NCCIC NATIONAL CYBERSECURITY AND COMMUNICATIONS INTEGRATION CENTER | US-CERT UNITED STATES COMPUTER EMERGENCY READINESS TEAM

Malware Initial Findings Report (MIFR) - 10130295

2017-06-30

Notification

This report is provided "as is" for informational purposes only. Department of Homeland Security (DHS) does not provide any warranties of any kind regarding any information contained within. The DHS does not endorse any commercial product or service referenced in this bulletin or otherwise.

This document is marked Traffic Light Protocol (TLP):WHITE. Disclosure is not limited. Sources may use TLP:WHITE when information carries minimal or no foreseeable risk of misuse, in accordance with applicable rules and procedures for public release. Subject to standard copyright rules, TLP:WHITE information may be distributed without restriction. For more information on TLP, see https://www.us-cert.gov/tlp.

Summary

Description

Submission included one unique file. This file is a dynamic-link library (DLL) designed to appear as ransomware, but which is, in effect, a data wiper. During runtime, this application attempts to steal victim's credentials and use them to spread laterally on a compromised network. The malware also utilizes the EternalBlue Server Message Block (SMB) exploit for lateral movement.

The malware encrypts the victim's files using a dynamically generated 128-bit key. The malware will encrypt the Master File Table (MFT) of the victim's hard drive with a dynamically generated encryption key. The algorithm utilized for the MFT encryption has been identified as a modified version of the Salsa20 cipher.

The malware generates and provides a unique ID to the victim. This ID could be used by the attacker to decrypt the victim's files if the victim pays a ransom. However, there is no evidence of a relationship between this unique ID and the dynamically generated Salsa20 encryption key. Therefore, it does not appear possible for the attacker to decrypt the victim's files even if the ransom is paid.

Files

Processed	2 71b6a493388e7d0b40c83ce903bc6b04 (71b6.dll) 7e37ab34ecdcc3e77e24522ddfd4852d (e84b.tmp)

Files

71b6.dll

Details	
Name	71b6.dll
Size	362360
Type	PE32 executable (DLL) (console) Intel 80386, for MS Windows
MD5	71b6a493388e7d0b40c83ce903bc6b04
SHA1	34f917aaba5684fbe56d3c57d48ef2a1aa7cf06d
ssdeep	6144:y/Bt80VmNTBo/x95ZjAetGDN3VFNq7pC+9OqFoK30b3ni5rdQY/CdUOs2:y/X4NTS/x9jNG+w+9OqFoK323qdQYKUG
Entropy	7.80193433518

Antivirus	
McAfee	RDN/Ransomware
K7	Trojan (0001140e1)
Systweak	trojan.ransom-petya
F-secure	Trojan:W32/Petya.F
Cyren	W32/Trojan.HEJY-4904
Symantec	Ransom.Petya
VirusBlokAda	TrojanRansom.Filecoder
ClamAV	Win.Exploit.CVE_2017_0147-6331310-0
Kaspersky	Trojan-Ransom.Win32.ExPetr.a
BitDefender	Trojan.Ransom.GoldenEye.B
Microsoft Security Essentials	Ransom:Win32/Petya
Sophos	Troj/Ransom-EOB
TrendMicro House Call	Ransom_.EA1AD694
TrendMicro	Ransom_.EA1AD694
Emsisoft	Trojan.Ransom.GoldenEye.B (B)
Avira	TR/Ransom.ME.12
Ahnlab	Trojan/Win32.Petya
ESET	Win32/Diskcoder.C trojan
NANOAV	Trojan.Win32.Petya.eqlcgp
Filseclab	W32.Diskcoder.C.nmvl
Vir.IT eXplorer	Trojan.Win32.CryptoPetya.F
Quick Heal	Ransom.Petya.A5
Ikarus	Trojan-Ransom.Petrwrap
AVG	SCGeneric2.BGFY

PE Information	
Compiled	2017-06-18T07:14:36Z

PE Sections

Name	MD5	Raw Size	Entropy
(header)	5d542908cdd140cab21a15c0430883f5	1024	2.62946685393
.text	c5bd3bb710ae377938b17980692b785b	48640	6.54653060932
.rdata	46418e52b546c1f696eb8a524f18c56e	34304	6.99212929533
.data	5216f0c62d1fd41b1d558e129e18d0fe	20992	5.42698913823
.rsrc	f07e68575f50a62382d99e182baa05d5	247808	7.9982879669
.reloc	c5d1d4cdade7dcfbe14ec10dcf66cfb1	3584	4.77168126134

Description

This file is a variant of ransomware that spreads via two primary methods. It uses the EternalBlue SMB exploit, and it attempts to steal the user's credentials via the Windows protected storage area. The malware will then attempt to use the stolen credentials to gain access to other systems on the local network. During runtime, the malware attains a file name within the user's TEMP path using the APIs

GetTempPathW and GetTempFileNameW.

