基础漏洞 csrf 挖掘

0x01:漏洞原理：

当我们可以使目标对象浏览器发送 HTTP 请求到别的 Web 网站时,就会发生跨站请求伪造（CSRF)攻击。该 Web 网站会执行一些操作,使得请求看起来是有效的,并且发自目标对象。这种攻击一般依赖于目标对象之前已经通过了具有漏洞的网站的身份认证,并且攻击者向该网站发起的提交动作和网站的响应不被目标对象感知。**当 CSRF 攻击成功时,我们就可以修改服务器端信息并很有可能完全接管用户的账号。**

下面用一个列子来讲解：

李华登录他的网银并查看余额。

李华查看完余额后,**登录不同域下**的邮箱账号并查收邮件。

李华看到了一封具有连接到不熟悉网站的链接的邮件，并打开了这个链接以查看相关内容。

当链接的网站内容加载时,该网站要求李华的浏览器向李华的网银发起一个 HTTP 请求,请求从李华的账号转账资金到攻击者的账号。

李华的网银接收到了不熟悉的(也是恶意的）网站发起的 HTTP 转账请求,但是由于网银没有任何 CSRF 防护机制,就处理了这次转账申请。

简而言之 csrf 攻击就是利用了网站用于请求进行身份认证的进程的缺陷。

0x02：对身份认证的理解：

简而言之 csrf 攻击就是利用了网站用于请求进行身份认证的进程的缺陷。

身份认证一般有基础身份认证协议、session、cookie 等方式，在此处我们重点了解 cookie 的方式，但是在 csrf 漏洞挖掘中，我们也会经常利用基础认定协议来挖掘。

cookie 是由网站创建并存储在用户端浏览器中的小文件。网站使用 cookie 有很多目的,例如存储用户的爱好或者用户访问网站的历史记录等。cookie 具有特定的被标准化为信息片的属性。这些属性告知浏览器 cookie 是做什么的,以及该如何处理它们。有些 cookie 属性可能会包含 domain、expires、max-age、 secure 和 httponly 等，

除了属性之外，cookie 也可能会包含-个名/值对,由一个标识符及其相关联的要发往网站（cookie 的 domain 属性定义了信息要发往的网站)的数值构成。)

浏览器定义网站能够设置的 cookie 的数量。但是一般来说，单个网站能够在通用浏览器中设置 50 ～150 个 cookie，据报道,在有些浏览器中可以设置多达 600 个以上的 cookie。浏览器通常允许网站使用每个 cookie 最大 4KB 的存储空间。cookie 的名和值没有标准，也就是说，网站可以自由地选择自己的名/值对及其目的地址。例如，网站可以选择命名为 sessionId 的名字,以便于记住用户是谁，而不用在用户访问每个网页或进行每次操作时都要求他们重新输入用户名和口令。**(HTTP 请求是无状态的。无状态意味着网站并不知道每次 HTTP 请求对应的用户是谁,因此,它必须对每次请求都重新进行用户认证。)**

作为一个例子:

cookie 中的名/值对可能是 sessionbId=9f86d081884c7d659a2feaaoc55ado15a3bf4f1b2bob822cd15d6c15bofooa08，并且 cookie 具有一个.baidu.com 域。因此，

sessionbld cookic 将被发送到用户访问的每个,<baidu>.com 网站,例如,foo.baidu.com、bar.baidu.com、www.baidu.com 等。

secure 和 httponly 属性告诉浏览器什么时候和怎样发送和读取 cookie。这些属性不包含取值,而是表示了在 cookie 中是否出现的标志。当 cooki 中包含 secure 属性时,浏览器将只在访问 HTTPS 网站时发送 cookie。例如,如果你带着一个 secure cookie 访问 http://www.<site>.com/(一个 HTTP 网站),你的浏览器将不会发送 cookie 到这个网站。原因就是为了保护你的个人隐私,因为 HTTPS 连接是加密的而 HTTP 连接是不加蜜的。**httponly 属性(这个对于 xss 玩的好的师傅肯定都知道这个的重要)**因此,浏览器不允许任何脚本语言,例如 JavaScript,来读取 cookie 的值。当 cookie 中没有设置 secure 和 httponly 属性时,这些 cookie 将被正常发送并且可能会被恶意读取。没有设置 secure 属性的 cookie 可能被发送到一个非 HTTPS 网站;同样地,没有设置 httponly 属性的 cookie 可能被 JavaScript 脚本读取。*

expires 和 max-age 属性表示 cookie 什么时候过期并被浏览器删除。expires 属性简单地告诉浏览器在一个确定的时间删除 cookie。例如,cookie 可以设置该属性为 expires=March 13, 2022 12:00:00 UTC(世界时间:2022 年 3 月 13 日,星期日)。对应地,max-age 属性是直到 cookie 超时的秒数,并以一个整数(max-age=300)的形式体现。

