BlackSuit Ransomware

Analyzing the Approach of a Ransomware Gang

Summary

Large Cobalt Strike beacon executed; no immediate follow-up; access method unclear.

System and environment enumeration; attacks using Rubeus and Sharphound.

Beacon transferred and executed on another workstation; credentials harvested; RDP used for further spread.

SystemBC deployed for persistence; switch to AWS for command and control; brief inactivity.

New beacons and discovery efforts; failed ADFind attempts.

Successful execution of discovery tools; deployment of BlackSuit ransomware; encryption completed in 15 days.

LianGroup's Decision: Transforming the Analysis into a Podcast

We're Launching a Podcast!

Initial Access

Earliest Sign of Threat Actor Presence:

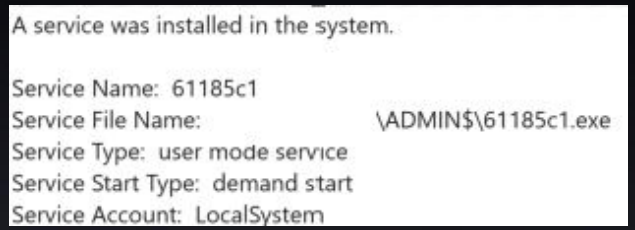
- Execution of a **Cobalt Strike** beacon (RtWin64.exe).
- Initial access point for the beacon's deployment remains undetermined despite thorough investigation.

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

Execution (CS 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.**

EventID 7045 in Windows System logs **shows the services created** on the system.



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

Execution (CS PsExec)

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

Execution (CS PsExec_Psh)

The **psexec_psh** module doesn't copy a binary to the target. Instead, it executes a PowerShell one-liner using the pattern

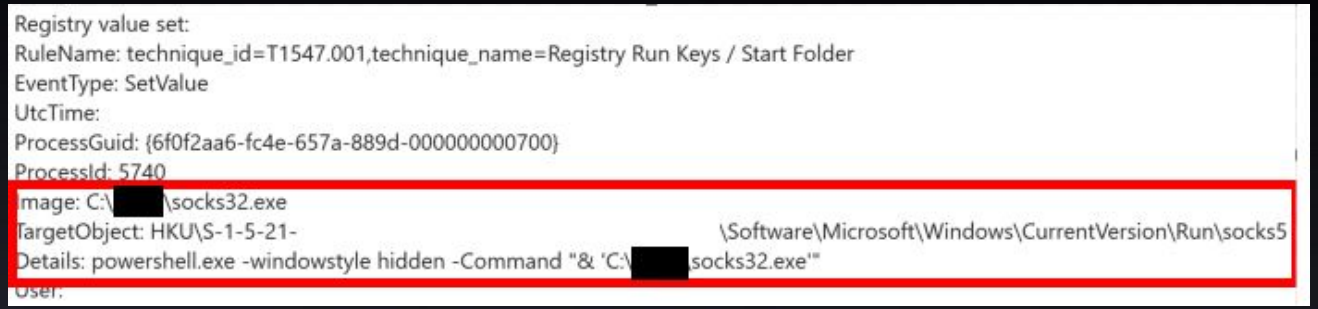
%COMSPEC% /b /c start /b /min powershell -nop -w hidden -encodedcommand ….

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. This registry key was configured to use PowerShell to launch a **SystemBC backdoor** named **socks32.exe** .

Sysmon EventID 13 (Registry value set) displays changes to a registry key value.



SystemBC

win.systembc (Back to overview)

SystemBC

aka: Coroxy, DroxiDat

VTCollection

SystemBC is a proxy malware leveraging SOCKS5. Based on screenshots used in ads on a underground marketplace, Proofpoint decide

SystemBC has been observed occasionally, but more pronounced since June 2019. First samples goes back to October 2018.

References

2024-08-26 · The DFIR Report · The DFIR Report
BlackSuit Ransomware
BlackSuit | Cobalt Strike | SystemBC

