ZLAB

Malware Analysis Report: Bladabindi.Dec17

Malware Analysts:

Antonio Pirozzi
Antonio Farina
Luigi Martire

CSE CyberSec Enterprise SPA
Via Giovanni Battista Martini, 6, Roma, Italia
Email: info@csecybsec.com
Website: www.csecybsec.com

18/12/17

Table of Contents

CSE CyberSec Enterprise SPA
Via Giovanni Battista Martini, 6, Roma, Italia
Email: info@csecybsec.com
Website: www.csecybsec.com

Introduction

Surfing the web, we came across a new malware hosted on an “apparently” looking-good web site. As other cases, malware authors mirrored a real site, in this case “www[.]camplace[.]com/live-cams”, and make it seems a legit website to unwise users.

The malicious website (www[.]6th-sense[.]eu), hosts 2 different malware samples:

- *“6thClient.exe”* can be downloaded clicking the pop-up button on the homepage inviting users to download the client indicated on the screen.
- “Firefox.exe” is hosted on the path “*www[.]6th-sense[.]eu/Firefox.exe*”

We analyze both malware samples, from our point of view, it would seem that both malware act as a spyware, in particular, the file “*Firefox.exe*” is also a BOT that waits for specific commands from C&C and sends the appropriate response to it.

Both Malware use *evoria-games[.]eu* as C&C, but, during our analysis *the* communication port of “*6thClient.exe”* was closed.

Both malware also use some evasion mechanism that isn’t sophisticated. Furthermore, it would seem that, the domain, does not to have hosted other malware in the past.

Both Malware belong to the *Bladabind family*. Bladabindi is a Trojan that steals confidential information from the compromised computer. Hackers also use it as a Malware downloader to deliver other payloads. With this malware, cybercriminals could steal

- Your computer name
- Your native country
- OS serial numbers
- Windows user name
- Operating system version
- Stored passwords in chrome
- Stored passwords in Firefox

Along with this report we also provide some YARA rules for those specific malware sample.

Basic static Analysis

File 1

File Name: “Firefox.exe”

MD5	ae7d8cb75d58979b3bb4b43e397d8134
SHA-1	7da98592a32c64f3800f23db0defccc3558f813c
SHA-256	65fb51ff1637ee21a56e94f57dc3cf1cc4c8fedd19c5434470fc2f604cb6c5f2
File Size	1.95 MB

Icon	

Sections

Name	Virtual Address	Virtual Size	Raw Size	Entropy	MD5
.text	8192	355204	355328	7.64	404bf3f3720fa53c7b3a4e57168d53f5
.rsrc	368640	162412	162816	6.26	5b4af263dc864f092e4d7ba98620a58f
.reloc	532480	12	512	0.1	c95fdde4f69824bd87efe453cc57e63c

File 2

File Name: “6thClient.exe”

MD5	2f2378f65a834732769e2889166f471d
SHA-1	ceabca33992adb2f738f7da7de125307c8fa381d
SHA-256	b4b3b8eecc331ffb641a87c2495221e21178f9943afb4e6487d73155055055ba
File Size	1.56 MB
Icon	

Sections

Name	Virtual Address	Virtual Size	Raw Size	Entropy	MD5
.text	8192	539540	539648	7.88	cfdf15fd0d845b0e141afc3a7f7874d1
.rsrc	548864	766674	766976	4.17	d987e0b053adb44b6e2f17ab7b1048f9
.reloc	1318912	12	512	0.1	98c78d18318b9b8fb2078e73d73ab003

Using some static analysis tools, such as PEiD, we discovered that both the malwares are based on .NET Framework and they are written in C#

