登录

CVE-2019-0604：微软SHAREPOINT远程执行的脆弱性分析

Pinging / 2019-03-22 09:15:00 / 浏览数 3351 安全技术 漏洞分析 顶(0) 踩(0)

上个月，Microsoft发布了补丁来解决SharePoint中的两个远程代码执行（RCE）漏洞。使用这两个被评级为严重的漏洞，攻击者可以发送任意命令请求并在SharePoint应
Markus Wulftange向ZDI计划报告了这两个漏洞。 他提供了有关CVE-2019-0604细节。

在搜索新漏洞时，一种有效方法是自下而上进行搜索。文章描述到，在分析寻找控制流数据流的过程中我们可以使用一种比较新颖的方法以便能够查明数据能否接触到sink

其中一个常用的sink是使用XmlSerializer反序列化方法。
通常，它被认为是一个安全的序列化程序，因为它必须使用指定的类型数据进行检测，并且不能出现在期望类型的对象图中。但是，如果预期的类型数据也可以被控制，那么
Mirosh在其第13次JSON攻击中记录下来PDF。

为了分析SharePoint 2016程序集，dnSpy是一个很好的工具，因为它可以用于.NET应用程序的反编译和调试。因此，在将dnSpy附加到运行SharePoint
2016的IIS工作进程w3wp.exe并且加载程序集之后，我们可以分析XmlSerializer■Type■构造函数的用法。现在，更为复杂的分析部分是我们必须看每个XmlSeriali

调用XmlSerializer■Type■构造函数的方法之一是Microsoft.SharePoint.dll中的Microsoft.SharePoint.BusinessData.Infrastructure.EntityInst

EntityInstanceIdEncoder

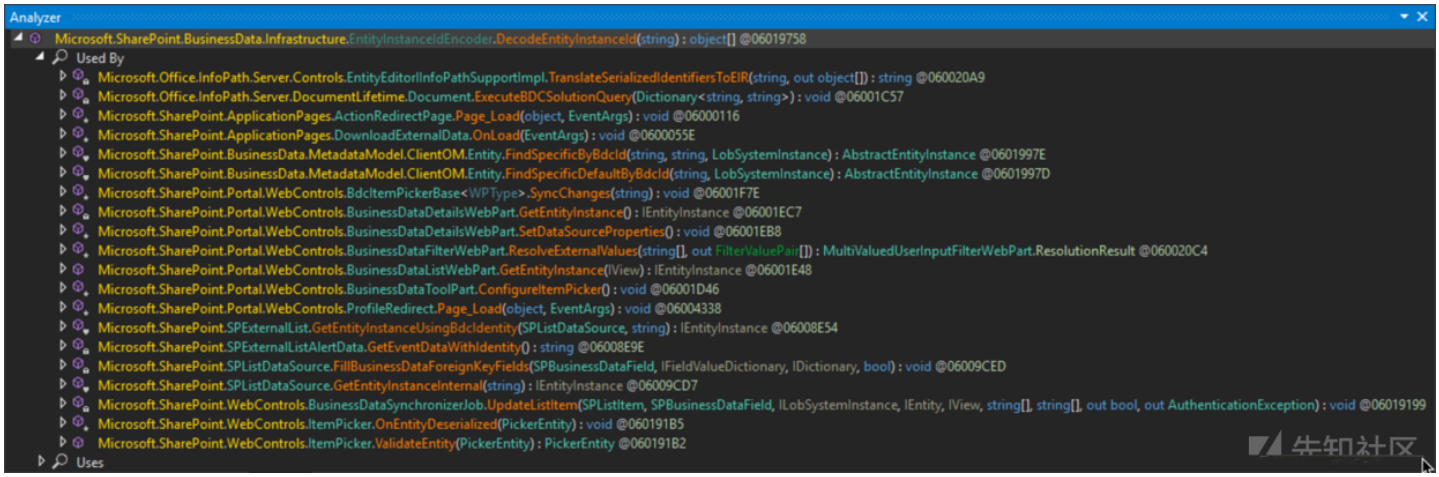
```
219         public static object[] DecodeEntityInstanceId(string encodedId)
220         {
221             if (string.IsNullOrEmpty(encodedId))
222             {
223                 throw new ArgumentException(Microsoft.BusinessData.Resources.ResourceManager.GetString("ApplicationRegistry_EntityInstanceIdEncoder_IdNull"), "encodedId");
224             }
225             if (encodedId.StartsWith("__", StringComparison.Ordinal))
226             {
227                 int num = "__".Length;
228                 object[] result;
229                 try
230                 {
231                     char c = encodedId[num++];
232                     object[] array = new object[(int)(c - 'a')];
233                     for (int i = 0; i < array.Length; i++)
234                     {
235                         char c2 = encodedId[num++] - 'a';
236                         Type type = EntityInstanceIdEncoder.types[(int)c2];
237                         char c3 = EntityInstanceIdEncoder.HexDecode(encodedId, num);
238                         num += 4;
239                         string text = (!type.Equals(typeof(Guid))) ? EntityInstanceIdEncoder.HexDecode(encodedId, num, (int)c3).ToString() : encodedId.Substring(num, (int)c3);
240                         num += (int)c3;
241                         if (type.Equals(typeof(string)))
242                         {
243                             array[i] = text;
244                         }
245                         else if (type.Equals(typeof(DateTime)))
246                         {
247                             array[i] = new DateTime(long.Parse(text.Substring(1), NumberFormatInfo.InvariantInfo), (DateTimeKind)(text[0] - 'a'));
248                         }
249                         else if (type.Equals(typeof(Guid)))
250                         {
251                             array[i] = new Guid(text);
252                         }
253                         else if (type.Equals(typeof(object)))
254                         {
255                             if (text.Equals("null", StringComparison.OrdinalIgnoreCase))
256                             {
257                                 array[i] = null;
258                             }
259                             else
260                             {
261                                 int num2 = text.IndexOf(':');
262                                 string typeName = text.Substring(0, num2);
263                                 string s = text.Substring(num2 + 1, text.Length - num2 - 1);
264                                 XmlSerializer xmlSerializer = new XmlSerializer(Type.GetType(typeName, true));
265                                 TextReader textReader = new StringReader(s);
266                                 array[i] = xmlSerializer.Deserialize(textReader);
267                                 textReader.Close();
268                             }
269                         }
270                         else
271                         {
272                             array[i] = EntityInstanceIdEncoder.parseMethods[(int)c2].Invoke(null, new object[]
273                             {
274                                 text
275                             });
276                         }
277                     }
278                     result = array;
```

