Analysis Report (TLP:WHITE) Analysis of a PlugX variant (PlugX version 7.0)

Conducted by CIRCL - Computer Incident Response Center Luxembourg Team CIRCL

March 29, 2013

Document version: 1.0



Contents

1	Sco	pe of work	}
2	Ana	dyzed samples 3	3
	2.1	Limitations	-
	2.2	Sharing	;
3	Exe	cutive summary 6	3
4	Ana	dysis	ì
	4.1	Techniques used	
	4.2	Execution process	7
		4.2.1 Diagram	7
		4.2.2 Explanation	7
	4.3	Implemented commands)
	4.4	Command details	2
		4.4.1 Option	2
		4.4.2 Disk	2
		4.4.3 Screen	2
		4.4.4 Process)
		4.4.5 Service	2
		4.4.6 Shell	2
		4.4.7 Telnet	3
		4.4.8 RegEdit	3
		4.4.9 Nethood	3
		4.4.10 Portmap	3
		4.4.11 SQL	3
		4.4.12 Netstat	3
		4.4.13 Keylogger	3
	4.5	Other notable commands and functions	3
		4.5.1 log	3
	4.6	Persistency	ó
	4.7	Control	ó
	4.8	Network and domain information	;
		4.8.1 Network	;
		4.8.2 Domain	7
	4.9	Current version and history of PlugX	3
A	A	andir	0
A		Dendix 18 Indicators of Compromise (IOC) 18	
	A.1	- ' '	
		A.1.1 Pipes	
		A.1.3 Registry	
	1.2	,	
	A.2		
	A.3	VirusTotal results	J

1 Scope of work

This report is the analysis of a Remote Access Tool (RAT) which we call a variant of Plugx¹. Plugx is an interesting piece of malware for several reasons:

- \bullet It demonstrates the attack principle of the fastest/cheapest path of attack² by abusing perfectly valid signed binaries to perform the attack
- It features ways to defeat other protection mechanisms like UAC³
- In contrast to many other pieces of malware, the author⁴ shows the ability to write good code, especially doing logging the right way to improve the piece of software
- It appears to be modularized and easily extensible

2 Analyzed samples

- Sample A Stage 1 of Malware
 - Description
 - * Hash found in a malware database
 - Original filename
 - * update.exe
 - Hashes
 - * MD5: f1f48360f95e1b43e9fba0fec5a2afb8
 - * SHA1: 70ceb467db7b0161d22e4545479f747417b9705a
 - *~SHA-256:~2bc5ce39dd9afe2157448d3f6d8cb9c549ed39543d159616e38480b9e6c11c49
 - Filetype
 - * PE32 executable (GUI) Intel 80386, for MS Windows, RAR self-extracting archive
 - Filesize
 - * 370702 Bytes (326KB)
 - Compile time
 - * Sat Jun 9 15:19:49 2012
- Sample B Valid, signed McAfee binary
 - Description
 - * File dropped by Sample A
 - Original filename
 - * mcvsmap.exe
 - Hashes

¹Known variant names: Gulpix, Korplug

²http://satoss.uni.lu/seminars/srm/pdfs/2012-Alexandre-Dulaunoy.pdf

³http://msdn.microsoft.com/en-us/library/windows/desktop/bb756996.aspx

⁴For better readability we do not distinguish between a single author or a group of authors. Hence the expression is a synonym for "the authors"

- * MD5: 4e1e0b8b0673937415599bf2f24c44ad
- * SHA1: 9224de3af2a246011c6294f64f27206d165317ba
- * SHA-256: ae16e10e621d6610a3f7f2c7122f9d1263700ba02d1b90e42798decb2fe84096
- Filetype
 - * PE32 executable (GUI) Intel 80386, for MS Windows
- Filesize
 - * 262672 Bytes (257K)
- Compile time
 - * Fri May 8 17:59:52 2009
- Authenticode⁵ verification

```
Verified: Signed
     Signers:
 3
       McAfee, Inc.
       VeriSign Class 3 Code Signing 2004 CA
       Class 3 Public Primary Certification Authority
     Signing date: 5:24 PM 5/8/2009
     Publisher: McAfee, Inc.
     Description: McAfee VirusMap Reporting module
     Product: McAfee VirusScan API
Version: 13,11,0,0
10
     File version: 13,11,102,0
11
12
     Strong Name: Unsigned
     Original Name: McVsMap.EXE
Internal Name: McVsMap
13
14
     Copyright: Copyright 2008 McAfee, Inc.
15
16
     Comments: n/a
     MD5: 4e1e0b8b0673937415599bf2f24c44ad
17
     SHA1: 9224 de3af2a246011c6294f64f27206d165317ba
     SHA256\colon\ ae16e10e621d6610a3f7f2c7122f9d1263700ba02d1b90e42798decb2fe84096
```

- Sample C DLL to be loaded by Sample B, loads Sample D
 - Description
 - * File dropped by Sample A
 - Original filename
 - * McUtil.DLL
 - Hashes
 - * MD5: ad4a646b38a482cc07d5b09b4fffd3b3
 - * SHA1: ae0f9bf2740d00c5d485827eb32aca33feaa3a90
 - * SHA-256: 0a99238e1ebebc47d7a89b2ccddfae537479f7f77322b5d4941315d3f7e5ca48
 - Filetype
 - * PE32 executable (DLL) (GUI) Intel 80386, for MS Windows
 - Filesize
 - * 49152 Bytes (48K)

