



ChocoShell Malware Analysis Report

Analyst	Turki Abdullah Almuraykhi
Analysis period	28 September - 5 October 2026
Environment	Isolated Windows virtual machine; disposable local accounts
Primary sample	PowerShell script, 55,877 bytes, 970 lines
SHA-256	BE99857449D2856DD5A84E21C8A3D5E0E01456ADB44062DDEC5A6B4970D8D42C
Report scope	Behavior, credential-access capability, privilege elevation, C2 staging, exfiltration design, and detection opportunities

Handling note: The real command-and-control infrastructure was not contacted intentionally. Dynamic testing was conducted with network isolation and an explicit C2 block.





1. Executive Summary

ChocoShell is a PowerShell-based credential-stealing component associated by Microsoft Threat Intelligence with the CaptiveCrunch campaign. The exact SHA-256 analyzed in this report is publicly listed by Microsoft as ChocoShell. Microsoft also identifies 213.145.86.112 as ChocoShell command-and-control infrastructure. This report treats that campaign attribution as external CTI context; the analyst did not independently attribute the operator.

Static analysis revealed a staged architecture centered on privilege elevation, defense evasion, browser and session-material collection, Microsoft TokenBroker artifact collection, Wi-Fi credential collection, host reconnaissance, and compressed exfiltration. The analyzed script contains both silent local-administrator elevation branches and a visible RunAs fallback. It also contains a separate Chrome DevTools Protocol path capable of requesting plaintext browser cookie values from a locally launched browser.

Controlled dynamic analysis established two valid behavioral baselines. Run 1, performed as a standard user, reached a visible UAC request that was denied. Run 2B, performed as a split-token local administrator, revealed an implementation flaw in the script's local-administrator detection: Windows reported the Administrators SID as deny-only, while the .NET Groups.Value check used by the sample returned false. As a result, the sample misclassified the account as a genuine non-administrator, selected its RunAs branch, and spawned a High-integrity PowerShell process through the Windows UAC/AppInfo flow.

The elevated process executed the embedded encoded stub and terminated within the same second. No C2-delivered follow-on ScriptBlock, browser process relaunch, TokenBroker access, Wi-Fi collection process, or audited read of the instrumented browser credential files was observed. The most likely explanation is that the isolated environment prevented the follow-on stage retrieval; however, the exact WinHTTP exception was not captured, so this remains an inference rather than a confirmed fact.

Finding	Assessment
Sample identity	Exact sample hash matches Microsoft's published ChocoShell IOC.
Privilege elevation	Dynamically confirmed: Medium PowerShell ->UAC/AppInfo -> High PowerShell encoded stub.
Credential capability	Confirmed in source: browser databases/keys, CDP cookies, TokenBroker .tbres, Wi-Fi profiles.
Credential theft in Run 2B	Not observed. Instrumented credential files generated zero matching Security 4663 events.
C2 staging	C2 URLs and WinHTTP logic confirmed in source; elevated stub invocation confirmed. Successful remote retrieval not confirmed.
Credential theft in Run 2B	Source confirms JSON -> UTF-8 -> GZIP -> Base64 -> {"gz":"..."} -> POST /t/event.



2. Analytic Method and Evidence Language

This report distinguishes between source-code capability, lab-observed behavior, external reporting, and analyst judgment. Confidence levels are assigned to analytic judgments and inferences, not to direct facts or observations.

Label	Use in this report
FACT	Directly present in the analyzed artifact or a cited public source.
OBSERVATION	Directly observed in the controlled lab telemetry.
CONFIRMED FINDING	Supported by converging direct evidence, such as source plus process/ event telemetry.
SUPPORTED CLAIM	Plausible and materially supported, but not independently complete.
INFERENCE	Analyst interpretation derived from evidence. Confidence is stated as LOW/MEDIUM/HIGH.
UNVERIFIED CLAIM	Statement found in code comments or another source but not independently validated.

2.1 Source Reliability

Source	Role	Reliability treatment
Analyzed PowerShell sample	Primary technical artifact	Direct evidence of implemented code paths and embedded values; does not prove every branch executed.
Windows/Sysmon/ PowerShell/Security logs	Primary dynamic evidence	Direct evidence for the instrumented execution window; interpretation is constrained to captured telemetry.
Microsoft Threat Intelligence	External primary CTI	Used for public campaign attribution and published ChocoShell IOC corroboration.
Developer comments inside the sample	Embedded claims/context	Not treated as factual merely because they appear in comments; corroborated separately where possible.



