本篇文章是对Portswigger这篇文章的总结。目的在于，目前该漏洞的利用共修复与我自己做的有点不一样。

简介

AngularJS是一个很流行的JavaScript框架，通过这个框架可以把表达式放在花括号中嵌入到页面中。例如，表达式1+2={{1+2}}将会得到1+2=3。其中括号中的表达式被执行了，这就意味着，如果服务端允许用户输入的参数中带有花括号，我们就可以用Angular表达式来进行xss攻击。

一个客户端的用户输入

我们来探究下HTML页面是如何安全的防护用户输入。在下面这个例子中，我们会使用HTML中的Thymeleaf来编码，然后在页面中的div标签的text属性中输出username的值。

```
<html xmlns:th="http://www.thymeleaf.org">
<head>
<title>AngularJS - Escaping the Expression Sandbox</title>
</head>
<body>
<div th:text="${username}"></div>
</body>
</html>
```

如果username的值是alert('Rob')，输出的页面就是以下的样子：

```
<html xmlns:th="http://www.thymeleaf.org"><head><title>AngularJS - Escaping the Expression Sandbox</title></head><body><div&gt;alert(&#39;Rob&#39;)&lt;/script&gt;</div></body></html>
```

可以看到，输出的内容都被HTML编码了，这就是说，目前为止这个应用对xss攻击是可以防御的。

添加AngularJS

当前，我们的应用可以防护xss攻击。接下来，我们加入AngularJS来改下：

```
<html xmlns:th="http://www.thymeleaf.org">
<head>
<title>Angular Expression - safe</title>
<script src="angular-1.4.8.min.js"></script>
</head>
<body ng-app>
<div th:text="${username}"></div>
</body>
</html>
```

你可以发现，有两处修改了：

1. 引入了angular-1.4.8.min.js
2. 给body元素添加了ng-app

现在，我们的应用就很容易受到xss攻击了，但是，我们该如何攻击呢？

就像我们在简介中介绍的那样，如果把username改成1+2={{1+2}}会怎么样呢？

结果如下：

```
<html>
<head>
<title>Angular Expression - safe</title>
<script src="angular-1.4.8.min.js"></script>
</head>
<body ng-app="">
<div>1+2={{1+2}}</div>
</body>
</html>
```

Angular将会把DOM解析成如下：

```
<html>
<head>
<title>Angular Expression - safe</title>
<script src="angular-1.4.8.min.js"></script>
</head>
<body ng-app="">
<div>1+2=3</div>
</body>
</html>
```

可以看到括号里面的表达式被运行了，我们现在把username换成{{alert('Rob')}}试试，但是这样做被表达式沙盒拦截了。此时，我们可以认为我们写的页面是安全的，因为威胁语句被拦截了。

表达式沙盒化

在AngularJS中，沙盒化的目的并不是为了安全，更主要的是为了分离应用，例如，用户在获取window的时候是不被允许的，因为这样可以避免在你的程序中引入全局变量。

但是，如果在表达式被处理之前，有攻击者修改了页面模板，这样的情况沙盒是不会拦截的。也就是说，这种情况下，任何在花括号内的语句都能被执行，

所以Angular官方建议开发这类应用时，最好不要让用户可以修改客户端模板。具体建议如下：

> 不要把客户端和服务端模板混在一起
>
> 不要通过用户输入来动态的生成模板
>
> 不要用$scope.$eval运行用户输入内容
>
> 可以考虑使用CSP（也不要只依赖于CSP）

这些意味着如果应用页面允许用户的输入修改到客户端的模板中，那么这个页面将很容易被xss攻击，接下来我们来看看这个具体的例子：

绕过表达式沙盒

如果我们的payload被沙盒化了，我们该怎么样绕过呢？

如果我们的username是以下的值，将会发生什么呢？

```
{{'a'.constructor.prototype.charAt=[].join;eval('x=1} } };alert(1)//');
}}
```

上面的例子中，通过覆盖原始函数charAt，我们就可以绕过Angular的表达式沙盒，并且执行我们的语句alert(1)。具体的攻击原理可以参考http://blog.portswigger.net/2016/01/xss-without-html-client-side-template.html。

注意，这些测试只是在Chrome和AngularJS1.4.8中成功。在其他的浏览器中不知道能不能成功。

结论

如果服务端允许用户输入到Angular模板，这将会让你的应用陷入xss的攻击中。不过话说回来，最好不要把服务端的用户输入和客户端模板混合起来。
关于我们你可以从这篇文章里面了解详细的内容https://github.com/rwinch/angularjs-escaping-expression-sandbox

*** 文章来源：spring.io，FB小编/老王隔壁的白帽子翻译，转载请注明来自FreeBuf黑客与极客（FreeBuf.COM）**