ODIN

TLP:CLEAR

ODIN RESEARCH · MONOGRAPH 37 / 30 · CREDENTIAL THEFT

Python Stealers and AuroraStealer C2

Evidence and Defensive Priorities

How a 3,508-file Python server and 2,695 commodity stealer feed endpoints separate what to hunt from what not to block

Two-week observation window ending 20 November 2026 · Prepared November 2026
Full Report · Public edition · Version 1.1

18 / 981
Directory slices examined closely (3,504 reported files), yielding 981 unique file entries

4
Custom Python tool suites on that server: two stealer builder and collector suites, a C2 workspace and a modular testing kit

3
Windows browser report bundles on the server holding renamed Microsoft Edge SQLite and DPAPI files (paths withheld)

0 / 3
Confirmed block-list endpoints, plus 3 isolated AuroraStealer-labelled endpoints observed on 13 November 2026 that remain unconfirmed (all 2,695 endpoints withheld)

What Is in This Report. 11 chapters and two appendices.

How a 3,508-file Python server and 2,695 commodity stealer feed endpoints separate what to hunt from what not to block

01	Executive Summary	04
02	Two Datasets, One Analytical Model	07
03	Custom Python Infostealers and Edge SQLite Bundles	10
04	Custom Python C2 Stagers and Modular Exploit Scripts	13
05	Eight Commodity Infostealer Families and AuroraStealer Endpoints	16
06	Feed Labels Against Unconfirmed Stealer Servers	19
07	What Defenders Should Do	22
08	Detection Rules	24
09	Indicators	27
10	Scope and Limits	29
11	Sources	31
A	Indicators of compromise (6)	33
B	Detection rules (4 files)	34

IN SHORT

In a two-week observation window ending 20 November 2026, Odin by Brandefense found 3,508 reported files across 21 open directory listings on one US server. Four custom Python offensive suites sat on that host. Separately, we examined 2,695 endpoints across eight commodity infostealer families (from 2,770 feed records). All 2,695 endpoints remained unconfirmed pending an independent adversary signal. Keep unconfirmed feed addresses off block lists, protect Microsoft Edge credential files, and close open lab directories.

01

CHAPTER 01

Executive Summary

IN SHORT

In a two-week observation window ending 20 November 2026, Odin by Brandefense found 3,508 reported files across 21 open directory listings on one US server. Four custom Python offensive suites sat on that host. Separately, we examined 2,695 endpoints across eight commodity infostealer families (from 2,770 feed records). All 2,695 endpoints remained unconfirmed pending an independent adversary signal. Keep unconfirmed feed addresses off block lists, protect Microsoft Edge credential files, and close open lab directories.

INSIDE THIS CHAPTER

Three Decisions for Leaders 04

The Study in Numbers 05

01 Executive Summary

Security teams meet credential-theft infrastructure in two very different forms; dark web monitoring covers the stolen data once it is offered for sale. Some operators build custom Python stealers, droppers and web testing scripts directly on rented virtual servers. Commercial stealer feeds, meanwhile, list thousands of candidate addresses across older and active malware families. This report examines both and shows how defenders can use cyber threat intelligence to keep unverified servers off firewall block lists.

1. **One open US server stages four custom Python stealer, C2 and exploit suites.**
   Across 21 open web directory listings, the server reports 3,508 files (981 unique entries across 18 directory slices) with timestamps from 8 to 14 August 2026. Two Python stealer suites sit beside three Windows browser report bundles holding renamed Microsoft Edge SQLite and DPAPI files. No runnable offensive code or victim data is published here.

2. **Custom C2 stagers and modular Python scripts combine droppers with web and LAN tools.**
   On the same server, a 50-file C2 folder pairs an operator key, Windows service droppers, PowerShell beacons and a screenshot decoder with a 2,177-file Python 3.12 environment. Nearby folders hold four reverse-shell logs and 26 Python scripts for web authorization checks, SQL injection, ARP spoofing and keylogging.

3. **All 2,695 commodity stealer endpoints remain unconfirmed for blocking.**
   Across eight stealer families (2,770 feed records, 2,695 endpoints on 2,440 host entries), three AuroraStealer-labelled endpoints observed on 13 November 2026 remain unconfirmed pending an independent adversary signal. The other 2,692 remain unconfirmed. Some show CDN, certificate, port or ISP collisions; most lack corroborating evidence.

Three Decisions for Leaders

1. **Keep unconfirmed stealer feed addresses off automatic block lists.** All 2,695 commodity stealer endpoints lack an independent adversary signal, so route them to TLS JARM and behavioural hunting instead of firewall blocking.
2. **Guard Chromium and Microsoft Edge credential files on endpoints.** Turn on Edge Application-Bound Encryption and alert whenever Python, PowerShell or script hosts read `Login Data`, `Cookies` or `Local State` files.
3. **Shut down unauthenticated HTTP directory listings on cloud and lab hosts.** Use external attack surface management to scan your own cloud ranges on high HTTP ports so test repositories, keys and browser artifacts never sit in public folders.

The Study in Numbers

Number	What it is
3,508 / 21	Files reported across 21 directory listings on one US web server (address, port and paths withheld)
18 / 981	Directory slices examined closely (3,504 reported files), yielding 981 unique file entries
4	Custom Python tool suites on that server: two stealer builder and collector suites, a C2 workspace and a modular testing kit
3	Windows browser report bundles on the server holding renamed Microsoft Edge SQLite and DPAPI files (paths withheld)
2,770 / 2,695	Feed records across eight commodity stealer families, and the 2,695 unique endpoints we examined
0 / 3	Confirmed block-list endpoints, plus 3 isolated AuroraStealer-labelled endpoints observed on 13 November 2026 that remain unconfirmed (all 2,695 endpoints withheld)
0	Shared hosts between the US Python server and the 2,440 commodity stealer host entries

How to use this report.

Security leaders can focus on Chapters 1, 6 and 7. Endpoint, browser and application security engineers should read Chapters 3, 4 and 7. SOC analysts and detection engineers should start with Chapters 5, 8 and 9, while threat intelligence analysts can use Chapters 2, 5, 6 and 10.

02

CHAPTER 02

Two Datasets, One Analytical Model

IN SHORT

Zero hosts link the US Python workbench to the 2,440 commodity stealer host entries. We present them together as an analytical model comparing custom tool development with commercial stealer feeds, not as one actor.

INSIDE THIS CHAPTER

Timestamps and withholding boundaries 08

Where the commodity stealer feed points 08

02 Two Datasets, One Analytical Model

Custom Python stealer builders show how an operator or student lab assembles browser harvesting, PowerShell persistence and web testing scripts from scratch. Commodity stealer feeds show the opposite problem: thousands of unverified HTTP and TLS records where almost every raw entry is noise. Putting either dataset on an automatic firewall block list would expose downloadable browser databases or cut off legitimate websites and mail servers.

We compared the US Python server against every host in our eight-family stealer set. None of the 2,440 stealer host entries matches the US server. The connection between the two datasets is therefore functional rather than actor-linked. Figure 1 shows the four groups in this model and the right defensive response for each.



Figure 1: Custom Python weapon suites and commodity stealer feeds split duties (analytical model)

Dataset	Volume examined	Host entries	Published addresses	Defensive action
US multi-suite Python server (port withheld)	3,508 reported files (21 listings; 981 unique entries)	1	0 (withheld)	Hunt open directory listings and protect Edge SQLite files
Eight-family commodity stealer set	2,695 endpoints (from 2,770 feed records)	2,440	0 (withheld)	Hunt the six TLS JARM fingerprints; do not block by IP

Timestamps and withholding boundaries

Files and directory listings on the US web server carry server timestamps from 8 to 14 August 2026. Those dates are the server's own file and folder timestamps, not how long Odin watched the host. The server mixes vocational-styled testing scripts, web applications and harvested Microsoft Edge database files. A reader with its address, port or paths could fetch them, so we withhold its IPv4 address, port, directory paths, filenames and file hashes.

