sekoia
#CYBERSECURITY

FastIR Collector on advanced threats

How SEKOIA's open source collector can help you detect advanced threats

V 1.5

Author: Paul Rascagnères

29/10/2015

TLP: WHITE

SEKOIA - 16, Place de la Madeleine - 75008 PARIS
Tél. : **01 44 43 54 13** - Fax : **01 47 23 68 14** - E-mail : **contact@sekoia.fr**

www.sekoia.fr

TABLE OF CONTENTS

TABLE OF IMAGES

1. CONTEXT

For years, the CERT SEKOIA helped many customers handle cybersecurity incidents. During these interventions, the incident response team needs to identify compromised systems by performing forensics investigations. To help in this task, the team developed an open-source tool called FastIR Collector.

The 29th of October 2015, during the Hackito Ergo Sum conference, Paul Rascagnères and Sébastien Larinier from the CERT SEKOIA did a talk called "Complex malware & forensics investigation". During this talk, they took several well-known malware cases (such as Uroburos or Babar) to explain:

- the malware's behavior;
- how FastIR Collector could be used to detect them.

This document explains what FastIR Collector is and details all the case studies mentioned during the talk. The purpose is to see the synergy between the different profiles of an incident response team (forensic analyst, reverser, threat intelligence analyst) and the importance of efficient tools and quality Indicator Of Compromise (IOC).

The purpose of this document is not to fully describe the threats or actors of the case studies mentioned. For more information, feel free to contact the CERT SEKOIA by email.

2. FASTIR COLLECTOR

FastIR Collector is an open source project developed by the experts of the CERT SEKOIA. The software can be downloaded on GitHub: http://github.com/SekoiaLab/FastIR_Collector.

The software is developed in Python but it is recommended to use the precompiled binaries available in the `build` directory. These binaries are standalone and do not have specific prerequisites.

The software supports 32 and 64 bits architectures and the following operating systems:

- Windows XP
- Windows Server 2003
- Windows Vista
- Windows 7
- Windows Server 2008
- Windows 8/8.1
- Windows Server 2012

The forensic collector offers the possibility to extract classic artifacts such as memory dump, auto-started software, MFT, MBR, Scheduled tasks, Services… (The list is not exhaustive). But FastIR Collector can also perform smart acquisitions thanks to the filecatcher, certificate filtering or support of Yara rules.

The first part of the name "Fast" was chosen because one of the prerequisite before the beginning of the development was to be able to perform forensic collections as quickly as possible. A standard collection (without filecatcher or dump) takes less than 1''30 minutes on a Windows 7 system.

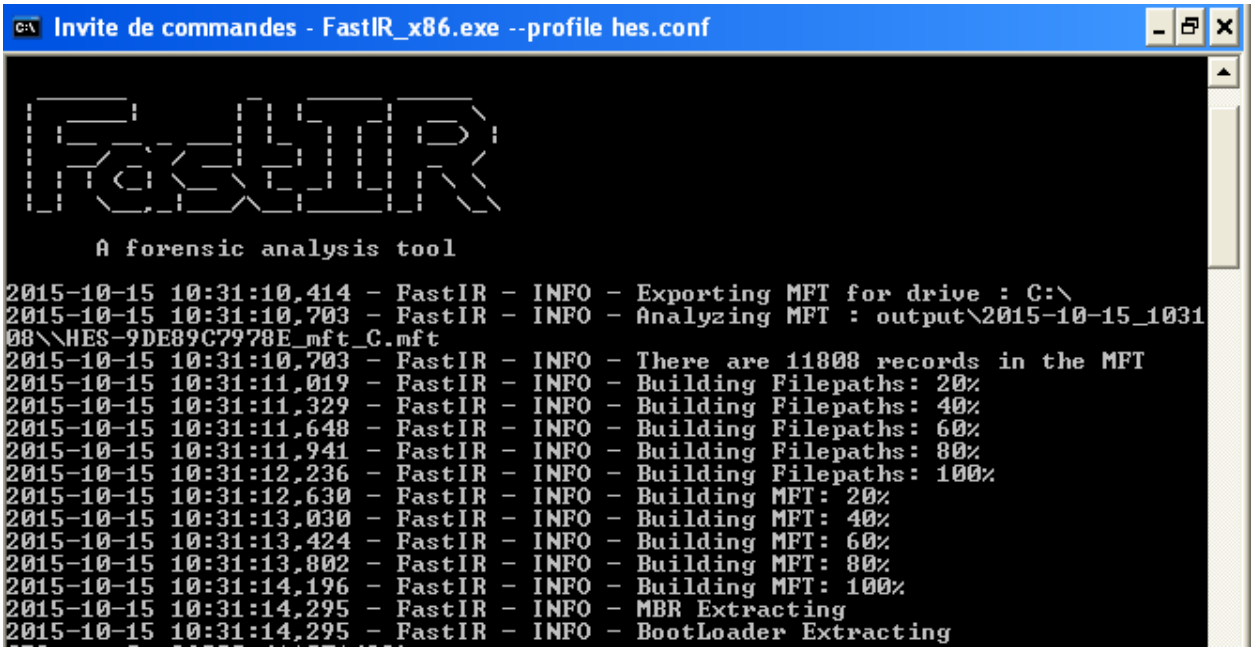

Figure 1: FastIR Collector screenshot

To have more information about the software, the collected data and the configuration, you can look at the documentation, which is available on the GitHub repository in the `Documentation` folder.

