

Abyssos Modular RAT Analysis | ThreatLabz

zscaler.com/blogs/security-research/abyssos-technical-analysis-new-modular-rat

ThreatLabz

August 10, 2026

SANDBOX DETAIL REPORT		
Report ID: IND55-F8A8B09CCECF8CD4678BA22EBF8313		
Analysis Performed: 20/07/2026 09:34:35		
File Type: exe64		
CLASSIFICATION	ZERO DAY REPORT	MACHINE LEARNING ANALYSIS
Class Type Malicious Category Malware & Botnet	Threat Score 86 	• Suspicious
MITRE ATTACK	VIRUS AND MALWARE	SECURITY BYPASS
This report contains 4 ATT&CK techniques mapped to 3 tactics	No known Malware found	• Sample Execution Stops While Process Was Sleeping (Likely An Evasion) • Sample Sleeps For A Long Time (Installer Files Shows These Property) • May Try To Detect The Virtual Machine To Hinder Analysis • Executes Massive Amount Of Sleeps In A Loop
NETWORKING	STEALTH	SPREADING
• Performs Connections To IPs Without Corresponding DNS Lookups • Detected TCP Or UDP Traffic On Non-Standard Ports • URLs Found In Memory Or Binary Data	No suspicious activity detected	No suspicious activity detected
INFORMATION LEAKAGE	EXPLOITING	PERSISTENCE
No suspicious activity detected	• Known MD5	No suspicious activity detected
SYSTEM SUMMARY	DOWNLOAD SUMMARY	ORIGIN
• PE File Has An Invalid Certificate • Program Does Not Show Much Activity • Contains Thread Delay • Submission File Is Bigger Than Most Known Malware Samples • PE File Has An Executable Text Section And No Other Executable Section • Reads Software Policies • PE File Has A Big Code Size	Original file: 1 MB Dropped files: No dropped files Packet capture: 65 KB	Origin information not identified



Abyssos: Technical Analysis of a New Modular RAT

Introduction

In late June 2026, Zscaler ThreatLabz identified a new malware family that we track as *Abyssos*. *Abyssos* is a new modular remote administration tool (RAT) written in C++ that supports a variety of features including credential theft, file exfiltration, and remote access via VNC. *Abyssos* is in active development with multiple version numbers and different obfuscation passes that are designed to improve evasion from security products.

In this blog post, ThreatLabz provides a technical analysis of *Abyssos*, including its core features, configuration, obfuscation, network communication protocol, and capabilities.

Key Takeaways

- In late June 2026, ThreatLabz identified a new malware family, Abyssos, which provides remote administration capabilities.
- Abyssos uses different intermediate representation (IR) passes, most likely using a publicly available LLVM obfuscator (e.g. [Pluto](#)) to thwart binary analysis.
- Abyssos uses a custom TCP protocol for network communication.
- Abyssos supports a number of different network commands and downloads additional modules from the command-and-control (C2) server to enhance its capabilities.
- Although not extensively used, ThreatLabz identified a few samples of Abyssos that implemented anti-analysis techniques.

Technical Analysis

In the following sections, ThreatLabz provides a technical analysis of Abyssos version 2.4F, including its obfuscation methods, anti-analysis techniques, network protocol, and supported commands.

Anti-analysis

Abyssos uses common obfuscation methods as an anti-analysis measure. ThreatLabz identified the following techniques:

- Checks for the presence of hypervisors. Specifically, Abyssos uses the CPUID instruction to detect the presence of hypervisors, including VMware, KVM, Xen, and VirtualBox. If it detects any of these, Abyssos terminates execution.

- Checks for the following process names and exits if any are running:
 - vmtoolsd.exe
 - vmwaretray.exe
 - vmwareuser.exe
 - VBoxService.exe
 - VBoxTray.exe
 - VBoxControl.exe
 - xenservice.exe
 - prl_tools.exe
 - qemu-ga.exe
 - spice-vdagent.exe
 - vdservice.exe

Furthermore, Abyssos uses a set of different intermediate representation (IR) passes to obfuscate the binary code. ThreatLabz assesses with medium-to-high confidence that Abyssos developers use open-source LLVM-based obfuscators to achieve these results. Overall, we have observed different obfuscation passes including:

- Control flow flattening
- Bogus control flow with common opaque predicates
- Constant integer encryption
- Stack-based string obfuscation

ANALYST NOTE: Not all samples identified implement the anti-analysis techniques. For example, the most recent version of Abyssos does not include them.

