DragonForce Ransomware: Unveiling Its Tactics and Impact | Idan Malihi

Archived: 2026-04-05 14:14:01 UTC

Evolution of Cyber Threat Groups

Introduction – DragonForce

DragonForce's Cyber Activities Against Israel

The group's attacks are primarily politically motivated; however, some of its activities extend across various industries and regions. It has claimed responsibility for a wide range of attacks, including DDoS attacks and data breaches.

DragonForce's Leak Site and Propaganda Strategy

A notable example from this research highlighted a negotiation attempt between a victim company and DragonForce. The company sought to reduce the ransom amount, but the group refused to make any concessions. They emphasized the urgency of the countdown timer and warned that failure to pay would lead to the full disclosure of the stolen data. By making these negotiations public, DragonForce sends a clear message to future victims: negotiations are futile, and the only way to prevent exposure is to pay the ransom in full.

Once the countdown expires, the stolen files become publicly accessible directly from the site. The files are neatly categorized, allowing anyone to browse through the leaked documents without needing to download them. The exposed data often includes financial records, employee and customer information, internal communications, contracts, and other highly sensitive documents. This structured data exposure model adds pressure, reinforcing DragonForce's ultimatum: **comply or face full-scale data disclosure**.

Dissecting the Threat: DragonForce Ransomware

The ransomware's entropy analysis suggests that only the overlay is packed. This indicates that the file itself is not actually packed; instead, it conceals some data within the binary inside the overlay. Essentially, this means that the compressed data is hidden at the end of the file (EOF).

I utilized `Binwalk` to analyze the binary ransomware and identify the various types of data concealed within it, thereby gaining insight into what files the attacker was attempting to hide from researchers and analysts. It

contains an XML document that serves as the manifest file. Finally, there are two compressed files that the ransomware will utilize during its runtime.

To extract both compressed files, I executed a Python script that reads and extracts the data located at the `0x66400` and `0x66D24` offsets.

The compressed files are graphic files used in the ransomware's operations when it drops files related to the ransomware during runtime.

`firstFile.ico: a0bbc666c39f80d6ac18ae1b253c3462`

`secondFile.png: 07fb997df804901c7f09bcce85ec2c05`

In addition, the ransomware encrypts the victim's files using the symmetric `ChaCha20` encryption method with a key of 256 bytes.

The strings below indicate that ransomware utilizes the `ChaCha20` encryption method. These strings relate to the key expansion process in the `ChaCha20` stream cipher, which is typically used for fast and secure encryption.

The following strings indicate that the ransomware uses logs printed during runtime to determine the status of its operations, including file name changes, run path, privilege escalation, desktop wallpaper changes, and more.

The string `SELECT * FROM Win32_ShadowCopy` shows that the ransomware is using WMI to query Volume Shadow Copies, which disables the victim's ability to restore backups.

The following strings relate to the extensions that the threat actor configures for the ransomware, specifying which extensions to encrypt on the victim's system.

The ransomware appears to be conducting reconnaissance on network shares and facilitating lateral movement to locate and encrypt files stored on shared network drives.

The ransomware generates a log file named `log.log` located in the `C:\Users\Public` directory to track its operations.

The ransomware configures which file extensions it will not encrypt, such as .exe, .dll, .lnk, etc.

The `421570` subroutine is responsible for the encryption process configuration. It seems that the ransomware configures dynamic configurations for efficient encryption, process termination, and system disruption. The presence of keys such as `encrypt_mode`, `full_encrypt_threshold`, and `encrypt_file_names` suggests that the ransomware employs selective encryption techniques to optimize the speed of the encryption process. Additionally, strings like `custom_wallpaper` and `custom_extension` indicate that the malware modifies the desktop wallpaper and changes the files' extension. Furthermore, the ransomware targets several system and database processes to terminate them and encrypt them, such as:

At the end of the configurations, it writes to the `log.log` file.