3. Sample Identification and Public CTI Context

Attribute	Value
File type	PowerShell script
Size	55,877 bytes
Line count	970
SHA-256	BE99857449D2856DD5A84E21C8A3D5E0E01456ADB44062DDEC5A6B4970D8D42C
Analyst-decoded elevated stub SHA-256	A133565A912C59D3D8FE577D44C574B73B5B4363A5BEFFCB488423F05375865F

External context. Microsoft Threat Intelligence lists the same sample SHA-256 as ChocoShell and describes ChocoShell as the PowerShell infostealer used in CaptiveCrunch. Microsoft attributes CaptiveCrunch to Storm-2945, which it assesses to be an operational sub-cluster of Midnight Blizzard, and identifies 213.145.86.112 as ChocoShell C2. Microsoft notes TTP similarities with Forest Blizzard activity but maintains the CaptiveCrunch attribution to Storm-2945. [1]

Attribution boundary. This report does not independently attribute the sample to a state actor. The malware-to-campaign and campaign-to-actor relationships are based on cited external assessments. The analyst findings below focus on the behavior and implementation of the specific sample hash.

4. Execution Architecture and Staging

The sample is a multi-stage PowerShell stealer rather than a single-purpose script. Static analysis identified a main script that handles initial C2 status communication, defense-evasion setup, elevation decisions, credential-collection logic, data packaging, and stage/tool retrieval. An embedded elevation stub is decoded from Base64 text, re-encoded as UTF-16LE Base64, and launched through PowerShell -EncodedCommand when elevation is required.

- Main PowerShell sample**
- > environment / privilege checks
 - > elevation decision
 - > elevated -EncodedCommand stub
 - > C2 status beacon
 - > follow-on stage/tool retrieval
 - > credential / session collection
 - > compressed exfiltration

FACT: The main sample contains a hard-coded base URL of https://213.145.86.112 and uses WinHttp.WinHttpRequest.5.1 for network operations. WinHTTP Option(4) is set to 13056, configuring SSL error-ignore behavior.



FACT: The embedded elevated stub contains a status request to /t/pixel.gif?m=stub_start and a follow-on retrieval path /cdn/chunks/vendor-9f3a1c.min.js?stub=1, followed by in-memory execution of the retrieved content.

FACT: The main stage also references /cdn/chunks/polyfill-7e2b.min.js for additional tooling and /t/event for data submission.

5. Defense Evasion and Anti-Analysis

5.1 AMSI State Manipulation

The sample contains reflection-based logic targeting System.Management.Automation.AmsiUtils and the internal amsiInitFailed field. The elevated stub contains the same type of logic. This is an implemented attempt to alter AMSI behavior before further PowerShell content executes. The analysis does not establish that AMSI was successfully disabled in the lab, because the internal AMSI state was not independently instrumented.

5.2 Encoded and In-Memory Staging

The elevation stub is stored as Base64-encoded ASCII source, decoded at runtime, and then re-encoded as UTF-16LE Base64 for PowerShell -EncodedCommand. Follow-on content is designed to be decoded and executed as an in-memory ScriptBlock. This reduces reliance on a conventional executable payload and complicates simple file-based controls.

5.3 Timing-Based VM Check

The main source contains a timing check around a 2.5-second sleep and can return early if the measured delay is below a threshold. This is a source-level anti-analysis capability. The lab did not separately instrument the measured timing variable, so whether the VM check triggered during either valid run remains unconfirmed.

6. Privilege Elevation

Privilege elevation is central to the sample’s design, as several higher-impact capabilities require an elevated or SYSTEM-equivalent security context. The source implements different execution paths for administrators and genuine non-administrators.

Source branch	Implemented mechanism	Dynamic status
Split-token/local administrator	SilentCleanup windir hijack; wsreset.exe AppX-class hijack; sdclt.exe /KickOffElev folder-shell hijack	Present in source. Not observed executing in Run 2B because the local-admin detection check returned false in this environment.
Genuine non-administrator / fallback	Start-Process powershell -Verb RunAs with hidden non-interactive -EncodedCommand stub	Observed in Run 1 and Run 2B. Run 1 prompt denied; Run 2B produced a High-integrity child.

