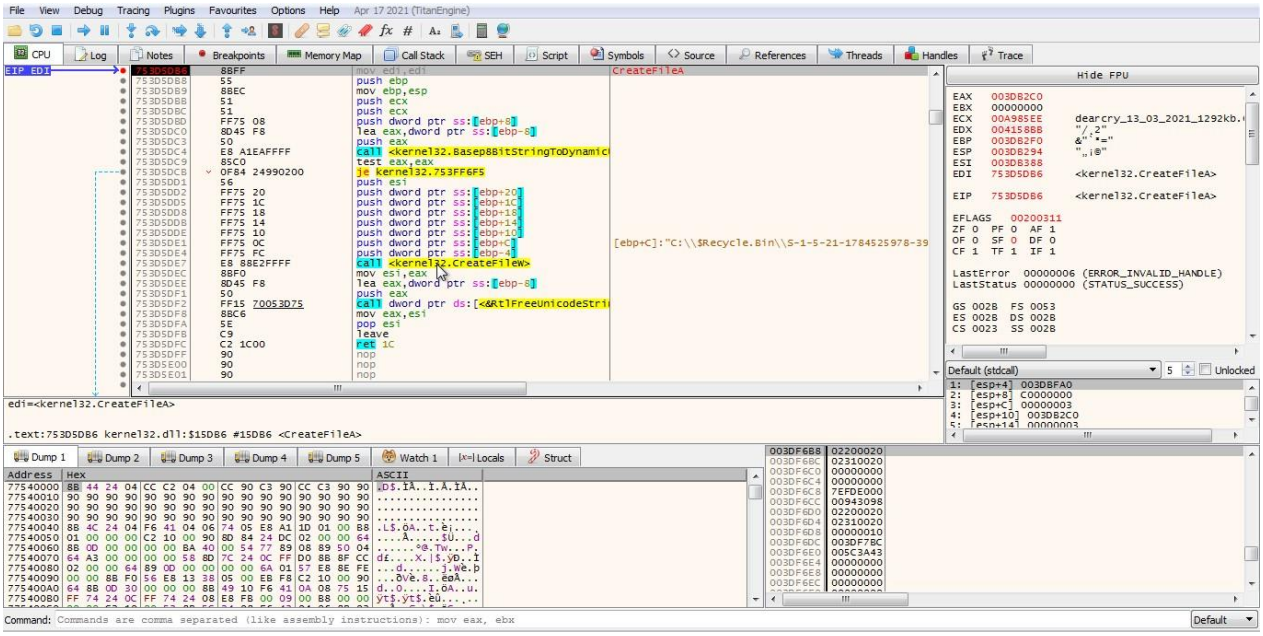

Figure 42

After stepping trough couple of opcodes we were able to find out the actual file in the victim machine the ransomware was trying to access as shown in the following figures.

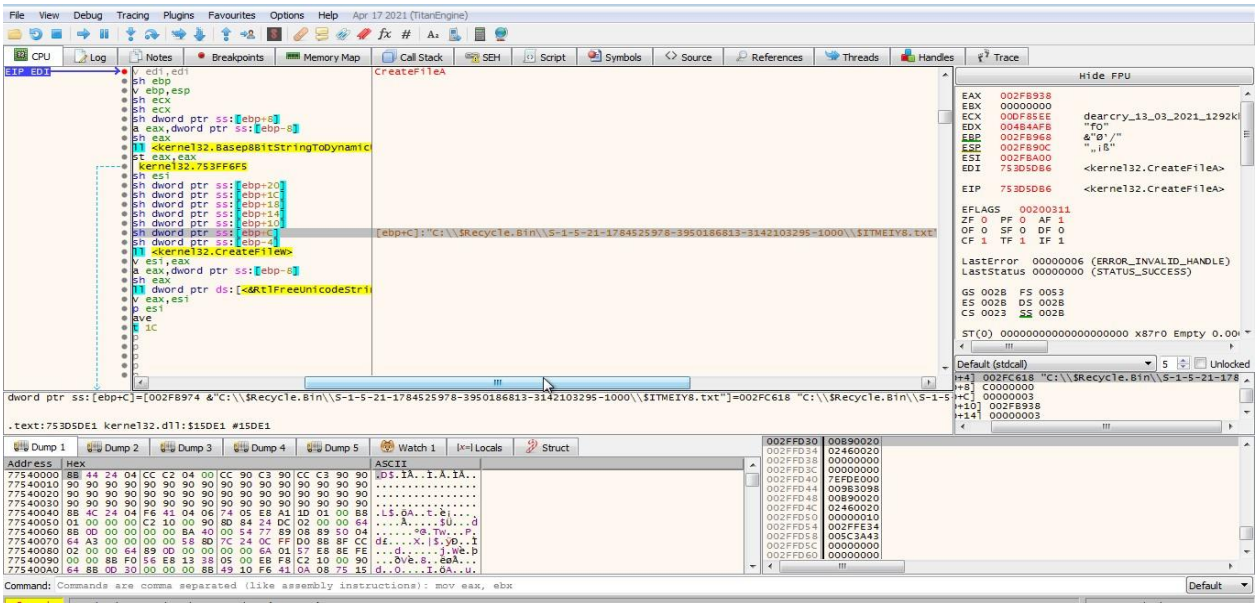

Figure 43

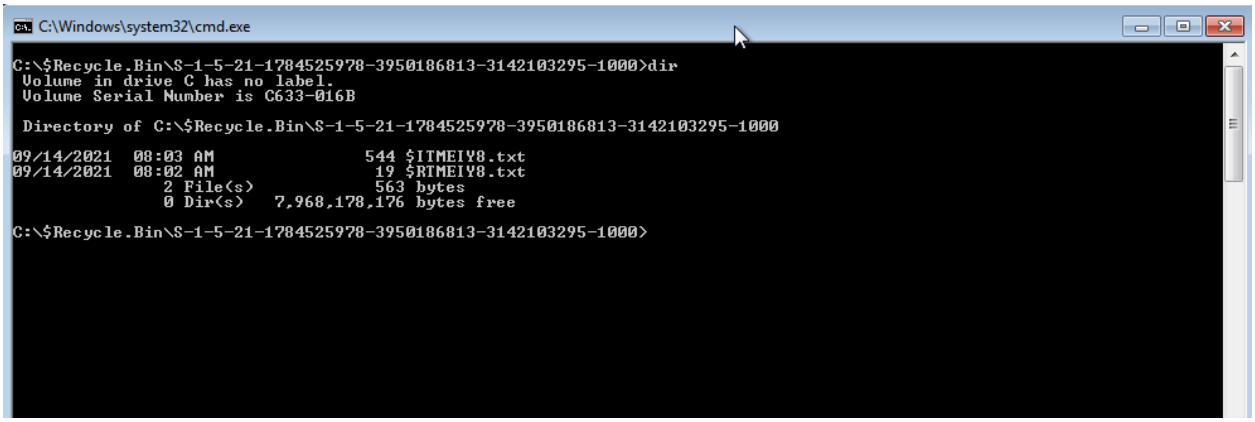

Figure 44

This proves the fact that Dearcry ransomware goes through the entire hard drive-in alphabetical order and the starts accessing files. In this case it is starting with $Recycle.bin directory because '$' resides before the letter 'A'. As explained previously it returns a handle to that particular file as shown in the image below.

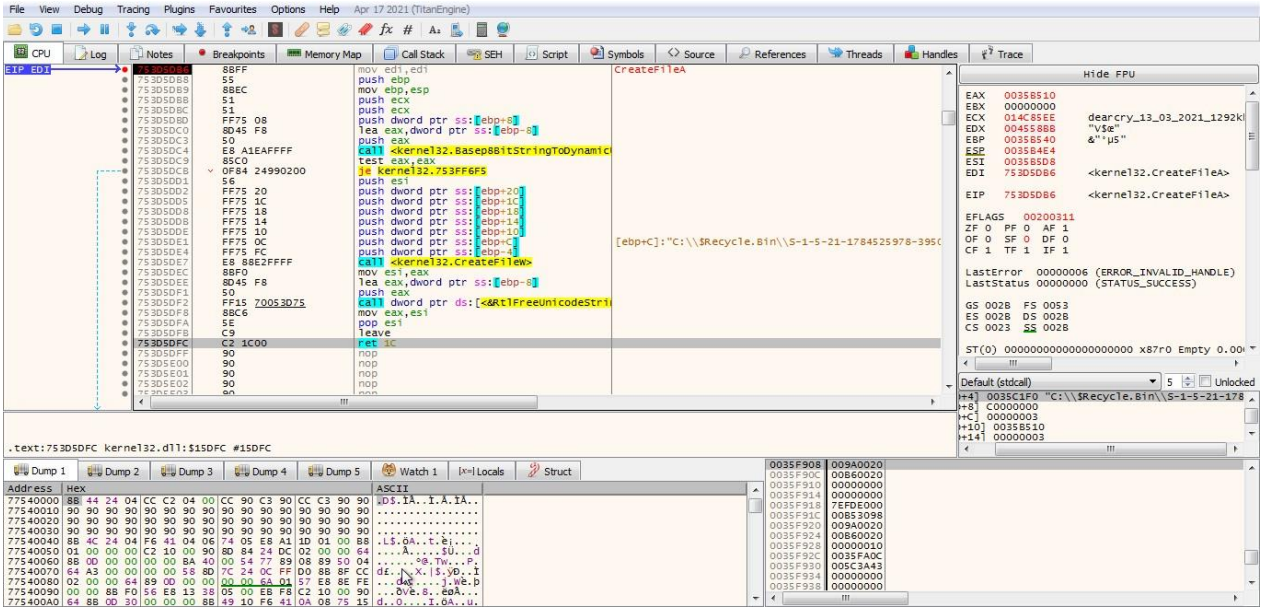

Figure 45

Then CreateFileW was invoked by the ransomware which does the exact same thing as CreateFileA. The reason for that is CreateFileA used to deal with files named in ANSI format

which comprised with 218 characters represented using 8 bits which was used from windows 95 to windows NT operating systems and considered as the standard character set back then. With the time UNICODE character set was introduced and then problem occurred with newly created API calls since they were not compatible with previous versions. To address this problem Microsoft cam up with an idea to end all API calls' names with an A suffix which are dealing with old formats and added a second set of the identical API calls with a W suffix for the Unicode variant. Nowadays the preprocessor maps the appropriate suffixed API call during the compilation time depending on the original name. Therefore CreateFileW was called right after the CreateFileA.