⁵http://msdn.microsoft.com/en-us/library/ms537359%28v=vs.85%29.aspx

- Compile time
 - * Wed Mar 13 02:52:28 2013
- Sample D Malicious payload to be loaded by Sample C
 - Description
 - * File dropped by Sample A
 - Original filename
 - * McUtil.DLL.PPT
 - Hashes
 - * MD5: 545bb4365a9b7cdb6d22844ebeedda93
 - * SHA1: a267f1183b4ff843d68a63264846abf78cc71d1f
 - * SHA-256: d4fe890a08d4dd44b58a3b85b2a7e89536338099c1c42a9b7e85f4007b0a37b7
 - Filetype
 - * pure code (IA32) without headers
 - Filesize
 - * 124820 Bytes (122K)
 - Compile time
 - * unknown (pure code)
- Sample E Stage 2 of Malware
 - Description
 - * Extracted malware from memory
 - Original filename
 - * dump00C60000.bin
 - Hashes
 - * MD5: 65 ceb 039 e 7 b 4731 a 165 cfee 081 e 220 af
 - * SHA1: b49766187971e3070644a9de2054bc93241b2263
 - * SHA-256: deeac56026f3804968348c8afa5b7aba10900aeabee05751c0fcac2b88cff71e
 - Filetype
 - * PE32 executable (DLL) (GUI) Intel 80386, for MS Windows
 - Filesize
 - * 176128 Bytes (172K)
 - Compile time
 - * Mon Nov 26 04:46:01 2012
- Sample F UAC circumvention
 - Description
 - * File temporarily created on filesystem
 - Original filename

- * UAC.TMP
- Hashes
 - * MD5: 52df5c2c07433e2a8f054c2347acb3b4
 - * SHA1: 8051474c1fc0d8f404a42ea32eca1699e54f02e1
 - * SHA-256: dc09091e5d0ce03c6144748f17bd636f2f0b2ca56f88b550c1d48860596dbdb1
- Filetype
 - * PE32 executable (DLL) (GUI) Intel 80386, for MS Windows
- Filesize
 - * 2560 Bytes (2.5K)
- Compile time
 - * Thu Mar 29 08:03:43 2012

2.1 Limitations

This work has been done with utmost care, following best practices in software reversing, forensic investigations and/or information gathering. However, the work is only covering small aspects (based on the indicators given, lacking full context) and not an exhaustive analysis, and hence the report is as-is, not giving any guarantees of completeness or claiming absolute accuracy. This work is provided for information only.

2.2 Sharing

The document is classified as TLP:WHITE, CIRCL authorizes everyone to share this analysis report as-is without modification.

3 Executive summary

The analyzed malicious software is an exhaustive Remote Access Tool (RAT) that defeats several protection methods of modern Windows operating systems, including execution of signed code and defeating UAC in Windows 7. It comes with a multitude of functionalities that are well implemented.

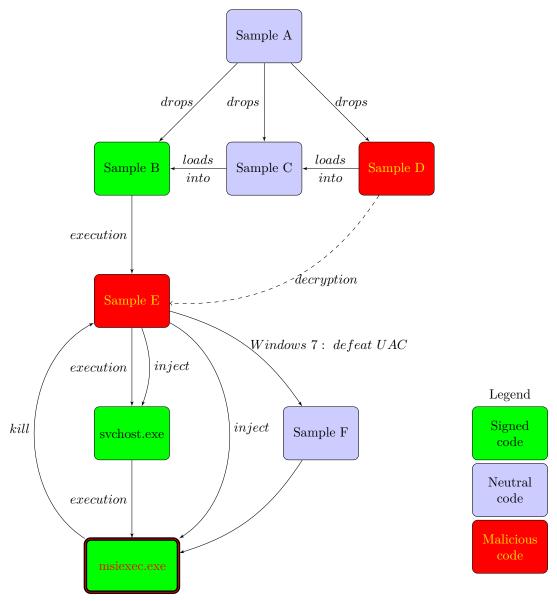
4 Analysis

4.1 Techniques used

The analysis has been done using a mixed-approach of dynamic analysis and static analysis in order to overcome some of the obfuscation and encryptions used by the malware. Some of the techniques might have also an impact on the interpretation of the malware. Unfortunately, when we started this investigation, the IP address is no longer accepting connections on the given ports when tested on 2013-03-26. An interaction following the protocol of this malware is therefore no longer possible.

4.2 Execution process

4.2.1 Diagram



4.2.2 Explanation

Sample A is a self-extracting archive which contains three files, Sample B, Sample C and Sample D. It is assumed that Sample A is a part of another attack vector, like PDF or Office document attacks where the user just opens a crafted document which exploits the document reader, drops and opens both a readable document and a malicious file like Sample A.

```
1 Type = Rar
2 Solid = -
```

```
Blocks = 3
   Multivolume = -
 5
   Volumes = 1
 6
                  Time
                                          Size
                                                   Compressed
      Date
                           Attr
                                                                Name
9
   2009{-}05{-}14\ 00{:}56{:}12
                                        262672
                                                       119784
                                                                mcvsmap.exe
                          . . . . A
  2013-03-13 09:52:28
                                                        20285
10
                          . . . . A
                                         49152
                                                                McUtil.DLL
   2013-03-13 14:56:12
                                        124820
                                                       124820
                                                                McUtil.DLL.PPT
12
                                        436644
                                                       264889
                                                                3 files, 0 folders
13
```

