

Detailed Analysis of DragonForce Ransomware

 medium.com/s2wblog/detailed-analysis-of-dragonforce-ransomware-25d1a91a4509

S2W

January 14, 2026



S2W is a big data intelligence company specialized in the Dark Web, Deepweb and any other covert channels.

| *First Published: Jan 14, 2026*



This Figure was generated using [Google Gemini Nano Banana](#)

Executive Summary

- The DragonForce Ransomware Group, first detected in December 2023, developed its own ransomware based on LockBit 3.0 (Black) and Conti Ransomware code.- This group operates a service called “Ransombay”, which allows affiliates to choose options such as receiving customized payloads, and refers to itself as a cartel.
- DragonForce Ransomware Group has been linked to BlackLock, RansomHub, and LockBit based on infrastructure migration and duplicate source code.- According to [VX-Underground](#), DragonForce, LockBit, and Qilin attempted to form an alliance and establish communication channels.
- The strings DragonForce Ransomware are obfuscated using a custom algorithm and decrypted during execution for use.- Supports 5 command-line arguments; the “” argument determines whether local or network encryption is used.- Target files are encrypted using the ChaCha8 algorithm, with 534 bytes of metadata appended to the end of each encrypted file.- If the “encrypt_file_name” value is enabled, it not only appends an extension to encrypted files but also encodes the original filename using Base32.- If the “custom_icon” or “custom_wallpaper” values are enabled, it changes the icon of encrypted files or the desktop wallpaper of the compromised system.
- While ransomware variants like DragonForce exist that are based on Conti and LockBit, the sample analyzed in this report was identified as DragonForce ransomware developed based on Conti ransomware.
- During threat hunting related to the DragonForce group, the S2W Threat Research and Intelligence Center TALON obtained a decryptor capable of decrypting data from a specific victim system.- The Windows decryptor identifies encrypted files based on their file extension (.RNP). It decrypts the ChaCha8 session key present at the end of the encrypted file and then decrypts the file using the ChaCha8 algorithm.- The decryptor for ESXi, in addition to the file extension (.RNP), checks whether the 8 bytes present at the end of the encrypted file match the magic bytes (the “build_key” value from the Config information). If they match, it proceeds with the decryption process. The decryption process is identical to that of the Windows decryptor.

Introduction

The DragonForce ransomware group was first discovered on December 13, 2023, when a user named @dragonforce, active on BreachForums uploaded a post titled “aglgases.com — gigabytes of data!” The post contained a DLS (Data Leak Site) link for the DragonForce ransomware. As of December 2023, at the time of discovery, 17 victim companies had been uploaded.

This group has developed its own ransomware based on LockBit 3.0 (LockBit Black) and Conti Ransomware code. It operates a “Ransombay” service that allows affiliates to choose options such as providing customized payloads, rapidly expanding its influence. Furthermore,

DragonForce refers to itself as a cartel in its activities.

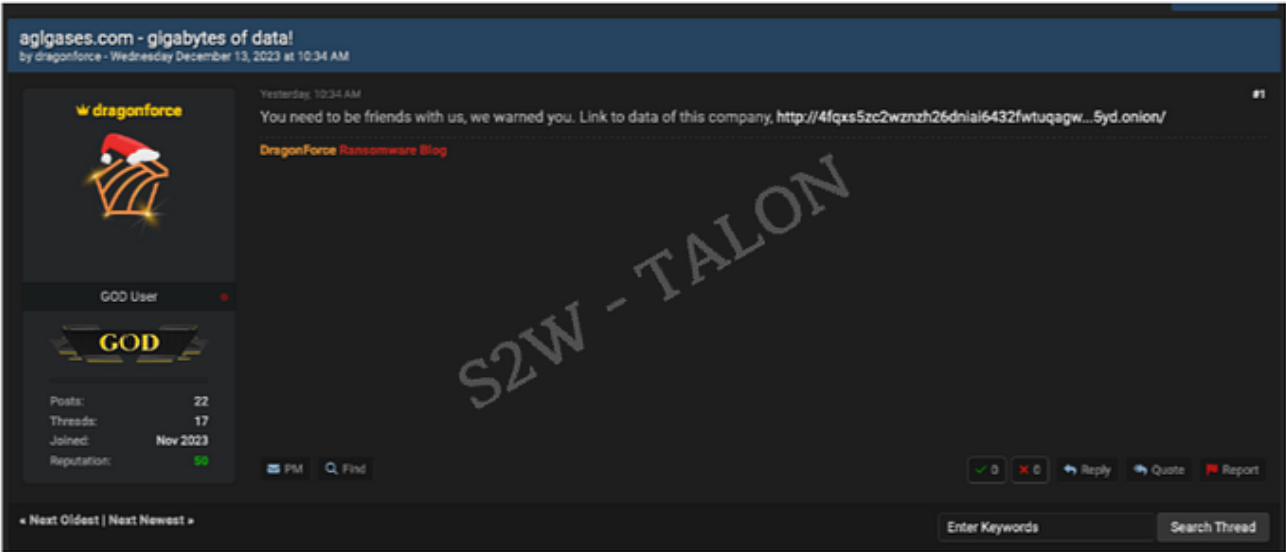


Figure 1. Post uploaded to BreachForums(Title: “aglgases.com — gigabytes of data!”)

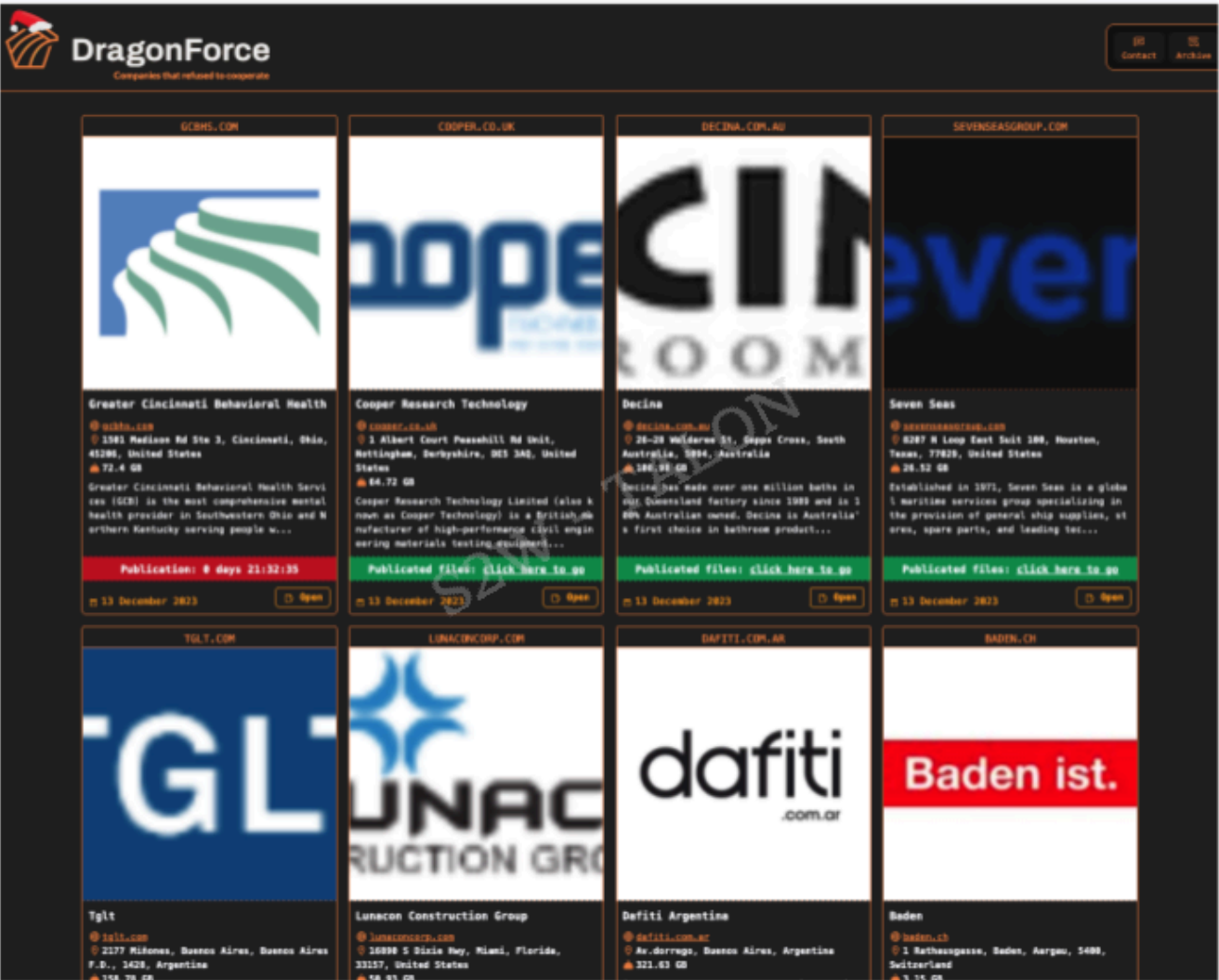


Figure 2. DragonForce's DLS as of December 2023

In June 2024, the DragonForce ransomware group, primarily active on BreachForums, uploaded a post titled “[RaaS] / DragonForce — we invite you to join our family” to the RAMP forum, signaling its transition to RaaS. The post outlined recruitment for affiliates, including roles such as IAB(Initial Access Brokers), individual pentester, and organized pentester team, stating that affiliates working with DragonForce would receive 80% of the ransom payment as profit.

