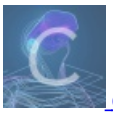
从Share your mind到RPO 攻击

原创

Sp4rkW 于 2018-03-28 21:44:17 发布 639 收藏

文章标签： RPO攻击

本文链接：https://blog.csdn.net/wy_97/article/details/79734731



前言

强网杯初赛已经告一段落，大佬们的`wp`也逐渐放出，这篇博客就从强网杯初赛`web`的`Share your mind`这题来探讨一下`RPO`攻击~

前置链接：

题目名称：Share your mind

第二届强网杯Web Writeup by @l3m0n

ROP is what?

`RPO(Relative Path Overwrite)`相对路径覆盖，是一种新型攻击技术，最早由Gareth Heyes在其发表的文章中提出。主要是利用浏览器的一些特性和部分服务端的配置差异导致的漏洞，通过一些技巧，我们可以通过相对路径来引入其他的资源文件，以至于达成我们想要的目的。

就目前来看此攻击方法依赖于浏览器和网络服务器的反应，基于服务器的Web缓存技术和配置差异，以及服务器和客户端浏览器的解析差异，利用前端代码中加载的css/js的相对路径来加载其他文件，最终浏览器将服务器返回的不是css/js的文件当做css/js来解析，从而导致XSS，信息泄露等漏洞产生。

初识 RPO 攻击

此部分来自RPO攻击 @Ethan

来自`google`的一个例子：

```
http://www.google.com/tools/toolbar/buttons/apis/howto_guide.html
```

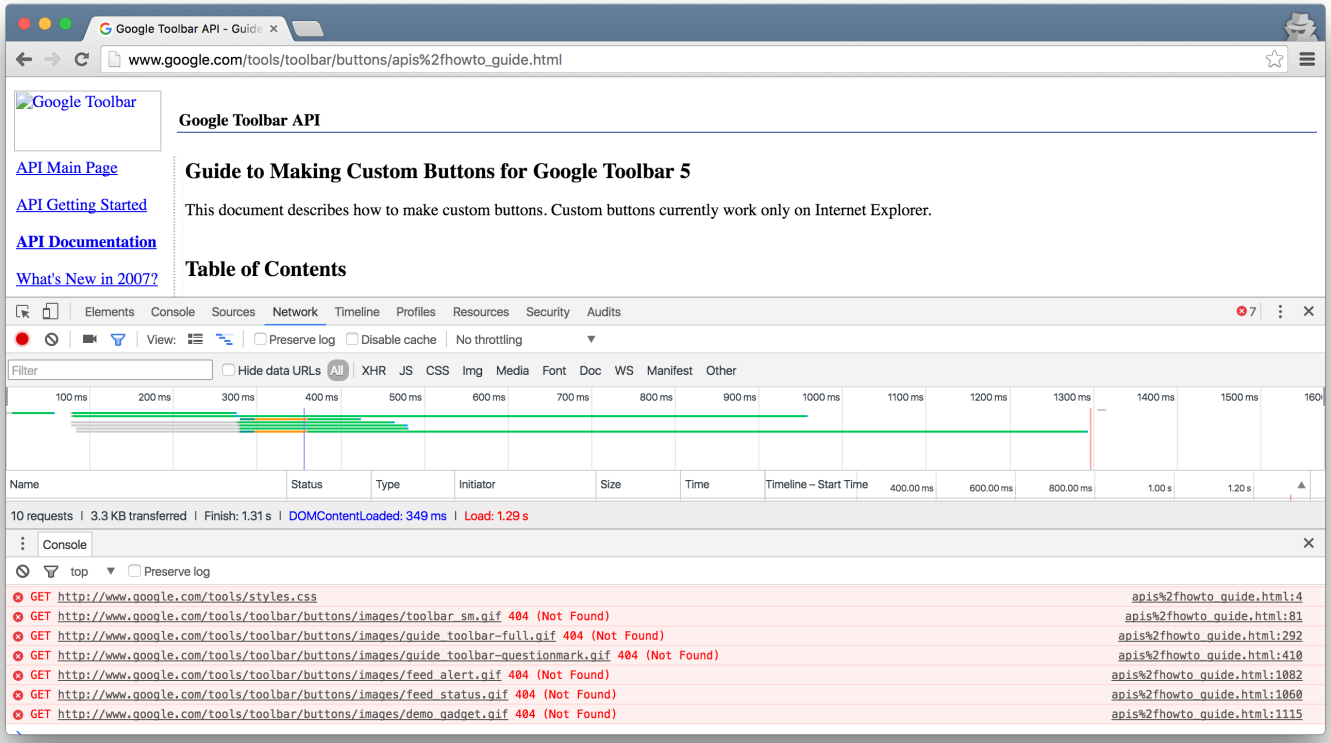
攻击利用点：

```
<html>
<head></head>
<title>Google Toolbar API -Guide to Making Custom Buttons</title>
<link href="../../styles.css" rel="stylesheet"type="text/css"/>
[..]
</html>
```