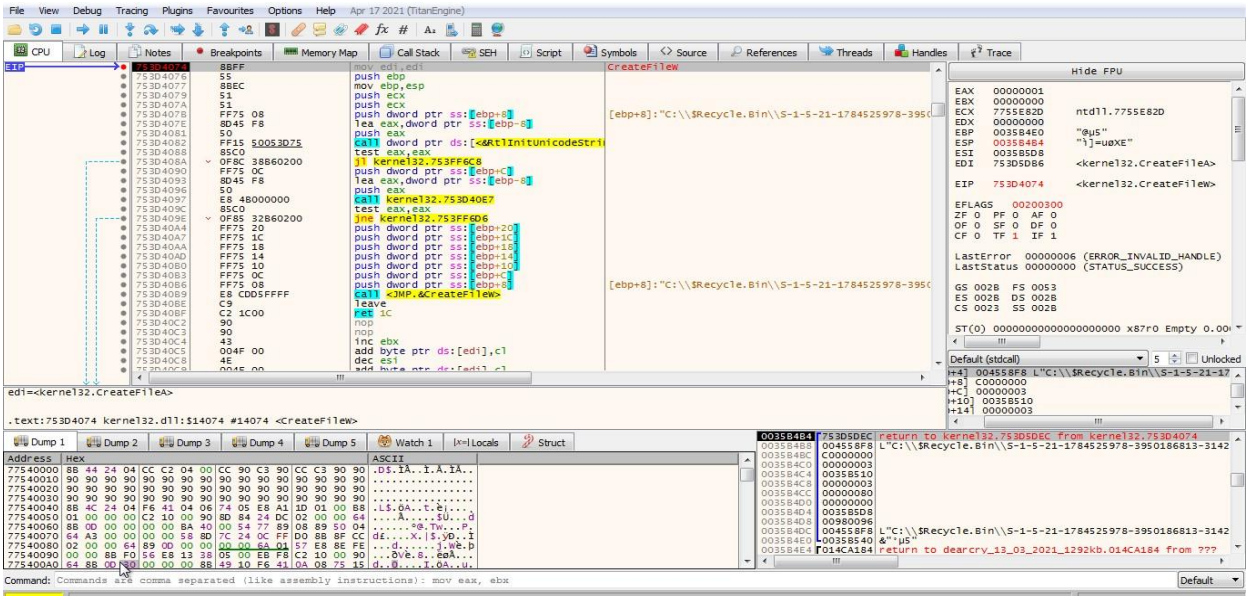

Figure 46

As explained here it creates a handle to that file as shown in the figure below.

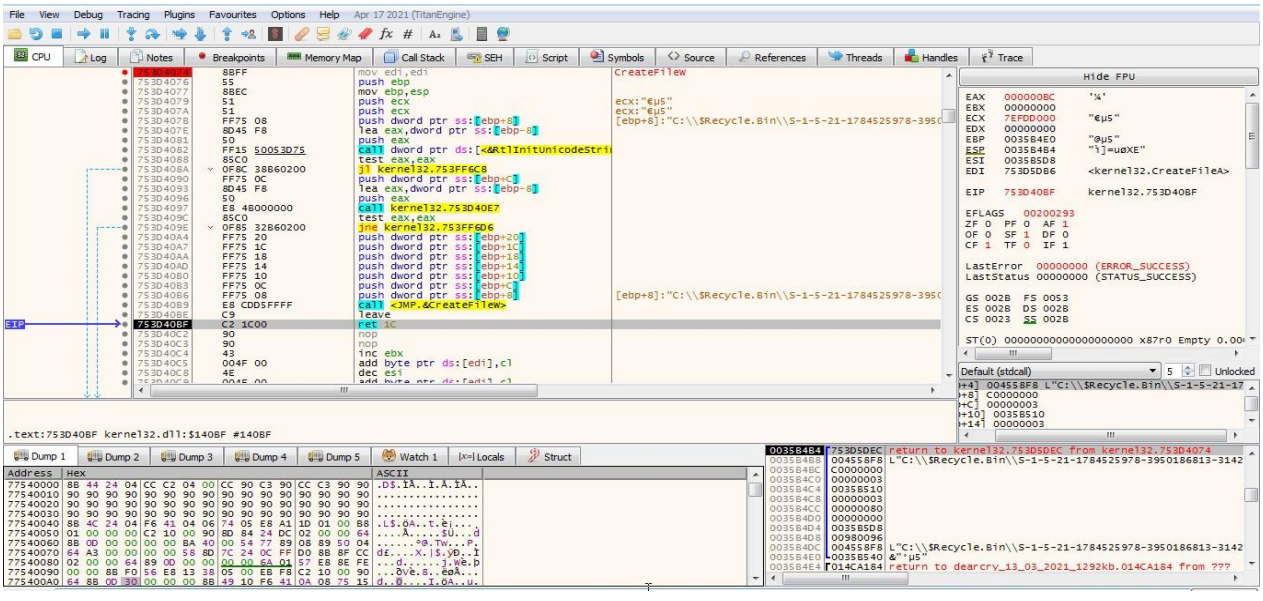

Figure 47

By examining the Dearcry process by process hacker tool we can see the exact same handle name which stored in the EAX register after the function execution is appeared in the handle section. So that we can verify the previous assumption.

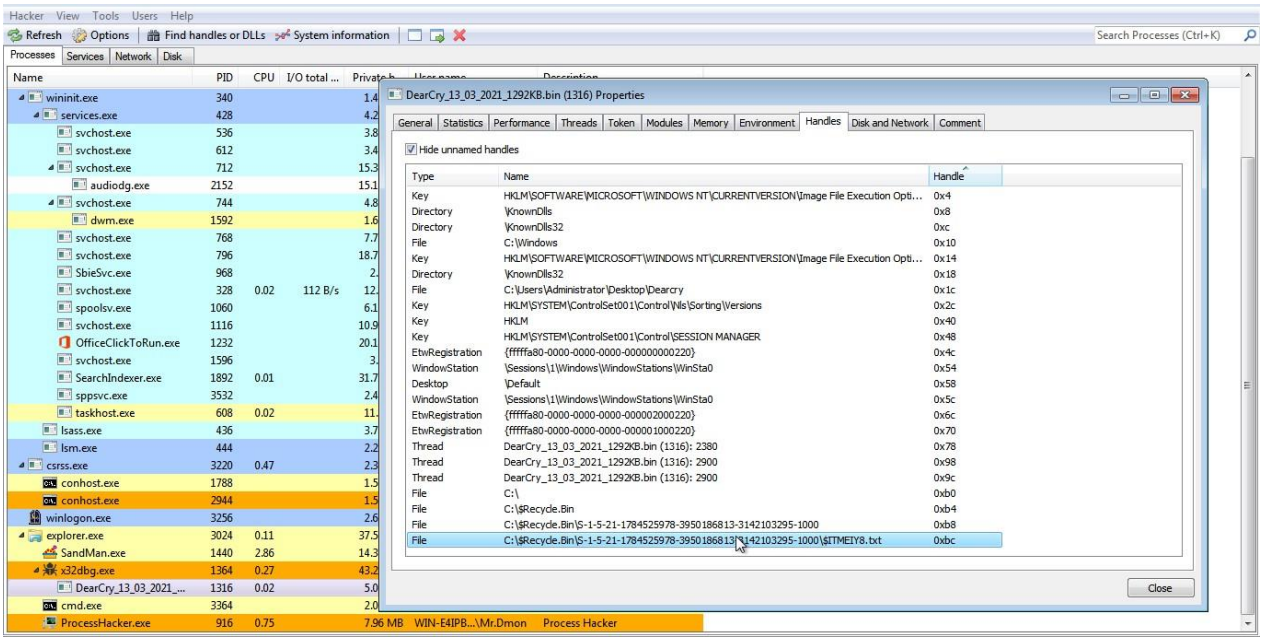

Figure 48

Then it hit the next break point, GetFileType function and as the name explained it is used to obtain the file type of the specific file that this ransomware is trying to access.

