DEVMAN Ransomware: Analysis of New DragonForce Variant

July 1, 2025

HomeMalware Analysis
DEVMAN Ransomware: Analysis of New DragonForce Variant

Editor's note: *The current article is authored by Mauro Eldritch, offensive security expert and threat intelligence analyst. You can* ***find Mauro on X***.

New ransomware strains continue to surface frequently, and many of them are loosely built on or repackaged from existing families. One such case involves a sample resembling DragonForce ransomware, yet bearing several unique traits and identifiers suggesting the involvement of a separate entity known as DEVMAN.

A previously analyzed campaign connected to the Mamona strain, itself linked to BlackLock affiliates and the Embargo group, also intersected with DragonForce activity. DragonForce published BlackLock's .env file, not just any target. This is the first case where we saw two gangs actively and publicly attacking each other.

This newer sample, uploaded by TheRavenFile, appears related but not entirely identical to the DragonForce lineage. Despite being labeled as a DragonForce or Conti variant by most AV engines, the sample displays unique behaviors that point toward DEVMAN involvement.

Our DragonForce/Conti sample on VT, but don't be fooled by appearances

DEVMAN: Key Takeaways

DEVMAN reuses DragonForce code but adds its own twists: The .DEVMAN extension and unique strings sit on top of a mostly DragonForce codebase.

Attribution is muddy: The sample does not contain any leak-site links to DEVMAN, while the ransom note is strictly a copy of the DragonForce one.

DragonForce's RaaS model allows affiliates to create spinoff variants:
That's likely how samples like DEVMAN emerged; built on DragonForce code, but customized and repackaged.

Ransom notes encrypt themselves: This happens likely due to a builder flaw

Most malicious activity takes place offline, aside from SMB probing: No external C2 communication was observed during analysis.

Three encryption modes are built in: full, header-only, and custom.

Behavior varies by OS: Wallpaper change fails on Windows 11 but works on Windows 10.

Dragons as a Service

Some time ago, DragonForce introduced their RaaS (Ransomware-as-a-Service) model, aiming to recruit both affiliates to operate their ransomware and others who wanted to use their infrastructure, branding, and reputation as a platform to publish stolen data.

This shift brought new actors into the landscape, increasing overall activity, noise, and irregularities, including the sample analyzed here. Depending on the analyst or tool, it may be labeled as DragonForce, Conti (the base framework for DragonForce), or DEVMAN.

DEVMAN? A relatively new actor has recently emerged under this name, featuring its own Dedicated Leak Site (DLS) called Devman's Place, a separate infrastructure, and nearly 40 claimed victims, primarily in Asia and Africa, with occasional incidents in Latin America and Europe.

A Hybrid Ransomware Sample

Let's analyze the sample inside ANY.RUN's secure interactive sandbox:

View analysis session

This sample, flagged by most antivirus engines as a DragonForce (or Conti), is actually, modified to behave like a new variant belonging to DEVMAN. It uses that name as the file extension for encrypted data but otherwise shares a large part of its codebase with DragonForce, including leftover strings and identifiers. That strongly suggests DEVMAN may be using a DragonForce build for some of its operations.

Encrypted file with the .DEVMAN extension

This appears to be a lightly customized version; one that hasn't attracted much attention, either from the threat intelligence community or from its own operator. The result is a tangled ransomware crossbreed with overlapping traits.

Automatic detection labels the sample as “DragonForce”

A closer look reveals more.

Initial Behavior and Detection

First things first — our newborn dragon does what dragons do: it burns down the village. Files are encrypted rapidly and automatically, also attempting to locate SMB shared folders to spread further — but in our lab environment, it wasn’t that lucky.

Two things caught our attention immediately. First, on Windows 11, the sample was unable to change the wallpaper for unknown reasons, while on Windows 10 it worked flawlessly.

Second, although desktop files are the most visible, they are not the last to be encrypted. The process continues beyond them.

SMB traffic attempting to laterally spread the infection

Ransom Note Issues and Deterministic Renaming

The ransom notes were not dropped as expected. Instead, every location where a note should have appeared contained, quite mysteriously, a file with a scrambled name and the .DEVMAN extension, suggesting the sample might be malfunctioning and targeting its own files.

Fortunately, ANY.RUN logs all activity, not just network traffic, but disk writes as well, allowing us to reconstruct one of those files right at the moment it was created. And, interestingly enough, the ransom note isn't just similar to the ones used by DragonForce. It is, in fact, a DragonForce ransom note.

A DragonForce ransom note

When retrieving the list of created and modified files, we noticed an interesting pattern: the sample scrambles file names instead of simply appending an extension.