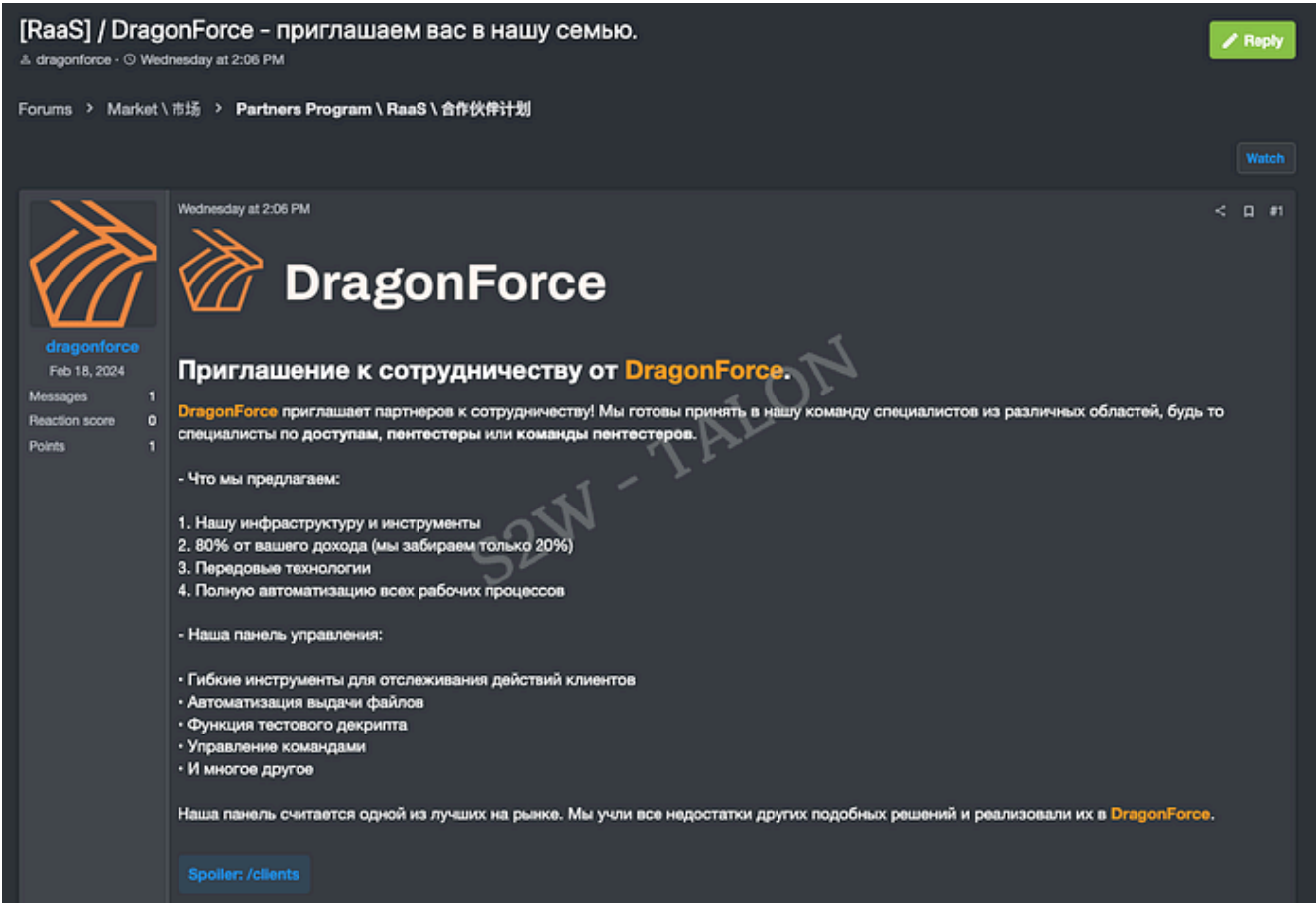


Figure 3. [RaaS] / DragonForce — we invite you to join our family

Other Related Groups

The groups mentioned as having ties to the DragonForce ransomware group and the evidence supporting this are as follows.

Related Other Group	Reason
BlackLock	Duplicate source code, identical ransom note
RansomHub	Infrastructure Migration
LockBit	Binary similarity
LockBit & Qilin	Establishing communication channels and strengthening influence among the three groups

Table 1. Groups Associated with the DragonForce Ransomware Group

DragonForce — BlackLock

According to [Resecurity](#), in March 2025, the DragonForce ransomware group identified that the real IP address of the BlackLock ransomware group's DLS had been exposed due to a misconfiguration. They subsequently accessed the infrastructure and exploited a Local File Inclusion (LFI) vulnerability to collect information, including credentials. As a result, data such as the BlackLock group's chat logs and "/etc/passwd" were exposed on the BlackLock DLS, and the page's logo was replaced to the DragonForce logo.

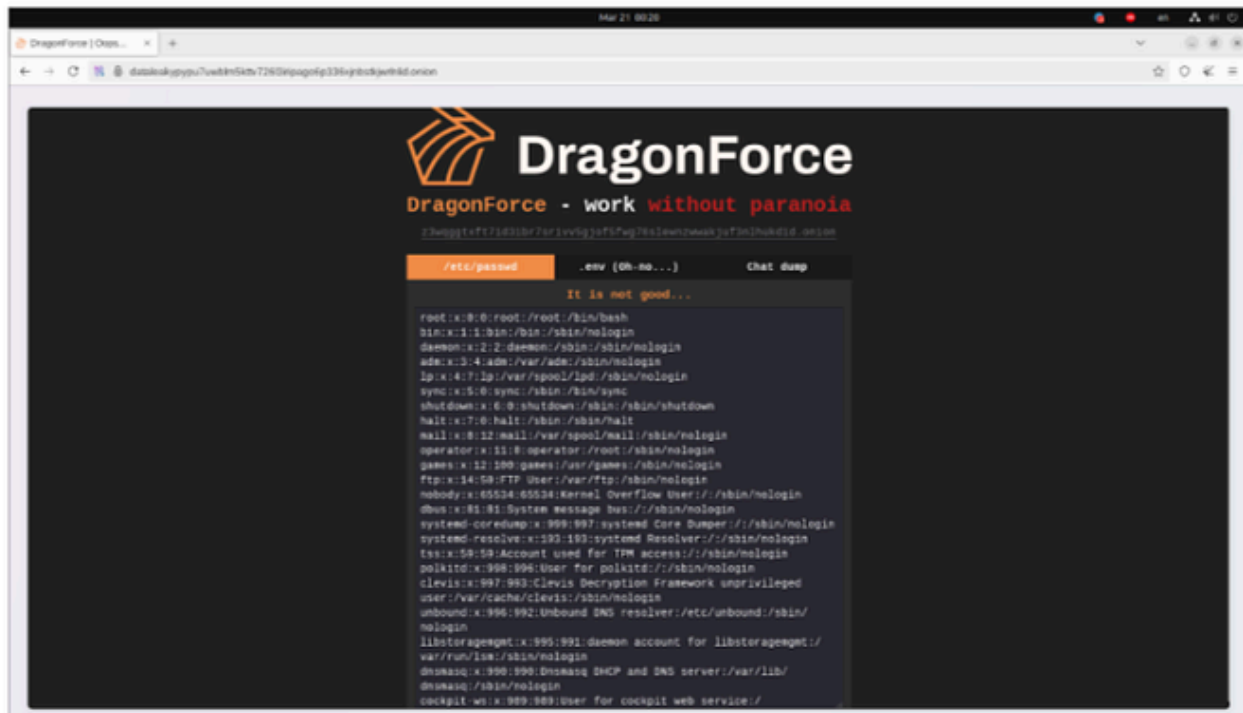


Figure 4. BlackLock DLS compromised by DragonForce (Source: [Resecurity](#))

Additionally, analysis of the BlackLock ransomware, which was provided to [Resecurity](#) researchers by a BlackLock representative, revealed overlapping code structures with DragonForce ransomware, and the ransom notes were nearly identical. However, a key difference was identified: DragonForce was written in C/C++, while the BlackLock ransomware was written in Go.

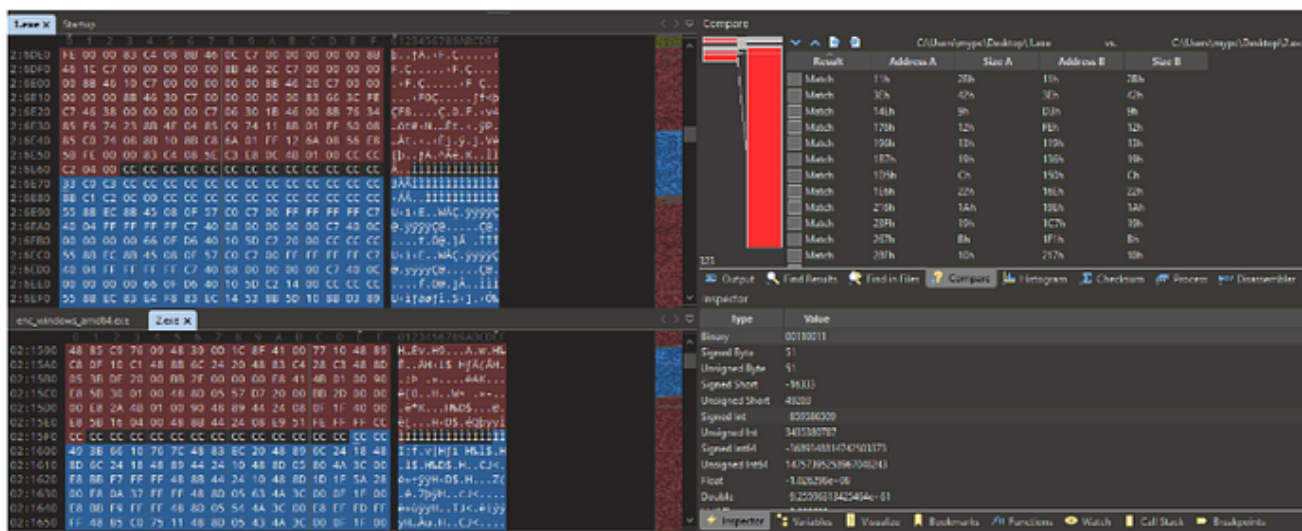


Figure 5. Analysis of overlapping code structures between BlackLock and DragonForce (Source: [Resecurity](#))