Executing the self-extracting archive extracts the files and runs mcvsmap.exe (Sample B). Sample B is a valid signed file that the author of the malware took from a software bundle from McAfee. Sample B, when executed, attempts to load a file McUtil.DLL from the same directory, which usually is another component of McAfee. The malware author instead bundled the valid McAfee file Sample B with a custom DLL (Sample C). Since the file will be loaded without hesitation (there are no protection mechanisms implemented; neither does McAfee check if the imported file meets any conditions nor is any protection implemented for loading unsigned libraries in signed code), the signed Sample B jumps into the beginning of the code section of Sample C (via Push/Return):

At the target location the following code is executed:

```
read_execute_file()
1
2
3
     NumberOfBytesRead = GetModuleFileNameW(hModule, &filename, 0x2000u);
4
     lstrcatW(&filename , L".PPT");
5
     hFile_mcutil.dll.ppt = CreateFileW(&filename, GENERIC_READ, 1u, 0, OPEN_EXISTING, 0,
6
     if (hFile_mcutil.dll.ppt == -1)
7
     {
8
       result = GetLastError();
9
     }
10
     else
11
       buffer = VirtualAlloc(0, 0x100000u, MEM_COMMIT, PAGE_EXECUTE_READWRITE);
12
13
       if (buffer && ReadFile(hFile_mcutil.dll.ppt, buffer, 0x100000u, &NumberOfBytesRead
14
         CloseHandle(hFile_mcutil.dll.ppt);
15
16
         buffer();
         Sleep (0xFFFFFFF);
17
         Sleep (0xFFFFFFF);
18
19
         Sleep (0xFFFFFFF);
20
         result = 0:
21
22
       else
23
24
         result = GetLastError();
```

The code retrieves the filename of itself (line 3), which is McUtil.DLL, and appends .PPT (line 4). A handle to the filename McUtil.DLL.PPT is created in line 5. In line 12 an exectuable memory region is created, which is filled with the content of the file McUtil.DLL.PPT (line 13). After closing the handle to the file (line 15), the memory region is called (line 16). The next screenshot shows that the memory contains only pure code without any overhead like MZ/PE headers. The entropy of this file is 7.997904 bits per byte:

```
edx, 0CBE1F4B7h
00C40000 cmp
00C40006 test
                  edx, OFF2A2004h
00C4000C xon
                  esi,
                       0BBE5876Fh
00040012 dec
                  edx
00040013 xon
                  edx,
                       2EA8D3E3h
00040019 cmp
                  edi,
                       45B286E4h
00C4001F
         and
                  esi,
                       -0A05A96A4h
00040025 inc
                  edx
00040026
                       2BFCC370h
                  edi,
         mov
                  edi, 0A347050Bh
0004002B cmp
00040031
                       3646EDD4h
                  esi,
         CMD
00040037
                       9072B29Ah
                  edx.
         mov
00040030
         call
                  $+5
00040041
         test
                  esi,
                       760A483Dh
00C40047
         dec
                  esi
00040048
         and
                  edx,
                       -0BB03BC9h
00C4004E or
                  edi, 0BE8D653Ah
00040054 and
                  edi, 0CF84EF63h
                  esi, 44A352FBh
0004005A or
00040060 and
                       231692BEh
                  edx,
00040066
         and
                  edi,
                       92FA7705h
00C4006C
         and
                  edi,
                       -00537FF3Bh
00040072
         pop
                  eax
00040073
         add
                  esi,
                       5F409FE1h
00040079
         dec
                  edi.
00C4007A on
                  edx, 0F77945A9h
00040080 cmp
                  edx,
                       64CCDD6Dh
                       808BF208h
00040086 add
                  esi,
00C4008C
                  edx,
                       60D05232h
         cmp
00040092
         dec
00C40093 cmp
                  edx, 0F3F3B05Ah
00040099
                  edi,
                       49319D2Ah
         mov
                  edx, 0F80B7589h
0004009E_sub
                  esi, 0F5C87C7Ah
000400A4 cmp
                  edx, 0FC22F58Eh
00C400AA add
```

The code, when executed, reveals the first hint about what we found:

```
eax, [epp+arg_u]
UUU4U69E MOV
00C406A1 mov
                  [esi+10h], eax
000406A4 mov
                  eax, [ebp+arg_10]
000406A7 mov
                  [esi+14h], eax
00C406AA mov
                  eax, [ebp+arg_14]
00C406AD mov
                  [esi+18h], eax
                  dword ptr [esi],
       0 mov
                  eax, [edi+28h]
00C406B6 mov
00040689 add
                  eax. esi
                  [esi+1Ch], eax
00C406BB mov
```

It decompresses and decrypts itself, using the Microsoft API call RtlDecompressBuffer and the custom decryption routine:

```
int crypt(unsigned int a1, int a2, int a3, int a4)
 2
 3
     if (a4 > 0)
 4
5
       v10 = a3 - a2;
 6
       do
 7
 8
         a1 = a1 + (a1 >> 3) - 0x111111111;
         9
10
         a1 += 0x444444444 - (a1 << 9);
         a1 += 0x33333333 - (a1 << 7);

v7 = *(v10 + a2++) \hat{} (a1 + a1)
11
12
                               (a1 + a1 + a1 + a1);
13
         v8 = a4 - = 1;
14
          *(a2 - 1) = v7;
15
16
       while ( !v8 );
17
18
     return 0;
```

The decrypted and decompressed file is not written onto disk, it always remains in memory. Sample E is the extracted version of this memory segment. At this point it can be mentioned that neither the encrypted Sample D nor the decrypted memory segment Sample E are detected by Virus scanners.

