# OAuth 2.0中文译本

（一）背景知识

OAuth 2.0很可能是下一代的“用户验证和授权”标准，目前在国内还没有很靠谱的技术资料。为了弘扬“开放精神”，让业内的人更容易理解“开放平台”相关技术，进而长远地促进国内开放平台领域的发展，笔者特意将OAuth 2.0协议翻译成中文。

目前OAuth 2.0还没有最后定稿，最新的修改版是第11个版本，本文下面的翻译即基于这个第11版本。原文见[http://tools.ietf.org/html/draft-ietf-oauth-v2-11](http://tools.ietf.org/html/draft-ietf-oauth-v2-11" \t "_blank)。

关于OAuth 2.0的更多背景知识，请参考我的另一篇文章：[http://itgeeker.com/mathml/readpaper?pid=65](http://itgeeker.com/mathml/readpaper?pid=65" \t "_blank)

（二）术语中英对照表

由于OAuth协议版本较多（1.0,1.0a,2.0等），并且各个版本中的技术术语也各不相同，关于英文技术术语与中文的对应关系，我们以OAuth 2.0的第11版本中的描述为准。

另外有一些情况，一些英文术语不容易找到普遍接受的汉语释义，翻译过来反而可能引起误解，而英文术语本身可能更容易理解，因此就不考虑对这部分词汇做翻译了。比如，“web service”、“endpoint”、“user-agent”、“URI”、“cookie”等，你只需要知道它是什么就好了。

还有一些特别难于翻译的词汇，比如“profile”，这个词用在协议里，大概表示：协议功能的某个剖面、子集、子视图或轮廓。如果翻译成“视图”，容易让人想到“view”这个词，产生冲突，最后，我在这里创造了一个新词汇：“子态”。

下面是整理出来的重要技术术语的中英对照表：

云计算 —— cloud computing

第三方 —— third-party

应用/程序 —— application

私有证书 —— credential

身份验证 —— authentication

授权 —— authorization

明文 —— clear-text

客户端 —— client {译者注：本文中的客户端与平常所说的“客户端”并不相同，是相对资源服务器和授权服务器来说的，它可能指第三方应用的服务器程序或客户端程序}

服务器 —— server

资源拥有者 —— resource owner

受保护资源 —— protected resource

资源服务器 —— resource server

访问令牌 —— access token

授权服务器 —— authorization server

访问许可 —— access grant

实体 —— entity

签名 —— signature

刷新令牌 —— refresh token

作用域 —— scope

授权码 —— authorization code

标识符 —— identifier

密钥 —— secret

断言 —— assertion

原生程序 —— native application

子态 —— profile

同源策略 —— same-origin policy

回调 —— callback

自治的 —— autonomous

查询参数部分 —— query component

分段参数部分 —— fragment component

媒体类型 —— media type

厂商特性的 —— vendor-specific

增强型巴科斯范式 —— ABNF

互联网编号分配机构 —— IANA

互联网工程指导组 —— IESG

标准轨道 —— standards-track

（三）中文译本

1. 引言

随着分布式web service和云计算的使用越来越多，第三方应用需要能访问到一些服务器托管资源。这些资源通常都是受保护的，并且要求使用资源拥有者的私有证书（典型的证书是用户名和密码）进行身份验证。

在传统的基于客户端-服务器的身份验证模型中，客户端为了访问服务器的受保护资源，是使用资源拥有者的私有证书来做身份验证的。为了让第三方应用能够访问受保护资源，资源拥有者必需将他/她/它的私有证书透露给第三方。这引出了很多问题并存在很多局限性：

* 第三方应用需要用明文保存资源拥有者的私有证书（一般是密码），留作以后再次使用。
* 虽然密码验证会造成安全隐患，服务器仍然需要支持用密码做身份验证（对称的密码验证）。
* 第三方应用对资源拥有者的受保护资源获得过多的使用权限，而资源拥有者没有能力限制访问到某个资源子集，限制持续时间，或限制这些资源所能支持的访问方式。
* 资源拥有者无法在不影响所有第三方的前提下单独撤销某个第三方的访问权限，他/她/它只能通过修改密码来回收所有权限。

OAuth通过将客户端和资源拥有者的角色进行分离来解决这些问题。在OAuth中，客户端（通常不是资源拥有者，而是代表资源拥有者来操作）提出请求来访问由资源拥有者控制并由资源服务器托管的资源，然后得到与资源拥有者不同的一套私有证书。

