XMRig Malware Analysis Report

Tags	CryptoMiner XMRig
Date	@April 8, 2023
MD5	e77dedf5bf9a251e0f10db9b3545fb03
SHA2565	9b4d6eb9b5bf2c99b4b977a12c284874b104d570524a013583ec125b3ba4f70e
Sample	9b4d6eb9b5bf2c99b4b977a12c284874b104d570524a013583ec125b3ba4f70e.zip
Signature	XMRig
Type	EXE

Introduction

HTML and JavaScript are two of the most ubiquitous programming languages used on the internet today. HTML, or Hypertext Markup Language, is a markup language used to create web pages and other online content. It provides the structure for web pages by defining elements such as headings, paragraphs, links, images, and more. JavaScript, on the other hand, is a scripting language that is used to add interactivity and functionality to websites. It allows website owners to create dynamic and interactive websites by manipulating the HTML and CSS on the page.

While both HTML and JavaScript are incredibly useful for creating rich and engaging web experiences, they can also be used maliciously. Malware, short for malicious software, is any program that is designed to harm or exploit computer systems. HTML and JavaScript-based malware are becoming increasingly common, as they offer a relatively easy way for attackers to compromise systems.

There are many different types of HTML and JavaScript-based malware, ranging from phishing scams to drive-by downloads. Some of the most common forms of HTML and JavaScript-based malware include adware, spyware, trojan horses, and viruses. These types of malware can be spread through infected email attachments, downloadable files, or through compromised websites.

To protect against HTML and JavaScript-based malware, it is important to keep your web browser and other web-related software up-to-date with the latest security patches. Additionally, exercise caution when visiting unfamiliar websites or clicking on suspicious links. Using reputable antivirus and anti-malware software can also help detect and remove any malware that may have infected your system.

Executive Summary

The malware sample is signed binary of Google LLC which comes as a Google Chrome Installer Software. The binary is obfuscated and trojanized with malicious file. When a user visit the website or webpage, it exploits browser based vulnerabilities and spawns a process under browser process and then extracts installer software's header data into a binary file. This file is then used for malicious intent. This is how this malware spreads on multiple computer with having signature of Google LLC.

Methodology

I have use below methodology to analyze this particular malware sample.

- Basic Static Analysis
  - Finding Properties
  - PE Header Inspection
  - Import and Exports Inspection
  - Strings Analysis

- Dynamic Analysis
  - Hybrid Analysis Automated Sandboxing

Results

Here are the Indicators Of Compromise listed below,

▼ Host-Based Indicators

- Version: 110,0,5481,104
- Invalid Data Dictionaries
- Very Less Amount of Imports
- Strings Listed in Analysis Phase

▼ Network-Based Indicators

- Not Found

Analysis

The analysis of the provided sample includes various types of analyses, ranging from basic static to advanced dynamic analysis. The report contains information specific to the date and time of analysis (**@April 8, 2023**). Please note that the information provided may vary at the time of reading.

Static Analysis

XMRig malware sample comes in a form of **.exe** (Windows Executable Format). This sample looks like a Chrome Installer having signature of Google LLC, look like a trojanized executable. (See figure 1.0 & 1.1)