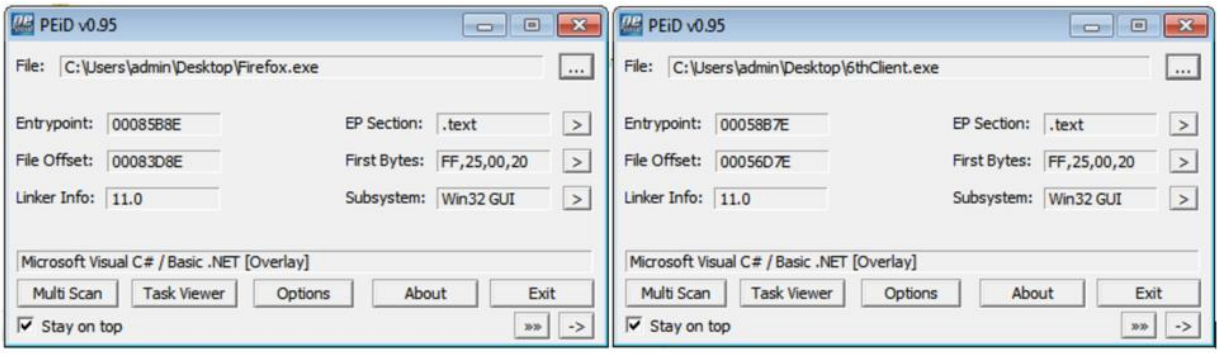


Figure 1 - PEiD views

Behavioral Analysis

Both the samples are retrieved from the site “www[.]6th-sense[.]eu”:

- “6thClient.exe” can be downloaded clicking the pop-up button on the homepage inviting users to download the client indicated on the screen.
- “Firefox.exe” is hosted on the path “www[.]6th-sense[.]eu/Firefox.exe”

The attack vector used by Bladabindi.Dec17 hackers is drive-by download attack. Attackers mirrored the homepage of a “meeting website” in a new domain “www[.]6th-sense[.]eu”.

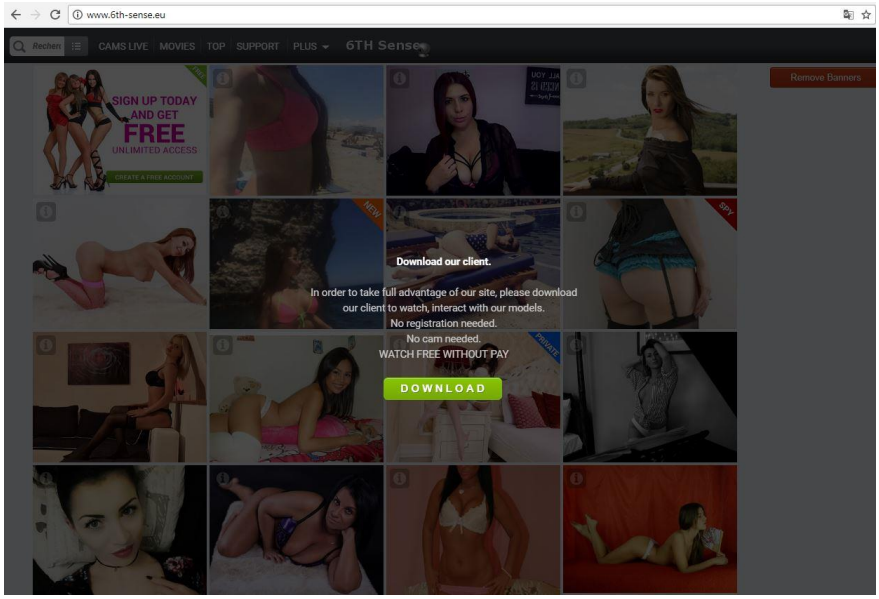


Figure 2 - Homepage of the malicious site

The original website’s url can be visible opening browser developers tools.

