Icon

Description automatically generated

Practical Malware Analysis & Triage

Malware Analysis Report

Ransom.Win32.WNCRY.A

April 2023 | NickT | v1.01

# Table of Contents

[Table of Contents 2](#_Toc132382078)

[Executive Summary 3](#_Toc132382079)

[High-Level Technical Summary 3](#_Toc132382080)

[Malware Composition 4](#_Toc132382081)

[tasksche.exe 5](#_Toc132382082)

[taskdl.exe 6](#_Toc132382083)

[taskse.exe 7](#_Toc132382084)

[@WannaDecryptor@.exe 8](#_Toc132382085)

[Other dropped files 8](#_Toc132382086)

[Basic Static Analysis 10](#_Toc132382087)

[Basic Dynamic Analysis 11](#_Toc132382088)

[Advanced Static Analysis 13](#_Toc132382089)

[Advanced Dynamic Analysis 14](#_Toc132382090)

[Indicators of Compromise 22](#_Toc132382091)

[Network Indicators 22](#_Toc132382092)

[Host-based Indicators 23](#_Toc132382093)

[Appendices 26](#_Toc132382094)

[A. Yara Rules 26](#_Toc132382095)

# Executive Summary

|  |  |
| --- | --- |
| SHA256 hash | ED01EBFBC9EB5BBEA545AF4D01BF5F1071661840480439C6E5BABE8E080E41AA |

The WNCRY ransomware is considered a blended type of malware, as it combines worm and ransomware capabilities for propagation and encryption. This malware sample is compiled in the C++ programming language and runs on x32 Windows operating systems. Its behavior is to drop a file, which then executes its malicious routine, including the payload to drop a ransom note

YARA signature rules are attached in Appendix A.

# High-Level Technical Summary

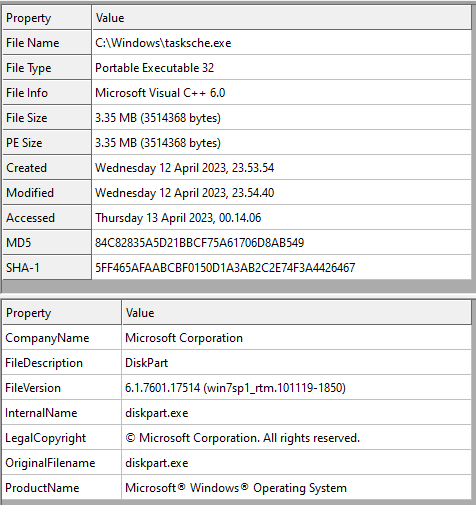
* The malware opened the registry and adding the "***mssecsvc2.0***" key to the registry path "***HKLM\SOFTWARE\ControlSet001\Services***". This malware adds six entries, which are "***DisplayName = Microsoft Security Center (2.0) Service***", "***ErrorControl = 0x00000001***", "***ObjectName = LocalSystem***", "***Start = 0x00000002***", and "***Type = 0x00000010***" also starts mssecsvc2.0 as a service in order to ensure its persistent method. This will drop ***“tasksche.exe”*** and it will execute to perform its malicious routine.
* The file ***tasksche.exe*** also drop several files including ***taskdl.exe, taskse.exe, @WanaDecrypt0r@.exe,*** and numerous files with the .wnry extension, such as ***b.wnry, c.wnry, m\_bulgarian.wnry, m\_chinese (simplified).wnry, m\_chinese (traditional).wnry, m\_croatian.wnry, m\_czech.wnry, m\_danish.wnry, m\_dutch.wnry, m\_english.wnry, m\_filipino.wnry, m\_finnish.wnry, m\_french.wnry, m\_german.wnry, m\_greek.wnry, m\_indonesian.wnry, m\_italian.wnry, m\_japanese.wnry, m\_korean.wnry, m\_latvian.wnry, m\_norwegian.wnry, m\_polish.wnry, m\_portuguese.wnry, m\_romanian.wnry, m\_russian.wnry, m\_slovak.wnry, m\_spanish.wnry, m\_swedish.wnry, m\_turkish.wnry, m\_vietnamese.wnry.*** The malware also drops ***@Please\_Read\_Me@.txt, 130591681393272.bat, 00000000.pky, 00000000.res, and 00000000.eky*** files as part of its malicious routine.
* The file ***tasksche.exe*** encrypts 132 file extensions, including but not limited to: ***.der, .pfx, .key, .crt, .csr, .p12, .pem, .odt, .ott, .sxw, .stw, .uot, .3ds, .max, .3dm, .ods, .ots, .sxc, .stc, .dif, .slk, .wb2, .odp, .otp, .sxd, .std, .uop, .odg, .otg, .sxm, .mml, .lay, .lay6, .asc, .sqlite3, .sqlitedb, .sql, .accdb, .mdb, .dbf, .odb, .frm, .myd, .myi, .ibd, .mdf, .ldf, .sln, .suo, .cpp, .pas, .asm, .cmd, .bat, .ps1, .vbs, .dip, .dch, .sch, .brd, .jsp, .php, .asp, .java, .jar, .class, .mp3, .wav, .swf, .fla, .wmv, .mpg, .vob, .mpeg, .asf, .avi, .mov, .mp4, .3gp, .mkv, .3g2, .flv, .wma, .mid, .m3u, .m4u, .djvu, .svg, .psd, .nef, .tiff, .tif, .cgm, .raw, .gif, .png, .bmp, .jpg, .jpeg, .vcd, .iso, .backup, .zip, .rar, .tgz, .tar, .bak, .tbk, .bz2, .PAQ, .ARC, .aes, .gpg, .vmx, .vmdk, .vdi, .sldm, .sldx, .sti, .sxi, .602, .hwp, .snt, .onetoc2, .dwg, .pdf, .wk1, .wks, .123, .rtf, .csv, .txt, .vsdx, .vsd, .edb, .eml, .msg, .ost, .pst, .potm, .potx, .ppam, .ppsx, .ppsm, .pps, .pot, .pptm, .pptx, .ppt, .xltm, .xltx, .xlc, .xlm, .xlt, .xlw, .xlsb, .xlsm, .xlsx, .xls, .dotx, .dotm, .dot, .docm, and .docb.***