In the following flow, the ransomware retrieves the current process ID using the `GetCurrentProcessId` function, which is then used in the `42A7A0` subroutine.

In the subroutine, the ransomware uses the `OpenProcess` function with the `0x400` parameter, which is related to the `PROCESS_QUERY_INFORMATION` access rights. This suggests an attempt to query the process's security context. If the function's execution is successful, it jumps to the `42A9E5` memory location, where it calls `OpenProcessToken` and `GetTokenInformation` to extract the security details about the running process.

Additionally, it calls `LookupAccountSidW` to resolve the account name associated with the SID (security identifier). The ransomware might determine whether it is executing under a privileged user. Then, it writes in the logs the privileges execution as a `"Running under: %s"` string.

The configuration of DragonForce Ransomware shows similarities to the LockBit builder that was leaked in 2022. Key settings observed include `encrypt_mode`, `local_disks`, `network_shares`, `kill_processes`, `kill_services`, `set_wallpaper`, and `set_icons`, all of which match the structure of LockBit's leaked `config.json`. This suggests that DragonForce may be a modified version of the original LockBit builder. Additionally, DragonForce incorporates several defense evasion techniques, including anti-forensics parameters like `kill_defender`, `delete_eventlogs`, and `self_destruct`. These features indicate that the ransomware aims to disable security defenses and eliminate forensic evidence. Furthermore, its functionality to alter desktop wallpaper and icons through `set_wallpaper` and `set_icons` is similar to the behavior of LockBit ransomware.

The malware checks and verifies whether the process has been executed with administrative privileges, which is crucial for the ransomware's subsequent operations. The ransomware is attempting to retrieve the security token to determine whether it has administrator or SYSTEM-level privileges. The sequence begins with a call to `GetCurrentProcess`, followed by `OpenProcessToken`, which allows access to the token of the current process. After obtaining the token, it is called `GetTokenInformation`, which is used to extract information about privilege levels, user group details, or integrity levels. The ransomware then evaluates the retrieved token information. If the token handle is valid, the execution continues; if not, the ransomware process terminates. Analysis of the ransomware's runtime value indicates it was executed with high privileges, as shown by the `"Process is elevated: %d"` value being `1`.

The ransomware scans the victim's system for logical drives using `GetLogicalDriveStringsW`. This operation is crucial for the ransomware to identify which drives exist in the victim's system and to infect them during execution.

After the ransomware configures the kill processes list, it looks for processes that are running on the victim's system and compares them to the list. It uses `OpenProcess` to access the `notepad.exe` process and then terminates it using `TerminateProcess`.

The ransomware employs the `NetShareEnum` function to facilitate lateral movement, expanding its encryption capabilities to network shares and NetBIOS. It configures the `servername` parameter to three types of subnets: `172.X.X.X`, `192.168.0.X`, and `169.X.X.X`, which are commonly used subnets in organizations worldwide. The `level` parameter is set to `1`, which, according to Microsoft's documentation, indicates that it will return details about shared resources, including the resource name and type.

The ransomware then decompresses two files from its overlay and drops them into the `C:\Users\Public` folder: `icon.ico` and `wallpaper_white.png`.

It then begins the process of encrypting the victim's files. For each file, the ransomware encrypts its contents, changes the file's name, and modifies the extension using `MoveFileW`.

It then creates `readme.txt` files using the `CreateFileW` function in every directory whose content has been encrypted.

The DragonForce Ransomware ransom note utilizes psychological pressure and clear communication to coerce victims into paying the ransom. It outlines the impact of the attack, the communication process, payment instructions, and the consequences of non-compliance. The note is signed as: 01000100 `01110010 01100001 01100111 01101111 01101110 01000110 01101111 01110010 01100011 01100101` This is the binary representation of "DragonForce," which confirms the ransomware strain responsible for the attack.

MITRE ATT&CK

Yara Rule

Yara Detection
