OAuth2.0 协议

一、定义

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的资源（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。

二、产生原因

例子：有一个“云冲印”的网站，可以将用户 A 存储在 google 的照片，冲印出来。A 为了使用这个服务，必须让“云冲印”读取自己存储在 google 的照片。

？？“云冲印”如何获得用户的授权？？、

答案：给“云冲印”用户的用户名密码

一系列问题：

（1）“云冲印”缓存用户名密码，可能没有加密保护，一旦受到攻击，用户躺枪。

（2）“云冲印”没有权限限制，访问用户在 google 所有照片。

（3）用户无法撤销授权，除非修改密码。

So，OAuth2.0 运营而生。

三、基础概念

1．Client ：第三方应用，例如“云冲印”、

2．RO（resource owner）：资源所有者，例如“A”

3．RS（resource server）：资源服务器，存储资源。例如“google 照片”

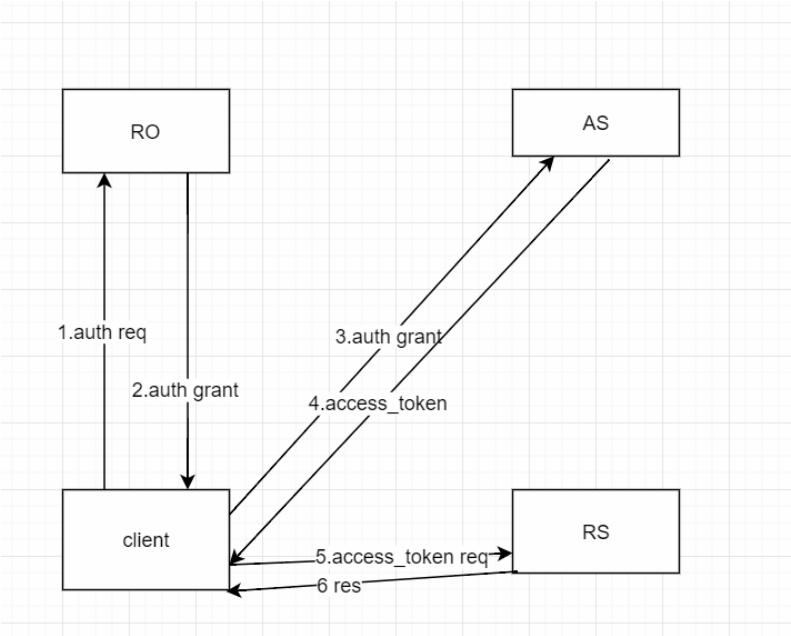
4．AS（authorization server）：授权服务器，认证 RO 身份，为 RO 提供授权审批流程，并最终颁发授权令牌（Access_token）

四、Oauth 思路

OAuth 在"客户端"与"服务提供商"之间，设置了一个授权层（authorization layer）。"客户端"不能直接登录"服务提供商"，只能登录授权层，以此将用户与客户端区分开来。"客户端"登录授权层所用的令牌（token），与用户的密码不同。用户可以在登录的时候，指定授权层令牌的权限范围和有效期。

