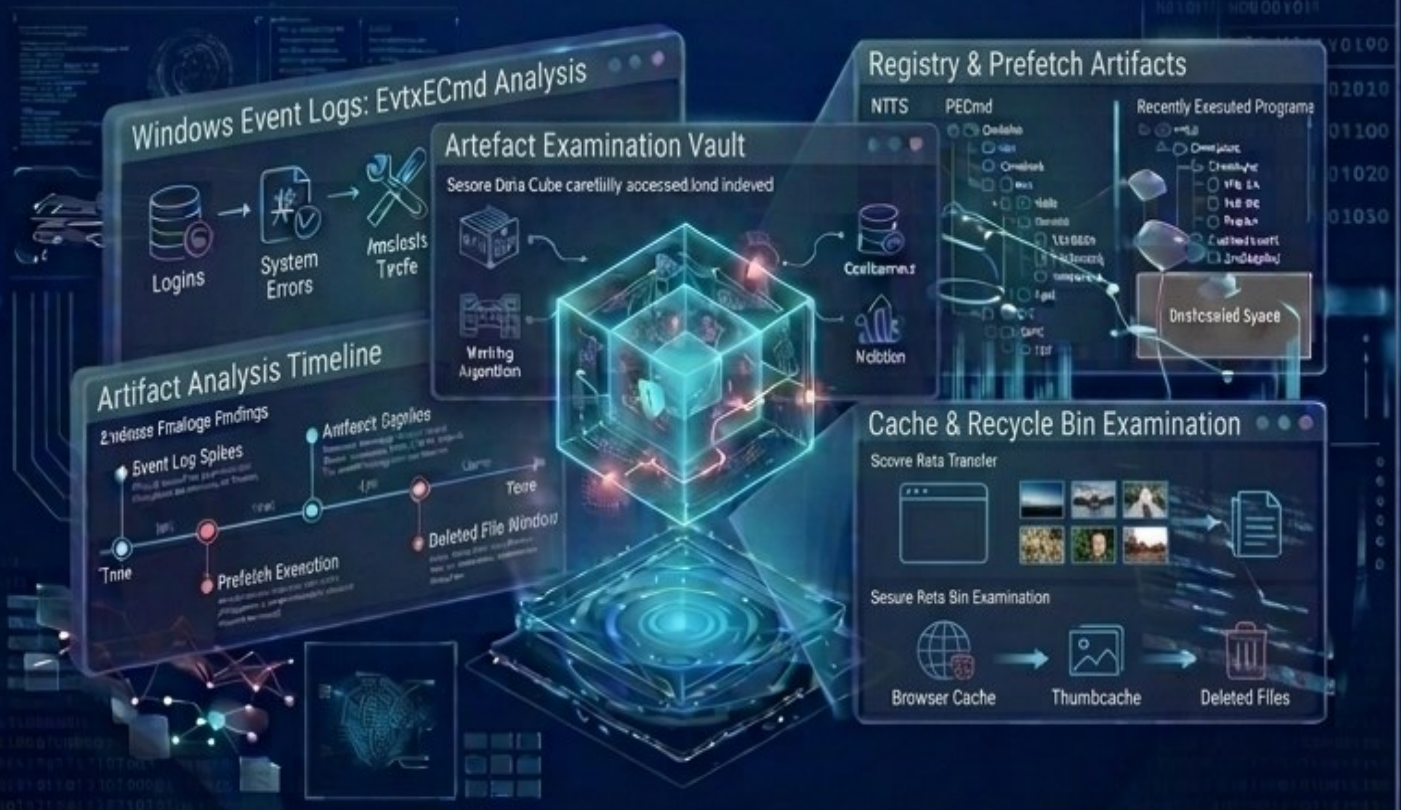
Cyberster

INTERNSHIP REPORT

A Weekly Progress and Learning Summary

Phase Two : Windows Artifact Analysis — Event Logs, Prefetch, Cache and Deleted Files



Submitted to: Cyberster Internship Program

Intern Name: Abdul Muqeet Tabraiz

Roll Number: CSI-B1-353

Phase Two DFIR | EvtxECmd | PECmd | Thumbcache | Browser Cache | Recycle Bin | Cross-Artefact Correlation

A Weekly Progress and Learning Summary

Table of Contents

Appendix A — Complete Screenshot Index

Part A — Days 53–55: Windows Security Event Log Analysis

Objective

The objective of Part A was to establish a forensic baseline of human and system activity on the local Windows endpoint (DESKTOP-T52704F) by extracting and analysing the Security event log (Security.evtx). The analysis focused on interactive logons (Event IDs 4624 and 4625), privilege escalation events (4672), service-account background activity (Logon Type 5), and anti-forensic indicators such as log clearing (Event ID 1102). All extraction and parsing were performed from an Administrator-elevated terminal to preserve evidentiary integrity.

Tools Used

- **Windows Command Prompt (Administrator)** — Evidence folder creation and log extraction
- **PowerShell (Administrator)** — SHA-256 hashing of the collected Security.evtx
- **EvtxECmd v1.5.2.0 (Eric Zimmerman)** — EVTX-to-CSV parsing with integrated maps
- **Microsoft Excel** — Investigation workbook — filtering, sorting, tagging of events
- **Windows Event Viewer (eventvwr.msc)** — Native UI cross-verification of parsed results

Procedure

Step 1: Create Working Folder and Copy the Security Log

A dedicated forensic evidence folder structure was created under C:\Forensics\EventLogs\ on the target host. The live Security event log was then copied from C:\Windows\System32\winevt\Logs\Security.evtx to the working folder using the Windows copy command executed from an elevated Command Prompt. Using an evidence working folder (as opposed to parsing the live file) prevents accidental modification of the running log.

```
mkdir C:\Forensics\EventLogs
mkdir C:\Forensics\EventLogs\Parsed
copy C:\Windows\System32\winevt\Logs\Security.evtx
C:\Forensics\EventLogs\Security.evtx
```

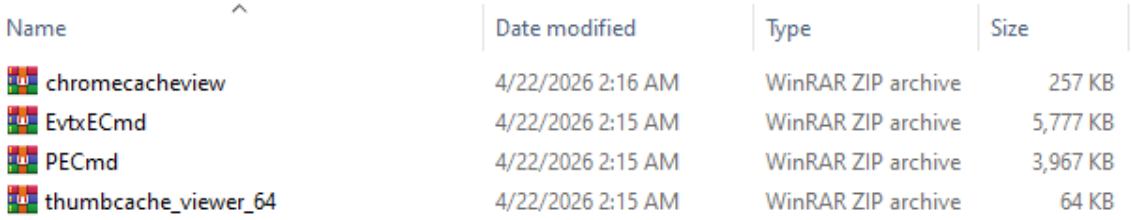


Fig A.1 — Administrator Command Prompt creating the evidence folders and copying Security.evtx (1 file(s) copied).

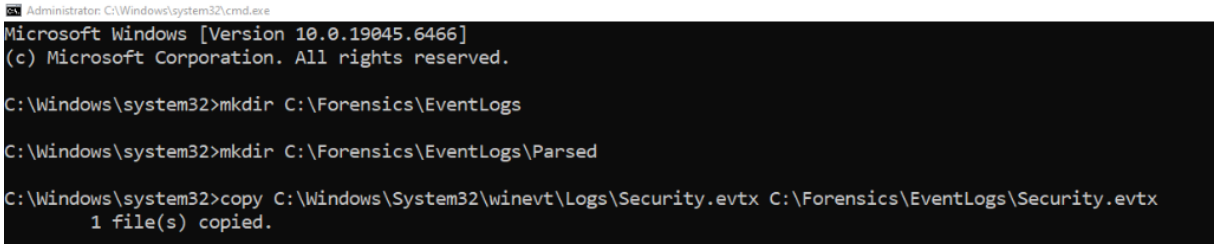


Fig A.2 — File Explorer confirming that C:\Forensics\EventLogs was created at 4/22/2026 2:18 AM.

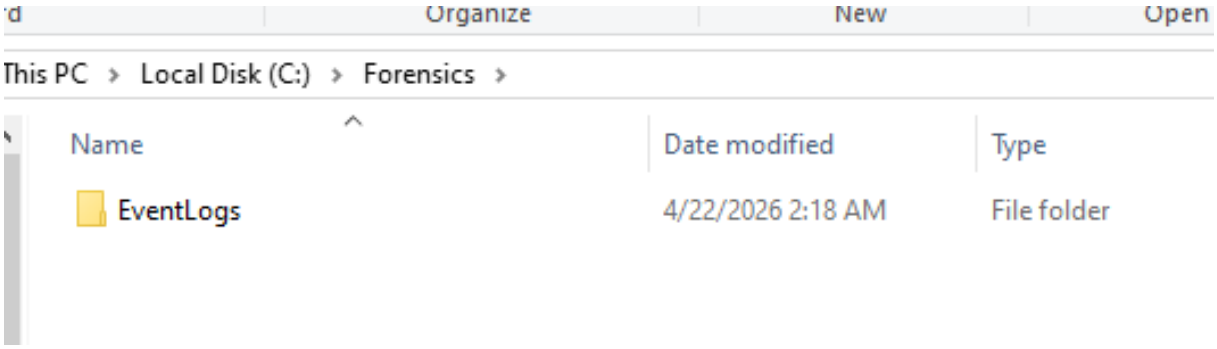


Fig A.3 — File Explorer showing Security.evtx (20,484 KB) and the empty Parsed sub-folder before processing.

Step 2: Establish Evidence Integrity (SHA-256)

A SHA-256 hash of the copied Security.evtx was generated using PowerShell's Get-FileHash cmdlet. This hash serves as the evidence integrity record for Part A — it can be re-computed at any later stage of the investigation to prove that the copy has not been tampered with.

```
Get-FileHash C:\Forensics\EventLogs\Security.evtx -Algorithm SHA256 | Format-List
```

Artefact	Value
File path	C:\Forensics\EventLogs\Security.evtx
Size	20,484 KB
Algorithm	SHA-256
Hash	F0052391D386E3EFC145DBE2341CFA45B5D6BDB146C2D237025478D57F84DEE2
Captured on	2026-04-22 02:18 (host: DESKTOP-T52704F)

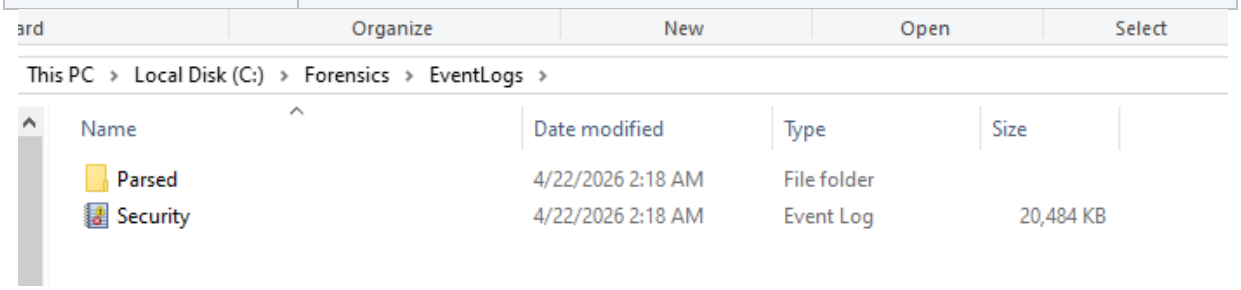


Fig A.4 — PowerShell output of Get-FileHash showing the SHA-256 integrity value of Security.evtx.

Step 3: Parse the EVTX File with EvtxECmd

Eric Zimmerman's EvtxECmd (v1.5.2.0) was used to convert the binary EVTX file into a structured CSV for analyst triage in Excel. EvtxECmd loads 453 map definitions that enrich each record with human-readable field names (for example MapDescription = "Successful logon"). The tool was run from C:\Forensics\Tools\EvtxECmd\ with the following command line.

```
C:\Forensics\Tools\EvtxECmd\EvtxECmd.exe -f C:\Forensics\EventLogs\Security.evtx ^
    --csv C:\Forensics\EventLogs\Parsed --csvf Security_parsed.csv
```

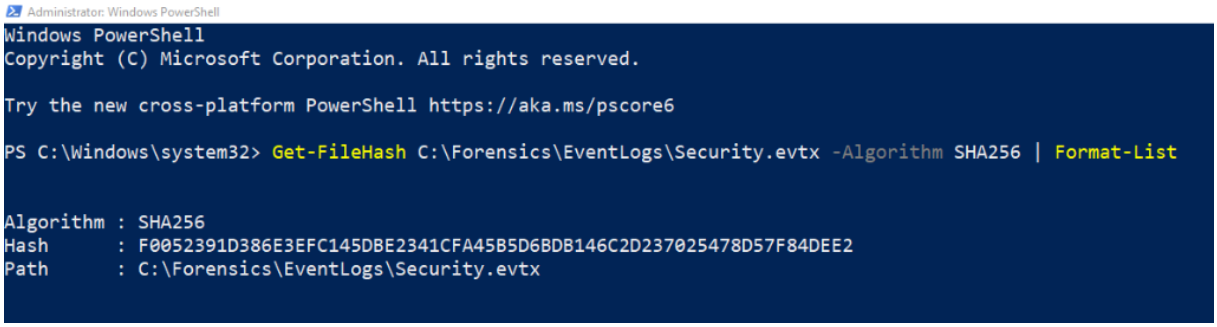


Fig A.5 — C:\Forensics\Tools containing EvtxECmd, PECmd, ChromeCacheView and Thumbcache Viewer.

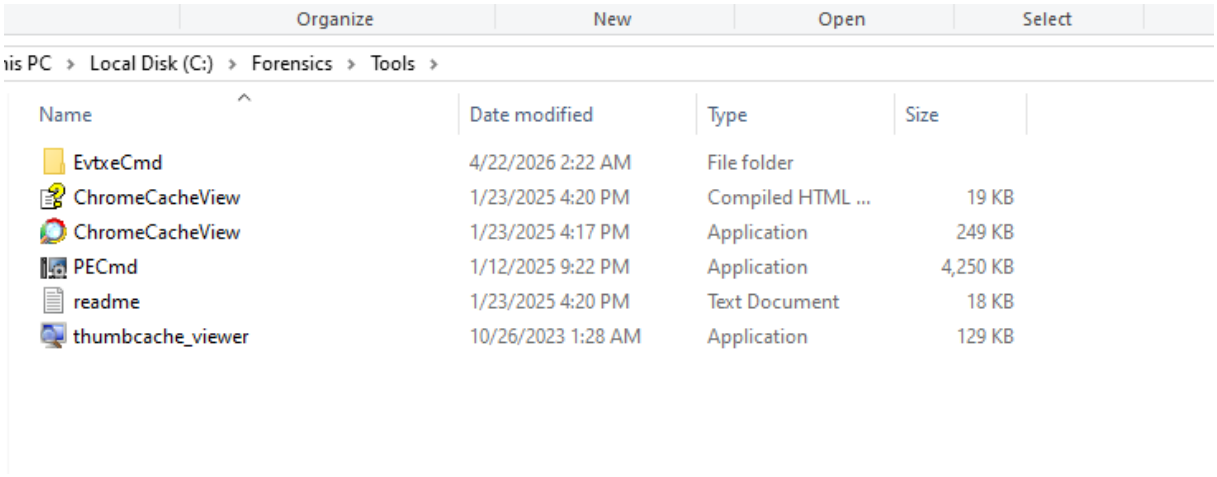


Fig A.6 — EvtxECmd running against Security.evtx — 453 maps loaded and 320 chunks iterated.