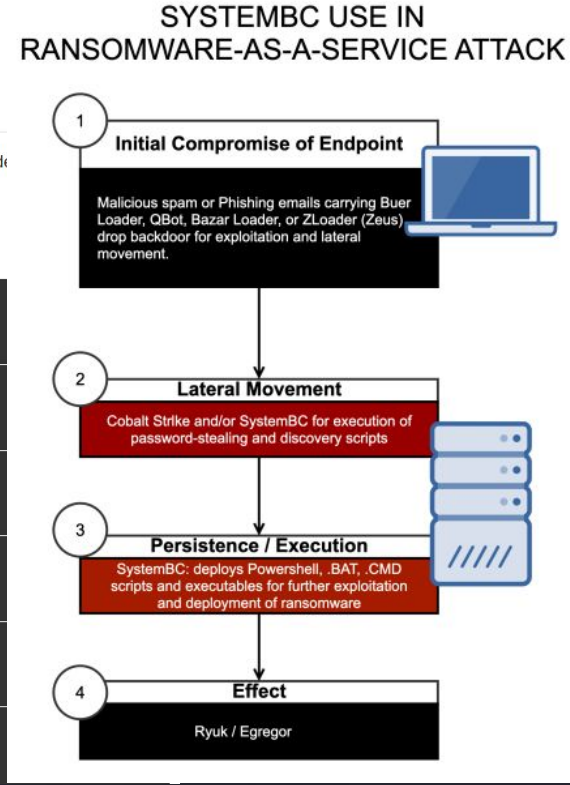
2024-07-29 · Mandiant · Ashley Pearson, Jake Nicastro, Joseph Pisano, Josh Murchie, Joshua Shilko, Raymond Leong
UNC4393 Goes Gently into the SILENTNIGHT
Black Basta | QakBot | sRDI | SystemBC | Zloader | UNC4393

2024-05-30 · Europol · Europol
Largest ever operation against botnets hits dropper malware ecosystem
BumbleBee | IcedID | SmokeLoader | SystemBC | TrickBot

2024-01-19 · Kroll · David Truman
Inside the SYSTEMBC Command-and-Control Server
SystemBC

2023-11-12 · Github (vc0RExor) · Aaron Jornet
The Swiss Knife: SystemBC | Coroxy
SystemBC

2023-09-12 · 🇫🇷 · ANSSI · ANSSI
FIN12: A Cybercriminal Group with Multiple Ransomware
BlackCat | Cobalt Strike | Conti | Hive | MimiKatz | Nokoyawa Ransomware | PLAY | Royal Ransom | Ryuk | SystemBC



Persistence (Registry Run Key)

The data is a string (REG_SZ) that starts with **powershell.exe windowstyle -hidden**, followed by a command concatenated with the current executable name. The executable name is obtained using GetModuleFileNameA with a null hModule as the first parameter.

```
RegCreateKeyExA(hKey, lpSubKey, 0, 0, 0, 0×F003Fu, 0, &phkResult, &dwDisposition);
GetModuleFileNameA(0, Filename, 0×100u);
wsprintfA(powershellCommand, "powershell.exe -windowstyle hidden -Command \"& '%s'\"", Filename);
commandSize = length(powershellCommand);
RegSetValueExA(phkResult, lpValueName, 0, dwType, (const BYTE *)powershellCommand, commandSize + 1);
```

Persistence (Scheduled Task)

SystemBC can **create scheduled tasks** using **COM**, as shown in the following example. Although other reports indicate SystemBC uses this feature, it was likely **not utilized in our case**, as **no evidence** of scheduled task creation was found during our investigation.

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

```
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 )
  {
```

Privilege Escalation (Pass-the-Hash)

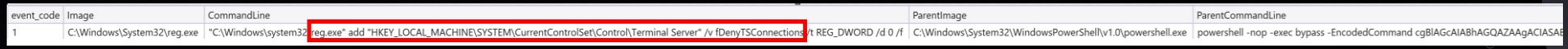
On a workstation that the threat actor moved laterally to, we observed the use of **named pipes** .

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

Typically, in **Cobalt Strike** , **this behavior** is associated with the **getsystem command** for privilege escalation. However, in this instance, the parent process was not services.exe, and the **threat actor was already running as SYSTEM** . This activity correlated with **pass-the-hash behavior** noted in Lateral Movement. The **threat actor switched to the context of a domain administrator** and **continued moving laterally using Cobalt Strike** , so we attribute this activity to pass-the-hash command execution rather than getsystem.