6.1 Split-Token Local Administrator Detection Flaw

Run 2B revealed an environment-specific implementation flaw in the local-administrator test. The account ChocoAdmin was a member of BUILTIN\Administrators, and whoami /groups displayed SID S-1-5-32-544 as “Group used for deny only” in the Medium-integrity token. However, the exact .NET expression used by the sample — `WindowsIdentity.GetCurrent().Groups.Value -contains S-1-5-32-544` — evaluated to false. `IsInRole(Administrator)` also returned false, as expected for the filtered Medium token.

CONFIRMED FINDING: Because both checks evaluated false, ChocoShell classified the split-token local administrator as a genuine non-administrator and selected its RunAs/UAC branch rather than the silent local-administrator branches.

7. Credential and Session-Material Collection

Credential access is the most consequential capability in the analyzed source. The implementation supports browser encryption-key recovery, raw browser database collection, plaintext-cookie retrieval through CDP, TokenBroker artifact collection, and Wi-Fi profile extraction. An important distinction is that these capabilities are evident in the source code, but Run 2B did not reach the collection stage.

7.1 Chromium Local State and Master-Key Handling

The Derive-MasterKey logic reads the browser Local State file and first checks `os_crypt.app_bound_encrypted_key`. The value is Base64-decoded, the APPB prefix is checked, and the protected blob is passed through a SYSTEM-context decryption path when needed. The source references `Invoke-Impersonate-System` for SYSTEM token impersonation and then calls `ProtectedData.Unprotect`. It subsequently reverts impersonation and attempts a second DPAPI layer in the user context.

For Edge/Brave/Opera/Vivaldi-style 32-byte content, the source can return the content as an ABE key and optionally derive the legacy `encrypted_key` as a DPAPI fallback. For the Chrome flag-3 path, the source references `Parse-ChromeKeyBlob`, NCrypt-based decryption, and AES-GCM handling. Several helper functions are referenced by the main script but are not defined in the analyzed file; the source is designed to obtain additional tooling before these paths are used.

7.2 Cookies and Login Data

For each Chromium profile, the collection function identifies `Network\Cookies` and copies the database into a temporary file. The result entry contains the raw cookie database encoded as Base64, along with recovered key material and key type. Login Data is copied and Base64-encoded when present, but only after the Cookies path succeeds for that profile.

IMPORTANT LIMITATION: No record-level SQL parsing of Chromium `password_value` or `encrypted_value` fields was identified in this analyzed stage. The stage packages the raw Login Data database and key material; this report does not assert where downstream record-level password extraction occurs.



7.3 Chrome DevTools Protocol Cookie Path

A separate path launches supported browsers with a remote debugging port and uses the Chrome DevTools Protocol. Hard-coded ports are Chrome 19876, Edge 19877, and Brave 19878. The source queries /json targets, establishes a WebSocket connection, and sends Network.getAllCookies. Returned cookie records include host, name, value, path, expiration, Secure, HttpOnly, and SameSite fields.

FACT: This path is designed to retrieve plaintext cookie values directly from the browser and therefore does not depend on decrypting every cookie record offline. Microsoft publicly describes the same general CDP behavior for ChocoShell. [1]

7.4 Firefox-Family Browsers

The source targets Firefox, Waterfox, LibreWolf, Floorp, and Zen profile structures. It copies cookies.sqlite and stores the raw database bytes as Base64. At this stage, ChocoShell copies the SQLite databases without parsing individual records.

7.5 Microsoft TokenBroker

The script enumerates %LOCALAPPDATA%\Microsoft\TokenBroker\Cache for *.tbres files and stores each file name, size, raw Base64 content, and modification time in the result object. The lab did not parse the semantic contents of individual .tbres blobs. The confirmed capability is therefore the collection of raw TokenBroker artifacts. Microsoft’s public ChocoShell analysis describes the operator objective as theft of Microsoft 365/identity session material. [1]

7.6 Wi-Fi Profiles

The source executes netsh wlan show profiles, extracts saved profile names, and requests each profile with key=clear. It parses the Key Content field and stores SSID/password pairs in the result object. This capability was not reached dynamically in Run 2B.