0x03:两种请求方式的 csrf 漏洞:

get 请求发起的 csrf:

恶意网站利用李华网银的方式取决于银行是通过 GET 请求还是通过 POST 请求接受转账申请的。如果 李华的网银接受**通过 GET 请求进行转账,恶意网站就会发送具有隐藏表单或者<img>标签的 HTTP 请求。GET 和 POST 方法都依赖于能够让浏览器发送质要的 HTTP 请求的 HTML 语法,并且两种方法都可以采用隐藏表单技术,但是只有 GEL 方法可以使用<img>标签技术**。

攻击者需要在发送给李华网银的 HTTP 转账请求中包含李华的 cookie 信息。但是因为攻击者没有办法读取到李华的 cookie,所以他不能只是简单地生成一个 HTTP 请求并发送给李华的网银。相反,攻击者可以使用 HTML 的<img>标签来精心生成一个包含李华 cookie 的 GET 请求。**<img>标签对网页上的图像进行渲染,并且包含一个用来告诉浏览器去哪里定位图像文件的 src 属性。当浏览器渲染<img>标签时,它会向标签的 src 属性发起一个包含任何已有的 cookie 的 HTTP GET 请求。**因此,假定恶意网站使用类似如下从 李华的账号转账 500 元到小明的账号的 URL:

http://www.bank.com/Transfer.php?toBankId=xm&money=500

仅仅须要一个 HTTP 请求。就能够构造一次简单的 CSRF。

危急站点 B:它里面有一段 HTML 的代码例如以下:

```
<img
src=http://www.mybank.com/Transfer.php?toBankId=xm&money=500
>
```

结果是,当李明访问我们准备好 poc 的网站时,网站会在其 HTTP 响应中包含上面的<img>标签，从而浏览器会讲一步发起向银行的 HTTP GET 请求。浏览器发送李华的身份认证信息以获取其认为的对应图片的相关资源信息。但是实际上，银行接收到了这个请求，处理了 URL 中的<img>标签的 src 属性，生成了转账申请。

post 请求发起的 csrf：

这种类型的 CSRF 危害没有 GET 型的大，利用起来通常使用的是一个自动提交的表单，如：

```
<form action=http://wooyun.org/csrf.php method=POST>
    <input type="text" name="xx" value="11" /></form><script>
document.forms[0].submit(); </script>
```

