DanaBot银行木马更新，被配置为MaaS

angel010 / 2018-10-03 16:39:50 / 浏览数 3721 技术文章 技术文章 顶(0) 踩(0)

本文翻译自：
https://www.proofpoint.com/us/threat-insight/post/danabot-gains-popularity-and-targets-us-organizations-large-campaigns

---

Proofpoint研究人员在2018年5月首次发现了DanaBot，分析结果显示与攻击澳大利亚企业的威胁单元有关。本文对DanaBot进行逆向分析并分析了其攻击美国企业的活动。

DanaBot近期活动

近期，ESET研究人员发文描述了DanaBot的最新攻击活动，受影响的国家包括波兰、意大利、大国和澳大利亚。9月底，研究人员发现攻击者将攻击美国使用的Panda银行木

Hancitor攻击活动

9月26日，Proofpoint研究人员发现一起针对美国接收者的上万的垃圾邮件攻击活动。邮件使用eFax诱饵文件（图1）和URL链接来下载含有恶意宏的文档（图2）。如果用户

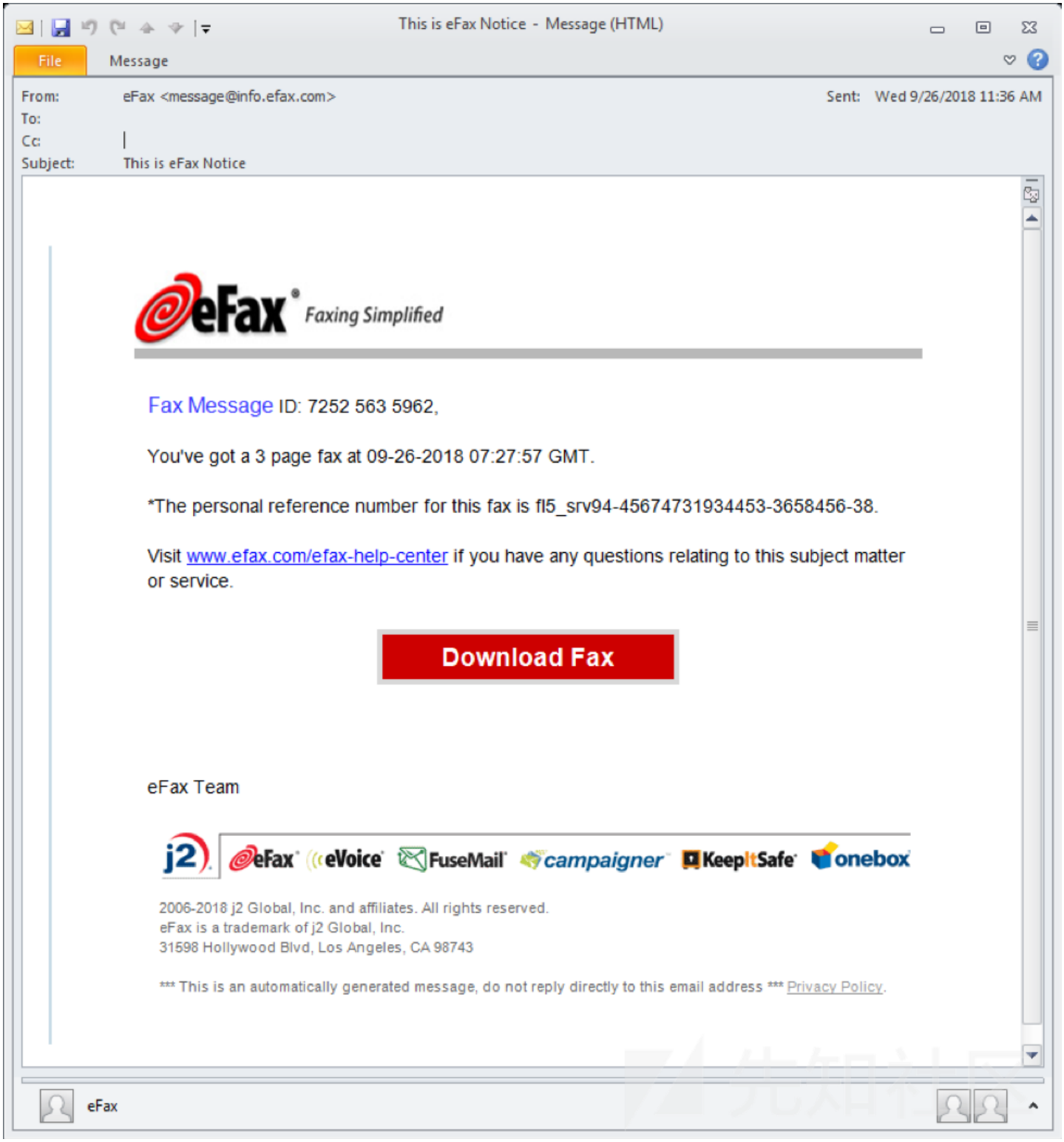

图1: 含有下载恶意payload的宏的URL消息示例

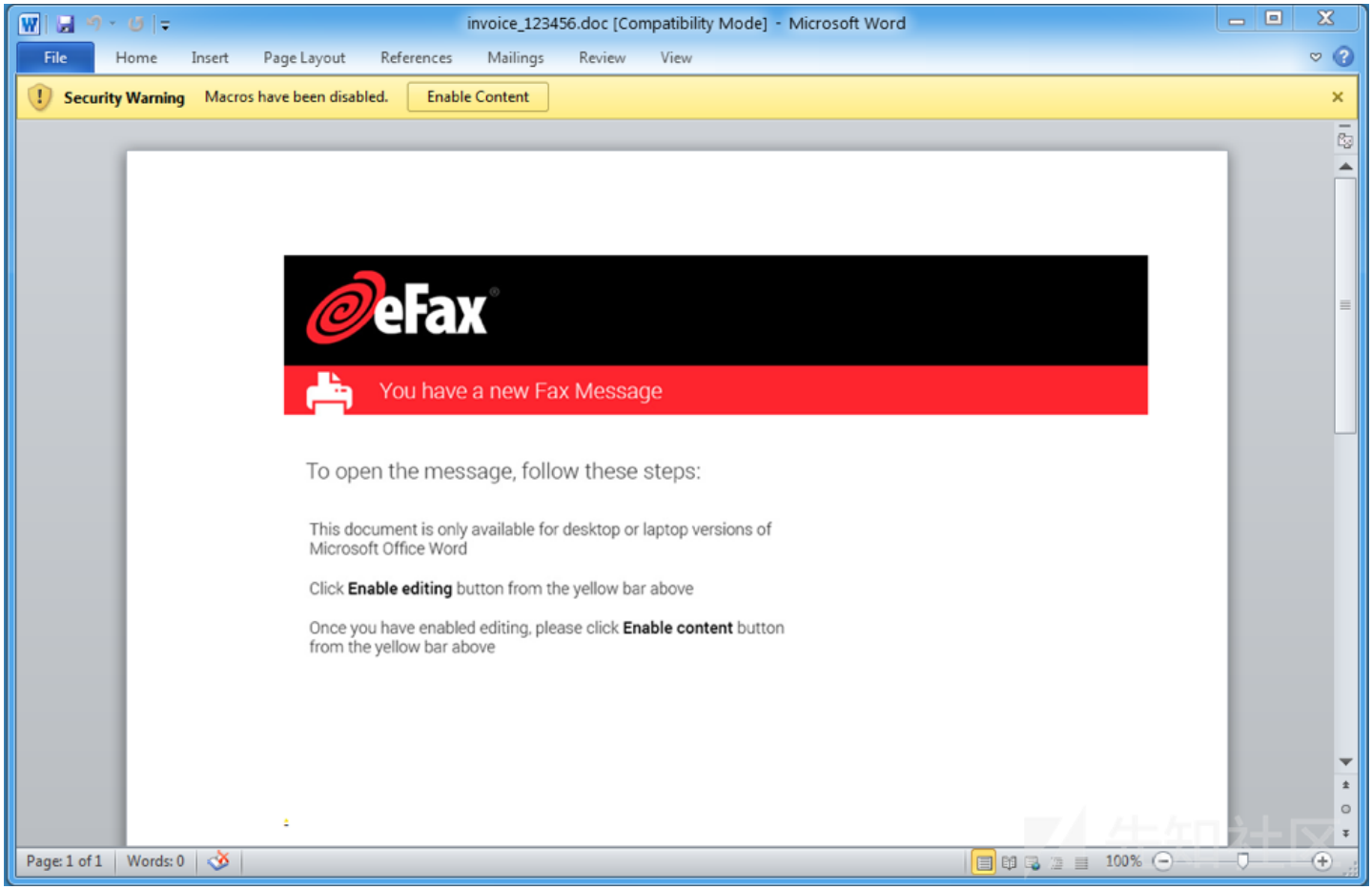

图2:含有Hancitor payload的宏文档

恶意软件分析 (v2.003)

DanaBot是一款用Delphi语言开发的银行恶意软件。本节在原有v2.003的基础上进行分析。这是目前最新的版本，最早是9月初出现的。该版本号是基于version字符串（图