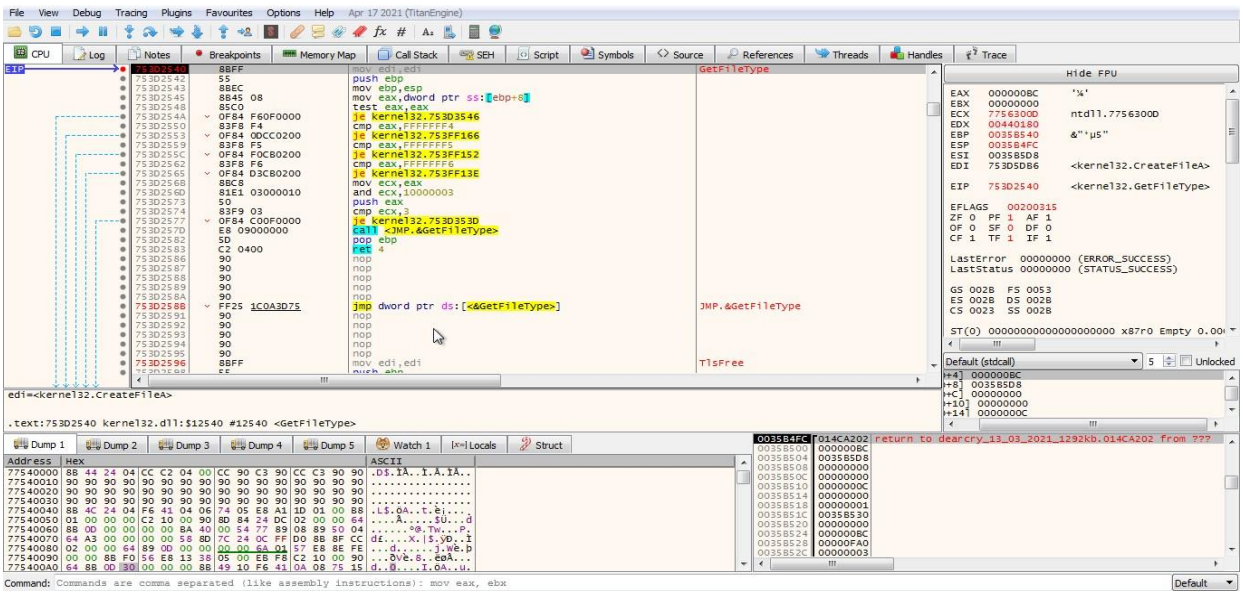

Figure 49

As shown in the following figure it identified that file as a disk file by storing the return value 1 in the EAX register.

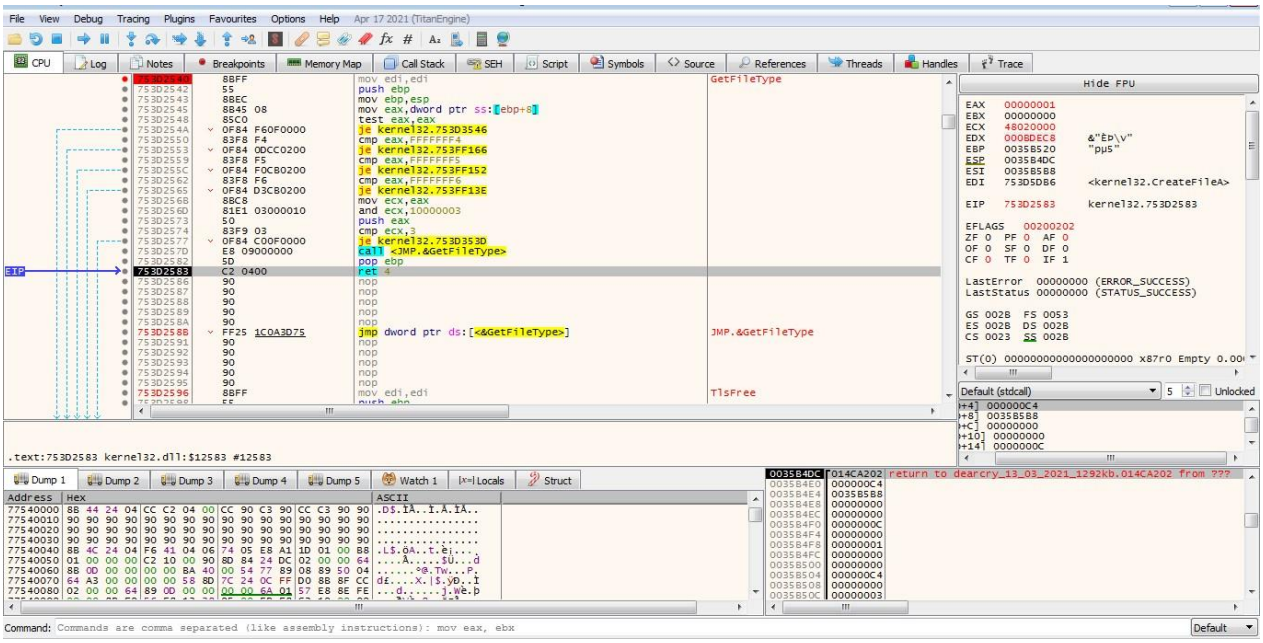

Figure 50

After executing several opcodes we were able to find out that list of file extensions we found out while performing the static analysis as shown in the below image. By observing the register values we can prove the fact that it determines the correct file type that it is trying to access by using that list of file extensions before the encryption process.

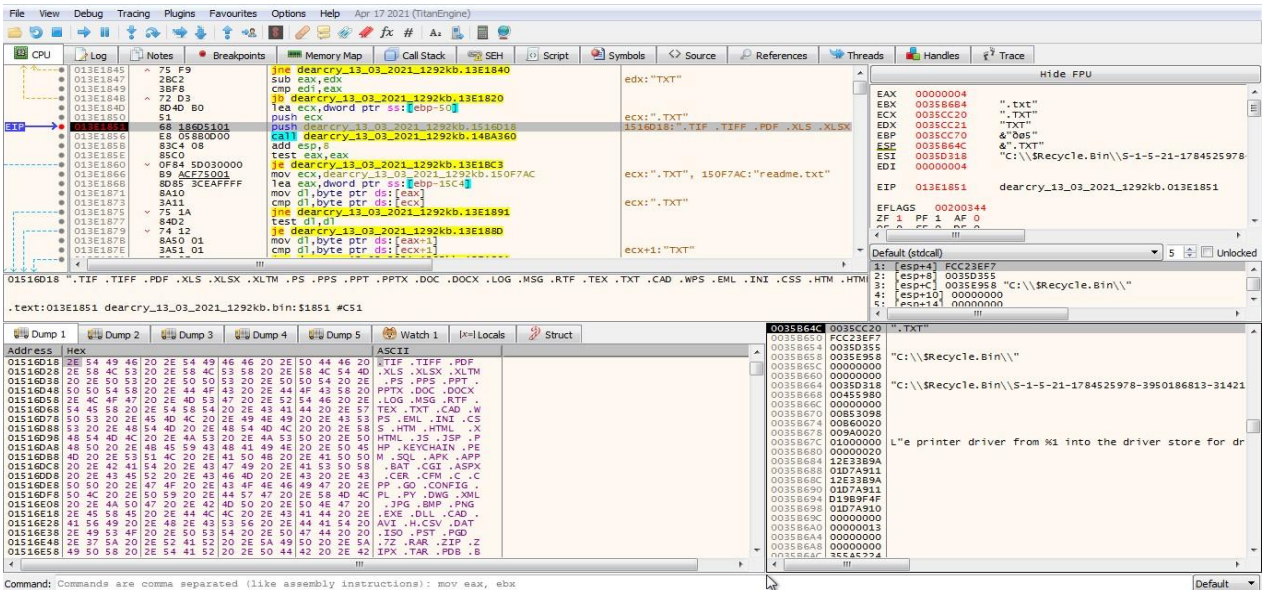

Figure 51

As the next step ransomware attempts to read the data from $ITMEIY8.txt file by calling ReadFile function.

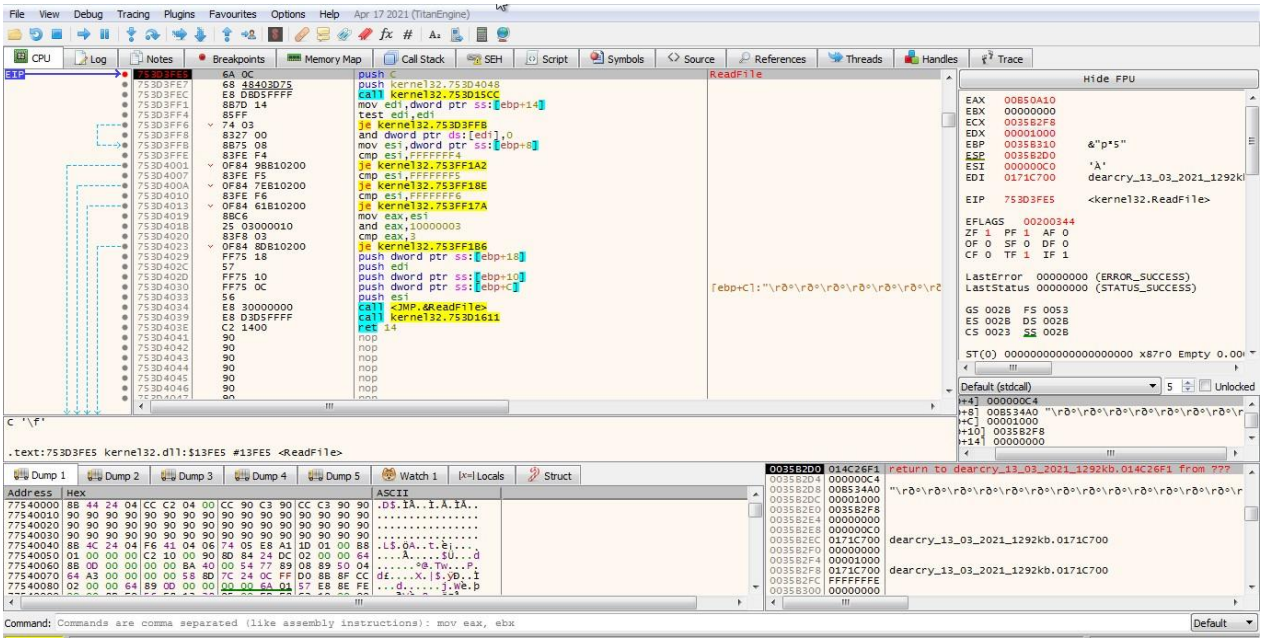

Figure 52

And ended up returning nonzero value to the EAX register as usual which indicates the successful execution of the function.