After some initialisation work like adjusting tokens (SeDebugPrivilege, SeTcbPrivileg⁶, to act as part of the operating system), a new process is started, the original sychost.exe from Microsoft, and the code from Sample E is injected into the memory of that process. In a next step, sychost.exe is instructed to execute the original msiexec.exe from Microsoft, where also memory is injected like it has been done for sychost.exe. Special conditions apply when run under Window 7, which is protected by User Account Control (UAC). UAC is supposed to protect the user better from running malware by requesting the administator for approval before running a potentially dangerous application. In the environment of Windows 7, the malware drops temporarily file Sample F, which it uses to evade or defeat the UAC mechanism. After killing the parent processes, only two processes are left: sychost and msiexec. Both are verified binaries, none of the includes a malicious DLL.



Nevertheless, they both contain the malicious code. At this point in time the malware is already talking to the C&C, no user interaction was required, all standard security mechanisms were defeated.

4.3 Implemented commands

The analysis of Sample B revealed the commands as shown in the table below:

⁶http://technet.microsoft.com/en-us/library/bb457125.aspx

Table 1: Implemented commands

Table 1: Implemented commands					
Source file	Internal command	subcommand	Description		
		0x2000	lock workstation		
	Option	0x2001	shutdown workstation (forced)		
XPlugOption.cpp		0x2002	reboot workstation		
		0x2003	shutdown workstation (graceful)		
		0x2005	show messagebox		
	Disk	0x3000	enumerate drives		
		0x3001	find file		
		0x3002	find file recursively		
		0x300A	create directory		
XPlugDisk.cpp		0x3004	read file		
		0x3007	write file		
		0x300D	file copy/rename/delete/move		
		0x300C	create process on hidden desktop		
		0x300E	get expanded environment string		
	Screen	0x4000	Remote Desktop capabilities		
VDI G		0x4004	send mouse event		
XPlugScreen.cpp		0x4005	send keyboard event		
		0x4006	send CTRL-Alt-Delete		
		0x4100	take screenshot		
	Process	0x5000	create process		
XPlugProcess.cpp		0x5001	enumerate processes		
		0x5002	kill process		
	Service	0x6000	query service config		
		0x6001	change service config (forced)		
XPlugService.cpp		0x6002	start service		
		0x6003	control service		
		0x6004	delete service		
XPlugShell.cpp	Shell	0x7002	start a cmd shell		
XPlugTelnet.cpp	Telnet	0x7100	start telnet server		
0 11	RegEdit	0x9000	enumerate keys		
		0x9001	create key		
		0x9002	delete key		
TADI D		0x9003	copy key		
XPlugRegedit.cpp		0x9004	enumerate values		
		0x9005	set value		
		0x9006	delete value		
		0x9007	get value		
XPlugNethood.cpp	Nethood	0xA000	enumerate network resources		
XPlugPortMap.cpp	Portmap	0xB000	starts port mapping		
	SQL	0xC000	get data source information		
XPlugSQL.cpp		0xC001	get data source information get driver description		
1000 00 10 PP		0xC001	execute statement		
	Netstat	0xD000	get TCP table		
XPlugNetstat.cpp		$0 \times D000$	get UDP table		
111 1481 (Change Chh		$0 \times D001$ $0 \times D002$	set TCP entry		
XPlugKeyLogger.cpp	Keylogger	0xE000	starts key logger thread		
711 Tugite y Logger.cpp	110 y 10 gg C1	UALUUU	boar os ney 10gger tilleau		

4.4 Command details

4.4.1 **Option**

XPlugOption implements commands to lock the workstation, shut it down or reboot it. In addition, XPlugOption can create a thread that calls MessageBoxW() in order to present a message box to the user.

4.4.2 Disk

XPlugDisk is used to enumerate connected disk drives and can be used to find and manipulate files and directories. In addition, XPlugDisk can be used to create a process, optionally on a hidden Windows desktop with the name "HH", as the code below illustrates:

```
if (a1->hidden)
2 3
        \label{eq:hDesktop} \mbox{hDesktop} = \mbox{CreateDesktopW}(\mbox{L"HH"}\,,\ 0\,,\ 0\,,\ 0\,x10000000u\,,\ 0)\,;
 4
        if (!hDesktop)
5
          \log("XPlugDisk.cpp", 665, 0);
 6
 7
     hidden = a1->hidden;
     StartupInfo.lpDesktop = (hidden != 0 ? L"HH" : 0);
8
9
     StartupInfo.cb = 68;
10
     StartupInfo.dwFlags = 1;
     {\tt StartupInfo.wShowWindow = hidden == 0;}
11
12
     if ( CreateProcessW(0, &a1->commandline, 0, 0, 0, 0, 0, 0, &StartupInfo, &
           ProcessInformation) )
13
     {
14
15
     }
```

4.4.3 Screen

XPlugScreen is not only taking screenshots, it is also implementing remote desktop capabilities. It is able to capture the screen (internal command: ScreenT1) and can send mouse and keyboard events (internal command: ScreenT2).

