

Fake GlobalProtect MSI Targets Myanmar Using Cloudflare and Google Sheets as C2

By Malware INFO Research Team

Published: 2026-09-02 · Archived: 2026-09-08 02:01:21 UTC

Executive summary

An organization received an email containing a link to a Google Sites landing page that presented a download as Palo Alto Networks GlobalProtect. To protect the organization and its personnel, this article does not reproduce the email, identify the recipient, or publish the landing page. Our analysis begins with the downloaded Windows Installer package.

The supplied `GlobalProtect.msi` was unsigned and contained an unsigned 64-bit executable named `GlobalProtect.exe`. Static analysis exposed a staged design: query a public IP-geolocation service, compare the returned country with `Myanmar`, contact a Cloudflare Worker configuration endpoint, obtain fields shaped like Google service-account credentials and a spreadsheet identifier, and use Google Sheets API structures as a command-and-output channel.

Controlled runtime evidence confirmed important parts of that sequence. The executable contacted `ip-api.com`, a Cloudflare address, and Google infrastructure. Packet capture retained a clear-text request to `ip-api.com/json` with the `SheetsBot/1.0` user agent and a response identifying the lab connection as Myanmar. Process Monitor recorded installation and execution activity, including local DLL writes and loads. An earlier full process-memory dump retained a constructed Worker request and the numbered Worker-host family. In a later, separately identified debugger session, `InternetReadFile` exposed the plaintext Worker response: it contained an actual spreadsheet identifier and a complete Google service-account credential object. Three memory captures retained that configuration, and two retained a signed RS256 JWT bearer assertion targeting Google's OAuth endpoint.

In a subsequent execution/capture, debugger and full-memory evidence confirmed a successful OAuth response with a Bearer access token, a read from `Sheet1!A1`, a successful update to `Sheet1!B1`, and polling of `Sheet1!C1`. The retained command-poll response contained no command value. C2 channel establishment and polling were observed, but operator tasking, command execution, output collection, `Sheet1!D1` result submission, and broader data exfiltration remain unobserved.

The sample shares Myanmar victim relevance with publicly reported HoneyMyte/CoolClient activity, but its delivery, code, execution chain, and infrastructure are different. We found no evidence sufficient to identify this sample as CoolClient, connect it to HoneyMyte, or attribute it to another named actor.

Safety warning: Unknown installers and executables belong only in an authorized, disposable, isolated analysis VM containing no trusted credentials, sensitive data, shared folders, or access to a trusted network. A VM is not automatically a secure sandbox. Do not contact suspected live C2 infrastructure unless the activity is explicitly authorized, risk-assessed, monitored, and isolated.

Evidence model

This investigation correlated eight kinds of local evidence:

- static inspection of the MSI and embedded executable;
- Windows Installer metadata and package tables;
- Process Monitor host activity from installation and direct execution;

- packet captures from controlled VM sessions;
- MalwareINFO Deep API Trace, Process Memory Capture, and Payload & Memory Analyzer exports;
- debugger observations of library activity, process exit, and a plaintext application response buffer;
- three identity-bound full-memory captures from the configuration-delivery session; and
- three additional identity-bound captures from a subsequent OAuth and Sheets session.

The sources came from separate executions. Process IDs and timestamps are not merged into a fictional single timeline. We use the following evidence labels throughout the article:

Label	Meaning
Observed	A retained runtime source recorded the event or data in the identified session.
Debugger observed	A debugger exposed plaintext application data at an identified API boundary; this is not equivalent to decrypted PCAP content.
Runtime-memory observed	Data existed in a mapped runtime allocation or request buffer; this does not by itself prove transmission or remote acceptance.
Static capability	Code, API names, strings, or package configuration support a capability, but do not prove execution.
Inferred	Multiple observations support an explanation, but a decisive event or decrypted transaction is unavailable.
Not proven	The available material cannot establish the claim.

This distinction matters particularly for memory strings. A full process dump includes application code, loaded libraries, heap data, operating-system components, and stale allocations. A string or API name in the dump may be relevant, but it is not automatically attributable to an executed sample function.

Delivery: trusted branding on a public web platform

The private incident report describes a link delivered by email and a Google Sites page styled as a GlobalProtect portal. We intentionally omit the message, sender and recipient details. At technique level, the delivery is consistent with [MITRE ATT&CK T1566.002, Spearphishing Link](#): a link takes the recipient to web content that prompts a download rather than attaching the payload directly.

Google Sites is a legitimate service, and its appearance in a delivery chain does not make the platform malicious. The defensive lesson is to validate the actual publisher, signature and approved software-distribution route instead of trusting a familiar logo or hosting domain.

Palo Alto Networks describes GlobalProtect as secure-access infrastructure built around GlobalProtect apps, portals and gateways. The supplied package instead installed a standalone executable that performed geolocation, configuration-gate and command-channel operations. This mismatch, combined with the absent signatures and deceptive naming, supports [T1036.005, Match Legitimate Resource Name or Location](#).

Sample identity

Artifact	Size	SHA-256	Signature state
GlobalProtect.msi	3,475,968 bytes	a124caa58d956c7430ecb8772a3794266235917670c5b56564342bd1456897e7	No Authenticod signature present
Embedded GlobalProtect.exe	425,984 bytes	33d696728101c9caf6ebb215ba176bbb94e5cc16218b574029b622fd6e3bee8c	No Authenticod signature present

The executable is a native PE32+ x64 Windows GUI image. MalwareINFO's disk-image profile parsed six sections and eight import modules, found no CLR header, no TLS directory, and no certificate table. Compile and MSI metadata timestamps are useful for chronology, but timestamps can be modified and are not attribution evidence.

The MSI advertised GlobalProtect VPN version 11.5.0 with manufacturer text PaloAlto. Those fields are attacker-controlled package metadata. They must not be treated as vendor verification.

MSI package analysis

The package contained five principal files: the main executable, copies named CRYPT32.dll, bcrypt.dll, and WININET.dll, and a branding image. The installer invoked:

```
GlobalProtect.exe /Install
```

The retained installation trace recorded PID 3604 running that command. It wrote a 146,944 -byte bcrypt.dll and a 5,039,616 -byte WININET.dll into the application directory, then loaded them. The local module names are significant because Windows may search the application directory when resolving DLLs. Microsoft's [DLL search-order documentation](#) explains that the application folder participates in module resolution, while its [DLL security guidance](#) warns that control of a searched directory can enable DLL preloading or binary planting.

That general Windows behavior does not, by itself, prove that these particular DLLs were malicious. The retained evidence shows that the files were written and loaded from the application folder. Their contents and signer identities require separate evaluation; defenders should not block common Windows DLL names globally.

Package and host evidence also identified two GlobalProtectVPN RunOnce values—one under HKCU and one under HKLM—and a scheduled task named GlobalProtectVPNUpdate. Microsoft's [Run and RunOnce documentation](#) confirms that RunOnce entries request one-time execution at logon. The behaviors align with [T1547.001, Registry Run Keys / Startup Folder](#) and [T1053.005, Scheduled Task](#).

Windows Installer is legitimate infrastructure. ATT&CK documents adversarial use of it under [T1218.007, Msiexec](#), but an MSI alone is not suspicious. The useful detection is the correlated chain: an unapproved or unsigned MSI, deceptive product identity, an unusual installation path, unexpected local DLLs, autostart configuration, and immediate network activity.