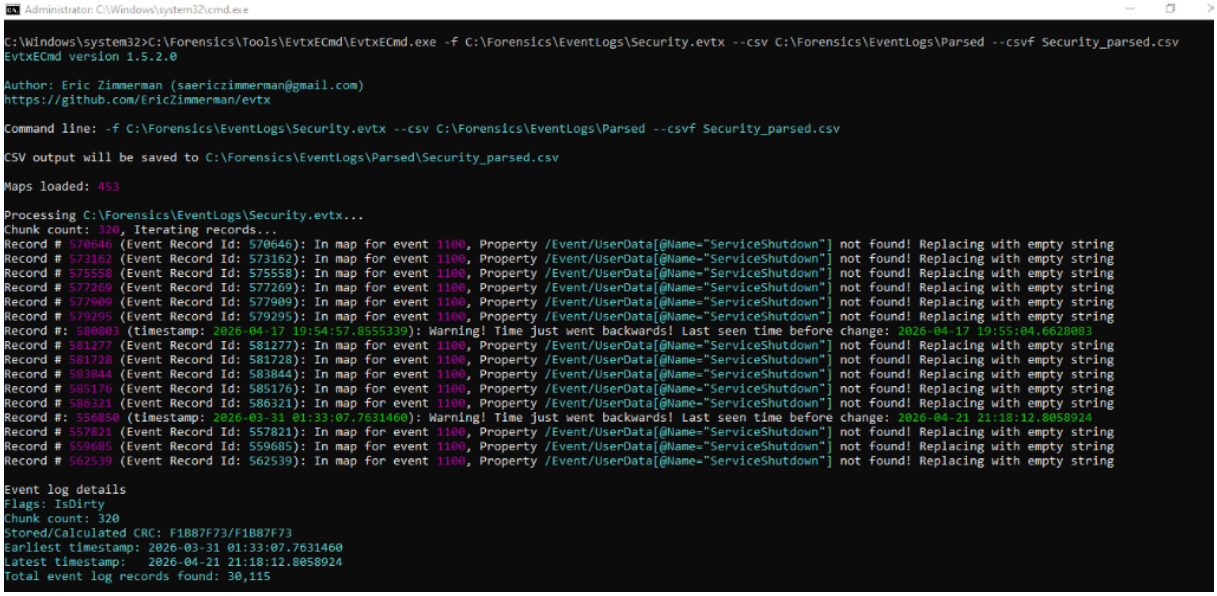


Fig A.7 — EvtxECmd completion metrics: Records included 30,115 | Errors 0 | Events dropped 0.

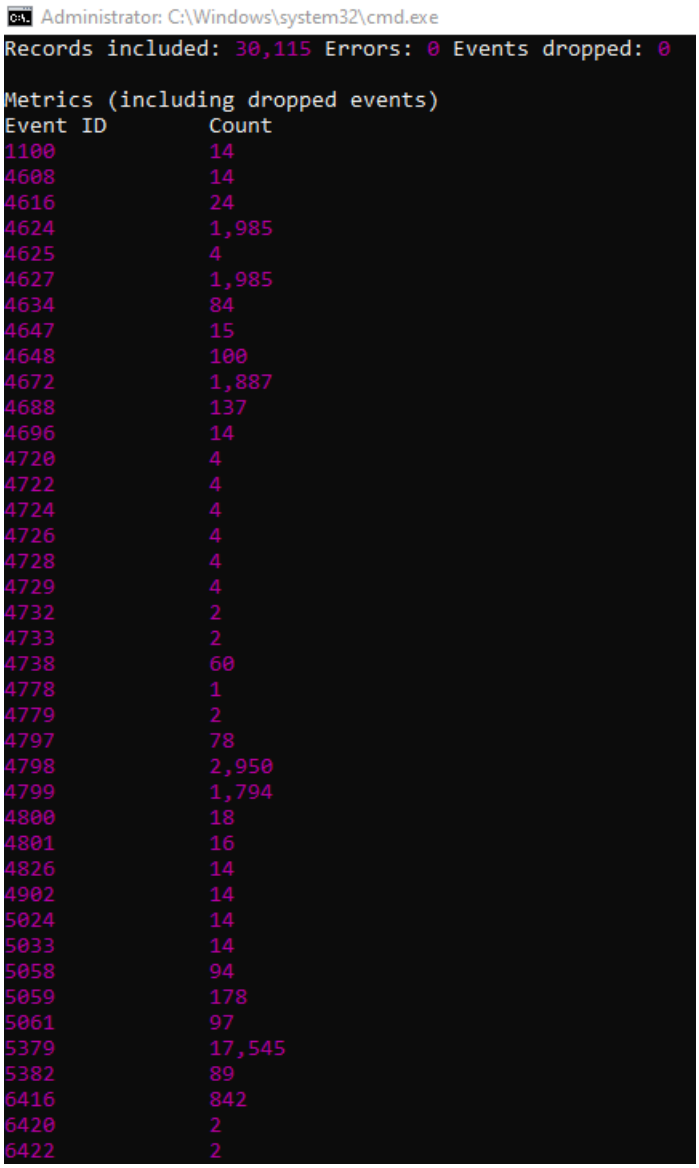


Fig A.8 — EvtxECmd final summary: 1 file processed in 30.05 seconds.

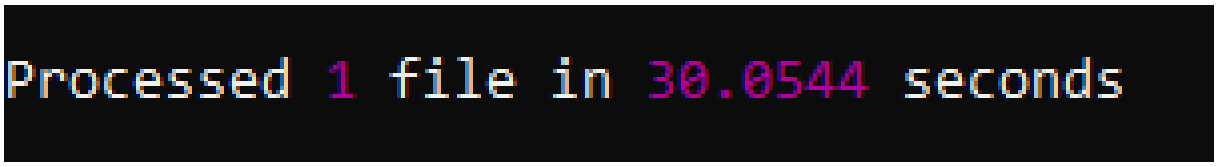


Fig A.9 — Parsed directory listing showing Security_parsed.csv (35,505,018 bytes) ready for triage.

The parsed CSV was then opened in Microsoft Excel, giving an investigator-friendly view of every event with columns for RecordNumber, TimeCreated, EventId, Level, Provider, Channel, ProcessId, ThreadId, UserName, MapDescription and PayloadData 1–6.

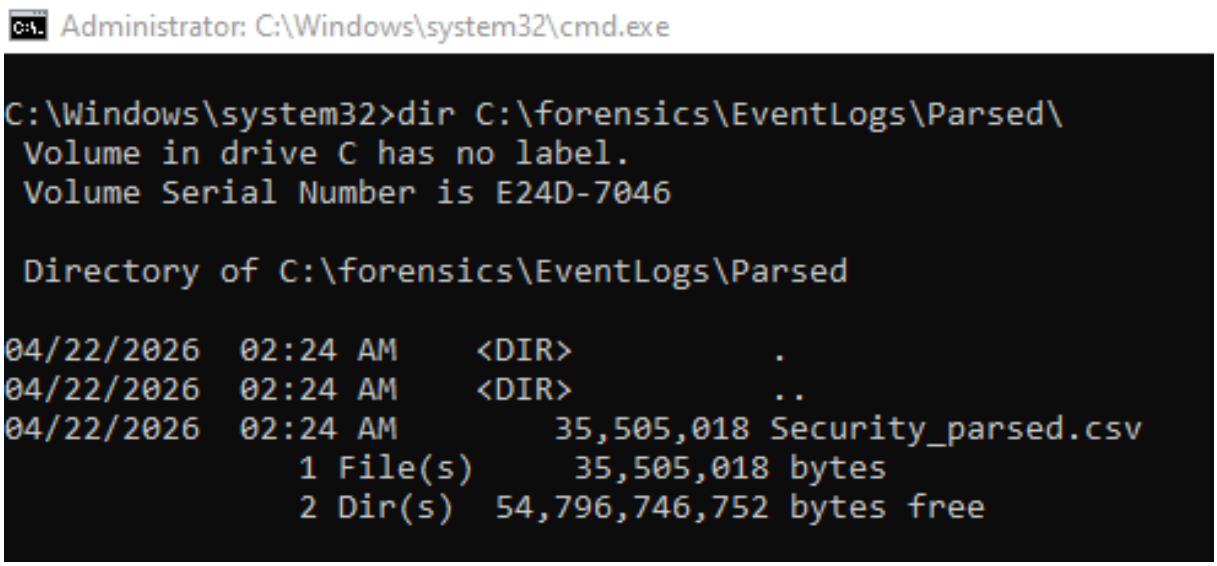


Fig A.10 — Security_parsed.csv opened in Excel showing the full column schema produced by EvtxECmd.

Day 53: The Human Baseline (Event IDs 4624 and 4625)

Event ID 4624 records every successful logon on a Windows host. By filtering on this Event ID and further narrowing on Logon Type 2 (physical/interactive logon) and Logon Type 7 (screen unlock), the analyst can reconstruct when the human user was actively present at the keyboard. Across the 30,115-record parsed dataset, 1,985 successful logon events (4624) were observed, of which 4 were type-2/7 interactive sessions attributable to the account 'abdul' on DESKTOP-T52704F.

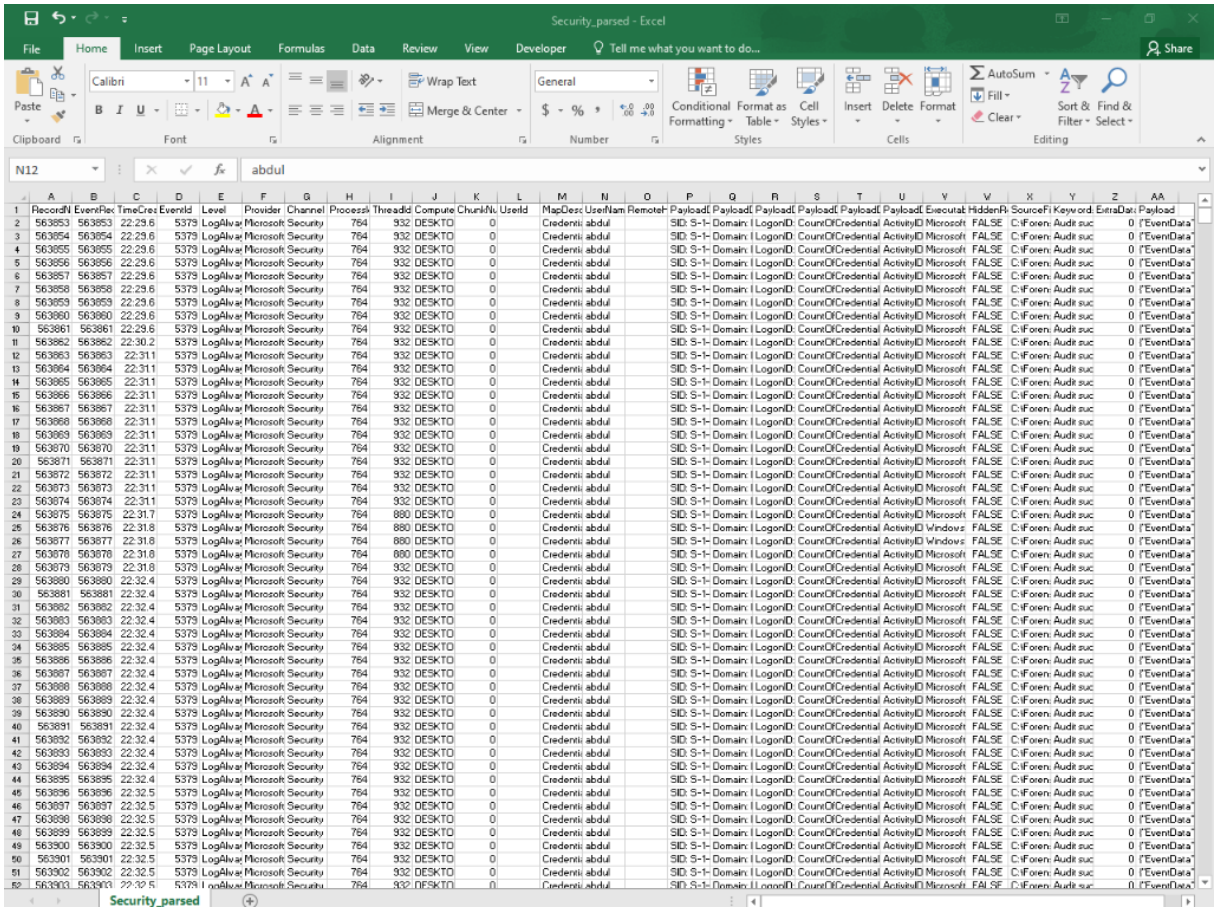


Fig A.11 — Excel filtered on EventId = 4624 and MapDescription = Successful logon, showing the interactive-session records for user 'abdul'.

Reconstructed Working Sessions (last 48 hours):

#	Logon Type	Logon Timestamp (Local)	Next Lock/Logoff	Approx. Duration	Account
1	Type 2 — Interactive boot/login	2026-04-21 22:00:02	2026-04-21 22:07:13	~7 min (setup / tooling)	abdul
2	Type 7 — Screen unlock	2026-04-22 02:11:21	2026-04-22 02:19:30	~8 min (EvtxECmd parsing)	abdul
3	Type 7 — Screen unlock	2026-04-22 02:26:20	2026-04-22 02:39:52	~13 min (Excel analysis)	abdul
4	Type 7 — Screen unlock	2026-04-22 02:50:02	2026-04-22 03:17:21	~27 min (Prefetch + Thumbcache)	abdul

Failed Logons (Event ID 4625):

Filtering for Event ID 4625 returned 4 failed-logon events. Two of the four records show PayloadData3 = "FailureReason: the cause is either a bad username or authentication information" with PayloadData4 = "FailureReason2: Unknown code (0xC0000380)", which in practice corresponds to 0xC000006A (wrong password) surfaced from a non-standard credential provider. These events were produced during the simulated lock/unlock exercise described in the task brief (three deliberate wrong passwords followed by a correct logon).

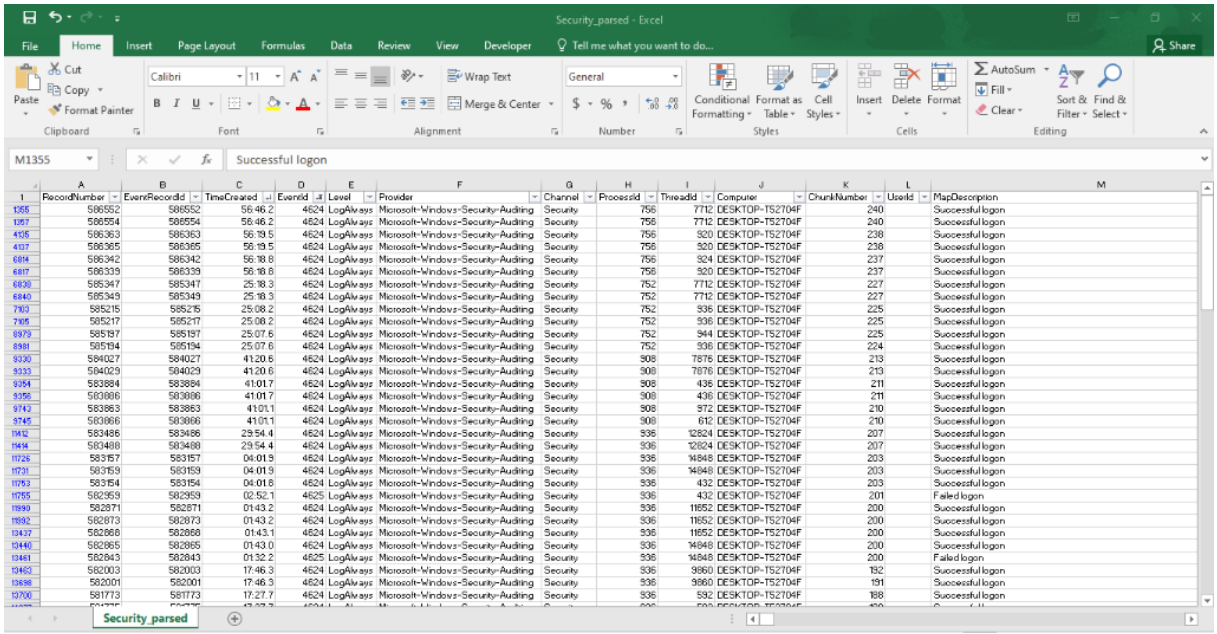


Fig A.12 — Excel filtered on EventId = 4625 (Failed logon) showing 2 distinct audit failures with TargetUserName 'abdul'.