And here's the most curious part; its own readme.txt files, once encrypted, are always renamed to e47qfsnz2trbkhnt.devman. This strongly suggests the use of a deterministic function that produces static outputs for identical inputs.

Encrypted Ransom notes, all sharing the same name

Offline Behavior and Local Footprint

So, let's focus on those local oddities, and a good place to start it's the binary itself.

Aside from the aforementioned SMB connections, no suspicious network dialogue was observed, suggesting that all malicious activity takes place locally and offline.

Using FLOSS, a tool by Mandiant, we can decode and extract additional strings to better understand the sample's internal logic prior to disassembly.

The first thing we notice is that the sample checks for Shadow Copies (probably just to make sure we've got a solid backup policy in place) and lists a series of file extensions that it deliberately avoids encrypting.

Decoded strings obtained via Floss

Encryption Modes and File Targeting

Further analysis reveals multiple encryption modes: full encryption, header-only encryption, and custom encryption, designed to prioritize either speed or complexity, depending on the intended scenario.

Header-only encryption, in particular, allows the malware to corrupt large volumes of data in less time, trading completeness for speed.

At least 3 different encryption modes are available

SMB Spread and Local Targeting

Further exploration reveals a bit more detail about the sample's attempts to connect to SMB folders, explicitly referencing local network octets and hardcoding the ADMIN$ share name, along with several error and debug messages.

Octects belonging to local addresses and direct mention to the ADMIN share

Persistence and File Lock Evasion via Restart Manager

Another interesting behaviour that further supports the Conti lineage of this sample is its interaction with the Windows Restart Manager. The malware creates temporary sessions under the registry key:

HKEY_CURRENT_USER\Software\Microsoft\RestartManager\Session0000

There, it logs metadata such as Owner, SessionHash, RegFiles0000, and RegFilesHash, pointing to system-critical files like NTUSER.DAT and its corresponding logs.

Each of these entries is quickly deleted after being written, likely an attempt to avoid leaving persistent forensic traces. This pattern mirrors behaviour seen in Conti and later carried on by DragonForce, which now appears to be inherited by DEVMAN (what a Zoo!).

The goal seems clear: use the Restart Manager to bypass file locks and ensure encrypted access to active user session files. It’s noisy, and somewhat old, but it works.

Regkeys altered by the sample

Learn to analyze cyber threats

See a detailed guide to using ANY.RUN's Interactive Sandbox for malware and phishing analysis

Read full guide

Mutex Usage and Sample Coordination

Another notable behavior involves the use of synchronization primitives, particularly mutexes, to coordinate the sample's execution and possibly prevent multiple instances from running in parallel. This is standard among ransomware families derived from Conti, and this case is no exception.

Right from the beginning, the sample creates a mutex named: hsfjuukjzloqu28oajh727190

This mutex is not randomly generated; it is hardcoded into the binary, as confirmed by decoded strings extracted using FLOSS. Its presence suggests that the sample uses it to detect existing instances of itself, a basic anti-reentry mechanism.

The sample also creates several mutexes and interacts with objects under the naming pattern:

> Local\RstrMgr[GUID]
>
> Local\RstrMgr-[GUID]-Session0000

These mutexes are tied to the Windows Restart Manager API and match the behaviour seen in previous ransomware families (notably Conti and its derivatives), which use this mechanism to query which processes are holding handles to specific files.

This facilitates forced encryption of locked resources, including user profile data like NTUSER.DAT.

The reuse of fixed strings can serve as a strong indicator of compromise (IOC) for future detection or correlation with other samples likely created using the same packer or builder. However, this is a volatile indicator that is likely to change over time.

When possible, assign a "trust" expiration date (or half-life) to indicators; it can be a valuable practice for maintaining detection accuracy over time.

Mutexes used by the sample

Final Observations

An Experimental Build with Unusual Behavior

This sample looks more like an affiliate testing a new build than something currently being deployed that you'd casually run into in a production environment. While not particularly sophisticated, it presents a number of unusual behaviors worth highlighting, particularly its tendency to encrypt its own ransom notes.

A Critical Flaw in the Builder

While it's ironic that no one could, at least not easily, pay the ransom without knowing who to pay (because the ransom note gets encrypted), the underlying message here is more serious: there's a core design flaw in the builder that allows it to self-encrypt key components.

That simple .txt file is often the only clue victims have to identify the threat actor and initiate negotiation; and for the threat actor, it's the best chance of getting paid.

I spoke with DEVMAN, who stated "[…] we stopped using DragonForce months ago […]".