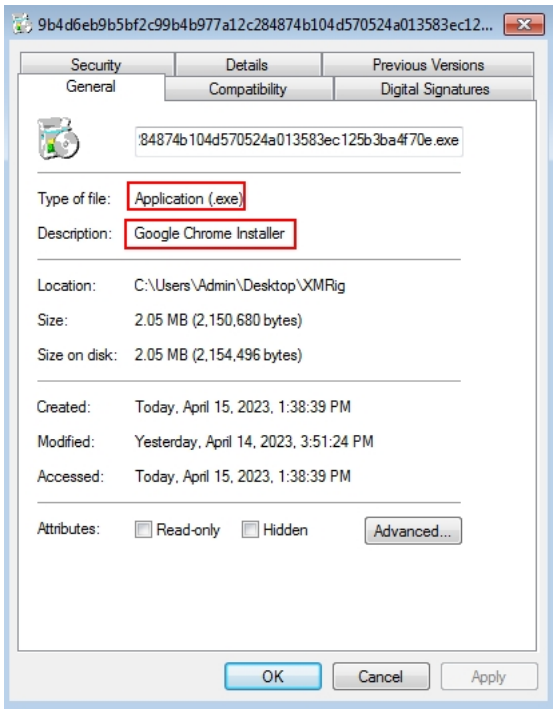


Figure 1.0

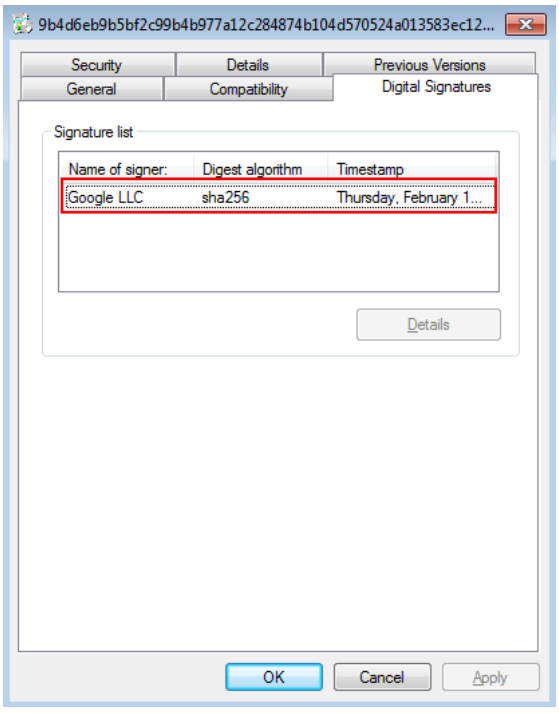


Figure 1.1

When I looked at CFF Explorer and Data Directories at seems like the malware is not obfuscated. The malware is PE64 (Portable Executable x64 bit) variant. This means only 64 Bit computers can run the file which is very weird because malware authors generally writes there code into PE32 format so that the infection can spread over more system. (See figure 1.2 & 1.3)

