UNIVERSITY OF TORONTO

MUNK SCHOOL OF GLOBAL AFFAIRS

Join the Global Conversation

The Citizen Lab

Research Brief
Number 06 – June 2012

Syrian Activists Targeted with BlackShades Spy Software

By Morgan Marquis-Boire and Seth Hardy

FINDINGS

The use of remote surveillance software against activists has been a feature of the ongoing conflict in Syria. In February 2012, CNN reported that "Computer spyware is the newest weapon in the Syrian conflict". Since then numerous electronic campaigns targeting Syrian activists have been observed. These have included: a phishing campaign involving the compromise of a high profile Syrian opposition figure; malware targeting activists by claiming to be documents regarding the foundation of a Syrian revolution leadership council; and, malware purporting to be a plan to assist the city of Aleppo.

The majority of these attacks have involved the use of Dark Comet RAT. Remote Administration Tools (RAT) provide the ability to remotely survey the electronic activities of a victim by keylogging, remote desktop viewing, webcam spying, audio-eavesdropping, data exfiltration, and more.

The use of Dark Comet in this conflict has been well documented. This RAT was the toolkit used in the malware reported on by CNN and also in the campaigns using fraudulent revolutionary documents.

In addition to Dark Comet, we have seen the use of Xtreme RAT reported on by the Electronic Frontier Foundation (EFF) and F-Secure.

Today, the EFF and Citizen Lab report on the use of a new toolkit by a previously observed attacker. This actor has been circulating malware which surreptitiously installs BlackShades RAT on victims machines. This RAT is a commercial tool which advertises the following:

"BlackShades Remote Controller also provides as an efficient way of turning your machine into a surveillance/spy-device or to spy on a specific system."

It is being distributed via the compromised Skype accounts of Syrian activists in the form of a ".pif" file purporting to be an important new video.

```
0d1bd081974a4dcdeee55f025423a72b new_new .pif
```

On execution the following files are dropped:

```
C:\Documents and Settings\Administrator\Templates\VSCover.exe
md5: 291ce2c51e5ea57b571d6610e1d324f9
C:\DOCUME~1\ADMINI~1\LOCALS~1\Temp\local3.exe
md5: 78902e074a7ed514e0f5dca584cd05a5
C:\DOCUME~1\ADMINI~1\LOCALS~1\Temp\D3D8THK.exe
md5: 0d1bd081974a4dcdeee55f025423a72b
```

A keylogger file is then created in a the user's temporary directory:

```
C:\DOCUME~1\ADMINI~1\LOCALS~1\Temp\data.dat
```

Note that 'D3D8THK.exe' is copy of the original file. 'local3.exe' appears to be a version of AppLaunch.exe, the Microsoft ClickOnce Launcher.
The following registry entries are created to allow the dropped programs through the firewall.

```
HKLM\SYSTEM\ControlSet001\Services\SharedAccess\Parameters\FirewallPolicy
\StandardProfile\AuthorizedApplications\List\C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727
\AppLaunch.exe:“C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727
\AppLaunch.exe:*:Enabled:Windows Messanger”

HKLM\SYSTEM\ControlSet001\Services\SharedAccess\Parameters\FirewallPolicy
\StandardProfile\AuthorizedApplications\List\C:\DOCUME~1\user\LOCALS~1\Temp\local3.exe:
“C:\DOCUME~1\user\LOCALS~1\Temp\local3.exe:*:Enabled:Windows Messanger”

HKLM\SYSTEM\CurrentControlSet\Services\SharedAccess\Parameters\FirewallPolicy
\StandardProfile\AuthorizedApplications\List\C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727
\AppLaunch.exe:“C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727
\AppLaunch.exe:*:Enabled:Windows Messanger”

HKLM\SYSTEM\CurrentControlSet\Services\SharedAccess\Parameters\FirewallPolicy
\StandardProfile\AuthorizedApplications\List\C:\DOCUME~1\user\LOCALS~1\Temp\local3.exe:
“C:\DOCUME~1\user\LOCALS~1\Temp\local3.exe:*:Enabled:Windows Messanger”
```