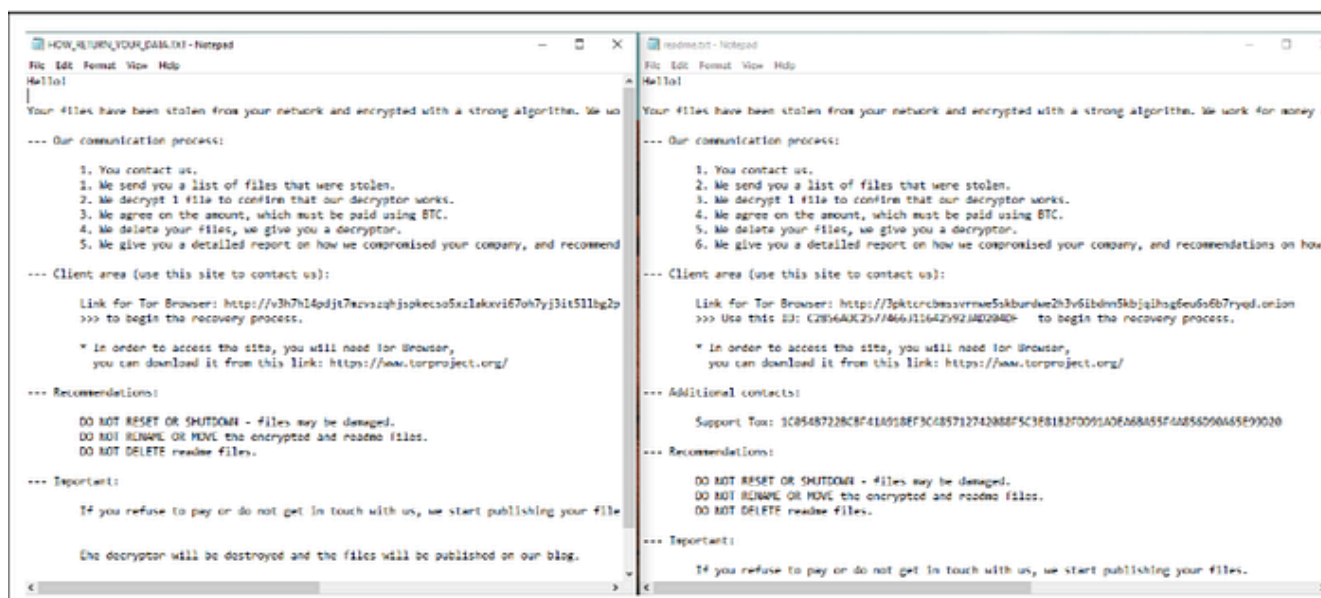


Figure 6. (Left) BlackLock ransom note / (Right) DragonForce ransom note (Source: [Resecurity](#))

DragonForce — RansomHub

In April 2025, a post uploaded for the RAMP forum mentioned a connection between the DragonForce ransomware group and the RansomHub ransomware group. The user @dragonforce, who uploaded the post, stated, “It has been decided that RansomHub’s infrastructure will be transferred to DragonForce, and the two groups are in a partnership.”

Immediately after the post was published, the DLS address previously used by the RansomHub group was changed to a new address, and it was confirmed that the contact information for the newly created DLS contained details related to the DragonForce group. Additionally, a new category named “DragonNews” was added to the new DLS. However, it remains unclear whether RansomHub’s infrastructure was simply transferred to the DragonForce group or if DragonForce seized the infrastructure through an attack, similar to the previous BlackLock case.

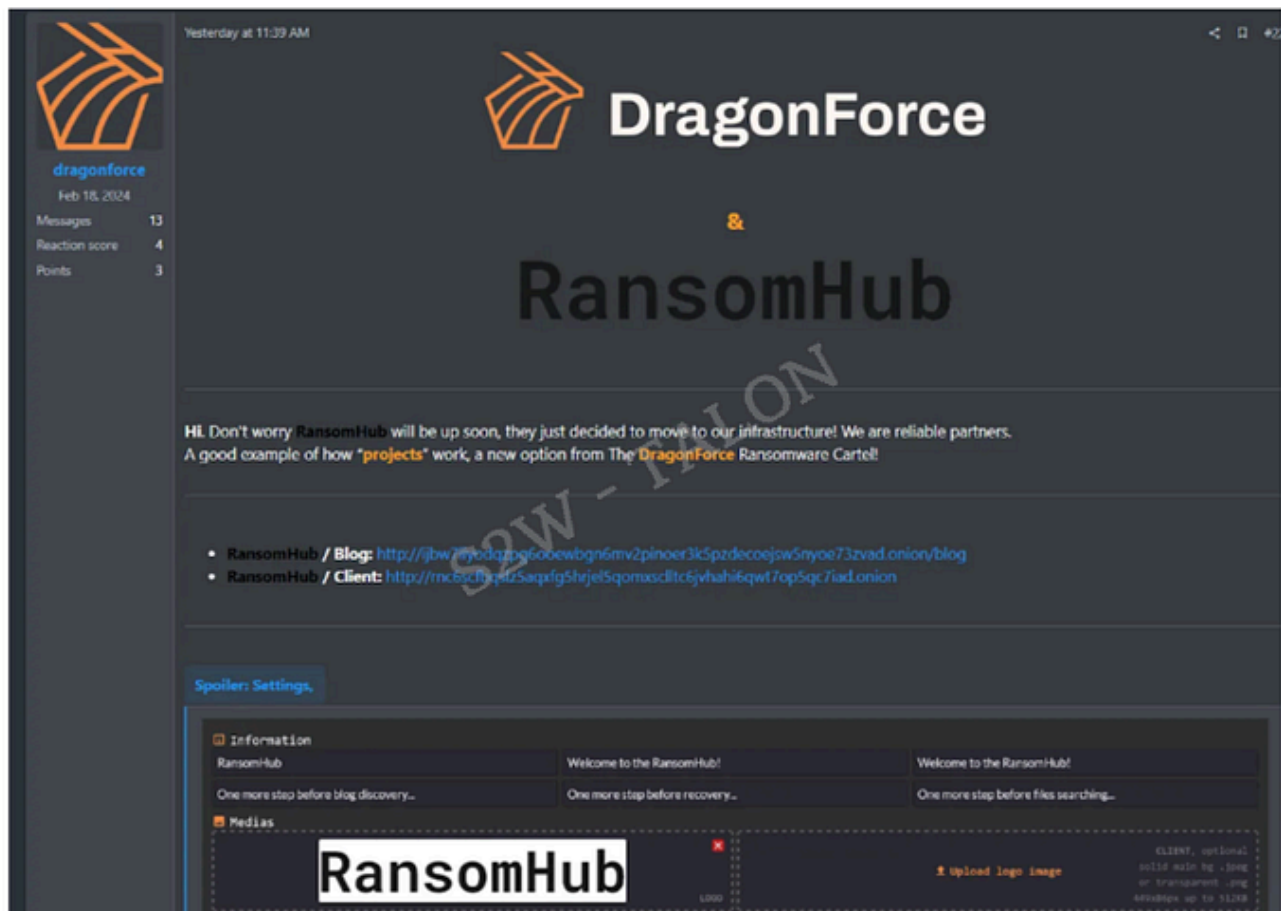


Figure 7. Post announcing the migration of the RansomHub infrastructure to DragonForce

DragonForce — LockBit

[CYBLE](#) identified the DragonForce ransomware binary as being based on LockBit 3.0 (Black) ransomware. BinDiff analysis suggested that the DragonForce ransomware was likely created using the leaked LockBit 3.0 (Black) builder. Comparing the two ransomware binaries using BinDiff revealed that 93.7% of all functions matched, while consecutively executed commands and branch structures matched approximately 99%.

- LockBit 3.0 Black Builder: a736269f5f3a9f2e11dd776e352e1801bc28bb699e47876784b8ef761e0062db
- DragonForce Ransomware: 1250ba6f25fd60077f698a2617c15f89d58c1867339bfd9ee8ab19ce9943304b