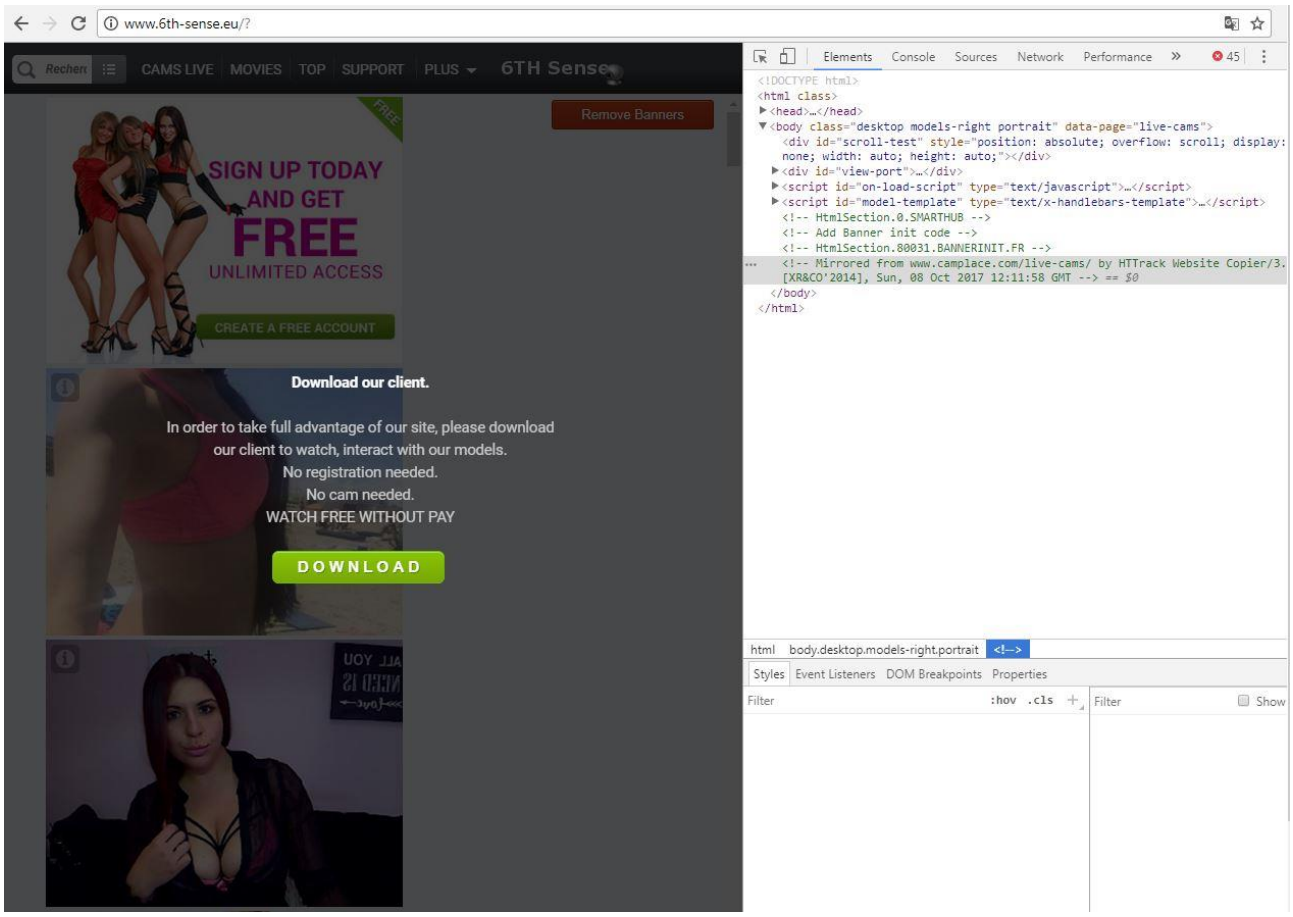


Figure 3 - Html code of the malicious site. On the top right the HTTrack firm.

The above figure shows the tool used to mirror the original page, HTTrack, and the original url “www[.]camplace[.]com/live-cams”.

The malware checks whether it has been executed in virtual host: if it so, it killed itself, otherwise, it continues its malicious activities.

Both samples use the same classic persistence mechanism: they set up a registry key for the automatic startup execution masquerading as a Java Updater process.

Operation	Path	Detail
RegSetValue	HKCU\Software\Microsoft\Windows\CurrentVersion\Run\Java Updater	Type: REG_SZ, Length: 92, Data: C:\Users\admin\Desktop\malwares\6thClient.exe

Figure 4 - Persistence mechanism

Network Analysis

File 1

The malware Firefox.exe contacts the C&C, “evoria-games[.]eu” at the port 491, and creates a TCP connection with the server sending some information about the infected machine.