# Malware Composition

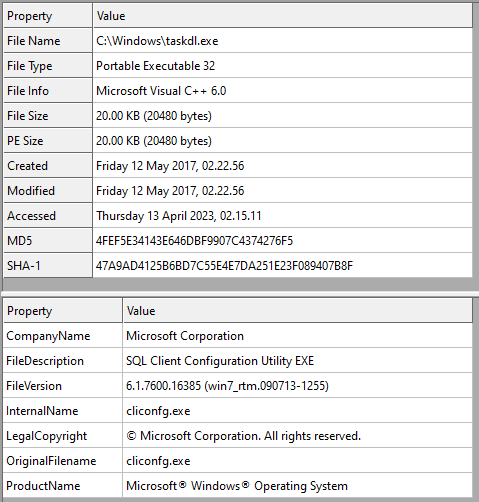
The WNCRY ransomware consists of the following components. CFF explorer and pestudio is used for static analysis to obtain initial information about the file.

|  |  |
| --- | --- |
| File Name | SHA256 Hash |
| tasksche.exe | ED01EBFBC9EB5BBEA545AF4D01BF5F1071661840480439C6E5BABE8E080E41AA |
| taskdl.exe | 4A468603FDCB7A2EB5770705898CF9EF37AADE532A7964642ECD705A74794B79 |
| taskse.exe | 2CA2D550E603D74DEDDA03156023135B38DA3630CB014E3D00B1263358C5F00D |
| @WanaDecryptor@.exe | B9C5D4339809E0AD9A00D4D3DD26FDF44A32819A54ABF846BB9B560D81391C25 |