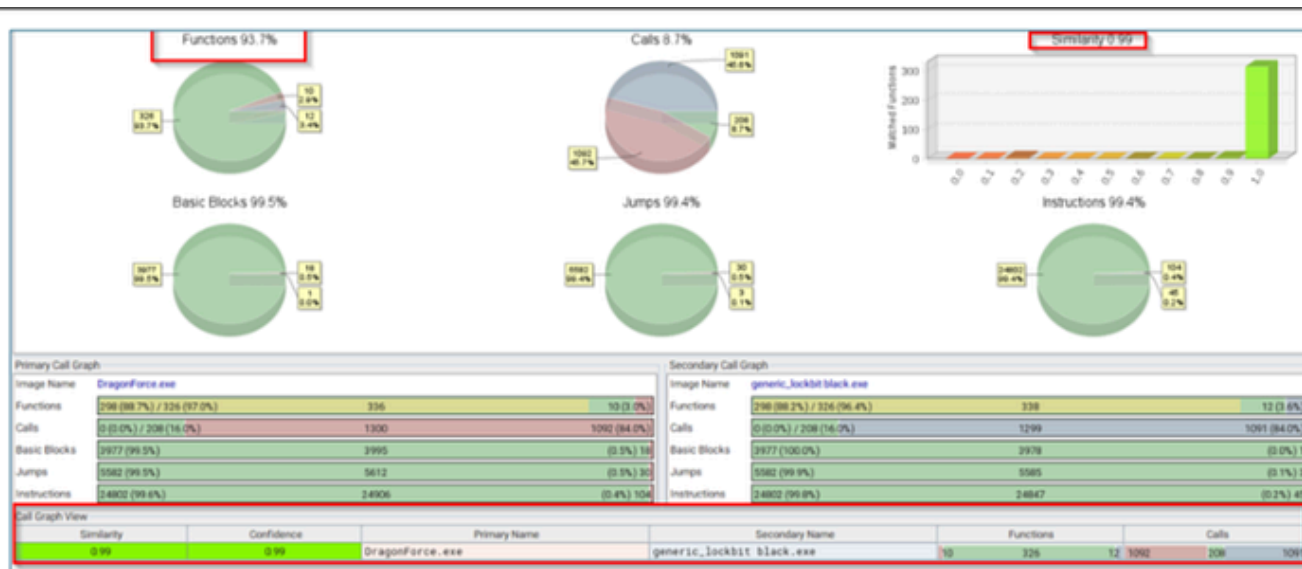


Figure 8. LockBit 3.0 (Black) — DragonForce Binary Similarity (Source: [CYBLE](#))

DragonForce — LockBit & Qilin

According to [VX-Underground](#), DragonForce proposed establishing communication channels with the LockBit and the Qilin group, arguing that prohibiting internal disputes or public insults would strengthen each other's influence. LockBit, in particular, supported this proposal and attempted to establish direct communication channels by exchanging Tox IDs.

TTPs

[Group-IB](#) investigated the breach incident caused by DragonForce ransomware and simultaneously released a report analyzing the group's panel.

The DragonForce ransomware group initially infiltrated the victim system network via a remote desktop server and attempted persistent logins using valid domain accounts(**Domain Accounts, T1078.002**). After successfully logging in, they executed a PowerShell command(**PowerShell, T1059.001**) to download additional payloads from a remote location using a Cobalt Strike Beacon, maintaining persistence throughout this process using SystemBC.

Subsequently, the attacker used Mimikatz to dump credential information(**LSASS Memory, T1003.001**) and collected Active Directory configuration(**Domain Trust Discovery, T1482**) and network information(**System Network Configuration Discovery, T1016**) via ADFind and netscanold.exe. The obtained credentials and network information were leveraged for lateral movement using Remote Desktop Protocol(**Remote Desktop Protocol, T1021.001**), ultimately deploying DragonForce ransomware across the entire network.

Group-IB conducted an analysis after accessing the DragonForce panel, in addition to investigating the incident. According to the analysis, the DragonForce panel is categorized into Clients, Builder, My Team, Publications, Rules, Blog, and Profile sections. The functions of each

category are as follows:

Category	Description
Clients	- Information about the victim company, including the ransom amount, client ID, DLS, and size of the leaked files
Builder	- Functionality enabling affiliates to build DragonForce ransomware according to their desired configuration - Option to select either the LockBit version or their own DragonForce ransomware (based on Conti)
My Team	- An interface for viewing partners related to affiliates
Publications	- Victim data information disclosed by the affiliate to DLS
Rules	- Rules, Guidelines, and Contact Information Related to the Use of DragonForce Ransomware
Blog	- Contains a DLS link
Profile	- Affiliate, authentication details, password change, and other unique information verification features

Table 2. DragonForce Panel Functions by Category (Source: [Group-IB](#))

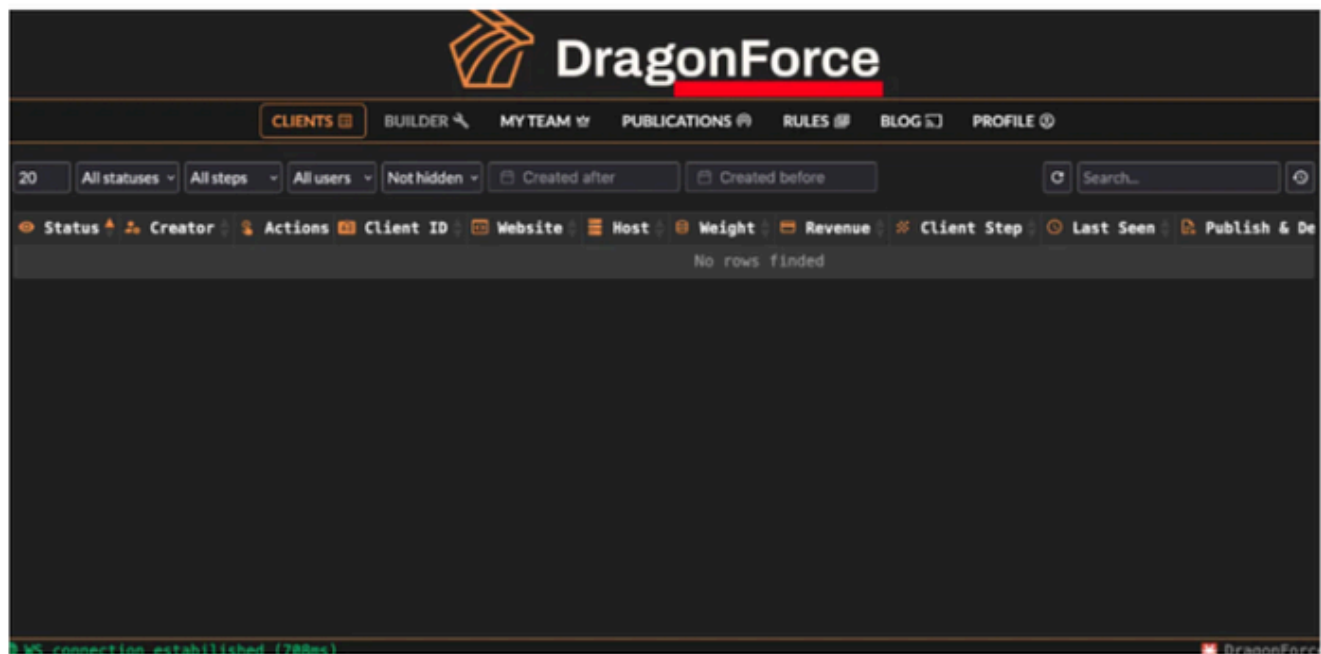


Figure 9. DragonForce Panel (Source: [Group-IB](#))

- File Type: Win32 EXE
- MD5: ada4e228e982a7e309bb6a3308e4872d
- SHA256: 451a42db9c514514ab71218033967554507b59a60ee1fc3d88cbeb39eec99f20

The main execution flow of the DragonForce ransomware is illustrated in Figure 10 and summarized as follows.

1. Decrypt Config Data & Check CommandLine Parameter
2. Register Job Scheduler & Time Synchronization
3. Kill Process
4. Encryption: ChaCha8 + RSA-4096 Algorithm
5. Delete Volume Shadow Copy

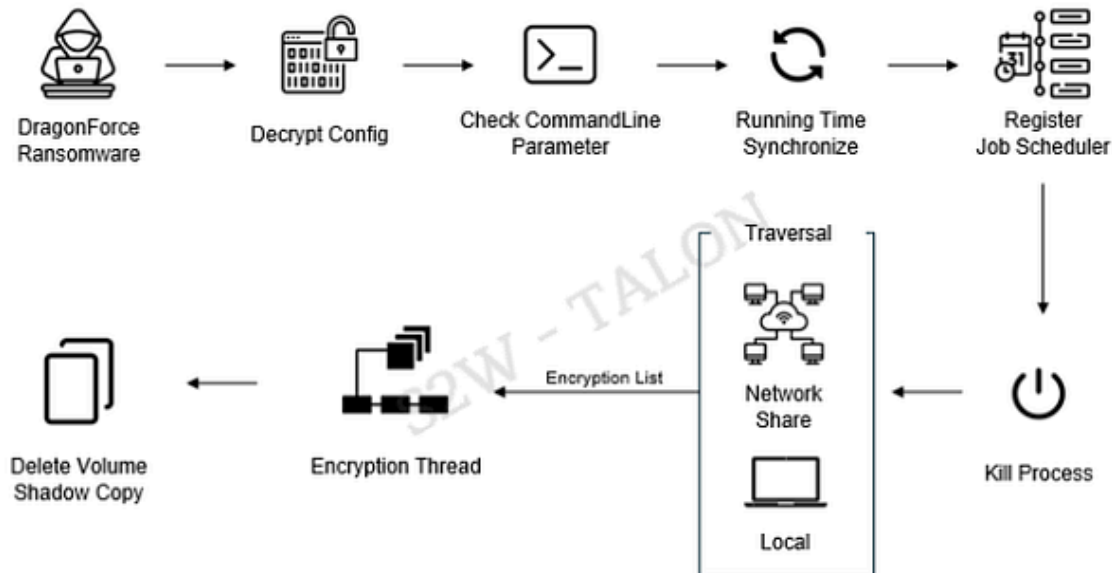


Figure 10. DragonForce Ransomware Execution Flow

1. String Obfuscation

All strings used by DragonForce ransomware are obfuscated and decrypted using a custom algorithm. This custom algorithm consists of two forms, and the values within the algorithm (Value1, Value2) differ for each string.