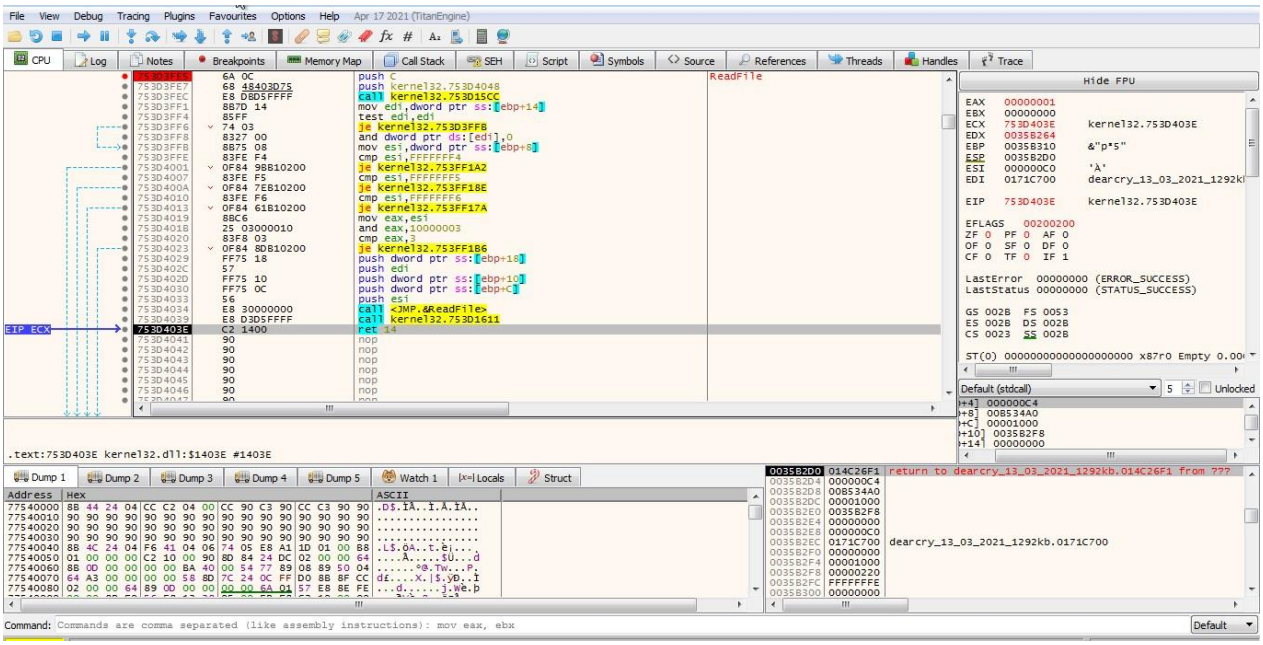

Figure 53

Then the ransomware calls lstrcpynA function which typically used to copy specified amount of characters to the buffer from the source string.

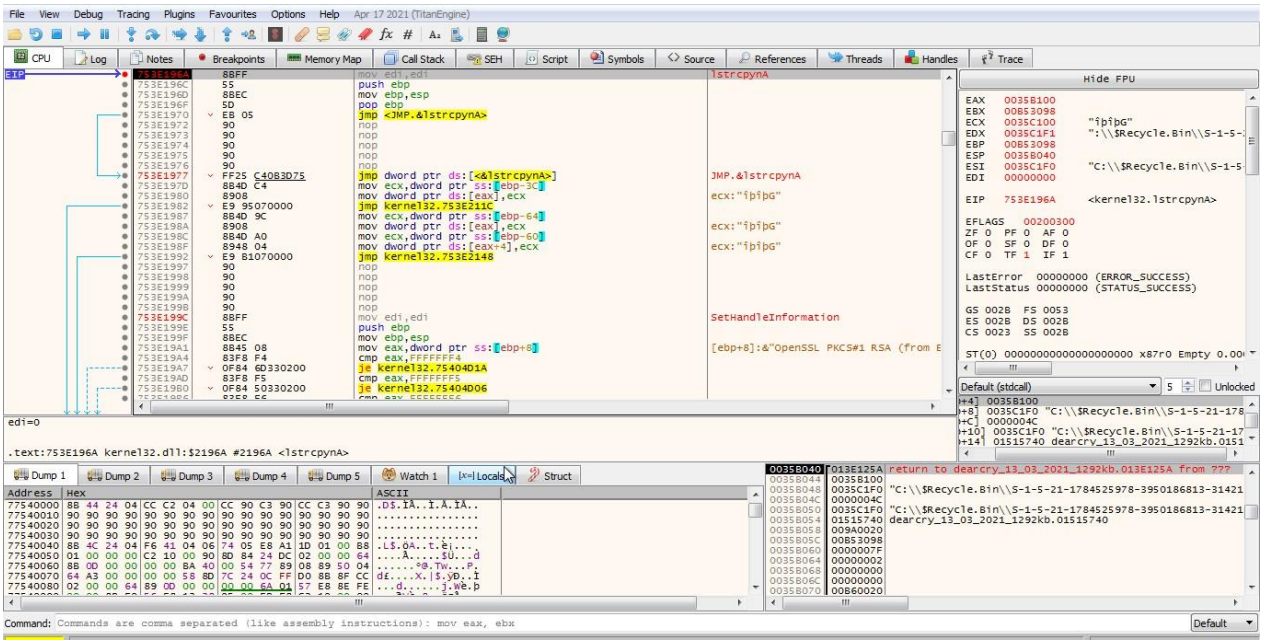

Figure 54

By calling lstrcat it appends strings which are read by the previous function and after successful execution it returns a pointer to the buffer.

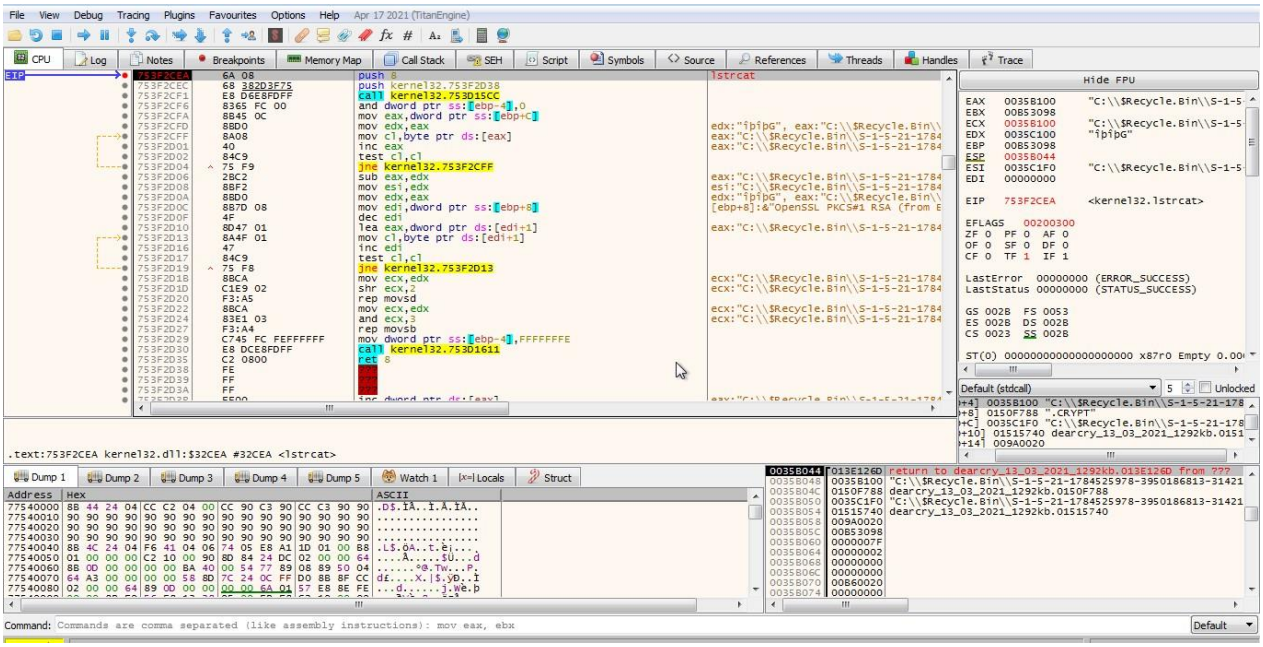

Figure 55

After that Dearcry ransomware returns a file which is named as $ITMEIY8.txt.CRYPT