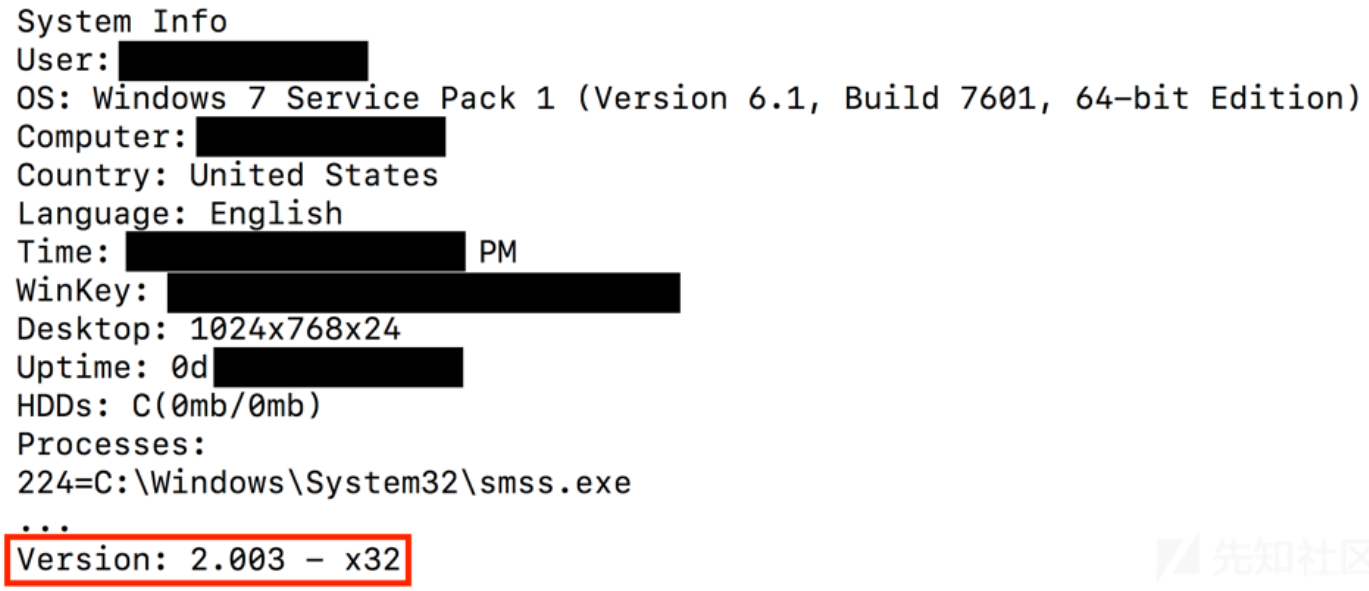

图3: 和系统信息一起发回给C2服务器的DanaBot版本字符串

DanaBot是由以下三个组件组成的：

1. Loader:负责下载和加载主组件。
2. Main component: 负责下载、配置和加载模块
3. Modules: 负责完成不同的恶意软件功能

反分析技术

DanaBot明显含有大量的垃圾代码，代码中含有额外的指令、条件描述和循环等。加之是用Delphi语言编写的，所以非常影响对DanaBot的逆向。另外，DanaBot使用Win
API函数哈希和加密字符串来防止分析师和自动化分析工具确定代码的作用。

加密的字符串的字符保存在DWORDs数组中，需要用key和基本的替换密文来解密。Github中有 IDA Pro Python脚本和加载器中使用的解密字符串和主模块。

Command & Control IPs

加载器和主模块C2 IP地址都以DWORDs的形式保存。图4是加载器组件的内存示例：

```
.data:004D2520 g_c2_45_77_231_138 dd 8AE74D2Dh          ; DATA XREF: XXX_path_to_comms:loc_4C7FDB↑r
.data:004D2520                                          ; Python>socket.inet_ntoa(struct.pack("I", 0x8AE74D2D))
.data:004D2520                                          ; 45.77.231.138
.data:004D2524 g_c2_149_154_152_64 dd 40989A95h         ; DATA XREF: XXX_path_to_comms+DF7↑r
.data:004D2528 g_c2_91_210_222_49 dd 31DED25Bh          ; DATA XREF: XXX_path_to_comms+EA5↑r
.data:004D252C g_c2_81_39_236_104 dd 68EC2751h          ; DATA XREF: XXX_path_to_comms+ECC↑r
.data:004D2530 g_c2_133_117_64_199 dd 0C7407585h        ; DATA XREF: XXX_path_to_comms+FFF↑r
.data:004D2534 g_c2_87_229_30_154 dd 9A1EE557h          ; DATA XREF: XXX_path_to_comms+1023↑r
.data:004D2538 g_c2_6_43_184_18 dd 12B82B06h            ; DATA XREF: XXX_path_to_comms:loc_4C82A9↑r
.data:004D253C g_c2_107_202_186_201 dd 0C9BACA6Bh       ; DATA XREF: XXX_path_to_comms:loc_4C8300↑r
.data:004D2540 g_c2_107_84_178_1 dd 1B2546Bh            ; DATA XREF: XXX_path_to_comms+11E8↑r
.data:004D2544 g_c2_216_45_35_66 dd 42232DD8h           ; DATA XREF: XXX_path_to_comms:loc_4C7F48↑r
.data:004D2548 g_rsa_key        db 6                    ; DATA XREF: decode_rsa_key_and_get_f39_data+40↑w
.data:004D2548                                          ; main_component_phonehome+C37↑o
.data:004D2549                  db    2
.data:004D254A                  db    0
.data:004D254B                  db    0
.data:004D254C                  db    0
.data:004D254D                  db 0A4h ; ¤
.data:004D254E                  db    0
.data:004D254F                  db    0
.data:004D2550                  db  52h ; R
.data:004D2551                  db  53h ; S
.data:004D2552                  db  41h ; A
.data:004D2553                  db  31h ; 1
.data:004D2554                  db    0
```

图4: DanaBot加载器模块所在内存中的C2 IP地址示例

C2通信

之前的分析显示DanaBot的加载器模块使用HTTP进行通信，其主模块使用的是二进制协议。在v2.003版本中，所有组件都使用TCP
443端口的二进制协议。虽然使用的是443端口，但并没有使用TLS。

该协议在183字节的header之后是可选的payload数据。请求中的大多数的header值都回在响应header中返回。Payload数据的格式与特定的命令有关。