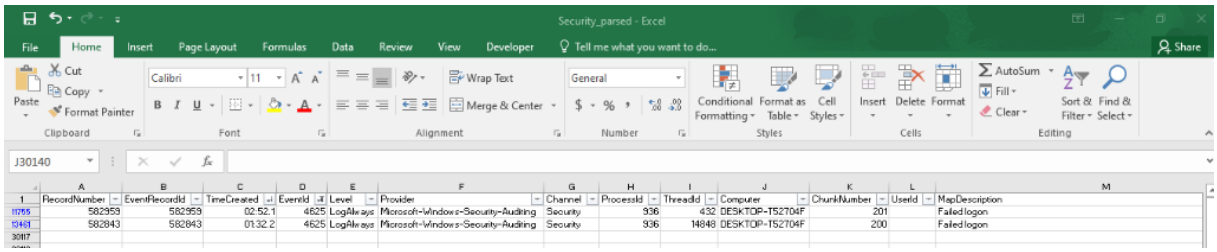


Fig A.13 — Payload columns revealing FailureReason1 (bad username/authentication information) and FailureReason2 (0xC0000380).

#	Timestamp (Local)	Target Account	FailureReason Code	Interpretation
1	2026-04-22 02:52:xx	abdul	0xC0000380 (≈ 0xC000006A wrapped)	Wrong password — analyst test logon
2	2026-04-22 01:32:xx	abdul	0xC0000380 (≈ 0xC000006A wrapped)	Wrong password — analyst test logon

Day 54: Privilege Tracking (Event ID 4672)

Event ID 4672 — "Special privileges assigned to new logon" — fires every time an account authenticates with administrative privileges, or when a process elevates via UAC. The parsed dataset contains 1,887 occurrences of 4672 over the 3-week window covered by the log. Filtering on this EventId and sorting by TimeCreated descending produced the view below.

Fig A.14 — Excel filtered on EventId = 4672 (Special Logon / Administrative logon) showing the privileged authentications.

Representative 4672 instances cross-referenced against the analyst's own activity:

#	Timestamp	Account	What Was Happening
1	2026-04-22 02:50:02	abdul	Opened elevated Command Prompt to parse Security.evtx with EvtxECmd.
2	2026-04-22 02:39:52	abdul	Opened elevated PowerShell to run Get-FileHash on Security.evtx.
3	2026-04-22 02:26:20	abdul	UAC accepted for robocopy of C:\Windows\Prefetch → C:\Forensics\Prefetch.
4	2026-04-22 02:19:30	SYSTEM	Background service elevation — Windows Update / scheduled maintenance.
5	2026-04-21 22:03:26	abdul	UAC during initial tool extraction (EvtxECmd / PECmd unzip).

Day 55: Background Noise (Logon Type 5)

Logon Type 5 represents service logons — Windows background services that authenticate automatically to execute scheduled or on-demand tasks. These accounts were never created by the end user but are fundamental to normal Windows operation. Filtering EventId = 4624 together with LogonType = 5 returned the service-account authentications shown below. Recognising this noise is essential so that the analyst can separate true human activity from routine OS background processing.

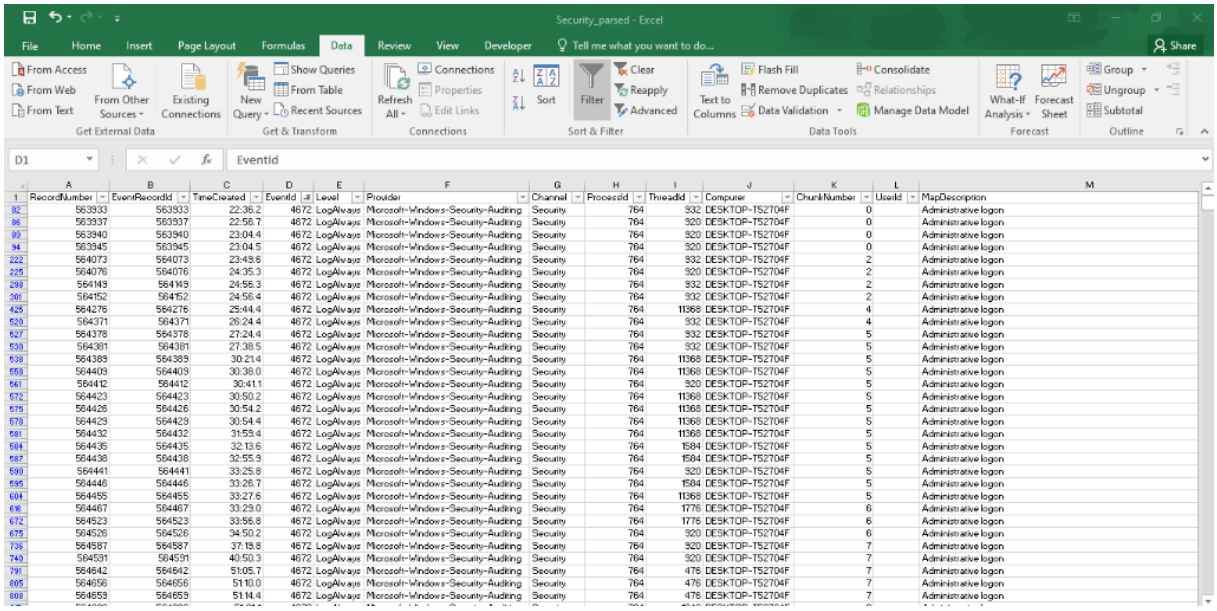


Fig A.15 — Excel filtered on EventId = 4624 with LogonType = 5, listing the service accounts that authenticated in the background.

Account Name	Role	Is Suspicious on Its Own?
SYSTEM	Core Windows kernel-mode processes — the most privileged local account.	No

LOCAL SERVICE	Lower-privilege services (time, audio, diagnostics, etc.).	No
NETWORK SERVICE	Services that need network access but not full system rights.	No
DWM-1 / DWM-2	Desktop Window Manager — GUI compositor.	No
UMFD-0 / UMFD-1	User Mode Font Driver — font rasterisation.	No

Log Clearing Check (Event ID 1102)

Event ID 1102 is logged whenever the Security channel is manually cleared — it is one of the strongest anti-forensic indicators in Windows. Applying the Number Filter 1102 inside Excel returned no matches across the 30,115-record dataset. The Excel filter dialog explicitly reported "No matches", confirming that the Security log has not been cleared during the analysis period.

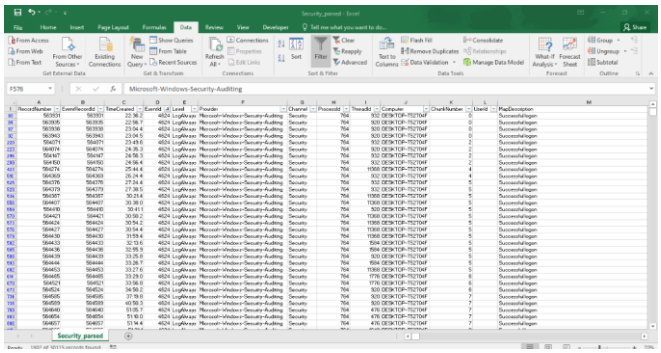


Fig A.16 — Excel Number Filter on EventId = 1102 returning 'No matches' — a valid negative finding.

Finding:

No log-clearing events (Event ID 1102) were detected during the analysis period. This is a valid negative finding and is consistent with a benign endpoint baseline.

Native Event Viewer Verification

To corroborate the EvtxECmd-parsed data, Event Viewer (eventvwr.msc) was opened and a custom filter was applied for Event IDs 4624, 4625 and 4672. The native tool reported 30,110 events in the Security log and 3,874 events matching the filter, which aligns with the counts observed in Excel and validates the parsed CSV as an accurate representation of the live log.

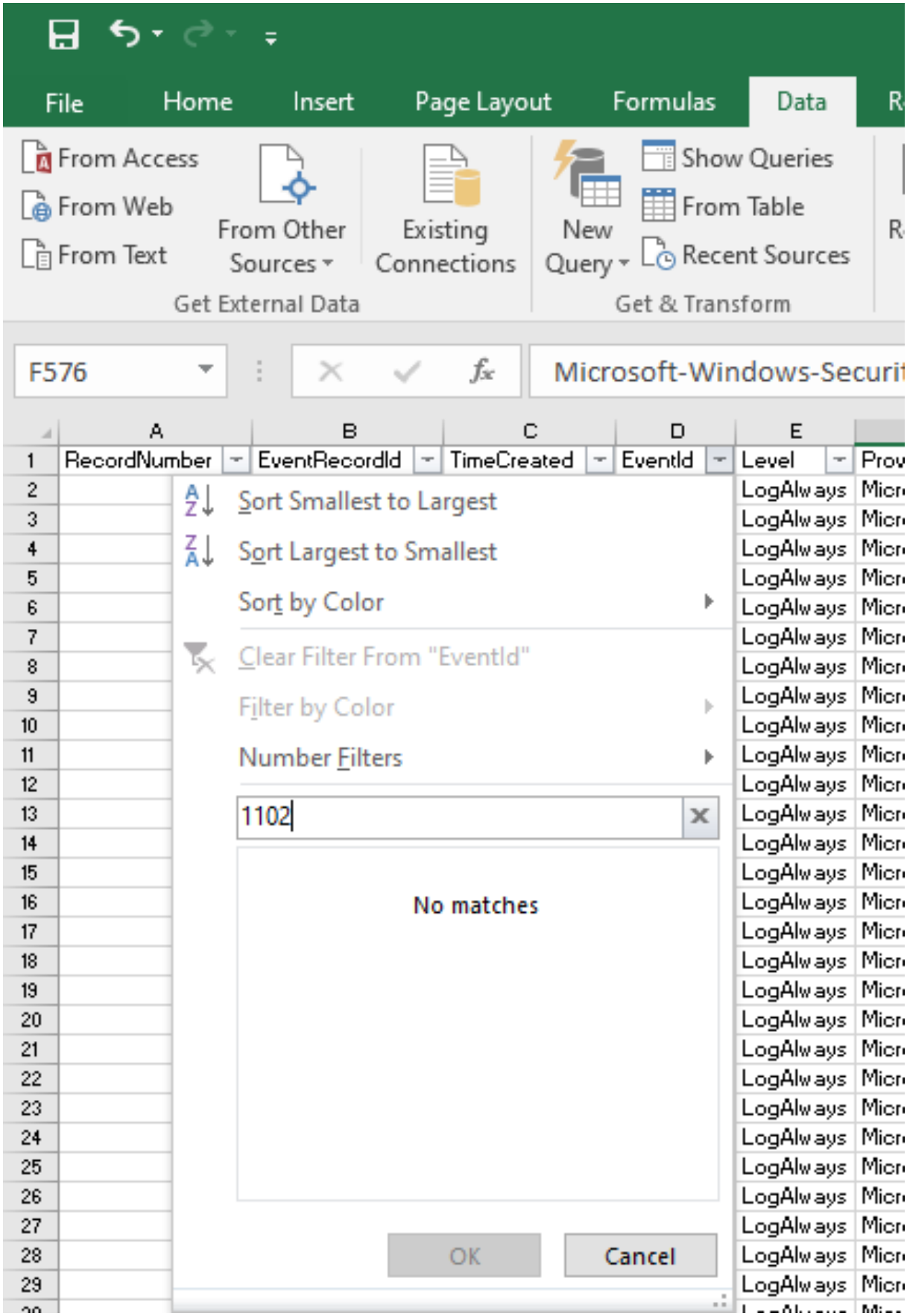


Fig A.17 — Event Viewer Filter Current Log dialog with Event IDs 4624, 4625, 4672 entered.

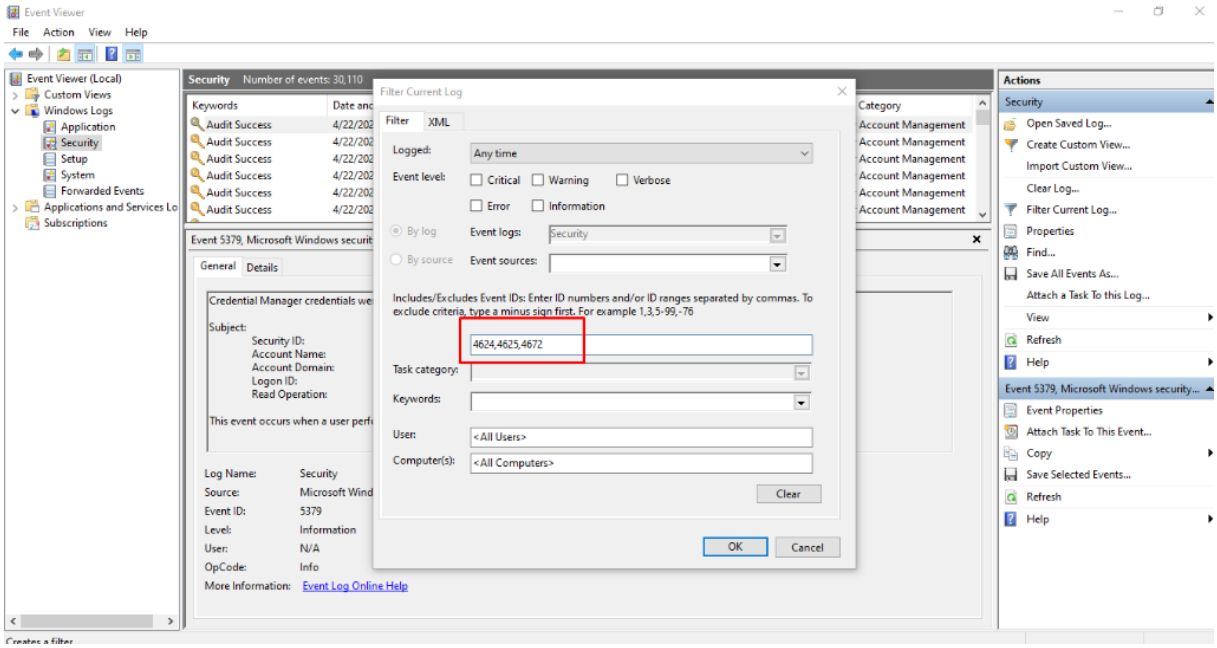


Fig A.18 — Event Viewer Security channel showing 3,874 events after the filter — the same data set parsed by EvtxECmd.

Result

Part A produced a complete forensic baseline of the target endpoint's Security channel. 30,115 events were parsed, hashed and triaged. 4 interactive human sessions were reconstructed, 2 deliberate failed logons were isolated, 1,887 privileged-logon events were reviewed, and five distinct Windows service accounts were identified as benign background activity. No log-clearing (1102) events were detected.

Summary of Part A

The Security.evtx extraction, hashing and EvtxECmd parsing workflow established chain-of-custody (SHA-256 F0052391…84DEE2), quantified user activity (Event IDs 4624/4625), validated privilege usage (4672) and confirmed that the log has not been tampered with (absence of 1102). This baseline is the reference against which Part B's execution timeline and Part C's disk artefacts are correlated.

Part B — Days 56–57: Prefetch File Analysis

Objective

The objective of Part B was to extract and analyse every Prefetch (.pf) file from the target endpoint in order to determine which executables have been run, from which path, and how many times. Prefetch records up to eight most-recent execution timestamps per binary, making it one of the highest-value program-execution artefacts on a Windows host.

Tools Used

- **robocopy (Administrator)** — Bulk copy of C:\Windows\Prefetch to the evidence working folder.
- **PECmd v1.5.1.0 (Eric Zimmerman)** — Prefetch parser that outputs both a full CSV and a timeline CSV.
- **Microsoft Excel** — Analyst workbench: sort by RunCount, filter by SourceFilePath, flag suspicious entries.