For the realization of this article, CERT SEKOIA used the following configuration file (if some changes were made for specific cases, these adjustments are mentioned in this document):

```
[profiles]
packages=fast,filecatcher,dump
[dump]
dump=mft,mbr,ram
mft_export=True
[output]
type=csv
destination=local
dir=output
[filecatcher]
recursively=True
path=c:\tmp|*,c:\temp|*,c:\recycler|*,%WINDIR%|*,%USERPROFILE%|*
mime_filter=application/msword;application/octet-stream;application/x-
archive;application/x-ms-pe;application/x-ms-dos-
executable;application/x-lha;application/x-dosexec;application/x-
elc;application/x-executable, statically linked,
stripped;application/x-gzip;application/x-object, not
stripped;application/x-zip;
mime_zip=application/x-ms-pe;application/x-ms-dos-
executable;application/x-dosexec;application/x-executable, statically
linked, stripped
compare=AND
```

```
size_min=6k
size_max=100M
ext_file=*
zip_ext_file=*
zip=True
[modules]
pe
yara
[pe]
pe_mime_type=application/x-ms-pe;application/x-ms-dos-
executable;application/x-ms-pe;application/x-dosexec;application/x-
executable, statically linked, stripped
filtered_certificates=True
cert_filtered_issuer=issuer;O=Microsoft Corporation|Microsoft Time-
Stamp PCA|Microsoft Time-Stamp PCA Microsoft Windows Verification
PCA|Microsoft Root Authority Microsoft Root Authority Microsoft
Timestamping PCA Microsoft Timestamping PCA Microsoft Root Authority
Microsoft Windows Verification Intermediate PCA Microsoft Root
Authority
cert_filtered_subject=subject;O=Microsoft Corporation|Microsoft Time-
Stamp Service|Microsoft Time-Stamp Service Microsoft Windows|Microsoft
Root Authority Microsoft Root Authority Microsoft Timestamping Service
Microsoft Timestamping Service Microsoft Timestamping PCA Microsoft
Windows Component Publisher Microsoft Windows Verification
Intermediate PCA
[yara]
filtered_yara=False
dir_rules=yara-rules
```

Figure 2: FastIR Collector configuration file

The forensic collection can be automatically uploaded on a Windows Share server or put in the `output` folder in the same directory than the FastIR Collector binary. The collector creates a subfolder called `date_hour` to be able to perform several collections without overwriting an old one.

3. CASE STUDIES

3.1. UROBUROS/TURLA/SNAKE

3.1.1. MALWARE DESCRIPTION

The Uroburos malware (also known as Turla or Snake depending of the name provided by security companies) is a rootkit publicly released in February 2014. The CERT SEKOIA identified samples of this rootkit from 2010, but the incident response team estimates that this threat is probably older.

The particularity of the Uroburos rootkit is the use of two virtual file systems (FAT32 & NTFS) mounted in memory.

The biggest difficulty during an incident response concerning a rootkit is that the forensic investigator cannot trust the compromised systems. Generally, the rootkit changes the system's behavior in order to hide its activities. In the Uroburos case, the driver (.sys file) cannot be seen from the Microsoft Windows Explorer (or MS-DOS prompt). The directory where the driver is installed is hidden.

Hashes of the analyzed sample:

```
Md5: f4f192004df1a4723cb9a8b4a9eb2fbf
Sha1: b24faec08f3ec818c0380145a333251284792995
Sha256: 50edc955a6e8e431f5ecebb5b1d3617d3606b8296f838f0f986a929653d289ed
```

3.1.2. FastIR Collector use and analysis

For the forensic collection of the artifacts to detect Uroburos, the incident response team used the default configuration file mentioned in the chapter "FastIR Collector".