4.4.4 Process

XPlugProcess implements three commands and is able to enumerate, create and kill processes.

4.4.5 Service

In the module XPlugService commands are available related to Windows services. Code is implemented to query service configurations, change service configuration, start, control and delete services.

4.4.6 Shell

A remote shell for the attacker is created in the module XPlugShell, by creating an asynchronous set of pipes (\pipe\a and \pipe\b) for cmd.exe and the console attached to it (AttachConsole()).

4.4.7 Telnet

cmd.exe $/\mathbf{Q}$ is executed in the module XPlugTelnet in order to start a telnet server on the attacked machine.

4.4.8 RegEdit

XPlugRegedit implements a set of commands to process the Windows registry. It is able to enumerate, create, delete and copy keys. It is also able to enumerate, set, delete and get values from the registry.

4.4.9 Nethood

XPlugNethood is the module to enumerate network resources like network shares.

4.4.10 Portmap

XPlugPortMap indicates that it performs some port mapping, however, the code is not understood, yet.

4.4.11 SQL

XPlugSQL implements three functions to query SQL servers: a function to get data source information, a function to get the driver description and a function to execute SQL statements.

4.4.12 Netstat

XPlugNetstat gets the TCP and UDP connection table and is able to set TCP table entries.

4.4.13 Keylogger

The keylogger implemented in XPlugKeyLogger catches Window titles, date, time and logs entered keys into the file

```
1 C:\Documents and Settings\All Users\VirusMap\NvSmart.hlp
```

It has the format following the example below:

```
2013-03-26 09:40:57 | C:\Program Files\Mozilla Firefox\firefox.exe - Mozilla Firefox
www.google.com

2013-03-26 09:47:49 | C:\WINDOWS\system32\notepad.exe | Untitled - Notepad
This is not a password

2013-03-26 09:48:06 | C:\WINDOWS\Explorer.EXE | C:\Documents and Settings\All Users\
VirusMap
```

4.5 Other notable commands and functions

$4.5.1 \log$

This function is called almost everywhere when the author expects that a functions returns an error, at 1036 places. This is obviously done to ensure code quality.

```
write log(LPCWSTR lpBuffer)
2
3
    ExpandEnvironmentStringsW(L"%ALLUSERSPROFILE%", &path_to_bug.log, 0x800u);
4
     // %ALLUSERSPROFILE%\SxS\bug.log
5
    lstrcatW(&path_to_bug.log, L"\\SxS");
    CreateDirectoryW(&path_to_bug.log, 0);
7
    SetFileAttributesW(\&path\_to\_bug.log\;,\;\;6u)\;;
8
    lstrcatW(&path_to_bug.log, L"\\bug.log");
    result = CreateFileW(&path_to_bug.log, 0x40000000u, 1u, 0, 4u, 2u, 0);
10
    if ( result !=-1 )
11
      if (SetFilePointer(result, 0, 0, 2u) != -1)
12
13
14
        GetLocalTime(&SystemTime);
        NumberOfBytesWritten = wsprintfW(
15
16
                                &Buffer,
17
                                L"%4.4d-%2.2d-%2.2d %2.2d:%2.2d:%2.2d: ",
18
                                SystemTime.wYear.
19
                                System Time. w Month \\
20
                                SystemTime.wDay,
21
                                SystemTime.wHour
22
                                SystemTime.wMinute,
23
        24
            0))
25
26
          len = lstrlenW(lpBuffer);
27
          WriteFile(result, lpBuffer, 2 * len, &len, 0);
28
29
30
      result = CloseHandle(result);
31
32
    return result;
33
```

Example log file entries from file

```
1 \ \ \% ALLUSERS PROFILE\% \backslash SxS \backslash bug. log
```

```
2013-03-25 11:43:28: file: XSetting.h, line: 57, error: [1300]Not all privileges
    referenced are assigned to the caller.
2013-03-25 11:51:12: file: XInstallUAC.cpp, line: 162, error: [5] Access is denied.
32013-03-25 13:59:45: file: XRTL.cpp, line: 186, error: [1300]Not all privileges
    referenced are assigned to the caller.
42013-03-25 14:07:12: file: XRTL.cpp, line: 186, error: [123]The filename, directory
    name, or volume label syntax is incorrect.
52013-03-25 14:07:12: file: XSetting.h, line: 58, error: [3]The system cannot find the
    path specified.
62013-03-25 14:21:12: file: dllmain.cpp, line: 47, error: [1300]Not all privileges
    referenced are assigned to the caller.
72013-03-25 17:31:58: file: XInstall.cpp, line: 451, error: [5] Access is denied.
82013-03-25 17:37:00: file: XSoTcpHttp.cpp, line: 646, error: [12029]*
```

In addition an exception filter is installed to fetch the circumstances of otherwise not caught errors:

```
TopLevelExceptionFilter(struct_a1_30 *a1)
{
...
if ( wsprintfA( &OutputString,
```