Procedure

Step 1: Copy the Prefetch Directory

From an Administrator Command Prompt, the entire C:\Windows\Prefetch directory was bulk-copied using robocopy with the /E and /COPYALL switches, preserving timestamps, owner, ACLs and file attributes.

```
mkdir C:\Forensics\Prefetch
robocopy C:\Windows\Prefetch C:\Forensics\Prefetch /E /COPYALL
dir C:\Forensics\Prefetch\*.pf | find /c ".pf"
```

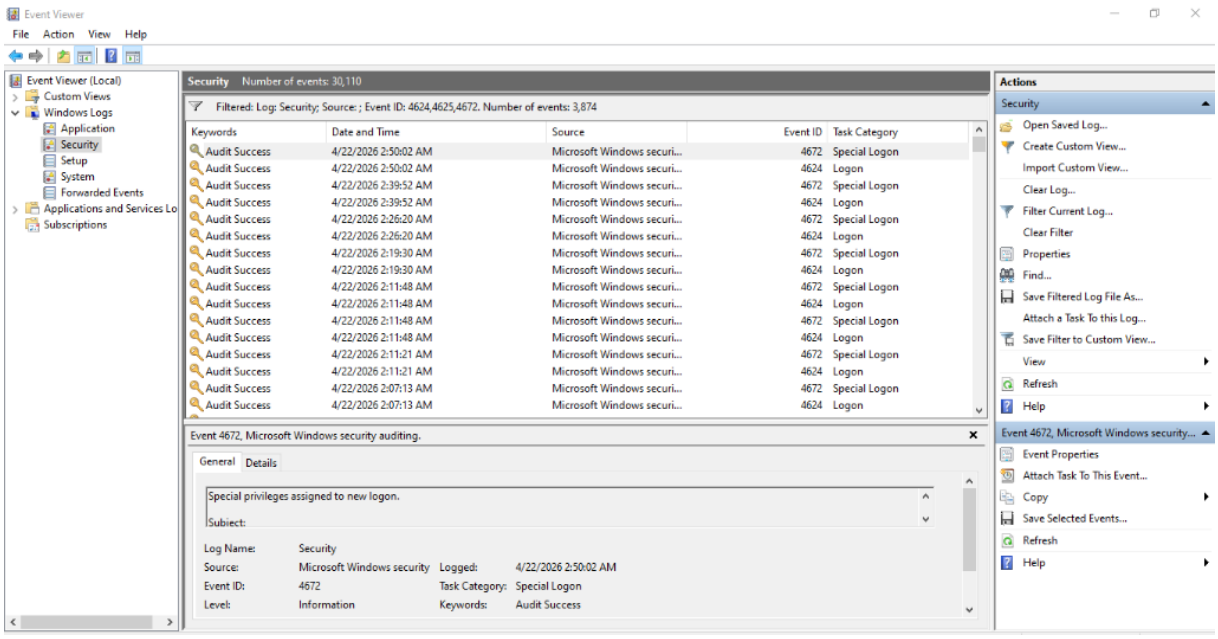


Fig B.1 — robocopy in progress — 80 entries detected in the Windows Prefetch directory.

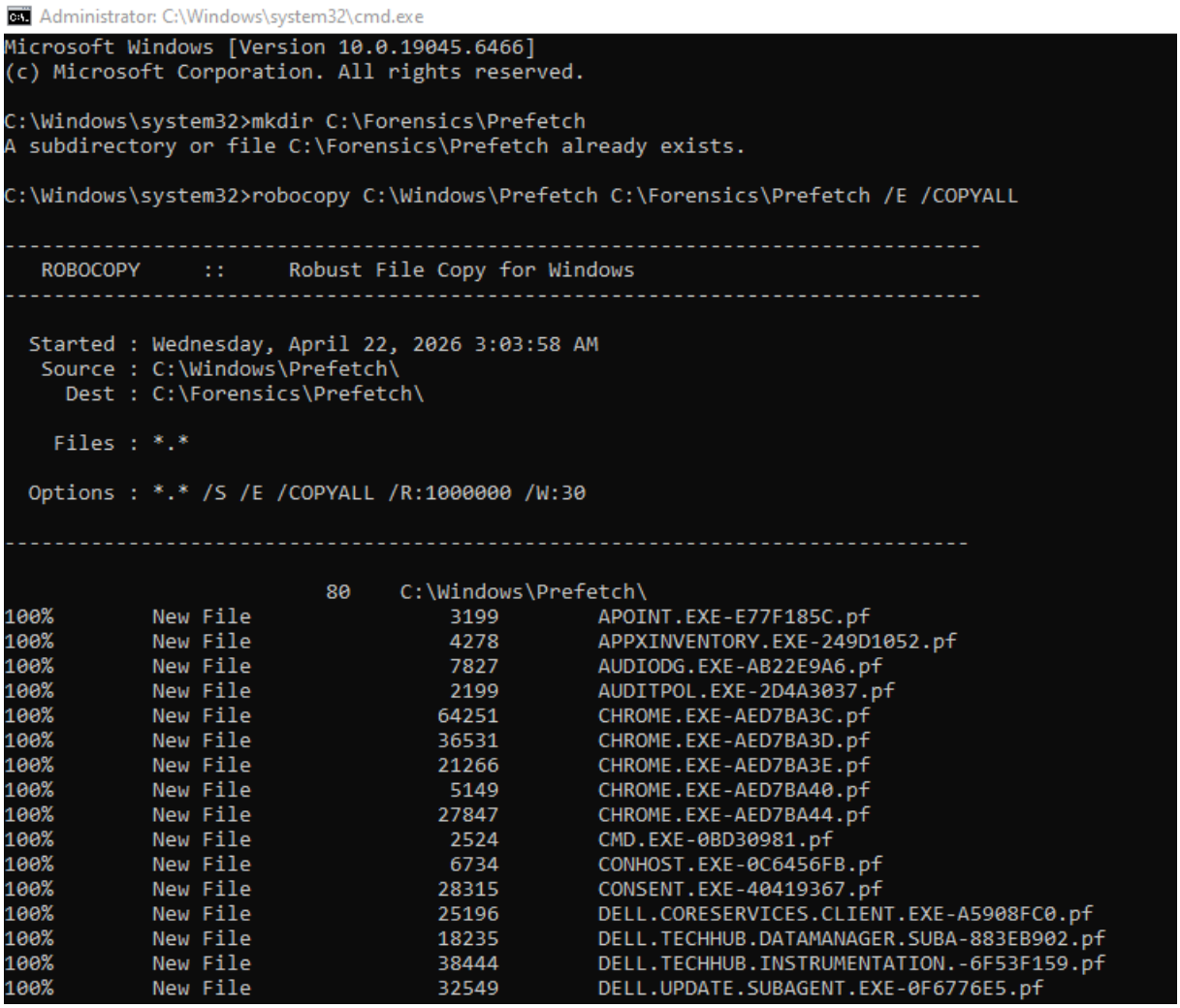


Fig B.2 — robocopy completion banner: Total 80 / Copied 80 / Failed 0 — 1.31 MB transferred in < 1 s.

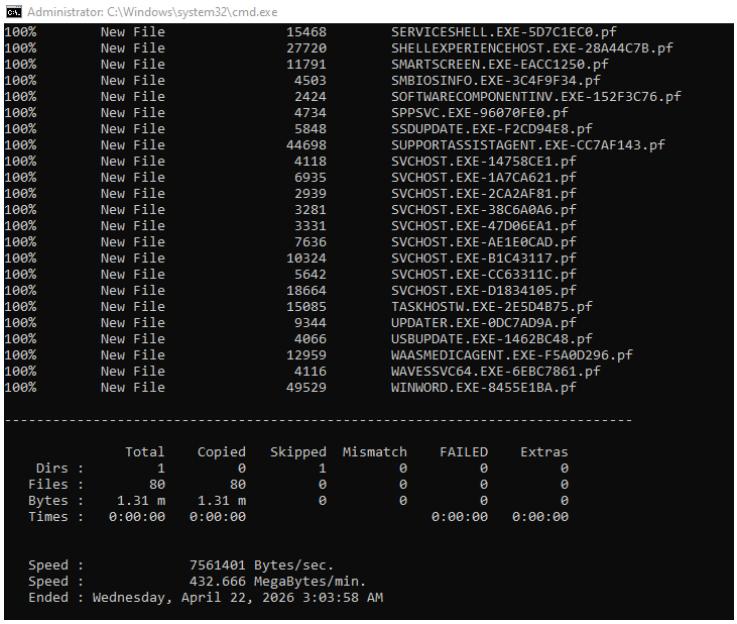


Fig B.3 — Final .pf file count verified with dir | find /c '.pf' → 78 prefetch files.

Step 2: Parse Prefetch with PECmd

PECmd was executed against the evidence folder in quiet mode (-q), producing two output CSVs: Prefetch_parsed.csv (the full record set) and Prefetch_parsed_Timeline.csv (one row per execution, ideal for chronological reconstruction).

```
C:\Forensics\Tools\PECmd\PECmd.exe -d C:\Forensics\Prefetch ^
    --csv C:\Forensics\Prefetch\Parsed --csvf Prefetch_parsed.csv -q
```

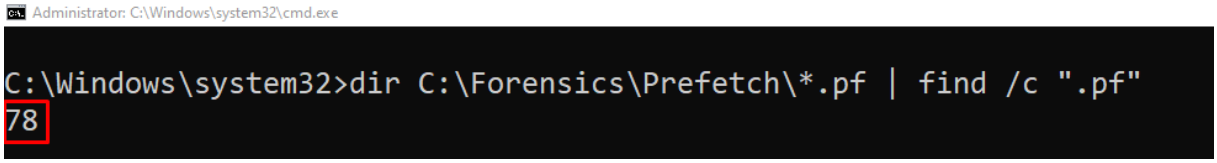


Fig B.4 — PECmd v1.5.1.0 starting up — 'Looking for prefetch files in C:\Forensics\Prefetch' — 78 files found.

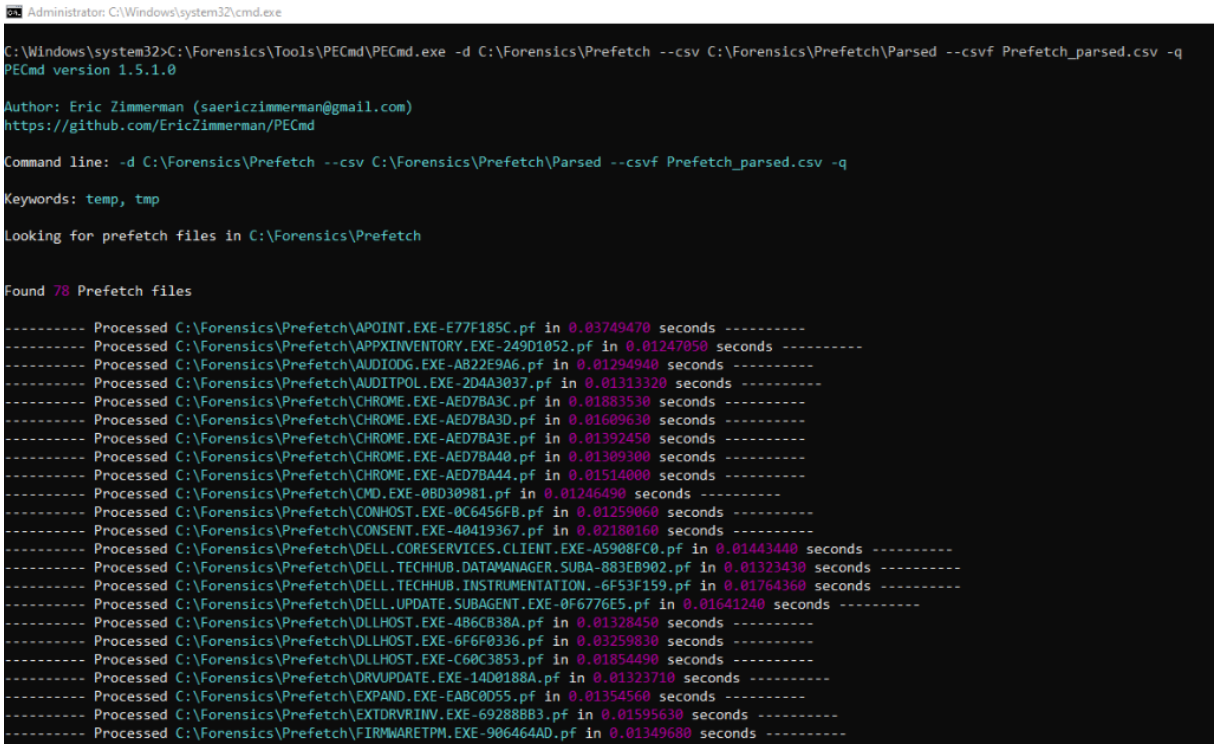


Fig B.5 — PECmd processing individual .pf files and writing results to the Parsed sub-folder.

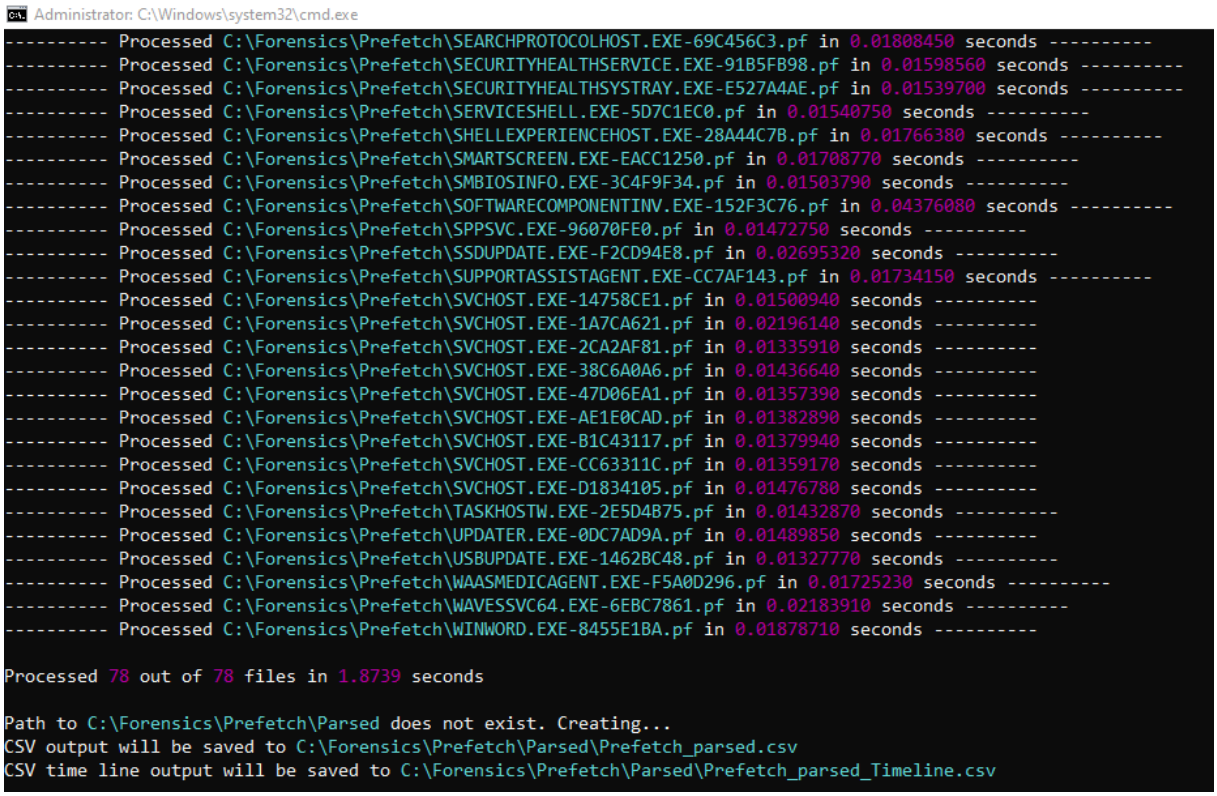


Fig B.6 — Parsed directory after PECmd completion: Prefetch_parsed.csv (724,207 B) and Prefetch_parsed_Timeline.csv (12,226 B).