3.1.2.1. Driver identification

With the default configuration, the filecatcher module archives all the PE files available in `%WINDIR%` and `%USERPROFILE%` that are not signed by Microsoft. These collected files are located in the `hostname_files_.zip` file. Here is the content of the archive:

```
paul@lab:~$ unzip -l HES-demo_files_.zip
Archive:  HES-demo_files_.zip
  Length      Date    Time    Name
---------  ---------- -----   ----
   210944  2015-10-08 11:07   WINDOWS/$NtuninstallQ817473$/fdisk.sys
   224768  2007-11-06 19:23   WINDOWS/WinSxS/x86_Microsoft.VC90/msvcm90.dll
    59904  2007-11-06 21:51   WINDOWS/WinSxS/x86_Microsoft.VC90/mfcm90.dll
    59904  2007-11-06 21:51   WINDOWS/WinSxS/x86_Microsoft.VC90/mfcm90u.dll
---------                     -------
   555520                     4 files
```

Figure 3: Filecatcher content for Uroburos

The last three files are legitimate files dropped by Microsoft Visual C++ Redistributable Setup.

The first file is more interesting. It is a driver located in `%WINDIR%\$NtuninstallQ817473$` directory. This directory is not standard for driver installation. The `fdisk.sys` file is the Uroburos rootkit driver. The hash and the VirusTotal link of the binary can be directly found in the `host_filecatcher_.csv` file:

```
[…]
"HES-demo","Filecatcher","2015-10-08
11:07:40.763156","C:\WINDOWS\$NtuninstallQ817473$\fdisk.sys","50edc955a6e8e43
1f5ecebb5b1d3617d3606b8296f838f0f986a929653d289ed ","application/x-ms-dos-
executable","True","False",http://www.virustotal.com/en/file/50edc955a6e8e431
f5ecebb5b1d3617d3606b8296f838f0f986a929653d289ed/analysis
[…]
```

Figure 4: Filecatcher CSV content for Uroburos

Even if the driver cannot be seen by Microsoft Windows Explorer, the FastIR Collector is able to collect it.

3.1.2.2. PERSISTENCE IDENTIFICATION

The FastIR Collector collects several persistence mechanisms. These mechanisms can be viewed in the `hostname_registry_services.csv` file:

```
[…]
"HES-demo","registry_services","2015-10-15
10:28:32","HKEY_LOCAL_MACHINE","System\CurrentControlSet\Services\Ultra3","Im
agePath","VALUE","REG_SZ","\SystemRoot\$NtuninstallQ817473$\fdisk.sys"
[…]
```

Figure 5: Persistence mechanism of Uroburos

The collector perfectly identifies the registry value. Here is two screenshots of the Windows registry editor and Autoruns from Windows Sysinternals. Both of the tools don't identify the persistence mechanism because the rootkit hides the `Ultra3\ImagePath` registry value:

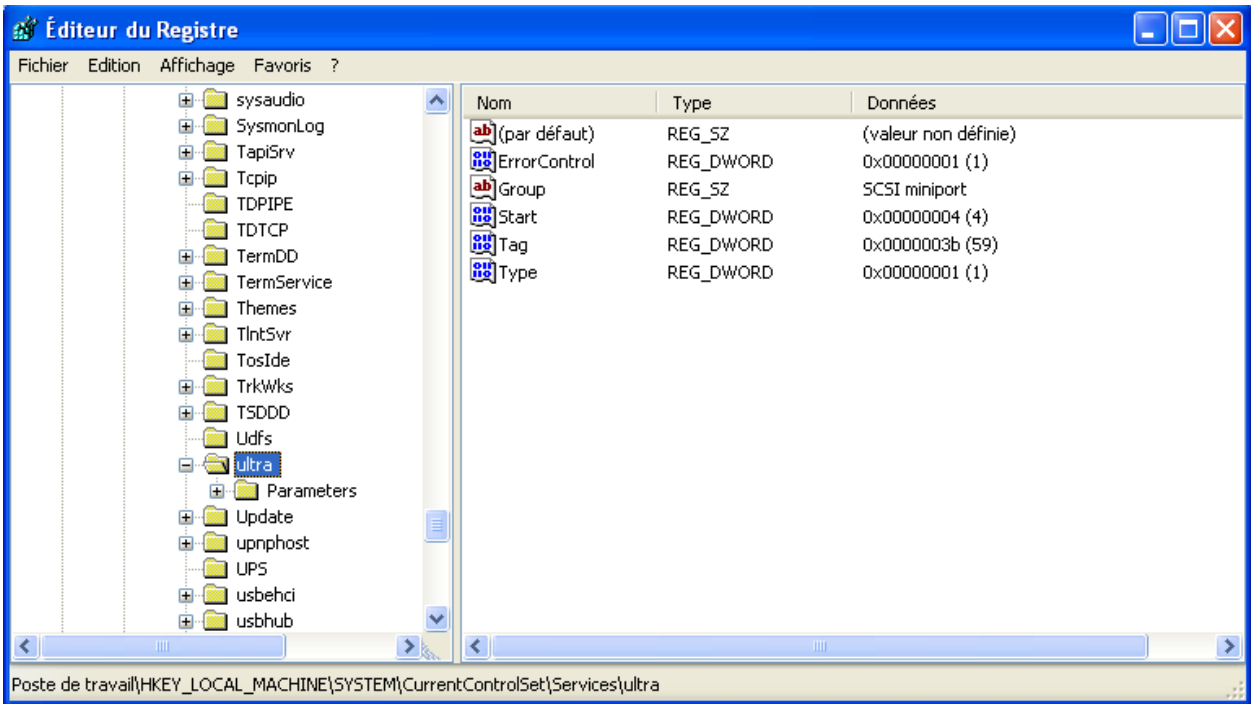

Figure 6: Uroburos persistence mechanism viewed by the Microsoft registry editor

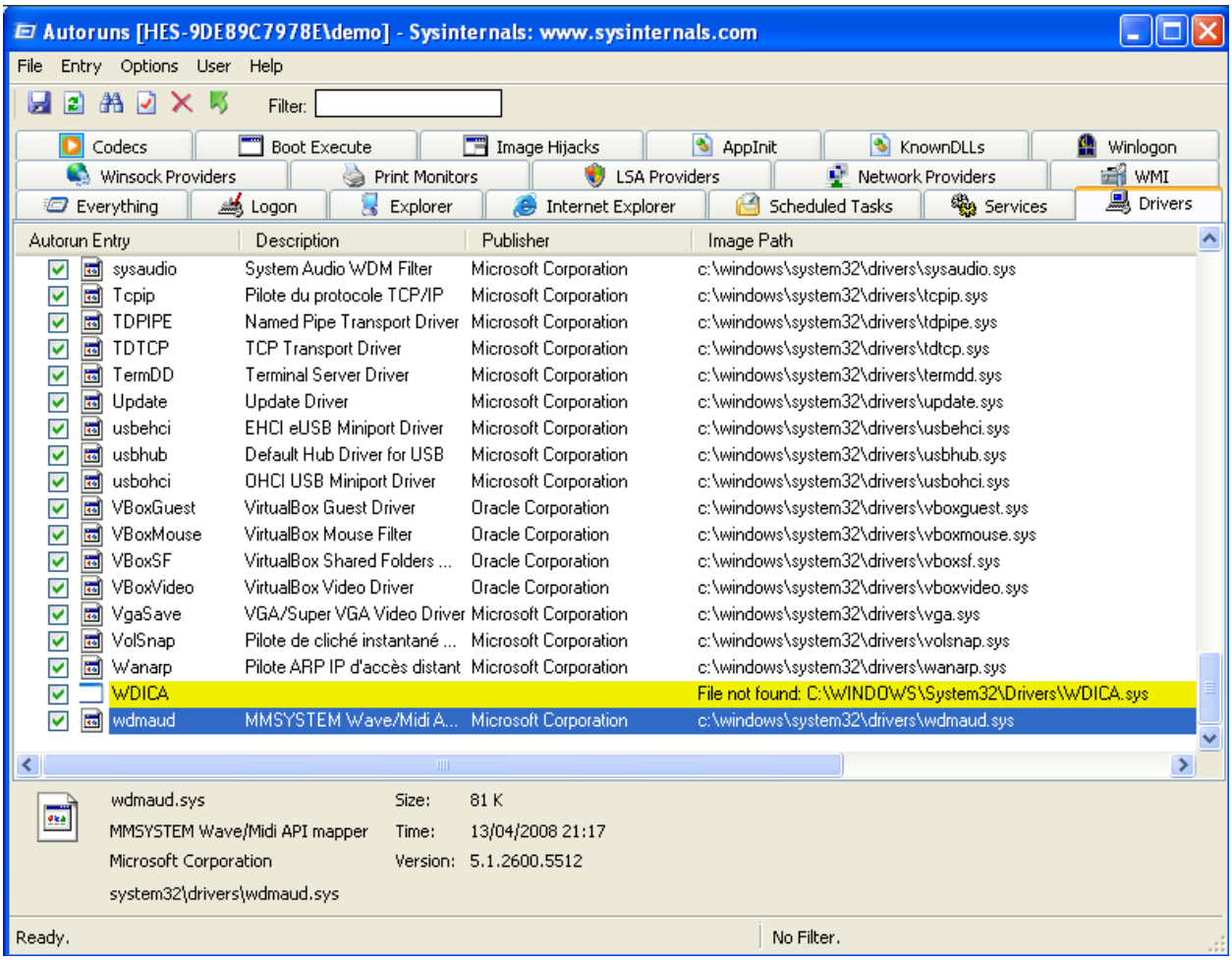

Figure 7: Uroburos persistence mechanism viewed by Autoruns

3.1.2.3. NAMED PIPES IDENTIFICATION

On several papers about Uroburos, the security experts specify the rootkit uses named pipes to communicate between each component of the malware. The named pipes are listed in the `hostname_named_pipes.csv` file:

```
[…]
"HES-demo","named_pipes","\\.\pipe\isapi_http2"
"HES-demo","named_pipes","\\.\pipe\isapi_dg2"
"HES-demo","named_pipes","\\.\pipe\isapi_http"
"HES-demo","named_pipes","\\.\pipe\isapi_dg"
[…]
```