这里，用于指定期望类型的typeName和反序列化的数据都源自文本函数，该文本源自方法的参数encodedId。

只要调用该方法并对传递的参数进行控制，那么就能够完美运行我们的程序。

追溯数据流源头

下一步是通过调用函数并查看它们是否来自可以从外部进行启动以及是否也可以提供参数值。



如果用户熟悉ASP.NET，则某些方法并不陌生，如Page_Load■object■EventArgs■或OnLoad■EventArgs■。
它们在ASP.NET生命周期中被调用，它们在扩展System.Web.UI.Page中定义的类型表示.aspx文件的基类型。 事实上，这三种类型都有相应的.aspx文件：

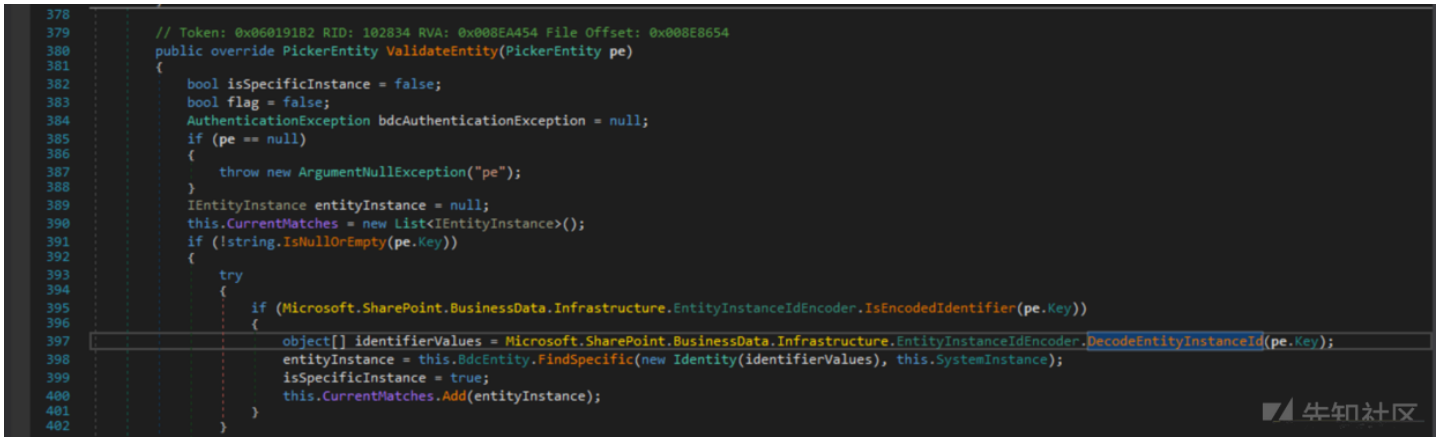
· Microsoft.SharePoint.ApplicationPages.ActionRedirectPage:
/_layouts/15/ActionRedirect.aspx