Initialization phase

Before executing its core functionality and features, Abyssos performs the following initialization steps:

1. Dynamically loads any required Windows API functions and libraries. Abyssos iterates the export directory of each loaded library, calculates the CRC32 checksum of the exported function, and compares the result against the expected/passed CRC32 checksum value.

2. Abyssos creates a mutex in order to ensure that only one instance of itself is currently running. Abyssos appends either the string `_Admin` or `_User` depending on the current user rights. Interestingly, Abyssos checks if the command line includes the parameter `--elevated`. This parameter appends the string `_Admin` to the Abyssos mutex name regardless if the current user actually has administrator privileges. The mutex name is hardcoded in the binary and follows the format `Global\[UUID4]`. For example `Global\68AA60E5-6C45-4C01-9F0E-E25FC57C652F`.
3. Initiates a TCP connection with the C2 server.
4. Collects host information such as the CPU architecture, computer name, username, user's integrity level, public IP, and country of origin.
5. Sends the host's information to the C2 server along with the binary's internal version to register the compromised system. The formatting string is `HELLO| %s | %s | %s | %s | %s | v2.4F | %s | %s | %s`.
6. Abyssos creates a dedicated thread for network communication. This thread is responsible for receiving, parsing, executing, and reporting the output of any supported network commands.
7. Lastly, Abyssos starts sending the network command `PING` to the C2 server every few seconds and waits for a network command to execute. Any incoming packet is received and parsed on the previously created thread.

Network communication

Abyssos primarily uses AES in GCM mode with a hardcoded 32-byte key for encrypting both incoming and outgoing network data. The only exception is for the encrypted Abyssos modules, which have an additional layer of encryption using either AES-CBC (with a 16-byte key/IV) or using a bitwise XOR operation (as described later in Table 2).

The Abyssos network packet structure is shown below.

```
#pragma pack(push, 1)
struct network_packet
{
    uint32_t data_size;
    uint8_t unknown_flag; // Unknown, set to `1` by default
    uint8_t iv[12];
    uint8_t* data;
    uint8_t aes_tag[16];
};
#pragma pack(pop)
```

Abyssos supports a plethora of network commands. The two tables below describe each command's name and functionality along with any supported parameters.

ANALYST NOTE: Abyssos uses the pipe character “|” to delimit network command parameters. For example, the network command PM_KILL | 1234 terminates the process with PID 1234.

Network Command Name	Description
PONG	Response to the PING command.
DISCONNECT	Abyssos stops execution.
HVNC_START	Starts a VNC session with a screen width/height as an optional parameter. The default screen settings are 1920x1080.
HVNC_STOP	Stops the VNC session.
HVNC_INPUT	Simulates mouse movement and keyboard in the VNC session.
HVNC_CLONE_START	Copies a specified browser's folders/data (e.g. cookies) in the fontconfigs folder located under the Windows temporary directory.

Network Command Name

Description

HVNC_PROG

Starts a specified application under the VNC session. These applications must already be present on the host, since Abyssos does not download them. The list of supported applications and their corresponding parameters are:

- chrome: Starts the Chrome browser.
- chrome_cdp: Starts the Chrome browser and injects cookies. Specifically, Abyssos creates an instance of Chrome with the debugging port 9222. Then Abyssos connects to it (using the WebSocket protocol) and sets the cookies stored at `fontconfigs\cookies.json` into the Chrome instance by using Chrome's API function `Network.setCookie`. The purpose of this is to hijack browser sessions.
- notepad: Opens the Windows Notepad application.
- cmd: Starts a command shell.
- powershell: Starts PowerShell.
- explorer: Starts a Windows Explorer instance.
- vivaldi: Starts the Vivaldi browser.
- opera: Starts the Opera browser.
- firefox: Starts the Firefox browser.
- edge: Starts the Microsoft Edge browser.
- brave: Starts the Brave browser.
- iexplore: Starts Internet Explorer.
- thunderbird: Starts the email client Thunderbird.
- emclient: Starts the email client eM Client.
- foxmail: Starts the email client Foxmail.

If the substring `_cloned` is included in the parameter, then Abyssos attempts to use cloned/copied data from one of the web browsers above.

