doi: 10.3969/j.issn.1671-1122.2013.03.018

OAuth 协议安全性研究

张天琪

（北京工业大学，北京 100022）

摘 要： OAuth 是当下最流行的开放式授权协议，目前在整个互联网得到了相当广泛的应用。文章主要对 OAuth 协议的整体安全性进行了研究，通过测试案例展示其安全问题造成的危害，并提出了相应的防御措施以及整体安全性的提升方法。

关键词： OAuth；授权协议；账号劫持；开放式授权

中图分类号： TP393.08 **文献标识码：** A **文章编号：** 1671-1122（2013）03-0068-03

Study on OAuth Protocol Security

ZHANG Tian-qi

(*Beijing University of Technology, Beijing* 100022,*China*)

Abstract: OAuth is now the most popular open standard for authorization.Among the Internet,a wide range of applications has been used. This article focuses on the overall security of the OAuth authorization protocol study, displays the harm caused by those security issues through the test case,and puts forward the corresponding defensive measures, as well as the overall security upgrade method.

Key words: OAuth; authorization protocol; account hijacking; open standard for authorization

1 OAuth 协议特性

OAuth（开放授权）是一个开放标准，允许用户让第三方应用访问该用户在某一网站上存储的私密资源（如照片、视频、联系人列表），而无需将用户名和密码提供给第三方应用[1]。OAuth 是一种授权协议而非认证协议。认证关注的是用户是谁，一般通过用户名和密码来进行认证。授权关注的是用户在做什么，它以资源为主体，通常的授权形式是组和权限。如果滥用授权和认证协议，就会导致很多安全隐患。OAuth 协议最大的进步是能够使第三方在不获得目标网站账号和密码的情况下使用目标网站的用户资源，这也是当前 OAuth 应用如此之广的一个原因。随着开放式 REST（representational state transfer，表述性状态转移）风格的 API 的广泛使用，OAuth 协议也被应用地越来越广泛，现在很多网站建立了自己的开放平台，这些平台会提供、执行一些操作的 API，有了这些操作的 API，就必然要有安全保证，OAuth 协议也就应运而生。OAuth 协议对用户、消费方和服务提供者都有极大好处。对于用户来说免去了繁琐的注册过程，降低了注册成本，提高了用户体验；对于消费方来说简化了自身会员系统，同时又带来更多的用户和流量；对于服务提供者来说围绕自身进行开发可以增加用户黏性。OAuth 协议分为 1.0 和 2.0 版本，下面将分别探讨它们及它们的安全问题。

2 OAuth1.0 及安全问题

国内很多应用的 OAuth 协议已经升级到了 2.0 版本，但是部分电商因业务比较广泛，使得升级 2.0 版本付出的代价很大，故而仍然使用 1.0 版本的 OAuth 协议。

OAuth1.0 授权流程并不复杂，但是实际上由于授权过程中存在很多密钥、令牌及签名运算，它们相互之间又存在各种各样的牵连，因而开发人员往往关注于繁琐的授权过程而忽略了一些可能造成安全隐患的地方。OAuth1.0 中存在一个比较严重的安全漏洞：session fixation attack（会话固话攻击）[2]，产生这个攻击的原因主要是用户被同意授权后，用户被跳转到一个回调地址，这个回调地址一般是在第三方平台。如果对这个回调地址没有进行检查和限制，也没有任何机制保证整个授权流程只由一个人完成，那么就会造成安全问题，导致攻击者可以访问目标网站的用户资源。以下是该安全漏洞的攻击流程：攻击者首先启动授权流程，记录未授权的 request_token，用当前 url 诱骗用户访问。用户被同意授权后，被重定向到其他位置。攻击者之前记录的 request_token 此时已经被目标授权，使用由这个令牌构造的相应 url 即可访问用户资源。

收稿时间： 2012-12-15

作者简介： 张天琪（1992-），男，北京，本科，主要研究方向：Web 相关安全研究。

下面给出了一个新浪微博存在的此类安全漏洞：

缺陷编号： WooYun-2010-00781
漏洞标题： Sina 微博OAuth 提供者存在session fixation attack漏洞
相关厂商： 新浪
漏洞作者： 路人甲
提交时间： 2010-11-03
公开时间： 2010-11-08
漏洞类型： 设计缺陷/逻辑错误
危害等级： 中
自评Rank： 10
漏洞状态： 漏洞已经通知厂商但是厂商忽略漏洞
漏洞来源： http://www.wooyun.org
Tags标签： sns类型应用xss url跳转 OAuth session+fixation+attack

OAuth1.0a 版本中已经修复了此漏洞。首先为防止攻击者篡改回调地址，oauth_callback（即回调地址）字段在请求未授权的 request_token 的时候即传递给平台方并参与签名计算。平台方获得用户授权后重定向用户到第三方应用时会返回一个随机值，用在第三方申请 access_token 的过程当中。由于随机值无法被猜测，因此系统能够有效防止攻击者劫持授权过程。