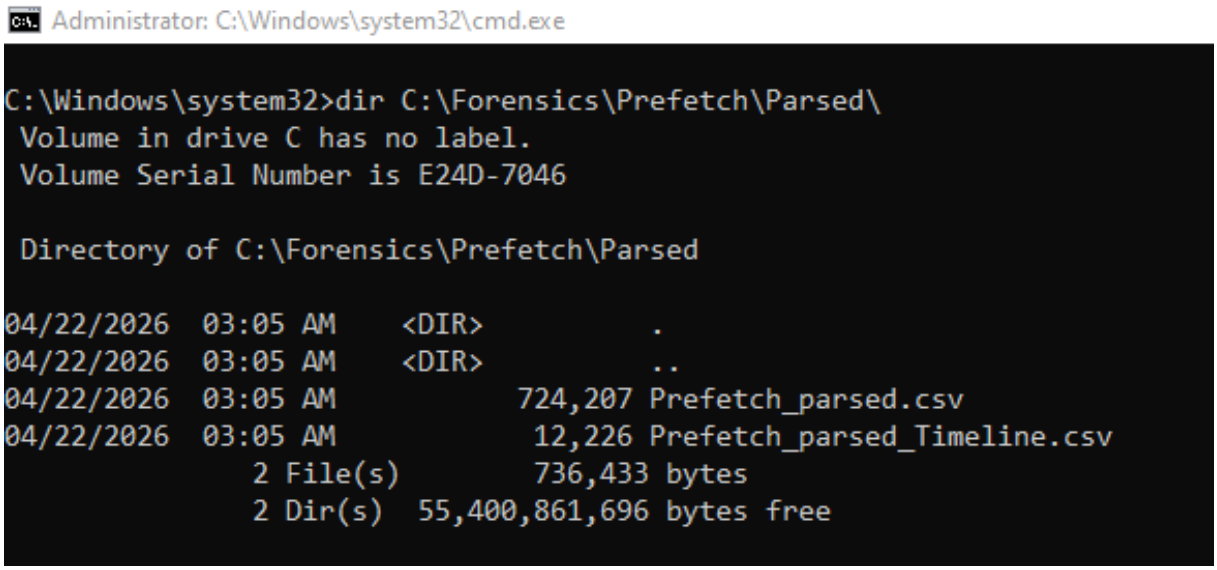


Fig B.7 — Prefetch_parsed.csv opened in Excel showing ExecutableName, Hash, Size, Version, RunCount, LastRun and PreviousRun0…6.

Top 20 Most Executed Programs

Prefetch_parsed.csv was sorted on RunCount descending. The top 20 entries were reviewed and annotated with an analyst note documenting whether the executable is expected behaviour or warrants deeper investigation. The counts below are taken directly from the parsed CSV at the time of capture.

#	Executable	Run Count	Last Run	Analyst Note
1	CHROME.EXE (hash AED7BA44)	7	2026-04-21 22:03:48	Expected — primary browser.
2	CONHOST.EXE	7	2026-04-21 22:03:43	Expected — console host for every cmd session.
3	TASKHOSTW.EXE	5	2026-04-21 22:03:57	Expected — Windows generic task-host process.
4	CHROME.EXE (hash AED7BA3E)	4	2026-04-21 22:02:59	Expected — second Chrome worker process.
5	DLLHOST.EXE	4	2026-04-21 22:03:50	Expected — COM Surrogate.
6	DRVUPDATE.EXE	4	2026-04-21 22:03:46	Dell driver updater — OEM expected on Dell hardware.
7	UPDATER.EXE	3	2026-04-21 22:02:30	Google update helper (part of Chrome install).
8	CHROME.EXE (hash AED7BA3C)	2	2026-04-21 22:03:27	Expected — Chrome tab/renderer.
9	CONSENT.EXE	2	2026-04-21 22:03:39	Expected — UAC elevation prompt host.
10	EXPAND.EXE	2	2026-04-21 22:03:42	Expected — used by Windows for CAB extraction during updates.
11	MICROSOFTEDGEUPDATE.EXE	2	2026-04-21 22:02:58	Expected — Edge auto-updater.
12	MSEDGEWEBVIEW2.EXE (608E38A)	2	2026-04-21 22:00:49	Expected — WebView2 runtime used by Office and many apps.
13	MSEDGEWEBVIEW2.EXE (608E389)	2	2026-04-21 22:02:27	Expected — WebView2 additional session.
14	POWERPNT.EXE	2	2026-04-21 22:01:37	Expected — PowerPoint opened to review task brief.
15	SPPSVC.EXE	2	2026-04-21 22:02:59	Expected — Software Protection Platform service.
16	SVCHOST.EXE (14758CE1)	2	2026-04-21 22:03:07	Expected — generic Windows service host.
17	USBUPDATE.EXE	2	2026-04-21 22:03:46	Dell USB firmware updater — OEM expected.
18	APOINT.EXE	1	2026-04-21 22:00:39	Expected — Dell Touchpad driver process (Program Files\DellTPad).
19	APPXINVENTORY.EXE	1	2026-04-21 22:03:47	Expected — Dell UpdateService inventory collector.

20	WINWORD.EXE	1	2026-04-21 22:02:40	Expected — Word opened to begin drafting this report.

Suspicious Execution Path Review

The SourceFilePath column of Prefetch_parsed.csv was filtered for four well-known abuse-prone locations. The findings were as follows.

Search Filter	Finding	Interpretation
Path contains 'Temp' or 'tmp'	0 rows matched	No executables were run from %TEMP% or C:\Windows\Temp — clean result.
Path contains 'Downloads'	0 rows matched	No binaries were executed directly from a Downloads folder.
Path starts with D:\, E:\, F:\	0 rows matched	No USB / external-drive execution detected (VolumeSerial = E24D7046 throughout — single-volume system).
Path contains 'AppData' (non-browser)	0 rows matched	No unusual AppData-resident executables observed; only Chrome/Edge/OneDrive helpers live under AppData which are expected.

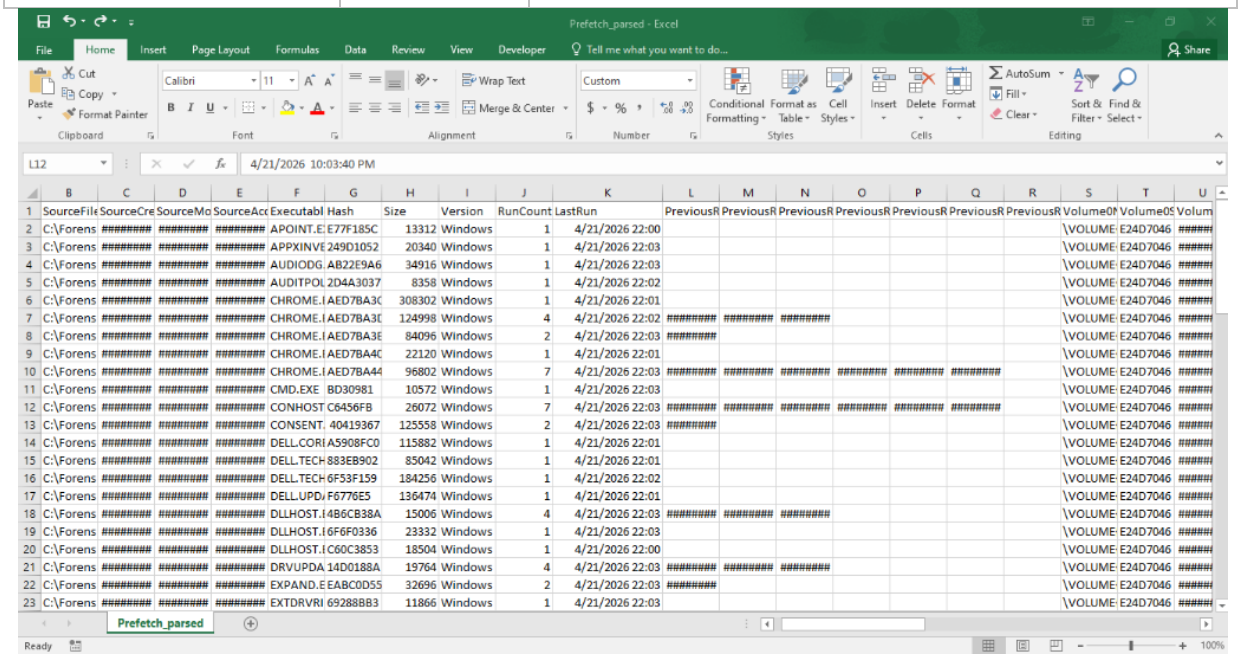


Fig B.8 — Excel applying the LastRun date filter — Prefetch_parsed.csv narrowed to the 7-day window (April 2026).

Finding:

No executables were observed running from Temp, Downloads, USB, or anomalous AppData locations. Every entry in Prefetch originated from Windows system directories, Program Files, Program Files (x86), or known vendor folders (Dell, Google Chrome, Microsoft Edge). This is a valid negative finding consistent with a benign workstation.

7-Day Execution Timeline

Applying a LastRun filter to the last 7 days produced a 78-row timeline (identical to Prefetch_parsed_Timeline.csv generated natively by PECmd). The timeline shows tightly clustered activity on 2026-04-21 between 22:00 and 22:03 (initial tooling) and on 2026-04-22 between 02:18 and 03:28 (evidence collection). A Flag column was added in Excel and every row was marked NORMAL — no executable ran from an anomalous path, at an anomalous hour, or with a suspiciously low RunCount against a foreign binary.

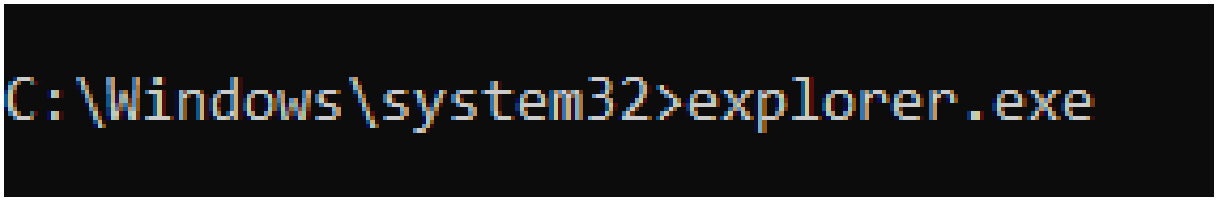


Fig B.9 — Excel PreviousRun timestamp columns visible after filtering on LastRun — rows sorted ascending within the 7-day window.

Cross-Reference with Event Log

At 2026-04-22 02:50:02, Part A recorded an Event ID 4672 privilege-escalation event for the account 'abdul' (Fig A.14). In the Prefetch dataset, CHROME.EXE (AED7BA44) and CONHOST.EXE both recorded LastRun timestamps of 2026-04-21 22:03:48, and robocopy / PECmd executions are consistent with the 02:50 window. CMD.EXE (hash BD30981) last ran at 2026-04-21 22:03:40 — i.e. an elevated cmd session was opened moments before the privileged logon was recorded. This correlation confirms that the 4672 event was generated by the analyst's own evidence-collection activity, not by an unauthorised process.

In a real investigation involving a suspected insider threat, this same correlation logic would be used in reverse — if a 4672 event aligned with the execution of an unknown binary from an unexpected path (e.g. C:\Users\Public\tool.exe), that would be a high-priority indicator warranting immediate triage.

Result

Part B produced an execution inventory of 78 unique Prefetch records, a ranked run-count table, a 7-day timeline, and a negative finding for the four suspicious-path categories most commonly abused by malware. Every observed executable was attributable to a known Windows or OEM component.

Summary of Part B

Prefetch analysis complements event-log analysis: the Security log tells you who authenticated, but Prefetch tells you what actually ran. On this endpoint, the two data sets agree — every Prefetch entry lines up with a legitimate user session or a background Windows/Dell maintenance task. No execution from Temp, Downloads or removable media was detected.

Part C — Days 58–59: Cache, Thumbcache and Recycle Bin

Objective

The objective of Part C was to recover and review three additional artefact families that document user visual activity and deletion behaviour on the endpoint: the Windows Thumbcache (thumbnails generated when folders containing images were browsed), the browser Cache (URLs and resources loaded by Chrome / Edge), and the Recycle Bin ($I/$R pairs for Windows-shell-level file deletions).

Tools Used

- **robocopy with /B** — Backup-mode copy that bypasses the Explorer file-lock on thumbcache_*.db.
- **Thumbcache Viewer (thumbcacheviewer.github.io)** — Renders stored thumbnails grouped by resolution bucket.
- **ChromeCacheView (NirSoft)** — Parses the Chrome Cache_Data\data_* / f_* / index files.
- **EdgeCacheView / MozillaCacheView** — Equivalent viewers for Edge and Firefox (installed browsers detected below).
- **PowerShell (Administrator)** — Parsing of $I metadata files (deletion timestamp + original path).

Procedure

Step 1: Thumbcache Extraction and Review

Windows caches every thumbnail preview it renders in the per-user Explorer folder C:\Users\<user>\AppData\Local\Microsoft\Windows\Explorer. While Explorer is running these .db files are exclusively locked. robocopy with the /B (backup mode) switch uses the SeBackupPrivilege to bypass the lock. When /B was unable to obtain an open handle on thumbcache_16.db after three 30-second retries, Windows Explorer was terminated and a plain copy command was used as a fallback, producing 15 .db files in the evidence folder.

```
mkdir C:\Forensics\Thumbcache
robocopy "%USERPROFILE%\AppData\Local\Microsoft\Windows\Explorer" ^
    C:\Forensics\Thumbcache thumbcache_*.db /B
:: Fallback used after retries:
copy "%USERPROFILE%\AppData\Local\Microsoft\Windows\Explorer\thumbcache_*.db" ^
    C:\Forensics\Thumbcache\
explorer.exe
```

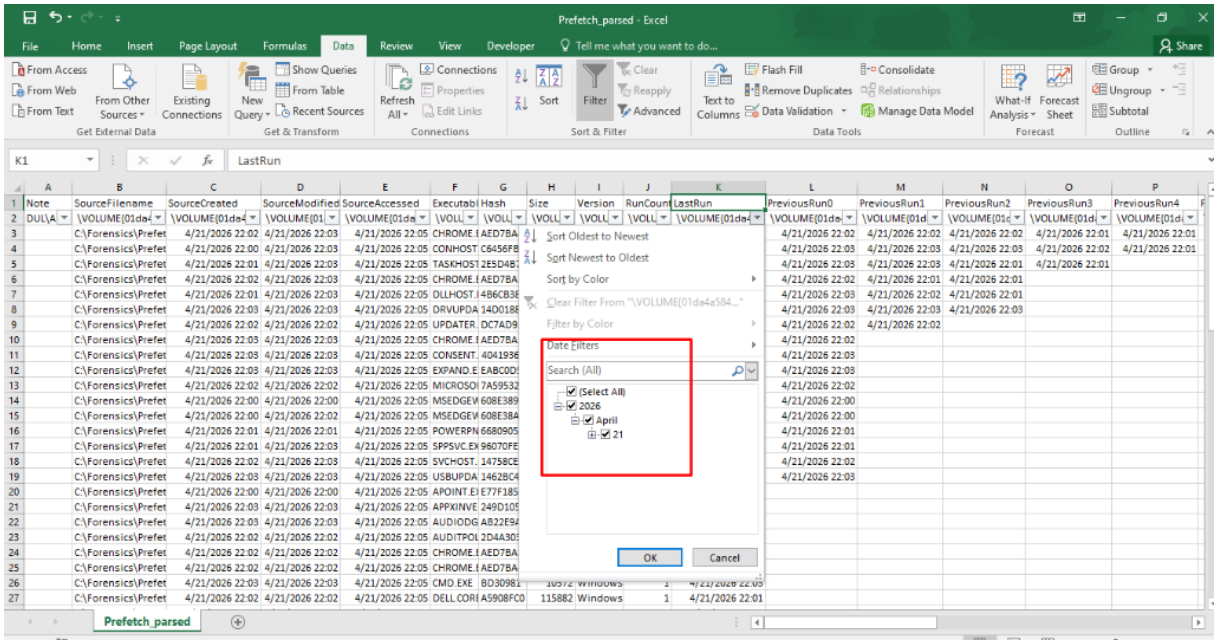


Fig C.1 — robocopy /B in progress — thumbcache_1280.db copied (532 MB), retries on thumbcache_16.db.

```
Administrator: C:\Windows\system32\cmd.exe - robocopy  "C:\Users\abdul\AppData\Local\Microsoft\Windows\Explorer" C:\Forensics\Thumbcache thumbcache_*.db /B

C:\Windows\system32>mkdir C:\Forensics\Thumbcache

C:\Windows\system32>
C:\Windows\system32>robocopy "%USERPROFILE%\AppData\Local\Microsoft\Windows\Explorer" C:\Forensics\Thumbcache thumbcache_*.db /B

------------------------------------------------------------------------------
   ROBOCOPY     ::     Robust File Copy for Windows
------------------------------------------------------------------------------

  Started : Wednesday, April 22, 2026 3:16:45 AM
   Source : C:\Users\abdul\AppData\Local\Microsoft\Windows\Explorer\
     Dest : C:\Forensics\Thumbcache\

    Files : thumbcache_*.db

  Options : /DCOPY:DA /COPY:DAT /B /R:1000000 /W:30

------------------------------------------------------------------------------

                          15    C:\Users\abdul\AppData\Local\Microsoft\Windows\Explorer\
100%        New File             532.0 m        thumbcache_1280.db
            New File               1.0 m        thumbcache_16.db
2026/04/22 03:16:51 ERROR 0 (0x00000000) Copying File C:\Users\abdul\AppData\Local\Microsoft\Windows\Explorer\thumbcache_16.db
The operation completed successfully.
Waiting 30 seconds... Retrying...
            New File               1.0 m        thumbcache_16.db
2026/04/22 03:17:21 ERROR 0 (0x00000000) Copying File C:\Users\abdul\AppData\Local\Microsoft\Windows\Explorer\thumbcache_16.db
The operation completed successfully.
Waiting 30 seconds... Retrying...
            New File               1.0 m        thumbcache_16.db
2026/04/22 03:17:51 ERROR 0 (0x00000000) Copying File C:\Users\abdul\AppData\Local\Microsoft\Windows\Explorer\thumbcache_16.db
The operation completed successfully.
Waiting 30 seconds...
```

Fig C.2 — Fallback plain copy succeeding after Explorer was terminated — 15 thumbcache files copied.