Figure 8: Named pipes used by Uroburos

3.1.2.4. VIRTUAL FILE SYSTEMS

The FastIR Collector collects and analyzes the Windows Prefetch files. The content is available in the `hostname_prefetch.csv` file. On Windows systems, there is a special Prefectch entry called

NTOSBOOT-B00DFAAD.pf. This file shows the list of files that are loaded during the Windows boot process. Some files stored in the virtual filesystem can be identified in the .csv file generated by the collector:

- \DEVICE\RAWDISK1\KLOG
- \DEVICE\RAWDISK1\$MFT
- \DEVICE\RAWDISK1\QUEUE

The KLOG file contains the kernel log generated by the rootkit.
The QUEUE file contains the configuration of the rootkit, the libraries to load in user-land…
The $MFT file is the NTFS partition table of the virtual filesystem.

3.2. COMRAT

3.2.1. MALWARE DESCRIPTION

The ComRAT malware is a RAT (Remote Administration Tool) developed by the same actors than Uroburos mentioned previously. Unlike Uroburos, ComRAT is not a rootkit but a software that runs in user-land.

The particularity of this RAT is the persistence mechanism: the malware hijacks a legitimate COM Object to execute a malicious payload instead of its legitimate code.

Hashes of the analyzed sample:

```
Md5: 51e7e58a1e654b6e586fe36e10c67a73
Sha1: 1f8e7d149bf8d3881a7a3db7c1e4aa0cd4779a2e
Sha256: 87d4edd9d833a41b776bbbbb2ecde0513ae0aa3d228caf3c85d2298c9977e89f
```

3.2.2. FASTIR COLLECTOR USE AND ANALYSIS

For the forensic collection of the artifacts to detect ComRAT, the incident response team used the default configuration file mentioned in the chapter "FastIR Collector".

3.2.2.1. MALWARE IDENTIFICATION

Using the same approach detailed previously, it is easy to identify malicious files thanks to the filecatcher:

```
paul@lab:~$ unzip -l HES-demo_files_.zip
Archive:  HES-demo_files_.zip
  Length      Date    Time    Name
---------  ---------- -----   ----
   260096  2008-04-14 14:00   Documents and Settings/demo/Application
Data/Microsoft/credprov.tlb
    51200  2008-04-14 14:00   Documents and Settings/demo/Application
Data/Microsoft/shdocvw.tlb
   224768  2007-11-06 19:23   WINDOWS/WinSxS/x86_Microsoft.VC90/msvcm90.dll
    59904  2007-11-06 21:51   WINDOWS/WinSxS/x86_Microsoft.VC90/mfcm90.dll
    59904  2007-11-06 21:51   WINDOWS/WinSxS/x86_Microsoft.VC90/mfcm90u.dll
```

```
---------                       -------
   655872                       5 files
```

Figure 9: Filecatcher content for ComRAT

The two malicious binaries are located in `%APPDATA%\Microsoft\`. The `shdocvw.tlb` file is used to load the `credprov.tlb` file.

3.2.2.2. PERSISTENCE IDENTIFICATION

The malware uses an uncommon persistence mechanism by creating the following registry key:

- `HKCU\Software\CLSID\{42aedc87-2188-41fd-b9a3-0c966feabec1}\InprocServer32`

This kind of registry key is not collected by FastIR Collector. In this specific case, the persistence mechanism cannot be identified in FastIR Collector's output.

3.2.2.3. LIBRARY INJECTION

The advantages for a malware developer to hijack a COM Object are:

- to be quiet: no evidence of persistence for several tools;
- to be automatically loaded on every processes that use the hijacked COM Object.