3 OAuth2.0 及安全问题

OAuth1.0 中只提供了一种授权流程，OAuth2.0 进行了扩充，它提供了多种授权流程，这里详细介绍其中 3 种比较流行的授权流程：Authorization Code 授权流程适用于有 Server 端配合的应用；Implicit Grant 授权流程适用于无 Server 端配合的应用；Resource Owner Password Credentials 授权流程直接使用用户名和密码进行授权。此外还有一种使用 refresh_token 获取 access_token 的方式。

OAuth2.0 与 1.0 版本的区别在于以下几点[6]：

（1）request_token 在 2.0 版本中不再使用。

（2）取消所有签名计算，整个授权流程使用 HTTPS 确保安全性。

（3）授权流程大大简化，安全性有所提高。

（4）2.0 版本与 1.0 版本不兼容，是一个全新的协议。这也是很多老的厂商不进行升级的原因。

下面先来看一下 Resource Owner Password Credentials 授权流程（如图 1 所示）。

图1 Resource Owner Password Credentials授权流程

图 1 中浏览器发出的 url 有几个关键字段：client_id 和 client_secret 都是第三方在开放平台注册初始时必须有的值，id 是公开的，secret 是保密的。 grant_type 在当前授权模式下的固定值是 password。 username 和 password 是用户在资源提供方注册的用户名和密码。这种授权方式需要开放平台对第三方有着充分的信任。恶意应用如果使用该方式授权，会导致暴力破解资源提供方的用户账号和密码。

Implicit Grant[4] 是客户端的授权流程，无需服务端配合。这样会导致 access_token 被暴露在客户端，是 OAuth2.0 中公认的最不安全的授权流程（如图 2 所示）。

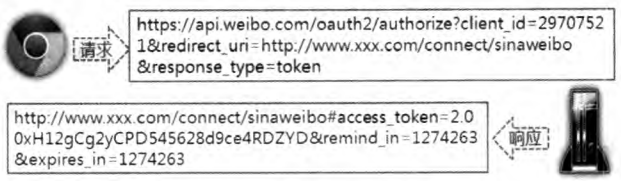

图2 Implicit Grant授权流程

第三方平台请求图 2 中第一个框中的 url，其中 redirect_uri 是一个回调地址，当用户同意授权后就跳转到该地址。response_type 在当前模式下的固定值是 token。开放平台响应这个 url，它直接将 access_token 挂在回调地址后面作为 url 的 fragment(url 中 # 号以后）部分，被用户直接在浏览器中看见。除了 access_token 外，开放平台还会返回 expires_in（过期时间）、refresh_token（用于刷新 access_token）、scope（请求的权限）。

下面给出了一个人人网上存在的此类安全漏洞：

缺陷编号： WooYun-2012-05804
漏洞标题： 人人网Oauth 2.0授权可导致用户access_token泄露
相关厂商： 人人网
漏洞作者： PiaCa
提交时间： 2012-04-05
公开时间： 2012-05-20
漏洞类型： 敏感信息泄露
危害等级： 高
自评Rank： 10
漏洞状态： 厂商已经确认
漏洞来源： http://www.wooyun.org
Tags标签： 无

这个漏洞产生的原因是 client_id 与 redirect_uri 没有做过校验或者信任域过大，配合信任域下的 XSS（跨站脚本攻击）就可以劫持用户的 access_token。例如，访问 http://graph.renren.com/oauth/grant?client_id=cd271e3051444285b8a18f1211a095cd&redirect_uri=http://zone.ku6.com/u/17958620&response_type=token，最后跳转到存在 XSS 的酷 6 网站页面 http://zone.ku6.com/u/17958620，攻击者可以在该页面插入一段恶意的 Javascript 代码来解析存在于 url 的 fragment(url 中 # 号以后）部分中的 access_token。

下面是隐式授权流程下的劫持攻击流程图（如图 3 所示）：

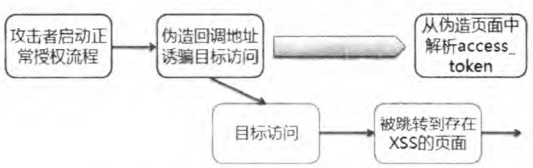

图3 隐式授权下的劫持攻击流程图

除了上文所说的安全问题，还存在一个越权 API 访问的安全问题。授权服务器通常会给予第三方应用一些访问基础功能 API 的权限。如果用高权限 API，需要指定 scope（请求

的权限）进行申请。若 access_token 没有与应用的 appkey（开放平台注册时所分配的唯一值）绑定，即任何令牌均可以去申请任何权限，则可以越权访问受限制的 API。通过在 url 后面添加 scope 参数，后面跟上想去越权访问的一个高权限的 API 名字来实现。最后在回调地址的 fragment 部分获得 access_token，此时该令牌已经被授权使用一些高级权限的接口。

