The PlugX malware revisited: introducing “Smoaler”

By **Gabor Szappanos**, Principal Researcher, SophosLabs

July 2013

In a recent SophosLabs article about the PlugX malware family [A], we concluded with these words:

> *There is no doubt that PlugX development will go on, and new features and tricks will be introduced.*
>
> *We'll keep an eye on them, and if any interesting or important new features appear, we'll be sure to let you know.*

Fast forward just under two months, and we're ready to tell you the next stage in this ongoing saga.

The malware family we'll be looking at in this report is known as Smoaler, and it shares many features with PlugX, notably that:

- Smoaler relies on the same **vulnerability**, CVE-2012-0158. [B]
- Smoaler uses the same **exploit shellcode**.
- Smoaler uses similar visual distractions, or **decoys**, with a Tibetan theme.
- Smoaler uses the same **initial malware modules** to initiate infection.

Thereafter, the new malware follows a different path to the PlugX samples we looked at last time.

We shall analyse the “what happens next” component of Smoaler later on.

To clarify the terminology we have used above, remember that:

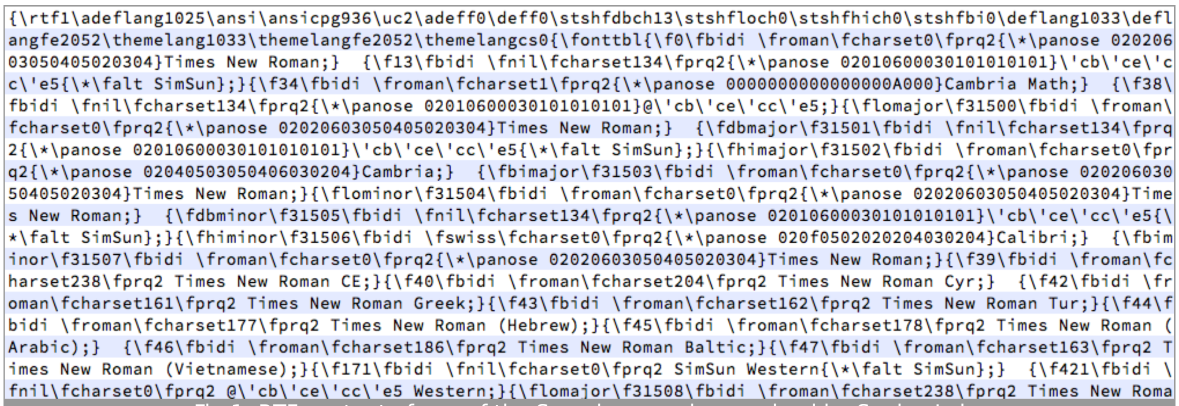
- A **vulnerability** is a software bug that could potentially be abused to make your computer behave insecurely.
- An **exploit** is a real-world trick by which a vulnerability can be activated to bypass security.

- **Shellcode** is runnable program code delivered inside a file that is supposed to be plain data, and therefore implicitly safe, but that can be executed without the user's knowledge or consent by exploiting a vulnerability.
- Initial malware modules, also called **droppers**, are malware components, often delivered or activated by shellcode, used to deliver the full malware, or payload, that the attacker wants to install.

How Smoaler arrives

The Smoaler samples seen by SophosLabs were all packaged into files with the extensions `.DOC` or `.DOCX`. These extensions usually denote files specific to Microsoft Word that were created by Microsoft Word.