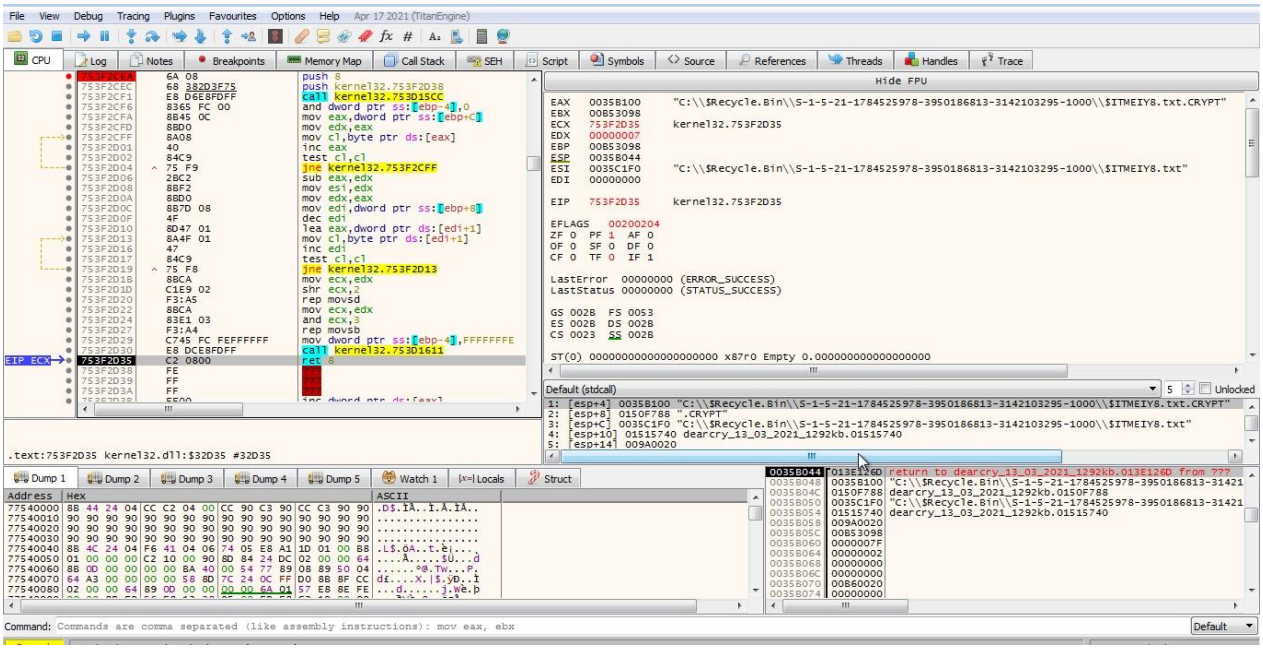

Figure 56

Even though it created a new file in the same location of the original file it was found out that the file was an empty one as shown in the following image.

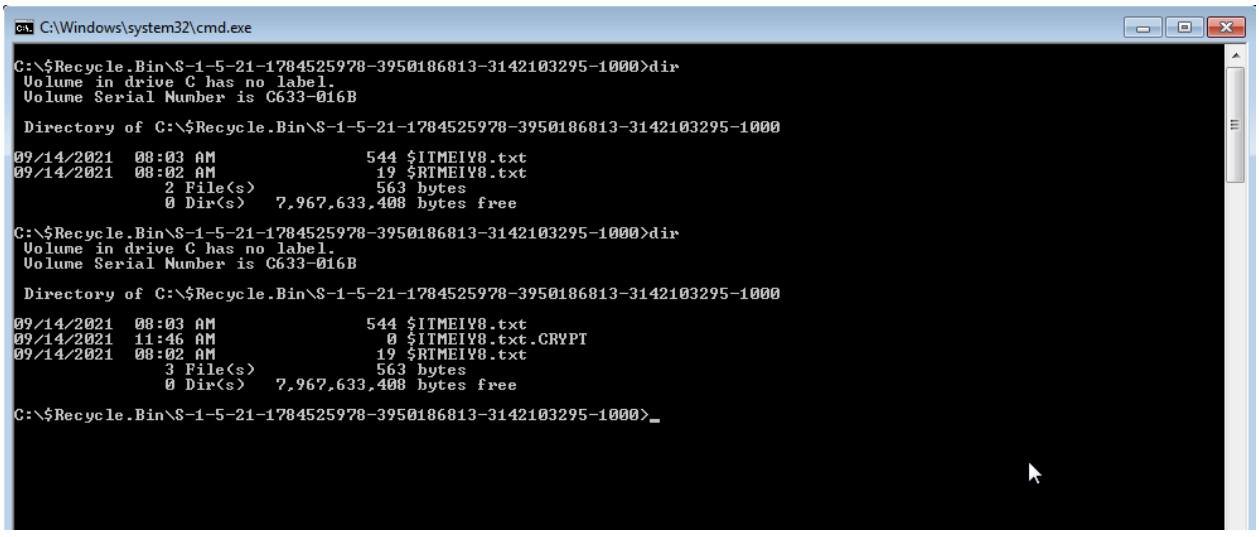

Figure 57

CryptAquireContextW is used to obtain a handle to particular key container within a predefined cryptographic service provider which ultimately results the ransomware to acquire the specific private key which is associated with the RSA public key that has been found in the previous analysis phase.

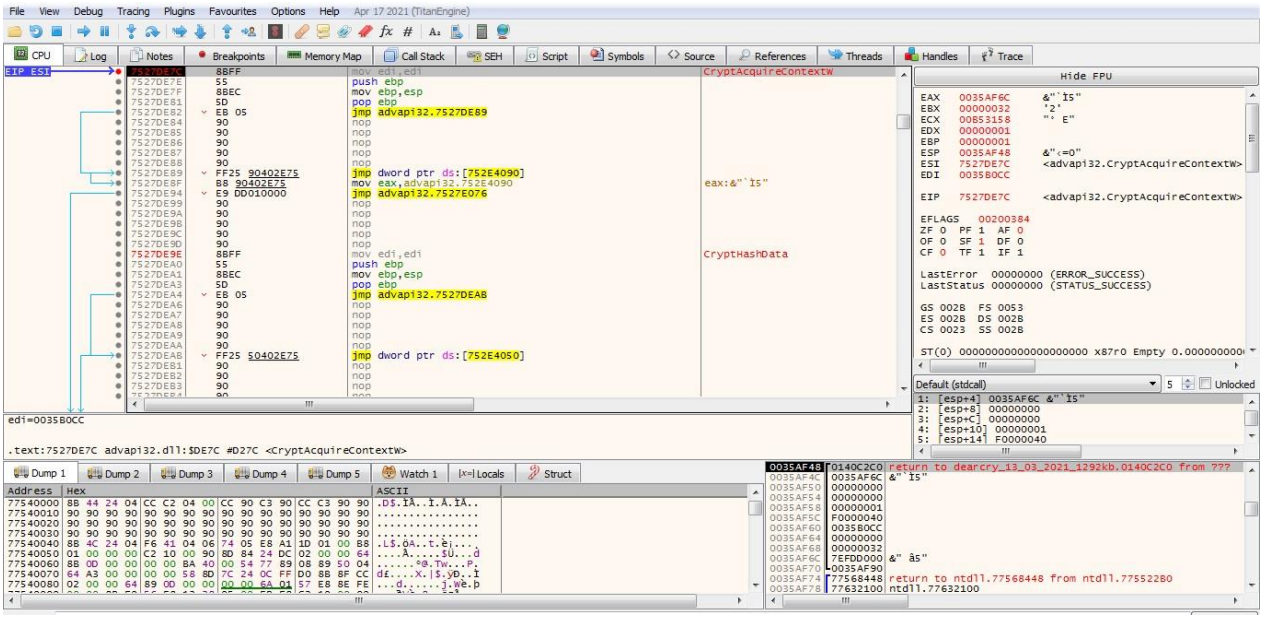

Figure 58

CryptGenRandom function is called after that, and it is typically used to generate pseudo random numbers which are cryptographically secure when it comes to cryptographic related. It is comprised in Microsoft CryptoAPI that enables computer programmers to encrypt data in windows-based platforms.

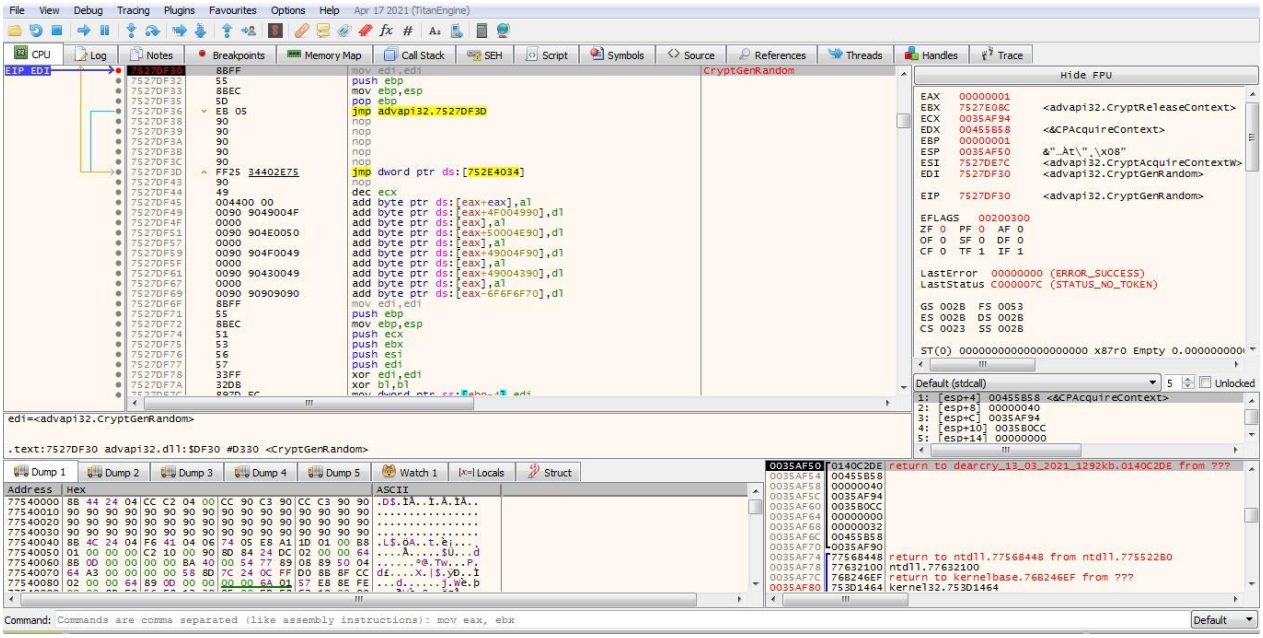

Figure 59

Random data generated by the Dearcry ransomware by calling aforementioned function can be found in the memory dump1 as shown in the following image.