31084	6230.173125	188.213.28.49	10.0.2.15	TCP	60	491 → 49300 [ACK] Seq=269 Ack=1324 Win=65535 Len=0
31091	6245.817925	188.213.28.49	10.0.2.15	TCP	60	491 → 49300 [PSH, ACK] Seq=269 Ack=1324 Win=65535 Len=2
31092	6245.818067	10.0.2.15	188.213.28.49	TCP	56	49300 → 491 [PSH, ACK] Seq=1324 Ack=271 Win=65535 Len=2
31093	6245.818197	188.213.28.49	10.0.2.15	TCP	60	491 → 49300 [ACK] Seq=271 Ack=1326 Win=65535 Len=0
31096	6252.172821	10.0.2.15	188.213.28.49	TCP	161	49300 → 491 [PSH, ACK] Seq=1326 Ack=271 Win=65535 Len=107
31097	6252.173014	188.213.28.49	10.0.2.15	TCP	60	491 → 49300 [ACK] Seq=271 Ack=1433 Win=65535 Len=0
31101	6260.174032	10.0.2.15	188.213.28.49	TCP	112	49300 → 491 [PSH, ACK] Seq=1433 Ack=271 Win=65535 Len=58
31102	6260.174238	188.213.28.49	10.0.2.15	TCP	60	491 → 49300 [ACK] Seq=271 Ack=1491 Win=65535 Len=0
31103	6260.919039	188.213.28.49	10.0.2.15	TCP	60	491 → 49300 [PSH, ACK] Seq=271 Ack=1491 Win=65535 Len=2
31104	6260.931241	10.0.2.15	188.213.28.49	TCP	56	49300 → 491 [PSH, ACK] Seq=1491 Ack=273 Win=65535 Len=2
31105	6260.931781	188.213.28.49	10.0.2.15	TCP	60	491 → 49300 [ACK] Seq=273 Ack=1493 Win=65535 Len=0
31110	6276.191226	188.213.28.49	10.0.2.15	TCP	60	491 → 49300 [PSH, ACK] Seq=273 Ack=1493 Win=65535 Len=2
31111	6276.191388	10.0.2.15	188.213.28.49	TCP	56	49300 → 491 [PSH, ACK] Seq=1493 Ack=275 Win=65535 Len=2
31112	6276.191579	188.213.28.49	10.0.2.15	TCP	60	491 → 49300 [ACK] Seq=275 Ack=1495 Win=65535 Len=0
31118	6284.227843	10.0.2.15	188.213.28.49	TCP	161	49300 → 491 [PSH, ACK] Seq=1495 Ack=275 Win=65535 Len=107
31119	6284.228043	188.213.28.49	10.0.2.15	TCP	60	491 → 49300 [ACK] Seq=275 Ack=1602 Win=65535 Len=0
31122	6292.160068	188.213.28.49	10.0.2.15	TCP	60	491 → 49300 [PSH, ACK] Seq=275 Ack=1602 Win=65535 Len=2
31123	6292.160202	10.0.2.15	188.213.28.49	TCP	56	49300 → 491 [PSH, ACK] Seq=1602 Ack=277 Win=65535 Len=2
31124	6292.160398	188.213.28.49	10.0.2.15	TCP	60	491 → 49300 [ACK] Seq=277 Ack=1604 Win=65535 Len=0
31130	6307.548129	188.213.28.49	10.0.2.15	TCP	60	491 → 49300 [PSH, ACK] Seq=277 Ack=1604 Win=65535 Len=2
31131	6307.548266	10.0.2.15	188.213.28.49	TCP	56	49300 → 491 [PSH, ACK] Seq=1604 Ack=279 Win=65535 Len=2
31132	6307.548422	188.213.28.49	10.0.2.15	TCP	60	491 → 49300 [ACK] Seq=279 Ack=1606 Win=65535 Len=0