Page 14 of 21

```
"EName: %s, EAddr: 0x%p, ECode: 0x%p, EAX: %p, EBX: %p, ECX: %p, EDX: %p, ESI: %p, EDI: %p, EBP
                       :%p,ESP:%p,EIP:%p\r\n",
 7
                &String1
                 a1->ECode[3].
 9
                 *a1->ECode,
10
                v6 \rightarrow reg_eax,
11
                 v6 \rightarrow reg_ebx,
12
                v6->reg_ecx,
13
                 v6 \rightarrow reg_edx,
14
                 v6->reg_esi,
15
                v6->reg_edi,
16
                v6\!\!-\!\!>\!\!\operatorname{reg\_ebp},
17
                v6\!\!-\!\!>\!\!\operatorname{reg\_esp}\,,
18
                 v6\!\!-\!\!>\!\!\operatorname{reg\_eip}) >\!\!= 256 \ )
19
         log("XException.cpp", 39, 0);
20
       call_write_log(&OutputString);
21
       call_OutputDebugStringA(&OutputString);
22
23
```

4.6 Persistency

The three files Sample B, C and D are copied into the directory

```
1 C:\Documents and Settings\All Users\VirusMap
```

respectively in

```
1 C:\ProgramData\VirusMap (C:\Users\All Users\VirusMap)
```

After that, a new registry entry is set:

```
1 \ | \ HKEY\_CURRENT\_USER \setminus Software \setminus Microsoft \setminus Windows \setminus Current Version \setminus Run
```

which calls mcvsmap.exe (Sample B) after login.

Another option is the installation as a service in

```
1 \begin{tabular}{ll} HKEY\_LOCAL\_MACHINE\SYSTEM\ ControlSet001\ Services\ VirusMap \end{tabular}
```

The key "Imagepath" calls the same binary mcvsmap.exe (Sample B).

4.7 Control

The attacked computer uses TCP and UDP to connect to port 443 on help.yahoo-upgrade.com (122.199.194.197). Unfortunately, the machine at that IP address doesn't seem to reply to our requests anymore on 2013-03-26.

The Passive DNS showed some other associated domains and hostnames with this IP address:

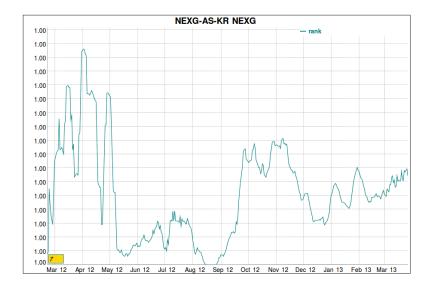
```
help.yahoo-upgrade.com
support.yahoo-upgrade.com
update.ayuisyahooapis.com
support.ayuisyahooapis.com
update.trendmicrosoft.co.in
```

It's highly probable that theses hostnames were also used for other campaigns. You might use these as additional indicators for the detection of potential infections.

4.8 Network and domain information

4.8.1 Network

The IP address is located in the ASN 17877 and the ISP is not a known bulletproof hoster as you can see on its historical BGP ranking evolution.



```
1 inetnum:
                    122.199.128.0\ -\ 122.199.255.255
   netname:
   descr:
                    NexG
   descr:
                    5F Seoul
Academy B/D, 967–6 Daechi–Dong, Gangnam–Gu, 135–280
   descr:
                    Allocated to KRNIC Member.
  descr:
   \operatorname{descr}:
                    If you would like to find assignment
                    information in detail please refer to
   descr:
                    the KRNIC Whois Database at:
   descr:
10 descr:
                    "http://whois.nic.or.kr/english/index.html"
11 descr:
                    KR
12
   country:
13 admin-c:
                    SL1625-AP
                    SL1625-AP
14 tech-c:
15
   remarks:
                    www.nexg.net
                    ALLOCATED PORTABLE
16 status:
17 mnt-by:
                    MNT-KRNIC-AP
                    MNT-KRNIC-AP
18
  mnt-lower:
                    hm-changed@apnic.net 20060606
19
  changed:
20 source:
                    APNIC
21
                    Sanguk Lee
22 person:
23 nic-hdl:
                    SL1625\!-\!\!AP
   e-mail:
                    ip@nexg.net
25
  address:
                    5F SeoulAcademy B/D, 967-6 Daechi-Dong, Gangnam-Gu, 135-280
26 phone:
                    +82-2-538-7060
27
  fax-no:
                    +82-2-571-8998
28
   country:
                    KR.
  changed:
                    hostmaster@nida.or.kr 20050105
```

⁷http://bgpranking.circl.lu/asn_details?asn=17877

```
30 mnt-by:
                   MNT-KRNIC-AP
31 source:
                   APNIC
32
33 inetnum:
                   122.199.128.0 - 122.199.255.255
34 netname:
                   VAAN-KR
35 descr:
                   NexG
36 country:
                   KR
37 admin-c:
                   LS151-KR
38 tech-c:
                   LS151-KR
                   ALLOCATED PORTABLE
39 status:
40 mnt-by:
                   MNT-KRNIC-AP
41 mnt-irt:
                   IRT-KRNIC-KR
42 remarks:
                   This information has been partially mirrored by APNIC from
                   KRNIC. To obtain more specific information, please use the
43 remarks:
                   KRNIC whois server at whois.krnic.net.
44 remarks:
                   hostmaster@nic.or.kr
45 changed:
46
  source:
                   KRNIC
```