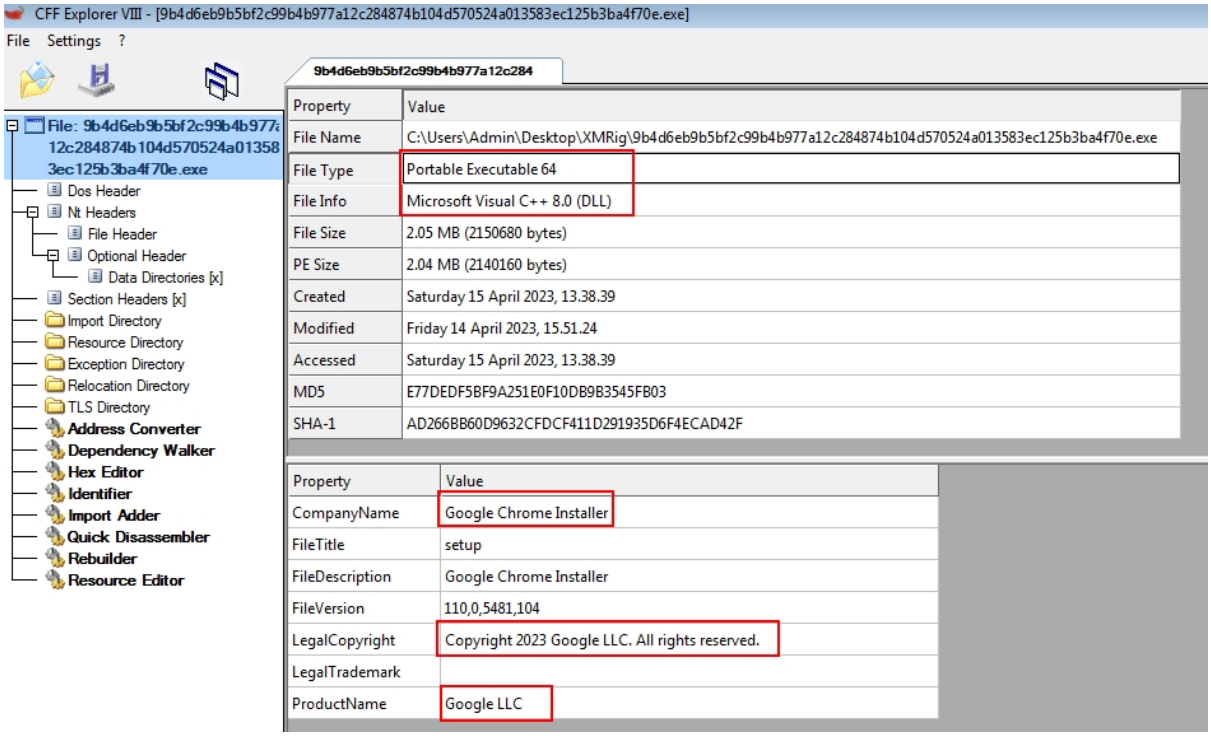


Figure 1.2

Name	Virtual Size	Virtual Address	Raw Size	Raw Address	Reloc Address
000001B0	000001B8	000001BC	000001C0	000001C4	000001C8
Byte[8]	Dword	Dword	Dword	Dword	Dword
.text	0000AAA8	00001000	0000AC00	00000400	00000000
.data	001FBAE0	0000C000	001FBC00	0000B000	00000000
.rdata	00000F00	00208000	00001000	00206C00	00000000
.pdata	000007E0	00209000	00000800	00207C00	00000000
.xdata	000006B4	0020A000	00000800	00208400	00000000
.bss	00000C38	0020B000	00000000	00000000	00000000
.idata	000008F0	0020C000	00000A00	00208C00	00000000
.CRT	00000068	0020D000	00000200	00209600	00000000
.tls	00000010	0020E000	00000200	00209800	00000000
.rsrc	00000A78	0020F000	00000C00	00209A00	00000000
.reloc	0000008C	00210000	00000200	0020A600	00000000

Figure 1.3

The malware is importing very few (Only Two Libraries), which is also very weird and does not export any function. This looks very suspicious because little program hash more imports then this. This seems like malware is obfuscated. (See figure 1.4)