The package tables add several reproducible pivots. ProductCode is {49CB5141-CEC5-4324-BB91-E2D843CD4DE1} and UpgradeCode is {BF73E425-31E9-4F43-AFA3-B928117CAEA2}. ALLUSERS=2 asks Windows Installer to use a per-machine context when privileges permit, subject to Windows Installer fallback behavior. The target directory resolves to C:\Program

Files\PaloAlto\GlobalProtect VPN , and the executable custom action is scheduled late in the installation sequence. These values describe this package; they do not authenticate its claimed publisher.

Extracted file inventory

Offline extraction recovered the following files. No executable or DLL is attached to this article.

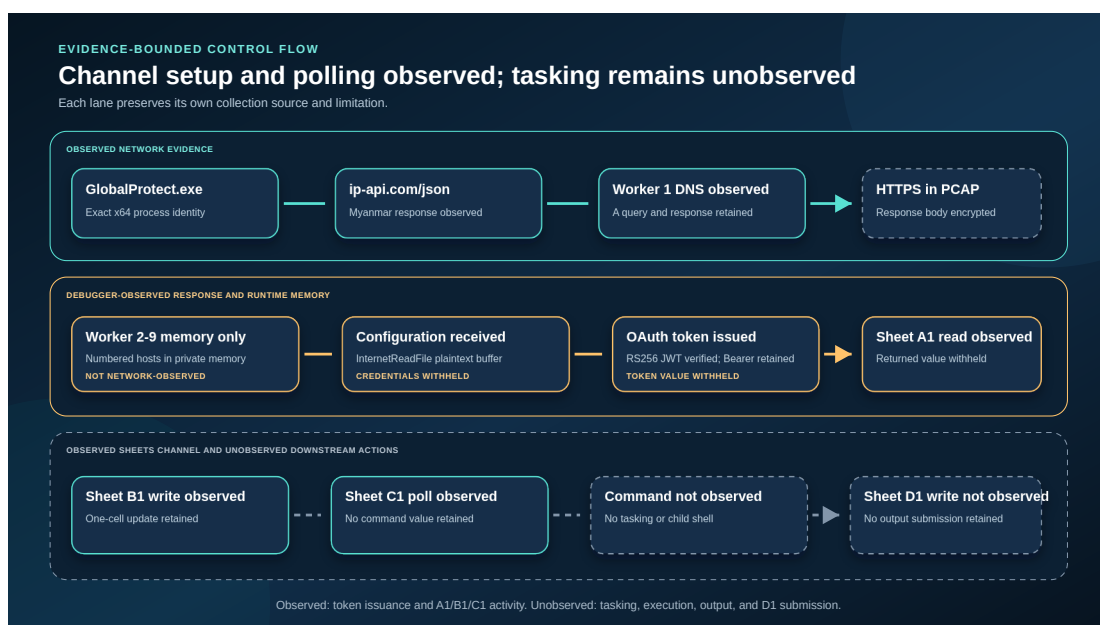
File	Size	SHA-256	Review result
GlobalProtect.exe	425,984 bytes	33d696728101c9caf6ebb215ba176bbb94e5cc16218b574029b622fd6e3bee8c	Unsigned native PE32+ x64 GUI executable; no CLR header
bcrypt.dll	146,944 bytes	811af66dd7e323a74db80648610bbeb956093ae7bd91c5b950ed6aca13d8026	Microsoft version metadata and a valid embedded signature in the extracted file
CRYPT32.dll	1,412,152 bytes	c692af3f8741669cd1b517c461e80bce19ad6e8608c81066cde9786f17be0099	Microsoft version metadata and a valid embedded signature in the extracted file
WININET.dll	5,039,616 bytes	948589c349b0205ff73c7049b0d2e5c4faaffb64271c64ee0033e384f61cf51d6	Microsoft version metadata; no embedded signature detected by the local check
paloalto.png	4,636 bytes	8a680fcc160c352ed2d9b96654e50bf0f5f504141828aa50b75b970f46f3f170	Branding resource

A validly signed supporting DLL does not make the enclosing MSI trustworthy. Conversely, an absent embedded signature on a copied Windows component is not sufficient to call that component malicious; catalog signing and file provenance require separate checks. The evidence of interest is the package's combined file set, placement, execution and network behavior.

Reverse-engineered control flow

Static control-flow analysis supports this sequence:

```
MSI installation -> GlobalProtect.exe /Install -> files and startup artifacts
Normal launch -> persistent device ID -> public-IP geolocation
Myanmar match -> Cloudflare configuration gate -> Google authorization design
Spreadsheet polling -> command-execution capability -> result-writing capability
```



Evidence-bounded flow showing observed Cloudflare configuration, OAuth token issuance, Google Sheets reads and writes, and downstream actions that remain unobserved

This is a map of implemented logic and evidence, not a claim that every stage completed. The geolocation exchange and Worker 1 DNS resolution were **Observed**. The configuration and successful OAuth responses were **Debugger observed**; credential retention, signed-JWT construction, token availability and the A1/B1/C1 operations were **Runtime-memory observed**. Command delivery, command execution and D1 result submission remain **Not observed**.

Encrypted configuration

The executable stores several operational constants as encrypted blobs. Offline analysis reproduced the program's AES-256-CBC decryption behavior and valid PKCS#7 padding without contacting any external service. Recovered values identify the geolocation host, the install directory, the Cloudflare gate, the gate request-header name and the Windows command-shell prefix.

The configuration decryption input is published because it is a static program constant rather than a live service credential:

AmountOfFreeDiskSpace . Direct disassembly shows the executable opening the SHA256 algorithm, hashing that input and using the 32-byte SHA-256(input) result as the AES-256 key. The first 16 bytes of each stored blob supply the CBC

initialization vector; the remaining bytes are ciphertext with PKCS#7 padding. This documents how the embedded configuration was recovered without publishing the derived key as an additional copy-ready artifact.

Property	Reverse-engineered value
Key-derivation input	AmountOfFreeDiskSpace
Key derivation	SHA-256(input)
Cipher	AES-256-CBC
Padding	PKCS#7

The boundary is deliberate: the configuration decryption input is published, while the live gate-authentication secret remains withheld. The request-header name `X-Bot-Secret` is useful for detection, but its value could facilitate replay. The later plaintext response strongly supports the header's role as authentication material used to access the configuration gate, but it does not reveal the Worker's server-side validation logic or prove the identity of whoever possessed the value.

Encryption here is local configuration obfuscation, not attribution evidence and not proof that the HTTPS stages succeeded. Both the encrypted data and its recovery logic are present in the executable; TLS separately protected the captured Worker and Google traffic in transit.

Myanmar activation guardrail

Static strings described a public-IP lookup and a comparison with `Myanmar`. Runtime traffic then supplied the missing behavioral link. The packet capture recorded:

```
GET /json HTTP/1.1
Host: ip-api.com
User-Agent: SheetsBot/1.0
```

The response contained `country: Myanmar` and `countryCode: MM`. Precise IP address, location and ISP values are intentionally omitted because they describe the analyst's environment and are not necessary for detection.

The corresponding runtime Process Monitor session began at `18:57:42.925` and exited with status `0` at `18:57:45.785`, a lifetime of approximately 2.86 seconds. During that interval, PID `7100` connected to `ip-api.com:80`, a Cloudflare address on TCP/443, and Google-owned infrastructure on TCP/443. The installation session independently showed the same broad destination sequence.