4.8.2 Domain

```
Domain Name: YAHOO-UPGRADE.COM
      Registrar: JIANGSU BANGNING SCIENCE & TECHNOLOGY CO. LTD
 3
      Whois Server: whois.55 hl.com
      Referral URL: http://www.55hl.com
 4
 5
      Name Server: DNS5.4CUN.COM
      Name Server: DNS6.4CUN.COM
 6
 7
      Status: ok
      Updated Date: 08-aug-2012
9
      Creation \ Date: \ 18-jul-2011
10
      Expiration Date: 18-jul-2013
11
12 >>> Last update of whois database: Wed, 27 Mar 2013 22:36:13 UTC <<<
13
14 Domain Name: yahoo-upgrade.com
15
16 Registrant Contact:
17 yahoo
18
  yahoo yahoo whiteyoo_123@yahoo.com
19 telephone: +48.56756756756
20 \mid \text{fax}: +48.56732453453
21
  yahoo yahoo yahoo 345345
22 CA
23
24 Administrative Contact:
25 yahoo yahoo whiteyoo_123@yahoo.com
26 telephone: +48.56756756756
  fax: +48.56732453453
28
  yahoo yahoo yahoo 345345
29 CA
30
31 Technical Contact:
32 yahoo yahoo whiteyoo_123@yahoo.com
33 telephone: +48.56756756756
34 \mid \text{fax}: +48.56732453453
35 yahoo yahoo yahoo 345345
36 CA
37
38 Billing Contact:
39 yahoo yahoo whiteyoo_123@yahoo.com
40 telephone: +48.56756756756
```

```
41 | fax: +48.56732453453
42 | yahoo yahoo yahoo 345345
43 | CA
```

4.9 Current version and history of PlugX

A version string can be found in this binary:

```
1 \begin{tabular}{l} d: \work \plug 7.0 (mcvsmap) (fking) j() \shellcode \shellcode \XPlug.h \end{tabular}
```

This could mean PlugX, version 7.0 codename fking, build for mcvsmap. References can be found on the internet for previous versions of this malware family:

```
 \begin{array}{c} 1 \\ d: \work \rangle 1 \\ d: \work \rangle 1 \\ (sxl) \\ shellcode \\ shellcode \\ XPlug.h \\ d: \work \rangle 1 \\ (sxl) \\ shellcode \\ Shellcode \\ Shellcode \\ 1 \\ (sxl) \\ Shellcode \\ Shellco
```

A Appendix

A.1 Indicators of Compromise (IOC)

This section summarizes the known indicators of compromise. The list might not be exhaustive, but the existence of any or all of the following indicators might help to discover an infection.

A.1.1 Pipes

```
1 \PIPE\a$PID
2 \PIPE\b$PID
3 \PIPE\RUN_AS_USER($PID)
```

(where \$PID is the process ID of the active malicious process)

A.1.2 Files and directories

- Static files (dropped files)
 - update.exe

```
      1 MD5:
      f1f48360f95e1b43e9fba0fec5a2afb8

      2 SHA1:
      70ceb467db7b0161d22e4545479f747417b9705a

      3 SHA-256:
      2bc5ce39dd9afe2157448d3f6d8cb9c549ed39543d159616e38480b9e6c11c49
```

mcvsmap.exe

```
MD5: 4e1e0b8b0673937415599bf2f24c44ad
SHA1: 9224de3af2a246011c6294f64f27206d165317ba
SHA-256: ae16e10e621d6610a3f7f2c7122f9d1263700ba02d1b90e42798decb2fe84096
```

- McUtil.DLL

Page 18 of 21

```
      1
      MD5:
      ad4a646b38a482cc07d5b09b4fffd3b3

      2
      SHA1:
      ae0f9bf2740d00c5d485827eb32aca33feaa3a90

      3
      SHA-256:
      0a99238e1ebebc47d7a89b2ccddfae537479f7f77322b5d4941315d3f7e5ca48
```

- McUtil.DLL.PPT

```
1 MD5: 545bb4365a9b7cdb6d22844ebeedda93
2 SHA1: a267f1183b4ff843d68a63264846abf78cc71d1f
3 SHA-256: d4fe890a08d4dd44b58a3b85b2a7e89536338099c1c42a9b7e85f4007b0a37b7
```

- UAC.TMP

```
    1 MD5:
    52df5c2c07433e2a8f054c2347acb3b4

    2 SHA1:
    8051474c1fc0d8f404a42ea32eca1699e54f02e1

    3 SHA-256:
    dc09091e5d0ce03c6144748f17bd636f2f0b2ca56f88b550c1d48860596dbdb1
```

Files and/or directories might be hidden and carry the system flag

```
C:\ProgramData\VirusMap (Windows 7)
C:\Users\All Users\VirusMap (Windows 7)
C:\Documents and Settings\All Users\VirusMap (Windows XP)

%ALLUSERSPROFILE%\SxS\bug.log
C:\Documents and Settings\All Users\VirusMap\NvSmart.hlp
```

A.1.3 Registry

```
1 | HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Services\VirusMap and a key referencing Sample B | HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run and a key referencing Sample B
```

A.1.4 Network (hostname and destination IP addresses)

```
1 help.yahoo-upgrade.com
2 122.199.194.197
```