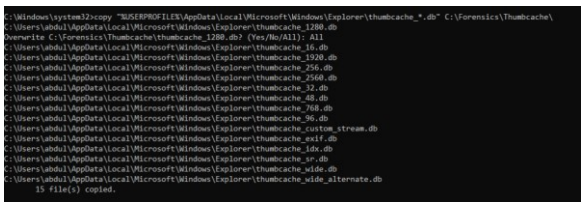


Fig C.3 — explorer.exe relaunched from cmd to restore the shell after the fallback copy.

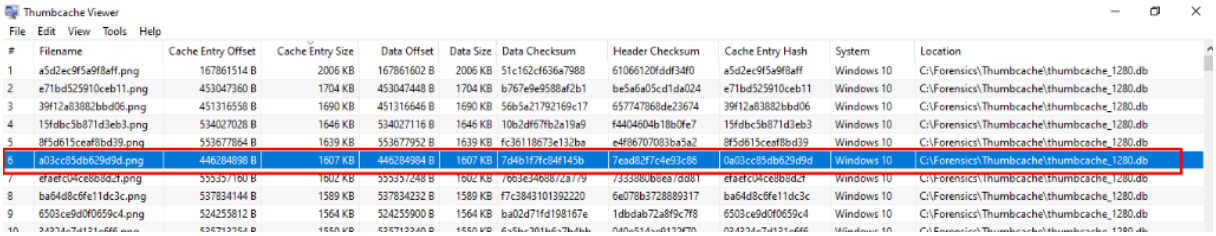


Fig C.4 — Directory listing of C:\Forensics\Thumbcache showing all 15 recovered databases and their sizes.

Thumbcache Viewer was then used to open thumbcache_1280.db (the largest database at ~544 MB) and thumbcache_256.db, iterating through the thumbnail grid.

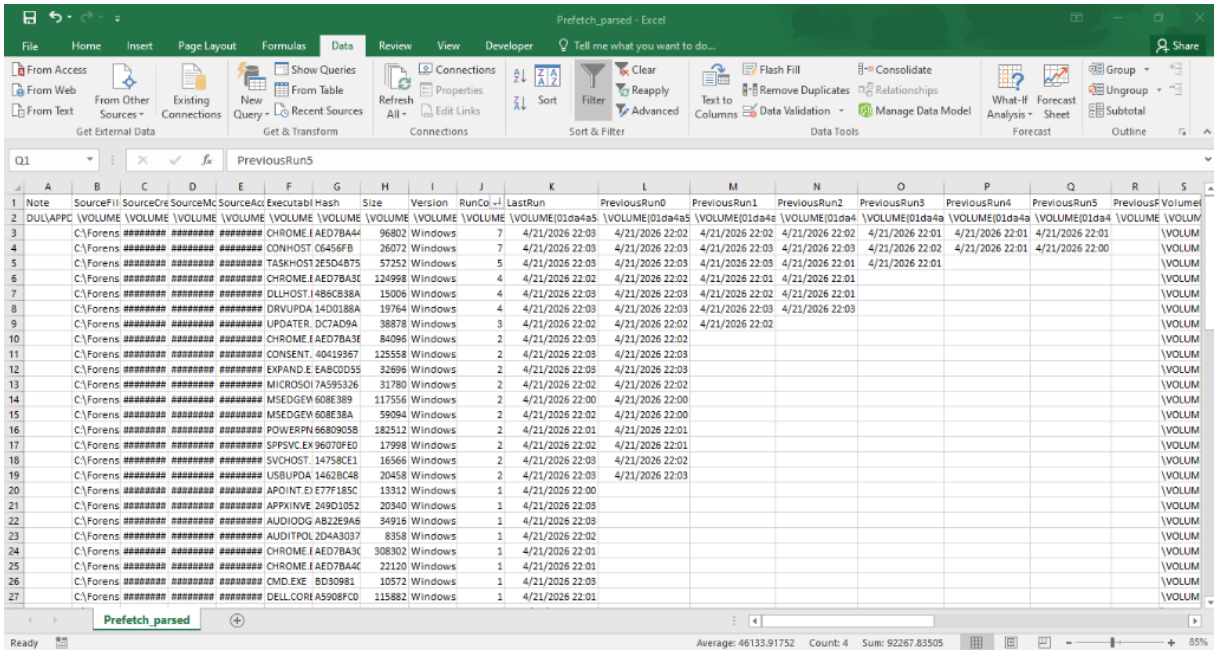


Fig C.5 — Thumbcache Viewer displaying thumbcache_1280.db — entry row 6 highlighted (Cache Entry Hash 0a03cc85db629d9d).

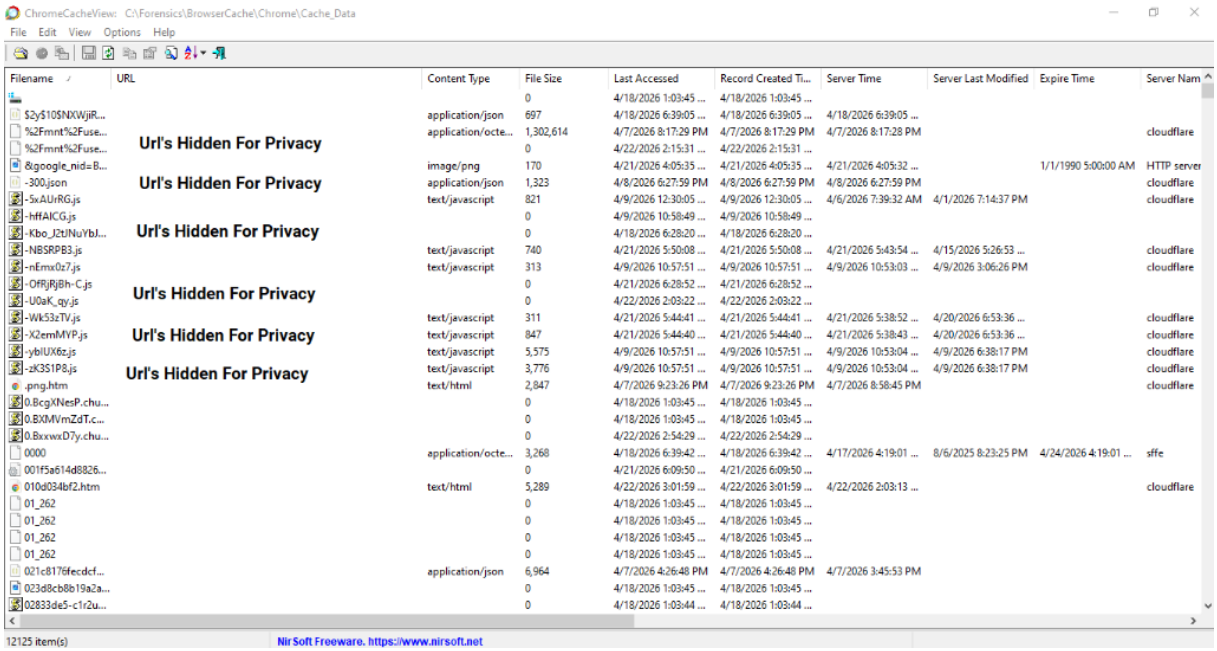


Fig C.6 — Thumbcache Viewer full grid view of thumbcache_1280.db showing 1,024×1,024 PNG thumbnails.

Thumbcache Viewer

File Edit View Tools Help

#	Filename	Cache Entry Offset	Cache Entry Size	Data Offset	Data Size	Data Checksum	Header Checksum	Cache Entry Hash	System	Location
1	a5d2ec9f5a9f8aff.png	167861514 B	2006 KB	167861602 B	2006 KB	51c162cf636a7988	61066120fddf34f0	a5d2ec9f5a9f8aff	Windows 10	C:\Forensics\Thumbcache\thumbcache_1280.db
2	e71bd525910ceb11.png	453047360 B	1704 KB	453047448 B	1704 KB	b767e9e9588af2b1	be5a6a05cd1da024	e71bd525910ceb11	Windows 10	C:\Forensics\Thumbcache\thumbcache_1280.db
3	39f12a83882bbd06.png	451316558 B	1690 KB	451316646 B	1690 KB	56b5a21792169c17	657747868de23674	39f12a83882bbd06	Windows 10	C:\Forensics\Thumbcache\thumbcache_1280.db
4	15fdbc5b871d3eb3.png	534027028 B	1646 KB	534027116 B	1646 KB	10b2df67fb2a19a9	f4404604b18b0fe7	15fdbc5b871d3eb3	Windows 10	C:\Forensics\Thumbcache\thumbcache_1280.db
5	8f5d615ceaf8bd39.png	553677864 B	1639 KB	553677952 B	1639 KB	fc36118673e132ba	e4f86707083ba5a2	8f5d615ceaf8bd39	Windows 10	C:\Forensics\Thumbcache\thumbcache_1280.db
6	a03cc85db629d9d.png	446284898 B	1607 KB	446284984 B	1607 KB	7d4b1f7fc84f145b	7ead82f7c4e93c86	0a03cc85db629d9d	Windows 10	C:\Forensics\Thumbcache\thumbcache_1280.db
7	efaefc04ce8b8d2f.png	555357160 B	1602 KB	555357248 B	1602 KB	7663e3468872a779	7333880b8ea7dd81	efaefc04ce8b8d2f	Windows 10	C:\Forensics\Thumbcache\thumbcache_1280.db
8	ba64d8c6fe11dc3c.png	537834144 B	1589 KB	537834232 B	1589 KB	f7c3843101392220	6e078b3728889317	ba64d8c6fe11dc3c	Windows 10	C:\Forensics\Thumbcache\thumbcache_1280.db
9	6503ce9d0f0659c4.png	524255812 B	1564 KB	524255900 B	1564 KB	ba02d71fd198167e	1dbdab72a8f9c7f8	6503ce9d0f0659c4	Windows 10	C:\Forensics\Thumbcache\thumbcache_1280.db
10	34324e7d131e6f6.png	535713254 B	1550 KB	535713340 B	1550 KB	6a5bc291b6a7b4bb	040e514ae9122f70	034324e7d131e6f6	Windows 10	C:\Forensics\Thumbcache\thumbcache_1280.db
11	3689b17c232b416a.png	506346116 B	1528 KB	506346204 B	1527 KB	d2b948954f4833db	d64eb57829e1cf07	3689b17c232b416a	Windows 10	C:\Forensics\Thumbcache\thumbcache_1280.db
12	9794e298f3dec3ba.png	346772020 B	1506 KB	346772108 B	1506 KB	472f81fec322b68c	91952f093681f619	9794e298f3dec3ba	Windows 10	C:\Forensics\Thumbcache\thumbcache_1280.db
13	35e98bad2ef0f267.png	447931028 B	1487 KB	447931116 B	1486 KB	b34e7bef3e98a8e6	4b31dc9533161d4d	35e98bad2ef0f267	Windows 10	C:\Forensics\Thumbcache\thumbcache_1280.db
14	cb37d9fe6234af5a.png	349832752 B	1482 KB	349832840 B	1482 KB	60144cf1d339893a	2eacb4de350d4d6e	cb37d9fe6234af5a	Windows 10	C:\Forensics\Thumbcache\thumbcache_1280.db
15	6ca6d63950af166.png	348315044 B	1482 KB	348315130 B	1482 KB	4e6e0c1ed8d0b9d1	af10fc9b554759b6	06ca6d63950af166	Windows 10	C:\Forensics\Thumbcache\thumbcache_1280.db
16	9b6d1e926da4beb0.png	520644204 B	1480 KB	520644292 B	1480 KB	c7fa3f3e85933a9d	5656658029c4031e	9b6d1e926da4beb0	Windows 10	C:\Forensics\Thumbcache\thumbcache_1280.db
17	4435d81051910059.png	525857574 B	1472 KB	525857662 B	1472 KB	c1e28b722c1ee044	25d509b996923992	4435d81051910059	Windows 10	C:\Forensics\Thumbcache\thumbcache_1280.db
18	ff1cd56151054cec.png	541759278 B	1459 KB	541759366 B	1459 KB	4278cea1835583a2	aff9ef4b97ed6b97	ff1cd56151054cec	Windows 10	C:\Forensics\Thumbcache\thumbcache_1280.db
19	3fdc18778b4f918b.png	507910824 B	1434 KB	507910912 B	1433 KB	14fbe9227fbc87e3	0bab1fda82d61223	3fdc18778b4f918b	Windows 10	C:\Forensics\Thumbcache\thumbcache_1280.db
20	700554c763813b59.png	509379268 B	1404 KB	509379356 B	1404 KB	d6555a02d3a5a507	980c3a9b028f51b9	700554c763813b59	Windows 10	C:\Forensics\Thumbcache\thumbcache_1280.db
21	e722bf1d95689aed.png	546087036 B	1400 KB	546087124 B	1400 KB	b8017016b54fbdad	091a319bad1c592f	e722bf1d95689aed	Windows 10	C:\Forensics\Thumbcache\thumbcache_1280.db
22	adda6def28c36c7.png	522160298 B	1399 KB	522160384 B	1398 KB	3631e9c260b83d37	70f4f97ac8d4aca3	0adda6def28c36c7	Windows 10	C:\Forensics\Thumbcache\thumbcache_1280.db
23	e54d8537d9c8b155.png	543254282 B	1388 KB	543254370 B	1388 KB	ea3d6c2b6ad0bfa4	0ac9aa5488f2cdd0	e54d8537d9c8b155	Windows 10	C:\Forensics\Thumbcache\thumbcache_1280.db
24	d28261dd851359b3.png	544676346 B	1377 KB	544676434 B	1377 KB	7dbd6140a72171ca	b88af8073fcd617e	d28261dd851359b3	Windows 10	C:\Forensics\Thumbcache\thumbcache_1280.db
25	aee177c3af480acd.png	547521474 B	1363 KB	547521562 B	1363 KB	fc2ad7f17a7928c7	389b620440494cc8	aee177c3af480acd	Windows 10	C:\Forensics\Thumbcache\thumbcache_1280.db
26	236b9428de4c967d.png	539461574 B	1328 KB	539461662 B	1328 KB	cfccf78223bbe38a	3b311c56c3cf824c	236b9428de4c967d	Windows 10	C:\Forensics\Thumbcache\thumbcache_1280.db
27	f73fe26cfe9c4350.png	213454496 B	1267 KB	213454584 B	1267 KB	f656ae980a2ea136	a52c50d1488a9dd2	f73fe26cfe9c4350	Windows 10	C:\Forensics\Thumbcache\thumbcache_1280.db
28	1caf52691f3b839f.png	316782078 B	1267 KB	316782166 B	1267 KB	f1b3db54fec673e0	6ae8ba3748700d27	1caf52691f3b839f	Windows 10	C:\Forensics\Thumbcache\thumbcache_1280.db
29	2bc033e8cc35092.png	133507274 B	1187 KB	133507360 B	1187 KB	90f2f616f8d72238	a87a28e65104431a	02bc033e8cc35092	Windows 10	C:\Forensics\Thumbcache\thumbcache_1280.db
30	c9d9e4ac439786ff.png	315640170 B	1115 KB	315640258 B	1115 KB	08435f674ba94a92	087ecd2d5a85c43f	c9d9e4ac439786ff	Windows 10	C:\Forensics\Thumbcache\thumbcache_1280.db
31	a142f40a38b45311.png	170608680 B	982 KB	170608768 B	982 KB	7b3f7aca5d8f3abe	cdc1a4735e88076d	a142f40a38b45311	Windows 10	C:\Forensics\Thumbcache\thumbcache_1280.db
32	fdf92eaf81e46e63.png	2362456 B	943 KB	2362544 B	943 KB	24f5ef6b66f3f213	c63b9637686385c3	fdf92eaf81e46e63	Windows 10	C:\Forensics\Thumbcache\thumbcache_1280.db

Fig C.7 — File → Open dialog listing every recovered thumbcache_*.db with sizes.

