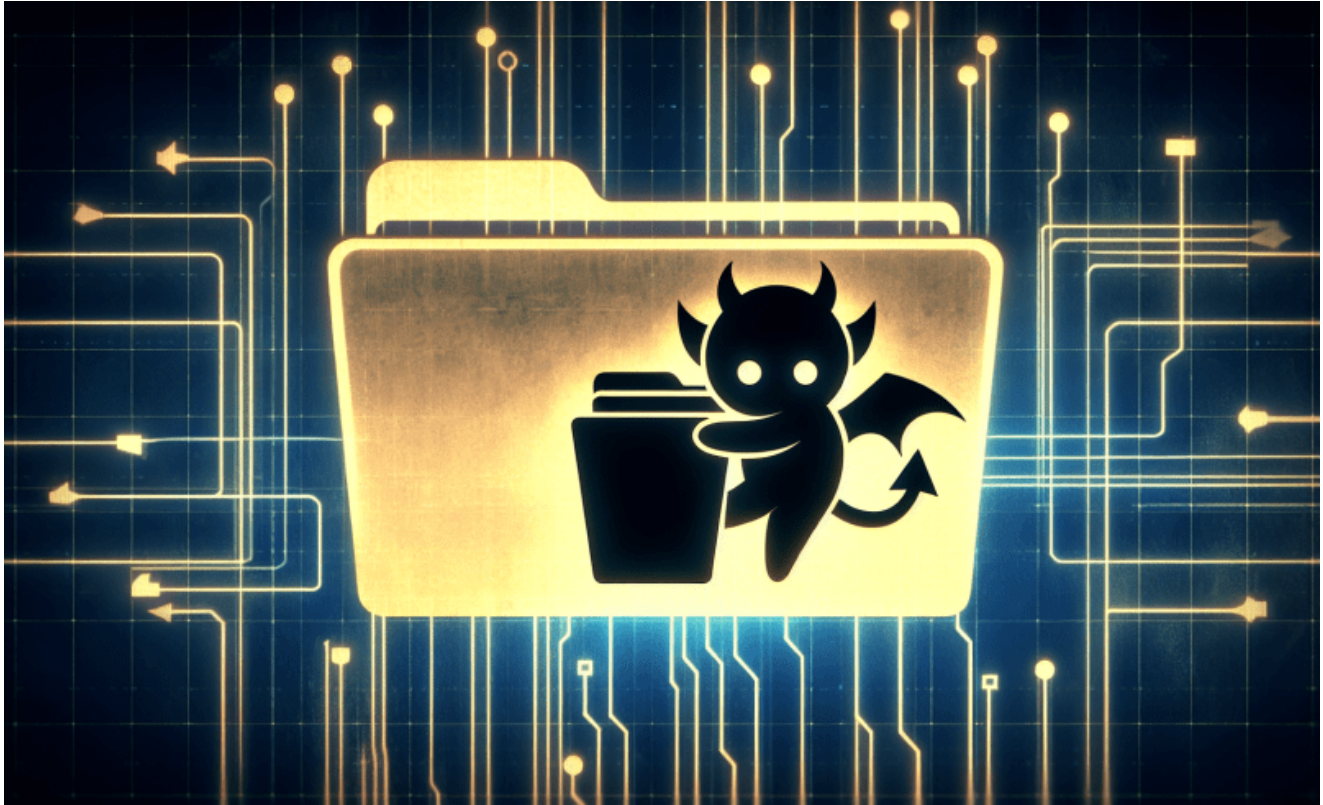


Distribution of PebbleDash Malware in March 2025

A asec.ahnlab.com/en/87621/

April 21, 2025





PebbleDash is a backdoor malware that was previously identified by the Cybersecurity and Infrastructure Security Agency (CISA) in the U.S. as a backdoor malware of Lazarus (Hidden Corba) in 2020. At the time, it was known as the malware of the Lazarus group, but recently, there have been more cases of the PebbleDash malware being distributed by the Kimsuky group, who have been targeting individuals, rather than the Lazarus group. This report will cover the latest distribution process of the PebbleDash malware by the Kimsuky group, other malware and additional modules that have been identified alongside PebbleDash.

As mentioned in multiple TI reports in the past, the Kimsuky threat group is known to use an open-source RDP Wrapper along with PebbleDash for remote control. However, there have been numerous recent cases where the threat actors directly patched `termsrv.dll`, which performs the role of terminal services.

The figure below shows the attack process using the PebbleDash malware by the Kimsuky group.