Despite the extensions, however, all the files were actually in Rich Text Format (RTF), a text-based file format for representing documents.



```
{\rtf1\adeflang1025\ansi\ansicpg936\uc2\adeff0\deff0\stshfdbch13\stshfloch0\stshfhich0\stshfbi0\deflang1033\defl
angfe2052\themelang1033\themelangfe2052\themelangcs0{\fonttbl{\f0\fbidi \froman\fcharset0\fprq2{\*\panose 020206
03050405020304}Times New Roman;}  {\f13\fbidi \fnil\fcharset134\fprq2{\*\panose 02010600030101010101}\'cb\'ce\'c
c\'e5{\*\falt SimSun};}{\f34\fbidi \froman\fcharset1\fprq2{\*\panose 00000000000000000A000}Cambria Math;}  {\f38\
fbidi \fnil\fcharset134\fprq2{\*\panose 02010600030101010101}@\'cb\'ce\'cc\'e5;}{\flomajor\f31500\fbidi \froman\
fcharset0\fprq2{\*\panose 02020603050405020304}Times New Roman;}  {\fdbmajor\f31501\fbidi \fnil\fcharset134\fprq
2{\*\panose 02010600030101010101}\'cb\'ce\'cc\'e5{\*\falt SimSun};}{\fhimajor\f31502\fbidi \froman\fcharset0\fpr
q2{\*\panose 02040503050406030204}Cambria;}  {\fbimajor\f31503\fbidi \froman\fcharset0\fprq2{\*\panose 020206030
50405020304}Times New Roman;}{\flominor\f31504\fbidi \froman\fcharset0\fprq2{\*\panose 02020603050405020304}Time
s New Roman;}  {\fdbminor\f31505\fbidi \fnil\fcharset134\fprq2{\*\panose 02010600030101010101}\'cb\'ce\'cc\'e5{\
*\falt SimSun};}{\fhiminor\f31506\fbidi \fswiss\fcharset0\fprq2{\*\panose 020f0502020204030204}Calibri;}  {\fbim
inor\f31507\fbidi \froman\fcharset0\fprq2{\*\panose 02020603050405020304}Times New Roman;}{\f39\fbidi \froman\fc
harset238\fprq2 Times New Roman CE;}{\f40\fbidi \froman\fcharset204\fprq2 Times New Roman Cyr;}  {\f42\fbidi \fr
oman\fcharset161\fprq2 Times New Roman Greek;}{\f43\fbidi \froman\fcharset162\fprq2 Times New Roman Tur;}{\f44\f
bidi \froman\fcharset177\fprq2 Times New Roman (Hebrew);}{\f45\fbidi \froman\fcharset178\fprq2 Times New Roman (
Arabic);}  {\f46\fbidi \froman\fcharset186\fprq2 Times New Roman Baltic;}{\f47\fbidi \froman\fcharset163\fprq2 T
imes New Roman (Vietnamese);}{\f171\fbidi \fnil\fcharset0\fprq2 SimSun Western{\*\falt SimSun};}  {\f421\fbidi \
fnil\fcharset0\fprq2 @\'cb\'ce\'cc\'e5 Western;}{\flomajor\f31508\fbidi \froman\fcharset238\fprq2 Times New Roma
```

Fig 1: RTF content of one of the Smoaler samples received by SophosLabs

We assume that the attackers chose RTF because its text structure makes it easier to manipulate to create a working exploit for the CVE-2012-0158 vulnerability.

As with PlugX, the Smoaler files arrived with Tibetan-themed subject lines:

Filenames	`Selected quotes from the book.docx`
	`Press Release - Heart is Noble book launch.doc`
	`TIBATIAN PROBLEM.doc` [sic]
	`Backgrounder on His Holiness the Karmapa.docx`
	`Why Tibet Matters Now Oct 1 final version.doc`