二进制协议header

Header示例如图5：

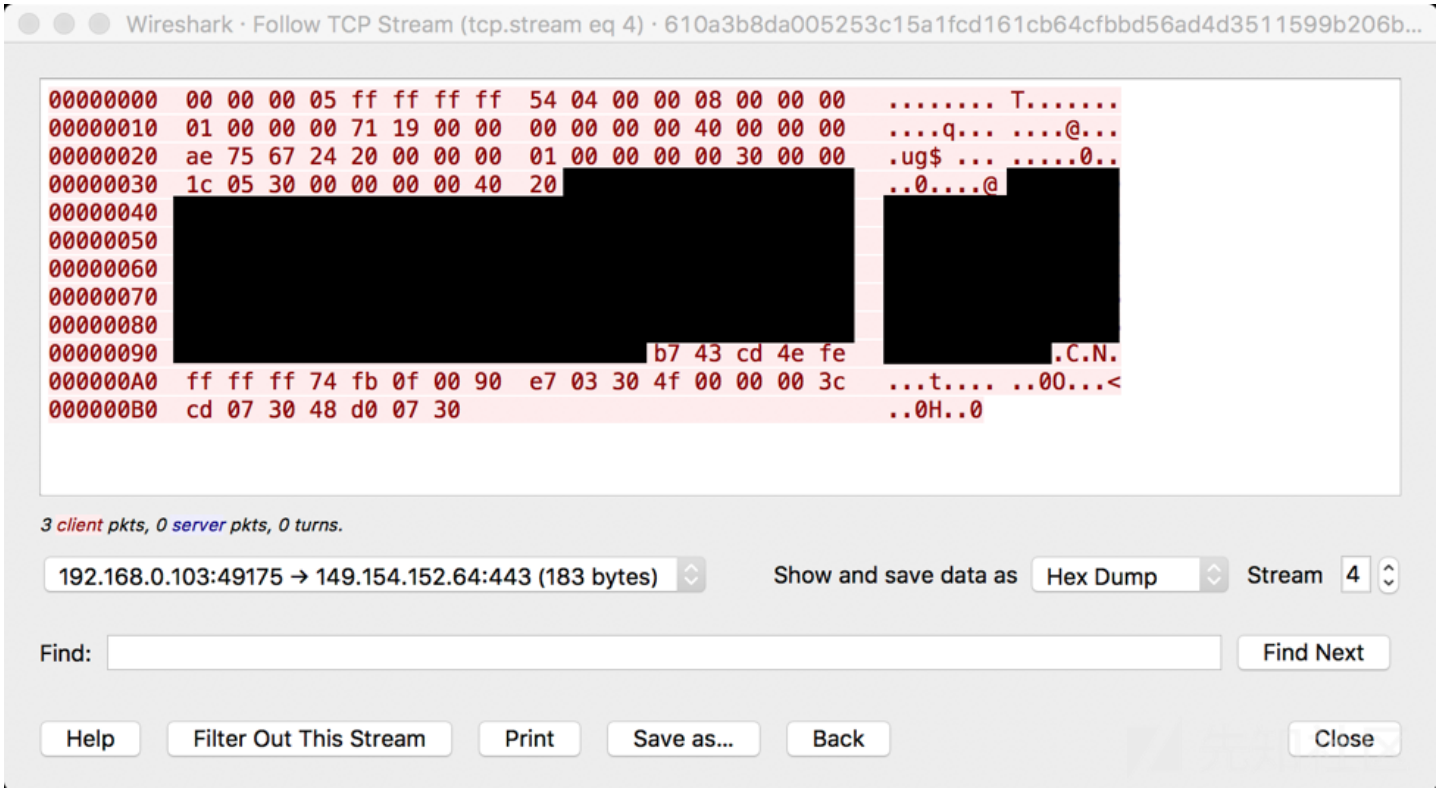

图5: DanaBot二进制协议中使用的183字节header

可以分成下面的域：

- Offset 0:■■■■■■stack■
- Offset 4: hardcoded -1 (DWORD)
- Offset 8: ■■(DWORD)
- Offset 0xc: affiliate ID (DWORD)
- Offset 0x10: hardcoded 1 (DWORD)
- Offset 0x14: random value based on a linear congruential generator (DWORD)
- Offset 0x18: unknown counter variable (DWORD)
- Offset 0x1c: system architecture (DWORD)
- Offset 0x20: Windows version information (DWORD)
- Offset 0x24: command argument (DWORD)
- Offset 0x28: admin status (DWORD)
- Offset 0x2c: process integrity level (DWORD)
- Offset 0x30: payload length (QWORD)

- Offset 0x38: length of next field (BYTE)
- Offset 0x39: bot ID (32 bytes)

- Offset 0x59: length of next field (BYTE)
- Offset 0x5a: command-dependent (32 bytes)

- Offset 0x7a: length of next field (BYTE)
- Offset 0x7b: a nonce (32 bytes)
- Offset 0x9b - end of header: random values (stack junk)

命令

研究人员共发现了以下命令，其中第一个命令是由loader执行的，其余的命令由主模块执行。

Command 0x454 (1108): “Request main component”

该命令由加载器执行，用于从C2服务器请求主模块。命令参数（header中的offset 0x24）含有整数32或64来请求x86或x64版本的组件。响应payload中含有加密的数据和加密的128字节RAS签名区块用于验证数据。解密的key由CryptDeriveKey Windows API函数生成。数据是用`\x00`字节的初始化向量（IV）进行AES-256-CBC加密的。解密是数据就是`rundll32.exe`会执行的主模块DLL。

Command 0x453 (1107): “Initial beacon”

这是主组件发送给C2服务器的第一条命令，请求和响应中都没有数据，所以研究人员认为这是一个信标。

Command 0x44c (1100): “Request module identifiers”

恶意软件用这条命令从C2服务器请求一个模块id列表。图6是含有6个模块id的响应示例：

```
759CBB3E1B883BDCA23E9052462F641E
E0FBBC92DB9927BFC474A64DF4F9C22F
D0C851FBCA030928B535FAF3188DAFBA
A5BBBAB3A17BA2119F47F0E4316EE5BF
4F06D71C93E4105307339704D21C49A3
8C59B6C9985F983E248E27CC0BF98A2D
```