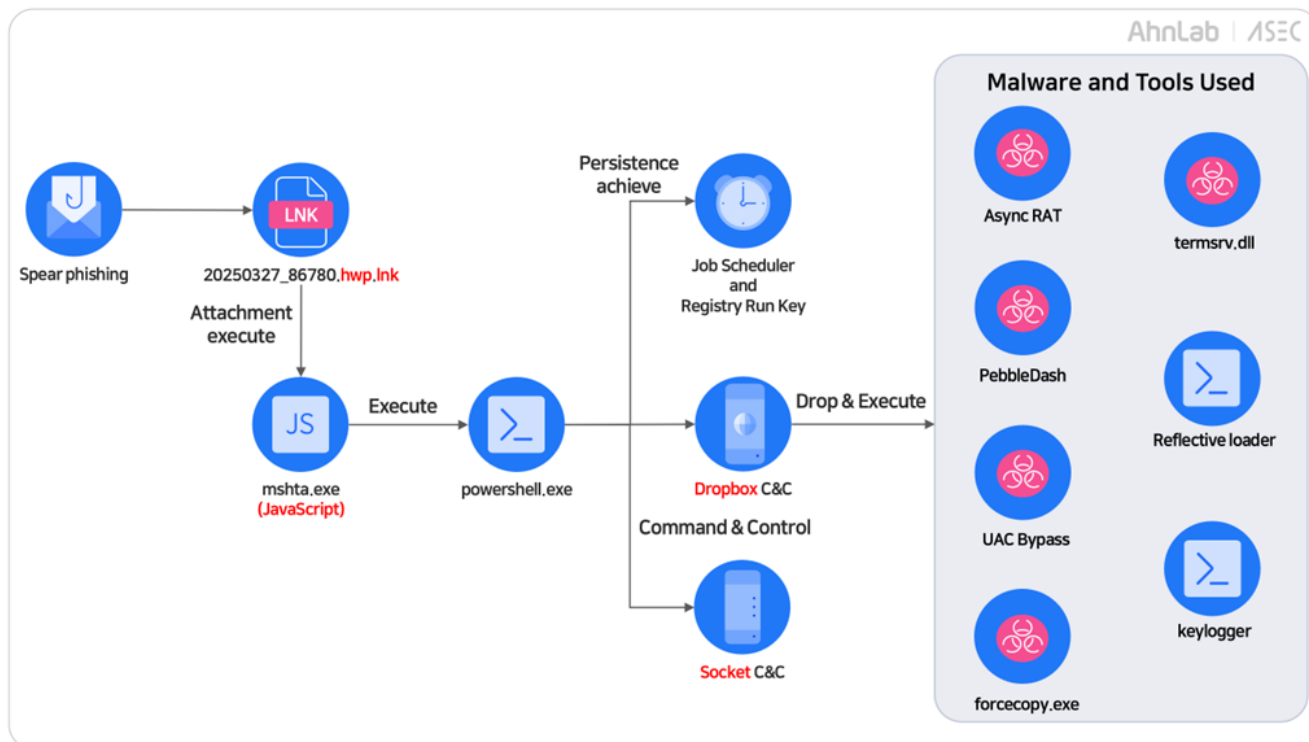


Figure 1. The latest PebbleDash attack process of the Kimsuky group

Attack Process

1. Gaining Initial Access, Maintaining Persistence, and Establishing Foothold

In cases where PebbleDash is used, the threat actor's attack process can be categorized into four main stages: initial access, persistence, establishing a foothold, and creating additional malware. First, the threat actor targets specific individuals with spear-phishing attacks to gain initial access. When a user opens the shortcut file attached to the spear-phishing email, the LNK file executes a JavaScript through the Cmdline. This JavaScript then executes PowerShell to perform tasks such as registering a task scheduler for system persistence, registering registry keys for auto-execution, and performing socket communications with Dropbox and the threat actor's C&C server. This allows the threat actor to create backdoors, RDP tools, and other malware like PebbleDash.

Target Type	File Name	File Size	File Path ⓘ
Current	mshta.exe	36 KB	%SystemRoot%\system32\mshta.exe
LoadedFile	20250327_86780.hwp.lnk	6.46 KB	%SystemDrive%\users\%ASD%\appdata\local\temp\%ASD%\20250327_86780.hwp.lnk
Parent	explorer.exe	2.65 MB	%SystemRoot%\explorer.exe
Target	powershell.exe	444 KB	%SystemRoot%\system32\windowpowershell\v1.0\powershell.exe

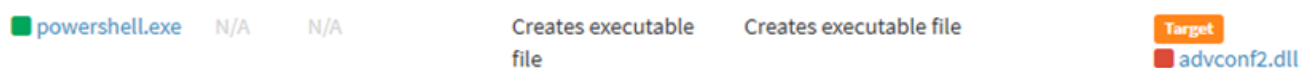
Figure 2. Initial infiltration process

2. Installing Additional Malware for Controlling Infected PCs

When PowerShell is executed by the LNK malware, the threat actor sends additional malware and CMD commands to the infected PC through Dropbox and TCP socket communication. The threat actor uses PebbleDash and AsyncRAT to control the infected PC. The following have been identified: termsrv.dll patched for RDP connection authentication bypass, UAC bypass malware for privilege escalation, and ForceCopy utility for data exfiltration.