· Microsoft.SharePoint.ApplicationPages.DownloadExternalData:
/_layouts/15/downloadexternaldata.aspx

· Microsoft.SharePoint.Portal.WebControls.ProfileRedirect:
/_layouts/15/TenantProfileAdmin/profileredirect.aspx

虽然在这三种情况下，参数值都来自HTTP请求，但它们同样是URL的查询字符串。
这可能会出现一个问题，因为十六进制编码会将长度乘以4，从而可以变得非常长并超过HTTP请求行的限制。

经过进一步分析，最后一个，ItemPicker.ValidateEntity■PickerEntity■方法证明是一个更好的选择。



```
// Token: 0x06019182 RID: 102834 RVA: 0x008EA454 File Offset: 0x008E8654
public override PickerEntity ValidateEntity(PickerEntity pe)
{
    bool isSpecificInstance = false;
    bool flag = false;
    AuthenticationException bdcAuthenticationException = null;
    if (pe == null)
    {
        throw new ArgumentNullException("pe");
    }
    IEntityInstance entityInstance = null;
    this.CurrentMatches = new List<IEntityInstance>();
    if (!string.IsNullOrEmpty(pe.Key))
    {
        try
        {
            if (Microsoft.SharePoint.BusinessData.Infrastructure.EntityInstanceIdEncoder.IsEncodedIdentifier(pe.Key))
            {
                object[] identifierValues = Microsoft.SharePoint.BusinessData.Infrastructure.EntityInstanceIdEncoder.DecodeEntityInstanceId(pe.Key);
                entityInstance = this.BdcEntity.FindSpecific(new Identity(identifierValues), this.SystemInstance);
                isSpecificInstance = true;
                this.CurrentMatches.Add(entityInstance);
            }
        }
```

这里，传递的PickerEntity的PickerEntity的Key属性用于EntityInstanceIdEncoder.DecodeEntityInstanceId■string■调用。
它由EntityEditor.Validate()调用，它迭代存储在EntityEditor.Entities属性中的每个条目中以用来进行验证。

```
588         [SharePointPermission(SecurityAction.Demand, ObjectModel = true)]
589         public virtual void Validate()
590         {
591             if (this.m_Validated)
592             {
593                 return;
594             }
595             this.IsValid = true;
596             this.m_entityHash = this.m_entityHashTemp;
597             this.m_listOrder = this.m_listOrderTemp;
598             this.m_Validated = true;
599             if (!this.Visible)
600             {
601                 return;
602             }
603             if (this.MaximumEntities > 0 && this.m_listOrder.Count > this.MaximumEntities)
604             {
605                 this.IsValid = false;
606                 this.ErrorMessage = SPResource.GetString("PickerSelectTooMany", new object[]
607                 {
608                     this.MaximumEntities
609                 });
610                 return;
611             }
612             Hashtable hashtable = new Hashtable();
613             ArrayList arrayList = new ArrayList();
614             StringBuilder stringBuilder = new StringBuilder();
615             foreach (object obj in this.Entities)
616             {
617                 PickerEntity pickerEntity = (PickerEntity)obj;
618                 PickerEntity pickerEntity2 = pickerEntity;
619                 try
620                 {
621                     using (new SPMonitoredScope("Validate a person or entity", 250u, new ISPScopedPerformanceMonitor[0]))
622                     {
623                         pickerEntity2 = this.ValidateEntity(pickerEntity);
```