[Algorithm 1]

```
for ( idx = 0; idx < len(ObfuscationString); ++idx )
  ObfuscationString[idx] = (Value1 * (Value2 - ObfuscationString [idx]) % 127 + 127) % 127;
```

[Algorithm 2]

```
for ( idx = 0; idx < len(ObfuscationString); ++idx )
  ObfuscationString[idx] = (Value1 * (( ObfuscationString [idx]) -Value2 )% 127 + 127) % 127;
```

Figure 11. Deobfuscation Algorithm

2. API Resolving

The DragonForce Ransomware analyzed in this report does not use the API Resolving method, but some samples (SHA256:

410db536a57c511b0ccac2639e0eb3320f303fc5c90242379ab43364c51ef321) were found to perform API resolving based on the MurMurHash2 algorithm to dynamically load the API.

3. Config Decrypt

The key configuration values (hereafter referred to as Config) required for execution are stored within the binary in an encrypted format. These are decrypted using the ChaCha8 algorithm for use. The decrypted data is recorded in a separate log file during execution. Based on an analysis of the field values recorded in this log file, the identified config information is as follows.

The “instance_key” is an 8-byte value generated randomly, but it is not used anywhere except during the process of storing it in the log file in binary format. However, when accessing DragonForce’s negotiation site using the Unique ID included in the ransom note, a new 32-byte SID value is provided to the victim. Considering that this SID value is composed of the first 16 bytes of the Unique ID in the ransom note + a 16-byte random number, it is also estimated that the “instance_key” field could potentially be used for this value.

Field	Description
Header marker	Header Marker (AD DE)
build_key	Encryption key for log files
instance_key	Random Value
custom_icon	Whether to change the icon of encrypted files
custom_wallpaper	Whether to change the desktop background
custom_extension	Encrypted file extension
time_sync	System time synchronization status
encryption_mode	Encryption Mode (all, local, backup)
full_encrypt_threshold	Full Encryption Target File Size Threshold
header_encrypt_threshold	Header Encryption Target File Size Threshold
header_encrypt_size	Header Encryption Size
other_encrypt_chunk_percent	Partial Encryption Ratio
encrypt_file_name	Changing the filename of an encrypted file (Base32)
schedule_job	Whether the task is registered in the Task Scheduler
job_executable	Path to the executable file to be registered in the Task Scheduler
job_title	Task Scheduler Name
job_description	Task Scheduler Description
job_start	Task Scheduler Start Time
kill	Whether to terminate the process
use_sys	Whether to utilize the BYOVD technique
priority	List of processes to be terminated
whitelist	Whether it is on the list of items excluded from encryption
path	Folders excluded from encryption
ext	Extensions excluded from encryption
filename	Files excluded from encryption

Table 3. Config Information



Figure 12 SID provided by the DragonForce negotiation site

4. Check CommandLine Parameter

DragonForce ransomware supports five executable arguments, with the supported arguments and their functions as follows. Among these, the encryption mode specification argument (**-m**) has four modes available. However, when the “**backups**” argument is passed, no separate process occurs, suggesting this mode is either under testing or still in development.

Parameter	Description
-p	Encryption Target Path
-m(all, local, net, backups)	Encryption Mode Specification
-log	Log File Path
-size	Encryption Ratio
-nomutex	Allow Duplicate Execution

Table 4. Execution Parameters

5. Logging

DragonForce ransomware supports logging functionality for its progress and configuration information, regardless of whether the “-log” execution argument is present. If no log file path is provided via the “-log” argument, log contents are saved to the path “C:\Users\Public\log.log”. If a file path is provided via the “-log” argument, log contents are saved to that file. In this case, log contents are encrypted using the ChaCha8 algorithm before storage. The log content format and example are as follows.

- Log Format: [HH:MM:SS] [Th: <Thread ID>] Message
- Log Example: [12:23:44] [Th:05FC] custom_icon: 1

6. Running Time synchronize

If the “time_sync” field in the Config information is enabled, it performs the task of synchronizing the victim system time using W32TimeSyncNow().

7. Mutex

If the “-nomutex” argument is not passed as an execution parameter, a mutex is created to prevent duplicate execution. The mutex name is also decrypted and is as follows. According to [SANGFOR](#) and [SK Shieldus](#), this mutex name has been used previously in Meow and LockBit Green ransomware, both of which are known to be Conti-based ransomware.

Mutex Name: hsfjuukjzloqu28oajh727190

8. Job Scheduler

When the “schedule_job” field in the Config information is enabled, the process of registering a job that runs with SYSTEM privileges to the scheduler is performed. At this time, the executable file path to register with the scheduler, the scheduler time (hour, minute), and the job scheduler name are set to the values defined in the Config field. If a job with the same name already exists, it is deleted and a new one is created.

9. Kill Process

DragonForce ransomware uses two methods to terminate predefined processes. The first method utilizes the BYOVD (Bring Your Own Vulnerable Driver) technique, exploiting vulnerable drivers, and operates when the “use_sys” field is enabled. If the “use_sys” value is 1, it uses the “truesight.sys” driver; if it is 2, it uses the “rentdrv2.sys” driver. It then passes the control code (IOCTL) defined for that driver as an argument to DeviceIoControl() to forcefully terminate the specified process. According to [SK Shieldus](#), truesight.sys contains a vulnerability in versions v3.4.0 and below that allows arbitrary process termination. It has also been confirmed that rentdrv2.sys contains a vulnerability that can terminate arbitrary processes.

Device Control Code- truesight.sys: 0x22E044- rentdrv2.sys: 0x22E010

The second method uses NtQuerySystemInformation() to enumerate the system's process list. If a process defined in the "priority" field is running, it calls TerminateProcess() to terminate it.

The processes targeted for termination are defined in the "priority" field. Based on the sample analyzed in this report, the following processes were identified as termination targets.

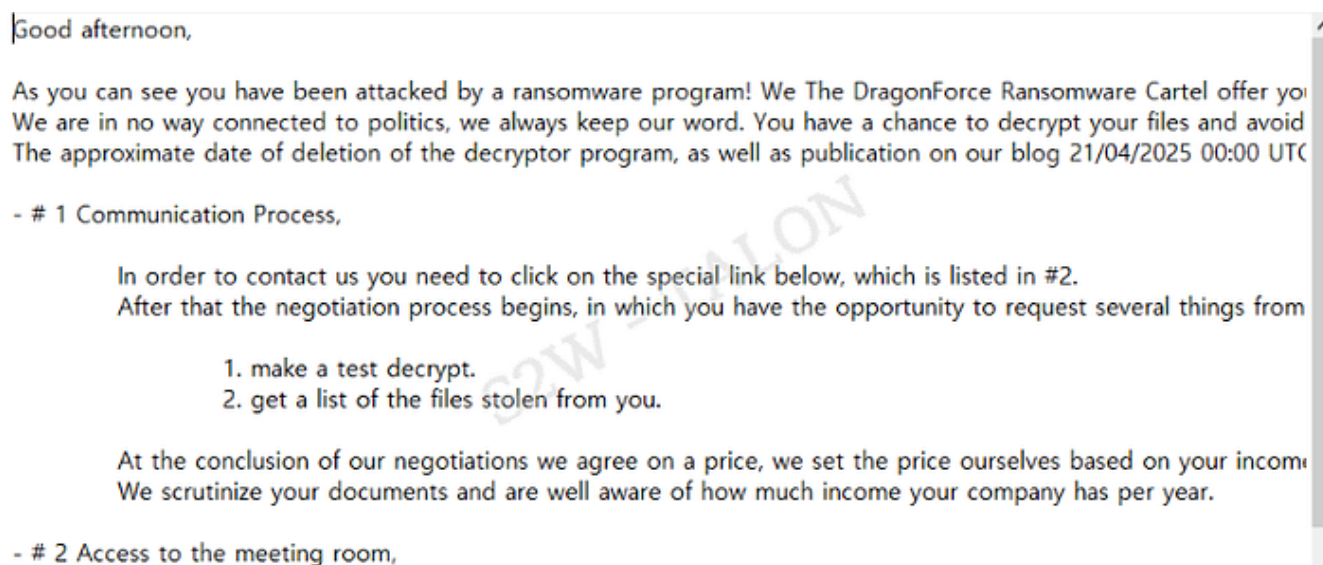
Terminate Process List: MsMpEng.exe, sql.exe, oracle.exe, ocssd.exe, dbsnmp.exe, synctime.exe, agntsvc.exe, isqlplussvc.exe, xfssvcon.exe, mydesktopservice.exe, ocautoupds.exe, encsvc.exe, firefox.exe, tbirdconfig.exe, mydesktopqos.exe, ocomm.exe, dbeng50.exe, sqbcoreservice.exe, excel.exe, infopath.exe, msaccess.exe, mspub.exe, onenote.exe, outlook.exe, powerpnt.exe, steam.exe, thebat.exe, thunderbird.exe, visio.exe, winword.exe, wordpad.exe, notepad.exe, calc.exe, wuauclt.exe, onedrive.exe, SQLAGENT.exe, sqlservr.exe, SQLWriter.exe

10. Directory Traversal

Before performing encryption, a directory traversal is conducted to identify files to be encrypted. If the "whitelist" field is disabled, encryption is performed on all files. If enabled, only files in the scan are compared against the exclusion list.

Excluded List- Directory: tmp, winnt, temp, thumb, \$Recycle.Bin, \$RECYCLE.BIN, System Volume Information, Boot, Windows, perflogs, Public- Extension: .exe, .dll, .lnk, .sys, .msi, .bat, .dragonforce_encrypted- File: readme.txt

During this process, a ransom note is created in each targeted directory, and both the ransom note filename and its content are also decrypted for use.



Good afternoon,

As you can see you have been attacked by a ransomware program! We The DragonForce Ransomware Cartel offer you
We are in no way connected to politics, we always keep our word. You have a chance to decrypt your files and avoid
The approximate date of deletion of the decryptor program, as well as publication on our blog 21/04/2025 00:00 UTC

- # 1 Communication Process,

In order to contact us you need to click on the special link below, which is listed in #2.
After that the negotiation process begins, in which you have the opportunity to request several things from

1. make a test decrypt.
2. get a list of the files stolen from you.