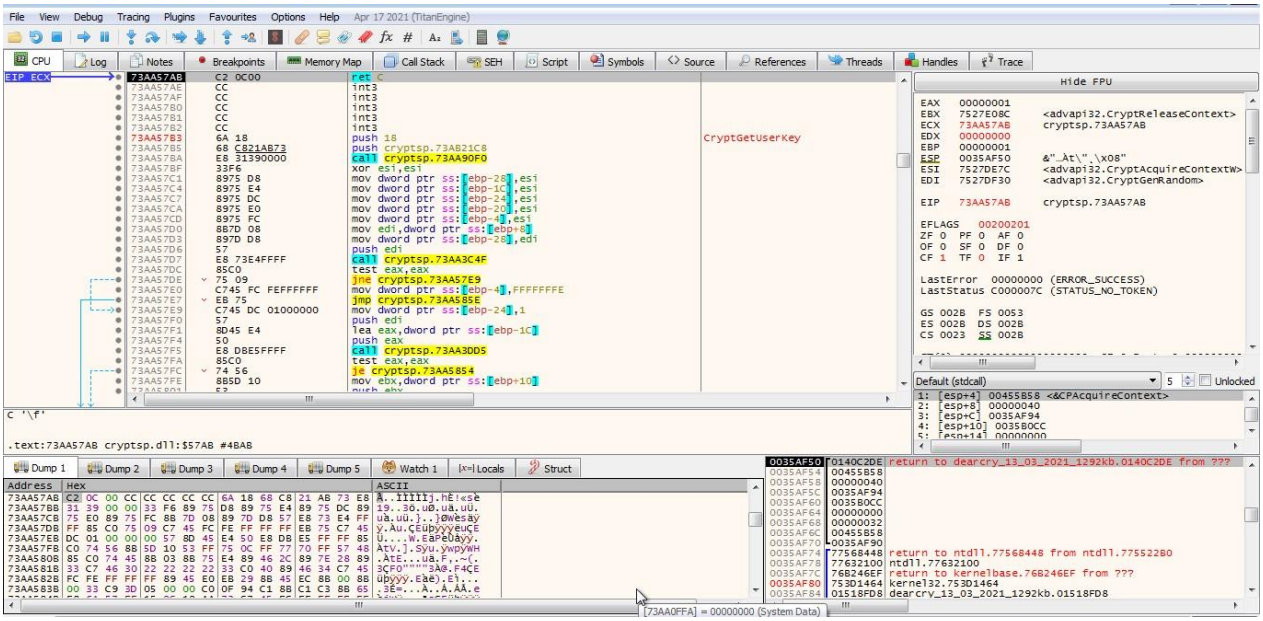

Figure 60

Then it reaches the next breakpoint CryptReleaseContext, that is responsible for releasing the handle to the cryptographic service provider and the key container.

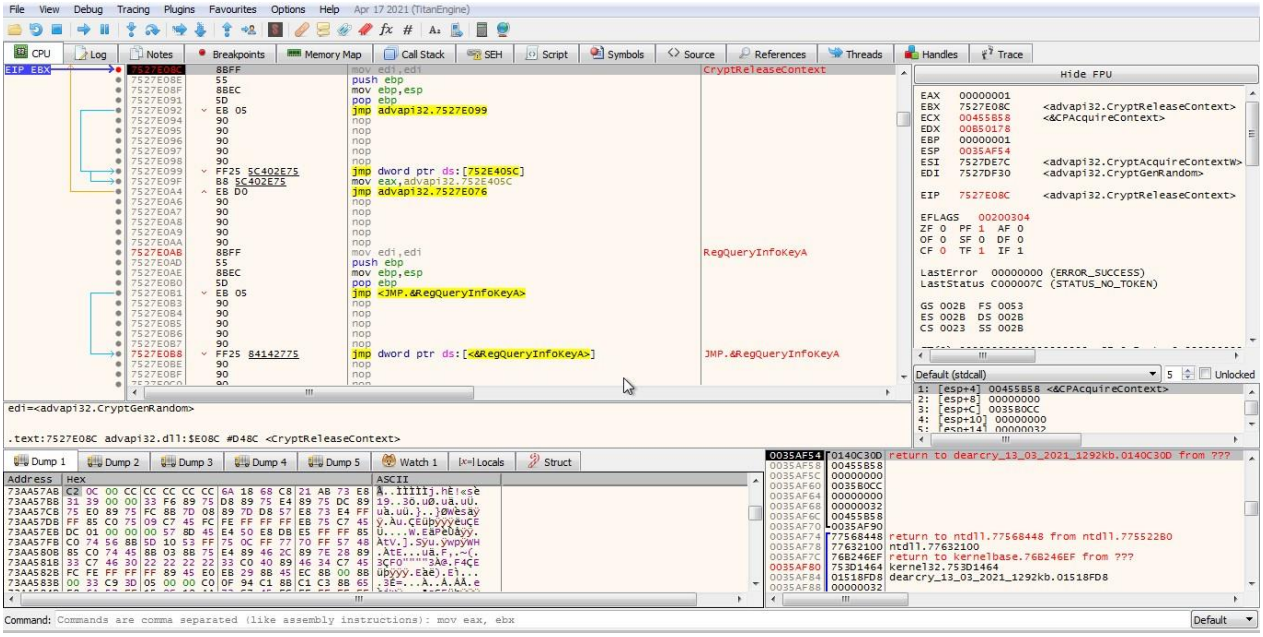

Figure 61

As the next step ransomware reattempts to call the ReadFile function. But this time it access newly created file $ITMEIY8.txt.CRYPT instead of the original file.

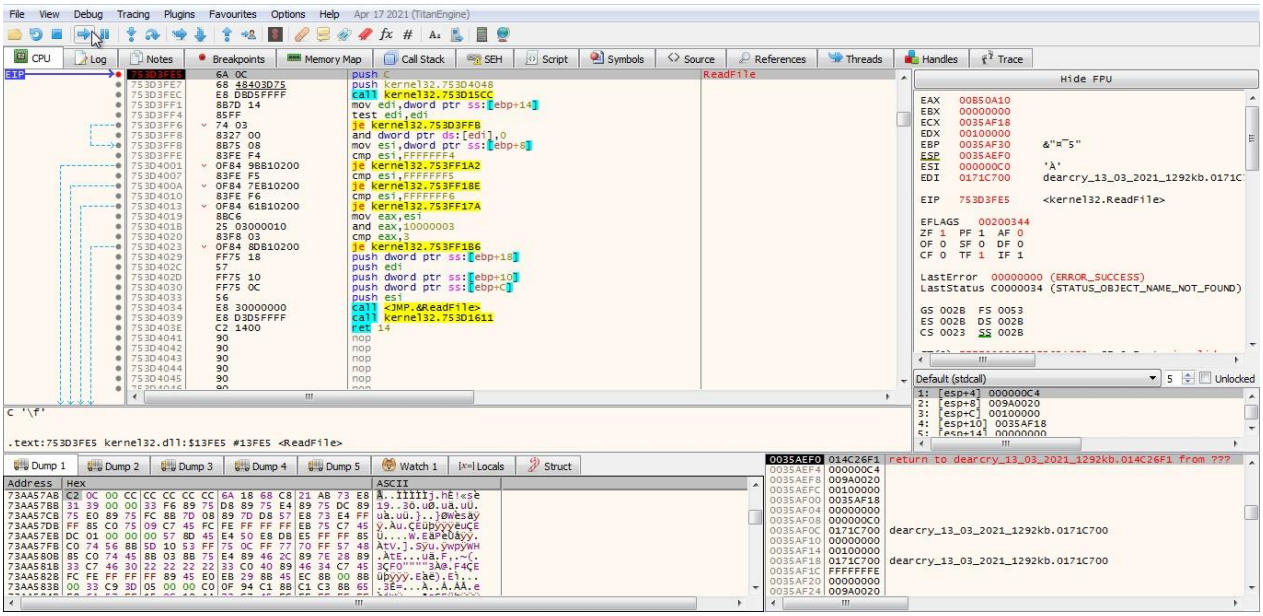

Figure 62

After that it calls the WriteFile function which is again function name itself proclaims the idea about what it does. Before the function execution “DEARCRY!” string is visible in the stack as show in the following figure.

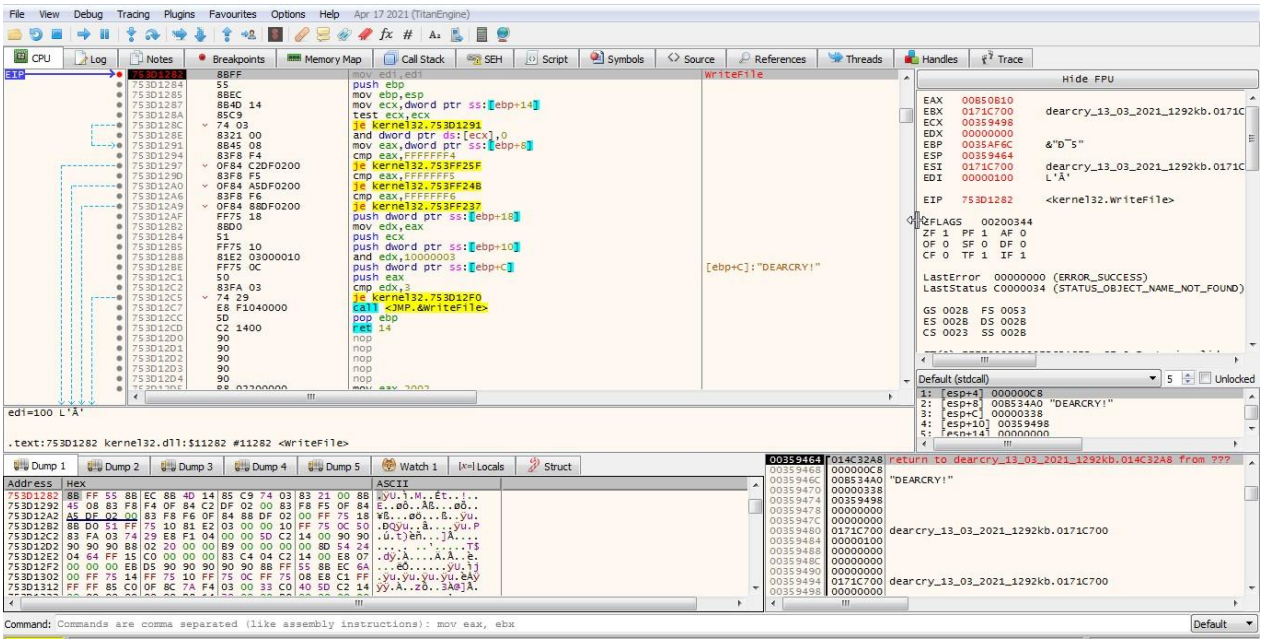

Figure 63

And after the execution of the function aforementioned “DEARCRY!” string has changed to set of “A”s and the snapshot of that is shown below.