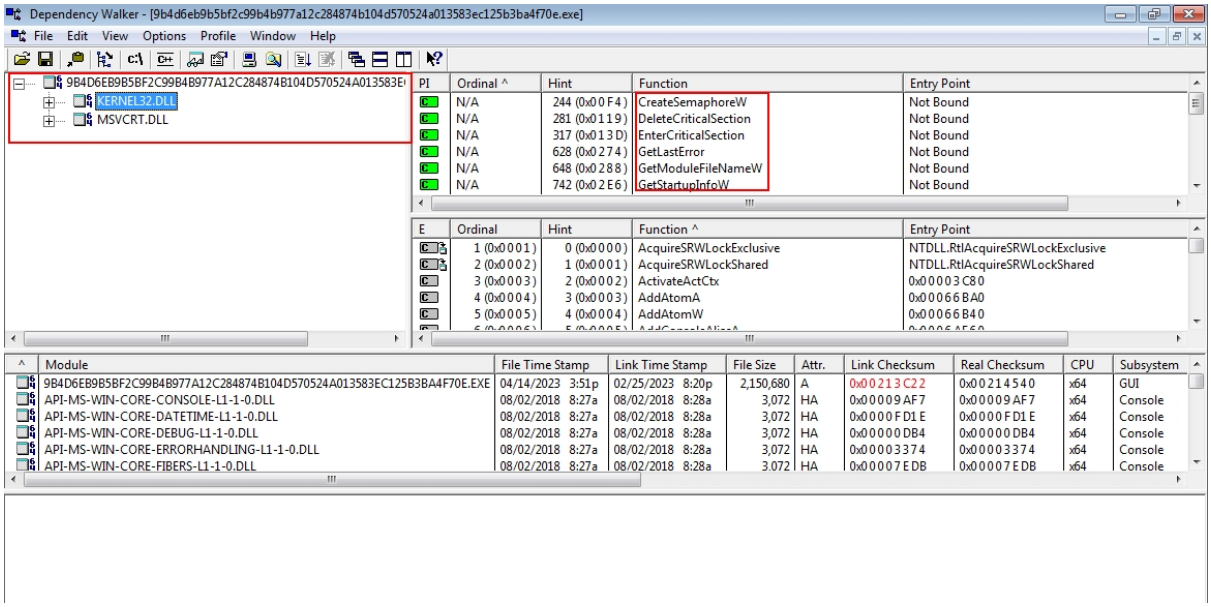


Figure 1.4

Member	Offset	Size	Value	Section
Export Directory RVA	00000108	Dword	00000000	
Export Directory Size	0000010C	Dword	00000000	
Import Directory RVA	00000110	Dword	0020C000	.idata
Import Directory Size	00000114	Dword	000008F0	
Resource Directory RVA	00000118	Dword	0020F000	.rsrc
Resource Directory Size	0000011C	Dword	00000A78	
Exception Directory RVA	00000120	Dword	00209000	.pdata
Exception Directory Size	00000124	Dword	000007E0	
Security Directory RVA	00000128	Dword	0020A800	Invalid
Security Directory Size	0000012C	Dword	00002918	
Relocation Directory RVA	00000130	Dword	00210000	.reloc
Relocation Directory Size	00000134	Dword	0000008C	
Debug Directory RVA	00000138	Dword	00000000	
Debug Directory Size	0000013C	Dword	00000000	
Architecture Directory RVA	00000140	Dword	00000000	
Architecture Directory Size	00000144	Dword	00000000	
Reserved	00000148	Dword	00000000	
Reserved	0000014C	Dword	00000000	

Figure 1.5

When I look at strings, it is gibberish. This indicates that binary is obfuscated. Some of the strings have valid URLs which shows that the binary is a signed malware. This kind of malware is very hard to detect. (See figure 1.6, 1.7 & 1.8)