8. Command-and-Control and Data Exfiltration

Artifact	Role	Evidence status
213.145.86.112	Hard-coded HTTPS C2	Present in sample; independently listed by Microsoft as ChocoShell C2. [1]
/t/pixel.gif?m=<status>	Status/beacon path	Present in sample; public Microsoft reporting corroborates this pattern. [1]
/cdn/chunks/polyfill-7e2b.min.js	Additional tooling retrieval	Present in sample; public Microsoft reporting corroborates this path. [1]
/cdn/chunks/vendor-9f3a1c.min.js?stub=1	Elevated stub follow-on retrieval	Present in analyst-decoded stub; not independently observed returning content.
/t/event	Exfiltration endpoint	Present in sample; public Microsoft reporting corroborates this path. [1]



The result structure contains categories for browsers, firefox, cdp_cookies, tokens, wifi, and system. The final data path serializes the result as compact JSON, converts it to UTF-8, compresses it using GZIP, Base64-encodes the compressed bytes, wraps the value in a JSON envelope, and submits the data via a WinHTTP POST request to /t/event.

- Result object**
- > ConvertTo-Json -Compress
 - > UTF-8 bytes
 - > GZIP
 - > Base64
 - > {"gz":"<base64>"}
 - > WinHTTP POST /t/event

UNVERIFIED CLAIM: Source-code comments reference nginx/VPS body-size limits and backend implementation details. These claims were not independently verified and should not be treated as confirmed infrastructure characteristics.

9. Controlled Dynamic Analysis

9.1 Safety Controls

- Disposable Windows virtual machine with snapshots before and after test phases.
- No IPv4 default route during the instrumented runs; IPv6 default route also removed before Run 2B.
- Explicit outbound firewall block for 213.145.86.112.
- No intentional connection, ping, or reachability test against the real C2.
- Run 2B used Sysmon, PowerShell Script Block Logging, Process Creation auditing, File System auditing, and Security 4663 SACL instrumentation on selected browser/TokenBroker targets.

9.2 Run 1 - Standard User / UAC Denied

Item	Observed value
Account	ChocoLab - standard user
Primary PID	7488
Integrity	Medium
Elevation behavior	Visible UAC/RunAs request; denied by analyst
Credential access	No matching TokenBroker, browser credential, or CDP activity attributed to PID 7488
Network activity	No TCP/UDP activity attributed to PID 7488 in the analyzed Procmon data
Outcome	Primary process exited; elevated chain did not execute



9.3 Run 2A - Invalid Test Excluded from Findings

An intermediate test was retained for documentation but excluded from the behavioral findings. Sysmon recorded the spawned PowerShell command line as -File "", indicating that the sample path variable was not defined in that PowerShell session. As a result, the sample was not executed, and this run is not used as evidence of malware behavior.

9.4 Run 2B - Split-Token Local Administrator

Time (local)	PID	Integrity	Event status
16:06:13	8432	Medium	Original ChocoShell sample launched under ChocoAdmin.
16:06:14	2492	SYSTEM	consent.exe created by svchost.exe AppInfo service, consistent with UAC elevation processing.
16:06:19	2952	High	PowerShell child of PID 8432 launched with -NonI - W H -EncodedCommand.
16:06:19	2952	High	PowerShell Event 4104 captured the decoded elevated stub command body.
16:06:19	2952	High	Process terminated in the same second.

CONFIRMED FINDING: Run 2B produced a Medium-to-High privilege transition through the Windows RunAs/UAC/AppInfo path and invoked the encoded ChocoShell elevation stub at High integrity.

OBSERVATION: Security Event 4663 queries for the instrumented Local State, Login Data, Network\Cookies, TokenBroker, and .tbres targets returned zero matching credential-access events during the Run 2B execution window.

OBSERVATION: No msedge, chrome, brave, netsh, taskkill, vssadmin, or wmic processes associated with the collection stage were observed in the Run 2B process window. No C2-delivered follow-on ScriptBlock appeared after the elevated stub.

INFERENCE - HIGH: The elevated stub most likely terminated during the uncaught WinHTTP follow-on stage retrieval. This is consistent with the lab’s network isolation and firewall restrictions, as well as the immediate PowerShell error-formatting activity after the stub. Because the exact exception text was not captured, the failure point remains an inference rather than a confirmed finding.



