# 跨站请求伪造

**跨站请求伪造**（英語：Cross-site request forgery），也被称为**one-click attack**或者**session riding**，通常缩写为**CSRF**或者**XSRF**，是一种挟制用户在当前已登录的Web应用程序上执行非本意的操作的攻击方法。[1] 跟跨網站指令碼（XSS）相比，XSS利用的是用户对指定网站的信任，CSRF利用的是网站对用户网页浏览器的信任。

## 攻擊的細節

跨站請求攻擊，簡單地說，是攻擊者通過一些技術手段欺騙用戶的瀏覽器去訪問一個自己曾經認證過的網站並執行一些操作（如發郵件，發消息，甚至財產操作如轉帳和購買商品）。由於瀏覽器曾經認證過，所以被訪問的網站會認為是真正的用戶操作而去執行。這利用了web中用戶身份驗證的一個漏洞：簡單的身份驗證只能保證請求是發自某個用戶的瀏覽器，卻不能保證請求本身是用戶自愿發出的。

### 例子

假如一家銀行用以執行轉帳操作的URL地址如下：
`https://bank.example.com/withdraw?account=AccoutName&amount=1000&for=PayeeName`

那麼，一個惡意攻擊者可以在另一个網站上放上 `<img src="http://www.examplebank.com/withdraw?account=Alice&amount=1000&for=Badman" />`

如果有賬戶名為Alice的用戶訪問了惡意站點，而她之前剛訪問過銀行不久，登錄信息尚未過期，那麼她就會損失1000資金。

這種惡意的網址可以有很多種形式，藏身於網頁中的許多地方。此外，攻擊者也不需要控制放置惡意網址的網站。例如他可以將這種地址藏在論壇，博客等任何用戶生成內容的網站中。這意味著如果伺服器端沒有合適的防禦措施的話，用戶即使訪問熟悉的可信網站也有受攻擊的危險。

透過例子能夠看出，攻擊者並不能通過CSRF攻擊來直接獲取用戶的帳戶控制權，也不能直接竊取用戶的任何信息。他們能做到的，是欺騙用戶的瀏覽器，讓其以用戶的名義執行操作。

## 防禦措施

### 令牌同步模式

令牌同步模式（英語：Synchronizer token pattern，简称STP）。原理是：当用户发送请求时，服务器端应用将令牌（英語：token，一个保密且唯一的值）嵌入HTML表格，并发送给客户端。客户端提交HTML表格时候，会将令牌发送到服务端，令牌的验证是由服务端实行的。令牌可以通过

任何方式生成，只要确保随机性和唯一性（如：使用随机种子的哈希链）。这样确保攻击者发送请求时候，由于没有该令牌而无法通过验证。

Django框架默认带有STP功能：[2]

```
<form method="post">
    {% csrf_token %}
</form>
```

渲染后的效果如下：

```
<form method="post">
    <input type="hidden" name="csrfmiddlewaretoken" value="KbyUmhTLMpYj7CD2di7JKP1P3qmLlkPt" />
</form>
```

STP能在HTML下運作順利，但會導致服务端的复杂度升高，复杂度源于令牌的生成和验证。因为令牌是唯一且随机，如果每个表格都使用一个唯一的令牌，那么当页面过多时，服务器由于生产令牌而导致的负担也会增加。而使用会话（英語：session）等级的令牌代替的话，服务器的负担将没有那么重。

### 檢查Referer字段

HTTP頭中有一個Referer字段，這個字段用以標明請求來源於哪個地址。在處理敏感數據請求時，通常來說，Referer字段應和請求的地址位於同一域名下。以上文銀行操作為例，Referer字段地址通常應該是轉帳按鈕所在的網頁地址，應該也位於bank.example.com之下。而如果是CSRF攻擊傳來的請求，Referer字段會是包含惡意網址的地址，不會位於bank.example.com之下，這時候伺服器就能識別出惡意的訪問。

這種辦法簡單易行，工作量低，僅需要在關鍵訪問處增加一步校驗。但這種辦法也有其局限性，因其完全依賴瀏覽器發送正確的Referer字段。雖然http協議對此字段的內容有明確的規定，但並無法保證來訪的瀏覽器的具體實現，亦無法保證瀏覽器沒有安全漏洞影響到此字段。并且也存在攻擊者攻擊某些瀏覽器，篡改其Referer字段的可能。

### 添加校驗token

由於CSRF的本質在於攻擊者欺騙用戶去訪問自己設置的地址，所以如果要求在訪問敏感數據請求時，要求用戶瀏覽器提供不保存在cookie中，并且攻擊者無法偽造的數據作為校驗，那麼攻擊者就無法再執行CSRF攻擊。這種數據通常是表單中的一個數據項。伺服器將其生成並附加在表單中，其內容是一個偽亂數。當客戶端通過表單提交請求時，這個偽亂數也一並提交上去以供校驗。正常的訪問時，客戶端瀏覽器能夠正確得到並傳回這個偽亂數，而通過CSRF傳來的欺騙性攻擊中，攻擊者無從事先得知這個偽亂數的值，伺服器端就會因為校驗token的值為空或者錯誤，拒絕這個可疑請求。

## 参考资料


1. Ristic, Ivan. Apache Security. O'Reilly Media. 2005: 280. ISBN 0-596-00724-8.
2. Cross Site Request Forgery protection | Django documentation | Django.

docs.djangoproject.com. [2020-01-21]. （原始内容存档于2020-01-23）.