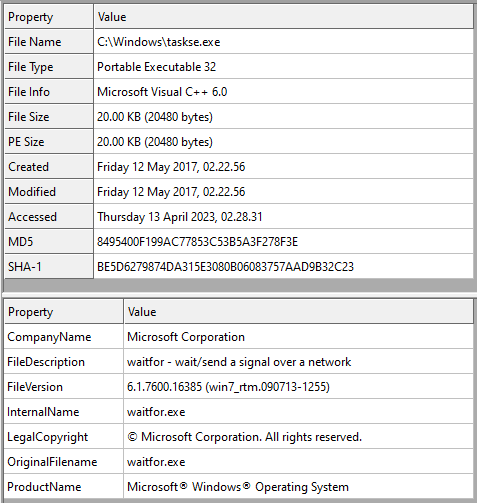
## tasksche.exe



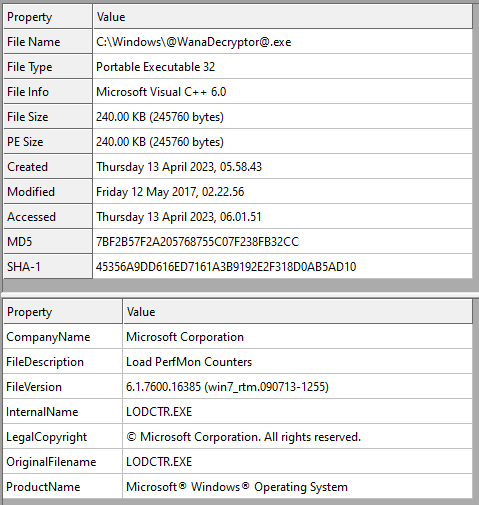
## taskdl.exe



## taskse.exe



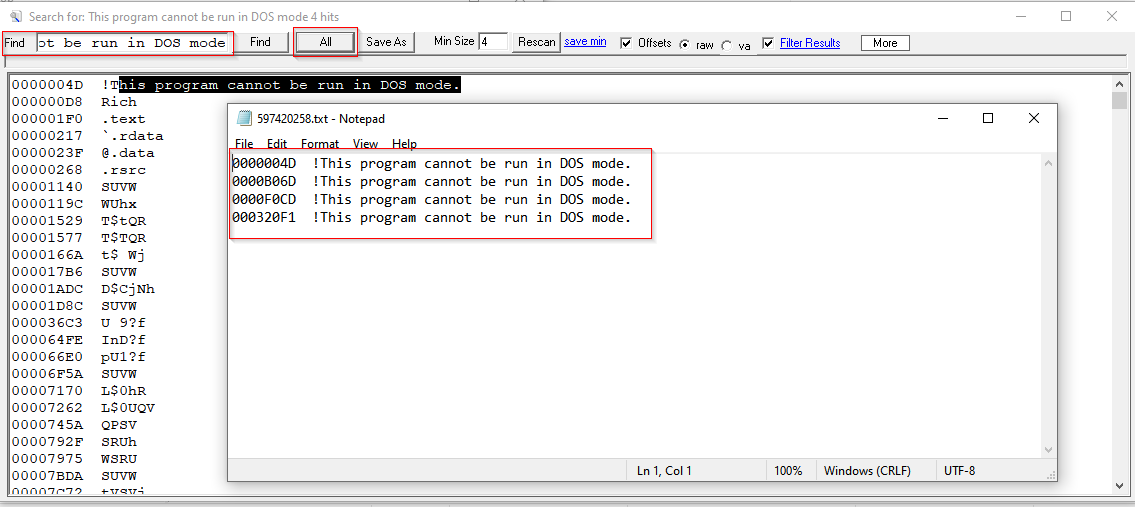
## @WannaDecryptor@.exe



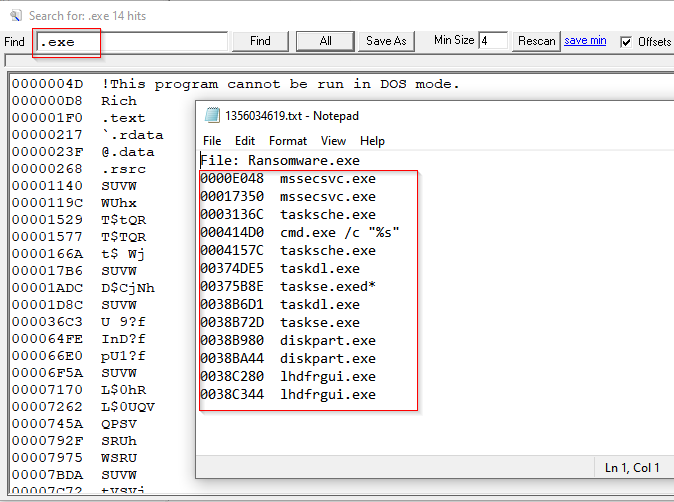
## Other dropped files

|  |  |  |
| --- | --- | --- |
| File Name | MD5 | Other details |
| b.wnry | c17170262312f3be7027bc2ca825bf0c |  |
| c.wnry | ae08f79a0d800b82fcbe1b43cdbdbefc | Link for TOR browser |
| m\_bulgarian.wnry | 95673b0f968c0f55b32204361940d184 |  |
| m\_chinese (simplified).wnry | 0252d45ca21c8e43c9742285c48e91ad |  |
| m\_chinese (traditional).wnry | 2efc3690d67cd073a9406a25005f7cea |  |
| m\_croatian.wnry | 17194003fa70ce477326ce2f6deeb270 |  |
| m\_czech.wnry | 537efeecdfa94cc421e58fd82a58ba9e |  |
| m\_danish.wnry | 2c5a3b81d5c4715b7bea01033367fcb5 |  |
| m\_dutch.wnry | 7a8d499407c6a647c03c4471a67eaad7 |  |
| m\_english.wnry | fe68c2dc0d2419b38f44d83f2fcf232e |  |
| m\_filipino.wnry | 08b9e69b57e4c9b966664f8e1c27ab09 |  |
| m\_finnish.wnry | 35c2f97eea8819b1caebd23fee732d8f |  |
| m\_french.wnry | 4e57113a6bf6b88fdd32782a4a381274 |  |
| m\_german.wnry | 3d59bbb5553fe03a89f817819540f469 |  |
| m\_greek.wnry | fb4e8718fea95bb7479727fde80cb424 |  |
| m\_indonesian.wnry | 3788f91c694dfc48e12417ce93356b0f |  |
| m\_italian.wnry | 30a200f78498990095b36f574b6e8690 |  |
| m\_japanese.wnry | b77e1221f7ecd0b5d696cb66cda1609e |  |
| m\_korean.wnry | 6735cb43fe44832b061eeb3f5956b099 |  |
| m\_latvian.wnry | c33afb4ecc04ee1bcc6975bea49abe40 |  |
| m\_norwegian.wnry | ff70cc7c00951084175d12128ce02399 |  |
| m\_polish.wnry | e79d7f2833a9c2e2553c7fe04a1b63f4 |  |
| m\_portuguese.wnry | fa948f7d8dfb21ceddd6794f2d56b44f |  |
| m\_romanian.wnry | 313e0ececd24f4fa1504118a11bc7986 |  |
| m\_russian.wnry | 452615db2336d60af7e2057481e4cab5 |  |
| m\_slovak.wnry | c911aba4ab1da6c28cf86338ab2ab6cc |  |
| m\_spanish.wnry | 8d61648d34cba8ae9d1e2a219019add1 |  |
| m\_swedish.wnry | c7a19984eb9f37198652eaf2fd1ee25c |  |
| m\_turkish.wnry | 531ba6b1a5460fc9446946f91cc8c94b |  |
| m\_vietnamese.wnry | 8419be28a0dcec3f55823620922b00fa |  |
| s.wnry | ad4c9de7c8c40813f200ba1c2fa33083 | Zip file contains TOR files |
| t.wnry | 5dcaac857e695a65f5c3ef1441a73a8f | Random encrypted stirngs |
| uwnry | 7bf2b57f2a205768755c07f238fb32cc |  |
| 00000000.pky | 6aaeb86593975cc17765786a922cf5dc |  |
| 00000000.res | 22a0f6a881d0cf222f78a9d2a18e59df |  |
| @Please\_Read\_Me@.txt | 7e6b6da7c61fcb66f3f30166871def5b | Ransom note (Instruction for the victim) |
| 00000000.eky | fa683ed9c434ad71c12c7b85c1c71590 |  |
| 130591681393272.bat | d47c2fdbbec44ff8a2c7f20c480d5b78 | Script contains vbs code |

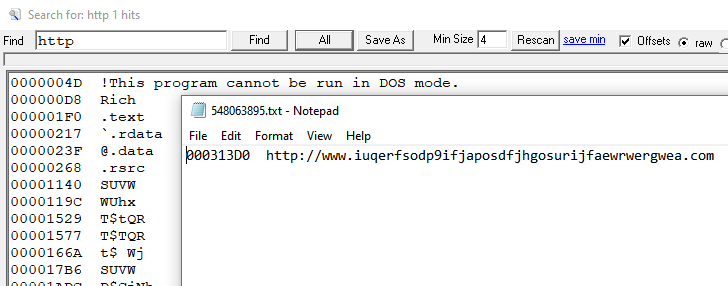
# Basic Static Analysis



During the static analysis of the ransomware.exe, the strings were examined using ShellEx. Four instances of the message 'This program cannot be run in DOD mode' were observed. This message is typically indicative of a program designed to be run as a Windows application, which requires a Windows operating system to function properly. The presence of these strings provided insight into the existence of four executable files within the ransomware.

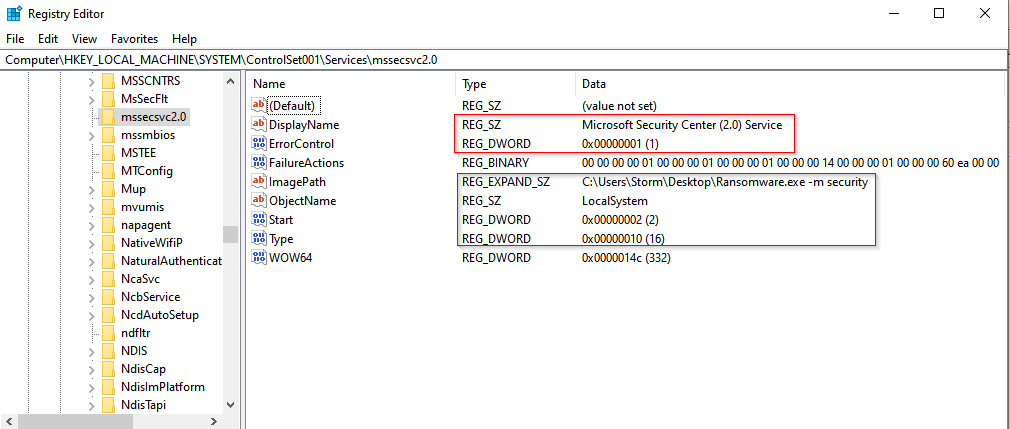


For further analysis, the strings of the ransomware.exe were filtered with “.exe” to identify executable extensions. This process revealed 13 such extensions, including one notable instance of "cmd.exe /c '%s'". This suggests that the ransomware may attempt to run various commands during its execution.

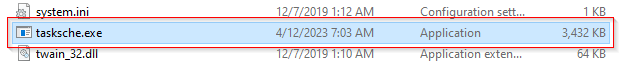


Another notable discovery during the analysis of the ransomware.exe was a link that was found: "http[:]//www[.]iuqerfsodp9ifjaposdfjhgosurijfaewrwergwea[.]com" (*masked with []*).