Note the misspelling of “Messenger” as “Messanger”.
“VSCover.exe” is added to autoruns to enable persistence.

```
HKU\s-1-5-21-1177238915-1336601894-725345543500\software\microsoft\windows
\currentversion\run\Microsoft® Windows® Operating System C:\Documents and
Settings\Administrator\Templates\VSCover.exe REG_SZ 0
```

VSCover.exe contains “Libra” (30209 bytes) as a .NET resource which is encrypted using a weak method. Decryption is possible using the key stored internally. Once decrypted, it is loaded as a .NET assembly and the Piept() function is called.[1]
One characteristic stands out in the Class1.Main() function: invoking an obfuscated function in a resource.

```
Class1.Main: Resource12.Method.Invoke(null, null)
```

Tracing the value of Resource12.Method provides:

```
Resource8.encryptedassembly = Resource13.decrypt(Libra,
 “zuk65x7F0h8034E2KJ8rkjZ2BudbdqD”);
Resource10.Appdomainn = AppDomain.CurrentDomain.Load(Resource8.encryptedassembly);
Resource11.Typee = Resource10.Appdomainn.GetType(“Libra.Mameloane”);
Resource12.Method = Resource11.Typee.GetMethod(“Piept”);
```

Substituting out the functions and variables, the decryption routine looks like this:

```
public static byte[] decrypt(byte[] x, string y)
{
reverse(x);
Resource1.cdfsdfsf = x[x.length - 1];
Resource3.kkuythfgh = Encoding.Defaults.GetBytes(y);
Resource2.cfhtfgjdty = new byte[x.length + 1];
int z = 0;
while (z <= (x.Length – 1))
{
Resource2.cfhtgjdty[z] = x[z] ^ Resource1.cdfsdfsf ^ Resource3.kuythfgh[Resource4.Bravo];
Array.Reverse(Resource3.kuytfhfgh);
if (Resource4.Bravo == (Resource3.kuytfhfgh.Length – 1))
{
Resource4.Bravo = 0;
}
else
{
Resource4.Bravo++;
}
z++;
}
Array.Resize(Resource2.cfhfgjdty, Resource2.cfhfgjdty.Length – 2);
return Resource2.cfhtfgjdty;
}
```

Libra has 3 classes: Mameloane[2], RuntimePortableExecutable (implementing PE headers, methods for binaries to run in WinXP and Win7), and Xorxorxor (implementing decryption methods including XOR). Libra also contains the VSCover.exe program itself as an unencrypted resource.

As seen above, VSCover will call the Mameloane.Piept() function in Libra. Piept() will reload the main D3D8THK program, splitting it by the string “p8D-T-4M0t_c__hy”. This string can be found in an invalid

Version resource of the original binary. Looking at the code, it can be seen that the data in this resource splits out to another encrypted binary and some configuration options.

The fields extracted from the resource this way are:

[0]: pre-useful stuff
[1]: encrypted binary
[2]: empty – used as key in Xorxorxor(key).PolyDeCrypt(binary)
[3]: False – does Piept() call Startup()?
[4]: True - does Piept() call Injection()?
[5]: False – does Piept() call AppLaunch()?
[6]: True - does Piept() call Vbc()?
[7]: False – does Piept() call FilePersistece()?
[8]: empty