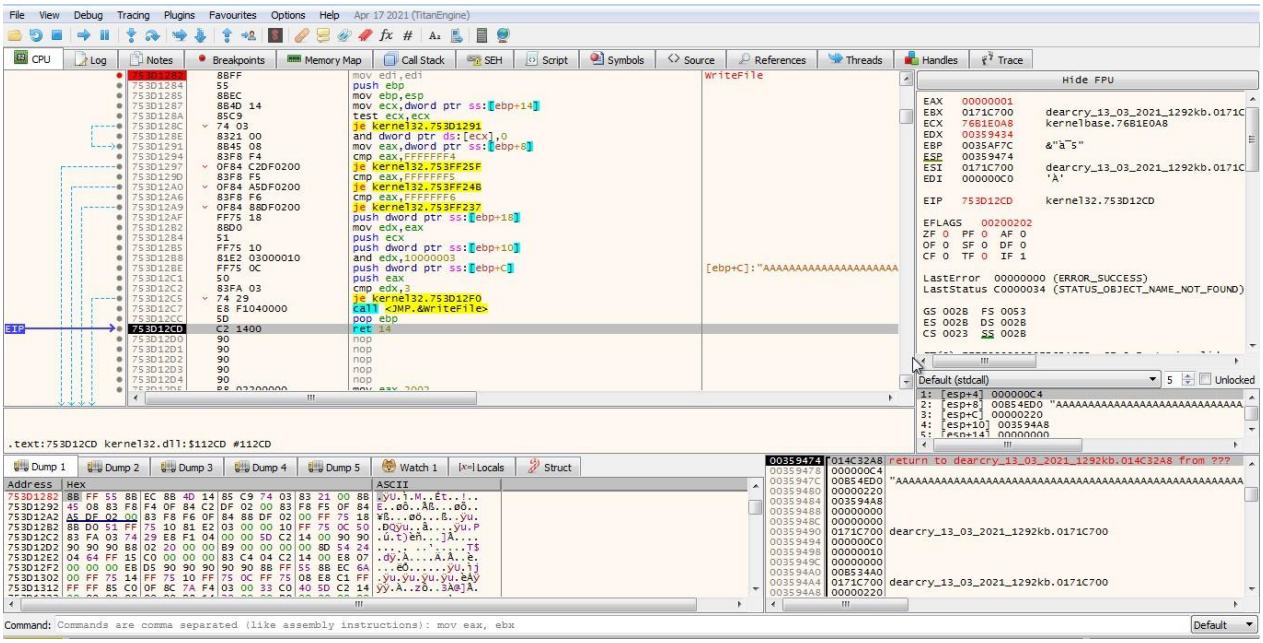

Figure 64

After the function execution some content has been written to the $ITMEIY8.txt.CRYPT file.

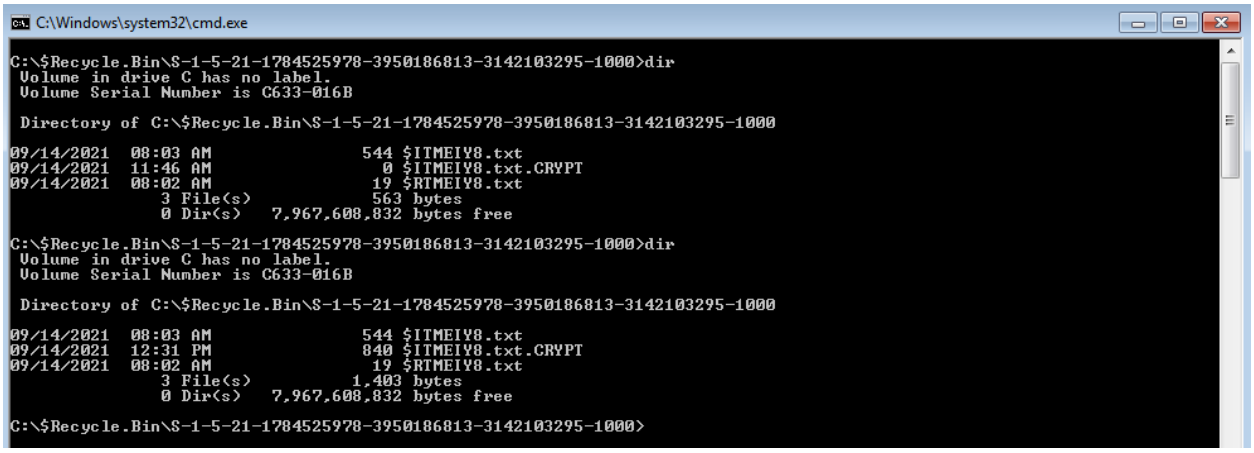

Figure 65

As the next step it calls the DeleteFileA function.

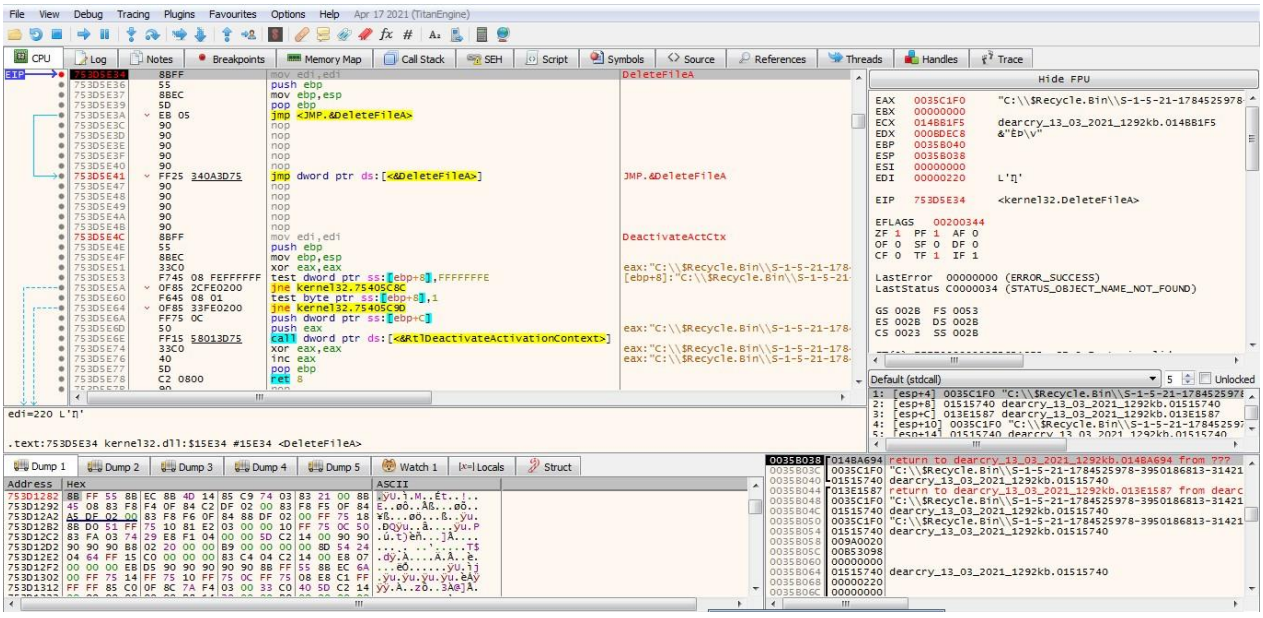

Figure 66

After the function execution it deletes the original file as expected.

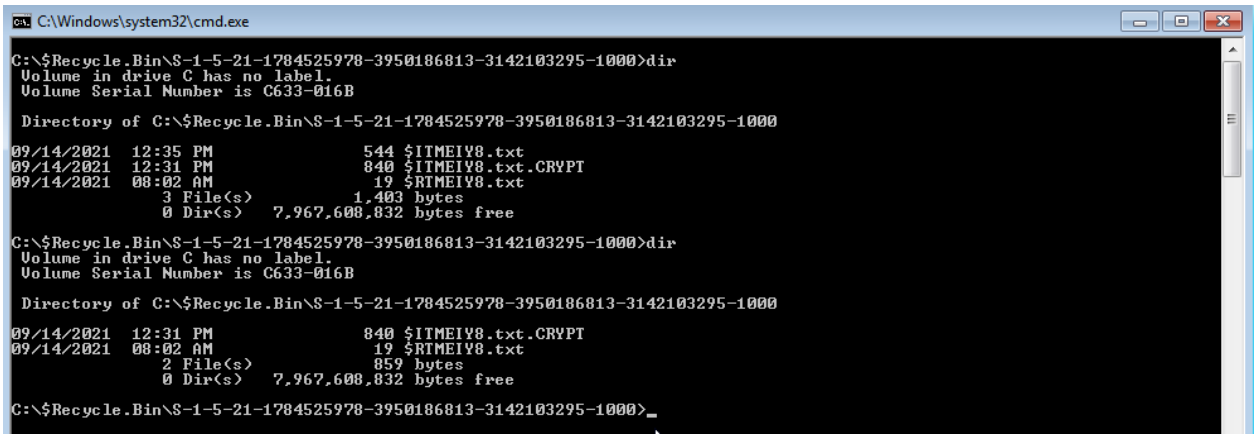

Figure 67

Then Dearcry continues to find other files using FindNextfileA call and repeats previously explained processes until it encrypts all the files in the system.