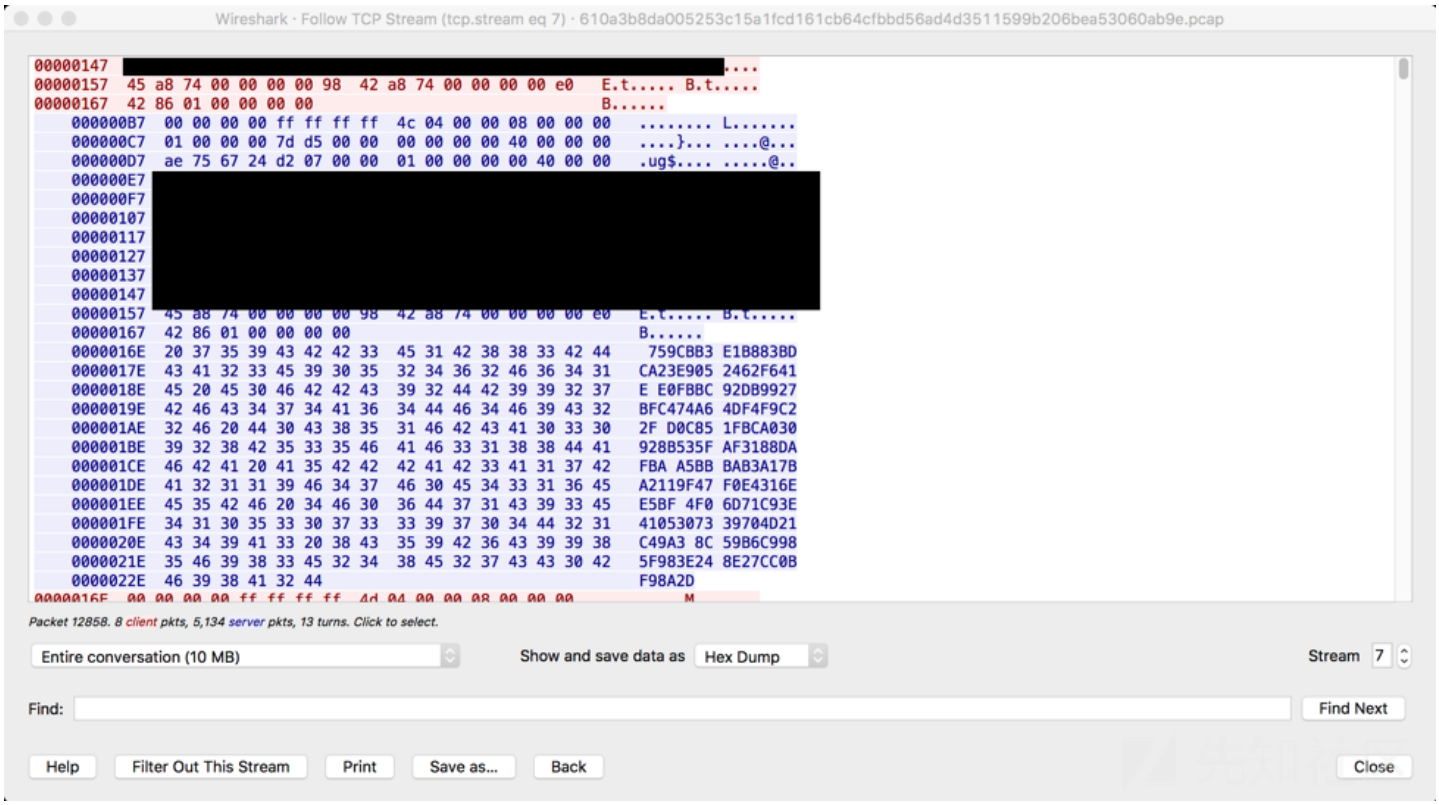

图6: 含有模块id列表的Command 0x44c response payload数据

Command 0x44d (1101): “Request module”

这条命令用于从C2服务器请求模块。Header的offset0x5a域含有一个模块id，用于确定应该下载哪个模块。响应payload数据中含有一个1699字节的subheader、加密数据

```
00000000  93 f7 36 00 00 00 00 00  ff ff ff ff 01 00 00 00  |..6.............|
00000010  46 00 46 00 31 00 00 00  00 00 00 00 00 00 00 00  |F.F.1...........|
00000020  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
*
00000210  00 00 00 00 00 00 00 00  46 00 46 00 31 00 2e 00  |........F.F.1...|
00000220  64 00 61 00 74 00 00 00  00 00 00 00 00 00 00 00  |d.a.t...........|
00000230  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
*
00000640  00 00 00 00 00 00 00 00  00 20 37 35 39 43 42 42  |......... 759CBB|
00000650  33 45 31 42 38 38 33 42  44 43 41 32 33 45 39 30  |3E1B883BDCA23E90|
00000660  35 32 34 36 32 46 36 34  31 45 02 2d 2d 00 00 00  |52462F641E.--...|
00000670  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
00000680  00 00 20 00 00 00 05 00  00 00 01 00 00 00 01 00  |.. .............|
00000690  00 00 f0 f0 36 00 00 00  00 00 01 7d 97 a1 f3 f6  |....6......}....|
000006a0  29 e5 40                                          |).@|
000006a3
```

图7: 含有1699字节subheader的Command 0x44d response payload数据

下面这些域都在subheader中出现过：

- Offset 0: total length of subheader and data (QWORD)
- Offset 8: hardcoded -1 (DWORD)
- Offset 0x10: module name (520-byte wide string)

- Offset 0x218: module filename (520-byte wide string)
- Offset 0x649: length of next field (BYTE)
- Offset 0x64a: module identifier (32 bytes)
- Offset 0x682: module architecture (DWORD)
- Offset 0x686: module type (DWORD)
- Offset 0x68e: data is ZLIB-compressed flag (DWORD)
- Offset 0x692: length of encrypted data (QWORD)

用于解密模块的解密key是由CryptDeriveKey Windows API函数生成的，该函数由以下过程初始化：

1. 复制1699字节的subheader到缓存中，并将以下域清0：
   a. Offset 0: subheader和数据的长度和(QWORD)
   b. Offset 0x692: 解密数据的长度 (QWORD)
2. 对缓存进行MD5哈希
3. 哈希的大写十六进制摘要就是MD5哈希本身