2.1. PebbleDash

Since 2021, the PebbleDash malware has been continuously used by the Kimsuky group. There are slight differences in the execution methods between the past and current versions. For example, in 2021, the threat actors executed the bait document file and PebbleDash directly using a PIF file. In the recently identified cases, the threat actors directly created advconf2.dll using PowerShell, as shown in the image below.



A log snippet showing the execution of PowerShell.exe. The process is PowerShell.exe, the module is N/A, and the target is N/A. The behavior is 'Creates executable file', and the data is 'advconf2.dll'. A red 'Target' label is next to the data.

powershell.exe	N/A	N/A	Creates executable file	advconf2.dll
----------------	-----	-----	-------------------------	--------------

Figure 3. Log showing the creation of PebbleDash malware by PowerShell

After advconf2.dll is created, cmd.exe and reg.exe are used to register and execute advconf2.dll as a service. The final executed PebbleDash feature is the same as the one introduced in the AhnLab SEcurity intelligence Center (ASEC) blog in the past.

Process	Module	Target	Behavior	Data
cmd.exe	N/A	reg.exe	Creates process	N/A
powershell.exe	N/A	N/A	Creates executable file	N/A
reg.exe	N/A	N/A	Registered DLL/driver as service	system\controlset001\services\advconf2\parameters

Figure 4. Registering the service-related registry key

2.2. UAC Bypass Malware

The Kimsuky group has been using various privilege escalation tools, mainly UACMe. They are still using multiple privilege escalation tools in 2024, but one particular type is more prevalent than the others. The threat actor only utilized the “AppInfo ALPC” technique among the UAC bypass techniques supported by UACMe to create their malware. This technique takes advantage of the fact that if a handle for a debug object of a specific process can be obtained, it can be used to gain a handle that provides full access to the said process. Logs show that this privilege escalation tool was created and executed by PowerShell in the AhnLab Smart Defense (ASD) infrastructure.







Target Type	File Name	File Size	File Path ⓘ
Target	 svchost.exe	97 KB	%ALLUSERSPROFILE%\svchost.exe
Current	 powershell.exe	444 KB	%SystemRoot%\system32\windowpowershell\v1.0\powershell.exe
Parent	 powershell.exe	444 KB	%SystemRoot%\system32\windowpowershell\v1.0\powershell.exe
ParentOfParentOfCurrent	 powershell.exe	444 KB	%SystemRoot%\system32\windowpowershell\v1.0\powershell.exe
GeneratedByParent	 svchost.exe	97 KB	%ALLUSERSPROFILE%\svchost.exe
GeneratedByParent	 File Less Submit t.zip		%ALLUSERSPROFILE%\t.zip

Figure 5. Creating and executing privilege escalation tool using PowerShell

```

lstrcatW(String1, L"taskmgr.exe");
memset(hProcess, 0, 0x18uLL);
memset(&DebugEvent, 0, sizeof(DebugEvent));
if ( fn_AicLaunchAdminProcess(String1, String1, 1, v5, v9, v11, v13, v15, v17, hProcess) )
{
    DbgUiSetThreadDebugObject(v18);
    if ( WaitForDebugEvent(&DebugEvent, 0xFFFFFFFF) )
    {
        while ( 1 )
        {
            if ( DebugEvent.dwDebugEventCode == 3 )
            {
                ExceptionRecord = DebugEvent.u.Exception.ExceptionRecord.ExceptionRecord;
                if ( DebugEvent.u.Exception.ExceptionRecord.ExceptionRecord )
                    break;
            }
            ContinueDebugEvent(DebugEvent.dwProcessId, DebugEvent.dwThreadId, 0x10002u);
            if ( !WaitForDebugEvent(&DebugEvent, 0xFFFFFFFF) )
                goto LABEL_16;
        }
        v19 = 0LL;
        v3 = NtDuplicateObject(

```

Figure 6. Code routine using the AppInfo ALPC technique for privilege escalation

2.3. Modified termsrv.dll

The threat actor used PowerShell to add the modified termsrv.dll file to the infected PC. Upon comparison with the normal termsrv.dll, a specific function was found to be patched. According to the analysis, the function (CDefPolicy::Query) responsible for RDP license authentication was disabled. This means that any user accessing the system is allowed to establish an RDP connection.

