



Malware Analysis

A REPORT ON ANALYSIS OF MALWARE USING MULTIPLE
TECHNIQUES

Prasann Deshpande | Introduction to Computer Security | 5/15/2020

B-Number - Boo814763

INDEX

Malware Sample	3
Sample Basic details:	3
File Details	3
Signatures	3
Static analysis	3
1. Files Hashes – MD5, SHA-1, SHA-256	4
2. File type and Size	4
3. Strings	5
4.Imports	6
5.YARA Rule matches	7
Static Analysis Observations	8
Dynamic analysis	9
1.Process Tree	9
Target Section	9
Process Behavior Section	10
2. Registry Manipulation	13
3.Cuckoo Screenshots	15
Dynamic Analysis Observations	18
Insights	18
References	19

Malware Sample

The malware sample used is from the Big Bing database. The launch year was 2017 and its filetype is win32 executable. Some details below have been extracted from analyzing the radare2 results.

Sample Basic details:

File Details

File type- Win32 EXE
File size 1.04 MB (1086808 bytes)

Signatures

MD5 8f2142edec463ef62d00bf8705d215da
SHA-1 19fe1a703cbe36a942boe4034864a5722ea4cf8b
SHA-256 1118593657f16184517d653a496be161a483fa3ef60748403ad3988151623fed

Static analysis

Static Analysis mainly deals with examining the malware sample without executing the code. In basic static analysis we examine the file without viewing its system level instructions. Static analysis is generally performed by scanning the malicious file using anti-virus scanners. But for malwares that have already been identified, we can simply search by using their file hashes.

Some techniques for static analysis –

1. Uploading the sample to Antivirus websites.
2. Looking for strings in the executable
3. Analyzing the Portable Executable file format section

Static analysis however is not as easy as it sounds. A lot of Malwares today have anti-debugging features that restrict the access to the malware code and make it difficult for someone to analyze

Some of the important static information found out about our sample is listed below –

1. Files Hashes – MD5, SHA-1, SHA-256
2. File type and Size.
3. Strings
4. Imports
5. YARA rule matches

Let's look at these one by one

1. Files Hashes – MD5, SHA-1, SHA-256

Hashes are the output of a hashing algorithm like MD5 (Message Digest 5) or SHA (Secure Hash Algorithm). These algorithms essentially aim to produce a unique, fixed-length string – the hash value, or “message digest” – for any given piece of data or “message”.

These file hashes help us explore the repositories of well-known malware libraries so that we can find out more about the sample. Details like file type, size, origin date, compression method etc. are common details that can be found. A few details for this malware sample are already listed in the “MALWARE SAMPLE” section.

Those details were found using the rahash2 tool of Radare2 as shown below-

```
jupyter-pdeshpa2@jupyter18:~$ rahash2 -q -a sha1 PXOIZAOOLUXMNKZACZNRGCTYLAOTJU
```

```
19fe1a703cbe36a942boe4034864a5722ea4cf8b
```

```
jupyter-pdeshpa2@jupyter18:~$ rahash2 -q -a md5 PXOIZAOOLUXMNKZACZNRGCTYLAOTJU
```

```
8f2142edec463ef62d00bf8705d215da
```

```
jupyter-pdeshpa2@jupyter18:~$ rahash2 -q -a sha256 PXOIZAOOLUXMNKZACZNRGCTYLAOTJU
```

```
1118593657f16184517d653a496be161a483fa3ef60748403ad3988151623fed
```

2. File type and Size

File type and size for this malware sample are –

File type	Win32 EXE
-----------	-----------

File size	1.04 MB (1086808 bytes)
-----------	-------------------------

4.Imports

Using command – **fs imports; f** , we can find out the system functions being used by the malware sample. Although there is a total of 248 imports listed in the result, below are the ones that seem interesting-