Across eight stealer families, 2,440 host entries represent 2,437 physical IPv4 addresses. Three AuroraStealer-labelled hosts were observed on 13 November 2026. The other 2,434 distinct addresses remain unconfirmed across 2,437 tokenized host entries.

The three November hosts also appear in older AuroraStealer feed entries with the same machine certificates and JARMs. No independent protocol response, beacon or workbench link confirms them, so we withhold all 2,437 addresses and their per-host ports. Seven withheld hosts carry candidate labels for two stealer families.

Where the commodity stealer feed points

Most unconfirmed feed records lack current network attribution. Across all 2,695 endpoints, 1,915 carry no resolved country code and 1,894 carry no resolved autonomous system. Most of those unattributed rows are older port 80 entries for LokiBot (681), AZORult (347), Oski (321) and AgentTesla (226).

Where hosting is resolved, shared content-delivery networks and European cloud providers lead the list. Attributed endpoints cluster in the United States (222), the Netherlands (209), Germany (163), Russia (88) and Finland (27).

Network provider	Endpoints in the 2,695-row set	Notes
Cloudflare, Inc.	152	Shared CDN proxy (73 LokiBot, 69 AZORult, 5 AgentTesla, 5 Meduza); never block by IP
Hetzner Online GmbH	143	90 of 93 Vidar rows and 52 AuroraStealer rows (including 1 isolated 13 November endpoint and 3 demoted collision rows)
Galaxy LLC	114	Mostly unconfirmed AuroraStealer (110) and RisePro (4) entries
AEZA GROUP Ltd	32	Unconfirmed AuroraStealer (20) and Meduza (12) entries
Partner LLC	29	Unconfirmed AuroraStealer entries
Delis LLC	26	LokiBot (16), AuroraStealer (8) and AgentTesla (2) entries
Hosting technology LTD	18	Unconfirmed AuroraStealer entries
Amazon.com, Inc.	16	Unconfirmed AZORult (13) and suspected AuroraStealer (2) entries

03

CHAPTER 03

Custom Python Infostealers and Edge SQLite Bundles

IN SHORT

Three directory slices (41 files) on the US server hold custom Python stealer builders and two HTTP receivers (paths withheld). Three companion report folders (15 files) hold renamed Microsoft Edge SQLite databases and DPAPI key files.

INSIDE THIS CHAPTER

How the custom Python stealer builders and collectors are organised 10

Inside the three Windows Edge report bundles 11

03 Custom Python Infostealers and Edge SQLite Bundles

Commercial stealers hide their builders and collection panels behind compiled binaries and rented dashboards. A self-hosted Python workbench leaves its entire file layout visible whenever an operator runs an open web directory (T1583.003). Figure 2 maps each suite on the server, along with the commodity stealer feed, to the target surface and our detection rules.

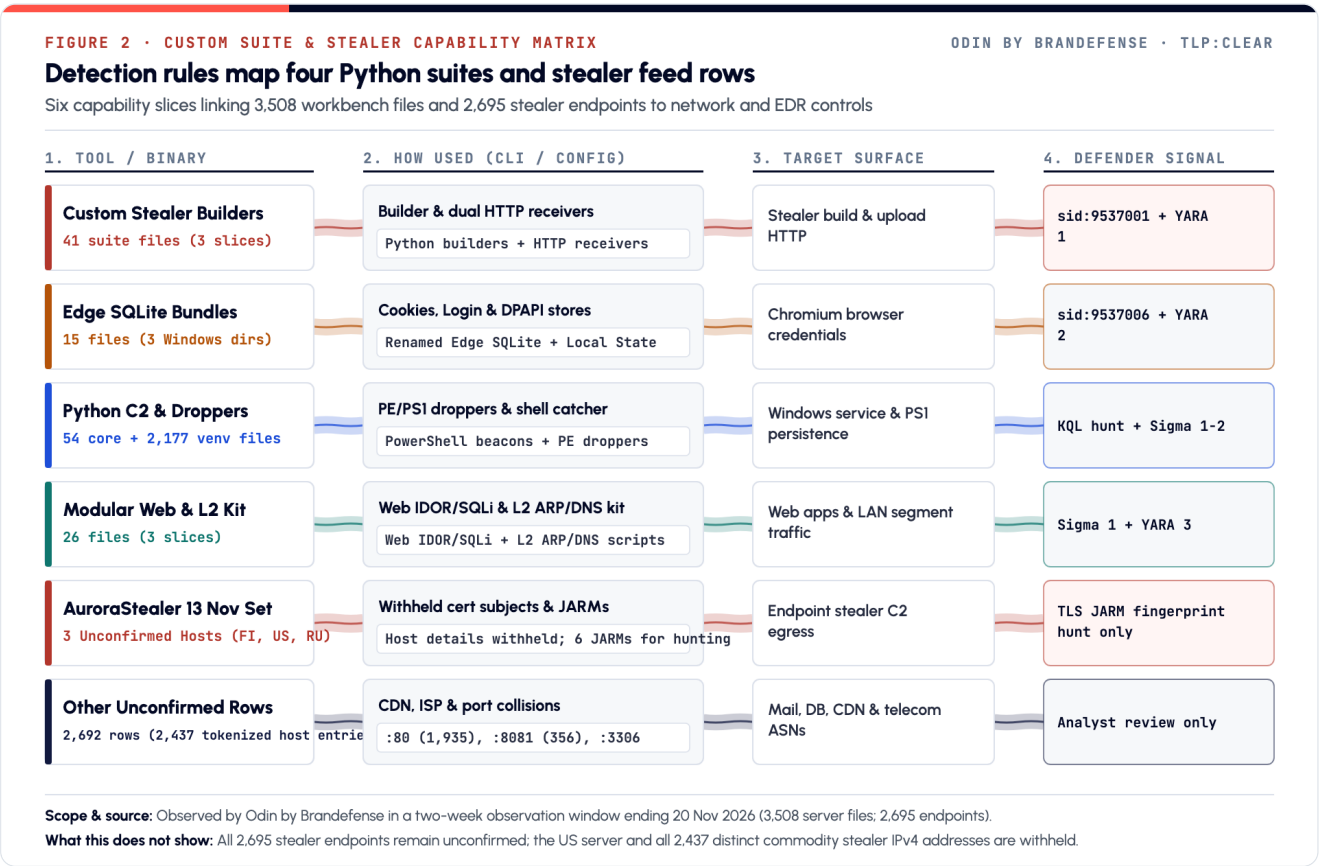


Figure 2: Detection rules map four Python suites and stealer feed rows

How the custom Python stealer builders and collectors are organised

Three directory slices totaling 41 reported files, backed by helper scripts in the server root (306 reported files; paths withheld), cover payload generation and archive collection:

- **Payload generator repository (37 files; path withheld).** Stored as a Git repository on branch `main`, this folder holds a Python stealer payload generator script and a payload binder script. In Python stealer kits, the generator writes callback URLs and target options into a new build, while the binder pairs the output with a decoy file.
- **Primary collector folder (3 files; path withheld).** Holds an HTTP upload receiver script (8,174 bytes), a report viewer script (1,435 bytes) and a listener monitor shell script (213 bytes). The receiver script accepts HTTP uploads and unpacks them into report folders, the viewer formats harvested summaries, and the shell script keeps the listener running.
- **Expanded collector folder (1 file; path withheld) and root scripts.** Holds a second HTTP upload receiver script measuring 11,049 bytes, 2,875 bytes larger than the primary receiver script, under a "fully undetectable" folder

naming scheme. The root folder also holds a top-level payload generator script (4,381 bytes) and a report-directory watcher script (1,909 bytes). No script code is reproduced in this report.