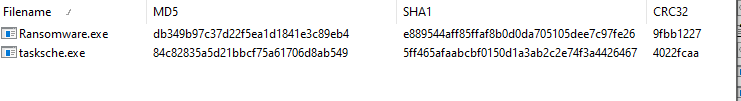
# Basic Dynamic Analysis



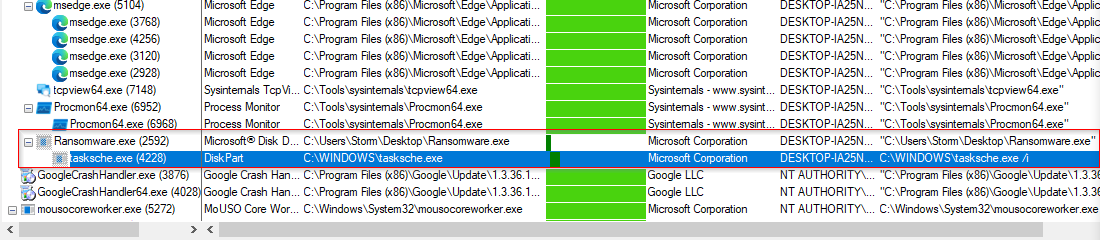
During the basic dynamic analysis, procmon was used to monitor the malware's execution. The analysis revealed that the ransomware created five registry entries, each with the registry key "mssecsvc2.0". These entries were placed in the HLKM\SYSTEM\ControlSet001\Services\ registry path, and the entries themselves were also placed under the "mssecsvc2.0" key. This tactic of adding registry entries is commonly used by malware to establish persistence on an infected system, ensuring that the malware is executed each time the system is started or restarted. In this case, the use of the "mssecsvc2.0" may be designed to masquerade as a legitimate Windows service, potentially to evade detection by security software.



Further investigation, it was revealed that the malware drops an executable file named "tasksche.exe" in the %Windows%\ directory. The purpose of this dropped file is not immediately clear, but it is possible that it may be used by the ransomware to execute various tasks during its operation.



To further investigate the role and function of the "tasksche.exe" fil, HasMyFile was used to check its MD5 and compare it with the MD5 of the ransomware executable file. The results confirmed that the MD5 of the dropped file was different from that of the ransomware executable, suggesting that the two files have different functions.

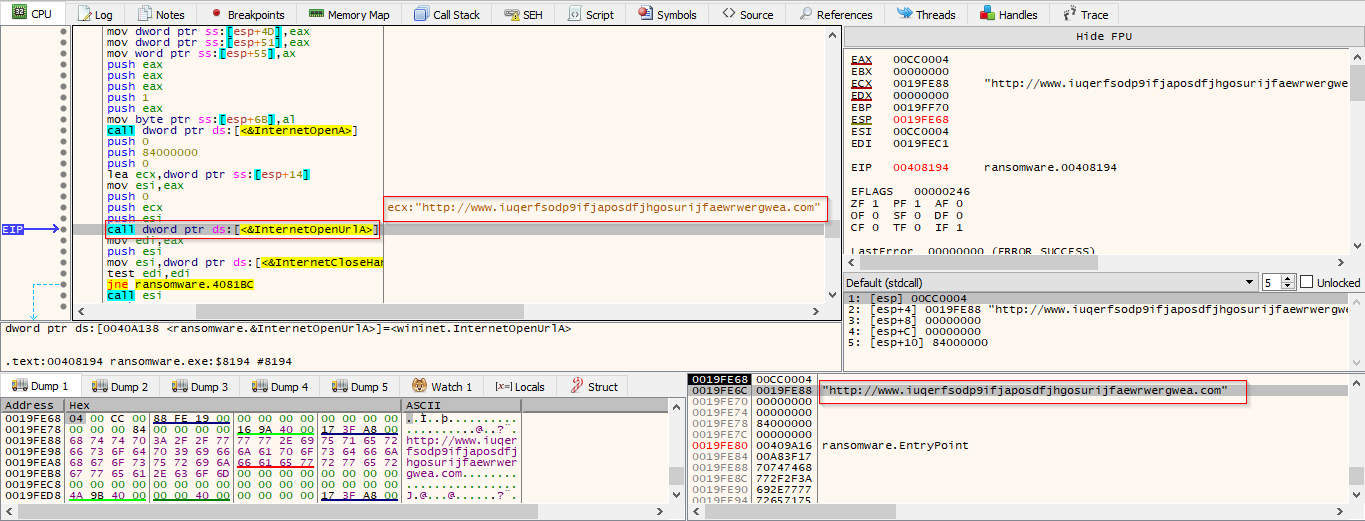


During the execution, it was observed in procmon that the malware creates a child process named "tasksche.exe". This is the same dropped file that was earlier identified as being present in the %Windows%\ directory.

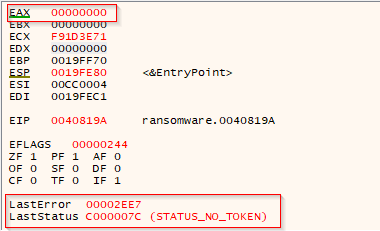
# Advanced Static Analysis

For advanced static analysis, the Cutter tool was used to analyze its main function. By examining the code at address 0040814a, it was observed that the malware attempts to move a link with the format "http[:]//www[.]iuqerfsodp9ifjaposdfjhgosurijfaewrwergwea[.]com" into the ESI register.