10. Findings Assessment

ID	Classification	Finding	Evidence
CF-01	CONFIRMED FINDING	The analyzed SHA-256 matches the ChocoShell sample publicly listed by Microsoft.	Sample hash + Microsoft IOC table [1]
CF-02	CONFIRMED FINDING	The sample implements a PowerShell RunAs elevation path, which was dynamically observed during testing.	Source branch + Run 1/Run 2B process telemetry
CF-03	CONFIRMED FINDING	Run 2B spawned the encoded ChocoShell stub at High integrity after UAC/AppInfo processing.	Sysmon process chain + PowerShell 4104
CF-04	CONFIRMED FINDING	The analyzed source implements browser/credential/session collection and compressed exfiltration.	Direct source analysis
SC-01	SUPPORTED CLAIM	Run 2B did not reach observable credential collection.	0 matching 4663 events + no collection-related child processes
IN-01	INFERENCE - HIGH	The elevated stub most likely terminated during blocked follow-on C2 retrieval.	Immediate termination + isolated network + error-formatting scriptblocks; exact exception absent

11. Detection and Hunting Opportunities

Behavioral correlations provide more resilient detection opportunities than isolated strings. The following opportunities are based on the analyzed source and lab telemetry; capability-only branches should be validated in the target environment before production alerting.

Behavior	Hunting idea	Status
PowerShell elevation	Medium-integrity powershell.exe launching powershell.exe with -Verb RunAs or an elevated child containing -EncodedCommand.	Observed
UAC/AppInfo sequence	consent.exe from Appinfo near a suspicious PowerShell process, followed by High-integrity PowerShell.	Observed
ChocoShell C2 pattern	PowerShell/WinHTTP access to 213.145.86.112 with /t/pixel.gif?m= or /t/event.	Source + external CTI
SilentCleanup branch	HKCU\Environment\windir modification followed by DiskCleanup\SilentCleanup and cleanup of the value.	Capability only in this lab



CDP cookie path	Browser launch with --remote-debugging-port=19876/19877/19878 followed by localhost CDP/WebSocket access.	Capability only in this lab
Credential file reads	PowerShell reads of browser Local State, Network\Cookies, Login Data, or TokenBroker\Cache*.tbres.	Capability; instrumented but not observed in Run 2B

11.1 Analyst Detection Artifacts

Artifact	Validation state
ChocoShell_v1.yar	Logic-validated by the analyst; 0 false positives across 258 benign PowerShell/module files in the analyst test corpus. No dedicated YARA engine validation was available in the recorded test.
ChocoShell_CDP_Sigma_v1.yml	Draft behavioral Sigma rule; not parser-validated in the recorded test.
Defender KQL / Splunk correlation	Analyst-developed hunting logic; should be treated as a draft until tested against production telemetry schemas.

12. MITRE ATT&CK Mapping

The mappings below reflect capabilities implemented in the analyzed sample and behaviors observed during testing.

“Capability” indicates that the code path is present but was not necessarily executed during the controlled run.

Technique	ID	Basis	Status
PowerShell	T1059.001	Primary execution and staged PowerShell commands	Observed
Obfuscated/Compressed Files and Information	T1027	Base64/EncodedCommand and encoded stage content	Observed/capability
Deobfuscate/Decode Files or Information	T1140	Runtime Base64 decode and ScriptBlock creation	Capability
Impair Defenses	T1562.001	AMSI state manipulation; Defender-related tooling references	Capability
Bypass User Account Control	T1548.002	SilentCleanup/wsreset/sdclt local-admin branches	Capability
Credentials from Web Browsers	T1555.003	Chromium/Firefox browser databases and key material	Capability



Steal Web Session Cookie	T1539	CDP Network.getAllCookies path	Capability
Access Token Manipulation: Token Impersonation/Theft	T1134.001	SYSTEM token impersonation helper referenced for ABE path	Capability
Scheduled Task/Job: Scheduled Task	T1053.005	SilentCleanup trigger and transient browser-launch tasks	Capability
Application Layer Protocol: Web Protocols	T1071.001	HTTPS/WinHTTP C2	Capability
Exfiltration Over C2 Channel	T1041	POST /t/event	Capability
File Deletion	T1070.004	Temporary elevation/collection artifact cleanup	Capability
Time Based Evasion	T1497.003	Sleep-delta VM/sandbox check	Capability