Inside the three Windows Edge report bundles

Three folders on the server (15 reported files in total; paths withheld) show the exact folder layout written when a Windows browser upload lands on the collector. All three folder timestamps fall on 8 August 2026:

Folder on server	Files	Contents and byte sizes
Browser report bundle A (path withheld)	6	Compressed upload archive (10,988 B), JSON metadata file (511 B), HTML summary report (3,403 B) and three renamed Microsoft Edge stores
Browser report bundle B (path withheld)	3	Renamed Microsoft Edge login SQLite database (61,440 B), cookie SQLite database (20,480 B) and DPAPI Local State JSON file (23,060 B)
Browser report bundle C (path withheld)	6	Compressed upload archive, JSON metadata file, HTML summary report, and renamed Microsoft Edge login, cookie and Local State files

In Browser report bundle A, the renamed Microsoft Edge login SQLite database measures 49,152 bytes. The cookie database measures 45,056 bytes, and the DPAPI Local State JSON file measures 18,113 bytes.

Chromium protects saved logins in `Login Data` and cookies in `Cookies` with an AES-256-GCM key stored in `Local State` (`os_crypt.encrypted_key`, T1555.003, T1539). Windows DPAPI wraps that key for the signed-in user, so the copied SQLite files alone do not unlock the credentials. Non-browser script access to `Login Data`, `Cookies` and `Local State` gives defenders a reliable host-agnostic pattern for endpoint and proxy alerts. We did not open database records, and all file hashes and exposed-host filenames are withheld.

CHAPTER 04

Custom Python C2 Stagers and Modular Exploit Scripts

IN SHORT

Alongside its stealer folders, the US server stages a custom Python C2 workspace (50 core files plus 2,177 Python 3.12 library files) and four reverse-shell log files (paths withheld). It also holds 26 modular Python scripts for web, Layer-2 and keylogger testing.

INSIDE THIS CHAPTER

How the Python C2 workspace and reverse-shell logs are built 13
What sits inside the modular Python testing kit 14
Full breakdown of the 3,508 reported files 14

04 Custom Python C2 Stagers and Modular Exploit Scripts

Putting browser stealers next to interactive C2 builders and network exploit scripts on one server shows how an operator can move from stolen cookies to persistent access. At the same time, the server also holds a vocational-styled testing folder and 889 co-hosted web-application files. Staged files on an open port therefore do not establish live intrusion activity on their own. Figure 4 shows how all 3,508 reported files divide across the server.



Figure 4: One open HTTP port exposes four offensive Python suites and apps

How the Python C2 workspace and reverse-shell logs are built

Two related C2 folders hold the server's C2 and post-exploitation components. They contain 50 core reported files and a second Git working-tree reference in the root listing; paths are withheld:

- **Operator key and Python 3.12 server stack.** In the C2 folder, an operator authentication file (411 bytes; path withheld) sits beside a desktop screenshot decoder script (path withheld), with key contents and hash withheld. The virtual environment (2,177 reported files; path withheld) reveals a Python 3.12 web and WebSocket C2 server built with `flask`, `werkzeug`, `jinja2`, `click`, `blinker`, `h11`, `dnspython`, `greenlet`, `socketio`, `engineio` and `bidict`.
- **Windows PE droppers, PowerShell beacons and HTA launcher.** In the C2 payload subfolders (paths withheld) sit a Python dropper-builder script, two compiled Windows executables (a direct dropper and a Windows service installer), a PowerShell privilege-escalation script and a persistent PowerShell beacon stager (T1059.001). The root folder adds a 1,346-byte PowerShell beacon script and a 235-byte HTML Application launcher meant for `mshta.exe` (filenames withheld).

· **Reverse-shell logs (4 files; paths withheld).** Holds a master reverse-shell listener log and three per-session log files whose filenames and embedded peer addresses are withheld.

What sits inside the modular Python testing kit

Three directory slices totaling 26 reported files (paths withheld) organise a modular Python testing and exploitation kit under a vocational-education folder name:

Subfolder group	Files	Structural contents (filenames withheld)	Target surface
Web testing subfolder (path withheld)	6	Six Python web scripts covering IDOR checks, SQL injection, XSS scanning, session cookie analysis, directory enumeration and web crawling	Web object authorization (IDOR), SQL injection, XSS and session cookies
Network exploitation subfolder (path withheld)	5	Five Python network scripts covering ARP cache poisoning, DNS spoofing, packet capture, credential brute-forcing and SYN floods	Local network ARP cache poisoning (T1557.002), DNS spoofing and packet capture
Suite root, post-exploitation and shell subfolders (paths withheld)	15	A Python keylogger script, a reverse-shell generator script, `.git/` metadata and study notes (46,847 bytes; author identifiers withheld)	Endpoint keystroke logging and reverse-shell one-liners

Full breakdown of the 3,508 reported files

The remaining 889 reported files belong to co-hosted developer applications and cloud maintenance scripts (paths withheld). They include a Python 3.12 web portal environment (501 files) and a Node and TypeScript web scraper project (355 files, 26 media assets and 2 database scripts). A DigitalOcean droplet update script (3,817 bytes; path withheld) and three small sub-directory listings (4 files) complete that count. An OAuth configuration file (3,261 bytes; filename withheld) also sits in the server root.

Server folder group	Listings (examined)	Reported files	Share
Custom C2 Python 3.12 virtual environment (path withheld)	1 (1)	2,177	62.06%
Co-hosted web apps and DigitalOcean script (paths withheld)	7 (4)	889	25.34%
Root directory listing (path withheld)	1 (1)	306	8.72%
Custom Python stealer suites and three browser report bundles (paths withheld)	6 (6)	56	1.60%
Custom C2 core files and reverse-shell logs (paths withheld)	2 (2)	54	1.54%
Modular Python web, network and post-exploitation suite (paths withheld)	3 (3)	26	0.74%
Total on server	**21 (18)**	**3,508**	**100.00%**

05

CHAPTER 05

Eight Commodity Infostealer Families and AuroraStealer Endpoints

IN SHORT

Across 2,695 endpoints labelled with eight commodity infostealer families, all 2,695 remain unconfirmed pending an independent adversary signal. Hunt the six TLS JARM fingerprints across outbound TLS traffic instead of blocking unverified feed addresses by IP.

INSIDE THIS CHAPTER

Three isolated AuroraStealer-labelled endpoints to hunt 16

Port habits and shared TLS fingerprints across 2,695 endpoints 17

05 Eight Commodity Infostealer Families and AuroraStealer Endpoints

Commercial infostealers harvest the exact same browser cookies and saved logins seen in the three Windows browser report bundles, but at broad scale. Our second dataset covers eight families named in upstream threat feeds: LokiBot, AZORult, AuroraStealer, Oski, AgentTesla, RisePro, Meduza and Vidar. Those family names come from feed rules matching HTTP banners, paths or TLS traits and do not name an actor on their own. Figure 3 shows how browser harvesting, C2 staging and stealer feed rows map to defender detections.