This loading can be identified in the `hostname_processes_dll.csv` file. This file contains the libraries properly loaded by the running processes:

```
[…]
"HES-demo","processes_dll","1420","C:\WINDOWS\Explorer.EXE","C:\Documents and Settings\demo\Application Data\Microsoft\shdocvw.tlb"
"HES-demo","processes_dll","1420","C:\WINDOWS\Explorer.EXE","C:\Documents and Settings\demo\Application Data\Microsoft\credprov.tlb"
[…]
```

Figure 10: Library injection by ComRAT

3.2.2.4. YARA RULES: FROM AGENT.BTZ TO COMRAT

In 2008, the U.S. Central Command network was compromised by a malware called Agent.BTZ. The same year, ThreatExpert published a blog on this topic[1]. The blog explains that the malware registers a windows class called "`zQWwe2esf3456d`". This information can be translated to a simple Yara rule:

```
rule AgentBTZ
{
      strings:
            $threatexpert = "zQWwe2esf3456d"
      condition:
```

[1] http://blog.threatexpert.com/2008/11/agentbtz-threat-that-hit-pentagon.html

```
            $threatexpert
}
```

Figure 11: Agent.BTZ yara rules

FastIR Collector can use this rule during collection. To enable this feature, the `[yara]` stanza needs to be updated with the following content:

```
[yara]
filtered_yara=True
dir_yara=yara-rules
```

Figure 12: Enable yara support for FastIR Collector

Thanks to these parameters, the `hostname_filecatcher.csv` file will be generated with the following contents:

```
[…]
"HES-demo","Filecatcher","2008-04-14 14:00:00","C:\Documents and
Settings\demo\Application
Data\Microsoft\credprov.tlb","4e553bce90f0b39cd71ba633da5990259e185979c2859ec
2e04dd8efcdafe356","AgentBTZ","yara","False",http://www.virustotal.com/en/fil
e/4e553bce90f0b39cd71ba633da5990259e185979c2859ec2e04dd8efcdafe356/analysis
[…]
```

Figure 13: Filecatcher CSV content for ComRAT with yara

The same string is used in Agent.BTZ (from 2008) and ComRAT (from 2014), ComRAT being an evolution of Agent.BTZ, 7 years later…

Thanks to an old Indicator Of Compromise (IOC), it is possible to find a new threat (7 years later). The `yara-rules` folder can contain several yara rules. Do not hesitate to store all the rules created on previous incidents to optimize the detection of infected systems.

3.3. BABAR

3.3.1. MALWARE DESCRIPTION

The Babar malware is a Remote Administration Tool (RAT) disclosed in 2015. The case was highly publicized because several researchers and journalists think that the malware could be developed by the French intelligence Agency.

Hashes of the analyzed sample:

```
Md5: 9fff114f15b86896d8d4978c0ad2813d
Sha1: 27a0a98053f3eed82a51cdefbdfec7bb948e1f36
Sha256: c72a055b677cd9e5e2b2dcbba520425d023d906e6ee609b79c643d9034938ebf
```

3.3.2. FASTIR COLLECTOR USE AND ANALYSIS

For the forensic collection of the artifacts to detect Babar, the incident response team used the default configuration file mentioned in the chapter "FastIR Collector".

3.3.2.1. MALWARE IDENTIFICATION

Like the previous cases, the filecatcher can easily identify the malware in `%APPDATA%\perf_585.dll`. For more information, please refer to the Uroburos or ComRAT cases.

3.3.2.2. PERSISTENCE IDENTIFICATION

The Babar malware uses a persistence mechanism that can be identified in the `hostname_startup.csv` file:

```
[…]
"HES-demo","startup","2015-10-08
11:20:21","HKEY_LOCAL_MACHINE","Software\Microsoft\Windows\CurrentVersion\Run
","MSSecurity","VALUE","REG_SZ","""regsvr32.exe"" /s /n /i ""C:\Documents and
Settings\All Users\Application Data\perf_585.dll"""
[…]
```

Figure 14: Persistence mechanism for Babar

This malware uses a classic `Run` registry key and is executed by `regsvr32.exe`.

3.3.2.3. PROCESS & INJECTION IDENTIFICATION

The malware is simply launched via `regsvr32.exe` and can be listed as a running process. This information is available in the `hostname_processes.csv` file:

```
[…]
"HES-demo","processes","1828","regsvr32.exe",
"""C:\WINDOWS\system32\regsvr32.exe"" /s /n /i ""C:\Documents and
Settings\All Users\Application
Data\perf_585.dll""","C:\WINDOWS\system32\regsvr32.exe"
[…]
```

Figure 15: Babar process

The malware injects itself properly in several other processes. The list can be seen in the `hostname_processes_dll.csv` file:

```
[…]
"HES-demo","processes_dll","1440","C:\WINDOWS\Explorer.EXE","C:\Documents and
Settings\All Users\Application Data\perf_585.dll"
"HES-
demo","processes_dll","1788","C:\WINDOWS\system32\VBoxTray.exe","C:\Documents
and Settings\All Users\Application Data\perf_585.dll"
"HES-
demo","processes_dll","1848","C:\WINDOWS\system32\ctfmon.exe","C:\Documents
and Settings\All Users\Application Data\perf_585.dll"
"HES-
demo","processes_dll","396","C:\WINDOWS\system32\wscntfy.exe","C:\Documents
and Settings\All Users\Application Data\perf_585.dll"
[…]
```

Figure 16: Process injection for Babar

The malicious library is injected in `Explorer.EXE`, `VBoxTray.exe`, `ctfmon.exe` and `wscntfy.exe`. The author of the Babar malware did not implement techniques to try to hide the malicious activity.