Wireshark · Segui flusso TCP (tcp.stream eq 7) · wireshark_8C6F89A6-C46E-4E27-A5BE-696A9DCAC05A_20170927105008_a02496


```
142.llnyanWVRCRTAzMTJfNDBGODc0M0Y=nyanADMIN-PCnyanadminnyan17-09-27nyannyanWin 7 Ultimate SP1 x64nyanNonyanNot
FoundnyanNot FoundnyanNot
Foundnyan123.infnyanWVRCRTAzMTINCmV2b3JpYS1nYW1lcy5ldTo0OTENCkRlc2t0b3ANCkZpcmVmb3gtY2xlYW5lZC5leGUNCkZhbHNlDQpGYWxzZ
Q0KRmFsc2UNCkZhbHNl0.0.71.actnyanQ2F0dHVyYSBkYSBDb25uZXNzaW9uZSBhbGxhIHJldGUgbG9jYWxlIChMQU4pAA==0.0.0.0.55.actnyanKk
Nvbm5lc3Npb25lIGFsbGEgcmV0ZSBsb2NhbGUgKExBTikA0.0.0.0.0.0.0.0.0.0.167.actnyanV2lyZXNoYXJrIMK3IFNlZ3VpIGZsdXNzbyBUQ1Ag
KHRjcC5zdHJlYW0gZXEgNykgwrcgd2lyZXNoYXJrXzhDNkY4OUE2LUM0NkUtNEUyNy1BNUJFLTY5NkE5RENBQzA1QV8yMDE3MDkyNzEwNTAwOF9hMDI0O
TYA0.0.0.0.43.actnyanU2VuemEgbm9tZSAtIEJsb2NjbyBub3RlAA==31.actnyanTm90ZXBhZCsrIHVwZGF0ZQA=0.0.55.actnyanKm5ldyAxIC0g
Tm90ZXBhZCsrIFtBZG1pbmlzdHJhdG9yXQA=43.actnyanU2VuemEgbm9tZSAtIEJsb2NjbyBub3RlAA==0.0.23.actnyanU29zdGl0dWlzY2kA0.0.0
.0.167.actnyanV2lyZXNoYXJrIMK3IFNlZ3VpIGZsdXNzbyBUQ1AgKHRjcC5zdHJlYW0gZXEgNykgwrcgd2lyZXNoYXJrXzhDNkY4OUE2LUM0NkUtNEU
yNy1BNUJFLTY5NkE5RENBQzA1QV8yMDE3MDkyNzEwNTAwOF9hMDI0OTYA0.0.55.actnyanKkNvbm5lc3Npb25lIGFsbGEgcmV0ZSBsb2NhbGUgKExBTi
kA0.0.0.0.11.actnyanAA==55.actnyanKkNvbm5lc3Npb25lIGFsbGEgcmV0ZSBsb2NhbGUgKExBTikA0.0.0.0.0.0.0.0.0.0.0.0.31.actnyanZ
G5TcHkgMy4wLjIgKHg4NikA0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.55.actnyan
KkNvbm5lc3Npb25lIGFsbGEgcmV0ZSBsb2NhbGUgKExBTikA0.0.0.0.103.actnyanUHJvY2VzcyBFeHBsb3JlciAtIFN5c2ludGVybmFsczogd3d3Ln
N5c2ludGVybmFscy5jb20gW2FkbWluLVBDXGFkbWluXQA=0.0.55.actnyanQ3JlYXRlIGR1bXAgb2YgRmlyZWZveC1jbGVhbmVkLmV4ZQA=0.0.103.a
ctnyanUHJvY2VzcyBFeHBsb3JlciAtIFN5c2ludGVybmFsczogd3d3LnN5c2ludGVybmFscy5jb20gW2FkbWluLVBDXGFkbWluXQA=47.actnyanSHhEI
C0gW0ZpcmVmb3gtY2xlYW5lZC5leGVdAA==0.0.0.0.103.actnyanUHJvY2VzcyBFeHBsb3JlciAtIFN5c2ludGVybmFsczogd3d3LnN5c2ludGVybmF
scy5jb20gW2FkbWluLVBDXGFkbWluXQA=0.0.31.actnyanZG5TcHkgMy4wLjIgKHg4NikA0.0.0.0.0.0.0.0.0.0.0.0.47.actnyanZG5TcHkgMy4w