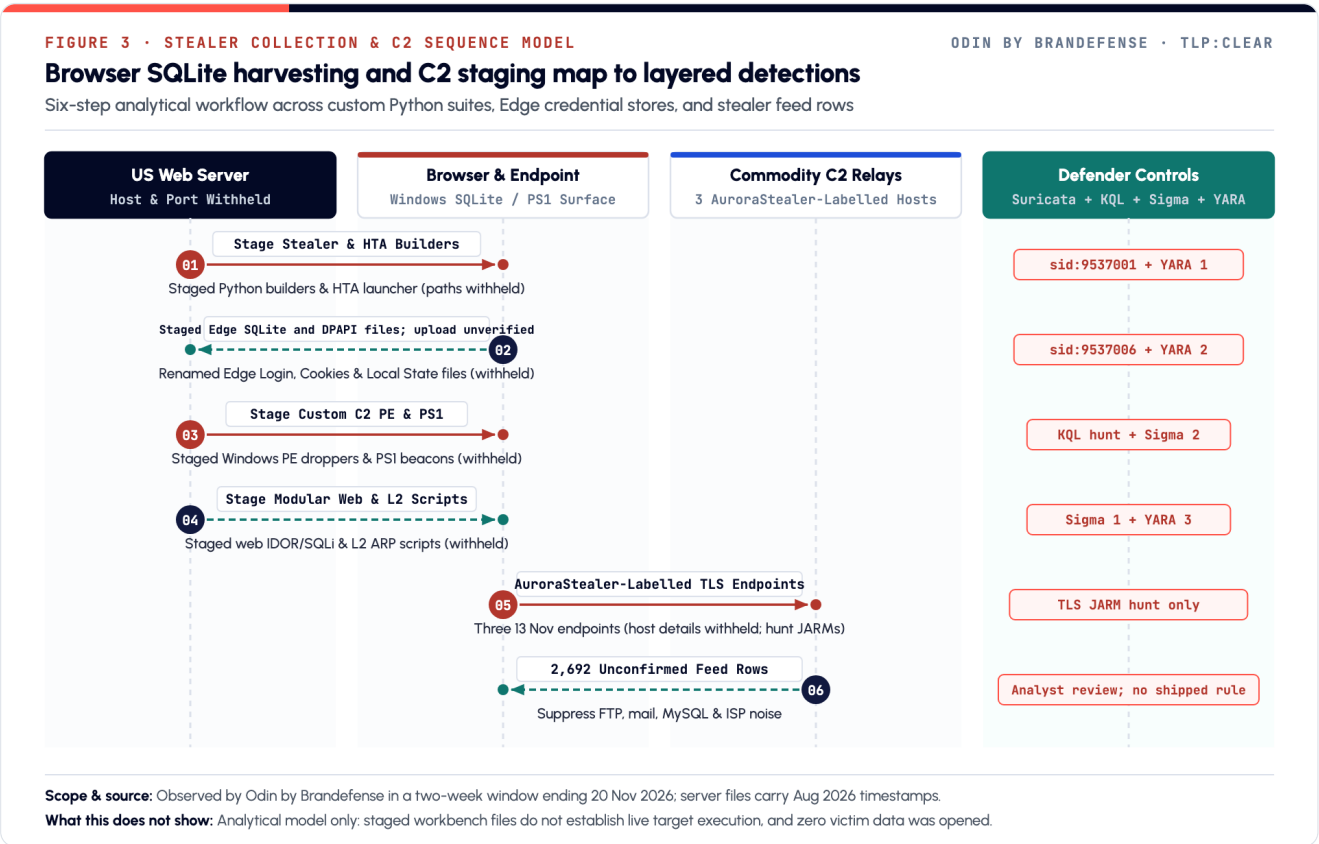


Figure 3: Browser SQLite harvesting and C2 staging map to layered detections (analytical model)

Three isolated AuroraStealer-labelled endpoints to hunt

Three AuroraStealer-labelled endpoints observed on 13 November 2026 remain unconfirmed pending an independent adversary signal. These three endpoints sit on dedicated virtual servers and present consistent self-signed machine certificates and TLS JARM fingerprints. Our dataset contains no live protocol response, beacon or workbench link for their exact IPv4 addresses:

Family	Address and port	Country and provider	Certificate CN	TLS JARM prefix
AuroraStealer	Address and port withheld	Finland, Hetzner Online GmbH	subject withheld	`2ad2ad0002ad`
AuroraStealer	Address and port withheld	United States, Majestic Hosting Solutions, LLC	subject withheld	`2ad2ad16d2ad`
AuroraStealer	Address and port withheld	Russia, UFO Hosting LLC	subject withheld	`14d14d00014d`

Three practical details stand out across these three unconfirmed hosts:

- **Same machine certificates across repeated feed observations.** All three hosts also appear in older AuroraStealer feed records carrying the exact same machine certificates (subjects withheld) and JARMs; per-host ports are withheld.
- **Provider shift on the Russian VM.** Earlier feed records for the Russian virtual machine resolved under STARK INDUSTRIES SOLUTIONS LTD before routing through UFO Hosting LLC with the same VM certificate.
- **Hunt by JARM, not by unverified IP block.** All three hosts pair management or web TLS listeners with older AuroraStealer feed records. We withhold all three IPv4 addresses, their service ports and their machine certificate subjects, and retain their JARMs for outbound TLS hunting.

Port habits and shared TLS fingerprints across 2,695 endpoints

Port choices divide cleanly between older PHP-panel stealers and newer Go or C++ builds. Plain HTTP on port 80 accounts for 1,935 of the 2,695 endpoints (71.80%), and HTTPS on port 443 accounts for 332 (12.32%). Those two ports dominate LokiBot, AZORult, Oski, AgentTesla, Meduza and Vidar. By contrast, TCP 8081 accounts for 356 endpoints (13.21%), driven by RisePro (226 of 239 endpoints; 94.56%) and AuroraStealer (129 of 382; 33.77%).

TLS JARM fingerprints cluster across multiple hosts and families, which makes them useful for hunting and unsafe for standalone blocking:

- **RisePro cluster.** Eleven RisePro-labelled endpoints share JARM prefix `15d3fd16d29d` (full values in Chapter 9).
- **AuroraStealer and Vidar cluster.** Nine AuroraStealer and three Vidar endpoints (12 total) share JARM prefix `3fd3fd000000`.
- **Three-family cluster.** Eight AuroraStealer, two Meduza and one AgentTesla endpoint (11 total) share JARM prefix `29d3fd00029d`.

CHAPTER 06

Feed Labels Against Unconfirmed Stealer Servers

IN SHORT

Blocking all 2,695 feed-labelled stealer endpoints would break Cloudflare websites, MySQL databases, POP3 and IMAP mail servers, and broadband ISP customers. None of the 2,695 endpoints (0.00%) carries the independent second signal required for a firewall block list.

INSIDE THIS CHAPTER

Four reasons all 2,695 endpoints stay off block lists 20

06 Feed Labels Against Unconfirmed Stealer Servers

A reliable block list is defined by what it refuses to block. Figure 5 compares the number of endpoints examined in each commodity stealer family against the zero confirmed C2 servers that met our two-signal block standard.