At the conclusion of our negotiations we agree on a price, we set the price ourselves based on your income.
We scrutinize your documents and are well aware of how much income your company has per year.

- # 2 Access to the meeting room,

Figure 13. DragonForce Ransomware's ransom note

11. Network Encryption

When net is passed as an argument to “-m”, network encryption is performed. First, the ARP table is queried using GetIpNetTable(), and only IP addresses within the corresponding range are filtered.

Filtering IP: 172., 192.168., 10., 169.

After filtering the IP addresses, it attempts an SMB (TCP/445 port) connection using IOCP. If the connection succeeds, it enumerates shared drives and directories using NetShareEnum(). Only shares of the temporary/special/disk type are included as encryption targets, and paths containing ADMIN\$ are excluded. All network paths matching the encryption criteria are passed to the encryption thread for processing.

12. Encryption

File encryption is performed through an encryption thread, and the number of threads generated varies depending on the encryption mode (**all**, **net**, **local**).

- all: Create encryption threads equal to the number of CPU cores
- net, local: Create (number of CPU cores * 2) encryption threads

DragonForce ransomware performs full encryption, header encryption, or partial encryption depending on the target file's extensions and size. For files with extensions associated with virtual machines or disk images, it encrypts only 20% of the file. In contrast, files related to databases are fully encrypted.

- 20% Encryption Extension: .vdi, .vhd, .vmdk, .pvm, .vmsn, .vmsd, .nvram, .vmx, .raw, .qcow2, .subvol, .bin, .vsv, .avhd, .vmrs, .vhdx, .avdx, .vmcx, .iso
- Full Encryption Extension: .4dd, .4dl, .accdb, .accdc, .accde, .accdr, .accdt, .accft, .adb, .ade, .adf, .adp, .arc, .ora, .alf, .ask, .btr, .bdf, .cat, .cdb, .ckp, .cma, .cpd, .dacpac, .dad, .dadiagrams, .daschema, .db, .db-shm, .db-wal, .db3, .dbc, .dbf, .dbs, .dbt, .dbv, .dbx, .dcb, .dct, .dcx, .ddl, .dlis, .dp1, .dqy, .dsk, .dsn, .dtsx, .dxl, .eco, .ecx, .edb, .epim, .exb, .fcd, .fdb, .fic, .fmp, .fmp12, .fmpl, .fol, .fp3, .fp4, .fp5, .fp7, .fpt, .frm, .gdb, .grdb, .gwi, .hdb, .his, .ib, .idb, .ihx, .itdb, .itw, .jet, .jtx, .kdb, .kexi, .kexic, .kexis, .lgc, .lwx, .maf, .maq, .mar, .mas, .mav, .mdb, .mdf, .mpd, .mrg, .mud, .mwb, .myd, .ndf, .nnt, .nrmlib, .ns2, .ns3, .ns4, .nsf, .nv, .nv2, .nwdb, .nyf, .odb, .oqy, .orx, .owc, .p96, .p97, .pan, .pdb, .pdm, .pnz, .qry, .qvd, .rbf, .rctd, .rod, .rodx, .rpd, .rsd, .sas7bdat, .sbf, .scx, .sdb, .sdc, .sdf, .sis, .spq, .sql, .sqlite, .sqlite3, .sqldb, .te, .temx, .tmd, .tps, .trc, .trm, .udb, .udl, .usr, .v12, .vis, .vpd, .vvv, .wdb, .wmdb, .wrk, .xdb, .xld, .xmlff, .abccdb, .abs, .abx, .accdw, .adn, .db2, .fm5, .hjt, .icg, .icr, .kdb, .lut, .maw, .mdn, .mdt

If the file extension of the targeted file does not match any of the extensions listed above, the encryption method is determined based on the “full_encrypt_threshold” and “header_encrypt_threshold” values. For the samples analyzed in this report, the “full_encrypt_threshold” is set to 2MB and the “header_encrypt_threshold” is set to 10MB. This means: For header encryption, the first 3MB is encrypted. For partial encryption, the portion corresponding to the percentage set by the “other_encrypt_chunk_percent” value is encrypted. For partial encryption, the chunk size and skipped intervals are as follows.

- Partial Encryption Chunk Count: 3
- Partial Encryption Chunk Size: $(\text{File Size} / 100) * \text{Encryption Ratio}$
- Partial Encryption Skip Size: $(\text{File Size} - \text{Chunk} * 3) / 2$

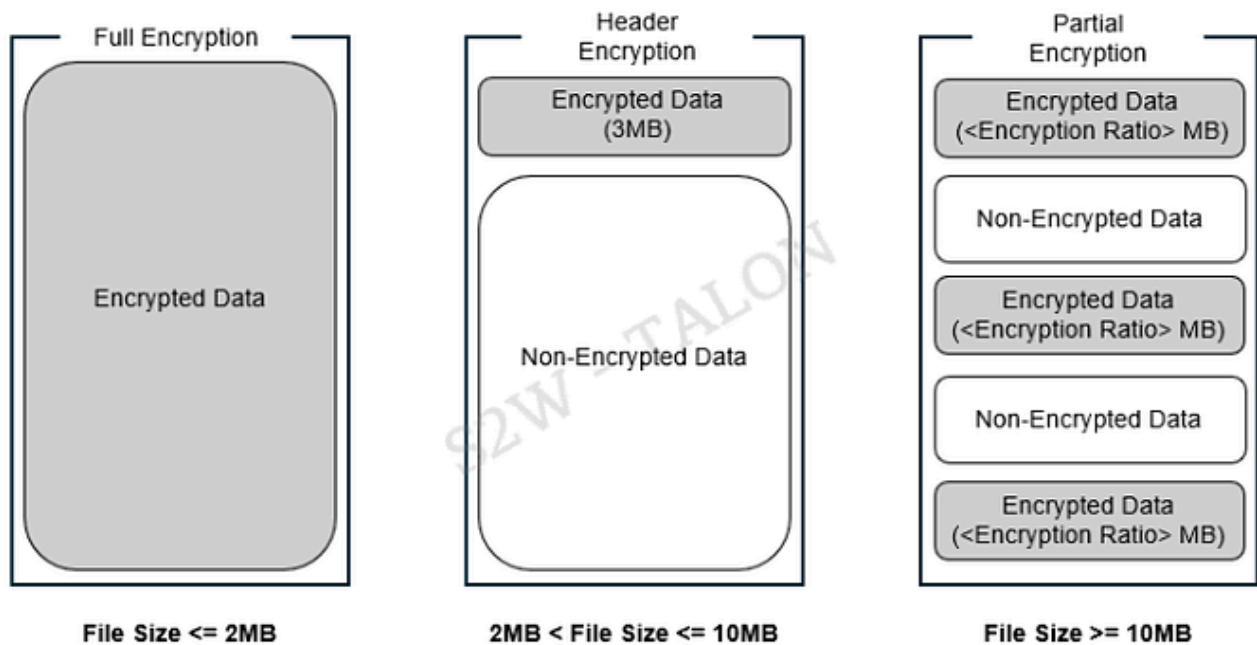


Figure 14. Encryption Methods by File Size

After determining the encryption method, before performing the actual encryption process, metadata of 534 bytes is appended to the end of the target file. The metadata consists of a Blob encrypted using the RSA-4096 algorithm containing a 524-byte ChaCha8 key and Nonce information, a 1-byte encryption type constant (0x24: full encryption, 0x25: partial encryption, 0x26: header encryption), a 1-byte encryption ratio value, and finally an 8-byte original file size information.

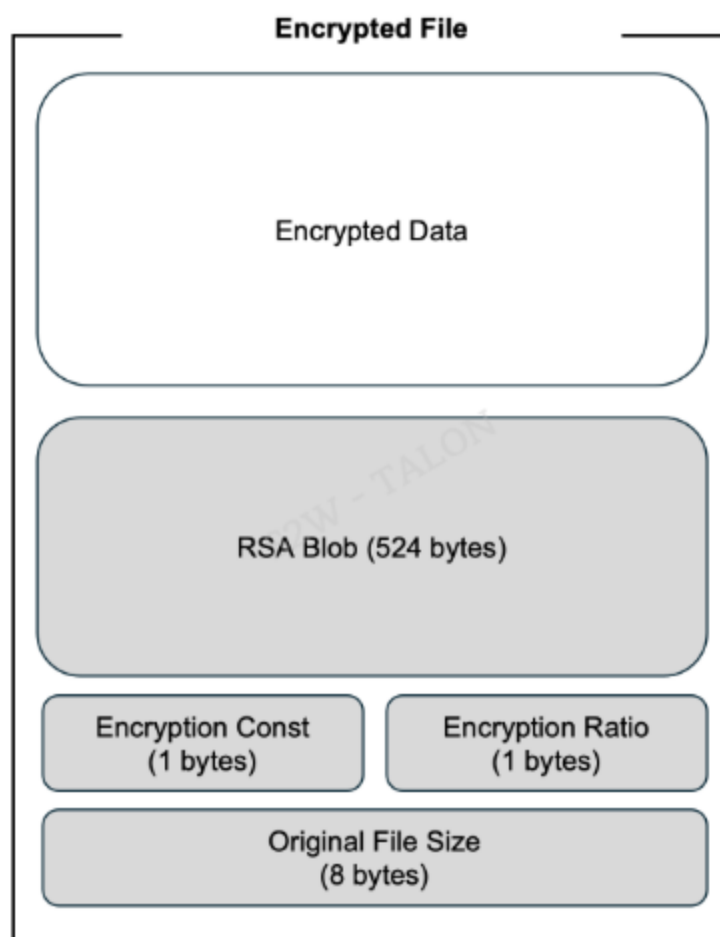


Figure 15. Metadata added to encrypted files

The target files are ultimately encrypted using the ChaCha8 algorithm, and the key used for encrypting each file is generated via `CryptGenRandom()`.