Network Command Name	Description
HVNC_MAXIMIZE	Displays the maximized VNC window.
SYSINFO	Gets system host information. This includes: • Username • Computer name • User's integrity level/privileges • Windows version • CPU architecture • Country code (based on public IP) • Public IP • Number of CPU logical processors • Total RAM of the system • System's uptime • GPU name
DNS_ADD	Adds a new (specified) record to the Windows hosts file.
DNS_DEL	Deletes the specified record from the Windows hosts file.
FM_COPY	Locally copies a file/directory.
CLIPBOARD_START	Starts a thread that intercepts clipboard data every second.
CLIPBOARD_STOP	Stops the clipboard interception thread.
UAC_BYPASS_FODHELPER	User Account Control (UAC) bypass method via the Windows fodhelper binary.
UAC_BYPASS_ICMLUAUTIL	UAC bypass method via COM interface ICMLuaUtil.

Network Command Name	Description
GRABBER_START	Creates a thread that scans and collects specified directories/files based on parameters. The available parameters are: • dirs: Names of directories to scan. • exts: File extensions to collect. • max: Maximum file size to collect. • ExcludeDir: List of directories to exclude from scanning.
GRABBER_STOP	Stops the grabber thread.
PM_LIST	Collects information about the system's running processes. The process information includes: • Process ID • Process name • Process's filepath • Process memory size • Process uptime
PM_START	Creates a thread that collects the system's running process information every 3 seconds.
PM_STOP	Stops the system's process information collection thread.
PM_KILL	Terminates a process by PID.
PM_SUSPEND	Suspends a process by PID.
PM_RESUME	Resumes a suspended process by PID.

Network Command Name	Description
SHUTDOWN	Shuts down the compromised host.
REBOOT	Reboots the host.
SLEEP	Puts the compromised system in sleep mode.
RESTART	Restarts Abyssos.
PF_START	Creates a thread that collects active TCP/UDP connections along with their associated processes every 2 seconds.
PF_STOP	Stops the thread that collects active TCP/UDP connections.
PF_KILL	Terminates a process by PID. The only difference with PM_KILL is that this command requires an extra (unknown) parameter.
FM_LIST	Lists files and directories along with their associated metadata in the specified directory.
FM_DEL	Deletes a file/directory.
FM_GET	Uploads a specified file from the compromised host to the C2.
FM_EXEC	Executes an already existing file on the compromised system.

Network Command Name	Description
FM_ARCHIVE	Compresses the files of a specified directory into a ZIP archive and sends them to the C2 server. The ZIP archive is stored in-memory only.
FM_ADDTOARCHIVE	Same as FM_ARCHIVE but the ZIP archive is written to disk and then sent to the C2.
FM_PUT	Downloads a file from the C2 server on to the host.
REMOTEDESKTOP_START	Creates a new thread that starts a screen recorder.
REMOTEDESKTOP_STOP	Stops screen recorder thread.
REMOTEDESKTOP_SETQ	Sets the quality (number of pixels) of any screenshots/images taken from the screen recorder.
KEYLOGGER_GETLOGS	Reads the captured keystrokes obtained from the keylogger by reading the hardcoded file <code>windows_update_cache.json</code> (located in the Windows temporary folder).
EXECURL	Downloads and executes a file. Abyssos will try to delete this file 5 seconds after it has been executed.
EXECURL_AES_HOL	Downloads encrypted shellcode and injects it into a specified process (by name). Abyssos decrypts the encrypted payload using AES-CBC. The network packet contains a 16-byte value that is used for both the AES key and IV. The downloaded file is deleted after the code injection.

Network Command Name	Description
EXECLOCAL_HEX	Receives a Windows executable file as a hex string and executes it. The downloaded file is deleted after it is executed.
EXECLOCAL_AES_HOL	Same as EXECURL_AES_HOL, but the payload is already embedded in the network packet.
AESHOL_DLL	Same as EXECLOCAL_AES_HOL. One notable difference is that this command supports payloads that have a maximum size of only 4,096 bytes.
SELF_DELETE	Abyssos deletes itself using the Windows shell command <code>cmd.exe /C ping 127.0.0.1 -n 3 >nul & del /F /Q file_path</code> .
C2CMD	Starts a remote shell session (using <code>cmd.exe</code>) with the C2 server.

Table 1: Network commands supported by Abyssos.

The table below describes the Abyssos modules (although they were not available at the time of our analysis) and decryption methods.