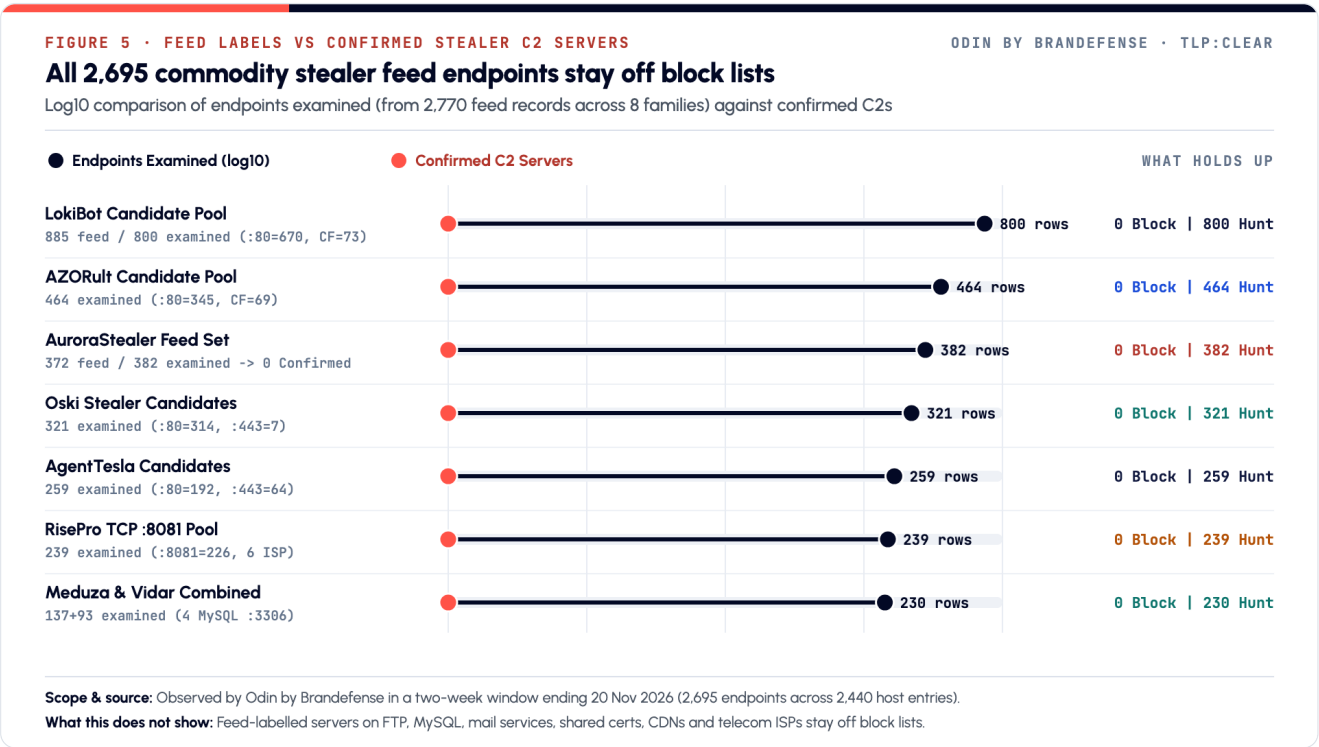


Figure 5: All 2,695 commodity stealer feed endpoints stay off block lists

In the table below, LokiBot's global feed count is 885 and we examined up to 800 exported endpoints. AuroraStealer's current single-port feed summary is 372, while deduplicating multi-port observations gave 382 endpoints across 261 hosts. Across all eight families, per-family host counts sum to 2,444 (2,440 unique host entries across the combined set).

Family	Feed records	Endpoints examined	Hosts	Confirmed (block)	Unconfirmed (hunt only, withheld)
LokiBot	885	800	751	0	800 (670 on port 80; 73 behind Cloudflare)
AZORult	464	464	440	0	464 (345 on port 80; 69 behind Cloudflare; 1 shared cert)
AuroraStealer	372	382	261	0	382 (3 isolated 13 Nov hosts, 9 suspected, 9 shared certs, 5 port collisions, 356 unverified)
Oski	321	321	315	0	321 (314 on port 80, 7 on 443)
AgentTesla	259	259	219	0	259 (192 on port 80, 64 on 443)
RisePro	239	239	232	0	239 (226 on port 8081; 6 telecom ISP; 1 open directory)
Meduza	137	137	137	0	137 (136 on port 80; 2 telecom ISP; 2 shared certs)
Vidar	93	93	89	0	93 (90 on Hetzner; 4 MySQL port 3306; 2 Kubernetes certs)
Total	**2,770**	**2,695**	**2,440 unique**	**0 (0.00%)**	**2,695**

Four reasons all 2,695 endpoints stay off block lists

1. **Upstream "confirmed" labels are not enough without an independent adversary signal.** Six AuroraStealer records carried a "confirmed" status in the upstream feed. Three mapped to one unconfirmed host with FTP and shared-hosting certificate signals; its ports are withheld. The other three, observed on 13 November 2026 in Finland, the United States and Russia, carried machine certificates and JARMs. Because they lacked a documented protocol response, beacon or workbench link, their addresses, ports and certificate subjects are withheld.
2. **Database, mail and CDN certificate collisions.** Four Vidar records on Hetzner pointed to MySQL database port 3306. Four suspected AuroraStealer records mapped to one Russian server with mail-service and parked-domain signals; its ports and certificate subject are withheld. Fourteen records presented `CloudFlare Origin Certificate` (5), `Kubernetes Ingress Controller Fake Certificate` (2), `Plesk` (1) or shared hosting certificates, and 152 rows pointed to Cloudflare proxy addresses.
3. **Broadband ISP address churn and uncorroborated candidate rows.** Six RisePro records on port 8081 and two Meduza records on port 80 belonged to telecom broadband networks in the Netherlands and Russia, where dynamic addresses move between subscribers. One more RisePro candidate in Russia served an open web directory, while 2,651 candidate endpoints across the eight families lacked corroborating evidence.
4. **Vocational coursework or shared-droplet ambiguity on the US web server.** One modular suite on the US server uses an acronym that stands for Technical and Further Education institutes in Australia and several Commonwealth countries. The co-located stealer builders, C2 droppers and three Windows browser report bundles go well beyond normal classroom labs. Because they sit beside study notes, web apps and harvested Microsoft Edge SQLite files, we withhold the server's address, port, directory paths and filenames.

07

CHAPTER 07

What Defenders Should Do

IN SHORT

This chapter lists the actions defenders should take, in priority order.

07 What Defenders Should Do

Network and perimeter security

- [ ] Hunt outbound TLS traffic matching the six TLS JARM fingerprints instead of blocking unverified feed addresses, as the three machine certificate subjects and host ports are withheld.
- [ ] Enable DHCP snooping and Dynamic ARP Inspection (DAI) on internal user and server VLANs to reduce ARP spoofing, and monitor DNS responses separately.

Endpoint and browser security

- [ ] Enable Microsoft Edge Application-Bound Encryption policies and restrict access to `Login Data`, `Cookies` and `Local State` under the user profile.
- [ ] Alert whenever `python.exe`, `powershell.exe`, `pwsh.exe` or `mshta.exe` reads browser SQLite files, invokes DPAPI decryption (`CryptUnprotectData` or `os_crypt`), runs an HTML Application (`.hta`) launcher or copies browser SQLite databases outside the user profile.

Cloud, application and lab security

- [ ] Scan your own cloud instances and lab subnets across high HTTP ports (TCP 8000 to 9000) for `Directory listing for /` pages exposing `.git/`, `.venv/` or security test folders.
- [ ] Enforce server-side object-level authorization checks on web APIs so automated IDOR enumeration scripts cannot step through another user's records.

Threat intelligence and SOC

- [ ] Deploy the host-agnostic Suricata, KQL, Sigma and YARA rules in Chapter 8 using exact hostname allowlists for approved red-team and simulation hosts.
- [ ] Keep Cloudflare proxy addresses, FTP, MySQL, POP3/IMAP mail and telecom ISP matches out of automatic firewall feeds.

CHAPTER 08

Detection Rules

IN SHORT

This chapter describes each published rule and where to deploy it.

INSIDE THIS CHAPTER

How the two Suricata signatures work 24
How the endpoint and SIEM rules work 25
MITRE ATT&CK mapping 25

08 Detection Rules

The full rules are in Appendix B. Offline checks verified the Suricata file with our syntax linter (0 problems) and parsed the KQL query with Microsoft `Kusto.Language` (0 errors). Both Sigma rules passed `pySigma` validation, and all three YARA rules compiled with `YARA-X`. Because the rules were not tested on live production traffic, test and tune them in your own environment first.