0x00441130 o sym.imp.KERNEL32.DLL_DeleteCriticalSection

0x00441134 o sym.imp.KERNEL32.DLL_HeapDestroy

0x00441298 o sym.imp.SHLWAPI.DLL_URLUnescapeW

0x004413a8 o sym.imp.WININET.DLL_InternetCloseHandle

0x004413ac o sym.imp.WININET.DLL_HttpSendRequestW

0x004413bo o sym.imp.WININET.DLL_InternetCrackURLW

0x004413b4 o sym.imp.WININET.DLL_InternetOpenW

0x004413b8 o sym.imp.WININET.DLL_HttpOpenRequestW

0x004413bc o sym.imp.WININET.DLL_InternetConnectW

A lot of entries in the results are related to KERNEL32.DLL,USER32,ADVAPI32,GDI32,ole32.DLL, SHLWAPI.DLL, WININET.DLL which are important system DLLs. Below is a brief description of each of these DLLs-

DLLs	General Use
KERNEL32.DLL	DLL containing core functionality. Can be used to access and manipulate memory, files and hardware
USER32	Contains all user interface components and control and respond components for use interactions
ADVAPI32	Provides access to advanced core windows components like service manager and registry
GDI32	Contains functions for displaying and manipulating graphics
OLE32.DLL	Object linking and embedding operations
SHLWAPI.DLL	contains functions for UNC and URL paths, registry entries, and color settings.
WININET.DLL	Contains high level networking functions for protocols such as HTTP, NTP and FTP

These DLLs also point towards the theory that the malware seems to work with internet connectivity. However, it could be much more as the DLLs listed above can cause a lot of manipulation of data and memory.

5.YARA Rule matches

YARA is a powerful malware classification tool that can help in understating the category of malware any malware sample falls into. It provides a rule-based approach to describe malware families. YARA scans the malware according to these rules which contain strings for matching and conditions for logic building.

Since, the malware seems to work with internet resources, it is best to create some general rules as well as rules that will be able detect internet connectivity DLL calls.

YARA rules can be created by understanding what a set of DLLs together can accomplish. There could be multiple DLLs that can achieve same tasks. Such DLL names can be passed as strings, few of which can then be matched to determine if the set is enough to perform a given task.

Since we already know that the malware is an EXE, the most basic rule and good way to start with YARA is to write a exe identifying rule and see if it matches.

```
The Rule – rule is_exe{
  meta:
    description = "Is an executable"
  strings:
    $a1 = "This program cannot be run in DOS mode"
    $a2 = "This program must be run under Win32"
  condition:
    $a1 or $a2
}
```

The above rule checks for the string “” or “” as such strings are common in windows executables and can confirm whether a given file is an executable. Such checks are very useful in case of obfuscated malware or malware samples that are hidden under some form of compression.

The above rule, as expected, matches with out malware sample. Thus, the malware is an executable file.

Some important rules matched with YARA for the malware sample are listed in the table below

Matched Rules	Description
find_kernel32,	Makes use of kernel32 DLL
antidebug,	Checks for debugging
http_rule,	Uses DLLs required for HTTP calls
screenshot,	Capable of Taking screenshots
windows_registry,	manipulation of system registries
user_profile,	Manipulating private user profile
Win32_Internet_API,	Use of Windows Internet API calls
Win32_Http_API	Use of Windows Http API calls

These matches clearly suggest that the malware tries to load DLLs that are important for internet connectivity. It also suggests that malware tries to check if it is being debugged.

Other entries are related to registries tell that the malware is trying to manipulate registries.

Static Analysis Observations

From all the analysis carried out above, the key observations are –

1. The Malware sample is a Windows executable file
2. The Malware's size is 1.04 MBs or 1086808 bytes
3. The malwares code contains a lot of garbage strings
4. Some interesting strings points towards the use of internet related commands
5. There is a full webpage code embedded inside the executable
6. Malware is a potentially unwanted program
7. A lot of system functions are being used by this malware
8. Some important DLLs imported make use of core functionality.
9. The malware is capable of manipulating graphics
10. YARA rule also identifies this malware as an executable
11. YARA also classifies the malware to be using kernel32.DLL, making HTTP calls, capable of taking screenshots, manipulating system registries etc.

Dynamic analysis

Dynamic malware analysis is the method used for extracting and determining malware's execution behavior. Dynamic or Behavioral analysis is performed by observing the behavior of the malware while it is running on a host system. This form of analysis is often performed in a sandbox environment to prevent the malware from infecting the analysts system. Many such sandboxes are virtual systems that can easily be returned to a clean state after the malware analysis is complete.

The Dynamic analysis of this malware was done using Cuckoo Sandbox.

