CSRF跨站请求伪造

### 攻击过程

CSRF（XSRF）尽管听起来很像XSS跨站脚本攻击，但是它与XSS完全不同。XSS是利用站点内的信任用户，而CSRF则是通过伪装来自受信任用户的请求来利用受信任的站点。

与XSS相比，CSRF攻击不大流行和难以防范，所以比XSS更具危险性。

以下是一个CSRF的例子

受害者 Bob 在银行有一笔存款，通过对银行的网站发送请求http://bank.example/withdraw?account=bob&amount=1000000&for=bob2可以使 Bob 把 1000000 的存款转到 bob2 的账号下。通常情况下，该请求发送到网站后，服务器会先验证该请求是否来自一个合法的 session，并且该 session 的用户 Bob 已经成功登陆。

黑客 Mallory 自己在该银行也有账户，他知道上文中的 URL 可以把钱进行转帐操作。Mallory 可以自己发送一个请求给银行：http://bank.example/withdraw?account=bob& amount=1000000&for=Mallory。但是这个请求来自 Mallory 而非 Bob，他不能通过安全认证，因此该请求不会起作用。

这时，Mallory 想到使用 CSRF 的攻击方式，他先自己做一个网站，在网站中放入如下代码：<img src=”http://bank.example/withdraw?account=bob&amount=1000000&for=Mallory” />。并且通过广告等诱使 Bob 来访问他的网站。当 Bob 访问该网站时，上述 url 就会从 Bob 的浏览器发向银行，而这个请求会附带 Bob 浏览器中的 cookie 一起发向银行服务器。大多数情况下，该请求会失败，因为他要求 Bob 的认证信息。但是，如果 Bob 当时恰巧刚访问他的银行后不久，他的浏览器与银行网站之间的 session 尚未过期，浏览器的 cookie 之中含有 Bob 的认证信息。这时，悲剧发生了，这个 url 请求就会得到响应，钱将从 Bob 的账号转移到 Mallory 的账号，而 Bob 当时毫不知情。等以后 Bob 发现账户钱少了，即使他去银行查询日志，他也只能发现确实有一个来自于他本人的合法请求转移了资金，没有任何被攻击的痕迹。而 Mallory 则可以拿到钱后逍遥法外。

### 防止CSRF跨站请求伪造

1. 对于web站点，将持久化的授权方法（例如cookie或者HTTP授权）切换为瞬时的授权方法（在每个form中提供隐藏field）。
2. “双提交”cookie。此方法只工作于Ajax请求，但它能够作为无需改变大量form的全局修正方法。如果某个授权的cookie在form post之前正被JavaScript代码读取，那么限制跨域规则将被应用。什么叫限制跨域规则呢？限制跨域规则就是：如果服务器需要在Post请求体或者URL中包含授权cookie的请求，那么这个请求必须来自于受信任的域，因为其它域是不能从信任域读取cookie的。上面那个例子的受信任域就是银行网站的某个域，而Mallory发给Bob的链接不是受信任的域。
3. 使用Post代替Get。Post方式不会在web服务器和代理服务器日志中留下数据尾巴，然而Get方式却会留下数据尾巴。
4. 以上三点都是正对web站点的防御手段，第4点是从用户的角度的防御手段。通过在浏览其它站点前登出站点或者在浏览器会话结束后清理浏览器的cookie来防止CSRF攻击。