Rule file	What it catches
`odin_python_stealer_workbench_and_aurorastealer.rules`	Suricata 7 (2 signatures): open HTTP directory listings exposing `.git/`, `.venv/` or `.ps1` / `.hta` / `.db` artifacts, and outbound HTTP `POST` uploads containing Chromium `os_crypt` and `encrypted_key` DPAPI data
`sentinel_python_stealer_workbench_and_aurorastealer_hunt.kql`	Microsoft Sentinel and Defender XDR: script process command lines mentioning Chromium store or stager markers on devices that also connected to TCP 8081 within seven days; the query does not verify a file read or same-process egress
`sigma_python_stealer_workbench_suites.yml`	Sigma (2 rules): Linux process command lines starting Python HTTP directory servers or raw-socket ARP/DNS spoofing constructs, and Windows `python.exe`, `powershell.exe`, `pwsh.exe` or `mshta.exe` command lines mentioning Chromium or Edge store markers
`odin_python_stealer_workbench_and_aurorastealer.yar`	YARA (3 rules): host-agnostic Python Chromium stealer and DPAPI bundles, harvested browser credential report manifests, and directory-listing manifests containing `.git/`, `site-packages/` or `.venv/` and at least two distinct `.ps1`, `.hta` or `.exe` stager links

How the two Suricata signatures work

SID	Looks for	Why
9537001	HTTP 200 `Directory listing for /` response containing links to `.git/`, `.venv/`, `venv/`, `.ps1`, `.hta` or `.db`	Open HTTP directory exposing a Git repo, Python virtual environment or staged payloads
9537006	Outbound HTTP `POST` request body containing `"os_crypt"` and `"encrypted_key"`	Chromium or Microsoft Edge `Local State` DPAPI master key material leaving the network

Because the three 13 November 2026 AuroraStealer-labelled hosts remain unconfirmed, their machine certificate subjects are withheld from Suricata rules. Defenders can retain the six JARM fingerprints in Chapter 9 for hunt-only use.

How the endpoint and SIEM rules work

The 7-day KQL query inspects Python, PowerShell and `mshta.exe` process command lines for Chromium store markers (`Login Data`, `Cookies`, `Local State`, `os_crypt`, `encrypted_key`, `CryptUnprotectData`) or `.hta` and service-stager strings. It joins those process creation events by device with `DeviceNetworkEvents` on TCP 8081 (`HighPortCount > 0`) across seven days. The query does not verify a credential-file read or same-process network egress, and it excludes approved simulation hosts by exact device name.

Sigma or YARA rule	Level or scope	Fires when
Sigma Rule 1: Python HTTP server or L2 spoofing on Linux	High	Python or shell starts `http.server` / `SimpleHTTPServer` or runs raw-socket `ARP(`, `sendp(`, `DNSQR(` or `AsyncSniffer` outside exact lab hostnames
Sigma Rule 2: Script command line mentioning Chromium store markers on Windows	High	`python.exe`, `powershell.exe`, `pwsh.exe` or `mshta.exe` command lines contain Edge or Chrome `User Data`, `Login Data`, `Local State`, `os_crypt`, `encrypted_key` or `CryptUnprotectData` strings outside exact simulation VMs (command-line match only, not a verified file read)
YARA Rules 1 to 3 (256 KB to 512 KB bounds)	File and manifest scan	A file or manifest combines Chromium `Login Data` / `Cookies` / `Local State` references with `os_crypt` / `encrypted_key` / `CryptUnprotectData`, SQLite credential schema fields with DPAPI key fields, or a `Directory listing for /` header with `.git/`, `site-packages/` or `.venv/` and at least two distinct `.ps1`, `.hta` or `.exe` stager links

MITRE ATT&CK mapping

Tactic	Technique	Observed evidence	Detection
Resource Development	T1583.003 Acquire Infrastructure: Virtual Private Server	Custom Python suites on a US web server (port withheld) and stealer feed rows on rented VPS hosts	SID 9537001; Sigma Rule 1; YARA Rule 3
Execution	T1059.001 Command and Scripting Interpreter: PowerShell	Staged PowerShell beacon and privilege-escalation scripts alongside an HTML Application launcher (paths withheld)	KQL; Sigma Rule 2; YARA Rule 3
Credential Access	T1555.003 Credentials from Web Browsers	Renamed Microsoft Edge login SQLite databases and DPAPI `Local State` JSON files in three browser report bundles (paths withheld)	SID 9537006; KQL; Sigma Rule 2; YARA Rules 1 and 2
Credential Access	T1539 Steal Web Session Cookie	Renamed Microsoft Edge cookie SQLite databases in three browser report bundles (paths withheld)	SID 9537006; KQL; YARA Rules 1 and 2
Adversary-in-the-Middle	T1557.002 ARP Cache Poisoning	Staged Python ARP and DNS spoofing modules in the modular network exploitation subfolder (paths withheld)	Sigma Rule 1; VLAN DAI, DHCP snooping and DNS response monitoring
Command and Control	T1071.001 Application Layer Protocol: Web Protocols	Python HTTP upload receivers, Flask/Socket.IO C2 stack and AuroraStealer-labelled HTTP/TLS ports	SID 9537006; KQL; JARM hunt

09

CHAPTER 09

Indicators

IN SHORT

This chapter lists the published indicators and explains what stays withheld.

INSIDE THIS CHAPTER

JARM fingerprints to hunt 27

09 Indicators

Appendix A holds 6 published indicators: 6 TLS JARM fingerprints for hunting. There are no published IPv4 addresses, machine certificate subjects or file hashes in this report. The three self-signed machine certificate subjects from the 13 November 2026 AuroraStealer-labelled endpoints are withheld (`subject withheld`) alongside their IPv4 addresses and ports, while their JARMs are retained for hunt-only use.

Type	Count	Where to use it
JARM fingerprints (in Appendix A)	6	TLS hunting only, never block on a JARM alone
Machine certificate subjects	0 (3 withheld)	Withheld (`subject withheld`) because the three hosts remain unconfirmed

JARM fingerprints to hunt

JARM	Family	Context
`2ad2ad0002ad2ad00042d42d0000002059a3b916699461c5923779b77cf06b`	AuroraStealer	Unconfirmed endpoint; host details withheld; JARM for hunting only (seen on 4 AuroraStealer endpoints)
`2ad2ad16d2ad2ad22c2ad2ad2ad2adfd9c9d14e4f4f67f94f0359f8b28f532`	AuroraStealer	Unconfirmed endpoint; host details withheld; JARM for hunting only (seen on 4 AuroraStealer endpoints)
`14d14d00014d14d08c000000000000217f0fec773ec97f0b3fdd2de2993c31`	AuroraStealer	Unconfirmed endpoint; host details withheld; JARM for hunting only
`15d3fd16d29d29d00015d3fd15d29d92faac93703c2dc5037f0c8e9572fc5b`	RisePro	Shared by 11 RisePro-labelled endpoints on ports 8081 and 443
`3fd3fd00000000000043d3fd3fd43d79451d8c63b099acafdbabb24551d0e6`	AuroraStealer, Vidar	Shared by 9 AuroraStealer and 3 Vidar feed endpoints (12 total)
`29d3fd00029d29d21c42d43d00041df61f6b5fa6973e9fa14c8155669c6633`	AuroraStealer, Meduza, AgentTesla	Shared by 11 feed endpoints across three families

We withhold the US web server's IPv4 address, service port and peer addresses in session log filenames. We also withhold 2,434 distinct unconfirmed commodity stealer IPv4 addresses (2,437 tokenized host entries). The three 13 November 2026 AuroraStealer-labelled IPv4 addresses, service ports and machine certificate subjects, exposed server paths and filenames, unverified certificate domains, file hashes and secrets are also withheld.

10

CHAPTER 10

Scope and Limits

IN SHORT

This chapter sets out what the evidence can show and what it cannot.

10 Scope and Limits

- **Observation window.** Findings cover a two-week observation window ending 20 November 2026. Dates from 8 to 14 August 2026 on the US web server are file and folder timestamps on that server, and older dates on unconfirmed feed rows are historical listing dates.
- **Analytical model, not one actor.** Zero hosts link the US Python server to the 2,440 commodity stealer host entries. Figures 1 and 3 show how custom workbenches and commodity feeds divide roles, not a shared operator.
- **Staged files are not target outcomes.** Seeing builders, droppers and three Windows browser report bundles on a server shows how the tools are structured. It does not establish where or whether those binaries were deployed externally, or whether an upload occurred over the wire.
- **Feed labels are leads.** Family names such as LokiBot, RisePro or AuroraStealer come from upstream feeds. All 2,695 endpoints, including the three AuroraStealer-labelled hosts observed on 13 November 2026, remain unconfirmed pending an independent adversary signal.
- **No victim data or offensive code.** We did not open harvested SQLite credential records, replay the operator authentication file or OAuth secrets, or reproduce runnable offensive scripts.
- **Rule validation.** Offline syntax, schema and compiler checks passed for Suricata, KQL, Sigma and YARA, but live traffic testing remains necessary in each defender's environment.