访问该页面后，表单会自动提交，相当于模拟用户完成了一次 POST 操作。

0x04：案例：

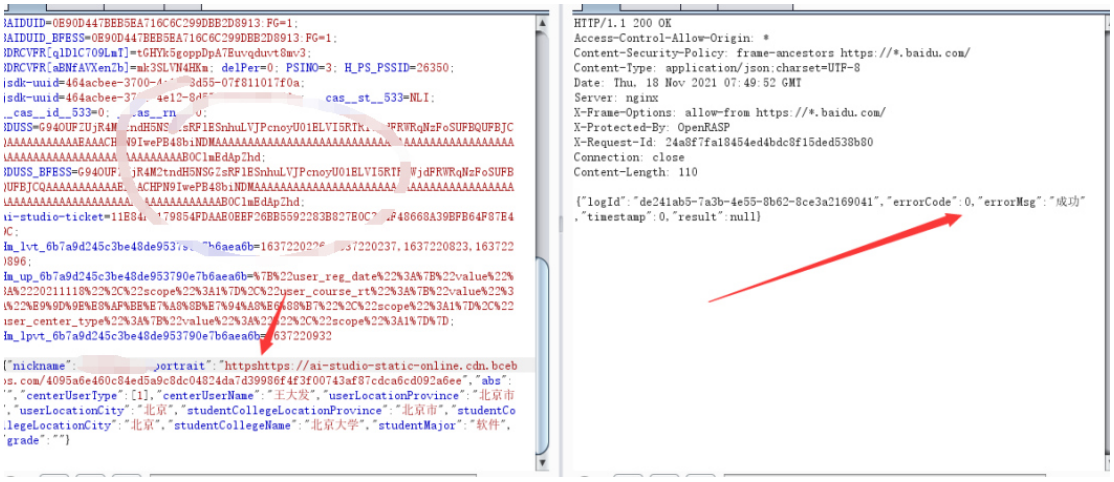
挖掘环境：

某度的一个站点的个人信息中心，发现保存图片的参数可控，于是发现有了下列：

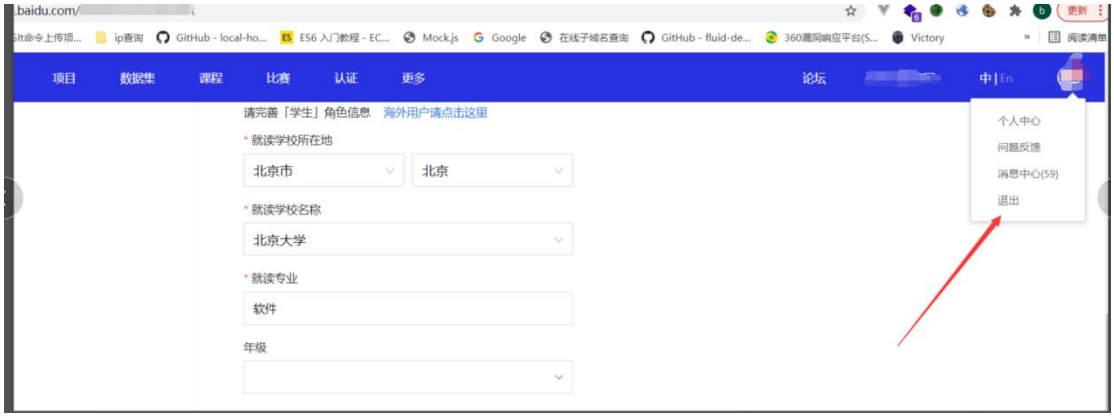
登录之后来到个人中心

再点击更换头像的时候抓包：

可以发现 portrait 参数 就是存头像的地址，把这个包发送到 repeat 尝试修改：



于是我们就去获取退出时的链接来替换此时的头像链接：



https://xxxxx.baidu.com/studio/logout?r=https://passport.baidu.com/?logout&u=http%3A%2F%2Fxxxx.baidu.com%2Faistudio%2Fuserinfoedit

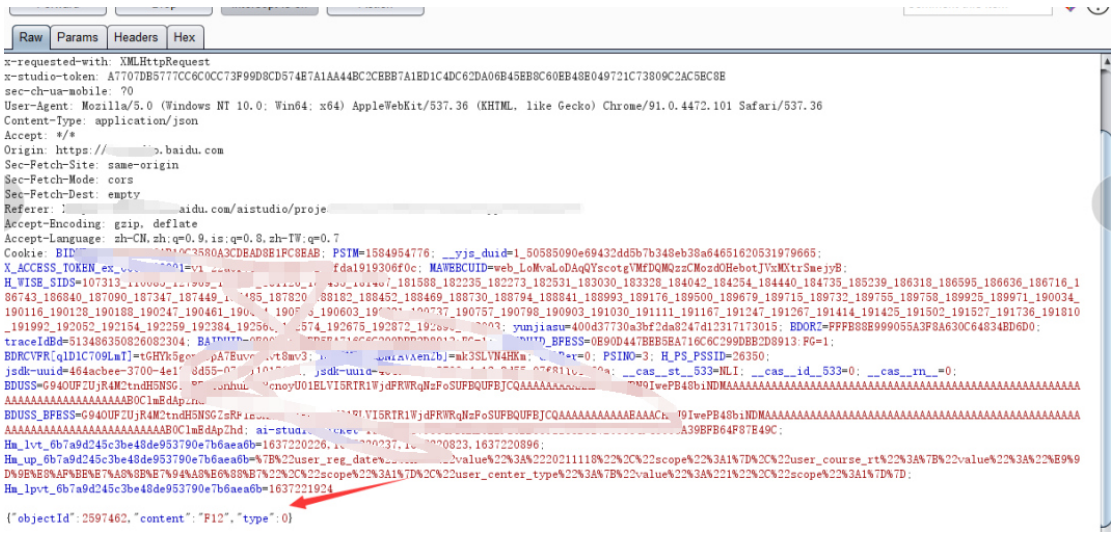
这是退出的请求，也是我们下面必须的操作，具体利用过程是下面的操作，上面的操作是解释这一个点：