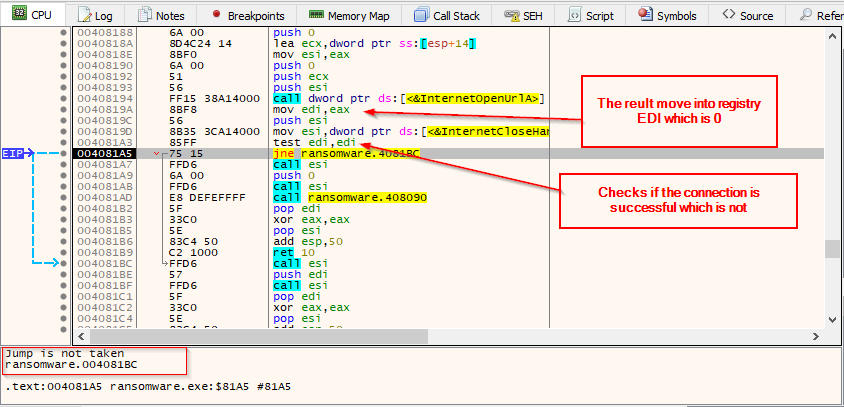
# Advanced Dynamic Analysis



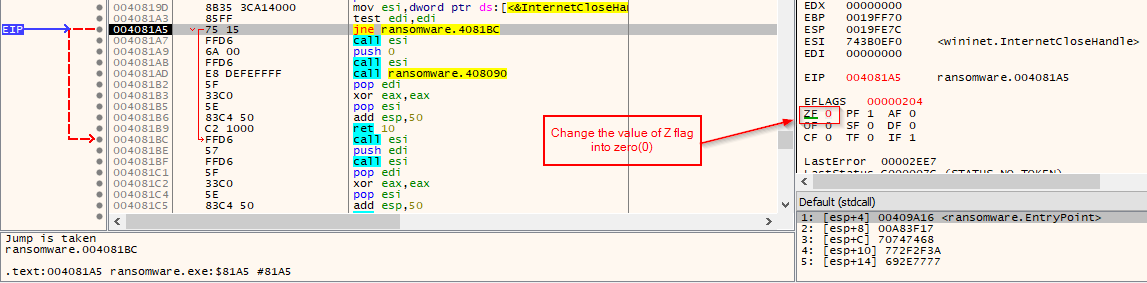
In order to perform further analysis of the malware, xdbg was used to debug the sample. Upon examination of the code, it was observed that after calling the InternetOpenA API, the malware pushes some parameters of the InternetOpenUrlA API, including the URL "http[:]www[.]iuqerfsodp9ifjaposdfjhgosurijfaewrwergwea[.]com" (*mask with []*), which is passed as the second parameter, stored in the ECX register. This suggests that the malware attempts to connect to the URL "http[:]www[.]iuqerfsodp9ifjaposdfjhgosurijfaewrwergwea[.]com".



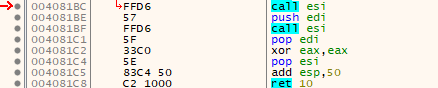
Continuing the analysis using xdbg, the program was stepped over using the F8 key. Upon examining the registry, it was observed that after attempting to connect to the URL mentioned earlier, the value stored in the EAX registry was 0, indicating that the connection was not successful. This suggests that the malware was not able to establish a connection with the URL "http[:]www[.]iuqerfsodp9ifjaposdfjhgosurijfaewrwergwea[.]com”.



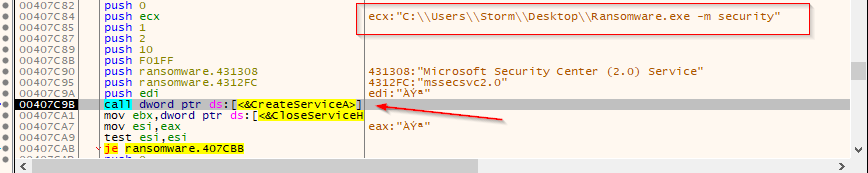
Notice here that the malware move the result of connection into registry EDI. After that it calls the test function to check if the result is successful or not and if the test is not successful it will not jump as we can see in the EIP which says “Jump is not taken”.



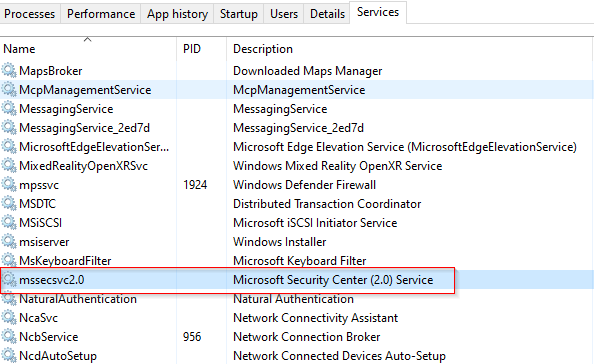
By changing the value of the Z flag to force the malware to take a different path to reveal additional information about the malware's behavior.



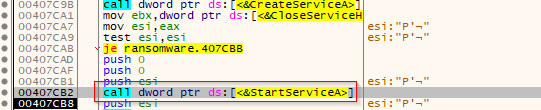
Based on this screenshot, this path appears that the malware attempts to terminate its operation. This suggest that malware connect to a specific URL and checks whether the connection is successful or not. If the connection is not successful, the malware proceeds with its execution and continues its malicious activities. However, if the connection is successful, the malware terminates its execution. It appears that the URL "http[:]//www[.]iuqerfsodp9ifjaposdfjhgosurijfaewrwergwea[.]com" serves as the killswitch.



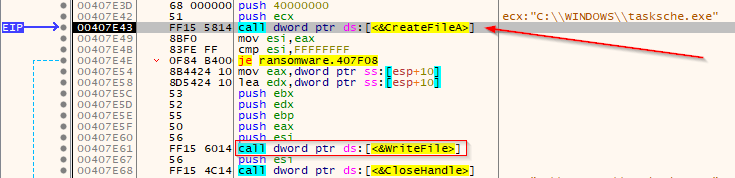
Another API used by the malware was 'CreateServiceA', and it pushed the parameter 'mssecsvc2.0' from address 407c90 with its service description “Microsoft Security Center (2.0) Service”.



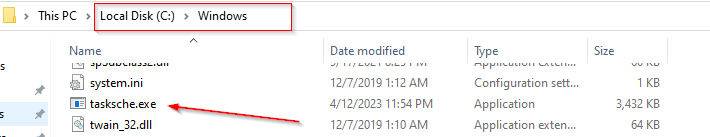
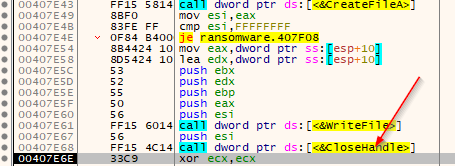
As observed the service is created right after the execution of API CreateServiceA



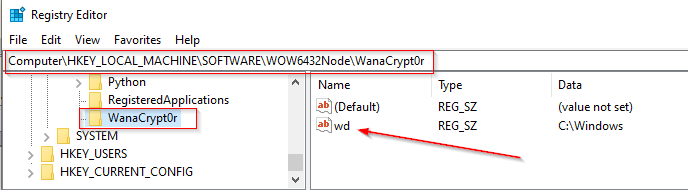
After the creation of service the malware call the API StartServiceA to execute the mssecsvc2.0