该方法由EntityEditor.LoadPostData■string■NameValueCollection■调用，该实现System.Web.UI.IPostBackDataHandler.LoadPostData■string■

```
702         [SharePointPermission(SecurityAction.Demand, ObjectModel = true)]
703         public virtual bool LoadPostData(string postDataKey, NameValueCollection values)
704         {
705             this.EnsureChildControls();
706             string text = EntityEditor.StrEatUpNbsp(this.HiddenSpanData.Value);
707             this.IsChanged = this.ParseSpanData(text);
708             if (!string.IsNullOrEmpty(text))
709             {
710                 foreach (object obj in this.m_listOrderTemp)
711                 {
712                     PickerEntity pickerEntity = (PickerEntity)obj;
713                     if (pickerEntity.IsResolved)
714                     {
715                         this.m_listRevalidation.Add(pickerEntity);
716                     }
717                     pickerEntity.IsResolved = false;
718                 }
719                 this.Validate();
```


因此，对PostPicker Web控件的消息请求会自动调用该方法。 调用图如下所示：

```
Microsoft.SharePoint.BusinessData.Infrastructure.EntityInstanceIdEncoder.DecodeE
ntityInstanceId(string) : object[] @06019753
     Used By

     Microsoft.SharePoint.WebControls.ItemPicker.ValidateEntity(PickerEntity) :
PickerEntity @060191AD
                    Used By

     Microsoft.SharePoint.WebControls.EntityEditor.Validate() : void @06006457
                                        Used By

     Microsoft.SharePoint.WebControls.EntityEditor.LoadPostData(string,
NameValueCollection) : bool @0600645B
```

并且我们要注意类型层次结构：

```
Microsoft.SharePoint.WebControls.EntityEditor
     Extended By
             Microsoft.SharePoint.WebControls.EntityEditorWithPicker
                     Extended By
                             Microsoft.SharePoint.WebControls.ItemPicker
```

数据流验证

现在有一种方法可以从ItemPicker
Web控件发送到达EntityInstanceIdEncoder.DecodeEntityInstanceId■string■，然而此时我们仍然不清楚是否可以控制PickerEntity的Key属性。

EntityEditor.Entities属性由私有字段m_listOrder决定，该字段仅在两个点处分配：在实例化期间和在EntityEditor.Validate()方法中。在后一种情况下，它

之后我们需要查看EntityEditor.ParseSpanData■string■对传递的数据所做的工作，以及它是否最终成为PickerEntity的Key。我们将跳过它，因为EntityEdit

标签的特殊结构并且被解析出来，其他的内容都会在PickerEntity■Key中结束，然后出现在m_listOrderTemp列表中。

所以，现在我们现在遍历了一个向量，并允许我们从ItemPicker的post back处理到达EntityInstanceIdEncoder.DecodeEntityInstanceId■string■，同时还可以控制输入。还剩下的是找到该Web控件的实例。