客户端并不是直接使用资源拥有者的私有证书来访问受保护资源，而是得到一个访问令牌——一个代表某一特定作用域、持续时间和其它属性的字符串{译者注：非常重要的一个概念，英文叫access token}。访问令牌由授权服务器在资源拥有者的授意下分发给第三方客户端。客户端使用访问令牌来访问由资源服务器托管的受保护资源。

例如，一个web用户（资源拥有者）能够准许一个打印服务（客户端）访问她存储在另一个照片共享服务（资源服务器）中的照片，而不用将她的用户名和密码透露给这个打印服务。她在一个被该照片分享服务信任的身份验证服务（授权服务器）上完成验证，而这个验证服务会将特定于委托服务的私有证书（令牌）分发给原打印服务。

基于资源服务器对安全的需求，访问令牌可以有不同的格式、结构和使用方式（例如密码学特性）。访问令牌的属性和用以访问受保护资源的方式不在本规范的规定范围之内，而是由相关的其它规范来定义。授权服务器和资源服务器之间的交互方式不在本规范的规定范围之内。

1.1. 符号规范

这篇文档中的关键词“必须”、“一定不能”、“要求”、“会”、“不会”、“应该”、“不应该”、“建议”、“可以”、“可选的”，遵从[RFC2119]中的解释。

这篇文档使用出自[I-D.ietf-httpbis-p1-messaging]的增强型巴科斯范式（ABNF）标记法。另外，介绍一些规则定义的出处：URI-Reference出自[RFC3986]；OWS、RWS和quoted-string出自[I-D.ietf-httpbis-p1-messaging]。

除非特别提到，否则所有协议参数的名字和值都是大小写敏感的。

1.2. 专业术语解释

受保护资源：能够使用OAuth请求获取的访问限制性资源。

资源服务器：能够接受和响应受保护资源请求的服务器。

客户端：获取授权和发送受保护资源请求的应用。

资源拥有者：能够对受保护资源进行访问许可控制的实体。

终端用户：起到资源拥有者角色的用户。

令牌：分发给客户端的代表访问授权的字符串。通常这个字符串对客户端来说是不透明的。令牌代表资源拥有者许可的访问作用域和持续时间，并由资源服务器和授权服务器强制保证。这个令牌可以代表一个标识符，用于检索授权信息，或以一种可验证的方式自包含授权信息（即一个包含数据和签名的令牌字符串）。令牌可能只代表纯粹的访问能力。而为了让客户端使用令牌，也可能需要一些多余的特定验证证书。

访问令牌：被客户端用来代表资源拥有者发送验证请求的令牌。

刷新令牌：被客户端用来获取新的访问令牌的令牌，而不用资源拥有者的参与。

授权码：一个短期令牌，代表终端用户的授权。授权码用于获取一个访问令牌和一个刷新令牌。

访问许可：用于描述中间形式的私有证书（如终端用户的密码或授权码）的一个通用词汇，代表资源拥有者的授权。客户端使用访问许可来获取访问令牌。通过将各种形式的访问许可都交换成访问令牌，资源服务器只需要支持一种验证机制。

授权服务器：能够成功验证资源拥有者和获取授权，并在此之后分发令牌的服务器。授权服务器可以和资源服务器是同一个服务器，也可以是不同的实体。单独一个授权服务器可以为多个资源服务器分发令牌。

终端用户授权endpoint：授权服务器上能够验证终端用户并获取授权的HTTP endpoint。终端用户授权endpoint在第4节详细描述。

令牌endpoint：授权服务器上能够分发令牌和刷新过期令牌的HTTP endpoint。令牌endpoint在第5节详细描述。

客户端标识符：分发给客户端的唯一标识，用于客户端向授权服务器标识自己。客户端标识符可以有一个对应的密钥。客户端标识符在第3节详细描述。

1.3. 概述

OAuth为客户端提供了一种代表资源拥有者访问受保护资源的方法。在客户端访问受保护资源之前，它必须先从资源拥有者获取授权（访问许可），然后用访问许可交换访问令牌（代表许可的作用域、持续时间和其它属性）。客户端通过向资源服务器出示访问令牌来访问受保护资源。

访问令牌提供了一个抽象层，将不同的授权结构（如用户名密码、断言）替换成资源服务器可以理解的单一令牌。这种抽象使得分发短期有效的访问令牌成为可能，也使得资源服务器不必理解多种多样的授权机制。