Authorization Code 授权流程需要 Server 端配合，属于两步授权。首先第三方应用跳转到授权应用页面请求 code，用户同意授权后，授权服务器返回 code。第三方应用使用 code 请求 access_token，授权服务器以响应体的形式返回 access_token。图 4 给出的是第三方应用两次请求的 url：

```
https://api.weibo.com/oauth2/authorize?client
_id=1111111&redirect_uri=http://www.xxx.co
m/connect/sinaweibo?&response_type=code
```

```
https://api.weibo.com/oauth2/access_token?client_
id=1111111&client_secret=l29dgp03m2&grant_type
=authorization_code&redirect_uri=http://www.xxx
.com/connect/sinaweibo?&code=1sk42lf0s
```

图4 第三方应用两次请求的url

第一次请求与之前的授权方式比较，仅 response_type 有了改变，现在固定值是 code。第二次请求使用从授权服务器获得的 code 请求 access_token。从图 4 中可以看到之前获得的 code 是 1sk42lf0s。

同样这种授权流程也存在安全问题。下面结合测试案例来展示这个 CSRF（跨站请求伪造）攻击[5]。

测试场景：攻击者通过新浪微博劫持目标的 360 主站以及 360 浏览器账号。

目标首先处于 360 平台登录状态，未绑定任何第三方账号。

攻击者通过新浪微博登录 360，点击授权后阻止页面自动跳转。用要跳转到的 url 诱骗目标访问，目标访问 url。此时在用户不知情的情况下将目标的 360 账户绑定到攻击者的新浪微博上，之后我们可以使用新浪微博账号登录目标的 360 浏览器，此时可以查看其历史记录、收藏夹等隐私数据。

假设我们已经通过 CSRF 劫持了用户的某个账号，且此时目标账号所在网站不仅使用 OAuth 协议，而且还提供 OAuth 授权登录机制。则我们不仅能够通过别的网站账号来登录我们的网站，还可以通过我们这个网站账号来登录别的网站。如果用户使用这个被劫持的账号登录过别的网站，攻击者则可以通过这个已被劫持的账号继续劫持该用户的其他网站账号，这样就形成了一个 CSRF 的递归劫持。有时网站会通过设置强制登录字段强制用户输入第三方的账号和密码信息，我们可以通过修改该字段为 false 直接读取当前会话来登录，即可以继续劫持该网站的账号。

除此之外，Authorization Code 授权流程还有另外一种安全问题：replay attack（授权码重放攻击）。一般情况下，Authorization Code 会随着 access_token 的过期而过期，规范中建议过期时间为 60~80min，且要保证授权码仅可以用一次。很多资源提供方为了降低授权成本，没有严格按照规范实施，有的过期时间甚至达到一周，从而可以通过这个 Authorization Code 授权流程去重复获得 access_token。我们可以通过以下几种方式获取目标授权码[3]：在获得服务器权限的情况下，通过日志获取大量的目标授权码；通过嗅探获得目标授权码，因为回调地址位于客户端，不会强制使用 HTTPS，这样就给了攻击者可乘之机；通过 XSS 获得目标授权码，配合目标网站信任域下的 XSS 可以实现劫持 code。

4 OAuth 漏洞防御及安全性提升措施

对于资源提供方来说，首先要对 client_id 和回调地址做严格校验，确保获取 access_token 的 code 仅能使用一次以避免受到重放攻击，尽量避免直接读取当前用户的 session 进行绑定。对于资源使用方来说，应该使用 Authorization Code 方式进行授权，授权过程中使用 state 随机哈希参数，并在服务端进行判断以防止 CSRF 攻击，尽量使用 HTTPS 保证整个授权过程的安全性。通过以上防御措施，可以提高整个授权流程的安全性。

5 结束语

由于 OAuth 协议在日常上网过程中时常存在，而且 OAuth 协议关系到用户的账号、密码等隐私信息，因此它的安全问题也受到了极大的重视。本文通过介绍 OAuth1.0 和 2.0 版本中存在的安全问题以及相应的防御方式，并且结合部分测试案例让大家对 OAuth 协议的安全问题有一个直观的了解，并希望更多的安全研究人员能够关注 OAuth 协议。（责编 马珂）

参考文献：

[1]Wikipedia-OAuth[Z/OL]. http://zh.wikipedia.org/wiki/OAuth ,2013.

[2]Eran Hammer. Explaining the OAuth Session Fixation Attack[Z/OL]. http://hueniverse.com/2009/04/explaining-the-oauth-session-fixation-attack/, 2009.

[3]Egor Homakov. The Story About Two OAuth2 Vulnerabilities[Z/OL]. http://homakov.blogspot.com/2012/09/a-couple-of-reasons-why-oauth2-spec-is.html, 2012.

[4]Egor Homakov. OAuth2: One access_token To Rule Them All[Z/OL].http://homakov.blogspot.com/2012/08/oauth2-one-accesstoken-to-rule-them-all.html, 2012.

[5]Egor Homakov. The Most Common OAuth2 Vulnerability[Z/OL]. http://homakov.blogspot.com/2012/07/saferweb-most-common-oauth2.html, 2012.

[6]Dndx. OpenID 和 OAuth 的区别及第三方登录的安全隐患分析[Z/OL]. https://idndx.com/2012/04/23/openid-vs-oauth-and-the-security-risk-of-oauth-login/, 2012.