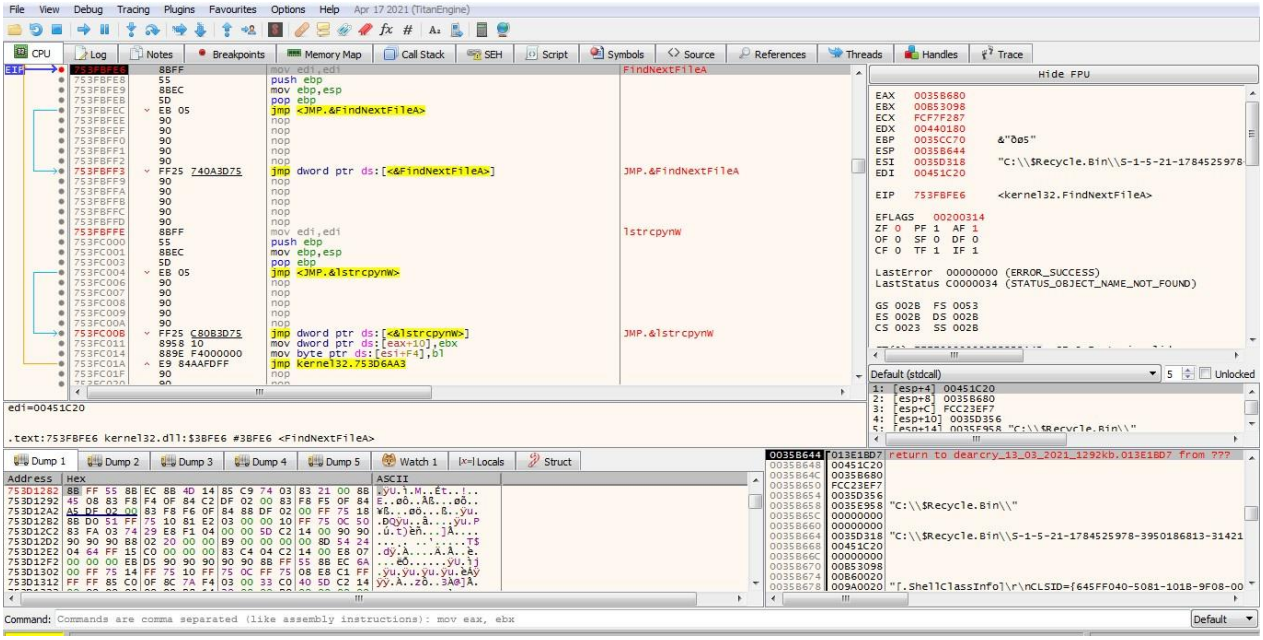

Figure 68

NOTE: After the WriteFile function call it overwrites the original files with set of “A”s before the deletion in order to make sure that recovery of the original file is impossible using any forensic mechanism.

$RTMEIY8.txt is filled with “A”s as shown in the following image.

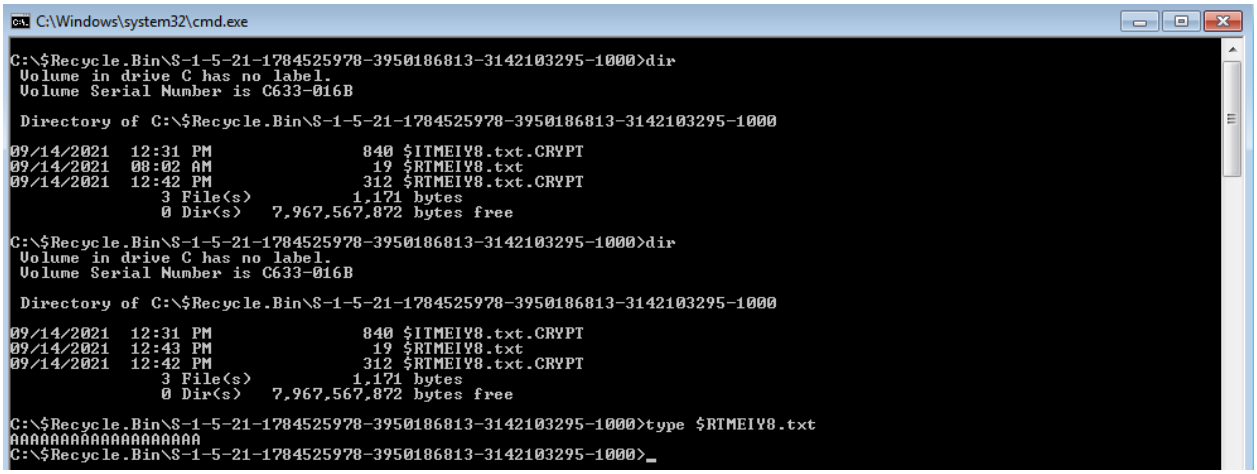

Figure 69

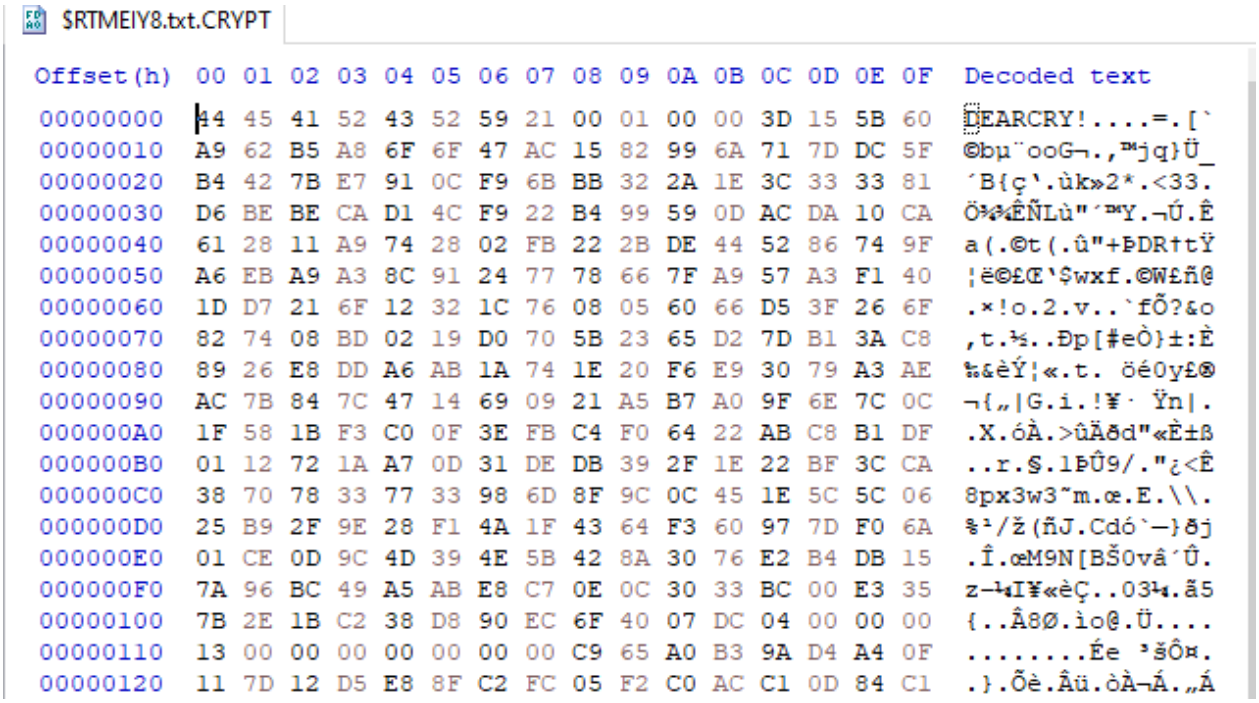

Figure 70

After examining that newly generated encrypted file we were able to find out that it embedded the word “DEARCRY!” in the beginning of the file header which was similar approach taken in Wannacry ransomware encryption process.

12 CONCLUSION

Dearcry also known as DoejoCrypt malware is a new ransomware initially spotted in March 2021 in Microsoft exchange servers. Adversaries who exploited proxy logon vulnerability in exchange servers deliberately sent Dearcry ransomware as a payload to compromise the victims in the post exploitation phase which resulted a severe damage in countries like Indonesia, India the United States, and some parts of European Union. When compared to the other ransomwares that were seen in the world this Dearcry ransomware was considered as a unsophisticated one due to its simple nature. During the analysis process it was found that this wasn’t not packed as expected and had no defensive mechanism against detection. Even though this utilized a hybrid encryption mechanism that initially encrypted the files of the victim machine with AES-256 symmetric encryption and later used RSA-2048 asymmetric algorithm to encrypt the AES key to harden the decryption process. The RSA key was hard coded and found while performing the string analysis. Furthermore list of file extensions was found and the files with those extension were encrypted as expected during the dynamic analysis phase. When it comes to encryption process Dearcry replaced original files with new file which had CRYPT file extension which had the word “DEARCRY!” in the file header and filled the content of the original files with set of “A” s before wiping off the original files making that completely irrecoverable. After completing the encryption process Dearcry ransomware left a readme file stating to contact with two email addresses to obtain necessary instructions to decrypt the content with a unique hash value.