"客户端"登录授权层以后，"服务提供商"根据令牌的权限范围和有效期，向"客户端"开放用户储存的资料。

五、运行流程

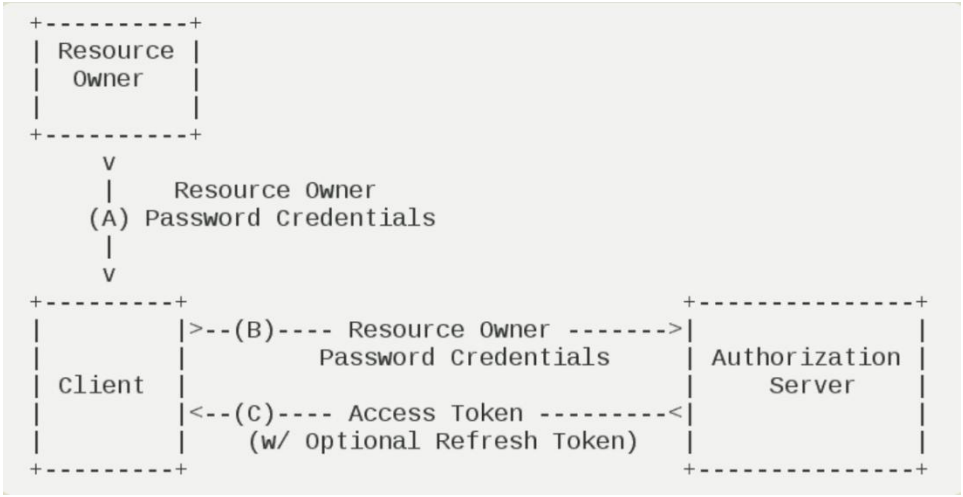


关键点：Ro 如何授权给 client

三种方式：授权码模式、简化模式、密码模式。

六、模式介绍

密码模式-----高度信任



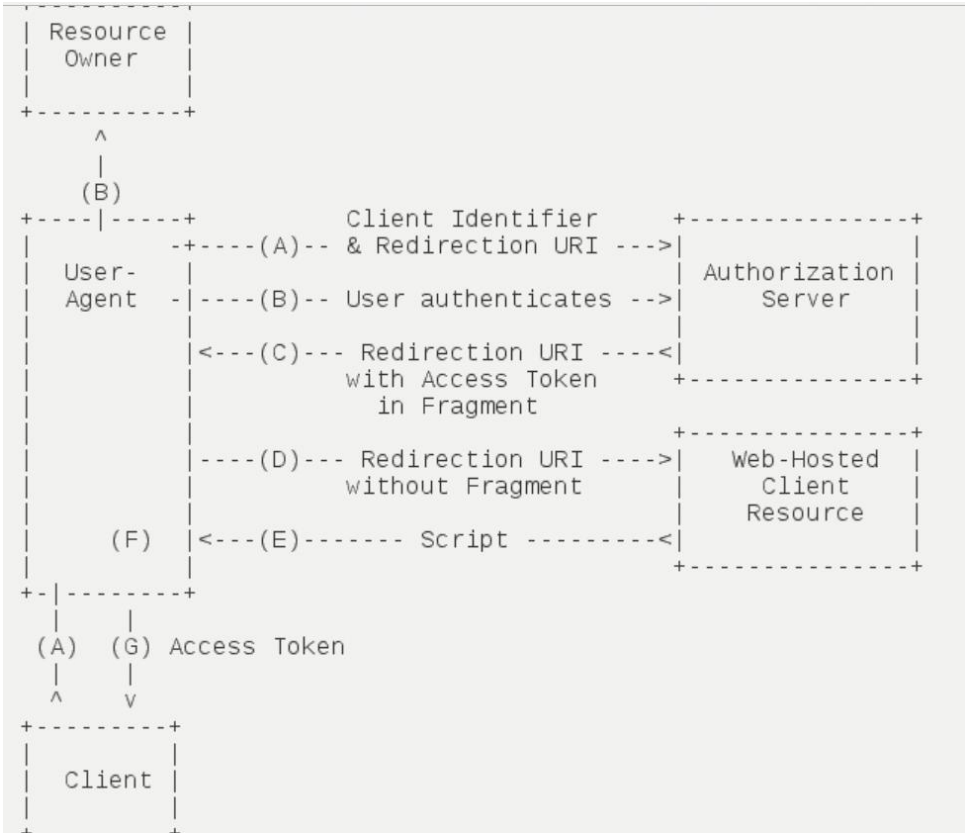
步骤介绍：

（1） 用户给客户的提供用户名密码

（2） Client 拿用户名密码去 As 验证，拿 token

（3） As 返回 token。

简化模式：-------web 站点



步骤描述：

（1） （A）client 向用户代理 request，302 重定向到 AS，请求参数 client_id 、scope、redirect_uri、state 、access_type、approval_prompt。

例如：

GET/authorize?response_type=code&client_id=s6BhdRkqt3&state=xyz&redirect_uri=https%3A%2F%2Fclient%2Eexample%2Ecom%2Fcb HTTP/1.1

[注].state csrf 攻击

（2） （B）approval_prompt = true，则 AS 通过交互界面得到用户的批准，false 默认可以。

（3） （C）返回 access_token

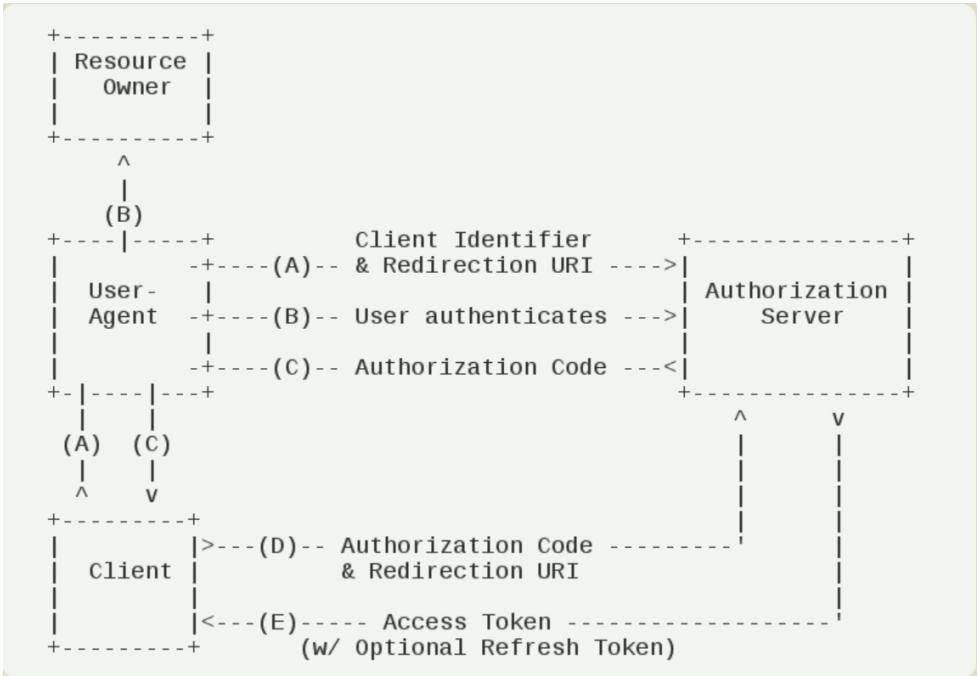
例如：

HTTP/1.1 302 Found
Location:http://example.com/cb#access_token=2YotnFZFEjr1zCsicMWpAA&state=xyz&token_type=example&expires_in=3600