Cuckoo Sandbox is the leading open source automated malware analysis system. Its API can be used to test any malware sample or any file. It is used to automatically run and analyze files and collect comprehensive analysis results that outline what the malware does while running inside an isolated Windows operating system.

The Analysis results in the following details being revealed –

1. Process Tree
 - a. Target
 - b. Process Behavior
2. Registry manipulation
3. Screenshots

1.Process Tree

The process tree that is generated by the execution of the malware can be understood by analyzing the cuckoo report. First, the target section is analyzed.

Target Section

First, the target section is analyzed to figure out the basic details about the malware.

Following are the details from the target section of the report –

name: "temp_file_name",

type: "PE32 executable (GUI) Intel 80386, for MS Windows",

sha1: "19fe1a703cbe36a942boe4034864a5722ea4cf8b",

sha256: "1118593657fi6184517d653a496be161a483fa3ef60748403ad3988151623fed",

size: 1086808

md5: "8f2142edec463ef62doobf8705d215da"

URLs:

- "http://legal.loginfaster.com/legal/terms",
- "http://crl.globalsign.com/gscodesignsha2g3.crlo",
- "http://www.IIS",
- "http://crl.globalsign.com/root-r3.crloc",
- "http://secure.globalsign.com/cacert/gscodesignsha2g3ocsp.crto8",
- "http://ocsp2.globalsign.com/rootr3o6",
- "https://www.globalsign.com/repository/o",
- "http://IIS",
- "http://ns.adobe.com/xap/1.0/mm/",
- "http://legal.",
- "http://ns.adobe.com/xap/1.0/sType/ResourceRef",
- "http://ocsp2.globalsign.com/gscodesignsha2g3oV",
- "http://legal.loginfaster.com/legal/privacy",
- "http://ns.adobe.com/xap/1.0/"

The name value "temp_file_name" is the temporary file name given by cuckoo to the sample. This name can be followed throughout the document to extract relevant data.

The URLs sections also gives a key piece of information which is the complete list of URLs directly referenced by the malware. This information can be used to check if any of the listed URLs are malicious and can help us identify the malware.

The above details also confirm the hashes and file details that are stated in the above section.

Process Behavior Section

Cuckoo specializes in behavioral analysis. Behavioral analysis tells us about how the malware behaves during execution, what files it uses, what registries it uses etc.

All the behavior related details can be found in the behavior section of the report.

Some important details regarding this malware's process tree are found in its behavior. Some other details are also obtained that corroborate with the current findings.

By finding the "temp_file_name" in the behavior section, we can find out more about the behavior of the malware file. Some key details –

- "pid": 2792
- file_created : "C:\\Users\\win7\\AppData\\Roaming\\{28e56cfb-e30e-4f66-85d8-339885b726b8}\\Uninstall.exe"
- "file_opened": "C:\\Program Files\\Internet Explorer\\iexplore.exe"

- "file_opened": "C:\\Users\\win7\\AppData\\Local\\Temp\\temp_file_name.exe"
- connects_host : "imp.searchffr.com"
- command_line :
 - "\"C:\\Program Files\\Internet Explorer\\IEXPLORE.EXE\" http://search.searchffr.com/?source=bing&uid=bd334940-402b-4323-be0a-20a68956658f&uc=20171116&ap=appfocus63&i_id=recipes__1.30",
 - "C:\\Windows\\System32\\cmd.exe /c FOR /L %V IN (1,1,10) DO del /F \"C:\\Users\\win7\\AppData\\Local\\Temp\\temp_file_name.exe\" >> NUL & PING 1.1.1.1 -n 1 -w 1000 > NUL & IF NOT EXIST \"C:\\Users\\win7\\AppData\\Local\\Temp\\temp_file_name.exe\" EXIT",

Apart from the above, the malware also manipulates a lot of registry files, especially related to the internet and internet explorer settings.

Now, following the PID (2792), the processes created by the malware can be traced and behavior can be understood.