13. Evidence Register

Artifact	SHA-256 / note
Original sample	BE99857449D2856DD5A84E21C8A3D5E0E01456ADB44062DDEC5A6B4970D8D42C
Analyst-decoded elevated stub	A133565A912C59D3D8FE577D44C574B73B5B4363A5BEFFCB488423F05375865F
Run 1 Procmon PML	D86C8F553BEE6471DB4F5717566A8A2D12433A48AA85A35F8F48AD783726E1BE
Run 1 filtered Procmon CSV	BEEBBDAFA2D21AFA4457A67F8505B9029EECE3A3EE8423EFB073DC68EE0CB4E8
Run 2 Sysmon configuration	9BE0212D51AA1452BBD2A5CFD0A9B07E3243BFD56FC8E6B14ADA6DF230A8E940
Run 2B Sysmon log	run2B_sysmon.evtx - hash not reproduced in this report because it was not recorded in the supplied evidence transcript.
Run 2B PowerShell log	run2B_powershell.evtx - hash not reproduced in this report because it was not recorded in the supplied evidence transcript.
Run 2B Task Scheduler log	run2B_taskscheduler.evtx - hash not reproduced in this report because it was not recorded in the supplied evidence transcript.
Run 2B Security log	run2B_security.evtx - hash not reproduced in this report because it was not recorded in the supplied evidence transcript.
Run 2B final assessment	311D031EF9C496CAA9A4B595D87FF444E80D4949308949E26CC9B4C7F83F0DA8

14. Scope, Limitations, and What Was Not Proven

- The real C2 was not intentionally contacted. The lab was isolated, and the identified C2 IP was explicitly blocked.
- The follow-on content expected from /cdn/chunks/vendor-9f3a1c.min.js?stub=1 was not retrieved from the real infrastructure.
- Credential-theft capability is strongly supported by source analysis, but Run 2B did not produce dynamic evidence of browser/TokenBroker/Wi-Fi collection.
- TokenBroker .tbres files were not semantically parsed in the lab; raw artifact collection is confirmed in source, while token semantics are described in Microsoft's public reporting.
- No record-level Chromium password parsing was identified in the analyzed stage. Raw Login Data and key material are packaged for downstream use.
- AMSI manipulation logic is present, but successful AMSI disablement was not independently instrumented.
- The exact exception that terminated PID 2952 was not captured. The assessment that network retrieval failed therefore remains a HIGH-confidence inference.
- Campaign attribution is based on Microsoft's public reporting and was not independently established by the analyst.

Conclusion

The analyzed ChocoShell sample is a credential-focused, staged PowerShell stealer designed with privilege elevation and defense evasion in mind. The source demonstrates considerable attention to modern browser credential protections, including Application-Bound Encryption handling, SYSTEM-context operations, raw browser database collection, and a parallel CDP method for retrieving plaintext cookie values. TokenBroker and Wi-Fi collection extend the potential impact of a successful infection beyond browser passwords.

Dynamic testing confirmed the privilege-elevation mechanism without contacting the real C2. The most significant lab result was the split-token administrator detection flaw: in the tested Windows environment, ChocoShell failed to recognize a deny-only Administrators SID through its .NET group enumeration and therefore selected the visible RunAs path. The resulting UAC/AppInfo sequence produced a High-integrity PowerShell process that invoked the encoded elevated stub.

The lab did not observe successful follow-on stage delivery or credential collection. This does not mean those capabilities are absent; rather, it establishes the limits of what was dynamically confirmed. The final assessment therefore separates implemented credential-theft capability from observed execution and avoids attributing unobserved behavior to the sample.

References

[1] Microsoft Threat Intelligence. “CaptiveCrunch: Midnight Blizzard targets travelers worldwide for malware delivery and credential theft.” Microsoft Security Blog, 31 July 2026. [Microsoft Security Blog](#)

Analyst evidence: ChocoShell sample source, Run 1 Procmon/Sysmon evidence, Run 2B Sysmon/PowerShell/Security/Task Scheduler logs, decoded elevation stub, and Run 2B final assessment artifact.