LjIgKHg4NiwgRGVidWdnaW5nKQA=0.0.0.0.0.0.0.0.31.actnyanZG5TcHkgMy4wLjIgKHg4NikA47.actnyanZG5TcHkgMy4wLjIgKHg4NiwgRGVid
WdnaW5nKQA=0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.11.actnyanAA==55.actnyanKkNvbm5lc3Npb25lIGFsbGEgcm
V0ZSBsb2NhbGUgKExBTikA0.0.0.0.167.actnyanV2lyZXNoYXJrIMK3IFNlZ3VpIGZsdXNzbyBUQ1AgKHRjcC5zdHJlYW0gZXEgNykgwrcgd2lyZXNo
YXJrXzhDNkY4OUE2LUM0NkUtNEUyNy1BNUJFLTY5NkE5RENBQzA1QV8yMDE3MDkyNzEwNTAwOF9hMDI0OTYA0.0.47.actnyanZG5TcHkgMy4wLjIgKHg
4NiwgRGVidWdnaW5nKQA=0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.
0.0.167.actnyanV2lyZXNoYXJrIMK3IFNlZ3VpIGZsdXNzbyBUQ1AgKHRjcC5zdHJlYW0gZXEgNykgwrcgd2lyZXNoYXJrXzhDNkY4OUE2LUM0NkUtNE
UyNy1BNUJFLTY5NkE5RENBQzA1QV8yMDE3MDkyNzEwNTAwOF9hMDI0OTYA0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0
.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.55.actnyanKkNvbm5lc3Npb25lIGFsbGEgcmV0ZSBsb2NhbGUgKExBTikA0.0.0.0.0.0.0.0.14
3.actnyanV2lyZXNoYXJrIMK3IFBhY2NoZXR0byAxNDUwIMK3IHdpcmVzaGFya184QzZGODlBNi1DNDZFLTRFMjctQTVCRS02OTZBOURDQUMwNUFfMjAx
NzA5MjcxMDUwMDhfYTAyNDk2AA==0.0.55.actnyanKkNvbm5lc3Npb25lIGFsbGEgcmV0ZSBsb2NhbGUgKExBTikA0.0.0.0.0.0.0.0.39.actnyanU
2xhY2sgLSBHb29nbGUgQ2hyb21lAA==47.actnyanTnVvdmEgc2NoZWRhIC0gR29vZ2xlIENocm9tZQA=0.0.39.actnyanU2xhY2sgLSBHb29nbGUgQ2
hyb21lAA==0.0.0.0.43.actnyanemxhYiBTbGFjayAtIEdvb2dsZSBDaHJvbWUA71.actnyanYW50b25pb2ZhcmluYTE3MDIgfCB6bGFiIFNsYWNrIC0
gR29vZ2xlIENocm9tZQA=0.0.11.actnyanAA==143.actnyanV2lyZXNoYXJrIMK3IFBhY2NoZXR0byAxNDg3IMK3IHdpcmVzaGFya184QzZGODlBNi1
DNDZFLTRFMjctQTVCRS02OTZBOURDQUMwNUFfMjAxNzA5MjcxMDUwMDhfYTAyNDk2AA==0.0.11.actnyanAA==71.actnyanYW50b25pb2ZhcmluYTE3
MDIgfCB6bGFiIFNsYWNrIC0gR29vZ2xlIENocm9tZQA=0.0.143.actnyanV2lyZXNoYXJrIMK3IFBhY2NoZXR0byAxNDg3IMK3IHdpcmVzaGFya184Qz
ZGODlBNi1DNDZFLTRFMjctQTVCRS02OTZBOURDQUMwNUFfMjAxNzA5MjcxMDUwMDhfYTAyNDk2AA==0.0.0.0.55.actnyanKkNvbm5lc3Npb25lIGFsb
GEgcmV0ZSBsb2NhbGUgKExBTikA
```