This is stronger than finding the word `Myanmar` in a binary. The clear request, returned country value, memory-resident country markers, and subsequent Cloudflare/Google connections support a Myanmar activation or targeting guardrail. The public-IP lookup is consistent with [T1016.001, Internet Connection Discovery](#). We use “execution guardrail” descriptively; ATT&CK's narrower [T1480.001, Environmental Keying](#) requires cryptographic environmental keying, which this evidence does not establish.

`ip-api.com` is a legitimate shared service. It is contextual evidence, not a domain that organizations should block globally without considering business use and the originating process.

Persistent device identifier

Reverse engineering shows that the executable attempts to read a locally persisted identifier and generates one when the expected value is unavailable. The generator uses an alphanumeric character set and includes the marker `-tk-`. Later

command-channel logic reuses that identifier, giving the design a stable way to associate polling and result records with one installation. The marker was present both in the mapped executable image and in writable private regions of the full process dump, providing runtime-memory corroboration of the identifier-building path.

The actual identifier from the analysis VM is not published. The retained evidence does not justify a public claim about every character, field length or lifecycle edge case. Defenders can instead correlate the uncommon marker, a newly created local identifier artifact, the executable hash and the surrounding network sequence. Finding the marker alone does not prove that an operator session occurred.

Cloudflare Worker configuration gate

The retained evidence identified a numbered Worker family:

```
hxxps://sheets-config-gate.hewlett-pack{1..9}[.]workers[.]dev/config0
```

All nine numbered hostnames occurred in writable private memory. The dump retained a coherent `/config0` request construction for `hewlett-pack3`, including the `SheetsBot/1.0` user agent and a populated `X-Bot-Secret` header. It also retained a less complete `hewlett-pack9` request remnant. The header value is intentionally withheld. These memory objects show that request material was constructed in the process; they do not establish that either request was transmitted or accepted.

Only `hewlett-pack1` was observed in DNS. Each retained PCAP contained an A-record query and response for `sheets-config-gate.hewlett-pack1[.]workers[.]dev`, followed by Cloudflare-connected traffic in the relevant sequence. Hosts 2 through 9 remain runtime-memory observations, not network observations. Direct executable inspection contained the common `.workers.dev` suffix and request artifacts, but not nine literal complete hostnames; the numbered family is therefore reported from the runtime dump rather than mislabelled as nine static strings.

Associated program strings described accepting or rejecting gate access and parsing fields named `google_creds`, `spreadsheet_id`, `client_email`, and `private_key`. The runtime PCAP did not expose the encrypted HTTPS request or response body.

Cloudflare's [Workers documentation](#) explains that a request to a `*.workers.dev` subdomain invokes user-defined Worker code on Cloudflare's network. That architecture makes a Worker suitable as a small configuration or redirection layer, but the platform remains a legitimate service used by many applications.

Because the HTTPS response was encrypted and no session keys were retained, the PCAP does not reveal the Worker response. A later debugger session supplies a different evidence source: plaintext application data returned through `InternetReadFile`. We did not retrieve or replay the secret or interact with the endpoint outside the recorded lab executions.

Full-memory evidence narrows the stopping point

The dump was parsed offline as a Windows minidump and each relevant hit was mapped back to its virtual-memory region. This separates constant data mapped from `GlobalProtect.exe` from transient or copied data in private process memory.

Memory finding	Location class	What it supports	What it does not prove
/config0 , SheetsBot/1.0 , and a populated X-Bot-Secret request construction associated with hewlett-pack3	Writable private memory	Request material existed in a runtime allocation	Network transmission, server receipt, or acceptance
Numbered hewlett-pack1 through hewlett-pack9 hostnames	Writable private memory	The process materialized the numbered host family	That every host was contacted
hewlett-pack1 DNS query and response	Observed network evidence	This exact numbered host was resolved during retained sessions	HTTPS application content or a successful gate response
Gate-success diagnostic	Mapped executable image only	The binary implements a success branch	That the success branch ran
Rejection or blocked-channel diagnostic	Mapped executable image and Writable private memory	A runtime allocation retained rejection-path text	Whether the server rejected the request, a channel was blocked, or the text was merely prepared
-tk- device marker	Mapped executable image and Writable private memory	Runtime-memory corroboration of identifier construction	A complete public-safe device identity or operator activity

For that earlier capture alone, the strongest defensible stopping point remained after geolocation and Worker request preparation. Its private rejection diagnostic was not decisive proof of a server outcome. The later debugger and memory evidence described next supersedes that negative boundary without turning the separate executions into one fictional timeline.

Debugger-confirmed Google configuration delivery

The new evidence comes from a separate execution of the same x64 executable identity. The debugger targeted PID 4408 ; three MalwareINFO Process Memory Capture manifests bind the captures to the same process-creation identity, executable path, architecture and executable SHA-256 33d696728101c9caf6ebb215ba176bbb94e5cc16218b574029b622fd6e3bee8c . Each capture completed with Complete coverage and an independently matching dump hash.

Capture	Start time (UTC)	Point-in-time evidence
Dump 1	2026-09-02 20:23:29	Worker configuration fields, spreadsheet identifier and complete service-account credential material in writable private memory; no compact JWT recovered
Dump 2	2026-09-02 20:30:33	Credential remnants plus a signed RS256 JWT and constructed JWT bearer-grant request material

Capture	Start time (UTC)	Point-in-time evidence
Dump 3	2026-09-02 20:36:58	The same signed JWT and request-construction remnants persisted; no usable OAuth access token was recovered

At `wininet!InternetReadFile`, the debugger showed a plaintext response buffer containing `spreadsheet_id`, `google_creds`, `project_id`, `private_key_id`, `private_key`, `client_email` and `client_id`. The same credential-shaped values appeared in writable private memory. This changes the Worker response from **Inferred** to **Debugger observed** and confirms that the process received Google configuration material. It does not mean the PCAP was decrypted: the network capture still contains encrypted HTTPS payloads.

A complete Google service-account private key was present in the debugger-observed configuration response. The key is intentionally withheld. The exact spreadsheet identifier, credential key identifier, gate secret and signed JWT are also withheld because they are unnecessary to reproduce the analytical conclusion and could expose or facilitate use of live resources. The debugger screenshots are not published because opaque redaction would remove much of their evidentiary context; the article instead reports the bounded findings.

Google service-account OAuth

Static analysis first recovered a coherent service-account authorization design: JWT fields including `alg`, `RS256`, `iss`, `scope`, `aud`, `iat` and `exp`; the Google Sheets OAuth scope; RSA signing and key-import paths; the JWT bearer grant; `access_token` parsing; and construction of a Bearer authorization header. The later memory captures elevate part of that design to runtime evidence.

Google's [server-to-server OAuth documentation](#) describes the legitimate model in which a service account signs an assertion, exchanges it for an access token and calls an authorized API. The sample appears designed to obtain credential material through the preceding configuration stage and apply that model to Sheets.

One compact JWT recovered from Dumps 2 and 3 decoded to the following public-safe claims. The complete signed token is intentionally withheld.

JWT field	Runtime-observed value
<code>alg</code> / <code>typ</code>	<code>RS256</code> / <code>JWT</code>
<code>iss</code>	<code>adminproject@elliptical-tree-505904-p7.iam.gserviceaccount.com</code>
<code>aud</code>	<code>https://oauth2.googleapis.com/token</code>
<code>scope</code>	<code>https://www.googleapis.com/auth/spreadsheets</code>
<code>iat</code> / <code>exp</code>	<code>1788380849</code> / <code>1788384449</code>
Assertion lifetime	3,600 seconds (approximately one hour)

Offline verification confirmed that the RS256 signature matched the private key recovered from Dump 1. Runtime memory also retained the JWT bearer-grant type, an `assertion` parameter and form-encoded request material. The malware therefore progressed beyond merely implementing service-account OAuth: it constructed a signed assertion and built the token-request body at runtime.