- First, temp_file_name.exe is started
- Temp_file_name.exe creates two processes – Iexplore.exe and cmd.exe
- IExplore.exe is opened with a suspicious URL as an argument.
- Cmd.exe is also started with an argument pointing to 1.1.1.1 which is the ip of Cloudflare
- Cmd.exe argument has /c so that it closes as soon as the execution is complete
- Cmd.exe argument is a shell command string with a few things to do – delete temp_file_name.exe, ping 1.1.1.1 , check if file deleted and exit.
- IExplore.exe creates another iexplore.exe. This one does not seem suspicious

The process tree is as follows –

Malware → CMD → Ping.exe

CMD → Internet Explorer → Internet Explorer

Another section we can use for this is the processtree section which confirms our traced process tree.

So, the malware is trying to connect to the internet and push the suspicious URL to internet explorer. It also deletes itself, probably in order to prevent any future sampling

and analysis. It's deletion however will not restore the system to its original state as the malware has already edited a lot of registries which hold the internet settings. By deleting itself, the malware avoids any antivirus software from flagging it and in turn alerting the user. It is quite clear that the malware is trying to attack internet explorer and that it is trying to make the user visit a specific URL. At this point it seems like this malware is an Adware.

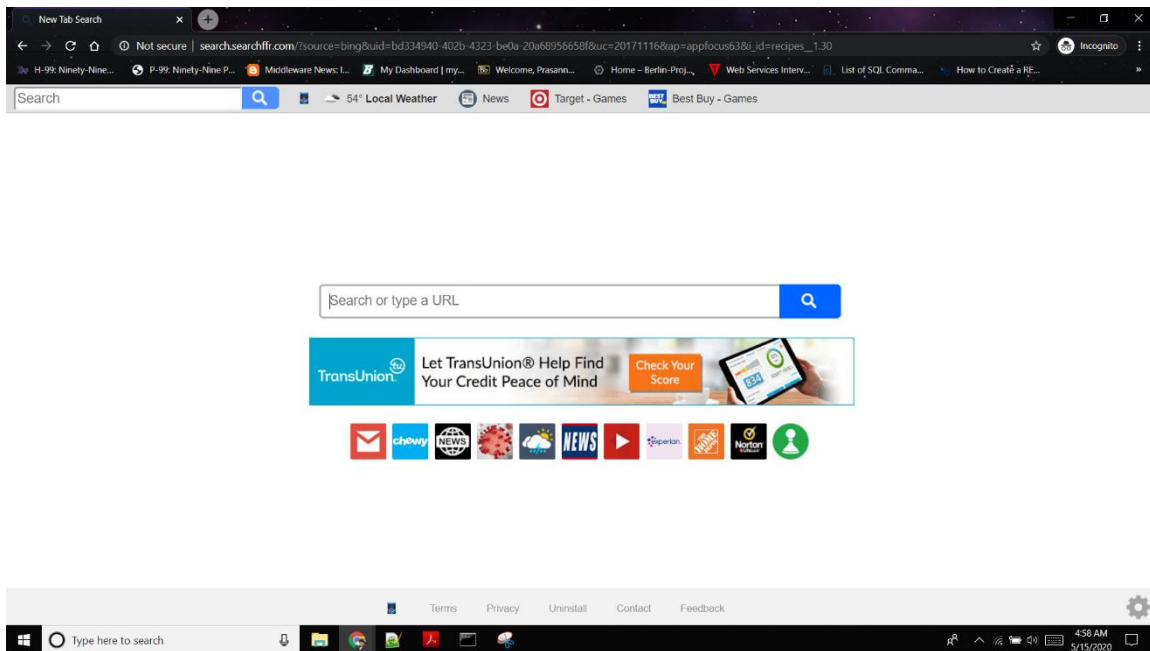
URL: http://search.searchffr.com/?source=bing&uid=bd334940-402b-4323-be0a-20a68956658f&uc=20171116&ap=appfocus63&i_id=recipes__1.30

Scanning this URL on URLscan.io gives us the following result –

This website contacted 11 IPs in 3 countries across 9 domains to perform 30 HTTP transactions.

The main IP is 54.225.149.21, located in Ashburn, United States and belongs to AMAZON-AES, US. The main domain is search.searchffr.com.

A screenshot from URLscan.io –



The URL page looks like a generic search engine page. Manually reading the URL, it is understood that the search is powered by Bing (www.bing.com). This behavior is seen with a lot of Adware which try to bombard the browser with offers and change the default search engines to something like the above screenshot. An interesting thing to note is that

these kinds of pages only try to look like search engines but leverage the searching capabilities of established search engines.