13. Change Filename & Add Extension

After the encryption process completes, the filename is modified based on whether the “`encrypt_file_name`” field is enabled. If this field is disabled, only the extension is appended to the encrypted file. If enabled, the original filename is replaced with a Base32-encoded string. At this point, encoding is performed using the string “**gwfn6l3bk45o2zecvi7xtyqrpsudmahj**” as the basis, rather than standard Base32 encoding. After changing the original filename, the extension identifying the encrypted file is appended.

Encrypted File Extension: `.dragonforce_encrypted`

14. Change File Icon & Background

When the “`custom_icon`” field is enabled, DragonForce ransomware not only changes the extension of encrypted files but also modifies their file icons. The image used for the file icon exists in compressed form within the binary and is decompressed using the `zlib` library. The

decompressed icon image is saved as the file “C:\Users\Public\icon.ico”. The icon file is then added to the registry to automatically change the icon for files with the .dragonforce_encrypted extension.

- ICON Registry Path:- HKCR\.dragonforce_encrypted\DefaultIcon (With administrator privileges)- HKCU\.dragonforce_encrypted\DefaultIcon
- Registry Value: C:\Users\Public\icon.ico- ICON File dhash: 060e1f3383066561



Figure 16. Icon of an encrypted file

If the “custom_wallpaper” field is enabled, the desktop background is also changed. The desktop image is decompressed and loaded in the same manner as icon images, and is stored at the path C:\Users\Public\wallpaper_white.png. Subsequently, if the token of the currently running process has administrator privileges, it accesses the registry key to modify the desktop-related setting value (WallPaper). If it does not have administrator privileges, it uses SystemParametersInfo() to change the desktop only for the current user account.

Background Registry Path:- HKEY_USERS\<SID>\Control Panel\Desktop\Wallpaper- Value: C:\Users\Public\wallpaper_white.png- HKEY_USERS\<SID>\Control Panel\Desktop\WallpaperStyle- Value: 10(Fill)

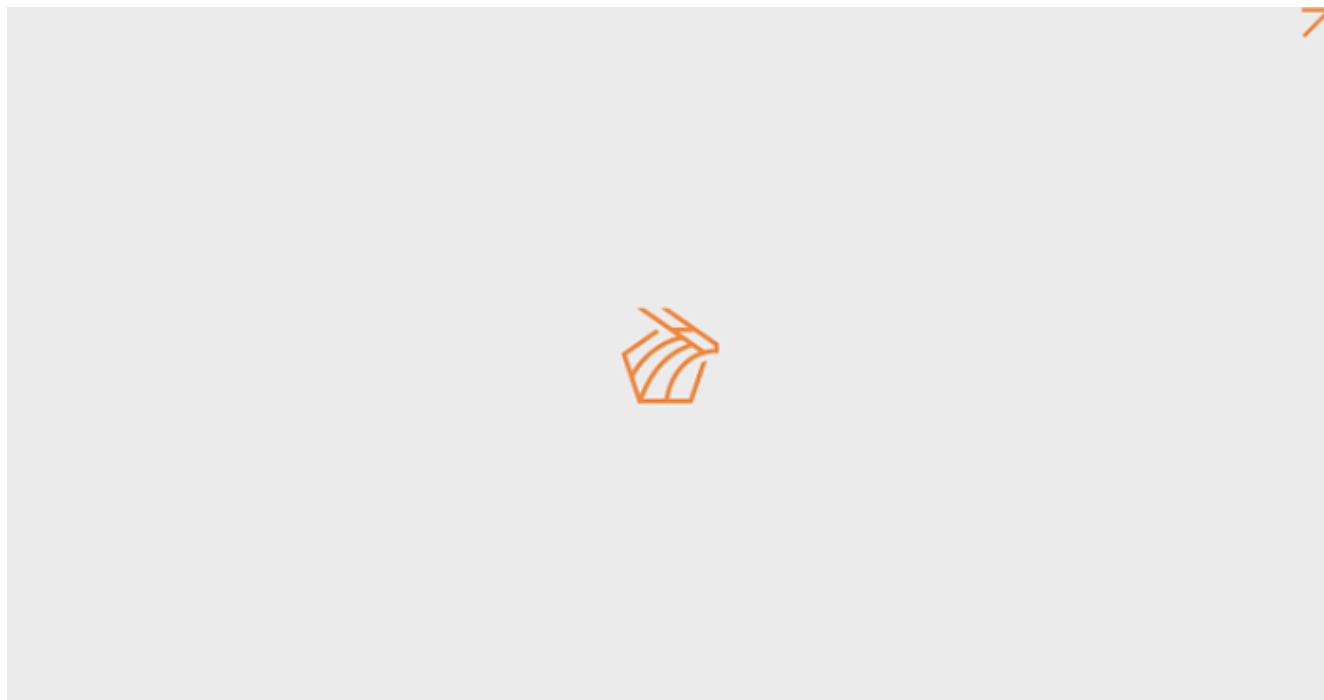


Figure 17. Decompressed desktop Image

15. Delete Volume Shadow Copy

Finally, the malware deletes the volume shadow copies. To do so, it uses a WQL query to enumerate the volume shadow copies present on the victim system and then passes a command to delete them as an argument to `CreateProcess()`.

- WQL Query: `SELECT * FROM Win32_ShadowCopy`
- Command: `cmd.exe /c C:\Windows\System32\wbem\WMIC.exe shadowcopy where "ID='%s'" delete`

Code Base: Conti

The sample analyzed in this report was identified as DragonForce ransomware based on Conti. The functionally identical aspects between Conti ransomware and DragonForce ransomware are as shown in Table 5.

Join Medium for free to get updates from this writer.

DragonForce ransomware used a mutex with the same name as that employed by Conti ransomware. Both ransomware variants supported five executable arguments, and the functionality and encryption mode (**-m**) for each argument were identical. The network share encryption functionality also followed the same process as Conti ransomware: it queried the ARP table, filtered IPs within the private IP range, enumerated the list of shares, and excluded the ADMIN\$ directory.

The encryption process was also largely identical across both ransomware strains, including the exclusion list, file extensions and sizes for full/header/partial encryption, encryption algorithms, and the structure of added metadata. However, differences existed: the exclusion list in DragonForce ransomware omitted Trend Micro and CONTI_LOG.txt, which were present in Conti ransomware. Additionally, while the ransom note filenames were identical for both ransomware variants, the ransom note data itself was structured differently.

(*For the Conti ransomware ransom note, see Appendix D)

Function	Description
Mutex	- Uses the mutex name employed by Conti-family ransomware - Mutex Name: hsfjuukjzloqu28oaJh727190
CommandLine Parameter	- Supports 5 command-line arguments and 3 encryption modes - Command-line arguments: -p, -m (local, net, all), -log, -size, -nomutex
Network Share Encryption	- After querying the ARP table, filter only IP addresses corresponding to private IP ranges (172., 192.168., 10., 169.) - Attempt SMB connections to the filtered IP addresses, excluding ADMIN\$
Encryption Thread	- For network shared encryption only, it creates CPU core count * 2 encryption threads; for local system encryption, it creates encryption threads corresponding to the CPU core count.
Encryption Excluded List	- The list of directories excluded from encryption is identical, except for the Trend Micro directory (Trend Micro exists only in Conti ransomware) - The file extensions and files excluded from encryption are identical, except for the CONTI_LOG.txt file (CONTI_LOG.txt exists only in Conti ransomware)
File Encryption	- The algorithm used for file encryption is ChaCha8 + RSA4096, which remains consistent - The conditional statements are identical: full encryption for database-related extensions, 20% partial encryption for virtual machine and disk-related extensions. The process of performing full, header, or partial encryption based on file size is also identical. (However, Conti sets file size thresholds at 1MB and 5MB, while DragonForce sets them at 2MB and 10MB) - The size and structure of the metadata appended to encrypted files are identical. - Metadata size: 534 bytes - Metadata structure: RSA Blob (524 bytes) + Encryption mode constant (1 byte) + Encryption ratio (1 byte) + Original data size (8 bytes)
Ransomnote	- The ransom note filenames (readme.txt) for both ransomware strains are identical.

Table 5. Identical Features Between DragonForce Ransomware and Conti Ransomware

Detailed Analysis of DragonForce Decryptor

S2W Threat Research and Intelligence Center TALON obtained a decryptor for DragonForce ransomware during the process of hunting threat intelligence related to the DragonForce group. This decryptor is capable of decrypting specific victim system data. The obtained decryptor consists of one Windows decryptor and three ESXi decryptors.