3.4. CASPER

3.4.1. MALWARE DESCRIPTION

The Casper malware is a Remote Administration Tool (RAT) that is linked to the previously mentioned Babar. Both RATs share the same developer, several parts of the code being simple copy-pastes.

Hashes of the analyzed sample:

```
Md5: 4d7ca8d467770f657305c16474b845fe
Sha1: e4cc35792a48123e71a2c7b6aa904006343a157a
Sha256: 8e6402c8703e9f10493222a26afeb0fc575bb879d6c82d89c1a79aa75be645d0
```

3.4.2. FASTIR COLLECTOR USE AND ANALYSIS

For the forensic collection of the artifacts to detect Casper, the incident response team used the default configuration file mentioned in the chapter "FastIR Collector".

3.4.2.1. MALWARE IDENTIFICATION

In this case study, the sample is not automatically collected by the filecatcher, but the file can be identified thanks to the `hostname_startup.csv` file:

```
[…]
"HES-demo","startup","2015-10-08
11:30:07","HKEY_LOCAL_MACHINE","Software\Microsoft\Windows\CurrentVersion\Run
","VBOX Audio Interface Device Manager","VALUE","REG_SZ","""C:\Program
Files\Fichiers communs\VBOX Audio Interface Device Manager\aiomgr.exe""
3071006457"
[…]
```

Figure 17: Persistence mechanism for Casper

The malware is located in `%ProgramFiles%`. By default, this directory is not scanned by the filecatcher because too many files not signed by Microsoft are located in this path. The analyst can add the `%ProgramFiles%` value in the path variable of the configuration file or simply copy this file from the collected machine or rebuild the binary from the memory dump.

3.5. POWELIKS

3.5.1. MALWARE DESCRIPTION

The Poweliks malware is a downloader with the particularity of being file-less. The malware is fully stored in the registry. Moreover, the author chose to use only non-ASCII characters in the registry key to hide itself: some tools have difficulties to read these non-ASCII characters (for example Microsoft Regedit).

Hashes of the Poweliks dropper:

```
Md5: 0181850239cd26b8fb8b72afb0e95eac
Sha1: bfa2dc3b9956a88a2e56bd6ab68d1f4f675a425a
Sha256: 4727b7ea70d0fc00f96a28de7fa3d97fa9d0b253bd63ae54fbbf0bd0c8b766bb
```

3.5.2. FastIR Collector use and analysis

For the forensic collection of the artifacts to detect Poweliks, the incident response team used the default configuration file mentioned in the "FastIR Collector" chapter.

3.5.2.1. Malware identification

In this case the malware cannot be identified with the filecatcher because the malware does not drop any files. However, the malware creates a registry key to be persistent (and exists in this case), so the interesting data can easily be identified in the `hostname_startup.csv` file:

```
[…]
"PC-demo","startup","2015-10-08 14:28:18","HKEY_USERS","S-1-5-21-2108495583-
517838646-1409168491-
1000\Software\Microsoft\Windows\CurrentVersion\Run","\x01\x00\x01","VALUE","R
EG_SZ","rundll32.exe javascript:""\..\mshtml,RunHTMLApplication
"";document.write(""\74script
language=jscript.encode>""+(new%20ActiveXObject(""WScript.Shell"")).RegRead("
"HKCU\\software\\microsoft\\windows\\currentversion\\run\\"")+""\74/script>""
)"
"PC-demo","startup","2015-10-08 14:28:18","HKEY_USERS","S-1-5-21-2108495583-
517838646-1409168491-
1000\Software\Microsoft\Windows\CurrentVersion\Run","","VALUE","REG_SZ","#@~^
kXcAAA==W!x^DkKxP^WTcV* ODH   ax    +h,)mDkp64N+1YcJ\dX:s cj+M\n.oHSuP:n
vcTr#IXRKw+ `r!2:JSJ4YO2=zz6C+(NGc^G:JVKo_VGL{JQVBWl^/nbp6Rdn
      Nc#pY;Mx,Fi)mmOm4`n#PDnO!Dx,Ti)8+{q+&pl{xnh~)1Yrpr(Ln^D`J   j1DrwD
Utn^Vr#iStbs+v+Z'W b`DDXPA'mR2X2Cx92      \rDGUs+UYUODbxLdvJ]Ar
      NrDuE*i2{h3J-'/HdhKc'-Ar      NWSdwKh+Md4+^V'--F T'-
[...]
```