寻找攻击点

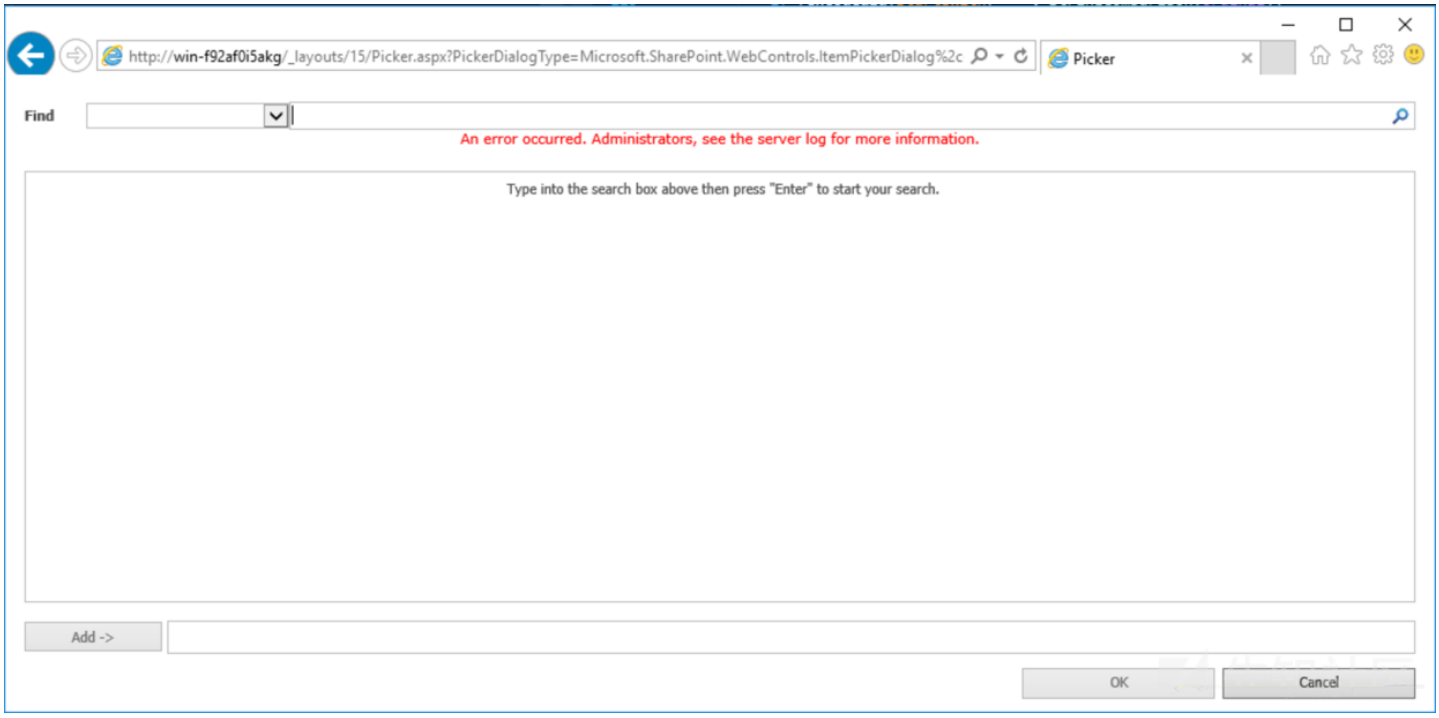
ItemPicker Web控件实际上从不直接在.aspx页面中使用。
但是当查看其基类型EntityEditorWithPicker时，我们发现在/_layouts/15/Picker.aspx中有一个使用它的Picker.aspx文件。

该页面要求使用对话框的类型并通过“PickerDialogType”URL参数以其程序集限定名称的形式。
在这里，可以使用以下两种ItemPickerDialog类型中的任何一种：

· Microsoft.SharePoint.WebControls.ItemPickerDialog in Microsoft.SharePoint.dll

· Microsoft.SharePoint.Portal.WebControls.ItemPickerDialog in Microsoft.SharePoint.Portal.dll