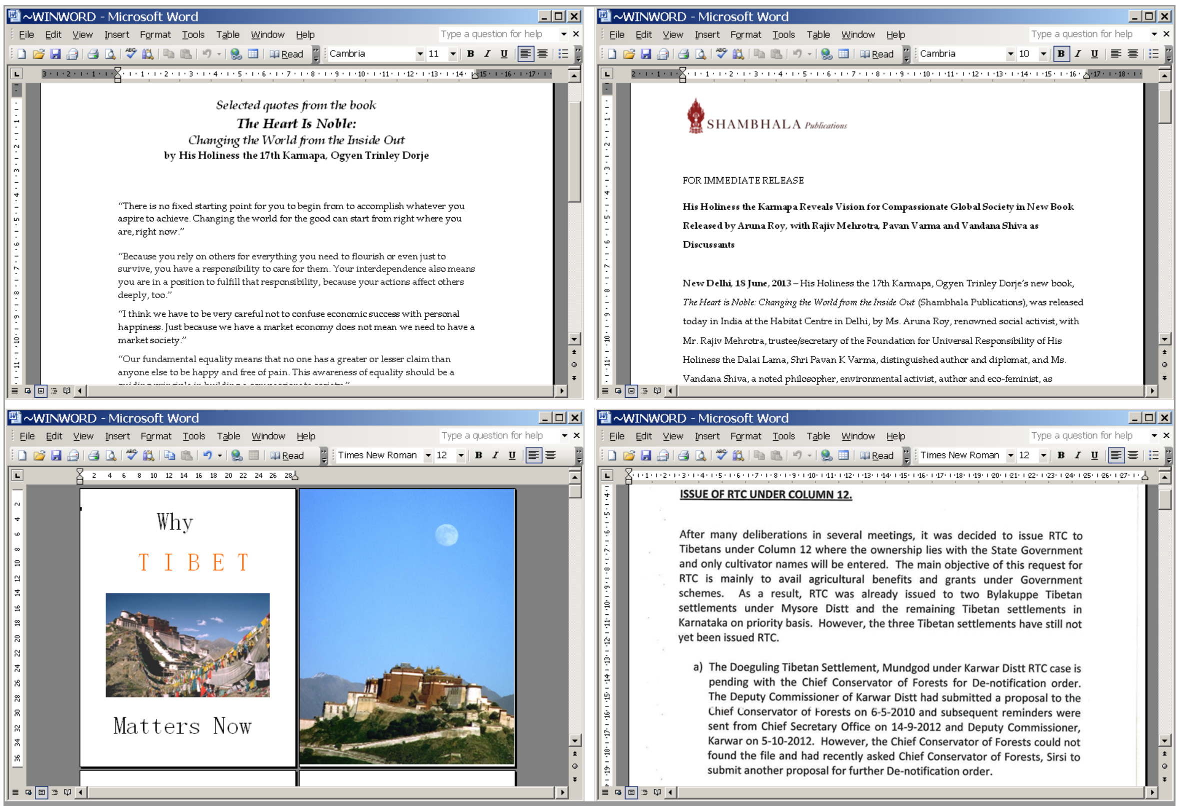
File sizes	404,220 bytes 294,096 bytes 1,086,312 bytes 397,564 bytes 7,858,097 bytes

SHA1 checksums	56e8c76cd88996da9b88901520f72ebb743e55ff a99b73b56fe94375ec46e51903f815d86afbd78d b2f854e9987bce5d110349a354588568ab49726b c093d4cd2390617da58bd412c9219e013de503a3 b84a133cf02eaa7b8a8096e997bda28fc482cf78
Sophos detection	Exp/20120158-A

If you open an infected document on an unpatched, unprotected computer, the shellcode in the document will activate and run.

After infecting your computer, the shellcode loads a new copy of Word to display a **decoy document** that is hidden inside the malicious file.

You will notice only notice a brief flicker when the old instance of Word closes and the new one with the decoy opens.



Selected quotes from the book
The Heart Is Noble:
Changing the World from the Inside Out
by His Holiness the 17th Karmapa, Ogyen Trinley Dorje

"There is no fixed starting point for you to begin from to accomplish whatever you aspire to achieve. Changing the world for the good can start from right where you are, right now."

"Because you rely on others for everything you need to flourish or even just to survive, you have a responsibility to care for them. Your interdependence also means you are in a position to fulfill that responsibility, because your actions affect others deeply, too."

"I think we have to be very careful not to confuse economic success with personal happiness. Just because we have a market economy does not mean we need to have a market society."

"Our fundamental equality means that no one has a greater or lesser claim than anyone else to be happy and free of pain. This awareness of equality should be a

FOR IMMEDIATE RELEASE

His Holiness the Karmapa Reveals Vision for Compassionate Global Society in New Book Released by Aruna Roy, with Rajiv Mehrotra, Pavan Varma and Vandana Shiva as Discussants

New Delhi, 18 June, 2013 – His Holiness the 17th Karmapa, Ogyen Trinley Dorje's new book, *The Heart is Noble: Changing the World from the Inside Out* (Shambhala Publications), was released today in India at the Habitat Centre in Delhi, by Ms. Aruna Roy, renowned social activist, with Mr. Rajiv Mehrotra, trustee/secretary of the Foundation for Universal Responsibility of His Holiness the Dalai Lama, Shri Pavan K Varma, distinguished author and diplomat, and Ms. Vandana Shiva, a noted philosopher, environmental activist, author and eco-feminist, as

ISSUE OF RTC UNDER COLUMN 12.