A subsequent execution materially advances that boundary. A debugger transcript retained response metadata identifying `token_type` as `Bearer` and `expires_in` as `3599`, while three identity-bound full-memory captures retained populated access-token material in writable private memory. One capture also retained the populated token in Bearer form. Together, these artifacts support a **Debugger observed** successful OAuth response and **Runtime-memory observed** token availability. The token itself is intentionally withheld and was not used for independent access.

The reported lifetime was approximately one hour. This confirms token issuance in that later execution; it does not establish who controlled the Google project, grant authorization beyond the captured process, or prove every subsequent API operation.

Google Sheets C2 becomes operational

In a subsequent execution/capture, three completed Full User-Mode Dumps identify PID `2272` as the same x64 executable path and SHA-256, with a consistent process-creation identity across the capture set. This is a separate run from the earlier PID `7268` and PID `4408` evidence and is not merged into either timeline.

The retained writable private memory contains request- and response-shaped Google Sheets API material that advances several protocol stages from static capability to observed runtime evidence:

Operation	Evidence class	What the evidence supports	What it does not support
Read <code>Sheet1!A1</code>	Runtime-memory observed	A request for the exact cell and a response-shaped object containing a value	Publication of the returned device value or proof of how the row was originally registered
Update <code>Sheet1!B1</code>	Runtime-memory observed	A response reports <code>updatedRange</code> <code>Sheet1!B1</code> with one updated row, column and cell	Publication of the victim/lab metadata or proof of any broader collection
Poll <code>Sheet1!C1</code>	Runtime-memory observed	Repeated request material and a response for <code>Sheet1!C1</code>	The retained response contained no <code>values</code> member and therefore no observed operator command
Write <code>Sheet1!D1</code>	Not observed	The binary still contains the static result-channel design	No successful D1 request or update response was recovered

The exact spreadsheet identifier, returned A1 value, access token and any analyst or device identifiers remain withheld. Generic command-shell strings in mapped images or memory are not counted as command execution. Runtime evidence confirms successful OAuth token issuance, Google Sheets access, a read from `Sheet1!A1`, a successful update to `Sheet1!B1`, and polling of `Sheet1!C1`. The retained command-poll response contained no command value, so operator tasking, command execution, output collection, and `Sheet1!D1` result submission remain unobserved. In short: **C2 channel establishment and polling were observed; operator tasking and downstream execution were not.**

Recovered Google Cloud infrastructure

The debugger-observed configuration exposed identifiers for Google Cloud infrastructure used by the malware. These values do not identify the human operator and do not establish whether the underlying project was attacker-created, compromised or reused.

Type	Public CTI pivot	Evidence boundary
Google Cloud project ID	elliptical-tree-505904-p7	Received in the plaintext configuration response
Service-account identifier	adminproject@elliptical-tree-505904-p7.iam.gserviceaccount.com	Google service-account identity used by the malware; not a victim or end-user email address
Google client ID	101714939433347726433	Numeric client identifier associated with the recovered service-account object; not an access token or victim ID
Private-key ID	Recovered; exact value withheld	Credential identifier retained internally
Spreadsheet ID	Recovered; exact value withheld	Active resource identifier retained internally
Private key	Complete material recovered; withheld	Reusable credential material is not published

Google Sheets command protocol

The executable and full memory dump contained a coherent set of Google API artifacts:

- oauth2.googleapis.com and an OAuth token path;
- sheets.googleapis.com and /v4/spreadsheets ;
- client_email , private_key , access_token , and Bearer ;
- the Google Sheets OAuth scope;
- spreadsheet ranges and command/output column markers; and
- command-shell and pipe-related strings.

Google documents that [server-to-server OAuth](#) can use a service account to request an access token for Google APIs. Its [Sheets API values.get reference](#) shows that reading a range requires a spreadsheet ID, a range and an authorized request. Google also documents APIs for [reading and writing cell values](#).

The sample's artifacts fit that legitimate API model, but appear designed to repurpose it. Direct binary inspection ties the four embedded ranges to these roles:

Range	Reverse-engineered role	Direction	Current evidence and public boundary
Sheet1!A	Victim/device identifier	Target -> Sheet / lookup	Code scans rows 1-100 for an existing identifier and contains a val- new-record marker. A later capture observed Sheet1!A1 read material and a returned value; the value is withheld.
Sheet1!B	Public IP / victim metadata	Target -> Sheet	A later response reported a successful one-cell update to Sheet1!B1 . The analysis VM's metadata is withheld.
Sheet1!C	Operator command queue	Sheet -> target	Later memory observed polling of Sheet1!C1 ; the retained response contained no command value.
Sheet1!D	Command output/result	Target -> Sheet	The result path combines an out- marker with collected process output. No successful Sheet1!D1 write was observed.

At a high level, the implemented protocol is: identify or create a device row in column A, associate its public IP in column B, poll column C for command material, and place a result in column D. Markers such as `command-`, `out-`, `val-`, `/values/` and `valueInputOption=USER_ENTERED` first established this as a reverse-engineered **Static capability**. The subsequent A1 read, B1 update and C1 poll now provide **Runtime-memory observed** corroboration for those specific operations.

The retained runtime evidence still did not observe `cmd.exe` as a child, a command line supplied by an operator, returned command output or a successful D1 result write. The correct claim is therefore: **the sample authenticated to Google and established its Sheets-based channel, but the retained C1 response contained no tasking and no downstream command transaction was demonstrated.**

Blocking all `workers.dev`, `oauth2.googleapis.com`, or `sheets.googleapis.com` traffic would disrupt legitimate activity. Detection should correlate the exact rare hostname and URI with the unsigned executable, `SheetsBot/1.0`, geolocation request, installation path, persistence artifacts, and process ancestry.

Prior use of Google Sheets as command and control

Google Sheets is a legitimate collaboration service, but its API can also be repurposed as a structured message store. A client that possesses valid authorization can read and update predictable cells without contacting a conventional command server. That design can make malicious requests appear alongside ordinary access to Google infrastructure, although Google API traffic is not inherently benign and is not “network invisible.” Several publicly documented malware families used this general model before the Fake GlobalProtect sample analyzed here.

In August 2024, [Proofpoint documented Voldemort](#), a C backdoor delivered through a multi-stage chain that ended in DLL side-loading. Its decrypted configuration contained an OAuth client ID, client secret and refresh token. Voldemort exchanged those values for an access token, located available cells, uploaded host information and read operator commands from Google Sheets. Proofpoint also described victim-specific pages named from host and user information. The backdoor supported operations including directory listing, file transfer, command execution, sleep and exit. This report is a strong precedent for a malware operator treating a spreadsheet as both a task queue and a result store, but it does not connect the Fake GlobalProtect sample to Voldemort or its attributed operator.

In January 2026, [Zscaler ThreatLabz reported SHEETCREEP](#), a lightweight C# backdoor used in the Sheet Attack campaign. SHEETCREEP decrypted an embedded configuration containing Google Cloud credentials and a Sheet identifier, generated a victim identifier and established a victim-specific area in the workbook. It then polled for new commands every three seconds, passed them to a hidden `cmd.exe` process and wrote encrypted, Base64-encoded output back to the sheet. The same research recorded fallback configuration locations in Firebase and Google Cloud Storage. This is relevant as an example of repeated polling and bidirectional command/result exchange through Sheets; its configuration format, encryption and execution chain are not evidence about this sample.