Next, the malware will write an embedded resource, named “1,” to this file. The malware will execute this dropped module, and create a named pipe to it in memory. Analysis indicates this dropped application is a modified version of the open-source tool known as “Mimikatz.” The ransomware will execute this modified Mimikatz binary using a command line argument, which will allow the malware to control the Mimikatz binary through a named pipe in virtual memory. The Mimikatz binary has been modified to prevent it from operating normally through STDOUT via the command line. Mimikatz is a tool that can be used to attain username and password credentials from a Windows system. In this case, it is likely the malware uses these stolen credentials to spread laterally on a network. However, this activity was not directly observed within the lab environment.

During runtime, the malware begins enumerating network resources accessible by the user by using the WNetOpenEnumW and WNetEnumResourceW Windows APIs. The malware will also search the protected storage area of the user’s profile looking for Terminal Server credentials. Specifically, it uses the API CredEnumerateW to enumerate the credentials stored in the protected storage area looking for a username that begins with the string “TERMSRV/.” If credentials are found, the malware attempts to establish a connection to this Terminal Server using the API WNetAddConnection2. The malware attempts to login to the network resource using the Terminal Server credentials it attained from the protected storage area. Specifically, the malware attempts to access the $admin share on this network resource.

If a connection to the server is successfully established, the malware will copy itself to the C:\Windows folder of the newly compromised server. The malware will attempt to run the newly installed copy of itself on that network resource, using the Windows Management Instrumentation Command (WMIC) - line. It attempts to connect to the server using WMIC with the following command:

--Begin Command--

<%s /node:"%ws" /user:"%ws" /password:"%ws" >

--End Command--

The malware will then attempt to execute itself on the remote server using the following WMIC command:

--Begin Command--

process call create "C:\Windows\System32\rundll32.exe \ C:\Windows\%s\" #1

--End Command--

During runtime, the malware generates a new 128-bit Advanced Encryption Standard (AES) key and encrypts the user’s files with specific file extensions. Additionally, the malware will back up and replace the master boot record (MBR) of the compromised system with an application, written in assembly language that is approximately 8194 bytes in length.

The malware will back up the original MBR by XORing it with the value 0x7h. Then it will write it to the physical disk after the modified MBR code. The backing up of the original MBR is presumably designed to allow the attacker to restore the machine to its original working state if the ransom is paid.

In addition to the malware using WNetAddConnection2 and WMIC to spread itself on an internal network, it will also attempt to use the EternalBlue exploit to spread itself onto other systems it has identified. During runtime, it will scan the local subnet from host 1 to host 255. It will also attempt to find other subnets on the victim's network using the Win32API DhcpEnumSubnets.

The malware will also attempt to identify other hosts on the target network by checking victim system’s IP physical address-mapping table using the Win32 API GetIpNetTable. The malware will then begin scanning all of the IPs to see if they are vulnerable to EternalBlue. After the scan is complete, the malware will begin the EternalBlue attack against identified vulnerable systems.

The payload sent to the victim systems via the EternalBlue exploit is encoded in portions via a four-byte XOR key. The four-byte XOR keys are as follows:

9EC8253D
3D9EC825
253D9EC8
C8253D9E

The following binary signatures may be utilized to detect this payload over port 445. They are simply the XOR keys themselves that will be used to represent NULL padding in the payload, as a NULL XOR bite by itself equals NULL. These signatures may quickly let an administrator know that this variant of ransomware is spreading across their internal network.
3D9EC8253D9EC8253D9EC8253D9EC825

253D9EC8253D9EC8253D9EC8253D9EC8

C8253D9EC8253D9EC8253D9EC8253D9E

9EC8253D9EC8253D9EC8253D9EC8253D

The individual victim's files are encrypted by a 128-bit AES algorithm, generated during runtime. Immediately before rebooting the system, the malware writes a file to the C:\drive named README.TXT. This file will contain a Bitcoin wallet and a hex encoded value. Static analysis indicates this hard-coded value is the AES key generated by the malware during runtime, which is used to encrypt the user's files. The malware contains the following hard coded public RSA key, which the malware imports shortly before rebooting the victim system.