After many deliberations in several meetings, it was decided to issue RTC to Tibetans under Column 12 where the ownership lies with the State Government and only cultivator names will be entered. The main objective of this request for RTC is mainly to avail agricultural benefits and grants under Government schemes. As a result, RTC was already issued to two Bylakuppe Tibetan settlements under Mysore Distt and the remaining Tibetan settlements in Karnataka on priority basis. However, the three Tibetan settlements have still not yet been issued RTC.

a) The Doeguling Tibetan Settlement, Mundgod under Karwar Distt RTC case is pending with the Chief Conservator of Forests for De-notification order. The Deputy Commissioner of Karwar Distt had submitted a proposal to the Chief Conservator of Forests on 6-5-2010 and subsequent reminders were sent from Chief Secretary Office on 14-9-2012 and Deputy Commissioner, Karwar on 5-10-2012. However, the Chief Conservator of Forests could not found the file and had recently asked Chief Conservator of Forests, Sirsi to submit another proposal for further De-notification order.

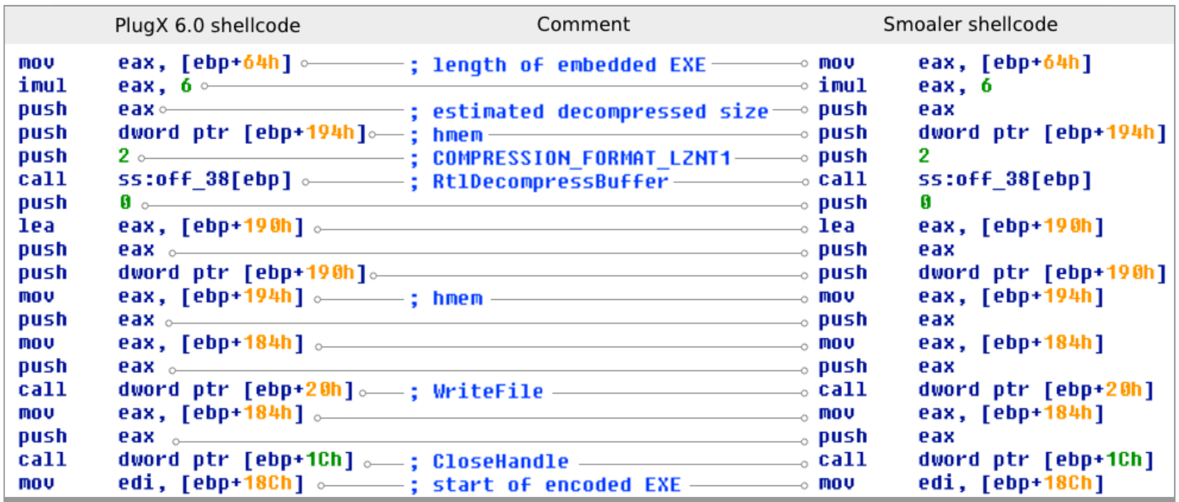
Fig 2: Sample Smoaler decoy documents

The decoy documents are all unexceptional, uninfected documents with content that matches the filenames given to the malicious RTFs. (See Fig 2.)

In preparing the decoy for display, the malware overwrites the infected RTF file with the contents of the decoy (which is in DOC format), thus removing one useful piece of evidence that might otherwise help pinpoint the source of infection.