Defense Evasion (Modify Registry)

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

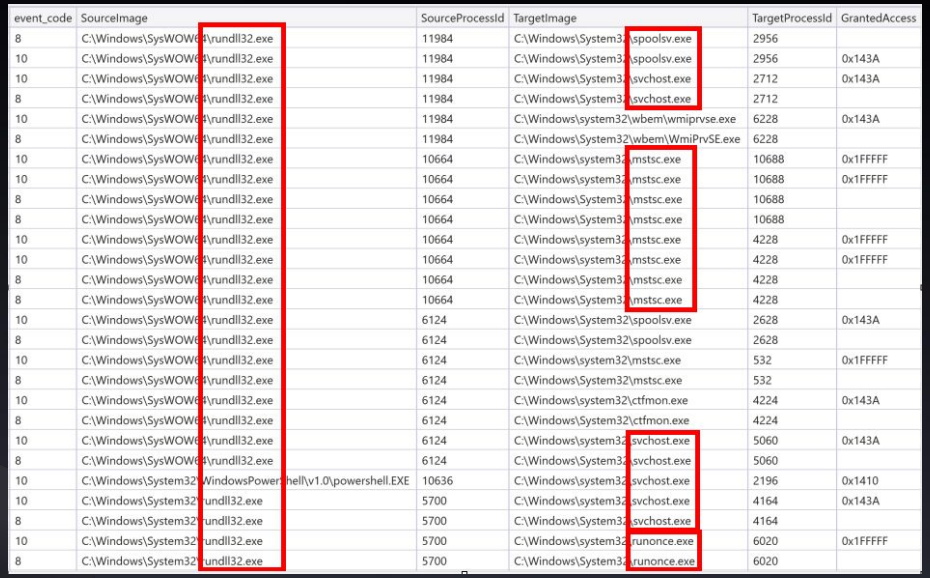


event_code	Image	CommandLine	ParentImage	ParentCommandLine
1	C:\Windows\System32\reg.exe	"C:\Windows\system32\reg.exe" add "HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Terminal Server" /v fDenyTSConnections /t REG_DWORD /d 0 /f	C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe	powershell -nop -exec bypass -EncodedCommand cgBlAGcAIABhAGQAZAAgACIASAE

Setting the registry key HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Terminal Server\DenyTSConnections to 0 allows terminal server connections to the host.

Defense Evasion (Process Injection)

Given the threat actor's extensive use of Cobalt Strike beacons, we anticipated **process injection** as a method to **evade 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	

Defense Evasion (Process Injection)

These injections can then be confirmed using methods such as 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 (Rubeus)

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

Weaponization

One common way attack tools are detected is through the weaponization vector for the code. If Rubeus is run through PowerShell (this includes Empire) the standard PowerShell V5 protections all apply (deep script block logging, AMSI, etc.). If Rubeus is executed as a binary on disk, standard AV signature detection comes into play (part of why we do not release compiled versions of Rubeus, as brittle signatures are silly ;). If Rubeus is used as a library then it's susceptible to whatever method the primary tool uses to get running. And if Rubeus is run through unmanaged assembly execution (like Cobalt Strike's `execute_assembly`) cross-process code injection is performed and the CLR is loaded into a potentially non-.NET process, though this signal is present for the execution of any .NET code using this method.

Credential Access (AS-REP Roasting)

AS-REP Roasting is a technique used to obtain password hashes for users who have the "Do not require Kerberos preauthentication" setting enabled.

Rubeus writes the output of the AS-REP Roasting results to a file.

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

Credential Access (AS-REP Roasting)

Indications of AS-REP Roasting can be detected by checking for Windows **EventID 4768 on** the target **domain controller**. This **event** represents a **request for Authentication Tickets (TGT)** where the "**Pre-Authentication Type**" is set to **0**, indicating that **no pre-authentication (or 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:

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

A large number of Kerberos Authentication Tickets were requested during the AS-REP roasting activity.

Credential Access (Kerberoasting)

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

Rubeus was executed in memory by Cobalt Strike, generating Kerberoast output.

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

EventID 4769 on a **domain controller** indicates a **request for Kerberos tickets using weak encryption**.

Credential Access (Kerberoasting)

AS-REP Roasting	Kerberoasting
User accounts with "Do not require preauthentication"	Service accounts with SPNs
Requests AS-REP from Domain Controller	Requests TGS for a registered SPN
No special permissions needed	Any domain user can perform this
Encrypted TGT (Ticket Granting Ticket)	Encrypted TGS (Ticket Granting Service)
Ensure pre-authentication is enabled for all users	Strong password, and password rotation
Less common; relies on misconfiguration	More common; targets service accounts

Credential Access (Kerberoasting)

The threat actor accessed the LSASS memory on a workstation using a specific access request of **0x1010**.
This included the **0x0010** access right, required to **read memory** via the **ReadProcessMemory** function.
The request came from a **process** that had been **injected** with **Cobalt Strike**.
Sysmon **Event ID 10** indicates that **mstsc.exe accessed the lsass.exe** process with an access mask of **0x1010**.