这个漏洞的思想就是利用 csrf 漏洞让别人在访问该站点的某个地方的时候突然掉线，而

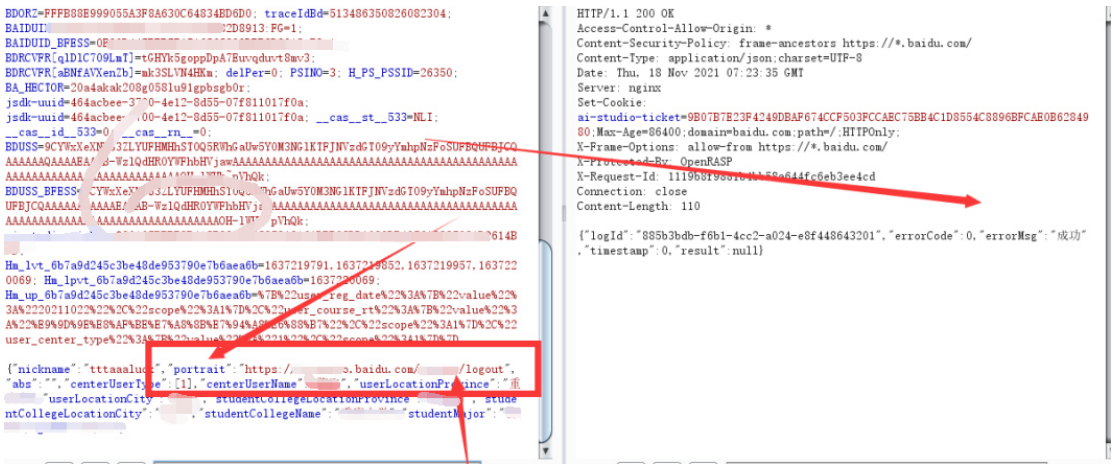
有交互的地方就是在论坛、评论、留言板等地方。

回到上述案例中，我们在该站点中发现了有评论留言的地方：

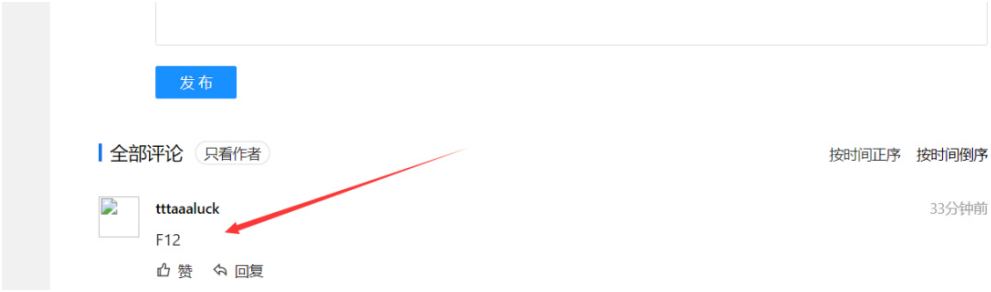
于是开始评论，然后发表的时候抓包：



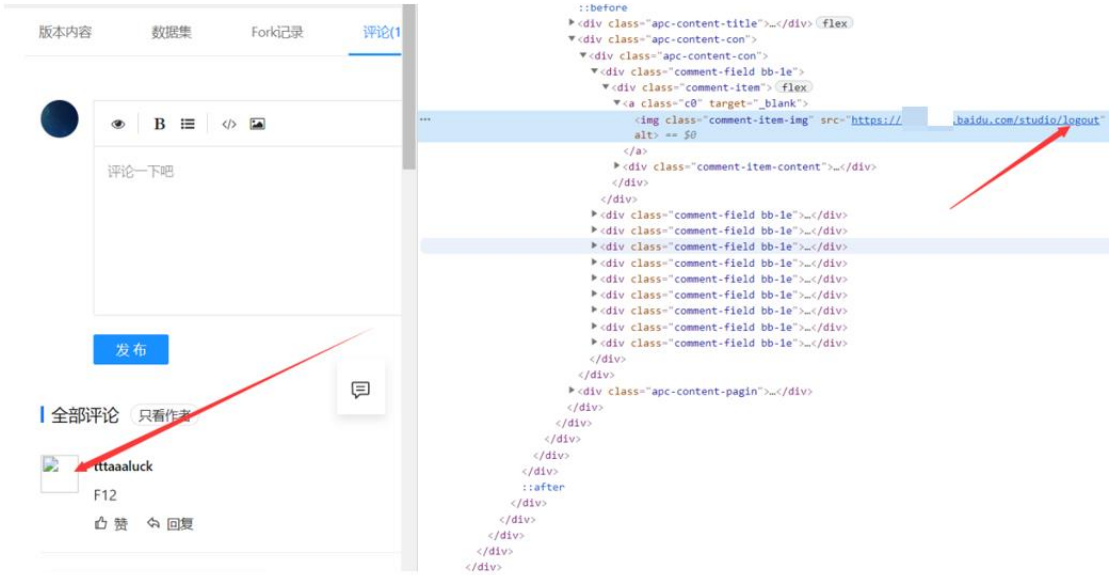
然后放包后会抓取到获取我们的个人信息的资料的包，此时我们就可以进行操作了：



然后放包即可评论成功，但是头像处的链接被我换成了退出连接，当别的用户来访问时，自动就退出登录：



进行元素审查，我们可以发现此时头像位置是退出登录的链接



此时当别的用户访问到我们的留言时，系统也是自动就退出来。

在企业 src 上，该漏洞为中危，但在项目时，可以达到高，f12 免费星球有讲诉过此漏洞，在 tsrc 也有许多此类的漏洞

星球文章链接：https://articles.zsxq.com/id_b8c7ae0wm8sj.html

总结:对于此类 csrf 漏洞我们应该多关注可以多用户进行交互的地方，然后观察自己的个人信息是否可以可控，进而继续挖掘。