The resource start and first marker string, followed by the encrypted binary data:

```
0000c320  64 00 75 00 63 00 74 00  4e 00 61 00 6d 00 65 00  |d.u.c.t.N.a.m.e.|
0000c330  00 00 00 00 2e 00 4e 00  65 00 74 00 20 00 53 00  |......N.e.t. .S.|
0000c340  65 00 61 00 6c 00 00 00  34 00 10 00 01 00 50 00  |e.a.l...4.....P.|
0000c350  72 00 6f 00 64 00 75 00  63 00 74 00 56 00 65 00  |r.o.d.u.c.t.V.e.|
0000c360  72 00 73 00 69 00 6f 00  6e 00 00 00 31 00 2e 00  |r.s.i.o.n...1...|
0000c370  33 00 2e 00 31 00 2e 00  30 00 00 00 38 00 10 00  |3...1...0...8...|
0000c380  01 00 41 00 73 00 73 00  65 00 6d 00 62 00 6c 00  |..A.s.s.e.m.b.l.|
0000c390  79 00 20 00 56 00 65 00  72 00 73 00 69 00 6f 00  |y. .V.e.r.s.i.o.|
0000c3a0  6e 00 00 00 31 00 2e 00  33 00 2e 00 31 00 2e 00  |n...1...3...1...|
0000c3b0  30 00 00 00 44 00 00 00  00 00 56 00 61 00 72 00  |0...D.....V.a.r.|
0000c3c0  46 00 69 00 6c 00 65 00  49 00 6e 00 66 00 6f 00  |F.i.l.e.I.n.f.o.|
0000c3d0  00 00 00 00 24 00 04 00  00 00 54 00 72 00 61 00  |....$.....T.r.a.|
0000c3e0  6e 00 73 00 6c 00 61 00  74 00 69 00 6f 00 6e 00  |n.s.l.a.t.i.o.n.|
0000c3f0  00 00 00 00 00 00 b0 04  70 38 44 2d 54 5f 34 4d  |........p8D-T_4M|
0000c400  30 74 5f 63 5f 5f 68 79  2d 7a d4 64 64 67 67 67  |0t_c__hy-z.ddggg|
0000c410  67 6b 6b 6b 6b 6a 69 69  69 21 21 21 21 21 21 21  |gkkkkjiii!!!!!!!|
0000c420  21 61 61 61 61 61 61 61  61 61 61 61 61 61 61 61  |!aaaaaaaaaaaaaaa|
0000c430  61 61 61 61 61 61 61 61  61 61 61 61 61 61 61 61  |aaaaaaaaaaaaaaaa|
0000c440  61 61 61 61 61 19 19 19  19 27 46 00 0e 0e c2 cb  |aaaaa....'F.....|
0000c450  98 b9 71 72 be 8b ac 00  68 d1 44 64 d4 46 b5 1c  |..qr....h.Dd.F..|
0000c460  8e ef 5c 7c df 40 ae 1c  8b ff 1f 81 e6 06 78 ed  |..\|.@........x.|
0000c470  5b 7b e4 52 72 b6 05 58  78 e5 54 b8 1d 4b 58 65  |[{.Rr..Xx.T..KXe|
```

The configuration options (note the True and False strings):

```
000603c0  fd 55 a5 e6 2a 6e b7 05  4c 9c dd 21 65 ae fc 43  |.U..*n..L..!e..C|
000603d0  9b f3 43 84 c8 0c 55 a3  ea 3a 7b bf 03 4c 9a e1  |..C...U..:{..L..|
000603e0  39 91 e1 22 66 aa f3 41  88 d8 19 5d a1 ea 38 7f  |9.."f..A...]..8.|
000603f0  d7 2f 7f c0 04 48 91 df  26 76 b7 fb 3f 88 d6 1d  |./...H..&v..?...|
00060400  75 cd 1d 5e a2 e6 2f 7d  c4 70 38 44 2d 54 5f 34  |u..^../}.p8D-T_4|
00060410  4d 30 74 5f 63 5f 5f 68  79 70 38 44 2d 54 5f 34  |M0t_c__hyp8D-T_4|
00060420  4d 30 74 5f 63 5f 5f 68  79 46 61 6c 73 65 70 38  |M0t_c__hyFalsep8|
00060430  44 2d 54 5f 34 4d 30 74  5f 63 5f 5f 68 79 54 72  |D-T_4M0t_c__hyTr|
00060440  75 65 70 38 44 2d 54 5f  34 4d 30 74 5f 63 5f 5f  |uep8D-T_4M0t_c__|
00060450  68 79 46 61 6c 73 65 70  38 44 2d 54 5f 34 4d 30  |hyFalsep8D-T_4M0|
00060460  74 5f 63 5f 5f 68 79 54  72 75 65 70 38 44 2d 54  |t_c__hyTruep8D-T|
00060470  5f 34 4d 30 74 5f 63 5f  5f 68 79 46 61 6c 73 65  |_4M0t_c__hyFalse|
00060480  70 38 44 2d 54 5f 34 4d  30 74 5f 63 5f 5f 68 79  |p8D-T_4M0t_c__hy|
00060490  70 38 44 2d 54 5f 34 4d  30 74 5f 63 5f 5f 68 79  |p8D-T_4M0t_c__hy|
000604a0  ef bb bf 3c 3f 78 6d 6c  20 76 65 72 73 69 6f 6e  |...<?xml version|
```