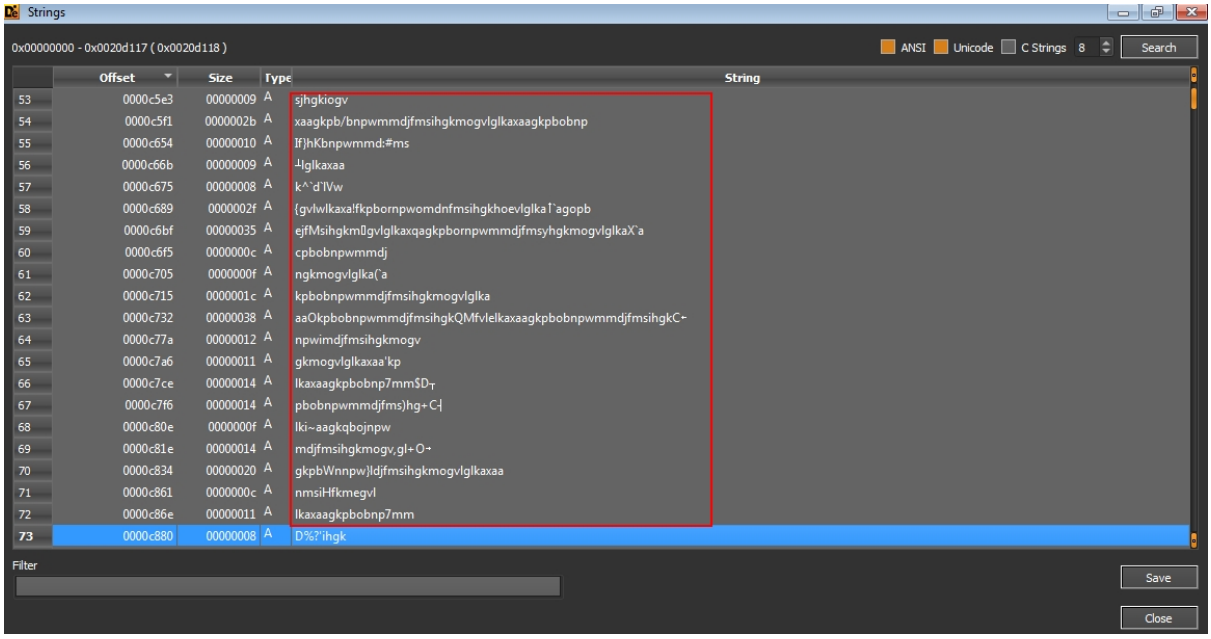


Figure 1.6

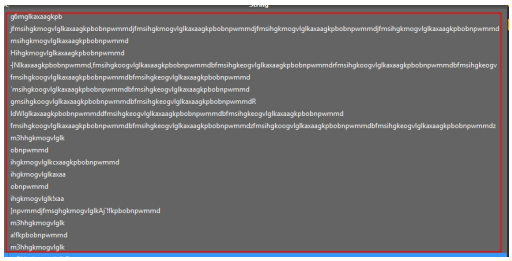


Figure 1.7

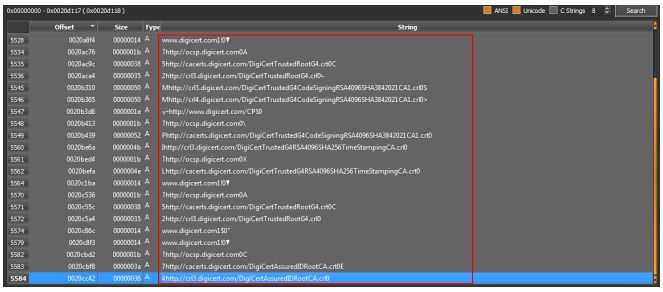


Figure 1.8

Tried to analyze the file strings with floss. Even floss is not able to decode the obfuscated strings. This sample is highly obfuscated and can not be de-obfuscated easily. The sample is made with Microsoft Visual C++ 8.0. (See figure 1.9 and 2.0)

```
FLARE FLOSS RESULTS (version v2.2.0-0-g783dd8f)
+------------------------+------------------------------------------------------------------------------------------+
| file path              | 9b4d6eb9b5bf2c99b4b977a12c284874b104d570524a013583ec125b3ba4f70e.exe                     |
| extracted strings      |                                                                                          |
|  static strings        | 3493                                                                                     |
|  stack strings         | 0                                                                                        |
|  tight strings         | 1                                                                                        |
|  decoded strings       | 1                                                                                        |
+------------------------+------------------------------------------------------------------------------------------+
```