数据是用IV和16个空\x00字节用AES-256-CBC加密的。解密的字符串可选用ZLIB压缩，一旦解压含有的模块dll就会被rundll32.exe执行。

表1: 模块列表

Module identifier	Name	Old name	Functionality
759CBB3E1B883BDCA23E9052462F641E	FF1	Sniffer	Proxy
E0FBBC92DB9927BFC474A64DF4F9C22F	FF2	Stealer	Stealer module
D0C851FBCA030928B535FAF3188DAFBA	FF3	NA	64-bit version of Stealer module (new)
8C59B6C9985F983E248E27CC0BF98A2D	FF4	NA	RDP module (new)
A5BBBAB3A17BA2119F47F0E4316EE5BF	FF5	TOR	TOR proxy
4F06D71C93E4105307339704D21C49A3	FF6	VNC	VNC

Command 0x44f (1103): “Get configuration files”

恶意软件使用该命令来从C2服务器请求配置文件。恶意软件在接收到183字节的响应header后，恶意软件会在C2服务器响应response payload数据前发送\xff\xff\xff\xff\xff\xff\xff\xff。Payload数据的格式和加密与模块类似，但会发送不同的配置文件。

表2: 配置文件

Config filename	Variants	Purpose	Comments
BitVideo	VVie	Processes to watch	For screenshots/video recording perhaps
KeyBit	BitKey, VKey	Processes to watch	For keylogging possibly
BitFiles	Vfiles, VBit	Cryptocurrency wallet files to steal	
PosWtFilter	PostWFilter, VFilter	List of websites for which to steal requests	PosWtFilter may be a typo (in affiliate IDs 3 and 9)
webinj33	uabanks	Proxying config	Incrementing versions
inj25	InjectZZ, InjectSW	Webinjects	Incrementing versions; Zeus-style injects

恶意软件用该命令来发送来发送系统信息或截屏等数据到C2。请求payload数据中含有656字节的subheader、加密数据和加密session key（图8是subheader示例）。

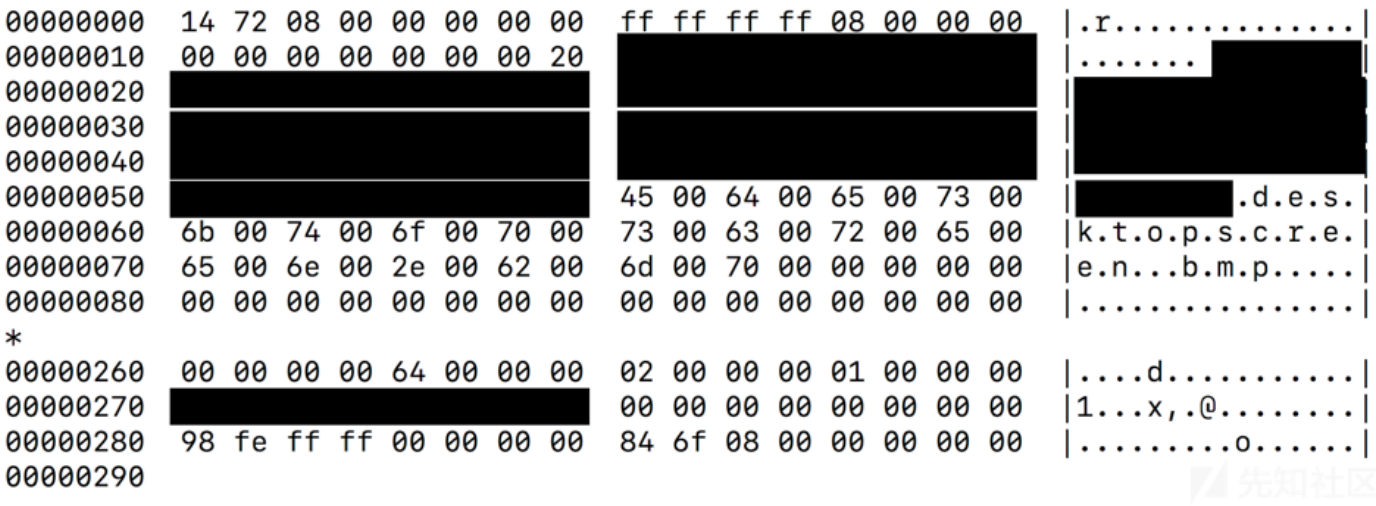

图8:含有656字节subheader的 Command 0x44e request payload数据

Subheader中含有下面的域：

- Offset 0: total length (QWORD)
- Offset 8: hardcoded -1 (DWORD)
- Offset 0xc: affiliate ID (DWORD)
- Offset 0x17: length of next field (BYTE)
- Offset 0x18: bot ID (32 bytes)
- Offset 0x38: length of next field (BYTE)
- Offset 0x39: MD5 hex digest of plaintext data (32 bytes)
- Offset 0x5a: filename (520-byte wide string)
- Offset 0x264: data type (DWORD)
- Offset 0x270: system time (unknown format) (QWORD)
- Offset 0x280: timezone bias (DWORD)
- Offset 0x288: encrypted data length (QWORD)

数据可以用IV和16\x00字节进行ZLIB压缩和AES-256-CBC加密。加密key是用CryptDeriveKey Windows函数生成的，RSA加密使用的是嵌入的RSA公钥。然后把RSA加密的AES加到加密数据的尾部。

表3: 发送的文件

Filename	Comments
desktopscreen.bmp	Screenshot
Cookies.txt	Stored web browser cookies
“System Info"	Various system information

C&C基础设施

DanaBot用加载器来从C2服务器下载主组件。主组件含有10个硬编码的C2 IP地址列表，该IP地址被用作恶意软件通信。研究人员发现硬编码的C2列表每个消失会变化一次，而此时主模块会下载下来。研究人员发现每个样本的C2列表是不同的。研