No matter what the configuration options, Piept() will decrypt the binary, possibly using a one-byte XOR key first. In this case, the field was empty (see above at 0×60419), so the following decryption was used only:

```
my $i = 0;
for($i = scalar(@x)-1; $i >= 1; $i--) {
  $result[$i-1] = chr( (ord($x[$i]) - ord($x[$i-1])) % 256 );
}
```

The binary decrypts to a Visual Basic executable, which can be identified as BlackShades RAT:

```
.text:00403670 dwVersion_PI    dd 1F4h                 ; DATA XREF: .text:lpProjectData↑o
.text:00403670                                         ; 5.00 in Hex (0x1F4). Version.
.text:00403674 lpObjectTable_PI dd offset lpHeapLink_OT ; Pointer to the Object Table
.text:00403678 dwNull_PI       dd 0                    ; Unused value after compilation.
.text:0040367C lpCodeStart_PI  dd offset dword_452040  ; Points to start of code. Unused.
.text:00403680 lpCodeEnd_PI    dd offset dword_452050  ; Points to end of code. Unused.
.text:00403684 dwDataSize_PI   dd 59B0h                ; Size of VB Object Structures. Unused.
.text:00403688 lpThreadSpace_PI dd offset unk_453000   ; Pointer to Pointer to Thread Object.
.text:0040368C lpVbaSeh_PI     dd offset loc_401320    ; Pointer to VBA Exception Handler
.text:00403690 lpNativeCode_PI dd 0                    ; Pointer to .DATA section.
.text:00403694 szPathInformation_PI:                   ; Contains Path and ID string. < SP6
.text:00403694                 unicode 0, <*\AC:\Users\Admin\Desktop\Blackshades project\Blackshades>
.text:00403694                 unicode 0, < NET\server\server.vbp>,0
.text:00403734                 db    0
.text:00403735                 db    0
```

This makes an outbound network connection on 4444/TCP to alosh66.myftp.org. On June 11th 2012, this had the address 31.9.170.140[3].

IP Address: 31.9.170.140
ISP: Syrian Telecommunications Establishment
Organization: Tarassul Internet Service Provider

This command and control domain is similar in naming convention to that used by the malware reported in the Guardian on March 20, 2012. In that case, a site which appeared to be a YouTube channel hosting pro-revolution videos installed malware disguised as an Adobe Flash Player update.

e58a1795277edc08d35c6898f9befc1c setup.exe

Instead of Adobe Flash, the malware installed Dark Comet RAT and contacted alosh66.no-ip.info.
On consecutive days in March 2012, the command and control domains for both pieces of malware pointed to the same IP address in Syrian (STE) address space:

2012-03-16 alosh66.no-ip.info. A 31.9.48.15
2012-03-17 alosh66.myftp.org. A 31.9.48.15

This evidence, combined with the similar naming convention, suggest that these attacks have been performed by the same actor.
This malware package (new_new .pif) is not well detected at this time, but it is detected by some anti-virus vendors. This version of the Black Shades RAT implant (VSCover.exe), is at the time of analysis (Jun 16th 2012) undetected.

We recommend following the recommendations laid out the in the EFF blog post for detection of this malicious software and, furthermore, exercising caution when receiving executable files via Skype or other internet chat media even if they supposedly come from trusted acquaintances.

ABOUT MORGAN MARQUIS-BOIRE

Morgan Marquis-Boire is a Technical Advisor at the Citizen Lab, Munk School of Global Affairs, University of Toronto. He works as a Security Engineer at Google specializing in Incident Response, Forensics and Malware Analysis.

ABOUT SETH HARDY

Seth Hardy is a Senior Security Analyst at the Citizen Lab, Munk School of Global Affairs, University of Toronto. He specializes in malware and network vulnerability analysis.

FOOTNOTES:

[1] piept is Romanian for “chest.”
[2] Romanian for “nipples”.
[3] The DNS registrar has since been contacted and the domain has been shut down.