Figure 1.9

```
h`!fkpbobnpwmmd
vl3hhgkmogvlglkah`!fkpbobnpwmmd
vl3hhgkmogvlglkId`!fkpbobnpwmmdJzl3hhgkmogvlglk
a!fkpbobnpwmmdjVl3hhgkmogvlglkqH`!fkpbobnpwmmdrVl3hhgkmogvlglkIH`!fkpbobnpwmmd
vl3hhgkmogvlglk
a!fkpbobnpwmmd
vl3hhgkmogvlglkA?a!fkpbobnpwmmd
Ym3hhgkmogvlglk1h`!fkpbobnpwmmd-%.II@
KOS\^EC]d-%.II@
KOS\^EC]d-%.II@
KOS\^EC]d-%.II@
KOS\^EC]djfmsihgkmogvlglkaxaagkpbobnpwmmdjfmsihgkmogvlglkaxaagkpbobnpwmmdjfmsihgkmogvlglkaxaa
pbGbopwxmdwsms!hfkMzgvUrlk  x`a'~pb#wnp
mld:sms8}gk
DgvpfmkaTaacZpb_copg\md
^gvnUlk5y`awYpb
jqb_Vnp0Xmd
ifk-Wgv"^lk
SgvdemkQFaa
YlkEz`agTpb@]np_oldZYms
-md#'ms jfk=.gv.$lk
mfv| lk}?aa
)lku{`ag$pb+-nponld:)ms
8gkO>gvTdmkQ)aa<:pb/aop
9pb\1np/nld*5ms
gvhamkaxaagkpbobnpwmmdjfmsihgkmogvlglkaxaagkpbobnpwmmdjfmsihgkmogvlglkaxaagkpbobnpwmmdjfmsihgkmogvlglkaxaagkpbobnpwmmdjfmsihg
aafdxb`c}p
dkfvh%lk
|gkdkfvh%lk
aafkpbkwnp`xmdJ!ms~}gklkfvh%lk`xaafkpbnbnpvmmdkfmshhgklogvmglk`xaafkpbnbnpvmmdkfmshhgklogvmglk`xaafB~bF
fkpbnbnpvmmdkfmshhgklogvmglk`xaafkpbnP~pE
v~jdygaSbXm
```

Figure 2.0

Dynamic Analysis

The dynamic analysis is a process of analyzing malware while it's execution on the system. This can be done in two ways, 1) Manual Dynamic Analysis, 2) Sandboxing. We are going to utilize online sandboxing method to analyze the malware. I have used Hybrid Analysis here for dynamic analysis.

The provided sample runs as below process tree and malware sandbox flagged them as malicious. (See figure 2.1)

Figure 2.1

The malware is using originally signed executable to spread malware on target systems. This certificate is signed by Google LLC (See figure 2.2)

Owner	Issuer	Validity	Hashes (MD5, SHA1)
CN=DigiCert Trusted G4 Code Signing RSA4096 SHA384 2021 CA1, O="DigiCert, Inc.", C=US	CN=DigiCert Trusted Root G4, OU=www.digicert.com, O=DigiCert Inc, C=US Serial: 8ad40b260d29c4c9f5ecda9bd93aed9	04/29/2021 00:00:00 04/28/2036 23:59:59	D9:12:99:EB:43:55:CD:BD:5A:86:79:5A:01:18:B6:E9 7B:0F:36:0B:77:5F:76:C9:4A:12:CA:48:44:5A:A2:D2:A8:75:70:1C
CN=Google LLC, O=Google LLC, L=Mountain View, ST=California, C=US	CN=DigiCert Trusted G4 Code Signing RSA4096 SHA384 2021 CA1, O="DigiCert, Inc.", C=US Serial: e4418e2dede36dd2974c3443afb5ce5	07/02/2021 00:00:00 07/10/2024 23:59:59	DC:42:9A:22:AA:63:D2:3D:B8:E8:4F:53:D0:5D:1D:48 26:73:EA:6C:C2:3B:EF:FD:A4:9A:C7:15:B1:21:54:40:98:A1:28:4C

Figure 2.2

The malware has unusual entropy section. Entropy refers to the amount of randomness or unpredictability in a section of code or data. Entropy is often used as a measure of the level of compression or encryption applied to a particular section of the executable.