--Begin Public RSA Key--

52534131000800000100010027 3A8A2B54712C367E3ADEF28ECDD954004DE9C525ECAEA3790296003D726142F45DF4D39D96A794858870BE326604AF1352FF31
7DF1DF4AFE9907FDF300953F9FA6EC6CEBCEE137EB6E36D9A7C66C89F391ACD3A4799BDEA80E53042522D1DA99C3B00D13F2C10F7B09BE0AA64DA3199B48B842
E32C159CD7EE82E6DF2442BD3DC7A25524F467FABF5C9DE4F5381861D7CFBBCBC2521044AC44067EB5830DD0CA94D9D09C05C882F92A97A226B6991421B7CD8E
3A7DDD98EA20DE894AF450B8C0418271F00F956B460787FB57564499897DDACE3712177978D38940C8F8F0AB90EB0E3D6100D448CA695DA9A53B1342C106C414
6A1526BDB7C834A7A8D5FFC4

--End Public RSA Key--

Several open-source reports indicate the malware encrypts the AES key generated during runtime, before writing it to the README.TXT file, using the hard coded RSA public key. However, within our lab environment the malware did not secure the AES key with the RSA key. Instead, the malware directly Base64 encoded and wrote the encoded value to the README.TXT file. Displayed below is the content of this value, extracted from README.TXT. The header clearly indicates it is an AES 128-bit key blob.

Static analysis indicates the modified MBR code encrypts the master file table of the victim's system after reboot. The algorithm utilized has been tentatively identified a modified version of Salsa20 cipher. As the publically available source code for Salsa20 below indicates, the cipher uses the constant string "expand 32-byte k" to initialize its key in the ENCRYPT_keysetup (ECRYPT_ctx *x,const u8 *k,u32 kbits,u32 ivbits) function.

This source code illustrates that bytes from this constant string are used within the actual encryption process. Specifically the static const char sigma variable is assigned to the constants variable within the ENCRYPT_keysetup function. The 16 bytes of the key are then placed in the first 16 bytes of the x-> input array. This array is then used within the ENCRYPT_ivsetup and ENCRYPT_encrypt_bytes functions. Therefore changing the value of this constant string may modify the result of the Salsa20 cipher itself.

Static analysis indicates this string has been modified within the Salsa20 implementation this malware uses. This constant string has been replaced with the 16-byte value "-1nvalid s3ct-id." A screenshot of this modification is attached to this product. This modification may also be designed to ensure identification of the Salso20 algorithm is more difficult.

The following hexadecimal value represents the op codes used to move this modified constant value into an array dereferenced off ESI. This value may be used as a signature to identify this modified Salsa20 algorithm:

--Begin Algorithm--

C646EF31C646F06EC646F176C646F261C646F36CC646F564C646F620C646F773C646F833
C646F963C646FA74B02D8846EE8846FBB0698846F48846FCC646FD64

--End Algorithm--

This value is also present, unobfuscated, within the malware DLL itself. This hex value may be utilized to specifically identify this malware variant.

Before rebooting, this malware generates a random 32-byte value using the API CryptGetRandom. This value will be used by the modified MBR code as an encryption key to encrypt the MFT once the system is rebooted. This newly generated value (encryption key), the original MBR (XOR encoded), and a dynamically generate 60 byte ID, are stored on the physical drive before the victim's system is rebooted – along with the ransomware's own MBR application stored at offset zero of the physical drive. Displayed below is a portion of this data pulled directly from the physical drive after the malware made the modifications to the physical drive:

Because this data is stored on the physical drive before the system is rebooted, the 32-byte random value will be available to encrypt the MFT and the XOR encoded original MBR would presumably be available for the replacement MBR to decode and reinstall as the new MBR, making the system bootable again to the user. This data is stored on the victim's hard disk beginning at approximately sector 32. The malicious MBR will be stored starting at offset zero.

It is important to note that there is no evidence of any correlation between the 32-byte encryption key generated and stored on the physical drive, and the 60-byte unique ID presented to the user after the system reboots. This is because the 32-byte encryption key is generated via the exact call to CryptGetRandom displayed in the screenshot below. This means that even if the 60-byte ID is given to the hacker, it would be impossible for them duplicate this 32-byte encryption key.

After infecting the victim system and encrypting their files, the malware spends exactly one hour trying to break into other machines on the local subnet. After one hour, the machine reboots, and the newly installed malware MBR application encrypts the MFT. This MFT code will present an image to the user, directing them to send $300 to the Bitcoin wallet hardcoded in the malware (see screenshot). It is not recommended for users to pay the ransom, because there is no way for the attacker to generate the 32-byte key used for MFT encryption from the personal installation key presented to them.

e84b.tmp

Details

Name	e84b.tmp
Size	56320
Type	PE32+ executable (console) x86-64, for MS Windows
MD5	7e37ab34ecdcc3e77e24522ddfd4852d
SHA1	38e2855e11e353cedf9a8a4f2f2747f1c5c07fcf
ssdeep	768:wglvV3eE6cYdk8TcqiAjpEhtFLIZJ92evTlvGTIRVIRYcwq22zDcENaiaX+kDt9J:3ihTaAjZNlGT5YcdYENvkA+ChJfSz
Entropy	5.68074084361