FileName	Description
Windows-*****_****_****_*****	Decryptor for Windows
ESXi-*****_****_****_*****	Decryptor for VMware ESXi
ESXi-*****_****_****_*****	Decryptor for VMware ESXi
ESXi-*****_****_****_*****	Decryptor for VMware ESXi

Table 6. Decryption Tool Information for the Obtained DragonForce Ransomware

The overall decryption process of the DragonForce ransomware decryptor is identical for both the Windows and ESXi versions and proceeds as follows:

1. Decrypt Config Data
2. Identifying encrypted files through file extensions or magic bytes
3. RSA Blob Decryption
4. Session Key & Nonce Parsing
5. Encrypted file Decryption

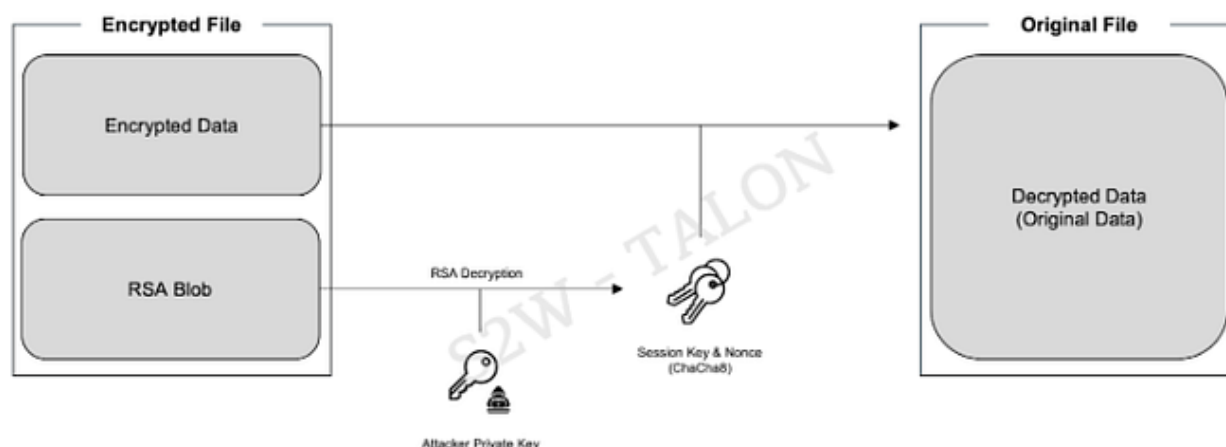


Figure 18. DragonForce Ransomware Decryption Process

1. Windows Decryptor

- SHA256: dc7e706587d4897789cc4a5f7ccbb539646b58aa9c86272728c8c1e6ec2a529
- MD5: a368564c74d7f288fa01e4c7241082fe

1.1 Config Decrypt

Similar to the operation process of DragonForce ransomware, upon execution, it decrypts the Config information embedded within the binary using the ChaCha8 algorithm. This is judged to be a process for loading the encryption mode, partial encryption ratio, and header encryption ratio values, in order to perform the decryption process corresponding to the encryption mode and ratios.

1.2. RSA Private Key Load

After decrypting the configuration information, `CryptImportKey()` is used to load the attacker's RSA-4096 private key blob. The loaded RSA private key is used to decrypt the metadata appended to the encrypted file.

1.3. Directory Traversal

Subsequently, the decryption process involves performing directory traversal to identify target files for decryption. In the case of the decryptor analyzed in this report, it was confirmed that the encrypted files have the .RNP extension, and decryption is performed only on files with the .RNP extension.

1.4. Decryption

To decrypt an encrypted file, the decryptor first accesses the target file using CreateFileW(). It then reads the metadata (537 bytes) located at the end of the encrypted file and decrypts it using an RSA private key. Through this process, the session key, nonce, encryption mode, encryption ratio, and original file size used during encryption are extracted. After all required values are parsed, the encrypted data is decrypted using the ChaCha8 algorithm.

1.5. Network Share Decryption

1. After querying the ARP table using GetIpNetTable(), filter only the private IP ranges (172., 192.168., 10., 169.) included in the network share encryption targets.
2. Attempt SMB connections to the filtered IP ranges. If a connection is established successfully, enumerate the shared directories and drives.
3. Register the enumerated directories and drives in the decryption task list and perform decryption.

2. ESXi Version

Three decryptors for ESXi were obtained, but aside from the filenames, all functional aspects were identical. The reason three separate decryptors exist for ESXi is that each encrypted system is assigned a unique “build_key” (8 bytes), and it is believed that a dedicated decryptor matching that key is provided individually.

- “build_key” Example: 0x12345678
- “build_key” Format: \b0x([0-9a-fA-F]{16})\b

Number	SHA256	MD5
1	4f29100b8f788ac937b4ef3689bfb159237d138e7615a2c6907e9dfeddac051c	3222c643933a09539f61d4b089130b7e
2	dfc472b2bcae060fc7f06bbbd39617632c1e74afc119cb946c423a9be3277661	f73e9eca0dfc7ab01800d9ef658916c1
3	d545012bf969c84d19aaa0c1789dbd753991b8e61e2f5440d93b1b3a45f808d5	21ea10c49500c6ba820779833402ac22

Table 7. Decryptor Information for ESXi

2.1. Config Decrypt

It decrypts the configuration information embedded within the binary using the ChaCha8 algorithm, identical to the decryptor for Windows.

2.2. Execution Parameter Check

The decryptor for ESXi differs from the Windows decryptor in that it includes a process for verifying execution parameters. During this process, 15 execution parameters were identified. However, considering the presence of features like test mode, background execution, and delayed execution, these are judged to be parameters used by the encryptor rather than parameters specifically for the decryptor. The details are as follows.

Parameter	Description
-e	Encryption mode determination, specifying encryption method and ratio via trailing parameter • p: Partial encryption at a ratio specified by the attacker • s: Partial encryption using the configured size as the chunk size • h: Header encryption
-n	Test Mode
-d	Daemon(Background) Execution
-h	Delay Execution Time(Hours)
-m	Delay Execution Time(Minutes)
-s	Delay Execution Time(Seconds)
-p	Path for encryption and decryption targets
-l	Log file Path
-j	Number of threads to create
-q	Enable console
-v	Output detailed logs
-nv	Whether to exclude from encryption and decryption
-ds	Not verified
-wvi	VM IDs to exclude from encryption and decryption
-wvn	Names to exclude from encryption and decryption

Table 8. Decryptor Execution Parameters for ESXi

2.3. RSA Private Key Load

After decrypting the config, it loads the attacker's RSA private key hardcoded into the binary. This is also to decrypt the ChaCha8 session key and nonce value contained within the metadata.

2.4. Traversal Directory

“-p” Performs a directory traversal process to identify files for decryption based on the path passed as an argument. Internally, it uses scandir() to traverse each directory, checking for the presence of files with the .RNP_esxi extension. Additionally, if a file named readme.txt (the ransom note) exists in the traversed path, it deletes the ransom note using unlink().

2.5. Decryption

Before decrypting a file with the .RNP_esxi extension, the last 8 bytes of the file are read to verify if they match the “build_key” within the configuration information. Decryption is performed only if they match. Subsequently, the metadata (512 bytes) present at the end of the encrypted file is read. This metadata is used to extract the ChaCha8 session key and nonce using the RSA private key, enabling the decryption of the encrypted data. After completing the decryption process for the encrypted data, the metadata is removed to restore the file size to its original dimensions. The file is then restored to its original filename using the rename() function.

Conclusion

- This report analyzes the DragonForce ransomware based on the Conti ransomware and its decryptor.
- DragonForce ransomware decrypts configuration information encrypted with ChaCha8 and utilizes it during execution.
- It performs full/header/partial encryption depending on file size. Target files are encrypted using the ChaCha8 algorithm, and 534 bytes of metadata are appended to the end of the encrypted file.- Modifies the original filename (Base32 encoded), file icon, and desktop background based on the Config settings.
- Since DragonForce ransomware performs initial penetration through publicly exposed remote desktop servers, restricting access to remote desktop servers or allowing only VPN-based access is recommended.
- The DragonForce decryptor supports decryption for Windows and ESXi environments.- For Windows, decryption is performed only on files with the .RNP extension.- For ESXi, decryption is performed only on files with the .RNP extension that also match the magic bytes.
- The metadata at the end of encrypted files contains the ChaCha8 session key encrypted with the RSA public key. Therefore, decrypt the metadata using the RSA private key present in the decryptor, then use the extracted session key to decrypt the files.

Appendix A. IoCs

| IoC list can be found our [github](#)

Appendix B. Detection Rules

| Detection Rule can be found our [github](#)

Appendix C. MITRE ATT&CK

Initial Access

(T1078) Valid Accounts

Execution

- (T1053) Scheduled Task/Job
- (T1059.001) PowerShell

Persistence

- (T1078.002) Domain Accounts
- (T1543.003) Windows Service
- (T1547.001) Registry Run Keys / Startup Folder

Privilege Escalation

(T1068) Exploitation for Privilege Escalation

Defense Evasion

- (T1027) Obfuscated Files or Information
- (T1027.007) Dynamic API Resolution
- (T1140) Deobfuscate/Decode Files or Information
- (T1070.001) Clear Windows Event Logs

Credential Access

(T1003.001) LSASS Memory

Discovery

- (T1016) System Network Configuration Discovery
- (T1018) Remote System Discovery
- (T1082) System Information Discovery
- (T1083) File and Directory Discovery
- (T1135) Network Share Discovery
- (T1482) Domain Trust Discovery

Lateral Movement

(T1021.001) Remote Desktop Protocol

Collection

(T1560.001) Archive via Utility

Command and Control

(T1071.001) Web Protocol

Impact

- (T1486) Data Encrypted for Impact
- (T1490) Inhibit System Recovery

Appendix D. Conti Ransomware ransom note



readme.txt - 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

All of your files are currently encrypted by CONTI ransomware.
If you try to use any additional recovery software - the files might be damaged or lost.

To make sure that we REALLY CAN recover data - we offer you to decrypt samples.

You can contact us for further instructions through:

Our website
TOR VERSION :
(you should download and install TOR browser first <https://torproject.org>)
[http://contirec\[REDACTED\]5oad.onion/](http://contirec[REDACTED]5oad.onion/)
HTTPS VERSION :
<https://contirecovery.top/>

YOU SHOULD BE AWARE!
Just in case, if you try to ignore us. We've downloaded your data and are ready to publish it on out

---BEGIN ID---
[REDACTED]
---END ID---

Figure 19. Conti Ransomware ransom note