In general, sections with high entropy are more difficult to analyze or reverse engineer because they contain more random or unpredictable data. This can make it harder for security researchers or malware analysts to determine what a particular section of code or data is doing, which can be useful for obfuscating malware or protecting software from reverse engineering.(See figure 2.3)

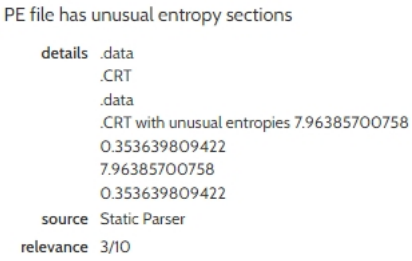


Figure 2.3

The malware drops executable file which are flagged as malicious. (See figure 2.4)

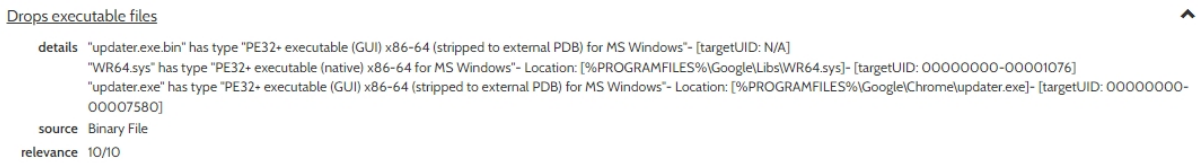


Figure 2.4

Here is interesting part comes into the picture. The malware sample copied some data from PE file header to a file. This means malware has injected code into PE file header. Malware authors can use different techniques to hide malicious code within the header, such as modifying unused fields or adding new fields and sections to the header. (See figure 2.5)

PE file also contains writeable header sections which clarifies the above information. (See figure 2.6)

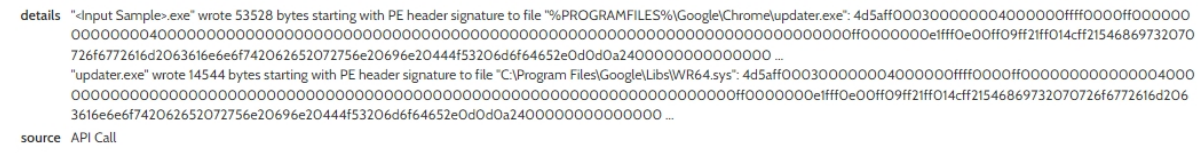


Figure 2.5

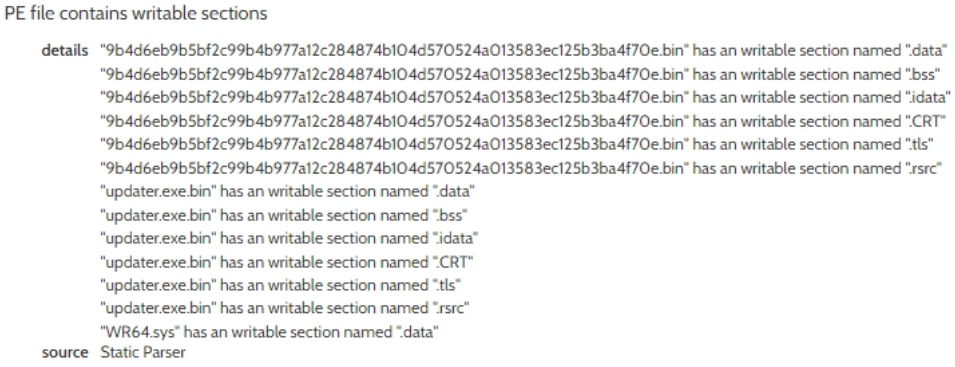


Figure 2.6

The malware hooks/patches existing process memory. (See figure 2.7)

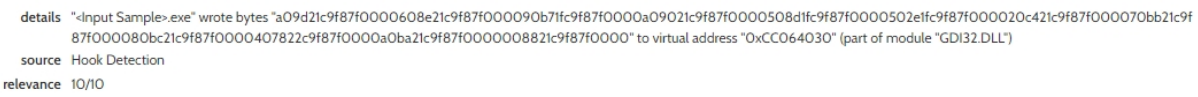


Figure 2.7

The malware loads cryptographic modules and RPC modules. The RPC can be used to to access and communicate with remote process. The cryptographic modules can be used to encrypt or decrypt any kind of process or files. (Figure 2.8)