Antivirus

McAfee	Ransom-Petya
K7	Trojan (00510da71)
F-secure	Trojan:W32/Petya.H
Cyren	W64/Trojan.MMAF-4376
Symantec	Ransom.Petya
ClamAV	Win.Ransomware.Agent-6331177-0
Kaspersky	Trojan-PSW.Win64.WinCred.a
BitDefender	Trojan.GenericKD.5502132
Microsoft Security Essentials	Trojan:Win64/Petya.B!rsm
Sophos	Troj/Petya-BG
Emsisoft	Trojan.GenericKD.5502132 (B)
Avira	TR/Mimipet.airfqbb
Ahnlab	Trojan/Win64.Petya
NANOAV	Trojan.Win64.Petya.eqlcks
Ikarus	Trojan-Ransom.Petrwrap
AVG	SCGeneric2.BGME

PE Information

Compiled	2017-06-06T13:32:49Z

PE Sections

Name	MD5	Raw Size	Entropy
(header)	752e1dc68a35ed927754e33b44dc39c6	1024	2.46098666312
.text	41c154154c3c592c37bb98a92e6ca5c5	33792	6.23735881579
.rdata	43150e3618628f745b83404e03a757a0	12800	4.63103564159
.data	e81f0fbc70eef7673da7b05c1e5f2fcf	5632	2.21925552282
.pdata	5a7dd8c7eefd5343b5ac0b43ff90c07c	2048	3.9658470189
.reloc	dd78f228d35f6857111389a37d55e371	1024	2.96092898223

Packers

Name	Version	Entry Point
Microsoft Visual C++ 8.0 (DLL)	NA	NA

Description

Analysis indicates this dropped application is a modified version of the open-source tool known as “Mimikatz.” The ransomware will execute this modified Mimikatz binary using a command line argument, which will allow the malware to control the Mimikatz binary through a named pipe in virtual memory. The Mimikatz binary has been modified to prevent it from operating normally through STDOUT via the command line. Mimikatz is a tool that can be used to attain user’s username and password credentials from a Windows system. In this case, it is likely the malware using these stolen credentials to spread laterally on a network. However, we did not directly observe this activity within our lab

environment.

Mitigation Recommendations

US-CERT reminds users and administrators of the following best practices to strengthen the security posture of their organization's systems:

- Maintain up-to-date antivirus signatures and engines.
- Restrict users' ability (permissions) to install and run unwanted software applications.
- Enforce a strong password policy and implement regular password changes.
- Exercise caution when opening e-mail attachments even if the attachment is expected and the sender appears to be known.
- Keep operating system patches up-to-date.
- Enable a personal firewall on agency workstations.
- Disable unnecessary services on agency workstations and servers.
- Scan for and remove suspicious e-mail attachments; ensure the scanned attachment is its "true file type" (i.e., the extension matches the file header).
- Monitor users' web browsing habits; restrict access to sites with unfavorable content.
- Exercise caution when using removable media (e.g., USB thumb drives, external drives, CDs, etc.).
- Scan all software downloaded from the Internet before executing.
- Maintain situational awareness of the latest threats; implement appropriate ACLs.

Contact Information

- 1-888-282-0870
- soc@us-cert.gov (UNCLASS)
- us-cert@dhs.sgov.gov (SIPRNET)
- us-cert@dhs.ic.gov (JWICS)

US-CERT continuously strives to improve its products and services. You can help by answering a series of short questions about this product at the following URL: https://forms.us-cert.gov/ncsd-feedback/

Document FAQ

What is a Malware Initial Findings Report (MIFR)? A MIFR is intended to provide organizations with malware analysis in a timely manner. In most instances, this report will provide initial indicators for computer and network defense. To request additional analysis, please contact US-CERT and provide information regarding the level of desired analysis.

Can I edit this document? This document is not to be edited in any way by recipients. All comments or questions related to this document should be directed to the US-CERT Security Operations Center at 1-888-282-0870 or soc@us-cert.gov.

Can I submit malware to US-CERT? Malware samples can be submitted via three methods. Contact us with any questions.

- Web: https://malware.us-cert.gov
- E-Mail: submit@malware.us-cert.gov
- FTP: ftp.malware.us-cert.gov/malware (anonymous)

US-CERT encourages you to report any suspicious activity, including cybersecurity incidents, possible malicious code, software vulnerabilities, and phishing-related scams. Reporting forms can be found on US-CERT's homepage at www.us-cert.gov.