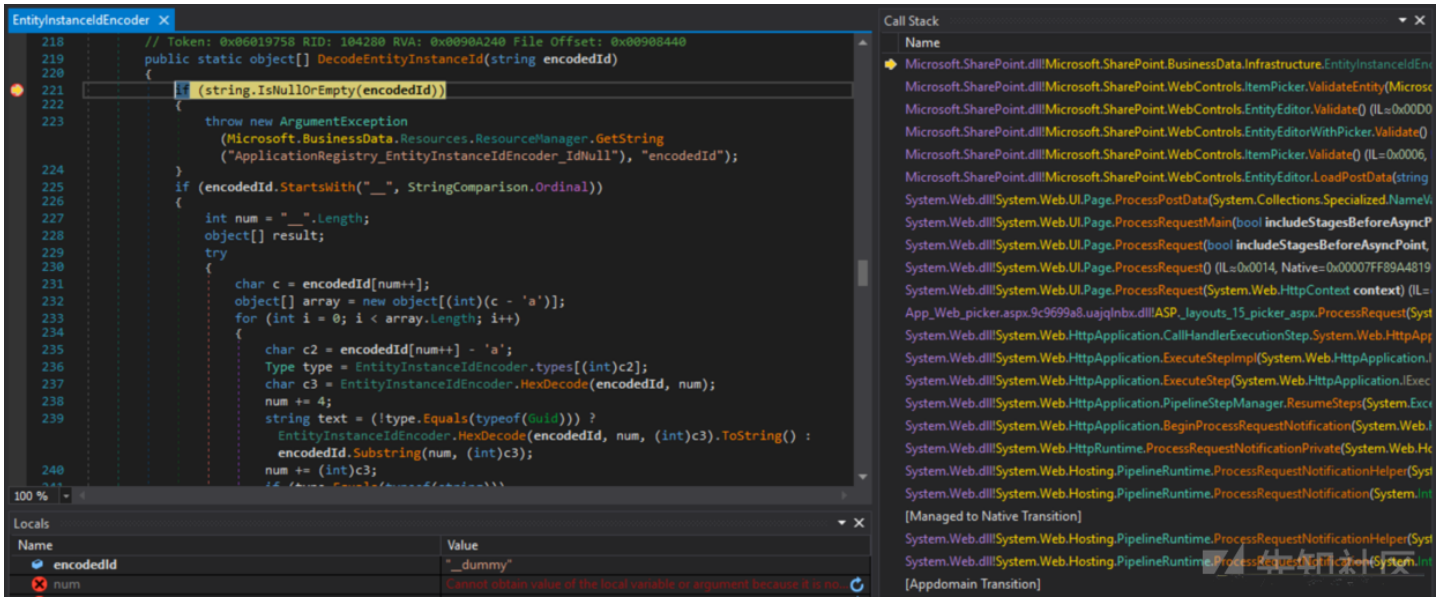
使用第一个ItemPickerDialog类型显示以下页面：



这里，底部文本字段与ItemPicker相关联。 并且还有HtmlInputHidden的通讯员，名字为ctl00 $ PlaceHolderDialogBodySection $ ctl05 $ hiddenSpanData。 这是我们的EntityInstanceIdEncoder.DecodeEntityInstanceId■string■sink的数据源。

概念证明

当使用以“”开头的**ctl00 $ PlaceHolderDialogBodySection $ ctl05 $ hiddenSpanData**值（如“dummy”）提交表单时，EntityInstanceIdEncoder.DecodeEntityInstanceId中的断点将显示以下情况。


EntityInstanceIdEncoder
// Token: 0x06019758 RID: 104280 RVA: 0x0090A240 File Offset: 0x00908440
public static object[] DecodeEntityInstanceId(string encodedId)
{
if (string.IsNullOrEmpty(encodedId))
{
throw new ArgumentException
(Microsoft.BusinessData.Resources.ResourceManager.GetString
("ApplicationRegistry_EntityInstanceIdEncoder_IdNull"), "encodedId");
}
if (encodedId.StartsWith("__", StringComparison.Ordinal))
{
int num = "__".Length;
object[] result;
try
{
char c = encodedId[num++];
object[] array = new object[(int)(c - 'a')];
for (int i = 0; i < array.Length; i++)
{
char c2 = encodedId[num++] - 'a';
Type type = EntityInstanceIdEncoder.types[(int)c2];
char c3 = EntityInstanceIdEncoder.HexDecode(encodedId, num);
num += 4;
string text = (!type.Equals(typeof(Guid))) ?
EntityInstanceIdEncoder.HexDecode(encodedId, num, (int)c3).ToString() :
encodedId.Substring(num, (int)c3);
num += (int)c3;
100 %
Locals
Name
Value
encodedId
"_dummy"
num
Cannot obtain value of the local variable or argument because it is no...
Call Stack
Name
Microsoft.SharePoint.dll!Microsoft.SharePoint.BusinessData.Infrastructure.EntityInstanceIdEnc
Microsoft.SharePoint.dll!Microsoft.SharePoint.WebControls.ItemPicker.ValidateEntity(Microsc
Microsoft.SharePoint.dll!Microsoft.SharePoint.WebControls.EntityEditor.Validate() (IL=0x00D0
Microsoft.SharePoint.dll!Microsoft.SharePoint.WebControls.EntityEditorWithPicker.Validate()
Microsoft.SharePoint.dll!Microsoft.SharePoint.WebControls.ItemPicker.Validate() (IL=0x0006,
Microsoft.SharePoint.dll!Microsoft.SharePoint.WebControls.EntityEditor.LoadPostData(string
System.Web.dll!System.Web.UI.Page.ProcessPostData(System.Collections.Specialized.NameV
System.Web.dll!System.Web.UI.Page.ProcessRequestMain(bool includeStagesBeforeAsyncP
System.Web.dll!System.Web.UI.Page.ProcessRequest(bool includeStagesBeforeAsyncPoint,
System.Web.dll!System.Web.UI.Page.ProcessRequest() (IL=0x0014, Native=0x00007FF89A4819
System.Web.dll!System.Web.UI.Page.ProcessRequest(System.Web.HttpContext context) (IL=
App_Web_picker.aspx.9c9699a8.uajqlnbx.dll!ASP._layouts_15_picker_aspx.ProcessRequest(Syst
System.Web.dll!System.Web.HttpApplication.CallHandlerExecutionStep.System.Web.HttpApp
System.Web.dll!System.Web.HttpApplication.ExecuteStepImpl(System.Web.HttpApplication.I
System.Web.dll!System.Web.HttpApplication.ExecuteStep(System.Web.HttpApplication.IExec
System.Web.dll!System.Web.HttpApplication.PipelineStepManager.ResumeSteps(System.Exce
System.Web.dll!System.Web.HttpApplication.BeginProcessRequestNotification(System.Web.H
System.Web.dll!System.Web.HttpRuntime.ProcessRequestNotificationPrivate(System.Web.Hc
System.Web.dll!System.Web.Hosting.PipelineRuntime.ProcessRequestNotificationHelper(Syst
System.Web.dll!System.Web.Hosting.PipelineRuntime.ProcessRequestNotification(System.Int
[Managed to Native Transition]
System.Web.dll!System.Web.Hosting.PipelineRuntime.ProcessRequestNotificationHelper(Syst
System.Web.dll!System.Web.Hosting.PipelineRuntime.ProcessRequestNotification(System.Int
[Appdomain Transition]