The shellcode

The executable code that kicks off infection is the same as was used in PlugX, being byte-for byte identical (see Fig 3) to the shellcode from [PlugX 6.0](#) [A].



```
        PlugX 6.0 shellcode                  Comment                              Smoaler shellcode
mov     eax, [ebp+64h]          ; length of embedded EXE             mov     eax, [ebp+64h]
imul    eax, 6                                                       imul    eax, 6
push    eax                     ; estimated decompressed size        push    eax
push    dword ptr [ebp+194h]    ; hmem                               push    dword ptr [ebp+194h]
push    2                       ; COMPRESSION_FORMAT_LZNT1           push    2
call    ss:off_38[ebp]          ; RtlDecompressBuffer                call    ss:off_38[ebp]
push    0                                                            push    0
lea     eax, [ebp+190h]                                              lea     eax, [ebp+190h]
push    eax                                                          push    eax
push    dword ptr [ebp+190h]                                         push    dword ptr [ebp+190h]
mov     eax, [ebp+194h]         ; hmem                               mov     eax, [ebp+194h]
push    eax                                                          push    eax
mov     eax, [ebp+184h]                                              mov     eax, [ebp+184h]
push    eax                                                          push    eax
call    dword ptr [ebp+20h]     ; WriteFile                          call    dword ptr [ebp+20h]
mov     eax, [ebp+184h]                                              mov     eax, [ebp+184h]
push    eax                                                          push    eax
call    dword ptr [ebp+1Ch]     ; CloseHandle                        call    dword ptr [ebp+1Ch]
mov     edi, [ebp+18Ch]         ; start of encoded EXE               mov     edi, [ebp+18Ch]
```

Fig 3: The PlugX 6.0 and Smoaler shellcodes are identical

This shellcode is unusual for its use of LZNT1 compression for the embedded executable payload. This technique has not been observed in any other APT (Advanced Persistent Threat) attacks.

The shellcode decompresses an embedded PE file, writes it into the `%TEMP%` folder with the name `DW20.DLL` (this mimics the filename used by the Dr. Watson utility on Windows that pops up when a program crashes), and runs it.

First-stage dropper

Filename	`DW20.DLL`
Sophos detections	Troj/Plugx-I Troj/Plugx-K

The first-stage payload created by the shellcode contains another program file embedded as data.

It locates this in its own executable file, drops it to disk (which is where the name **dropper** comes from for this sort of malware component), and runs it.

There are two main sorts of dropper, detected by Sophos as Troj/Plugx-I and Troj/Plugx-K. (See Fig 4.) The names reflect the similarity, thus far, to earlier PlugX malware.

Troj/Plugx-I type Smoaler dropper PSEUDOCODE Troj/Plugx-K type Smoaler dropper

Troj/Plugx-I type Smoaler dropper

```
tempName = GetTempFileName();
hDropper = CreateFile(dw20Name,READ);
hNewfile = CreateFile(tempName,WRITE);
// Embedded EXE is at offset 0xA00
SetFilePointer(hDropper,0xA00);
repeat {
   buffer = ReadFile(hDropper,1024);
   outLen = WriteFile(hNewfile,buffer,1024);
} until outLen < 1024;
CloseHandle(hDropper);
CloseHandle(hNewfile);
WinExec(tempName);
```

Troj/Plugx-K type Smoaler dropper

```
tempName = GetTempFileName();
hDropper = CreateFile(dw20Name,READ);
totSize  = GetFileSize(hDropper);
// Embedded EXE is at offset 0xC00
newSize  = totSize - 0xC00;
hnewfile = CreateFile(tempName,WRITE);
SetFilePointer(hDropper,0xC00);
buffer = VirtualAlloc(newSize);
buffer = ReadFile(hDropper,newSize);
WriteFile(hNewfile,buffer,newSize);
CloseHandle(hdropper);
CloseHandle(hnewfile);
WinExec(tempName);
```

Fig 4: Side-by-side pseudocode for Smoaler first-stage dropper variants

But what happens next is quite different from a PlugX attack, which is why this malware has been given the general name Smoaler, rather than PlugX.