7 / 7 Primary Unmatched Functions		
<input type="text"/> <input type="button" value="▼"/> <input type="button" value="✕"/> <input type="button" value="⚙"/>		
Address /	Name	Type /
000000018001F300	sub_18001F300	Normal
000000018000F560	sub_18000F560	Library
000000018002ABF0	sub_18002ABF0	Library
00000001800D1F40	exception::what(void)	Imported
00000001800D1F48	exception::exception(char const * const &,int)	Imported
00000001800D1FD8	exception::~~exception(void)	Imported
00000001800D2028	exception::exception(char const * const &)	Imported

Figure 7. Comparison values of the malicious file before and after patching in BinDiff (sub_18002F300 function mismatch)



Figure 8. Comparison of the normal and patched modules (CDefPolicy::Query function)

To replace a legitimate system DLL in the Windows path with a modified DLL, the threat actor changed the registry key value related to the RDP service.

HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Services\TermService\Parameters

As a result, the RDP service loads the %SystemRoot%\System32\termsrv.dll file by default, so the path must be changed to the modified DLL to load the tampered DLL. In addition, the threat actor used takeown.exe to change the ownership of the termsrv.dll file in the existing system path to Administrators for DLL replacement.

```
takeown /F C:\Windows\System32\termsrv.dll /A
```

Response Guide

1. Double Extension

The Kimsuky group distributes LNK malicious shortcut files disguised as normal documents by attaching a double extension to emails. For example, a file named “pdf.lnk” appears to be a PDF document, but it is actually a Windows shortcut (.lnk) file that can execute malicious scripts or programs. As such, regular users must prevent such suspicious files from being executed by verifying the actual file extension.

[How to Enable File Extensions Display]

Open File Explorer and select the “File name extensions” checkbox in the “View” tab of the top menu, or use Group Policy in Windows settings to force the display.

2. Handling of Modified termsrv.dll File

The hash value must be calculated to check if the legitimate termsrv.dll file has been replaced with the malicious version. To perform this verification, run the following command in Command Prompt (Run as administrator) to calculate the hash value (MD5) of the modified file.

```
certutil -hashfile C:\Windows\System32\termsrv.dll MD5
```

The calculated hash value is compared to “641593eea5f235e27d7cff27d5b7ca2a” and “70d92e2b00ec6702e17e266b7742bbab”. If the values are the same, it means that the file has been tampered with and needs to be replaced with the normal termsrv.dll. Windows provides the sfc program to restore normal programs. Users can restore the patched termsrv.dll to a normal program by entering the following command after executing CMD with administrator privileges:

```
sfc /scannow
```

3. Hidden Administrator Account (“Root”)

If there is a suspicious account named “Root” that was not created by the administrator, the account must be disabled or removed. Run the following command in the command prompt (Run as administrator) to check the account information.

```
net user
```

Search for accounts with suspicious names such as “Root” and suspicious creation time and attributes, excluding standard administrator accounts. If a suspicious account is found, take actions such as removing the hidden attribute and deleting/deactivating the account.

Open the registry editor and navigate to the following path. Then, delete the item corresponding to the account that has been hidden.

```
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows  
NT\CurrentVersion\Winlogon\SpecialAccounts\UserList
```

This prevents the account from being used by threat actors.

```
net user Root /delete
```

Conclusion

While the Kimsuky group uses various types of malware, in the case of PebbleDash, they execute malware based on an LNK file by spear-phishing in the initial access stage to launch their attacks. They then utilize a PowerShell script to create a task scheduler and register it for automatic execution. Through communication with a Dropbox and TCP socket-based C&C server, the group installs multiple malware and tools including PebbleDash. Recently, the group has moved away from their previous method of using the open-source RDP Wrapper. Instead, they have begun directly modifying the system DLL (termsrv.dll) to disable RDP authentication. This demonstrates that the Kimsuky group is continuously evolving their attack techniques to suit their target environments.

This blog post analyzed the latest distribution and execution process of the PebbleDash malware by the Kimsuky group. Considering that the group mainly targets individuals, individual users must be cautious of initial access techniques like spear-phishing and keep their security products up to date to prevent such attacks in advance.

MD5

641593eea5f235e27d7cff27d5b7ca2a

70d92e2b00ec6702e17e266b7742bbab

876dbd9529f00d708a42f470a21a6f79

a5cca2b56124e8e9e0371b6f6293e729

a8976e7dc409525a77b0eef0d0c3c4f2

IP

159[.]100[.]13[.]216

213[.]145[.]86[.]223

216[.]219[.]87[.]41

64[.]20[.]59[.]148

Gain access to related IOCs and detailed analysis by subscribing to **AhnLab TIP**. For subscription details, click the banner below.