此时调用堆栈如下所示：

```
>
     Microsoft.SharePoint.dll!Microsoft.SharePoint.BusinessData.Infrastructure.E
ntityInstanceIdEncoder.DecodeEntityInstanceId(string encodedId) (IL=0x0000,
Native=0x00007FF89EA38790+0x29)
     Microsoft.SharePoint.dll!Microsoft.SharePoint.WebControls.ItemPicker.Valida
teEntity(Microsoft.SharePoint.WebControls.PickerEntity pe) (IL≈0x003E,
Native=0x00007FF89EA361B0+0xD7)
     Microsoft.SharePoint.dll!Microsoft.SharePoint.WebControls.EntityEditor.Vali
date() (IL≈0x00D0, Native=0x00007FF89E990740+0x2A3)
     Microsoft.SharePoint.dll!Microsoft.SharePoint.WebControls.EntityEditorWithP
icker.Validate() (IL=0x001B, Native=0x00007FF89E98FF10+0x7D)
     Microsoft.SharePoint.dll!Microsoft.SharePoint.WebControls.ItemPicker.Valida
te() (IL=0x0006, Native=0x00007FF89EA36010+0x2C)
     Microsoft.SharePoint.dll!Microsoft.SharePoint.WebControls.EntityEditor.Load
PostData(string postDataKey, System.Collections.Specialized.NameValueCollection
values) (IL=0x008D, Native=0x00007FF89E06A410+0x165)
     System.Web.dll!System.Web.UI.Page.ProcessPostData(System.Collections.Specia
lized.NameValueCollection postData, bool fBeforeLoad) (IL≈0x0193,
Native=0x00007FF89D075CD0+0x3A1)
     System.Web.dll!System.Web.UI.Page.ProcessRequestMain(bool
includeStagesBeforeAsyncPoint, bool includeStagesAfterAsyncPoint) (IL=0x0530,
Native=0x00007FF89A483B20+0x1024)
     System.Web.dll!System.Web.UI.Page.ProcessRequest(bool
includeStagesBeforeAsyncPoint, bool includeStagesAfterAsyncPoint) (IL=0x003C,
Native=0x00007FF89A481A60+0xA2)
     System.Web.dll!System.Web.UI.Page.ProcessRequest() (IL≈0x0014,
Native=0x00007FF89A4819B0+0x4B)
     System.Web.dll!System.Web.UI.Page.ProcessRequest(System.Web.HttpContext
context) (IL=epilog, Native=0x00007FF89A4809F0+0x69)
     App_Web_picker.aspx.9c9699a8.uajqlnbx.dll!ASP._layouts_15_picker_aspx.Proce
ssRequest(System.Web.HttpContext context) (IL=0x0007,
Native=0x00007FF89E050A50+0x2D)
     System.Web.dll!System.Web.HttpApplication.CallHandlerExecutionStep.System.W
eb.HttpApplication.IExecutionStep.Execute() (IL≈0x018D,
Native=0x00007FF89A47FF20+0x335)
     System.Web.dll!System.Web.HttpApplication.ExecuteStepImpl(System.Web.HttpAp
plication.IExecutionStep step) (IL=epilog, Native=0x00007FF8996CB5C0+0xD5)
     System.Web.dll!System.Web.HttpApplication.ExecuteStep(System.Web.HttpApplic
ation.IExecutionStep step, ref bool completedSynchronously) (IL≈0x0015,
Native=0x00007FF8996CB210+0x5B)
     System.Web.dll!System.Web.HttpApplication.PipelineStepManager.ResumeSteps(S
ystem.Exception error) (IL≈0x027A, Native=0x00007FF8996C8C20+0x74D)
     System.Web.dll!System.Web.HttpApplication.BeginProcessRequestNotification(S
ystem.Web.HttpContext context, System.AsyncCallback cb) (IL=0x0031,
Native=0x00007FF8996C8760+0x83)
     System.Web.dll!System.Web.HttpRuntime.ProcessRequestNotificationPrivate(Sys
tem.Web.Hosting.IIS7WorkerRequest wr, System.Web.HttpContext context)
(IL≈0x00B0, Native=0x00007FF8996C26E0+0x1E7)
     System.Web.dll!System.Web.Hosting.PipelineRuntime.ProcessRequestNotificatio
nHelper(System.IntPtr rootedObjectsPointer, System.IntPtr nativeRequestContext,
System.IntPtr moduleData, int flags) (IL≈0x0131,
Native=0x00007FF8996C0830+0x3FD)
     System.Web.dll!System.Web.Hosting.PipelineRuntime.ProcessRequestNotificatio
n(System.IntPtr rootedObjectsPointer, System.IntPtr nativeRequestContext,
System.IntPtr moduleData, int flags) (IL≈0x0000, Native=0x00007FF8996C03C0+0x13)
     [Managed to Native Transition]
     System.Web.dll!System.Web.Hosting.PipelineRuntime.ProcessRequestNotificatio
nHelper(System.IntPtr rootedObjectsPointer, System.IntPtr nativeRequestContext,
System.IntPtr moduleData, int flags) (IL≈0x01E7,
Native=0x00007FF8996C0830+0x4A4)
     System.Web.dll!System.Web.Hosting.PipelineRuntime.ProcessRequestNotificatio
n(System.IntPtr rootedObjectsPointer, System.IntPtr nativeRequestContext,
System.IntPtr moduleData, int flags) (IL≈0x0000, Native=0x00007FF8996C03C0+0x13)
     [Appdomain Transition]
```