Threat Actor Communication

One noteworthy indicator of a threat actor's maturity is their ability to maintain polite, detailed, and respectful communication; a trait that also applies to DEVMAN. This attitude seems to echo in their technical approach, even in cases where their ransomware encrypts its own ransom notes.

A Familiar Build Beneath the Surface

Now, if we strip this sample of its oddities, there's not much to talk about it on its own merits (no offense meant to the developers), or at least nothing to say that we haven't covered in other articles about ransomware.

Still, its oddities make it a valuable case study, not for technical innovation, but for the way it reflects shifting actor dynamics and common development pitfalls in the ransomware ecosystem.

Turning Oddities into Actionable Intelligence

Unusual samples like this DEVMAN variant can easily slip past traditional analysis workflows. With ransom note encrypted, scrambled filenames, and unexpected behavior across operating systems, manual investigation becomes time-consuming and risky to overlook.

This is where ANY.RUN's Interactive Sandbox proves invaluable. By logging every action in real time, from file system changes to mutex creation and registry modifications, it enables analysts to trace even fragmented or malfunctioning ransomware behavior.

This kind of visibility gives security teams a real operational advantage:

- **Faster detection and response:** Immediate insight into threat behavior, even in offline or misconfigured attacks.
- **Clearer attribution:** Links to reused infrastructure, code similarities, and TTP patterns are surfaced early.
- **More efficient investigation workflows:** Analysts can extract IOCs, map persistence mechanisms, and understand impact without switching tools.
- **Better collaboration across teams:** Findings can be shared easily with SOCs, threat intel units, and communications teams, ensuring faster alignment during incidents.

Start 14-day trial of ANY.RUN's Interactive Sandbox in your SOC today

MITRE ATT&CK Mapping

Let's jump to drafting a quick ATT&CK matrix for this sample, which ANYRUN does automatically for us:

T1204.002 – User Execution: Malicious File

The executable requires user (or threat actor) interaction to launch.

T1053.005 – Scheduled Task/Job: Scheduled Task

Presence of scheduling-related strings implies possible persistence via tasking.

T1027 – Obfuscated Files or Information

Internal file renaming and readme scrambling suggest static obfuscation logic.

T1070 – Indicator Removal on Host

The sample deletes registry keys and values shortly after writing them.

T1135 – Network Share Discovery

Explicit scanning for SMB shares (ADMIN$, IP ranges like 192.x, 172.x).

6T1021.002 – SMB/Windows Admin Shares

Uses netapi32, srvcli, and netutils to interact with administrative shares.

T1005 – Data from Local System

Enumerates and encrypts user data including NTUSER.DAT and log files.

T1486 – Data Encrypted for Impact

Core functionality: encrypting files with .DEVMAN extension.

T1490 – Inhibit System Recovery

Attempts to interact with volume shadow copies.

IOCs

MD5:e84270afa3030b48dc9e0c53a35c65aa

SHA256:df5ab9015833023a03f92a797e20196672c1d6525501a9f9a94a45b0904c7

403

FileName:hsfjuukjzloqu28oajh727190

FileName:e47qfsnz2trbkhnt.devman

SHA256:018494565257ef2b6a4e68f1c3e7573b87fc53bd5828c9c5127f31d37ea964f8

References

Analysis: https://app.any.run/tasks/64918027-01e6-415a-85b3-474fca5fc5c4

VirusTotal Analysis (multiple labeling/attribution): https://www.virustotal.com/gui/file/

df5ab9015833023a03f92a797e20196672c1d6525501a9f9a94a45b0904c7403

Original Intel Pulse (OTX): https://otx.alienvault.com/pulse/

68535853fe15cff17229577d

Mauro Eldritch

+ posts

Mauro Eldritch is an Argentinian-Uruguayan hacker, founder of BCA LTD and DC5411 (Argentina / Uruguay). He has spoken at various events, including DEF CON (12 times). He is passionate about Threat Intelligence and Biohacking. He currently leads Bitso's Quetzal Team, the first in Latin America dedicated to Web3 Threat Research.

Follow Mauro on:
X
LinkedIn
GitHub

ANYRUN cybersecurity malware analysis

Mauro Eldritch

Mauro Eldritch is an Argentinian-Uruguayan hacker, founder of BCA LTD and DC5411 (Argentina / Uruguay). He has spoken at various events, including DEF CON (12 times). He is passionate about Threat Intelligence and Biohacking. He currently leads Bitso's Quetzal Team, the first in Latin America dedicated to Web3 Threat Research.

Follow Mauro on:
X
LinkedIn
GitHub
View all posts Twitter
What do you think about this post?

2 answers

- Awful
- Average
- Great

No votes so far! Be the first to rate this post.

0 comments