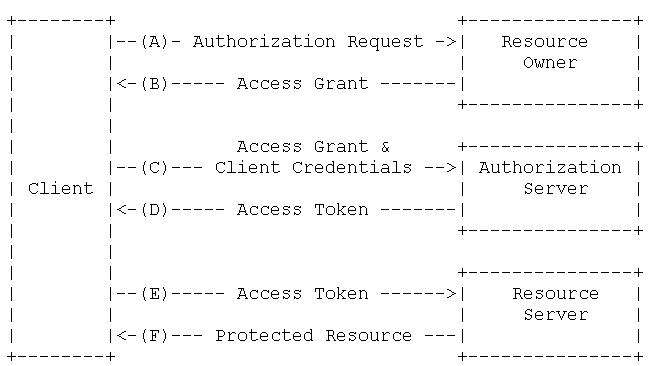


图1: 抽象的协议流程

图1所示的抽象流程协议的总体架构，它包含下列步骤：

(A) 客户端从资源拥有者那里请求授权。授权请求能够直接发送给资源拥有者，或者间接地通过授权服务器这样的中介，而后者更为可取。

(B) 客户端收到一个访问许可，它代表由资源服务器提供的授权。

(C) 客户端使用它自己的私有证书到授权服务器上验证，并出示访问许可，来请求一个访问令牌。

(D) 授权服务器验证客户端私有证书和访问许可的有效性，如果验证通过则分发一个访问令牌。

(E) 客户端通过出示访问令牌向资源服务器请求受保护资源。

(F) 资源服务器验证访问令牌的有效性，如果验证通过则响应这个资源请求。

1.4 访问许可

访问许可代表资源拥有者提供的授权。访问许可的类型取决于客户端使用的获取方式和授权服务器所支持的方式。

1.4.1 授权码

授权码是通过将终端用户引导到授权服务器而获得的一种访问许可。授权服务器验证终端用户，获得授权，然后向客户端分发一个授权码。因为终端用户只在授权服务器上进行验证，所以终端用户的密码从来不用分享给客户端。

当客户端通过一个user-agent同终端用户进行交互的时候，授权码这种访问许可是很合适的。

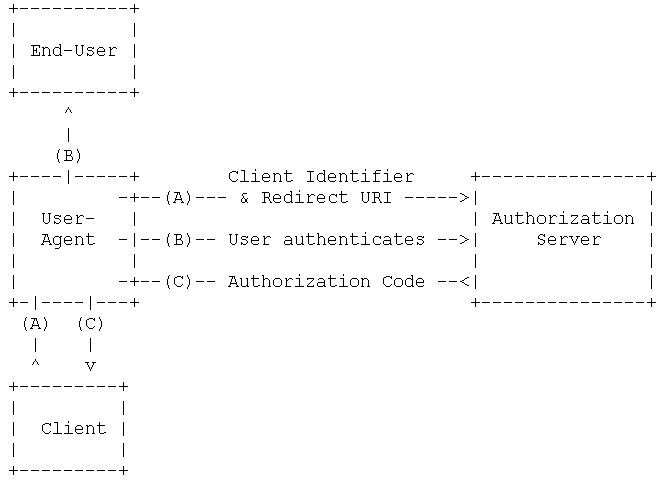


图2: 获取授权码

图2所示的授权码获取流程包含下列步骤：

(A) 客户端通过将终端用户的user-agent引导到授权服务器的终端用户授权endpoint来发起这个流程。客户端传入标识符、请求作用域、本地状态，和一个重定向URI（在访问被许可或被拒绝后授权服务器会重新将user-agent引导回这个URI）。

(B) 授权服务器验证终端用户的身份（通过user-agent），并且确定终端用户是许可还是拒绝了客户端的访问请求。

(C) 如果访问被许可，授权服务器会使用重定向URI将user-agent引导回客户端。授权服务器传回一个授权码给客户端，用于进一步获取访问令牌。

一旦客户端获得了授权码，它会到授权服务器上去做验证（使用客户端私有证书）并出示授权码（访问许可），以借此请求一个访问令牌。

在客户端无法维护它自己的私有证书的情况下（如原生程序或用某种user-agent脚本实现的程序），授权服务器在(C)步直接给客户端分发一个访问令牌，而不再分发一个授权码。

获得授权码的过程在第4节详述。

1.4.2 资源拥有者密码证书

资源拥有者密码证书（例如用户名和密码）可以直接用作访问许可来获取访问令牌。这种私有证书只应该在以下两种情况下使用：当在资源拥有者和客户端之间有很强的信任关系的时候（例如，资源拥有者的计算机操作系统，或具有很高特权的程序），以及当其它访问许可类型（如授权码）不可用的时候。