11

CHAPTER 11

Sources

IN SHORT

This chapter lists the public references used in the report.

11 Sources

1. MITRE ATT&CK. T1555.003 Credentials from Password Stores: Credentials from Web Browsers. https://attack.mitre.org/techniques/T1555/003/ (verified on 20 November 2026)
2. MITRE ATT&CK. T1539 Steal Web Session Cookie. https://attack.mitre.org/techniques/T1539/ (verified on 20 November 2026)
3. Microsoft Edge Documentation. Password Manager Security and Local State DPAPI Encryption. https://learn.microsoft.com/en-us/deployedge/microsoft-edge-security-password-manager-security (verified on 20 November 2026)
4. MITRE ATT&CK. T1059.001 Command and Scripting Interpreter: PowerShell. https://attack.mitre.org/techniques/T1059/001/ (verified on 20 November 2026)
5. MITRE ATT&CK. T1557.002 Adversary-in-the-Middle: ARP Cache Poisoning. https://attack.mitre.org/techniques/T1557/002/ (verified on 20 November 2026)
6. OWASP Cheat Sheet Series. Insecure Direct Object Reference Prevention Cheat Sheet. https://cheatsheetseries.owasp.org/cheatsheets/Insecure_Direct_Object_Reference_Prevention_Cheat_Sheet.html (verified on 20 November 2026)
7. MITRE ATT&CK. S0447 Lokibot. https://attack.mitre.org/software/S0447/ (verified on 20 November 2026)
8. Salesforce Engineering. JARM: Active Transport Layer Security Server Fingerprinting. https://github.com/salesforce/jarm (verified on 20 November 2026)

APPENDICES

Indicators and Detection Rules

FOR DEFENDERS

Appendix A lists the indicators published for this study. Appendix B carries every published detection rule in full.

INSIDE

A. Indicators of compromise (6) 33

B. Detection rules (4 files) 34

A Indicators of Compromise

Values are defanged for display. The CSV, JSON and STIX files on the online edition carry the raw values. Values the report marks as withheld are not listed.

ID	Type	Indicator	Port	Context	Action
R37-IOC-01	JARM	2ad2ad0002ad2ad00042d42d0000002059a3b916699461c5923779b77cf06b	withheld	Unconfirmed endpoint; host details withheld; JARM for hunting only	HUNT
R37-IOC-02	JARM	2ad2ad16d2ad2ad22c2ad2ad2ad2adfd9c9d14e4f4f67f94f0359f8b28f532	withheld	Unconfirmed endpoint; host details withheld; JARM for hunting only	HUNT
R37-IOC-03	JARM	14d14d00014d14d08c000000000000217f0fec773ec97f0b3fdd2de2993c31	withheld	Unconfirmed endpoint; host details withheld; JARM for hunting only	HUNT
R37-IOC-04	JARM	15d3fd16d29d29d00015d3fd15d29d92faac93703c2dc5037f0c8e9572fc5b	8081	risepro · shared by 11 RisePro-labelled endpoints on TCP 8081 and 443	HUNT
R37-IOC-05	JARM	3fd3fd00000000000043d3fd3fd43d79451d8c63b099acafdbabb24551d0e6	443	aurorastealer · shared by 9 AuroraStealer and 3 Vidar feed endpoints (12 total)	HUNT
R37-IOC-06	JARM	29d3fd00029d29d21c42d43d00041df61f6b5fa6973e9fa14c8155669c6633	443	aurorastealer · shared across AuroraStealer (8), Meduza (2) and AgentTesla (1) feed endpoints	HUNT

B Detection Rules

Each rule file loads in its own engine (Suricata 7, YARA-X, Kusto, pySigma). Tune thresholds and allow-lists to your environment before alerting or blocking. The same files can be downloaded from the online edition.

odin_python_stealer_workbench_and_aurorastealer.rules

SURICATA

```
# Odin by Brandefense - Monograph 37 / 46 Suricata 7.x Detection Suite
# Coverage: Open HTTP directory listings exposing Git or Python virtual-environment
#           artifacts, and outbound HTTP POST exfiltration of Chromium DPAPI Local State
#           keys
# Classification: TLP:CLEAR

alert http $EXTERNAL_NET any -> $HOME_NET any (msg:"ODIN [R37] Open HTTP Directory Listing Exposing Git
Virtualenv or Script Artifacts"; flow:established,to_client; http.stat_code; bsize:3; content:"200";
http.response_body; bsize:>48; content:"Directory listing for /"; fast_pattern; pcre:"/href=\x22[^\x22]*(\.git\/|
\.venv\/|venv\/|\.ps1\x22|\.hta\x22|\.db\x22)/i"; classtype:attempted-recon; sid:9537001; rev:2;
metadata:created_at 2026_11_20, deployment perimeter, tactic resource_development;)

alert http $HOME_NET any -> $EXTERNAL_NET any (msg:"ODIN [R37] Outbound HTTP POST Exfiltration of Chromium DPAPI
Local State Key"; flow:established,to_server; http.method; bsize:4; content:"POST"; http.request_body; bsize:>32;
content:"os_crypt"; fast_pattern; content:"encrypted_key"; distance:0; classtype:credential-theft; sid:9537006;
rev:2; metadata:created_at 2026_11_20, deployment perimeter_egress, tactic exfiltration;)
```

odin_python_stealer_workbench_and_aurorastealer.yar

YARA

```
/*
   Odin by Brandefense - Monograph 37 / 46 Structural YARA Suite
   Coverage: Host-agnostic detection of Python Chromium/Edge SQLite and DPAPI
             credential harvesters, harvested browser credential report manifests,
             and directory-listing manifests containing .git/, site-packages/ or .venv/
             and at least two distinct .ps1, .hta or .exe stager links
   Classification: TLP:CLEAR
*/

rule ODIN_Python_Chromium_Stealer_And_DPAPI_Bundle
{
    meta:
        author = "Odin by Brandefense"
        monograph = "37 / 46"
        date = "2026-11-20"
        tlp = "CLEAR"
        action = "hunt_python_chromium_credential_harvester"
        description = "Detects Python scripts or report manifests combining Chromium Login Data and Cookies
SQLite references with DPAPI Local State key markers"
    strings:
        $s_login = "Login Data" ascii nocase
        $s_cookies = "Cookies" ascii nocase
        $s_state = "Local State" ascii nocase
        $k_oscrypt = "os_crypt" ascii
        $k_enckey = "encrypted_key" ascii
        $k_dpapi = "CryptUnprotectData" ascii
    condition:
        filesize < 256KB and 2 of ($s_*) and 2 of ($k_*)
}

rule ODIN_Harvested_Chromium_SQLite_And_LocalState_Manifest
{
    meta:
        author = "Odin by Brandefense"
        monograph = "37 / 46"
        date = "2026-11-20"
        tlp = "CLEAR"
```

```
        action = "hunt_harvested_browser_credential_manifest"
        description = "Detects browser credential report manifests pairing Chromium SQLite schema fields with
DPAPI Local State key fields"
    strings:
        $db_pw = "password_value" ascii
        $db_ck = "encrypted_value" ascii
        $db_url = "origin_url" ascii
        $ls_os = "os_crypt" ascii
        $ls_ek = "encrypted_key" ascii
    condition:
        filesize < 256KB and 2 of ($db_*) and all of ($ls_*)
}

rule ODIN_Open_Directory_Listing_Exposing_Git_Venv_And_Stagers
{
    meta:
        author = "Odin by Brandefense"
        monograph = "37 / 46"
        date = "2026-11-20"
        tlp = "CLEAR"
        action = "hunt_open_directory_offensive_workspace"
        description = "Detects HTTP directory-listing manifests containing .git/, site-packages/ or .venv/
folders and at least two distinct .ps1, .hta or .exe stager links"
    strings:
        $dir = "Directory listing for /" ascii
        $env1 = ".git/" ascii
        $env2 = "site-packages/" ascii
        $env3 = ".venv/" ascii
        $stg1 = ".ps1\"" ascii nocase
        $stg2 = ".hta\"" ascii nocase
        $stg3 = ".exe\"" ascii nocase
    condition:
        filesize < 512KB and $dir and 1 of ($env*) and 2 of ($stg*)
}
```