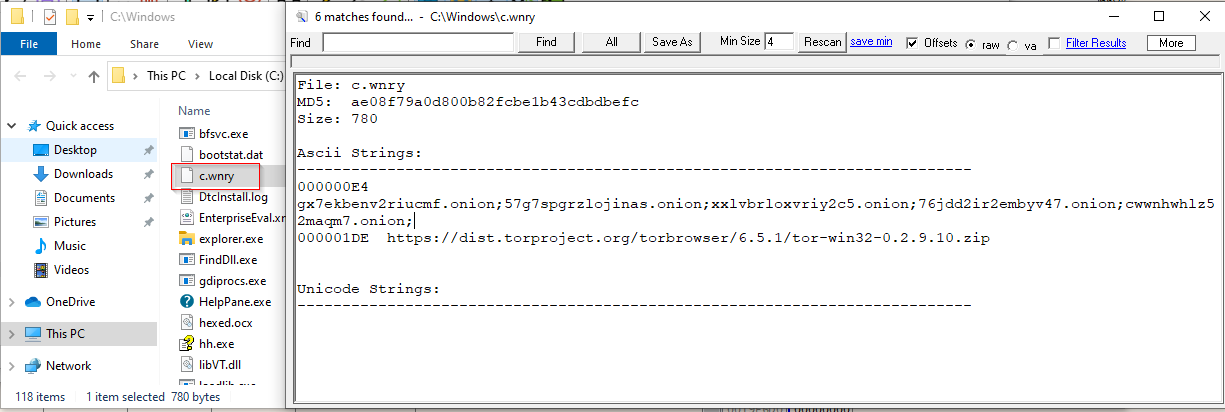
Upon further analysis, it can be observed that the malware uses three API calls for the creation of the file tasksche.exe. The first API call is CreateFileA, which has a parameter of C:\WINDOWS\tasksche.exe. This is the directory where the file will be placed. The second API call is WriteFile, which is used to write the malicious code to the file. Lastly, the malware uses the API call CloseHandle to exit its write mode.



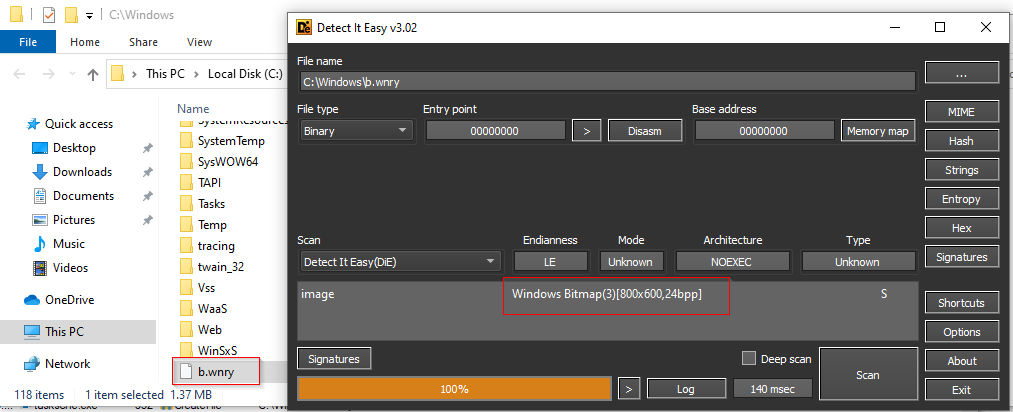
As observed here that after stepping over from CloseHandle the tasksche.exe is fully created and placed in the %Windows% directory. For in-dept understanding, x32dbg is used to debug the dropped file “tasksche.exe”.



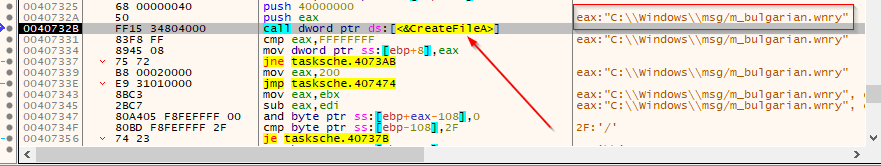
It appears that the dropped file, tasksche.exe, creates a registry key called WanaCrypt0r under HKLM\SOFTWARE\WOW6432NODE\ with an entry named "wd" and a value of "C:\Windows".

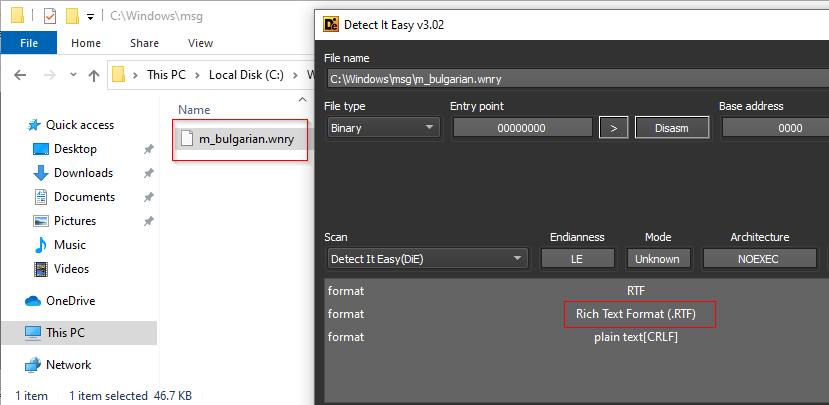


The file "c.wnry" dropped by the malware contains a link to download the Tor browser, which may be used by the malware for anonymous communication or to access the dark web.

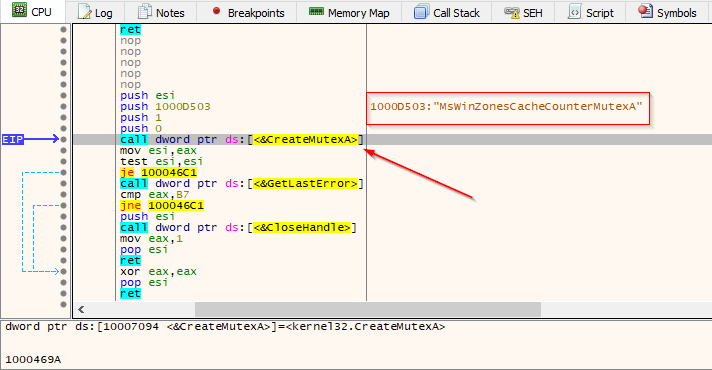


Upon further analysis of the tasksche.exe dropped file, it was found to also drop a file named "b.wncry". Upon examination of this file using Detect It Easy, it was determined that the file is likely related to images and may be a bitmap file.