In July 2026, [Seqrite described SheetAgent in Operation ShadowRecruit](#). That .NET RAT carried hardcoded Google service-account material, including an RSA private key, and created a Sheets service with access to Google Sheets and Drive. Seqrite characterized Sheets as a backup C2 channel and documented a fixed column layout: computer identity, timestamp, commands, command output, public IP and agent status occupied defined fields. The malware searched for an existing computer name or registered the next available row, then reused that row for command and result exchange. The resemblance to a row-oriented protocol is analytically useful, but shared use of columns, service accounts or Google APIs does not establish common code, infrastructure or authorship.

Also in 2026, [Google Threat Intelligence Group and Mandiant reported GRIDTIDE](#), a C-based Linux backdoor used in intrusions attributed by Google to UNC2814. GRIDTIDE used the Google Sheets API as a high-availability communications channel for shell commands and raw data transfer, placing spreadsheet operations inside traffic to a widely used cloud

service. Google's disruption included terminating attacker-controlled Cloud projects and revoking the API access used for C2. GRIDTIDE broadens the precedent beyond Windows and .NET malware: the underlying technique is service abuse, not a platform-specific malware architecture.

Across these reports, the recurring design choices are authorization to an attacker-accessible cloud resource, deterministic victim or task placement, periodic reads for instructions and writes for status or results. The implementations differ in language, credential model, workbook organization, encryption, infection chain and operating system. Subsequent runtime evidence confirms that this sample authenticated and interacted with its configured Sheet, including A1 read, B1 update and C1 polling activity. It still does **not** show that an operator issued a command, that a command ran, or that command output left the host. This comparison does not establish a relationship between this sample and Voldemort, SHEETCREEP, SheetAgent, GRIDTIDE or any associated threat actor.

What remains unobserved

The earlier full dump contained field names, endpoint fragments, OAuth vocabulary, Sheets ranges and command-protocol markers. Most were located in the mapped executable image and therefore remained static implementation evidence. Later evidence progressively confirmed configuration delivery, signed-assertion construction, token issuance, an A1 read, a B1 update and C1 polling.

The final boundary is now narrower. The retained C1 response had no command value. No operator-issued command, child command shell, command output, successful D1 result write or broader exfiltration payload was recovered. Static command names, process APIs and shell strings are not counted as executed behavior.

This is a point-in-time boundary. An empty retained poll does not prove that the Sheet never contained a command before or after capture, and it does not prove inactivity. It supports only the claim that channel establishment and polling were observed while operator tasking and downstream command activity were not observed in the retained evidence.

Command execution and output capture

Disassembly connects command data to a Windows command-shell prefix and a process-creation helper. That helper configures an inherited pipe for both stdout and stderr, starts the child with the `CREATE_NO_WINDOW` flag, reads bounded output and waits for completion before closing handles. This supports a general-purpose command-and-result capability consistent with [T1059.003, Windows Command Shell](#) and cloud-service abuse consistent with [T1102, Web Service](#).

We intentionally omit a copy-ready reconstruction of the command protocol. More importantly, the retained runtime evidence did not observe a child command shell, an operator-supplied command, captured output or a completed D1 result write. Implemented capability is not the same as an observed operator action.

Persistence

Package and host evidence identify `GlobalProtectVPN` `RunOnce` values under both HKCU and HKLM, referencing the installed executable. Microsoft's [Run and RunOnce documentation](#) explains that `RunOnce` values request one-time execution at logon.

Separate code and host artifacts identify a scheduled task named `GlobalProtectVPNUpdate` with PaloAlto-themed metadata and an action referencing `GlobalProtect.exe`. Task Scheduler COM use and the recovered task strings support the implementation, but the exact principal, run level and every trigger property were not fully established and are not invented here.

These behaviors align with [T1547.001, Registry Run Keys / Startup Folder](#) and [T1053.005, Scheduled Task](#). A registry value or task name is not a verdict by itself; verify its referenced path, hash, signer, creator, action and timestamps.

Build and packaging intelligence

The executable header timestamp corresponds to `2026-08-19 06:08:59 UTC`. The MSI Summary Information “last saved” value corresponds to `2026-08-19 06:09:17 UTC` —18 seconds later. That tight interval is consistent with the executable being packaged immediately after the recorded build time.

Both timestamps are attacker-controlled and may be forged. They do not prove the real compilation time, build host, actor location or delivery date. No PDB path or trustworthy executable product-version resource was recovered, so compiler and developer attribution are not asserted. The combination of timestamps, MSI GUIDs, install path, task/value names, user agent and cloud protocol markers is more useful as a clustering hypothesis than any single field.

Dynamic corroboration

The runtime sources corroborate substantial but still incomplete parts of the recovered control flow. The PCAP's clearest evidence is the unencrypted geolocation request and Myanmar response, followed by target-process connections to Cloudflare and Google-owned infrastructure. TLS prevented network-layer inspection of the latter application data. Separate debugger and memory sessions exposed the plaintext Worker response, OAuth construction and successful token response, followed by A1 read, B1 update and C1 poll artifacts. Those sources establish channel setup and polling, but not operator tasking, command execution, output collection or D1 result submission.

Background network traffic is not C2

Debugger captures showed `WinHttpConnect` activity involving `c.pki.goog`, and the case retained a file named `37BFu49bNWI.cr1`. Hash and content inspection identified it as a Google Trust Services certificate revocation list. The call stack passed through Windows cryptographic networking components rather than establishing the investigated command channel.

This is a useful analytical control. A hostname appearing inside a suspicious process does not automatically belong to the attacker. Certificate validation, proxy discovery, operating-system services, and loaded dependencies can generate background traffic. In this case, `c.pki.goog` is excluded from the C2 assessment.

MalwareINFO evidence workflow

What Deep API Trace observed

A separate MalwareINFO Deep API Trace session targeted the exact x64 executable hash. It ran for approximately 12.9 seconds with the matching-bitness native collector and retained 524 events. The target exited and the evidence package was finalized.

The trace captured exact allowlisted API metadata including memory allocation, protection and write operations, along with lower-level I/O activity and WinHTTP library loading. It did not retain a complete high-level HTTP request containing the investigated hostname, and it did not establish an allocate-write-protect-execute chain attributable to injection.

Coverage was explicitly incomplete: 44 events were dropped, the collector reported a final-drain gap, Sysmon was unavailable, and no Guard Network or Guard Registry records were retained for the target. The session also produced no memory snapshot—its terminal evidence state recorded `DeepLifecycleTerminalCaptureUnavailable` with zero captured artifacts.

Accordingly, Deep API Trace contributes an identity-bound API timeline and diagnostics. It does not supply the full dump used later in this article, prove every static capability, or establish a complete behavior history. For a broader explanation, see [Understanding Malware INFO Deep API Trace](#).

From live process capture to reviewable memory evidence

MalwareINFO Process Memory Capture was used separately to preserve the running process. It produced a completed Full User-Mode Dump of the x64 `GlobalProtect.exe` process, PID `7268`. The capture manifest bound the artifact to the process creation identity, executable path, architecture, file identity and exact executable SHA-256.

The supplied evidence archive had SHA-256 `f4531d6c4f7d675fa297672ad7fd82fa49d943e360e078bcf0bb65318d59dec8` and contained one file, `memory-full.dmp`. After extraction, the dump identity was independently recomputed before analysis rather than trusted from its filename.