The intermediate infector

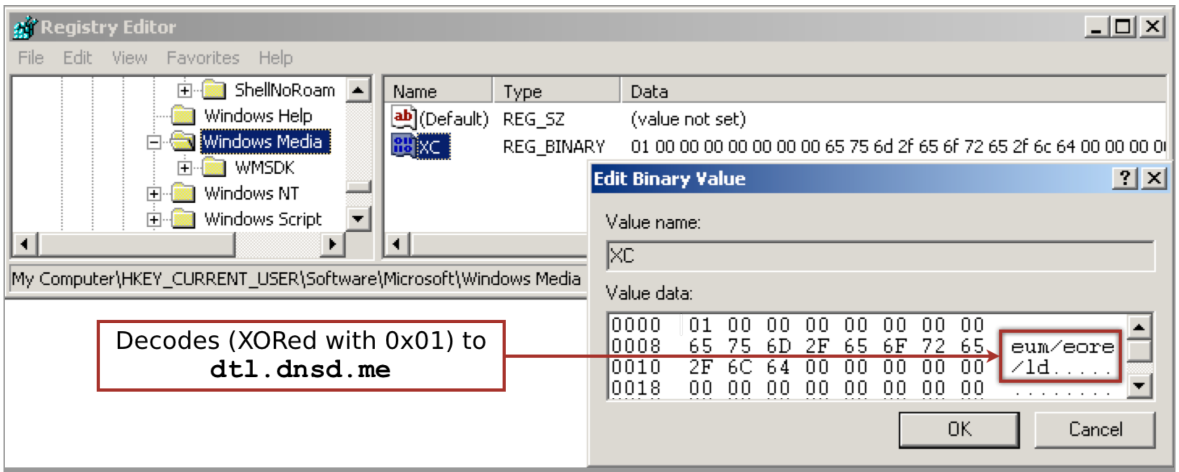
Filename	`Random temporary name (deleted after use)`
Sophos detection	Mal/Smoaler-A

The temporary file dropped by `DW20.DLL` contains a compressed DLL (dynamic link library: a special sort of program file) stored in its resources. This compressed program is unpacked using code built on the stack.

This component decodes the Command-and-Control (C&C) server names that the final malware will use, and saves them to the registry entry `HKCU\Software\Microsoft\Windows Media\XC`. (See Fig 5.)

C&C servers are the computers to which infected computers, often called *bots* or *zombies*, connect in order to fetch instructions on what to do next. Putting the C&C server names in the registry, rather than in the zombie executable itself, means that if the user becomes suspicious and submits the malware file to a security vendor, the locations of the C&C servers will not be revealed.

The data in the registry entry is obscured by flipping the least significant bit of the non-zero bytes (i.e. XORing them with 0x01). This provides a light disguise against an inquisitive user.



My Computer\HKEY_CURRENT_USER\Software\Microsoft\Windows Media

Name	Type	Data
(Default)	REG_SZ	(value not set)
XC	REG_BINARY	01 00 00 00 00 00 00 00 65 75 6d 2f 65 6f 72 65 2f 6c 64 00 00 00 0

Value name: XC

Value data:

```
0000  01 00 00 00 00 00 00 00
0008  65 75 6D 2F 65 6F 72 65  eum/eore
0010  2F 6C 64 00 00 00 00 00  /ld.....
0018  00 00 00 00 00 00 00 00  ........
```

Decodes (XORed with 0x01) to **`dtl.dnsd.me`**

Fig 5: Smoaler registry entry used to store C&C server names

After storing the C&C names in the registry, the intermediate infector loads the dropped DLL using the `LoadLibrary()` Windows API call.

The intermediate infector is deleted from the system after execution.

Main malware component

Filename	`Randomly-generated name`
Sophos detection	Troj/Smoaler-A

The DLL dropped by the intermediate infector is the principal component of Smoaler.

The DLLs created during infection are variable in size, and huge, ranging from 20 MBytes to 50 MBytes.

This is not because they are complex and packed with functionality: the useful content of Smoaler is less than 40 KBytes. The bulk of each DLL consists of a number (two to six) binary resources of 5 MBytes to 10 MBytes each. These resources are filled with zero bytes, entirely for the purpose of bulking up the file.

We assume that the purpose of deliberately bloating the main DLL is to disguise its original source, since even a suspicious user might not connect a multi-megabyte DLL with the original infectious RTF of a few hundred kilobytes. (The original RTF is also replaced with the decoy DOC file after infection, further diverting attention from the origin of the malware.)

When the DLL first runs, it installs itself permanently into two places on the victim's computer:

- `C:\Documents and Settings\All Users\Application Data\Microsoft\Windows\Burn\%COMPUTERNAME%.dll`