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

Discovery

Hands On Keyboard

Discovery on the beachhead host began six hours after access, starting with "systeminfo" and "nltest /dclist" commands.

ADFind

ADFind is a command-line tool used to query and extract information from Active Directory, often utilized for enumeration by attackers.

Windows Utilities

The threat actor executed multiple discovery commands using various Windows utilities at different times throughout the intrusion.

Sharphound

BloodHound is a tool used to map Active Directory relationships and identify potential attack paths within a network.

Get-DataInfo.ps1

Get-DataInfo.ps1 is a PowerShell script used to collect detailed system and network information, often for reconnaissance purposes.

Administrator Consoles

On the final day, the threat actor accessed the DNS and Group Policy administrative consoles before deploying ransomware across the environment.

Lateral Movement (Pass the hash)

Logon activity analysis showed evidence of **pass-the-hash** attacks, with Windows Security logs **(Event ID 4624)** indicating **logon type 9** and "**seclogo**" as the Logon Process.

An account was successfully logged on.

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

Logon Information:
- Logon Type: 9
- Restricted Admin Mode: -
- Virtual Account: No
- Elevated Token: Yes

Impersonation Level: Impersonation

New Logon:
- Security ID: S-1-5-18
- Account Name: SYSTEM
- Account Domain: NT AUTHORITY
- Logon ID: 0x20AA98C2
- Linked Logon ID: 0x0
- Network Account Name:
- Network Account Domain:
- Logon GUID: {00000000-0000-0000-0000-000000000000}

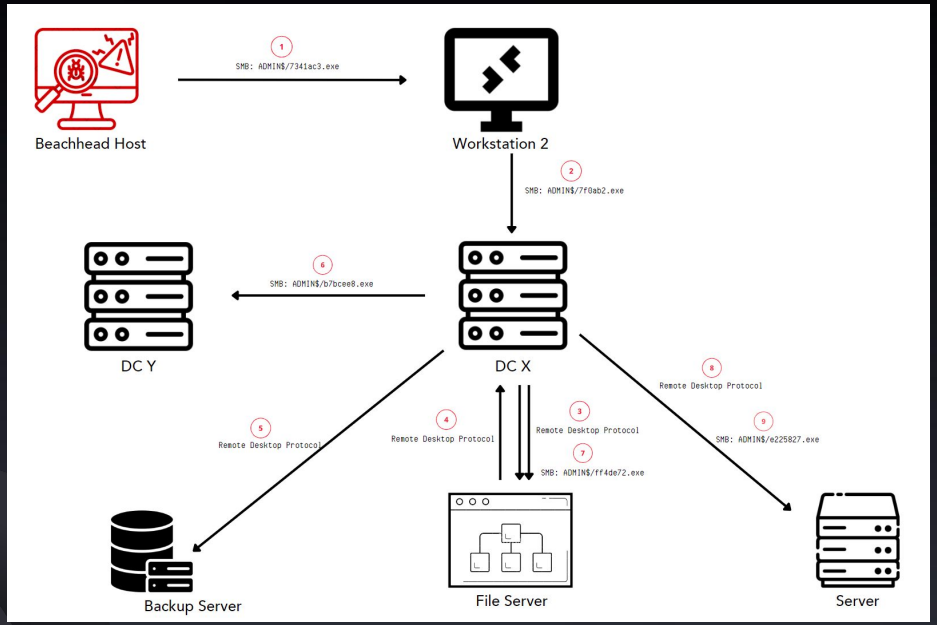
Process Information:
- Process ID: 0x1dc4
- Process Name: C:\Windows\System32\svchost.exe

Network Information:
- Workstation Name: -
- Source Network Address: ::1
- Source Port: 0

Detailed Authentication Information:
- Logon Process: seclogo
- Authentication Package: Negotiate
- Transited Services: -
- Package Name (NTLM only): -

Lateral Movement (Pass the hash)

The threat actor employed three main lateral movement methods: using **Cobalt Strike** with **SMB ADMIN$ shares** for beacon distribution, **Remote Desktop Protocol** to access Backup Server and File Server and conduct discovery, and the **hidden SMB share C$** to **deploy ransomware**. **A domain controller served as the primary pivot point.** Overview of lateral movement with SMB ADMIN$ shares and RDP :



What's going on?!