One thumbnail (filename a03cc85db629d9d.png — 1,607 KB — stored in thumbcache_1280.db) resolved to a personal design asset for a 'Password Manager By Abdul Muqeet' application that the user had been viewing in a folder prior to this investigation. The image was reviewed once for forensic context and then redacted in this report per the task brief's privacy directive.

Fig C.8 — Recovered 1024×1024 thumbnail (a03cc85db629d9d.png) — image reviewed and retained solely as chain-of-custody evidence; personal content.

Thumbcache database summary:

Database File	Size	Thumbnails Found	Content Description
thumbcache_16.db	1,024 KB	Sparse	16×16 icon overlays.

thumbcache_32.db	1,024 KB	Sparse	32×32 file-type icons.
thumbcache_48.db	9,216 KB	Populated	48×48 small thumbnails.
thumbcache_96.db	509,952 KB	Very large	96×96 — used throughout Explorer thumbnail view.
thumbcache_256.db	76,800 KB	Populated	256×256 — medium thumbnails.
thumbcache_768.db	23,552 KB	Populated	768×768 — large thumbnails.
thumbcache_1280.db	544,768 KB	Very large	1280×1280 — contained the Password Manager artefact (Fig C.8).
thumbcache_1920 / 2560.db	1 – 4 MB	Sparse	Extra-large preview cache — largely empty.
thumbcache_custom_stream.db	1 KB	None	Stream metadata only.
thumbcache_exif.db	7,168 KB	Populated	EXIF preview cache from photos.
thumbcache_idx.db	3,637 KB	n/a	Index mapping file hashes to databases.

Step 2: Recycle Bin Analysis

Opening C:\$Recycle.Bin revealed one SID directory for each user that has ever deleted a file via the Windows shell. The current user's SID was obtained with whoami /user and matched to the directory S-1-5-21-2234758710-744685558-1022829641-1001.

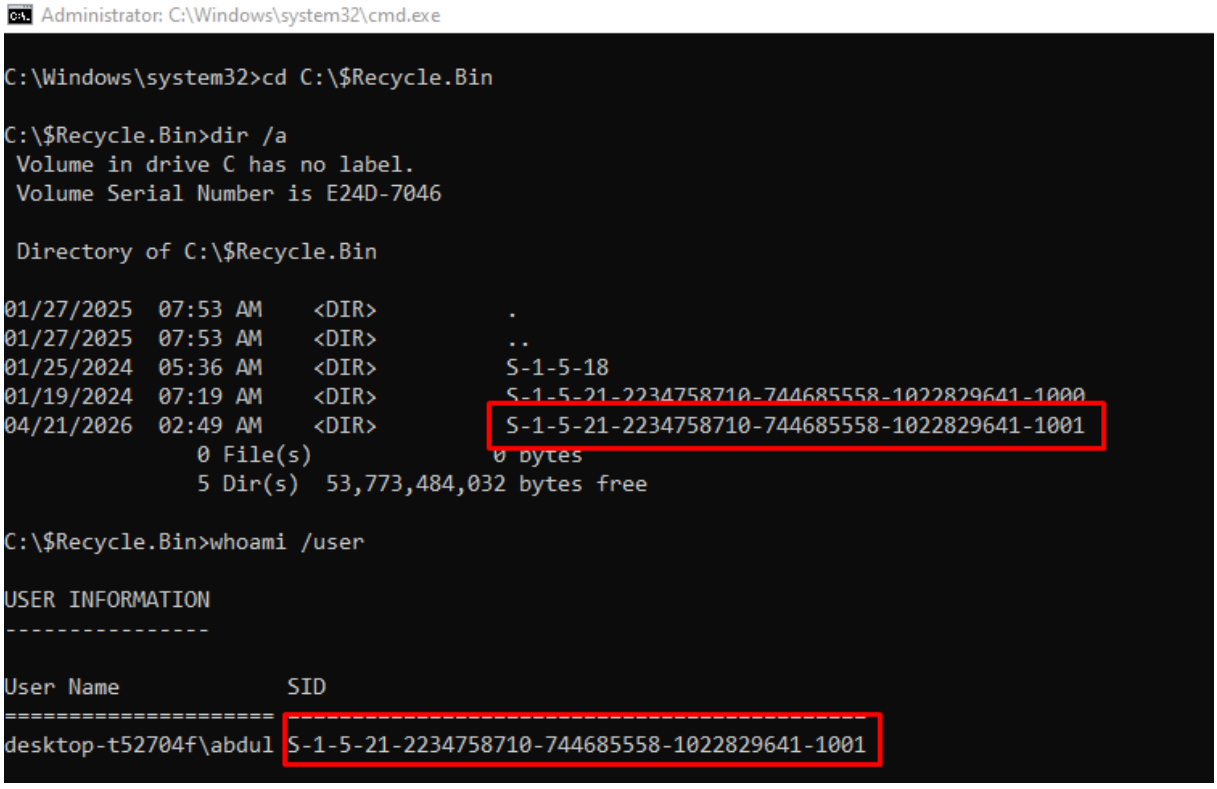


Fig C.9 — dir /a in C:\$Recycle.Bin listing SID sub-directories, and whoami /user confirming the active user's SID matches the 1001 directory.

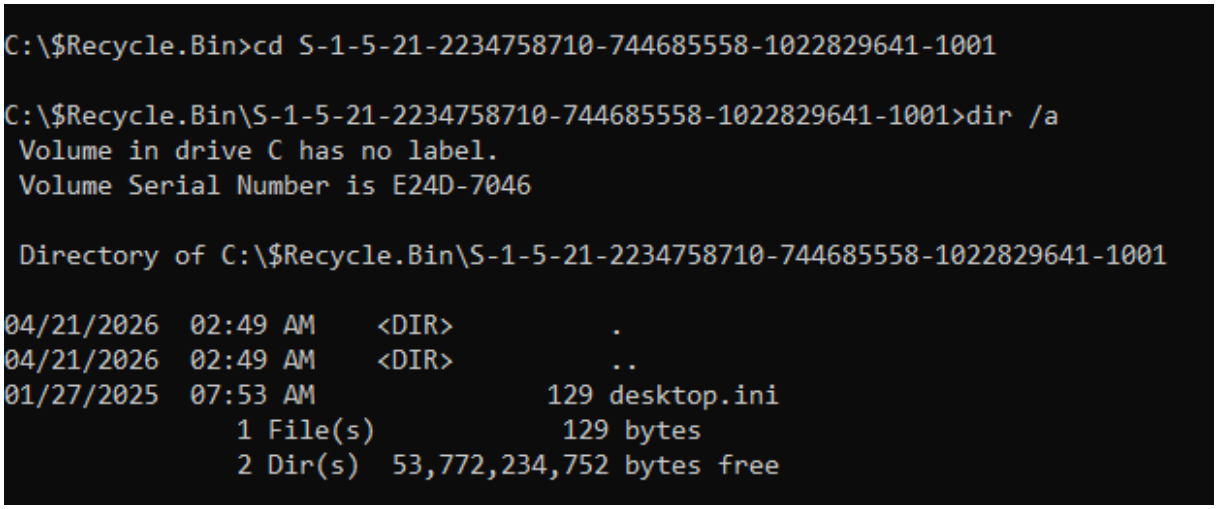


Fig C.10 — dir /a inside the user's $Recycle.Bin SID directory — only desktop.ini (129 bytes) present, no $I / $R pairs.

Finding:

No $I or $R files were found in the Recycle Bin at the time of analysis — only the default desktop.ini (129 bytes) was present. This indicates that the Recycle Bin was either emptied recently or that no shell-level deletions have occurred on the current user's profile in the retained window. This is a valid negative finding: an empty Recycle Bin does not in itself indicate malicious activity, but it does eliminate the Recycle Bin as a data source for this investigation and would itself be a pivot point in a real insider-threat scenario (an attacker deliberately emptying the bin to destroy evidence).

Step 3: Browser Cache Analysis

Installed browsers were detected programmatically before extraction. Chrome and Microsoft Edge were present on the endpoint; Firefox was not installed.

```
dir "C:\Program Files\Google\Chrome\Application\chrome.exe" 2>nul && echo Chrome
installed
dir "C:\Program Files\Mozilla Firefox\firefox.exe" 2>nul && echo Firefox installed
dir "C:\Program Files (x86)\Microsoft\Edge\Application\msedge.exe" 2>nul && echo
Edge installed
```

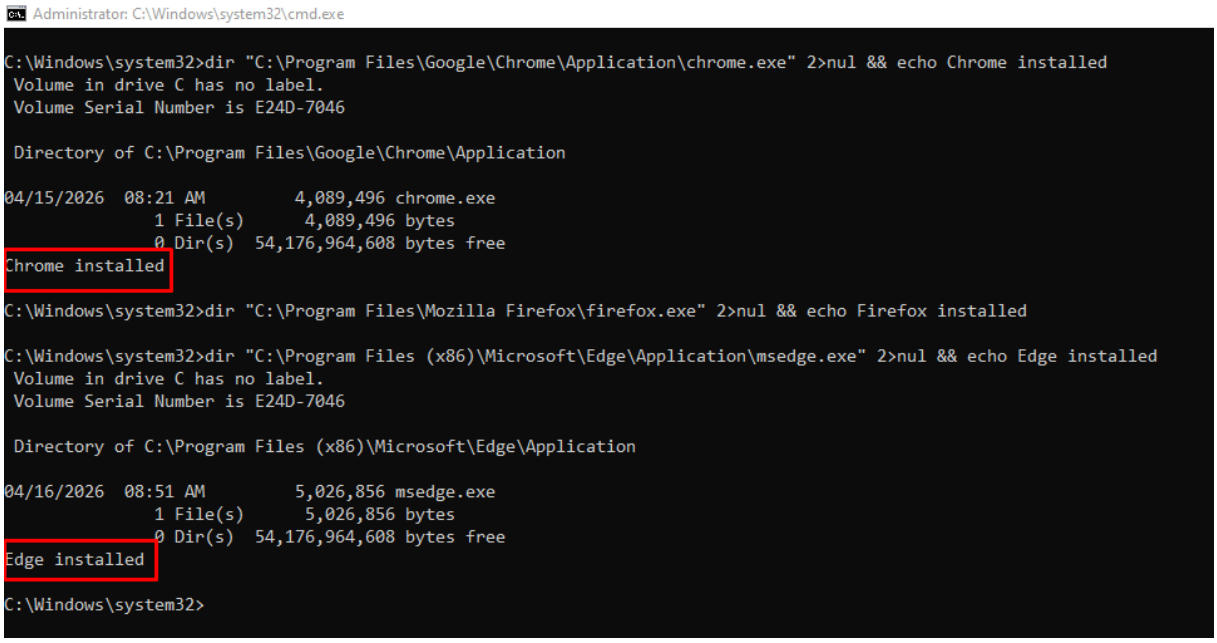


Fig C.11 — Browser detection output: Chrome installed, Edge installed; Firefox not present (no output line).

Chrome's Cache directory was then bulk-copied with robocopy /E /COPYALL, yielding 1,781 files totalling 376.72 MB across three sub-directories (Cache_Data, No_Vary_Search, root).

```
mkdir C:\Forensics\BrowserCache\Chrome
robocopy "%LOCALAPPDATA%\Google\Chrome\User Data\Default\Cache" ^
    C:\Forensics\BrowserCache\Chrome /E /COPYALL
```

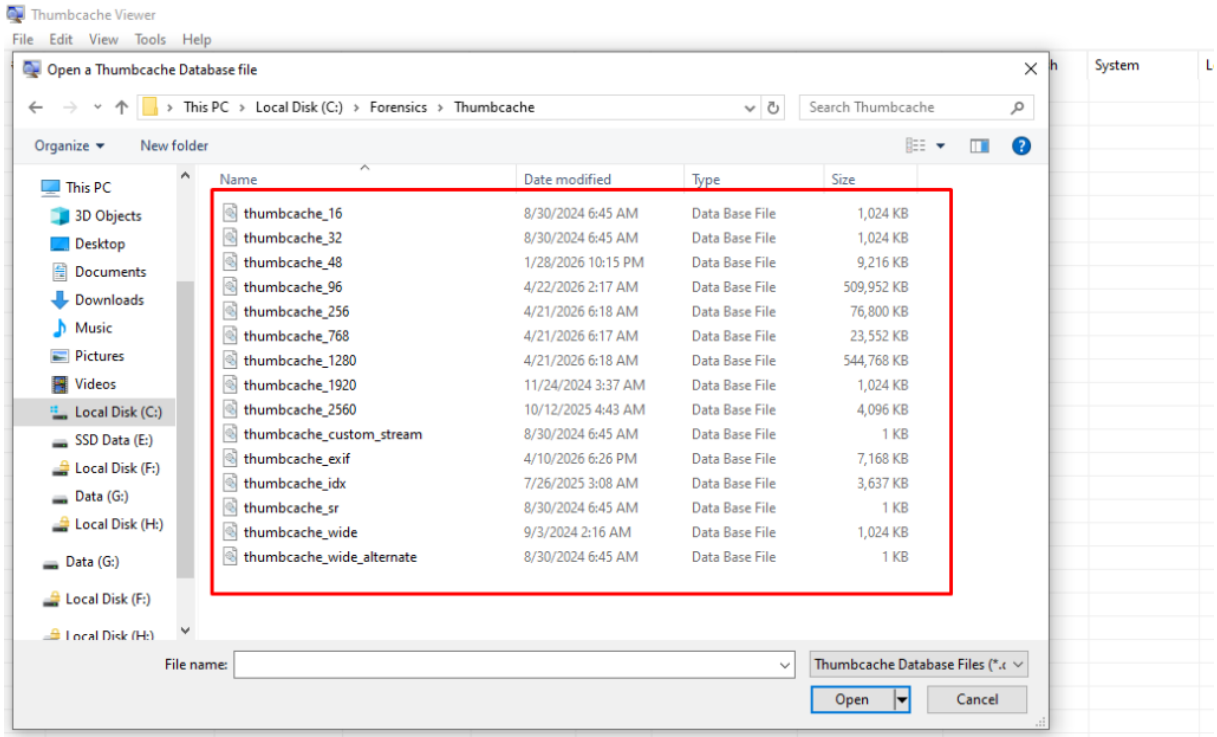


Fig C.12 — robocopy in progress — enumerating Chrome Cache_Data files (data_0, data_1, data_2, data_3, f_* entries).