A.2 References

- WHITE PAPER: PLUG X PAYLOAD EXTRACTION
 - http://www.contextis.com/files/PlugX_-_Payload_Extraction_March_2013_1.
 pdf
 - Context Information Security http://www.contextis.com/
 - Published 2013-03-22
- An Analysis of PlugX
 - http://lastline.com/blog.php
 - Lastline http://www.lastline.com/
 - no publication date found
- PlugX is becoming mature
 - http://www.securelist.com/en/blog/208193974/PlugX_is_becoming_mature
 - Kaspersky Lab http://www.kaspersky.com/

- Published 2012-11-27
- Unplugging PlugX Capabilities
 - http://blog.trendmicro.com/trendlabs-security-intelligence/unplugging-plugx-capabilities
 - TrendMicro http://www.trendmicro.eu/
 - Published 2012-09-17
- Tracking down the author of the PlugX RAT
 - http://labs.alienvault.com/labs/index.php/2012/tracking-down-the-author-of-the-plugx-ra
 - AlienVault http://labs.alienvault.com
 - Published 2012-09-13

A.3 VirusTotal results

• Sample A

```
1 MicroWorld-eScan:
                           Trojan. Agent. AZDK
 2 nProtect:
                           Trojan . Agent . AZDK
 3 McAfee:
                          RDN/Generic BackDoor!gq
 4 Malwarebytes:
                           Trojan. Dropper.CH
 5 | Symantec:
                          W\!S.\,Reputation.1
 6 Norman:
                           Agent.APIJH
 7 TrendMicro-HouseCall: BKDR POISON.PQ
 8 Avast:
                           Win32: Gulpix-B [Trj]
9 Kaspersky:
                           Backdoor.Win32.Gulpix.bo
10 Bit Defender:
                           Trojan . Agent . AZDK
11 Agnitum:
                           Backdoor. Gulpix! EFaRR6zLtc4
12 ViRobot:
                           Backdoor. Win32.A. Gulpix.370702.B
13 Comodo:
                           UnclassifiedMalware
14 F-Secure:
                           Trojan . Agent . AZDK
15 DrWeb:
                           Trojan. Click2.52215
16 VIPRE:
                           Trojan . Win32 . Generic !BT
17 AntiVir:
                          TR/Agent.azdk.3
18 TrendMicro:
                          BKDR_POISON.PQ
19 McAfee-GW-Edition:
                          RDN/Generic BackDoor!gq
                           Troj/Agent-AATT
20 Sophos:
21 Kingsoft:
                           Win32. Hack. Gulpix. (kcloud)
22 Microsoft:
                           Backdoor: Win32/Plugx.A
23 GData:
                           Trojan . Agent . AZDK
24 AhnLab-V3:
                           {\it Backdoor/Win32.Gulpix}
25 Ikarus:
                           Backdoor.Win32.Gulpix
26 Fortinet:
                           W32/Gulpix.BO!tr.bdr
27 AVG:
                           Agent4.AKAP
                           Trj/CI.A
28 Panda:
29 Scanned: 2013-03-21 \ 04:01:12-45 \ scans-28 \ detections (62.0\%)
```

• Sample B (mcvsmap.exe)

```
1 Scanned: 2013-03-21 13:29:45 - 44 scans - 0 detections (0.0%)
```

• Sample C (McUtil.DLL)

Page 20 of 21

```
1 MicroWorld-eScan:
                             Trojan . Agent . AZDK
                            Trojan . Agent . AZDK
 2 nProtect:
 3 McAfee:
                            RDN/Generic BackDoor!gt
 4 Malwarebytes:
                            Backdoor. Gulpix
 5 Symantec:
                            WS. Reputation.1
 6 Norman:
                            Agent.APIJH
 7 | TrendMicro-HouseCall: \overrightarrow{TROJ}_GEN.RCBCRCJ
   Avast:
                             Win32:Gulpix-B [Trj]
                            Backdoor. Win32. Gulpix. bo
 9 Kaspersky:
10 BitDefender:
                            Trojan . Agent . AZDK
11 Agnitum:
                             Backdoor.Gulpix!EFaRR6zLtc4
12 Comodo:
                             Unclassified Malware\\
13 F-Secure:
                            Trojan . Agent . AZDK
14 DrWeb:
                            Trojan. Click2.52215
15 VIPRE:
                            Trojan . Win32 . Generic !BT
16 AntiVir:
                            TR/Agent.azdk.2
17
   TrendMicro:
                            TROJ GEN.RCBCRCJ
18 McAfee-GW-Edition:
                            RDN/\operatorname{Generic\ BackDoor!gt}
19 Sophos:
                            Troj/Agent-AATT
20 Microsoft:
                            Backdoor: Win32/Plugx.A
21 GData:
                            Trojan . Agent . AZDK
                            W32/Backdoor.IYCB-5867
22 Commtouch:
23 Ikarus:
                            Backdoor. Win32. Gulpix
24 Fortinet:
                            W32/Gulpix.BO!tr.bdr
25 AVG:
                            Agent4.AKAP
26 Panda:
                            \mathrm{Trj}/\mathrm{CI}.\mathrm{A}
   Scanned: 2013-03-21 13:46:10 - 44 scans - 26 detections (59.0%)
```

• Sample D (McUtil.DLL.PPT)

```
Not uploaded to VirusTotal.
```

• Sample E

```
1 Not uploaded to VirusTotal.
```

• Sample F (UAC.TMP)

```
1 | Panda: Suspicious file
2 | Scanned: 2012-09-20 02:33:55 - 43 scans - 1 detections (2.0%)
```