Network Command Name	Description	Decryption Method
KEYLOGGER	Likely a keylogger module written to disk under the Windows temporary folder with a randomly generated filename and the prefix <code>klog</code> . The module is executed via the export function name <code>abyss</code> .	Bitwise XOR with key <code>1234567890abcdef</code> .

Network Command Name	Description	Decryption Method
RECOVERY	Possibly a Chrome credentials harvester module. The module is stored under the Windows temporary folder with a randomly generated filename and the prefix rcv.	Bitwise XOR with key 1234567890abcdef.
RECOVERY_GECKO	Likely a module that recovers Firefox credentials. The module is stored under the Windows temporary folder with a randomly generated filename and the prefix rvq.	Bitwise XOR with key 1234567890abcdef.
SENDTXT	Unknown purpose. The downloaded module is written to the Windows temporary folder with a randomly generated filename and the prefix plq.	Bitwise XOR with key 1234567890abcdef.
SENDTXT2	Unknown purpose. The downloaded module is written to the Windows temporary folder with a randomly generated filename and the prefix plq.	Bitwise XOR with key 1234567890abcdef.
HDRPFILE	Unknown purpose. The downloaded module is written to the Windows temporary folder with a randomly generated filename and the prefix plq. The export name to execute the downloaded module is abyss.	Decrypted using AES-CBC with the key and IV 1234567890abcdef.

Network Command Name	Description	Decryption Method
DCFINDER	Likely a module that scans the network to locate the Domain Controller. The downloaded module is written to the Windows temporary folder with a randomly generated filename and the prefix dcf. Abyssos executes the module's export function GetDCFinderText.	Decrypted using AES-CBC with the key and IV 1234567890abcdef.
VULNSCAN	Likely a module that scans the network or compromised host for vulnerabilities. The downloaded module is written to the Windows temporary folder with a randomly generated filename and the prefix vul. Abyssos executes the module's export function GetVulnScanJson and sends the resulting output to the C2.	Decrypted using AES-CBC with the key and IV 1234567890abcdef.
ELEVATE_SYS_TOKEN	Likely a module to escalate token privileges to SYSTEM. The module is not executed if the current user has SYSTEM privileges. The downloaded module is written to the Windows temporary folder with a randomly generated filename and the prefix plg. Abyssos executes the module's export function abyss.	Decrypted using AES-CBC with the key and IV 1234567890abcdef.

Network Command Name	Description	Decryption Method
DATASCAN	Unknown purpose. The encrypted module is stored under the Windows temporary folder location with a random filename with the suffix <code>datascan.png</code> . Once decrypted, Abyssos deletes it. The decrypted module is stored in the same temporary folder but with the prefix string <code>ds</code> and a random filename. Abyssos executes the module's export <code>GetDataScanText</code> .	Decrypted using AES-CBC with the key and IV <code>1234567890abcdef</code> .
GRABCOOKIES	Likely a module that recovers browser cookies. The module is stored under the Windows temporary folder with a randomly generated filename and the prefix <code>gc</code> . The module's export name is <code>abyss</code> . After executing the module, Abyssos searches and sends any files located at <code>%TEMP%\fontconfigs\</code> to the C2 server. This command might be combined with the aforementioned network command <code>HVNC_PROG</code> .	Bitwise XOR with key <code>1234567890abcdef</code> .

Network Command Name	Description	Decryption Method
RDPWRAP	This module may be related to the open source library rdpwrap . The module is written to the Windows temporary folder with a randomly generated filename and the prefix rdp followed by a randomly generated name. The exports abyss and GetRdpWrapText are executed and any results are sent to the C2 server.	Decrypted using AES-CBC with the key and IV 1234567890abcdef.

Table 2: Abyssos network commands requiring external modules.

Conclusion

Abyssos is a new malware with post-exploitation framework features that provides filesystem access capabilities and expandable modular functionality. Abyssos leverages different obfuscation methods such as control flow flattening and string encryption to complicate reverse engineering along with techniques to evade malware sandboxes and analysis environments. Considering its active development, ThreatLabz expects Abyssos to continue to evolve.

Zscaler Coverage

Zscaler's multilayered cloud security platform detects indicators related to Abyssos at various levels. The figure below depicts the Zscaler Cloud Sandbox, showing detection details for Abyssos.