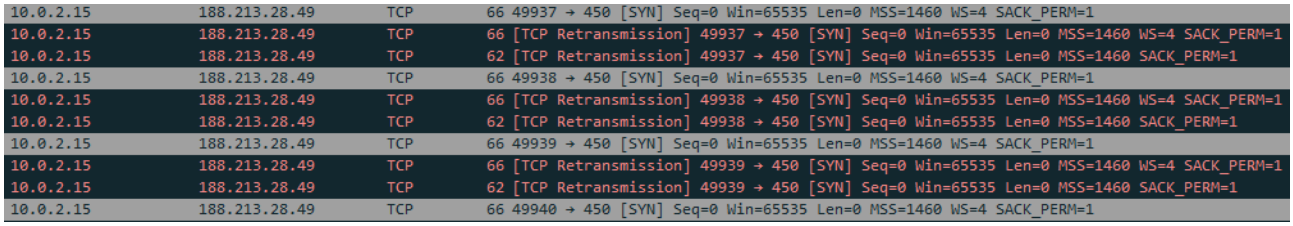
Analyzing the TCP stream, we can see the communication session performed by malware with the C&C:

1. The fist row shows the PC's name, User's name and the OS's version.
2. There are two recurrent words: "nyan" and "act"
    a. the first word represents a separator among the information sent to the C2C
    b. the second one represents the category of the information sent by the bot. in this case it is the 'action' performed by the host, in particular it is the name of the window in foreground
3. In the middle we can see some strings coded in Base64. These strings represent the window's title in foreground.

The C2C acknowledges the result sending the number Zero to the bot, probably this value indicates that there are no commands to do execute on the host.

File 2

The malware 6thClient.exe contacts the same server, but at a different port, 450. During the analysis this port resulted closed.



10.0.2.15	188.213.28.49	TCP	66 49937 → 450 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=4 SACK_PERM=1
10.0.2.15	188.213.28.49	TCP	66 [TCP Retransmission] 49937 → 450 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=4 SACK_PERM=1
10.0.2.15	188.213.28.49	TCP	62 [TCP Retransmission] 49937 → 450 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 SACK_PERM=1
10.0.2.15	188.213.28.49	TCP	66 49938 → 450 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=4 SACK_PERM=1
10.0.2.15	188.213.28.49	TCP	66 [TCP Retransmission] 49938 → 450 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=4 SACK_PERM=1
10.0.2.15	188.213.28.49	TCP	62 [TCP Retransmission] 49938 → 450 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 SACK_PERM=1
10.0.2.15	188.213.28.49	TCP	66 49939 → 450 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=4 SACK_PERM=1
10.0.2.15	188.213.28.49	TCP	66 [TCP Retransmission] 49939 → 450 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=4 SACK_PERM=1
10.0.2.15	188.213.28.49	TCP	62 [TCP Retransmission] 49939 → 450 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 SACK_PERM=1
10.0.2.15	188.213.28.49	TCP	66 49940 → 450 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=4 SACK_PERM=1

Advanced Analysis

Using specific tools to analyze .NET applications we decompiled the bot and we found much more information about it.

Evasion techniques

As all the sophisticated malware it applies some evasion techniques in order to avoid the detection and the analysis:

```
namespace jheoOAZkspGFuVN
{
    // Token: 0x02000004 RID: 4
    internal class phKzNiCCfLUrs
    {
        // Token: 0x06000008 RID: 8 RVA: 0x00002358 File Offset: 0x00000558
        private static void KNeHcCSGAWDHijvldfi()
        {
            try
            {
                using (ManagementObjectSearcher managementObjectSearcher = new ManagementObjectSearcher("Select * from
                  Win32_ComputerSystem"))
                {
                    using (ManagementObjectCollection managementObjectCollection = managementObjectSearcher.Get())
                    {
                        using (ManagementObjectCollection.ManagementObjectEnumerator enumerator =
                          managementObjectCollection.GetEnumerator())
                        {
                            while (enumerator.MoveNext())
                            {
                                ManagementObject managementObject = (ManagementObject)enumerator.Current;
                                string text = managementObject["Manufacturer"].ToString().ToLower();
                                if ((text == "microsoft corporation" && managementObject["Model"].ToString().ToUpperInvariant
                                 ().Contains("VIRTUAL")) || text.Contains(LrLEXuWt.XqXRheg("erawmv")) || managementObject
                                 ["Model"].ToString() == LrLEXuWt.XqXRheg("xoBlautriV"))
                                {
                                    Environment.Exit(1);
                                }
                            }
                        }
                    }
                }
            }
            catch
            {
            }
        }
```

Figure 5 - Evasion techniques of the malwares

In the above figure we can see that the malware makes a query to a particular Windows Database:

“Select * from Win32_ComputerSystem”

The result of this query is a table where it checks the presence of some keywords such as: “VirtualBox” and “vmware” written in a reverse order.