The dump was `71,815,859` bytes with SHA-256:

```
4f12bac08422ba0f7a16c30b94572953f6a9640403f043b7c3bf3838a0c5cafa
```

This binding records which process was captured and reduces the risk of treating evidence from another execution as evidence from this one. It does not, by itself, establish maliciousness. Process memory can also contain credentials, documents, decrypted content, tokens and other sensitive data; the dump must be handled as confidential evidence and must not be shared publicly or with unauthorized parties. See [Process Memory Capture: Preserving and Analyzing Live-Process Evidence](#) for the operational workflow and privacy boundary.

MalwareINFO Payload & Memory Analyzer then examined the dump locally under a static-only safety contract. It did not execute the dump or copy the raw artifact into the report package. The resulting package contained `77,474` normalized evidence records across six manifest-hashed chunks, retained `50,000` bounded strings, and produced eight capability findings for analyst review.

The minidump inventory reported 56 loaded images, 12 threads and 618 memory regions. Of those regions, 435 were committed, 114 were executable and nine were private executable regions. These are navigation and prioritization measurements. Private executable memory is not automatic proof of shellcode, process injection, or malicious execution.

The retained strings included the investigated `ip-api.com` URL, Cloudflare Worker path, Google OAuth/Sheets artifacts, and command-shell terms. The Analyzer grouped related artifacts into hypotheses including network communication, persistence, command execution, cryptographic operations, and process-memory modification potential. These findings are leads, not proof that every capability ran. The package recorded zero runtime-observed API calls.

The report also preserved its limitations. Exact API-call-to-dump mapping was disabled because the imported artifact lacked the complete compound session binding required for that correlation. `GapAffectedMemory` coverage was unavailable, disk-to-memory comparison was not applicable, the dump remained “runtime context only,” and `finalCisoEligible` was `false`. This is why the article does not turn a large evidence count into a complete-behavior claim.

Together, the two MalwareINFO workflows demonstrate a practical evidence pipeline: Process Memory Capture preserves a hash-bound dump from a selected process, while Payload & Memory Analyzer turns that large artifact into searchable strings, memory-region inventory, capability hypotheses, evidence relationships, diagnostics and integrity-verifiable chunks. The value is structured analyst review—not an automatic malware verdict. For further detail, see [From Captured Memory to Defensible Evidence](#).

Recovered attacker configuration

This summary deliberately separates recovered constants from unobserved outcomes. Secrets, credentials, analyst identifiers and victim information remain withheld.

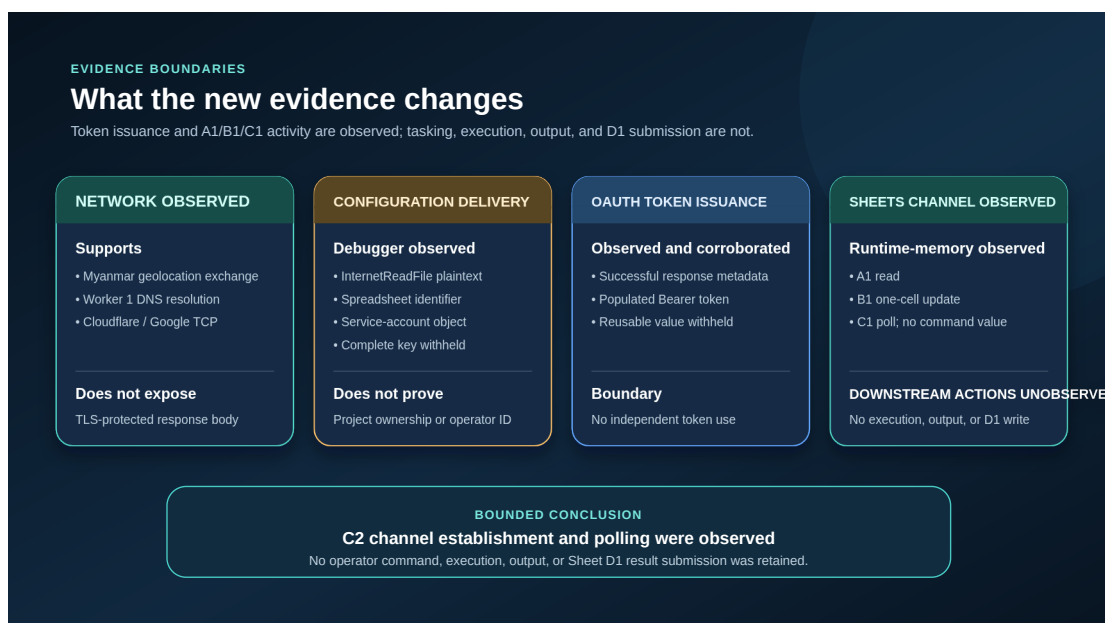
Item	Public-safe result	Evidence class	Boundary
Package identity	GlobalProtect VPN 11.5.0 / PaloAlto	Static capability	Attacker-controlled metadata, not vendor verification
Install directory	C:\Program Files\PaloAlto\GlobalProtect VPN	Static capability and host corroboration	Case-specific path
Geolocation	ip-api.com/json , SheetsBot/1.0 , Myanmar condition	Observed	Shared service; analyst IP omitted
Worker gate	numbered hewlett-pack1 through hewlett-pack9 family and /config0	Runtime memory; hewlett-pack1 DNS-observed	Hosts 2-9 were not network-observed; PCAP response body remained encrypted
Gate request	SheetsBot/1.0 and populated X-Bot-Secret ; value withheld	Runtime-memory observed	Header is strongly supported as gate-authentication material; server logic remains unknown
Configuration response	google_creds , spreadsheet identifier and service-account fields	Debugger observed via InternetReadFile	Reusable credentials and resource identifiers withheld
Google Cloud project	elliptical-tree-505904-p7	Debugger observed	Usage does not prove ownership or account provenance
Service account	adminproject@elliptical-tree-505904-p7.iam.gserviceaccount.com	Debugger observed	Infrastructure identifier, not a human attacker identity
Google client ID	101714939433347726433	Debugger observed	Client identifier, not an OAuth token or victim ID
Spreadsheet ID	Recovered; exact value withheld	Debugger observed	Later A1/B1/C1 activity was observed; the resource value remains withheld
Private-key ID	Recovered; exact value withheld	Debugger observed	Complete private key also recovered and withheld
OAuth authorization	Verified RS256 assertion, successful response and populated Bearer token	Debugger observed and Runtime-memory observed	Reusable token withheld; authorization scope beyond the captured process is not asserted
Sheets channel	A1 read, B1 update and C1 poll	Runtime-memory observed	No command value or successful D1 result write was observed

Item	Public-safe result	Evidence class	Boundary
Command handling	hidden child process with combined output collection	Static capability	No operator command or output observed
Startup artifacts	GlobalProtectVPN and GlobalProtectVPNUpdate	Static and host evidence	Validate referenced action and identity

Campaign fingerprints and clustering pivots

Useful pivots include the two MSI GUIDs, the exact `hewlett-pack1` DNS hostname, the memory-resident numbered Worker pattern, `/config0`, `SheetsBot/1.0`, `elliptical-tree-505904-p7`, the recovered service-account identifier, client ID, the `-tk-` device marker, the RunOnce and task names, the install directory, the four Sheets ranges and the 18-second build/package timestamp interval. Analysts should search for combinations across packages, endpoints and permitted telemetry. A Worker family plus the gate-header name, campaign user agent, Google project identity, service account, Sheets protocol and `AmountOfFreeDiskSpace` KDF input has more clustering value than any field alone. Only `hewlett-pack1` has retained network evidence in this case.