当使用其他`ItemPickerDialog`类型时，只有两个最顶层的条目是不同的，如下所示：

```
>
     Microsoft.SharePoint.Portal.dll!Microsoft.Office.Server.ApplicationRegistry
.Infrastructure.EntityInstanceIdEncoder.DecodeEntityInstanceId(string encodedId)
(IL=0x0000, Native=0x00007FF89EA41610+0x23)
     Microsoft.SharePoint.Portal.dll!Microsoft.SharePoint.Portal.WebControls.Ite
mPicker.ValidateEntity(Microsoft.SharePoint.WebControls.PickerEntity pe)
(IL≈0x002C, Native=0x00007FF89EA40060+0x98)
```

这是ctl00 $ PlaceHolderDialogBodySection $ ctl05 $ hiddenSpanData的数据最终在EntityInstanceIdEncoder.DecodeEntityInstanceId中的证明过程。
其余的只是处理实体实例id编码并找到适当的XmlSerializerpayload。

■■■■■■■■■■■https://www.zerodayinitiative.com/blog/2019/3/13/cve-2019-0604-details-of-a-microsoft-sharepoint-rce-vulnerabili

上一篇：某sso开源单点登录系统后台代码执行 下一篇：漏洞分析：WordPress 5....

1. 1 条回复


HYWZ 2019-03-22 20:36:19

啊啊啊啊，大哥撞文了

0 回复Ta