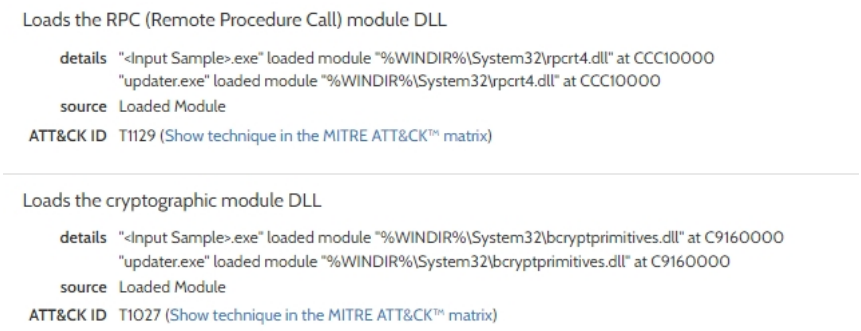


Figure 2.8

The malware sample tries to access non existence DLL file in program files directory. This DLL file looks like an essential part of malware which can be used to extend the functionality of the malware. (See figure 2.9)

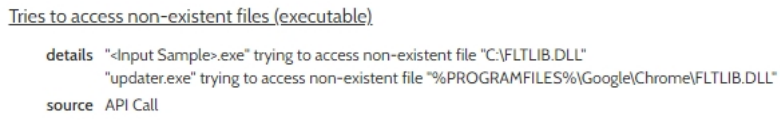


Figure 2.9

The malware creates or modifies windows services. This can be used for persistence technique or to hide any kind of malicious payload. (See figure 3.0)

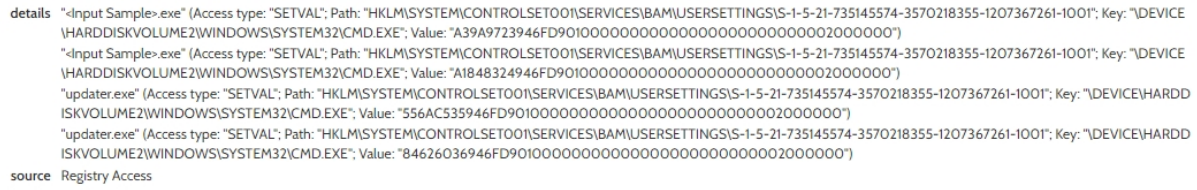


Figure 3.0

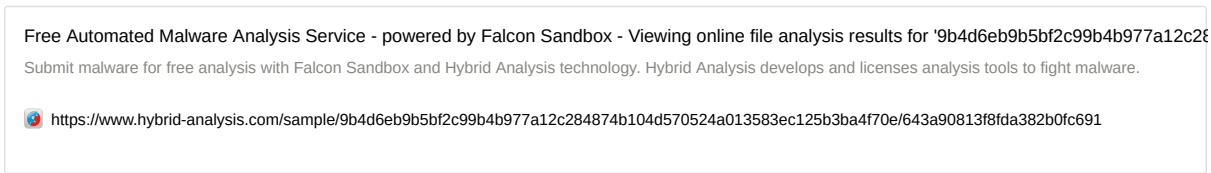
Conclusion

Now in the era of technology, a person can be hack just by visiting a web page. Whether it is online hosted web page or locally served. People should be aware of this kind of attacks and it's effects. Here are some remediation to avoid this kind of activity.

- Keep updated your browser.
- Keep updated Operating System and other software applications.
- Do not click on any kind of unknown link or open any kind of file sent from unknown sources.
- Educate your staff and arrange awareness campaigns.
- Implement web URL whitelisting technology in corporate environment.

Appendices

1. Hybrid Analysis Report :



2. VirusTotal Report :

VirusTotal

https://www.virustotal.com/gui/file/9b4d6eb9b5bf2c99b4b977a12c284874b104d570524a013583ec125b3ba4f70e