Clustering is not attribution. Similarities may result from shared builders, copied code, reused infrastructure or deliberate imitation. Geography is especially weak by itself. Kaspersky's public HoneyMyte/CoolClient reporting provides regional context, but its reported PlugX, DLL side-loading, injected process and kernel-driver chain were not established in this case.



Evidence boundaries separating observed configuration and Google Sheets activity from unobserved operator tasking, command execution, and result submission

Evidence matrix

Claim	Evidence level	Basis
The supplied MSI and embedded x64 executable are unsigned	Verified static	Authenticode and PE/package inspection
The executable checked a Myanmar geolocation result	Observed	PCAP request/response, process memory and static control-flow artifacts

Claim	Evidence level	Basis
The process contacted Cloudflare and Google infrastructure	Observed	DNS/TCP packet evidence and Process Monitor network events
<code>hewlett-pack1</code> was resolved in the retained sessions	Observed	A-record query and response in each PCAP
Numbered Worker hosts 1-9 were materialized by the process	Observed data in runtime artifact	Full-dump private-memory regions; only host 1 was DNS-observed
A <code>/config0</code> request with the campaign user agent and populated secret header was constructed	Runtime-memory observed	Coherent writable private-memory request material; secret withheld
Cloudflare returned configuration JSON to the process	Debugger observed	Plaintext <code>InternetReadFile</code> response buffer; PCAP body remained encrypted by TLS
An actual spreadsheet ID and Google credential object were received	Debugger observed	Response fields and matching writable private-memory data; sensitive values withheld where necessary
Google project ID, service-account identity and client ID were received	Debugger observed	Values recovered from the plaintext configuration response
Complete private-key material was received	Debugger observed; withheld	Complete credential delimiters and content were present; no key material is published
An RS256 JWT assertion was constructed and signed	Runtime-memory observed	Decoded claim structure and signature verified against the recovered key; complete token withheld
A JWT bearer-grant request body was constructed	Runtime-memory observed	Grant type, assertion parameter and form-encoded request material in later captures
An OAuth access token was obtained	Debugger observed; runtime-memory corroborated	Successful response metadata and populated token/Bearer material; token withheld
<code>Sheet1!A1</code> was read	Runtime-memory observed	Exact request and response-shaped value material; returned value withheld
<code>Sheet1!B1</code> was updated	Runtime-memory observed	Response reports one updated row, column and cell
<code>Sheet1!C1</code> was polled	Runtime-memory observed	Exact request and a retained response with no command value
<code>Sheet1!D1</code> result submission occurred	Not observed	No successful D1 request or update response recovered
The design can execute shell commands and capture output	Static capability	Command, process and pipe artifacts; no operator-issued command observed

Claim	Evidence level	Basis
An operator command, output or exfiltration occurred	Not proven	No operator command, child shell, output, Sheet result or exfiltration payload recovered
The sample belongs to HoneyMyte or CoolClient	Not supported	No code, component, IOC or operational linkage sufficient for attribution

ATT&CK context

Technique	Case status	Scope
T1566.002: Spearphishing Link	Incident-reported	Email link led to a Google Sites landing page; private message evidence is not published.
T1036.005: Match Legitimate Resource Name or Location	Verified static	GlobalProtect branding and naming were used by an unsigned, behaviorally unrelated package.
T1218.007: Msiexec	Installation context	A Windows Installer package launched the embedded executable; use of <code>msiexec</code> is not malicious by itself.
T1547.001: Registry Run Keys / Startup Folder	Package/host evidence	<code>GlobalProtectVPN</code> <code>RunOnce</code> entries referenced the installed executable.
T1053.005: Scheduled Task	Package/host evidence	A <code>GlobalProtectVPNUpdate</code> task artifact requested scheduled execution.
T1016.001: Internet Connection Discovery	Observed	The target requested public IP and country information from <code>ip-api.com</code> .
T1102: Web Service	Observed plus runtime-memory corroborated	Cloudflare configuration, OAuth token issuance, A1 read, B1 update and C1 polling were retained; operator tasking and D1 result submission were not observed.
T1059.003: Windows Command Shell	Static capability	Shell and pipe artifacts support command execution; no runtime command was retained.

ATT&CK mappings organize behavior. They do not add attribution, severity, intent, or response authority.

Detection and response opportunities

Defenders should hunt for the combined pattern rather than broad cloud-service indicators:

- exact MSI and executable hashes;
- an unsigned `GlobalProtect.exe` installed under `C:\Program Files\PaloAlto\GlobalProtect VPN\` ;
- `GlobalProtect.exe /Install` launched from an MSI custom action;
- local `bcrypt.dll` , `WININET.dll` , or `CRYPT32.dll` beside the executable, verified by path, signer and hash;
- `GlobalProtectVPN` values under HKCU or HKLM `RunOnce`;
- a scheduled task named `GlobalProtectVPNUpdate` ;
- `SheetsBot/1.0` requesting `ip-api.com/json` ;
- the exact rare Worker hostname followed by Google OAuth or Sheets API traffic from the same executable;

- the recovered Google Cloud project, service-account identifier and client ID when correlated with the rare Worker family and campaign strings; and
- any verified `GlobalProtect.exe` child `cmd.exe /c` sequence.

If the chain appears on a production endpoint, follow the organization's incident-response plan. Isolate affected systems in a coordinated manner, preserve volatile evidence when safe, retain the installer and relevant logs, and investigate associated accounts and hosts. The [CISA StopRansomware Guide](#) emphasizes both prompt isolation and preservation of volatile evidence such as memory and security logs. Do not upload memory dumps to public services: they may contain credentials, personal data, documents and decrypted application content.

Defensive YARA hunting rule

The following rule combines the gate path and header name, all four spreadsheet ranges, all three protocol markers, and at least two campaign-specific artifacts. It was syntax-checked with the repository's YARA library, matched the supplied executable, and produced no matches across a bounded smoke corpus of 50 Windows System32 PE files selected alphabetically under 2 MiB. That small corpus is not a false-positive guarantee. This rule is a hunting aid, not a standalone malware verdict; analysts should validate file identity, signer, path, process ancestry, persistence and network context before taking action.

```
rule MALWAREINFO_Fake_GlobalProtect_Myanmar_Sheets_C2
{
    meta:
        description = "Detects the investigated Fake GlobalProtect Myanmar Google Sheets C2 implementation"
        author = "Malware INFO Research Team"
        date = "2026-09-02"
        reference_sample = "33d696728101c9caf6ebb215ba176bbb94e5cc16218b574029b622fd6e3bee8c"
        confidence = "High for this sample; analyst correlation required"
        scope = "Hunting aid; requires analyst correlation"

    strings:
        $gate_path = "/config0" ascii
        $gate_header = "X-Bot-Secret" ascii
        $gate_creds = "google_creds" ascii

        $sheet_a = "Sheet1!A" ascii
        $sheet_b = "Sheet1!B" ascii
        $sheet_c = "Sheet1!C" ascii
        $sheet_d = "Sheet1!D" ascii

        $protocol_val = "val-" ascii
        $protocol_command = "command-" ascii
        $protocol_output = "out-" ascii

        $campaign_agent = "SheetsBot/1.0" ascii
        $campaign_kdf = "AmountOfFreeDiskSpace" ascii
        $campaign_task = "GlobalProtectVPNUpdate" wide ascii

    condition:
        uint16(0) == 0x5a4d and
        filesize < 2MB and
        all of ($gate_*) and
        all of ($sheet_*) and
```

```

    all of ($protocol_*) and
    2 of ($campaign_*)
  }

```

Indicators

Network indicators are defanged. Shared services are contextual and should not be blocked globally without correlation.