A quick search about “search.searchffr.com” confirms the suspicion that this URL is in fact part of an Adware.

2. Registry Manipulation

From the behavior section of the malware, it is evident that a lot of registry manipulation is taking place during the execution of the malware (temp_file_name.exe). The registry stores low level data for Windows OS and the applications to operate properly. While not all applications depend on the registry files to work, a lot of internet settings reside in the registry.

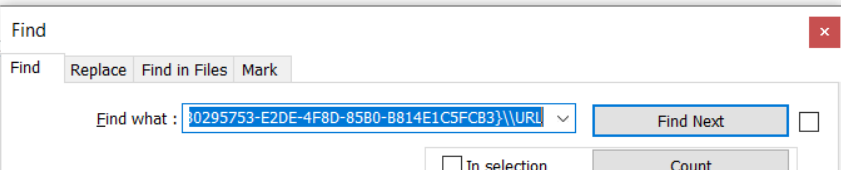
Below are some registries being manipulated-

- "HKEY_CURRENT_USER\\Software\\Microsoft\\Internet Explorer\\SearchScopes\\{80295753-E2DE-4F8D-85Bo-B814E1C5FCB3}\\SuggestionsURL",
- "HKEY_CURRENT_USER\\Software\\Microsoft\\Internet Explorer\\SearchScopes\\{80295753-E2DE-4F8D-85Bo-B814E1C5FCB3}\\DisplayName",
- "HKEY_CURRENT_USER\\Software\\Microsoft\\Internet Explorer\\Main\\Start Page"
- "HKEY_CURRENT_USER\\Software\\Microsoft\\Internet Explorer\\SearchScopes\\{80295753-E2DE-4F8D-85Bo-B814E1C5FCB3}\\URL",

Just reading the ending parts of the registry entries gives a vague idea that the malware is editing the start page, URL, suggested URLs and Display name for Internet Explorer.

Finding the last entry from the above list in the cuckoo report reveals the value of this entry –

```
"category": "registry",
"status": 1,
"stacktrace": [],
"api": "RegSetValueExW",
"return_value": 0,
"arguments": {
  "key_handle": "0x0000009c",
  "value": "http://search.searchffr.com/s?source=bing&uid=bd334940-402b-
  "regkey_r": "URL",
  "reg_type": 1,
  "regkey": "HKEY_CURRENT_USER\\Software\\Microsoft\\Internet Explorer\\
},
"time": 1589484205.812375,
"tid": 2796,
"flags": {
  "reg_ty
}
"category":
```



The screenshot shows a Windows 'Find' dialog box. The 'Find what' field contains the path: {80295753-E2DE-4F8D-85B0-B814E1C5FCB3}\\URL. The 'Find Next' button is highlighted. The dialog box is overlaid on the bottom of the JSON log snippet.

This is the same URL that was scanned in the last section. Now it is a part of the Internet Explorer registry, which means Internet Explorer will read this registry and go to the URL.

A similar search for the start page registry reveals something similar.

```
{
  "category": "registry",
  "status": 1,
  "stacktrace": [],
  "api": "RegSetValueExW",
  "return_value": 0,
  "arguments": {
    "key_handle": "0x0000015c",
    "value": "http://search.searchffr.com/?source=bing&uid=bd334940-402b-4323-be0a-20a6895665f",
    "regkey_r": "Start Page",
    "reg_type": 1,
    "regkey": "HKEY_CURRENT_USER\\Software\\Microsoft\\Internet Explorer\\Main\\Start Page"
  },
  "time": 1589484206.484375,
  "tid": 2796,
  "flags": {
```

The above screenshot gives concrete evidence that the malware is changing the start page from the default page to the suspicious URL.