Figure 18: Poweliks in registry

The first input contains the technique used by Poweliks to automatically start automatically by using `rundll32.exe` and the second line contains the payload encoded with `jscript.encode()`. The value of the registry key (as seen on the first line) is `\x01\x00\x01`. Here is a screenshot of Microsoft Regedit software when this value is accessed:

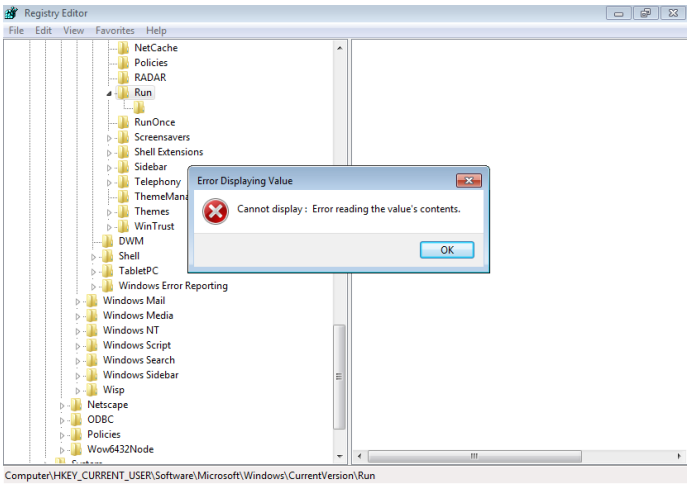

Figure 19: regedit use with Poweliks registry key

3.6. HDROOT

3.6.1. MALWARE DESCRIPTION

The HDRoot piece of code is not exactly a malware. It is a tool used to compromise the Master Boot Record (MBR) of the targeted system to automatically execute a binary (the malware) during the boot process of the machine. The MBR is a boot sector located on the beginning of the hard drive of a system. Usually the MBR contains code to boot the system.

The tool needs two arguments: the binary to execute and the drive to compromise. It is complicated for an incident response team to identify compromised MBR because the MBR is not directly visible from the operating system.

3.6.2. FASTIR COLLECTOR USE AND ANALYSIS

CERT SEKOIA performed missions with this kind of MBR compromise, that is why FastIR Collector can dump the MBR. Moreover, the collector interprets the assembly code contained in it. The raw MBR can be found in the `mbr` file and the assembly code in the `bootLoaderAssemblyCode.txt` file.

Here is an example of the beginning of a clean MBR code (from the `bootLoaderAssemblyCode.txt` file):

```
00000000: 33c0                          XOR AX, AX
00000002: 8ed0                          MOV SS, AX
00000004: bc007c                        MOV SP, 0x7c00
00000007: 8ec0                          MOV ES, AX
00000009: 8ed8                          MOV DS, AX
0000000b: be007c                        MOV SI, 0x7c00
0000000e: bf0006                        MOV DI, 0x600
00000011: b90002                        MOV CX, 0x200
00000014: fc                            CLD
```

Figure 20: Clean MBR extracted by FastIR Collector

The code starts by storing values in several registers (`SS, SP, ES, DS, SI, DI, CS`). Here is the beginning of the MBR code of a compromised system by HDroot:

```
00000000: 33c0                          XOR AX, AX
00000002: 8ed0                          MOV SS, AX
00000004: bc0070                        MOV SP, 0x7000
00000007: eb69                          JMP 0x72
00000009: 8ed8                          MOV DS, AX
0000000b: be007c                        MOV SI, 0x7c00
0000000e: bf0006                        MOV DI, 0x600
00000011: b90002                        MOV CX, 0x200
00000014: fc                            CLD
```

Figure 21: Compromise MBR extracted by FastIR Collector

At the offset 0x7, the allocation was changed by a jump (`JMP`). The standard execution flow is modified to execute a malicious code located at the offset 0x72 of the MBR.

Thanks to this MBR dump, the incident response team is able to clearly and easily identify this compromised MBR.

4. CONCLUSION

This document was created to explain how SEKOIA's FastIR Collector works by showing how the analysts use it in real and complex cases. This tool is used on every Incident Response case. However, not all the features are showcased in this paper (windows event logs or MRU collection for example). These features will be explained in further papers available soon on SEKOIA's blog.

During an incident response, the CERT SEKOIA involves different profiles such as incident coordinators, forensic investigators, reverse engineers and threat intelligence analysts. The purpose is to have complementary profiles to maximize the efficiency of the compromise identification and response.

SEKOIA's incident response team decided to open source FastIR Collector in order to help other incident response teams to perform forensic collection and investigation. The maintainers of the tool are perfectly aware that some elements can be improved and some others are missing. The CERT SEKOIA encourages everyone to contribute to FastIR Collector by opening issues, requests, notifying bugs and proposing patch, features…