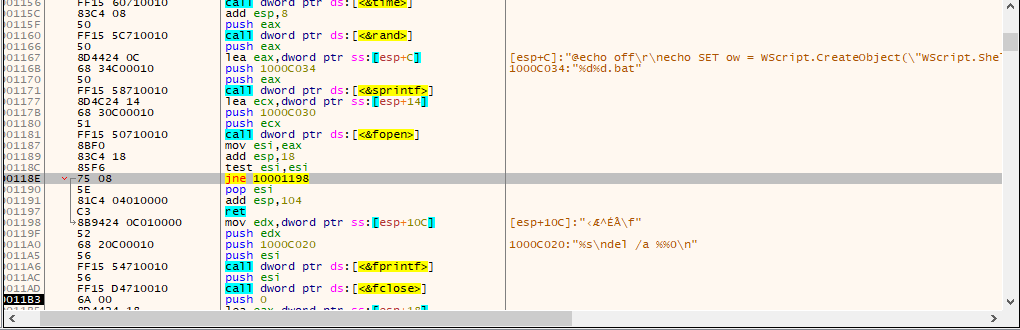




The malware drops a file named "m\_bulgarian.wnry" in a folder named "msg" located in the C:\Windows\ directory. Upon analyzing the file using Detect It Easy, it appears to be in Rich Text File format with unorganized content.



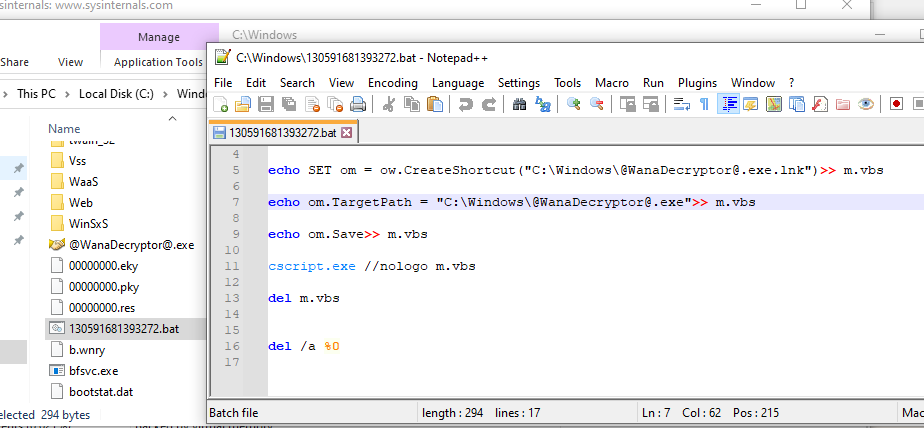
In this case, the malware is creating a mutex object with the name "MsWinZonesCacheCounterMutexA", which is likely unique to this particular malware, and will use this to ensure that only one instance of itself is running on the infected system.

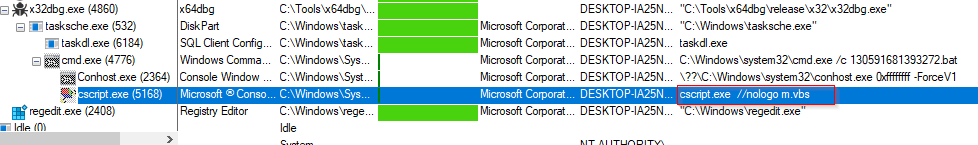


The malware drops a random file name with an extension of “.bat” using the API fopen, fprintf, fclose.

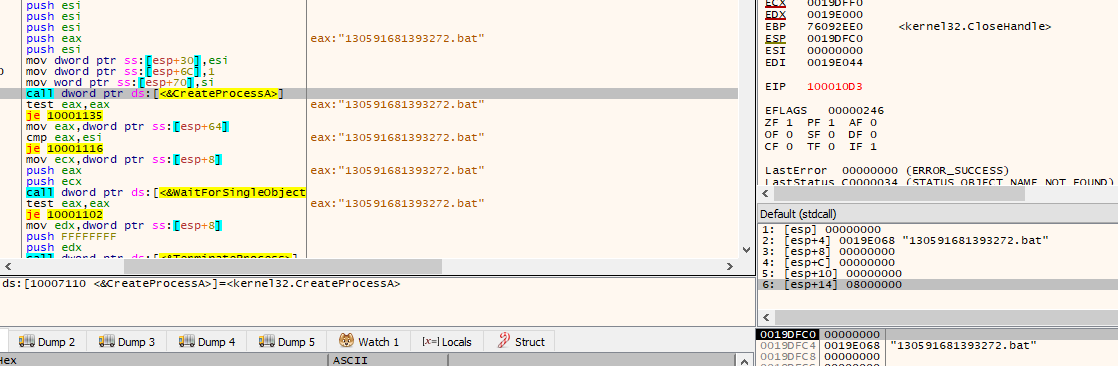


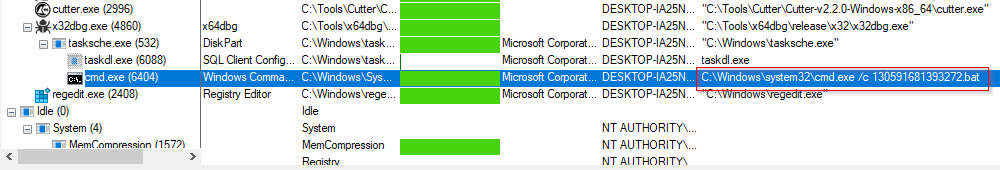
As observed in procmon





The batch file contains a VBScript that will dump its content and save it as "m.vbs". After creating this file, the malware runs the dump file by executing m.vbs. We can verify this behavior by checking Procmon process tree.



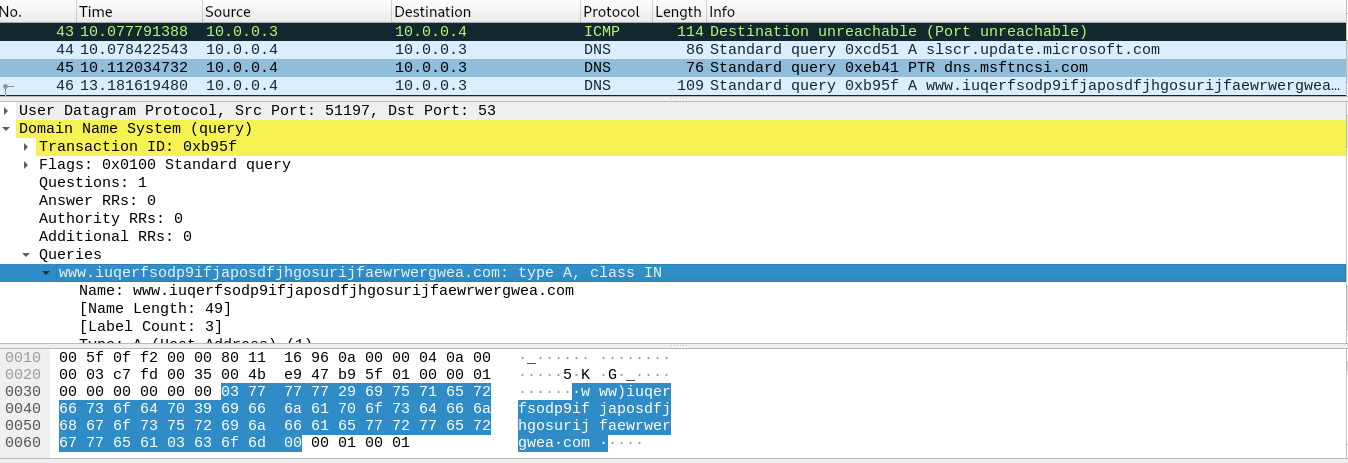


As seen in the code, the malware initiates the API CreateProcessA to execute the batch file that was created earlier, named “130591681393272.bat”.