Another evasion technique is checking the identity of the system is equals to “PSPUBWS-PC\\PSPUBWS” representing a particular virtual machine used by some category of sandboxes.

```
        if (WindowsIdentity.GetCurrent().Name.ToString() == "PSPUBWS-PC\\PSPUBWS")
        {
            Environment.Exit(1);
        }
```

Figure 6 - Another evasion technique

Both the malwares contain respectively two payloads which are executed at runtime by the initial dropper. These payloads are initially compressed and cyphered using AES algorithm. Each payload is encrypted using different hardcoded keys.

```
// Token: 0x04000003 RID: 3
private static string KeyPass =
  "nqRHhvUUyjjLgctOWKPyuEoBZHtrQHLhQROjfnsOexeBfoTVTXwtrUEPXAWAsfZuFEwShPSyAtUVVvNiXyEQdbmLfroLfOAPoipmTniDEkIxFSivdaEhzhMgpnSGnezvuOA
  dZpGnzYKQFUeJqtHnEKWnmlnkqLXWvHULXcDlCBzNBBaMOClTXSipcSXXWAFLCpbSHcBqpOycAFDCVPISCJeescgiJRWARhohjQRhqQIFSvFVAZHCXDIC";
```

```
// Token: 0x04000002 RID: 2
private static string Key = "HvOBzptBTQmAQrfCaofVDjqVfiJTMGRgBbMJJVDhMyNMBoqbJuRRnGrchaPWyeAuQTqsdUcAncHaGauLiPq";
```

```
MemoryStream memoryStream = new MemoryStream();
AesManaged aesManaged = new AesManaged();
aesManaged.KeySize = 256;
aesManaged.BlockSize = 128;
Rfc2898DeriveBytes rfc2898DeriveBytes = new Rfc2898DeriveBytes(bytes, salt, iterations);
aesManaged.Key = rfc2898DeriveBytes.GetBytes((int)Math.Round(Math.Round((double)aesManaged.KeySize / 8.0)));
aesManaged.IV = rfc2898DeriveBytes.GetBytes((int)Math.Round(Math.Round((double)aesManaged.BlockSize / 8.0)));
aesManaged.Mode = CipherMode.CBC;
CryptoStream cryptoStream = new CryptoStream(memoryStream, aesManaged.CreateDecryptor(), CryptoStreamMode.Write);
cryptoStream.Write(encodedBytes, 0, encodedBytes.Length);
cryptoStream.Close();
return memoryStream.ToArray();
```

Figure 7 – The keys and the algorithm used by the dropper to decipher the other payloads.

In the final payload there is a section where are some static variables that enable some features of the malware. During the analysis they are set to false, but probably these could be modified at runtime in order to customize the behavior of the malware by the C&C.

```
// Token: 0x0400000A RID: 10
public static bool BD = Conversions.ToBoolean("False");

// Token: 0x0400000B RID: 11
public static bool Idr = Conversions.ToBoolean("False");

// Token: 0x0400000C RID: 12
public static bool IsF = Conversions.ToBoolean("False");

// Token: 0x0400000D RID: 13
public static bool Isu = Conversions.ToBoolean("False");
```

Figure 8 - Static Boolean variables used to define the malicious behavior

Yara Rules

```
import "pe"

rule Firefox_Executable {

	meta:
	  description = "Yara Rule for Firefox.exe of Bladabindi.Dec17"
	  author = "CSE CybSec Enterprise - Z-Lab"
	  last_updated = "2017-12-14"
	  tlp = "white"
	  category = "informational"

	strings:
		$a = "Select * from Win32_ComputerSystem" wide
		$b = "erawmv" wide
		$c = "xoBlautriV" wide

	condition:
		all of them and
		pe.version_info["ProductName"] contains "MSI"
		and pe.sections[2].virtual_size == 12
}



rule SixthClient_Executable {

	meta:
	  description = "Yara Rule for 6thClient.exe of Bladabindi.Dec17"
	  author = "CSE CybSec Enterprise - Z-Lab"
	  last_updated = "2017-12-14"
	  tlp = "white"
	  category = "informational"

	strings:
		$a = "UeXPbtqnJyQJFhz.mp3"
		$b = "aUulTmscifNCReV.mp3"

	condition:
		all of them and
		pe.version_info["ProductName"] contains "shell"
		and pe.sections[2].virtual_size == 12
}
```