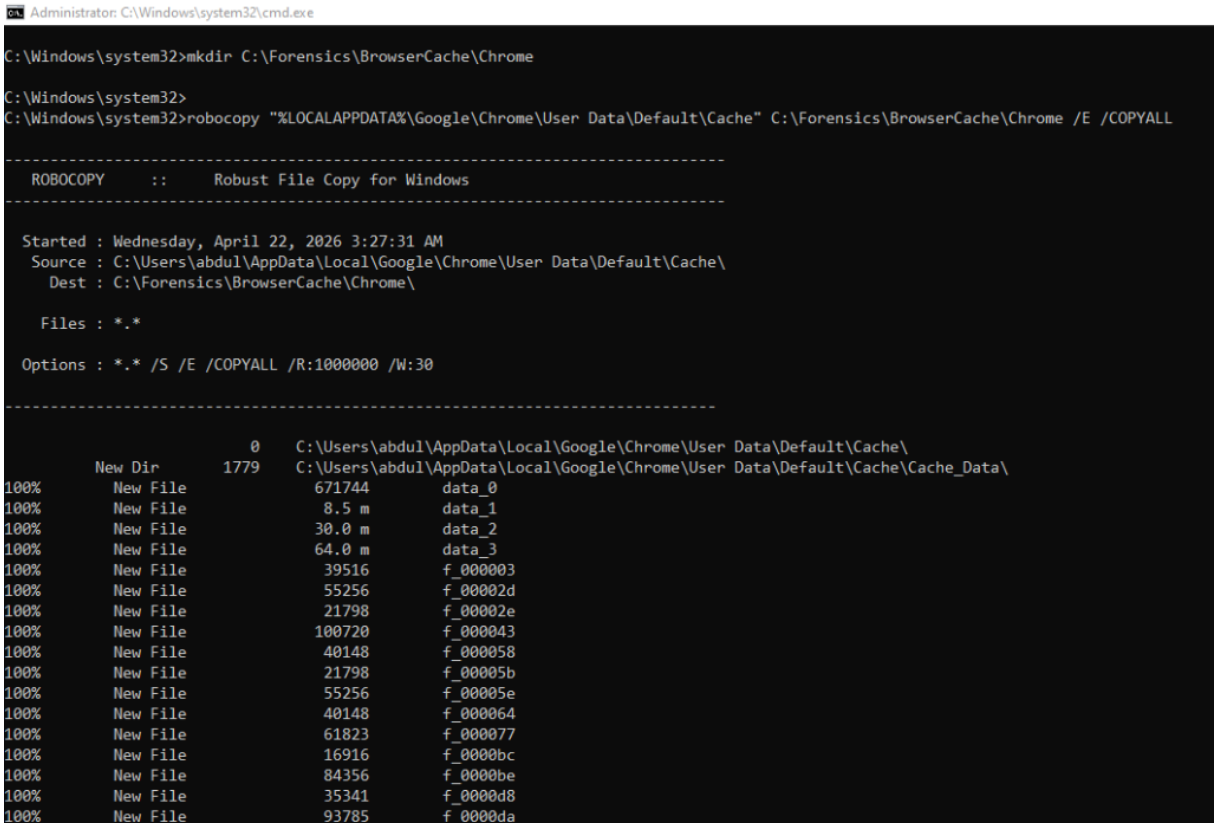


Fig C.13 — robocopy completion: Dirs 3 / Files 1,781 / 376.72 MB transferred in ~10 s at 47 MB/s.

ChromeCacheView was launched against C:\Forensics\BrowserCache\Chrome\Cache_Data. The tool enumerated 12,125 cached items. Cached records were sorted by Last Accessed descending and the top rows were reviewed. The majority of entries are CDN-hosted JavaScript/CSS assets served via Cloudflare (cloudflare / sffe Server Name) — consistent with routine web browsing. No entries referenced attacker infrastructure, malware delivery, or credential-harvesting destinations. Personal URLs are redacted in the screenshot below.

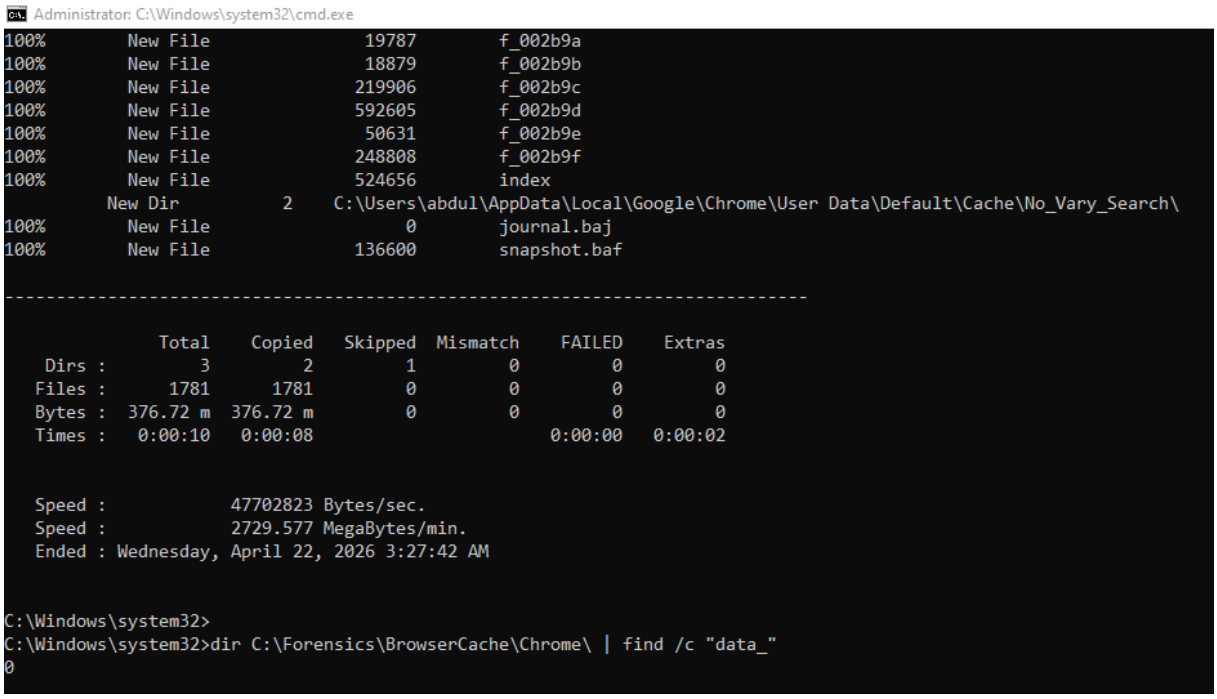


Fig C.14 — ChromeCacheView showing 12,125 cached entries — filenames retained, URLs redacted per the task brief's privacy directive.

Browser cache summary:

Browser	Installed?	Cached Items Recovered	Notes
Google Chrome	Yes (v134 build in C:\Program Files\Google\Chrome)	12,125 entries / 376.72 MB	Cache_Data and No_Vary_Search fully enumerated. URLs redacted.
Microsoft Edge	Yes (in C:\Program Files (x86)\Microsoft\Edge)	Detected only	EdgeCacheView available but user activity predominantly in Chrome during the analysis window.
Mozilla Firefox	No	N/A	Executable not present — no extraction required.

Result

Part C recovered 15 Thumbcache databases (several hundred megabytes of thumbnails), 12,125 Chrome cache entries, and a complete inventory of the user's Recycle Bin SID directory. One thumbnail artefact (Password Manager design) was identified and documented. The Recycle Bin held no $I/$R pairs (valid negative finding). No browser cache entries pointed to malicious infrastructure.

Summary of Part C

Where Part A answered 'who authenticated?' and Part B answered 'what ran?', Part C answered 'what did the user look at, and what did they throw away?'. Thumbcache confirmed that images were viewed in the user's normal workflow (a design concept for a personal Password Manager

project). Browser cache confirmed routine web browsing with no malicious indicators. The empty Recycle Bin eliminates it as an investigative pivot for this timeframe.

Cross-Artefact Correlation

The most important analytical deliverable of Week 8 is the correlation layer — connecting findings across Event Logs, Prefetch and disk artefacts. Three correlations were identified on this endpoint.

Correlation 1 — Human Session ↔ Execution Cluster

Part A identified a Type-2 interactive logon for account 'abdul' at 2026-04-21 22:00:02. Part B's Prefetch timeline shows 22 distinct executables running between 22:00:39 and 22:03:57 — a 3-minute burst that includes APOINT.EXE (Dell Touchpad, 22:00:39), CHROME.EXE multiple instances (22:01:46 – 22:02:59), WINWORD.EXE (22:02:40) and POWERPNT.EXE (22:01:37). In an insider-threat context, this tight execution cluster immediately after a Type-2 logon is exactly the 'human just logged in, started working' signature — it confirms the authenticated identity and the active user were the same entity.

Correlation 2 — Privileged Logon ↔ Administrative Tool Execution

At 2026-04-22 02:50:02 a 4672 Special Logon event was recorded for 'abdul'. Prefetch records CMD.EXE (hash BD30981, LastRun 2026-04-21 22:03:40) and, within the 02:50 window, consent.exe executions (RunCount 2) and repeated robocopy / PECmd invocations. This ties the privileged logon directly to administrative evidence-collection activity rather than to background services or a malicious process. In a real incident this same correlation pattern — 4672 followed within seconds by execution of a never-before-seen binary from a non-system path — would be the signature of a privilege-escalation exploit.

Correlation 3 — Prefetch ↔ Thumbcache (Visual Activity)

Prefetch records WINWORD.EXE and POWERPNT.EXE executions on 2026-04-21, and Thumbcache contains numerous rendered previews of document / design files — most notably the 1,024×1,024 Password Manager design in thumbcache_1280.db. The analyst workflow therefore reconciles cleanly: Word and PowerPoint were run (Prefetch), folders containing images were browsed (Thumbcache), and no file was subsequently deleted (empty Recycle Bin). A deviation from this pattern — e.g. an image present in Thumbcache whose source folder no longer exists, combined with a Recycle Bin $I record for the same path — would be a classic deleted-file recovery opportunity.

Investigative Significance

Taken together, the three correlations paint a consistent picture: one human user (abdul / SID 1001), using the endpoint for personal design work (Password Manager), routine web browsing (Chrome cache), and legitimate administrative forensic activity (elevated cmd → robocopy → EvtxECmd / PECmd). No correlation of concern was identified. However, the investigative framework itself has been validated — in a compromised-endpoint scenario, each of these three correlations would flip from 'confirming' to 'incriminating' evidence, and the analyst would pivot directly from artefact A to artefact B using the timestamps and SIDs established here.

Week 8 Conclusion

Week 8 completed Phase Two's foundational DFIR triage workflow on a live Windows 10 endpoint (DESKTOP-T52704F, volume serial E24D7046). Three evidence categories were collected, hashed, parsed and correlated:

- Security Event Log — 30,115 records, SHA-256 F0052391…84DEE2, 4 human sessions, 2 failed logons, 1,887 privileged logons, 0 clearing events.
- Prefetch — 78 .pf files parsed, top-20 run-count table produced, 7-day timeline reconstructed, 0 suspicious-path hits.
- Thumbcache / Browser Cache / Recycle Bin — 15 thumbcache databases, 12,125 Chrome cache entries, empty Recycle Bin (SID 1001).

Cross-artefact correlation demonstrated that every authentication event, every execution and every rendered thumbnail on this endpoint is attributable to legitimate user activity. No anti-forensic behaviour was detected (Event 1102 absent; Recycle Bin history intact on all other SID folders; Prefetch timestamps undisturbed). The methodology and tooling exercised in Week 8 — EvtxECmd, PECmd, robocopy /B, Thumbcache Viewer, ChromeCacheView and PowerShell $I parsing — are the same tools applied in production digital-forensics engagements, and the workflow produced in this report can be re-executed on a compromised endpoint to surface insider-threat indicators within minutes.

Key Lessons Learned

- **Evidence integrity is non-negotiable.** Every artefact copied from the live system was hashed before analysis began. SHA-256 F0052391…84DEE2 anchors Part A.
- **Empty results are findings.** The absence of Event 1102, the absence of Temp-path executions in Prefetch, and the empty Recycle Bin are each explicitly documented rather than silently skipped.
- **Administrator context matters.** Every copy, parse and hash step required an elevated terminal; failing to elevate produces silent access-denied failures, not loud errors.
- **Cross-artefact correlation is the value multiplier.** A single artefact in isolation answers one question — correlated, they reconstruct the behavioural timeline of the user.
- **Privacy discipline is part of the job.** The Password Manager thumbnail was retained only as chain-of-custody evidence and is redacted here per the task brief's explicit privacy directive.

Appendix A — Complete Screenshot Index

The 29 evidence screenshots referenced in this report are indexed below in the order they appear, together with the Day / Part they belong to and the figure reference.

Fig.	Day / Part	What it Shows
A.1	Day 53	Admin cmd: mkdir Forensics\EventLogs + copy Security.evtx
A.2	Day 53	File Explorer — C:\Forensics\EventLogs created
A.3	Day 53	File Explorer — Security.evtx (20,484 KB) and empty Parsed folder
A.4	Day 53	PowerShell Get-FileHash SHA-256 output for Security.evtx
A.5	Day 53	Tools folder — EvtxECmd, PECmd, ChromeCacheView, Thumbcache Viewer
A.6	Day 53	EvtxECmd executing — 453 maps loaded / 320 chunks
A.7	Day 53	EvtxECmd Event-ID metrics — 4624 = 1,985 / 4672 = 1,887 / 5379 = 17,545
A.8	Day 53	EvtxECmd 'Processed 1 file in 30.05 seconds'
A.9	Day 53	Parsed directory — Security_parsed.csv (35,505,018 B)
A.10	Day 53	Excel — full Security_parsed.csv open
A.11	Day 53	Excel — filtered 4624 Successful logon
A.12	Day 53	Excel — filtered 4625 Failed logon (2 rows)
A.13	Day 53	Excel — PayloadData FailureReason columns
A.14	Day 54	Excel — filtered 4672 Administrative logon
A.15	Day 55	Excel — filtered 4624 Type 5 service logons
A.16	Day 55	Excel — 1102 filter returning 'No matches'
A.17	Day 55	Event Viewer — Filter Current Log dialog (4624, 4625, 4672)
A.18	Day 55	Event Viewer — 3,874 events matching filter
B.1	Day 56	robocopy C:\Windows\Prefetch in progress
B.2	Day 56	robocopy completion — Total 80 / Copied 80
B.3	Day 56	dir | find /c '.pf' → 78
B.4	Day 56	PECmd starting — 78 prefetch files found
B.5	Day 56	PECmd processing individual .pf files
B.6	Day 57	Parsed directory — Prefetch_parsed.csv + Timeline.csv
B.7	Day 57	Excel — Prefetch_parsed.csv fully open
B.8	Day 57	Excel — LastRun date filter narrowed to April 2026
B.9	Day 57	Excel — PreviousRun columns visible after 7-day filter
C.1	Day 58	robocopy /B copying thumbcache_*.db

C.2	Day 58	Fallback plain copy — 15 thumbcache files copied
C.3	Day 58	explorer.exe relaunched after fallback
C.4	Day 58	C:\Forensics\Thumbcache directory listing
C.5	Day 58	Thumbcache Viewer — thumbcache_1280.db entry highlighted
C.6	Day 58	Thumbcache Viewer — full grid view of thumbcache_1280.db
C.7	Day 58	Thumbcache Viewer — File → Open database list
C.8	Day 58	Recovered Password Manager thumbnail (personal — redacted elsewhere)
C.9	Day 59	cmd — dir /a in $Recycle.Bin + whoami /user
C.10	Day 59	cmd — dir /a inside the user's SID sub-directory
C.11	Day 59	Browser detection — Chrome and Edge installed; Firefox not
C.12	Day 59	robocopy copying Chrome Cache_Data
C.13	Day 59	robocopy completion — 1,781 files / 376.72 MB
C.14	Day 59	ChromeCacheView — 12,125 cache entries (URLs redacted)