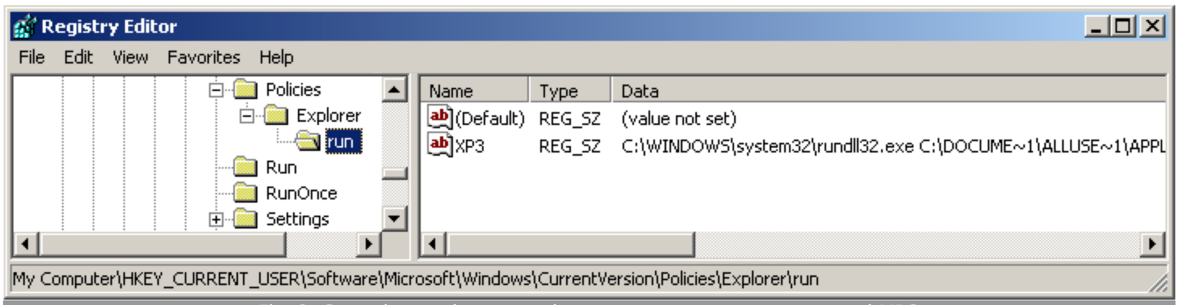
(`%COMPUTERNAME%` is generated by querying the name of the computer.)

- `C:\Documents and Settings\All Users\Application Data\Microsoft\Windows\LiveUpdata_Mem\%RANDOM%.dll`

(The `%RANDOM%` part of the name is variable, e.g. `B6go3s_One.dll` or `7n5HjV.dll.`)

Two or three copies of the DLL are dropped. They typically differ in size, because the zero-byte resource padding may vary in each file.

The DLL is added as a registry value called `%COMPUTERNAME%` in the registry key `HKCU\Software\Microsoft\Windows\CurrentVersion\Policies\Explorer\run`. This provides what is known as **persistence**, meaning that the malware is automatically re-loaded every time the victim reboots or logs on. (See Fig 6.)



Name	Type	Data
(Default)	REG_SZ	(value not set)
XP3	REG_SZ	C:\WINDOWS\system32\rundll32.exe C:\DOCUME~1\ALLUSE~1\APPL

My Computer\HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Policies\Explorer\run

Fig 6: Smoaler registry persistence on a computer named XP3

Note that Smoaler is launched using the Windows utility `rundll32.exe`, because DLLs cannot execute on their own. They require a host program inside which to run.

When connecting to its C&C servers, Smoaler injects itself into the process `IEXPLORE.EXE`. This is a common trick used by malware to make its traffic appear to originate from a browser, thus arousing much less suspicion.

While running, however, Smoaler keeps a lookout for processes called `vsserv.exe`, `fsdfwd.exe`, `AvastSvc.exe`, `uiWatchDog.exe`, and `avp.exe`. It avoids injecting itself into Internet Explorer if any of them are active.

These processes belong to various security products, so this is a malware precaution often jocularly called **anti-anti-virus**, intended to avoid suspicious activity that might attract the attention of security software.

After installing itself, Smoaler attempts to connect to its C&C servers. The domain names it uses are already known from earlier Tibet-related malware attacks: `dtl.dnsd.me`, `dtl.eauto.com` and `dtl6.moo.com`.

What happens next?

Unfortunately, we can't say precisely how an infected computer will behave if Smoaler gets this far.

That is because the content that the primary Smoaler DLL fetches from its C&C servers is whatever malware the attackers choose to serve up next.

Worse still, the programs downloaded by Smoaler are not in regular EXE or DLL format.

Firstly, the downloads are encrypted; secondly, their headers are stripped off so that, even decrypted, they are not immediately obvious as programs; thirdly, they are loaded directly into, and run directly from, memory, rather then being written to disk and launched via the Windows API.

That means that if you kill off the Smoaler process and delete its primary executable, as you will almost certainly want to do if you find out that you are infected, you may no longer have a complete copy of the malware to submit for analysis.

In this way, the Smoaler authors avoid showing all the cards in their hand at once, in the hope of staying one step ahead.

However, this multi-stage approach also has a significant disadvantage for cybercriminals: there are now multiple points in the attack at which you can spot an anomaly, intervene, and win.

Prevention is better than cure

Malware like Smoaler is a strong reminder of why proactivity and prevention are better than cure: once you are infected, it can be hard to go back and unravel what happened, because of the tricks that the malware uses along the way.

References

[A] http://nakedsecurity.sophos.com/inside-the-plugx-malware

[B] http://www.cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2012-0158

For further security information

http://www.sophos.com/why-sophos/our-people/technical-papers.aspx

http://nakedsecurity.sophos.com/

http://podcasts.sophos.com/