A lookup of the suggestion URL registry reveals its value to be -

"https://ie.search.yahoo.com/os?appid=ie8&command={searchTerms}"

Navigating to this URL downloads an XML file onto the device named "os"

The XML file doesn't reveal much but contains two more urls -

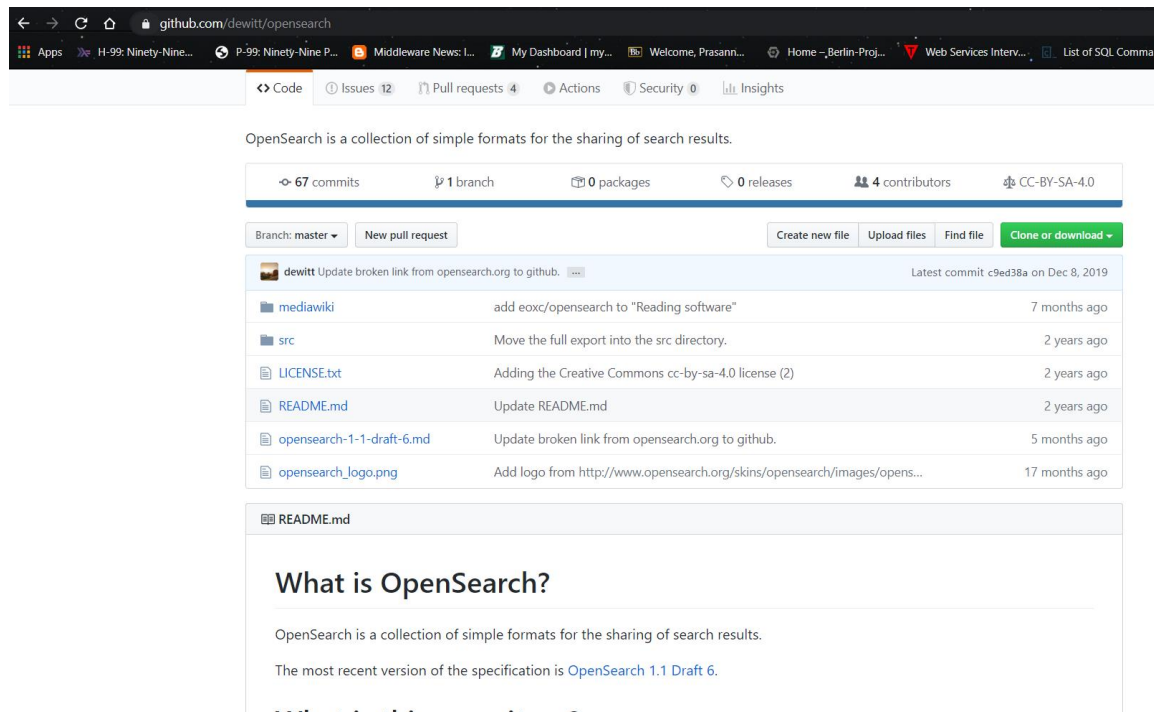
- <https://s.yimg.com/pv/static/img/us-metro-sprite-201210111417.png>
- <http://opensearch.org/searchsuggest2>

The first one is an image -



The image looks very low resolution which clearly suggests that it is not actually being sent over by the real Yahoo.

The second URL redirects to a github page



Thus, the malware is using OpenSearch as well to its benefit.

As the screenshot shows, OpenSearch is a collection of simple formats for the sharing of search results.

Mozilla defines it as - The OpenSearch description format lets a website describe a search engine for itself, so that a browser or other client application can use that search engine.

This definition somewhat aligns to the malware's behavior. It can be inferred that the malware is leveraging OpenSearch techniques for its benefit.

3.Cuckoo Screenshots

Cuckoo provides the powerful way of screenshots to have a look at various points in the malware execution. This can help make sure that the notions built by static and dynamic analysis are correct or not. The screenshots can be retrieved from the remote system using cuckoo's /tasks/screenshots API call

Below are some screenshots from cuckoo during its dynamic analysis of this malware.