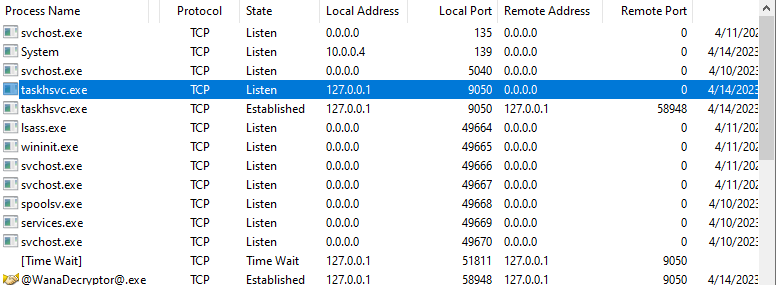
# Indicators of Compromise

The full list of IOCs can be found in the Appendices.

## Network Indicators

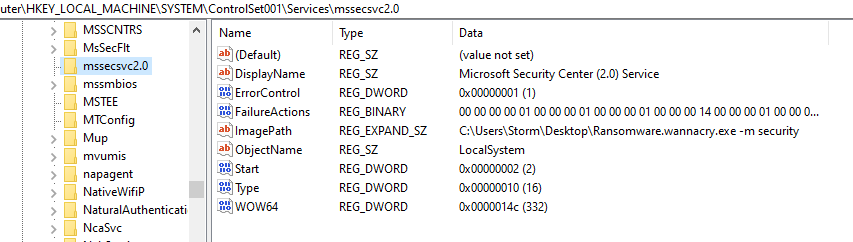


*Fig 1: Query to specified domain*

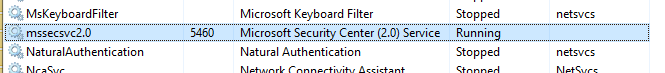


*Fig 2: Opens listening port*

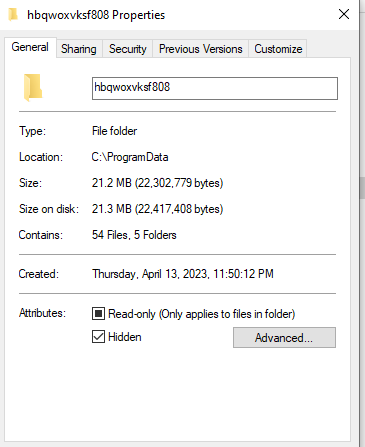
## Host-based Indicators



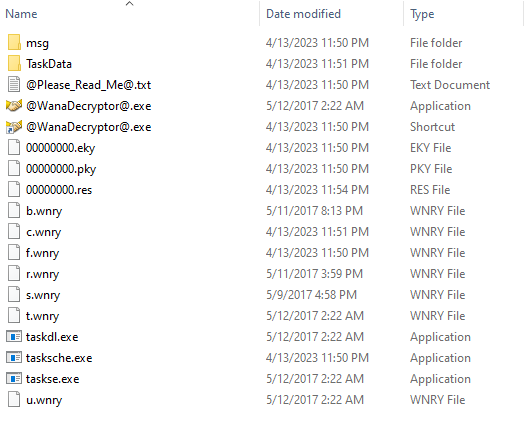
*Fig 1: Registry added*



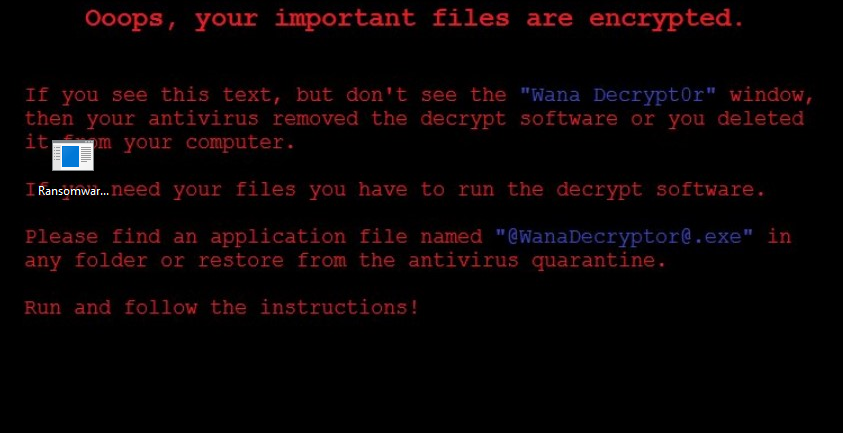
*Fig 2: Service execution*



*Fig 3: Directory creation*



*Fig 4: Dropped files*



*Fig 5: Background wallpaper modification*



*Fig 6: Display ransom window*

# Appendices

## Yara Rules

rule wncry\_rules {

    meta:

        last\_updated = "2023-04-14"

        author = "4E 69 63 6B 54"

        description = "Yara rules for WannaCry ransomware detection"

    strings:

        // Fill out identifying strings and other criteria

        $PE\_magic\_byte = "MZ"

        $UrlLink = "http:www.iuqerfsodp9ifjaposdfjhgosurijfaewrwergwea.com" ascii

        $PersistenceMethod = "mssecsvc2.0"

        $cmdExecution = "cmd.exe /c "%s""

        $ExecutableEncryptor = "tasksche.exe"

        $Executable1 = "taskdl.exe"

        $Executable2 = "taskse.exe"

        $IpAddress = "172.16.99.5"

        $IpAddress2 = "192.168.56.20"

        $StringCryptor = "WanaCrypt0r"

        $EncryptedExtension = ".doc"

    condition:

        // Fill out the conditions that must be met to identify the binary

        $PE\_magic\_byte at 0 and

        any of them

}