从上述代码中我们很容易看到`../../`这样的`css`引用结构
下一步是找出目标服务器解析路径的方式。对**浏览器**来说，目录是通过 `/` 来分隔的，然而对**服务器**来说，路径中包含 `/` 并不意味着存在一个目录。例如，`JSP`接收路径参数是以;作为分隔符的，例如`http://example.com/path;/notpath`，而浏览器并不识别这种模式，而当做是存在目录的路径。
同样，`Google Toolbar`(谷歌工具栏)也有它自己独特的解析路径的方式。在发送请求给服务器之前，会对请求进行处理并解码所有的路径，但是谷歌浏览器并不会强制转换`%2f`为`/`,因此可以将路径中的/替换为`%2f`。

跳目录

访问`http://www.google.com/tools/toolbar/buttons/apis%2fhowto_guide.html`



针对上面的链接，服务器和浏览器对路径的解析是不一致的.

- 服务器视角: `/tools/toolbar/buttons/apis/` + howto_guide.html
- 浏览器视角: `/tools/toolbar/buttons/` + apis%2fhowto_guide.html
- 导入的css样式: `/tools/` + toolbar/buttons/…/…/style.css

以上”+”左边高亮部分表示基本路径，浏览器认为基本路径是 `/tools/toolbar/buttons/` 而不是 `/tools/toolbar/buttons/apis/`, 因此导入相对路径的样式 `../../style.css` 会额外多跳一层目录路径，本应该导入`css`的路径为`”tools/toolbar/style.css”`, 而现在为`”tools/style.xss”`。

制造假目录

除了跳目录以外，还能制作假目录，例如，我们想在导入的样式路径为`/tools/fake/styles.css`, 可以构造如下`url`:`http://www.google.com/tools/fake/..%2ftoolbar/buttons/apis%2fhowto_guide.html`

- 服务器视角: /tools/fake/…/toolbar/buttons/apis/ + howto_guide.html
- 浏览器视角: tools/fake/…%2ftoolbar/buttons/ + apis%2fhowto_guide.html
- 导入的css样式: /tools/fake/…%2ftoolbar/buttons/…/…/ + style.css

这里我们添加了两个虚假的路径:`fake`/和`..%2f`,以便他们能在服务器相互抵消，同时浏览器认为`fake/`是一个真实的目录，并且，`..%2ftoobar`是另外一个目录。理论上，我们可以再根路径上导入任何样式，通过`https://www.google.com/*/styles.css`,然而不幸的是代理只在`https://www.google.com/tools/*/styles.css`路径上有效，换句话说，任何在`/tools/`路径之外的路径采用编码魔术(`%2`f代替`/`)将不起作用,也就是说，只能在`https://www.google.com/tools/*/styles.css`导入任何样式。

PS：很遗憾的是，上述**google**的这个例子早在**14年**被发现，现已无法验证，故后续相关利用这里不再继续说明，下面用18强网杯的**share your mind**来举例说明`RPO`的利用方法~

share your mind，biubiubiu！

参考链接：强网杯-writeup @Pupil

```
    </div>
</div>


    <script src="../static/js/jquery.min.js"></script>
    <script src="../static/js/bootstrap.min.js"></script>
  </body>
</html>
```


注意到`index.php`存在相对路径引用，考虑RPO
新建一个文章，内容输入js代码比如alert(1)

然后访问`http://39.107.33.96:20000/index.php/view/article/635/..%2f..%2f..%2f..%2findex.php`
把635替换成你的文章代码,这里对于服务器来说访问的是

```
http://39.107.33.96:20000/index.php
```

但是对于浏览器来说他访问的就是

```
http://39.107.33.96:20000/index.php/view/article/635/..%2f..%2f..%2f..%2findex.php
```

然后这个时候浏览器会发起`js`请求去请求原本`index.php`会加载的`../static/js/bootstrap.min.js`就是向

```
http://39.107.33.96:20000/index.php/view/article/635/..%2f..%2f..%2f..%2findex.php/../static/js/bootstrap.min.js
```

相当于

```
http://39.107.33.96:20000/index.php/view/article/635/static/bootstrap.min.js
```

这里访问的结果和访问

```
http://39.107.33.96:20000/index.php/view/article/635/
```

也就是你的文章的内容是一样的(不明白的可以自己本地测试)，不同的是浏览器是以`js`引擎去解析你的文章的，也就是会把你的文章**当成一段js去执行**。所以这里就可以绕过`<>`的过滤执行`xss`了。

ps：这里还有一个点，Mark一下就是`String.fromCharCode`解决过滤了`"`和`'`

补充链接：RPO攻击技术浅析