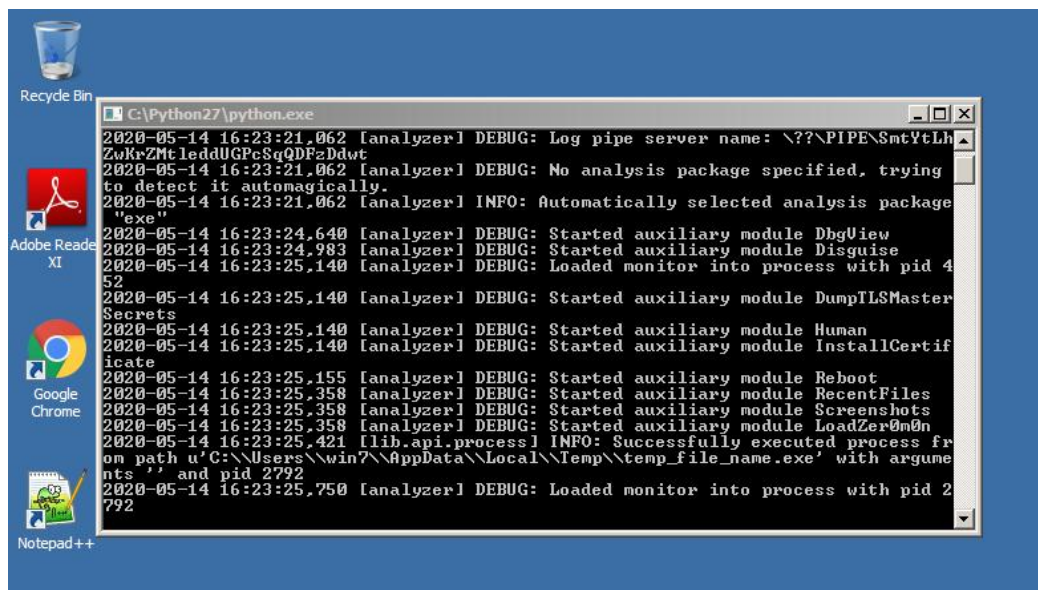


Fig 1. Analysis begins

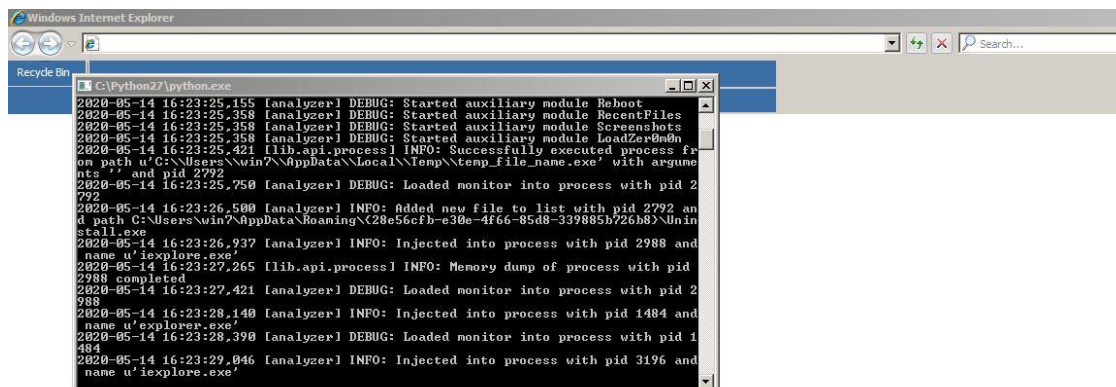


Fig 2. Internet explorer starts

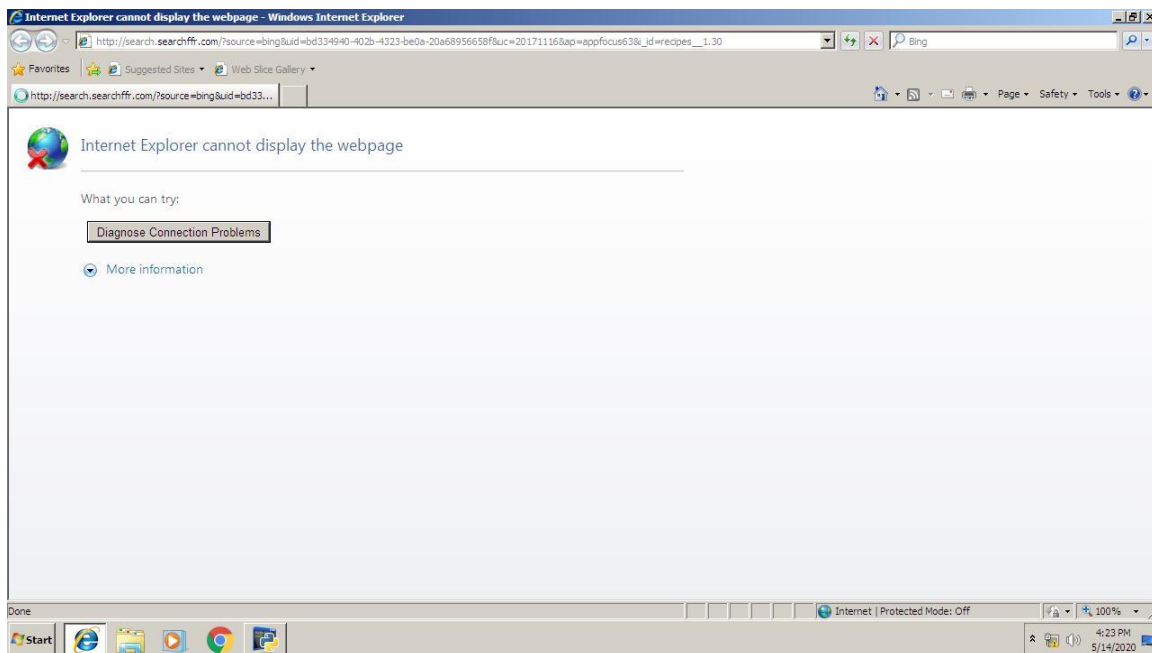


Fig 3. Connection not available

The screenshots show that the malware sample attempts to open an internet explorer instance and tries to navigate to a suspicious website. However, this attempt fails due to the sandbox not being connected to the internet. Still, it is quite clear that it would only lead to an unwanted site.

It is also worth noting that the Command prompt started by the malware does not show up in the screenshots. This is due to the way the cmd command is built.

- It uses NUL – which is used to hide the output or any errors during execution
- It uses /c – which tells cmd to exit as soon as the command is run.

Thus, it is quite possible that the command prompt disappears very fast and a screenshot of it might not have been captured.

Dynamic Analysis Observations

Using cuckoo for dynamic analysis has given a lot of information about this malware. Some important observations are –

1. The process tree of the malware sample has been revealed.
2. The URLs (malicious and normal) that the malware attempts to connect to are found
3. The commands that the process runs in command line are now available for further analysis
4. The malware deletes itself after it has infected the registries and changed internet settings
5. The malware manipulates Internet explorer registries to redirect to unwanted URLs
6. The malware might be leveraging OpenSearch technology for its benefit
7. The malware might download XML file on the system
8. The malware is an Adware which aims to change the settings of the browser to generate traffic on a specific URL.
9. The Adware might be trying to gather search data by redirecting to the malicious URL
10. The type of changes the malware is attempting are clear sign of Browser-hijacking.

Insights

Since, it is now determined that the sample is an Adware, let's look at what exactly are Adware.