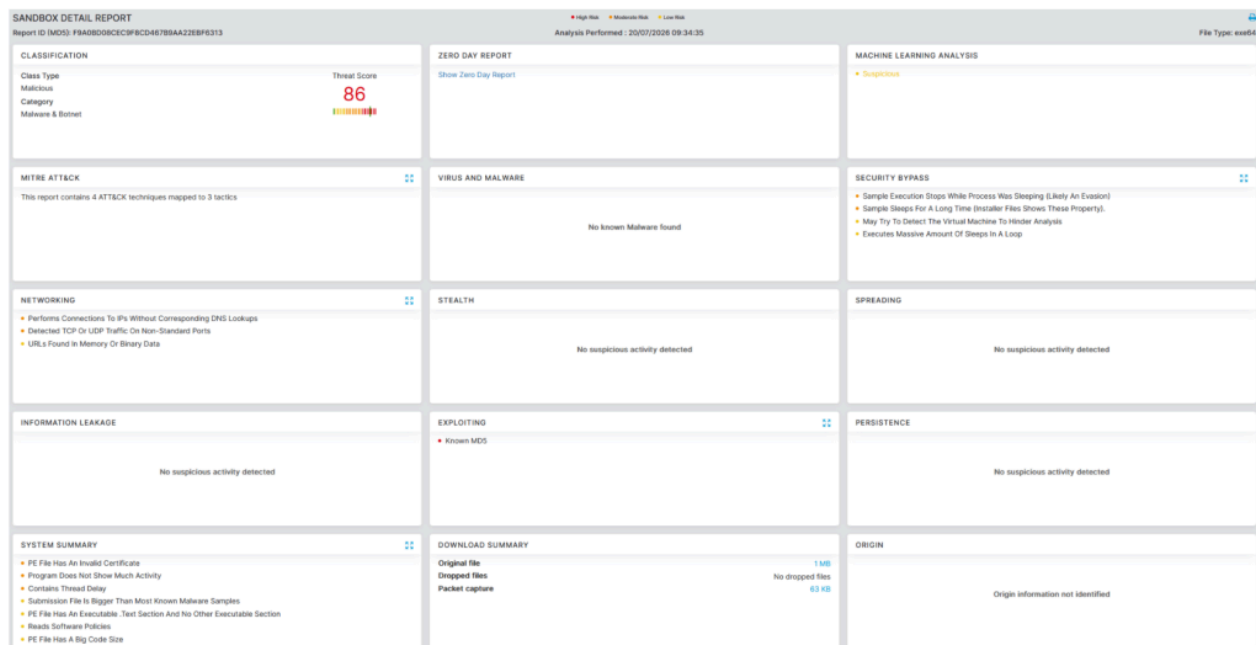


Figure 1: Zscaler Cloud Sandbox report for Abyssos.

In addition to sandbox detections, Zscaler's multilayered cloud security platform detects indicators related to Abyssos at various levels with the following threat names:

Win64.PWS.Abyssos

Indicators Of Compromise (IOCs)

IOC	Description
52b400c5be1557a8df146f62fde76d906e7e0a92ed76788717ef61c758f315aa	Abyssos sample version 2.4F
213[.]145.86.42	Abyssos C2

IOC	Description
ca94d95413210a2a325155740eb8a5c58627ad5c4e704478621e7fc8165fe173	Abyssos sample version 2.1F
209[.]99.184.223	Abyssos C2

Was this post useful?

[Yes, very!](#)

[Not really.](#)

Disclaimer: This blog post has been created by Zscaler for informational purposes only and is provided "as is" without any guarantees of accuracy, completeness or reliability. Zscaler assumes no responsibility for any errors or omissions or for any actions taken based on the information provided. Any third-party websites or resources linked in this blog post are provided for convenience only, and Zscaler is not responsible for their content or practices. All content is subject to change without notice. By accessing this blog, you agree to these terms and acknowledge your sole responsibility to verify and use the information as appropriate for your needs.

Explore more Zscaler blogs



Technical Analysis of SnappyClient

[Read post](#)



Technical Analysis of the Latest Variant of ValleyRAT

[Read post](#)



Technical Analysis of DarkVision RAT

[Read post](#)

Get the latest Zscaler blog updates in your inbox

By submitting the form, you are agreeing to our [privacy policy](#).

© 2026 Zscaler, Inc.

All rights reserved. Zscaler™ and other trademarks listed at zscaler.com/legal/trademarks are either (i) registered trademarks or service marks or (ii) trademarks or service marks of Zscaler, Inc. in the United States and/or other countries. Any other trademarks are the properties of their respective owners.