sentinel_python_stealer_workbench_and_aurorastealer_hunt.kql

KQL (SENTINEL / DEFENDER XDR)

```
// Odin by Brandefense - Monograph 37 / 46 Microsoft Sentinel & Defender XDR Hunt
// Correlates script process command lines mentioning Microsoft Edge or Chromium store markers
// (Login Data, Cookies, Local State, DPAPI os_crypt) or HTA and service-stager strings
// on devices that also connected to TCP 8081 within seven days (device-level join;
// does not verify a file read or same-process network egress)
// Classification: TLP:CLEAR

let Lookback = 7d;
let ApprovedRedTeamHosts = dynamic(["REDTEAM-BUILD-VPS-01", "OFFSEC-LAB-NODE-01"]);
let ApprovedSimHosts = dynamic(["PURPLE-WIN-SIM-01", "ATOMIC-TEST-VM-01"]);
let BrowserCredMarkers = dynamic([
    "Login Data", "Cookies", "Local State", "os_crypt", "encrypted_key", "CryptUnprotectData"
]);
let StagerMarkers = dynamic([
    "mshta.exe", ".hta", "New-Service", "sc.exe create", "FromBase64String"
]);
let HostScriptProcs = DeviceProcessEvents
    | where Timestamp >= ago(Lookback)
    | where not(DeviceName in~ (ApprovedRedTeamHosts)) and not(DeviceName in~ (ApprovedSimHosts))
    | where FileName in~ ("python", "python3", "python.exe", "powershell.exe", "pwsh.exe", "mshta.exe")
    | where ProcessCommandLine has_any (BrowserCredMarkers) or ProcessCommandLine has_any (StagerMarkers)
    | project ProcTime = Timestamp, DeviceId, DeviceName, InitiatingProcessFileName, FileName, FolderPath,
ProcessCommandLine;
DeviceNetworkEvents
| where Timestamp >= ago(Lookback)
| where not(DeviceName in~ (ApprovedRedTeamHosts)) and not(DeviceName in~ (ApprovedSimHosts))
| where RemotePort == 8081
| summarize
    NetFirstSeen = min(Timestamp),
    NetLastSeen = max(Timestamp),
    DistinctRemoteIps = dcount(RemoteIP),
```

```
    HighPortCount = countif(RemotePort == 8081),
    SampleEndpoints = make_set(strcat(RemoteIP, ":", tostring(RemotePort)), 12),
    SampleUrls = make_set(RemoteUrl, 10),
    ConnectionCount = count()
    by DeviceId, DeviceName, InitiatingProcessFileName
| join kind=inner (HostScriptProcs) on DeviceId, DeviceName
| extend ResolvedDevice = coalesce(DeviceName, DeviceName1)
| where HighPortCount > 0 and isnotempty(ProcessCommandLine)
| extend AlertAction = case(
    ProcessCommandLine has_any (BrowserCredMarkers), "HUNT_CHROMIUM_CREDENTIAL_ACCESS_WITH_HIGHPORT_EGRESS",
    "HUNT_SCRIPT_STAGER_WITH_HIGHPORT_EGRESS"
)
| project NetFirstSeen, NetLastSeen, ProcTime, ResolvedDevice, AlertAction, InitiatingProcessFileName, FileName,
FolderPath, ProcessCommandLine, SampleEndpoints, SampleUrls, ConnectionCount, HighPortCount
| sort by coalesce(ProcTime, NetLastSeen) desc
```

sigma_python_stealer_workbench_suites.yml

SIGMA

```
title: Python HTTP Directory Server or Raw-Socket ARP and DNS Spoofing on Linux
id: 6d4e31a2-8c7b-4f19-a512-9037a1b2c301
status: experimental
description: Detects Linux Python or shell execution starting an unauthenticated HTTP directory server on high
ports or invoking raw-socket ARP and DNS spoofing constructs outside approved lab hosts.
author: Odin by Brandefense
date: 2026-11-20
modified: 2026-11-22
references:
    - https://attack.mitre.org/techniques/T1583/003/
    - https://attack.mitre.org/techniques/T1557/002/
logsource:
    category: process_creation
    product: linux
detection:
    selection_interp:
        Image|endswith:
            - '/python'
            - '/python3'
            - '/python3.12'
            - '/bash'
            - '/sh'
    selection_cli:
        CommandLine|contains:
            - 'http.server'
            - 'SimpleHTTPServer'
            - 'sendp('
            - 'ARP('
            - 'DNSQR('
            - 'AsyncSniffer'
    filter_approved_lab:
        ComputerName:
            - 'REDTEAM-BUILD-VPS-01'
            - 'OFFSEC-LAB-NODE-01'
    condition: selection_interp and selection_cli and not 1 of filter_*
falsepositives:
    - Authorized offensive security lab servers explicitly listed in filter_approved_lab
level: high
tags:
    - attack.resource-development
    - attack.t1583.003
    - attack.adversary-in-the-middle
    - attack.t1557.002
---
title: Non-Browser Script Command Line Mentioning Chromium Credential Store Markers on Windows
id: 6d4e31a2-8c7b-4f19-a512-9037a1b2c302
status: experimental
description: Detects Windows process creation where Python, PowerShell, or MSHTA command lines contain Microsoft
Edge or Chromium User Data, Login Data, or DPAPI Local State markers outside approved simulation hosts; inspects
command-line strings rather than file-read events.
```

```
author: Odin by Brandefense
date: 2026-11-20
modified: 2026-11-22
references:
    - https://attack.mitre.org/techniques/T1555/003/
    - https://attack.mitre.org/techniques/T1059/001/
logsource:
    category: process_creation
    product: windows
detection:
    selection_interp:
        Image|endswith:
            - '\python.exe'
            - '\python3.exe'
            - '\powershell.exe'
            - '\pwsh.exe'
            - '\mshta.exe'
    selection_stores:
        CommandLine|contains:
            - '\Microsoft\Edge\User Data\'
            - '\Google\Chrome\User Data\'
            - 'Login Data'
            - 'Local State'
            - 'os_crypt'
            - 'encrypted_key'
            - 'CryptUnprotectData'
    filter_approved_sim:
        ComputerName:
            - 'PURPLE-WIN-SIM-01'
            - 'ATOMIC-TEST-VM-01'
    condition: selection_interp and selection_stores and not 1 of filter_*
falsepositives:
    - Dedicated purple-team endpoint simulation hosts listed by exact ComputerName
level: high
tags:
    - attack.credential-access
    - attack.t1555.003
    - attack.execution
    - attack.t1059.001
```

ODIN

About Odin Research

Odin Research is the public research series of Odin by Brandefense. It documents adversary infrastructure seen on the internet and turns each finding into indicators and detection rules that defenders can use. Victims and third parties stay anonymous, addresses appear only for confirmed adversary infrastructure, and no working attack code is published.

Odin is the agentic analyst inside the Brandefense threat intelligence platform, which unifies Cyber Threat Intelligence, Digital Risk Protection and External Attack Surface Management.

Online edition, indicators and rules · brandefense.io/odin-research/reports/python-stealers-and-aurorastealer-c2/
Short Read · brandefense.io/odin-research/blog/custom-python-stealers-vs-commodity-c2-feeds/
Defense Kit · brandefense.io/odin-research/defense-kit/