- 158.255.215[.]31 (in 7 lists)
- 149.154.152[.]64 (in 7 lists)
- 37.235.53[.]232 (in 6 lists)
- 95.179.151[.]252 (in 5 lists)
- 178.209.51[.]227 (in 5 lists)
- 149.154.157[.]220 (in 5 lists)
- 45.77.54[.]180 (in 4 lists)
- 45.77.96[.]198 (in 3 lists)
- 45.77.51[.]69 (in 3 lists)
- 45.77.231[.]138 (in 3 lists)

整个C2的IP列表中，只有下面10个好像是有响应的：

- 149.154.152[.]64
- 149.154.157[.]220
- 158.255.215[.]31
- 178.209.51[.]227
- 37.235.53[.]232
- 45.77.231[.]138
- 45.77.51[.]69
- 45.77.54[.]180
- 45.77.96[.]198
- 95.179.151[.]252

研究人员还发现这些重叠、交叉的IP列表中还含有一些不能路由的IP：

- 10.181.255[.]78
- 225.100.146[.]224
- 225.21.55[.]173
- 226.181.243[.]104
- 228.226.171[.]37
- 234.106.187[.]114
- 234.63.249[.]87
- 234.97.12[.]178
- 235.40.105[.]171
- 238.87.111[.]55

因此，研究人员推测，主组件可能只含有几个真实的C2地址，其他的都是随机的诱饵地址。

Affiliate System

根据传播方法和攻击目标，研究人员将DanaBot活动用affiliate ID进行分组：

Affiliate ID	Targeting	Distribution
3	Poland, Austria, Germany, Italy	Zipped-VBS attachments in email campaigns
4	Australia	Links in email campaigns
5	No webinjects	unknown
8	UK, Ukraine, and Canada	Various email campaigns
9	Same as affiliate ID 3	Fallout Exploit Kit
11	US, No webinjects	Hancitor downloader malware from links in email campaigns
12	Australia	unknown
13	Germany	unknown
20	No webinjects	unknown

不同affiliate ID的DanaBot样本也会使用相同的C2 IP地址。因此，研究人员推测DanaBot可能被设置为恶意软件即服务（MAAS）系统。

对比CryptXXX勒索软件

Proofpoint研究人员在2016年分析过CryptXXX文件加密勒索软件，该勒索软件与Reveton
“police”勒索软件有一些相似处。而且也是用Delphi语言编写，使用基于TCP 443端口的定制C2协议。

DanaBot的C2流量看似是该协议的进化版，使用AES加密和ZLIB压缩。CryptXXX checkin的格式是：

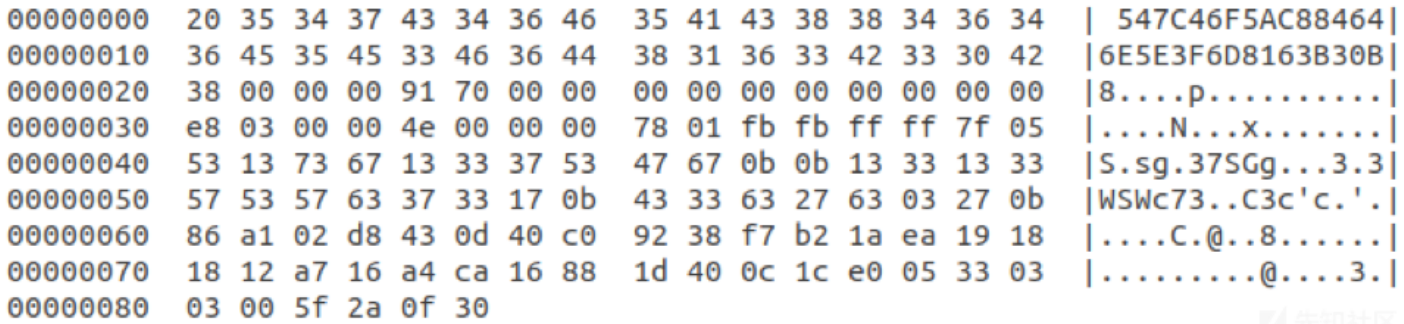

图9: CryptXXX checkin格式

CryptXXX和DanaBot都有下面的域：

- Offset 0: length of next field (BYTE)
- Offset 2: bot ID (32 bytes)
- Offset 0x34 : length of compressed buffer
- Offset 0x38: Zlib-compressed buffer (0x4e bytes)