即使这种许可类型需要客户端直接访问资源拥有者的私有证书，资源拥有者的私有证书也只是在一个请求中使用，并交换成访问令牌。与[RFC2617]定义的HTTP Basic验证机制不同，这种许可类型不再需要客户端存储资源拥有者的私有证书以备日后使用。

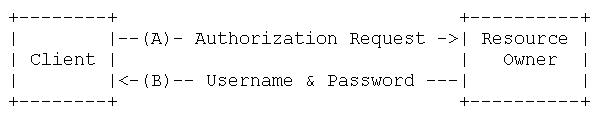


图3: 获取资源拥有者密码证书

在图3中，客户端直接从资源拥有者请求授权。当资源拥有者是一个终端用户时，客户端通常的做法是提示终端用户输入用户名和密码。

1.4.3 客户端私有证书

当授权作用域限制在客户端所控制的受保护资源或之前与授权服务器约定好的受保护资源时，客户端本身的私有证书可被用作访问许可。客户端私有证书用作访问许可的典型例子是，当客户端代表它自己执行操作时（客户端同时也是资源拥有者）。

1.4.4 刷新令牌

访问令牌的生命周期通常比资源拥有者授予的要短一些。当分发一个访问令牌时，授权服务器可以同时传回一个刷新令牌，在当前访问令牌超时后，客户端可以用这个刷新令牌重新获取一个访问令牌。当请求新的访问令牌时，刷新令牌担当起访问许可的角色。使用刷新令牌，不再需要再次与资源拥有者交互，也不需要存储原始的访问许可来获得访问令牌和刷新令牌。

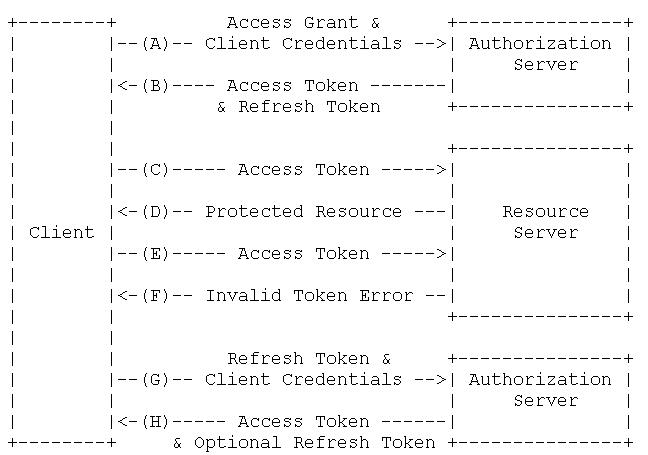


图4: 刷新访问令牌

图4所示的刷新令牌流程包含下列步骤：

(A) 客户端通过使用它自己的私有证书在授权服务器上验证，并出示一个访问许可。

(B) 授权服务器验证客户端私有证书和访问许可的有效性，如果通过则分发一个访问令牌和刷新令牌。

(C) 客户端通过出示访问令牌向资源服务器请求受保护资源。

(D) 资源服务器验证访问令牌的有效性，如果通过，则相应这个请求。

(E) 步骤(C)(D)不停重复，直到访问令牌过期。如果客户端不知道访问令牌过期，它会再请求一次受保护资源。否则，跳到步骤(G)。

(F) 因为访问令牌是无效的（过期了），资源服务器返回一个无效令牌错误。

(G) 客户端通过使用它的私有证书在授权服务器上验证并出示刷新令牌（用作访问许可），来请求一个新的访问令牌。

(H) 授权服务器验证客户端私有证书的有效性，如果通过则分发一个新的访问令牌（也可能还有一个刷新令牌）。

1.4.5 断言

断言在OAuth和其它信任框架之间架起一座桥梁。它们允许客户端利用现成的信任关系来获取访问令牌。一个断言所代表的访问许可取决于断言类型、断言内容，以及断言被分发的方式，而这些内容不在本规范的规定范围之内。

断言可以用在协议扩展模型的部分，它为授权服务器提供了一种支持其它访问许可类型的方式。

2. 客户端的各种子态

2.1 Web Server子态

Web Server子态适用于有能力与终端用户的user-agent（通常是浏览器）交互并能够从授权服务器接收（通过重定向）请求（即有能力担当HTTP服务器的角色）的客户端。