（4） （D），访问 Location 的 uri，

（5） （E），返回的代码提取 Location 参数 access_token，获得资源。

授权码模式：------第三方应用（相比于上一种，多了 Authorization_code 验证）



步骤描述：

（6） （A）client 向用户代理 request，302 重定向到 AS，请求参数 client_id 、scope、redirect_uri、state 、access_type、approval_prompt。当 access_type=offline 时，AS 将在颁发

access_token 时，同时还会颁发一个 refresh_token。因为 access_token 的有效期较短（如 3600 秒），为了优化协议执行流程，offline 方式将允许 Client 直接持 refresh_token 来换取一个新的 access_token

例如：

GET/authorize?response_type=code&client_id=s6BhdRkqt3&state=xyz&redirect_uri=https%3A%2F%2Fclient%2Eexample%2Ecom%2Fcb HTTP/1.1

[注].state csrf 攻击

（2）approval_prompt = true，则 AS 通过交互界面得到用户的批准，false 默认可以。

（3）（C）用户同意授权，则重定向返回 client authorization_code 和 state 例如：

HTTP/1.1 302 Found

Location:https://client.example.com/cb?code=SplxlOBeZQQYbYS6WxSbIA&state=xyz

（4）（D）向 AS 获取 access_code.

例如：grant_type=authorization_code&code=SplxlOBeZQQYbYS6WxSbIA&redirect_uri=https%3A%2F%2Fclient%2Eexample%2Ecom%2Fcb

(5)(E)返回 access_code 以及 refresh_token(若 access_type=offline)。

例如：

HTTP/1.1 200 OK

Content-Type: application/json;charset=UTF-8

Cache-Control: no-store

Pragma: no-cache

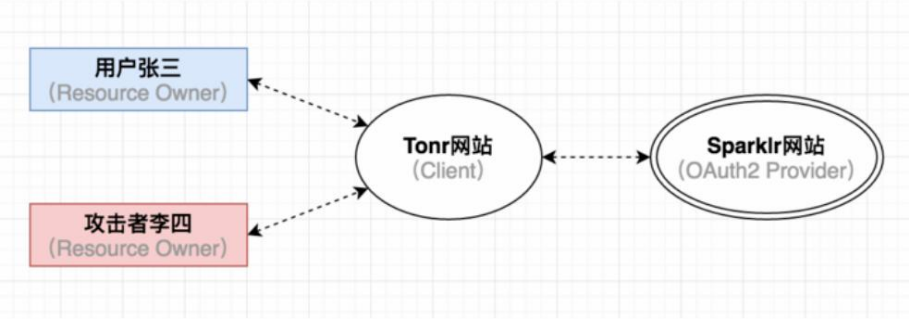
```
{
  "access_token":"2YotnFZFEjr1zCsicMWpAA",
  "token_type":"example",
  "expires_in":3600,
  "refresh_token":"tGzv3JOkF0XG5Qx2TlKWIA",
  "example_parameter":"example_value"
}
```

七、 授权码模式详细介绍

（1） OAuth2.0 存在的安全漏洞

a、csrf 攻击，前提 get 请求没有 state 字段

发生场景：



Step1.攻击者李四登录 Tonr 网站，选择绑定 Sparklr 账号

Step2.Tonr 重定向到 Sparklr（没有 state 字段），然后返回 Authorization_code.

Step3.攻击者李四制造了一个恶意 web 网页，网页会向 Sparklr 发送带有李四的 Authorization_code 的请求 access_token 的页面

Step4.用户张三登录 Tonr，但是没有任何账号绑定，然后张三访问李四的 web 页面，成功的从 Sparklr 获得 access_token.

Step5.攻击者李四可以成功的操作张三的 Tonr 账户信息。

本质：请求分步执行。

避免方式：添加 state。

b、redirect_uri 绕过导致授权劫持

本质：redirect_uri 校验失败。

例如:网易账号登录 56 网。

https://api.t.163.com/oauth2/authorize 烦 忙

client_id=II5coZy8DdAtKt7a&redirect_uri=http%3A%2F%2Fapp.56.com%2Fcooperate%2Findex.php%3Faction%3DWeibo%26tag%3Dwy%26do%3DCheckLogin%26from%3Dregbox&response_type=code&state=unk-qogvtqoomz

替换为：redirect_uri=http%3A%2F%2Fapp.56.com%40wooyun.org

解析匹配 redirect-uri：用 ./解析。

漏洞修补：正则表达式，rfc，很复杂。

（2） 为什么要 Authorization_code

Redirect_uri 不安全，通过 HTTP referrer 伪造后，造成 access_token 被盗用。