压缩的缓存解码后：

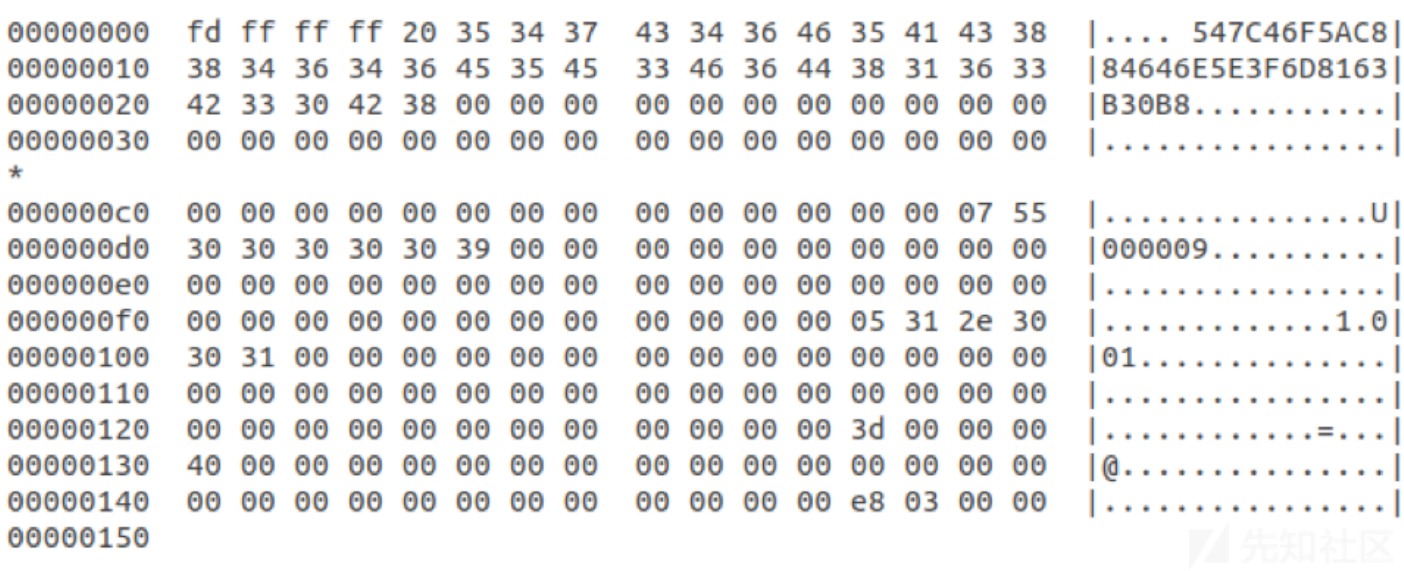

图10: 解码的payload buffer

解码的缓存中有以下域：

- Offset 4: length of next field (BYTE)
- Offset 5: bot ID (32 bytes)
- Offset 0xce : length of next field (BYTE)
- Offset 0xcf : Affiliate ID (7 bytes)
- Offset 0xfc : length of next field (BYTE)
- Offset 0xfd : Version string (5 bytes)

之后的通信中会有一个解码的请求来下载Stealer模块——stiller.dll：

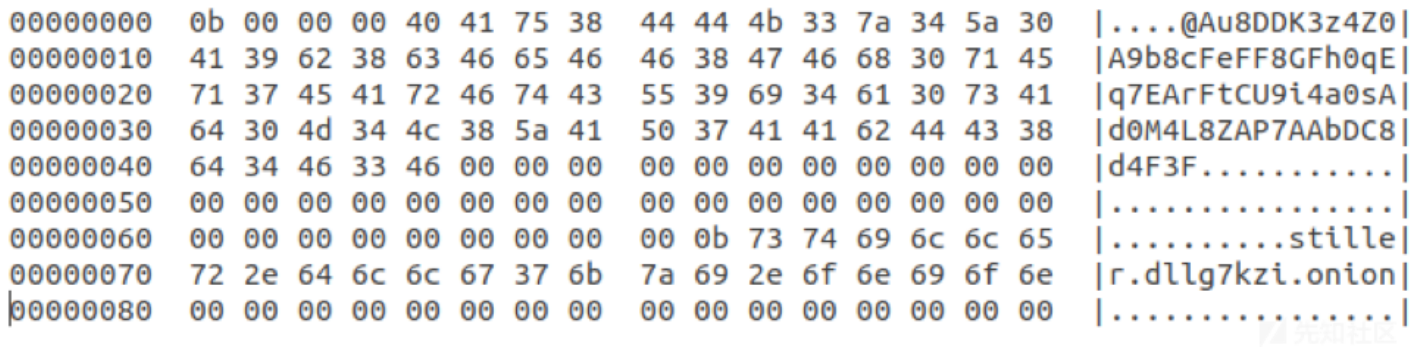

图11:解码的下载Stealer模块的请求

这样看来Danabot可能是同一个组织的不同恶意软件。该恶意软件家族最早的产品是恶意软件，随后在Reveton加入窃取器功能，进一步进化为CryptXXX勒索软件，现在在

结论

DanaBot恶意软件的传播已经超越澳大利亚国界，目前攻击波兰、意大利、德国、奥德利、美国等。DanaBot是一款银行木马，也就是说其攻击目标一定程度上是有地理范

参考文献

[1] https://www.proofpoint.com/us/threat-insight/post/danabot-new-banking-trojan-surfaces-down-under-0

[2] https://www.welivesecurity.com/2018/09/21/danabot-targeting-europe-adds-new-features/

[3] https://www.proofpoint.com/us/threat-insight/post/hancitor-ruckguv-reappear

[4] https://0ffset.wordpress.com/2018/08/12/post-0x16-hancitor-stage-1/

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

[6] http://malware-traffic-analysis.net/2016/04/20/index.html

[7] https://github.com/EmergingThreats/threatresearch/blob/master/danabot/func_hashes.py

[8] https://github.com/EmergingThreats/threatresearch/blob/master/danabot/loader_func_hashes.txt

[9] https://github.com/EmergingThreats/threatresearch/blob/master/danabot/main_func_hashes.txt

[10] https://github.com/EmergingThreats/threatresearch/blob/master/danabot/decrypt_str_ida.py

[11] https://github.com/EmergingThreats/threatresearch/blob/master/danabot/loader_strings.txt

[12] https://github.com/EmergingThreats/threatresearch/blob/master/danabot/main_strings.txt

[13] https://github.com/EmergingThreats/threatresearch/blob/master/danabot/24_hours_of_ips.txt

上一篇：Java沙箱逃逸走过的二十个春秋（三） 下一篇：Java沙箱逃逸走过的二十个春秋（四）

1. 0 条回复
   - 动动手指，沙发就是你的了！