Type	Indicator	Scope
SHA-256	a124caa58d956c7430ecb8772a3794266235917670c5b56564342bd1456897e7	Supplied MSI
SHA-256	33d696728101c9caf6ebb215ba176bbb94e5cc16218b574029b622fd6e3bee8c	Embedded executable
Domain	sheets-config-gate.hewlett-pack1[.]workers[.]dev	DNS query and response observed in the retained PCAPs
Domain pattern	sheets-config-gate.hewlett-pack{1..9}[.]workers[.]dev	Full-dump runtime-memory family; hosts 2-9 were not network-observed
URI	/config0	Correlate with the exact hostname and process
User agent	SheetsBot/1.0	Correlate with the process and destination sequence
Registry value	GlobalProtectVPN under HKCU/HKLM RunOnce	Verify referenced executable identity
Scheduled task	GlobalProtectVPNUpdate	Verify task action, creator and timestamps
Path	C:\Program Files\PaloAlto\GlobalProtect VPN\GlobalProtect.exe	Case-specific deceptive installation path
Contextual service	ip-api[.]com/json	Legitimate shared geolocation service
Contextual service	oauth2.googleapis[.]com	Legitimate shared Google service; successful token issuance was observed only in the correlated target-process evidence
Contextual service	sheets.googleapis[.]com	Legitimate shared Google service; correlate with observed A1/B1/C1

Type	Indicator	Scope
		operations and exact process identity
Google Cloud project ID	elliptical-tree-505904-p7	Runtime-delivered CTI pivot; project ownership is not established
Service-account identifier	adminproject@elliptical-tree-505904-p7.iam.gserviceaccount.com	Campaign-used cloud identity; not a victim or human attacker email
Google client ID	101714939433347726433	Identifier associated with the recovered service-account object
Spreadsheet resource ID	Recovered; exact value withheld	Active resource identifier retained in the internal evidence ledger
Credential key ID	Recovered; exact value withheld	Identifier retained internally; private key is not published

The memory-dump hash is retained in the internal evidence ledger but is not offered as a downloadable artifact. Publishing a dump would expose far more than an inert identifier.

Comparison with public HoneyMyte reporting

Kaspersky's August 2026 Securelist report, [HoneyMyte upgrades CoolClient: the backdoor gets a kernel-mode rootkit](#), describes intrusions affecting several Asian countries, including Myanmar. Its analyzed chain used PlugX as an initial implant, a legitimate Sangfor application for DLL side-loading, encrypted stages, injection into `synchost.exe`, and a kernel-mode driver.

None of those defining components was established in this case. Our sample arrived as an MSI, installed an unsigned standalone executable, used GlobalProtect branding, and contained a Cloudflare/Google Sheets design. Myanmar relevance is not enough to connect two operations. This comparison supplies regional context and, more importantly, demonstrates why victim geography alone is weak attribution evidence.

Limitations

- The organization and email are intentionally withheld, so the public article cannot independently validate the complete delivery chain.
- No public search result for the exact MSI or executable SHA-256 was identified during research on September 2, 2026.
- HTTPS payloads remained unavailable in PCAP. A separate debugger session exposed the plaintext Worker response, but it did not decrypt the packet capture.
- Writable private-memory strings prove that data existed in the captured process. Response-shaped records and their surrounding context support the specific A1/B1/C1 findings, but memory alone does not reconstruct every network transaction or server-side event.

- Only `hewlett-pack1` was observed in DNS; the other numbered Worker hosts were recovered from process memory only.
- The later token and Sheets findings supersede earlier negative token/transaction searches in separate captures. Negative findings for a C1 command value, D1 result write, child shell and output remain point-in-time observations and do not prove those events never occurred before or after capture.
- Runtime sources came from separate executions with different process IDs and must not be treated as one continuous session.
- Deep API Trace dropped 44 events and did not capture a memory snapshot.
- Payload & Memory Analyzer performed static inspection of an imported dump and retained no runtime-observed API calls.
- Debugger observations can alter timing and behavior. Here they support the narrow plaintext response-buffer finding and do not establish network-layer decryption or every subsequent action.
- Process exit status `0` does not explain why execution ended.
- No evidence supports attribution to HoneyMyte, CoolClient, Palo Alto Networks, or another named actor.
- A quiet or short execution does not establish inactivity; environment checks, unavailable C2, timing, configuration and analysis conditions can all limit observable behavior.

Conclusion

The evidence supports a high-confidence assessment that the supplied MSI is a deceptive GlobalProtect-themed package containing a backdoor design gated for a Myanmar environment. Runtime traffic verified the geolocation request, DNS resolution of the first numbered Worker host, and connections to Cloudflare and Google infrastructure. Private process memory retained a constructed gate request and the complete numbered Worker family. Later debugger observation confirmed that execution progressed beyond that gate: the process received a spreadsheet identifier and complete service-account credential object in plaintext.

The later captures retained a signed RS256 JWT bearer assertion and constructed OAuth token-request material. In a subsequent execution, the evidence advanced again: a successful OAuth response and populated Bearer token were retained, followed by an A1 read, successful B1 update and C1 poll. The retained poll contained no command value. The most consequential claim remains deliberately bounded: **channel establishment and polling were observed, but operator tasking, command execution, output collection and D1 result submission were not.**

MalwareINFO helped preserve that boundary. Deep API Trace retained an identity-bound but incomplete API timeline. Process Memory Capture produced a complete, hash-bound Full User-Mode Dump. Payload & Memory Analyzer converted the dump into integrity-verifiable, searchable evidence while preserving its missing correlation and coverage limits. These products supported analyst reasoning; they did not replace it.

References

- [Palo Alto Networks — GlobalProtect Overview](#)
- [Microsoft — Dynamic-link library search order](#)
- [Microsoft — Dynamic-Link Library Security](#)
- [Microsoft — Run and RunOnce Registry Keys](#)
- [Cloudflare — How Workers works](#)
- [Google — Using OAuth 2.0 for Server to Server Applications](#)
- [Google — Sheets API: spreadsheets.values.get](#)
- [Google — Read and write cell values](#)
- [MITRE ATT&CK — Spearphishing Link](#)
- [MITRE ATT&CK — Masquerading: Match Legitimate Resource Name or Location](#)

- [MITRE ATT&CK — System Binary Proxy Execution: Msiexec](#)
- [MITRE ATT&CK — Registry Run Keys / Startup Folder](#)
- [MITRE ATT&CK — Scheduled Task](#)
- [MITRE ATT&CK — Internet Connection Discovery](#)
- [MITRE ATT&CK — Web Service](#)
- [MITRE ATT&CK — Windows Command Shell](#)
- [CISA — StopRansomware Guide](#)
- [Kaspersky Securelist — HoneyMyte upgrades CoolClient: the backdoor gets a kernel-mode rootkit](#)
- [Malware INFO — How to Use Malware INFO Safely](#)

Source: <https://www.malwareinfo.app/blog/posts/fake-globalprotect-msi-targets-myanmar-cloudflare-google-sheets-c2/>