Adware are malicious software that automatically display ads whenever connected to the internet. Sometimes, even a connection may not be required as Adware are capable of downloading data to the user's system. The Adware are also capable of collecting marketing data from the user by exporting their search history.

Adware are generally not categorized as a critical threat. However, they are very annoying and can reduce the productivity of the user. They can affect the performance of the browser and it is advised that they should be removed as soon as detected.

Adware make their way to user's system by attaching themselves to freeware. Many of the free software available on the internet come packaged with Adware. This is a very easy way to spread as free software are preferred by a lot of people. Another reason for Adware spreading easily is the fact that people aren't too careful while downloading and installing software. The habit of clicking "Next" till the installation completes introduces the user to

a host of unwanted software. These software are then sometimes too difficult to track and remove.

Let us look at what are the best methods to remove Adware –

1. The first method is to be careful on the internet. Although this is a prevention, still it is the most important way to be safe from malicious software like Adware.
2. Uninstalling suspicious looking applications from control panel
3. Delete suspicious files/folders in common installation directories
4. Registry files cleaning. If analyzed correctly, registry cleaning is a good option
5. Checking the “installed on” in programs and features to check what type of extensions are installed on browsers.
6. Using malware removal tools like Malwarebytes, AdwCleaner etc.
7. Using Junkware removal tools like JRT.

References

1. <https://radare.gitbooks.io/radare2book/content/>
2. <https://cuckoo.readthedocs.io/en/latest/usage/api/>
3. <https://YARA.readthedocs.io/en/v3.4.0/YARAPython.html>
4. <https://malwaretips.com/>
5. <https://github.com/dewitt/opensearch>