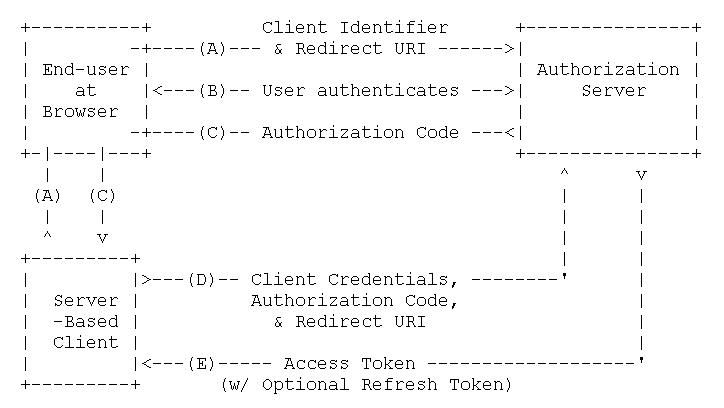


图5: Web Server流程

图5所示的web server流程包含下列步骤：

(A) 如第4节所述，web客户端通过将终端用户的user-agent重定向到授权服务器来发起这个流程。客户端传入它的客户端标识符、请求作用域、本地状态和一个重定向URI，在访问被许可（或被拒绝）后授权服务器会重新将终端用户引导回这个URI。

(B) 授权服务器验证终端用户（借助于user-agent），并确定终端用户是否许可客户端的访问请求。

(C) 假定终端用户许可了这次访问，授权服务器会将user-agent重定向到之前提供的重定向URI上去。授权服务器为客户端传回一个授权码做获取访问令牌之用。

(D) 如第5节所述，客户端通过验证并传入上一步取得的授权码从授权服务器请求一个访问令牌。

(E) 授权服务器验证客户端私有证书和授权码的有效性并返回访问令牌。

2.2 User-Agent子态

User-Agent子态适用于常驻user-agent的客户端应用，典型的例子是用诸如JavaScript语言编写并运行在浏览器的程序。这些客户端不能保存客户端私有证书，并且客户端的验证基于user-agent的同源策略。

在其它子态中，客户端对于终端用户的授权和访问令牌使用分开的不同请求来完成，而与之不同的是，在user-agent子态中，客户端以HTTP重定向的方式在终端用户授权请求的结果中获取到访问令牌。客户端请求授权服务器将user-agent重定向到另一个web服务器或user-agent能访问到的本地资源，而且user-agent有能力从响应信息中提取出访问令牌并传给客户端。

这种user-agent子态并不使用客户端密钥，因为客户端执行程序常驻于终端用户的计算机或设备上，这使得客户端密钥可以被访问或收集到。因为访问令牌被编码到重定向URI中，所以它可能会暴露给终端用户和常驻计算机或设备上的其它应用。

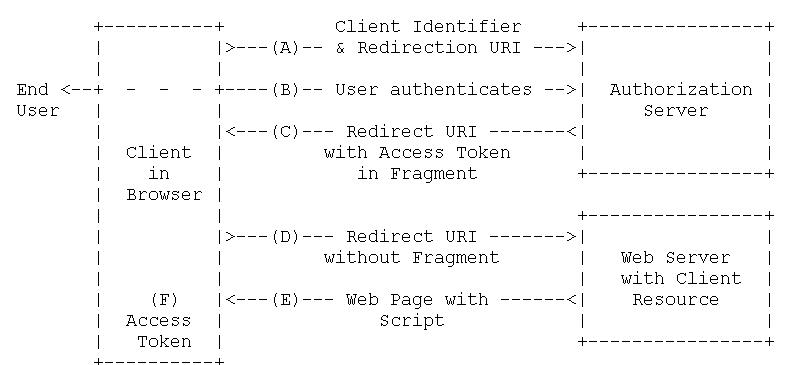


图6: User-Agent流程

图6所示的user-agent流程包含下列步骤：

(A) 如第5节所述，客户端将user-agent引导到终端用户授权endpoint。客户端传入它的客户端标识符、请求作用域、本地状态和一个重定向URI，在访问被许可（或被拒绝）后授权服务器会重新将终端用户引导回这个URI。

(B) 授权服务器验证终端用户（通过user-agent）并确认终端用户是许可还是拒绝了客户端的访问请求。

(C) 如果终端用户许可了这次访问，那么授权服务器会将user-agent引导到之前提供的重定向URI。重定向URI会在URI片断{译者注：URI片断是指URI中#号之后的内容}中包含访问令牌。

(D) user-agent响应重定向指令，向web服务器发送不包含URI片断的请求。user-agent在本地保存URI片断。

(E) web服务器返回一个web页面（通常是嵌入了脚本的HTML网页），这个页面能够访问完整的重定向URI，它包含了由user-agent保存的URI片断，同时这个页面能够将包含在URI片断中的访问令牌（和其它参数）提取出来。

(F) user-agent在本地执行由web服务器提供的脚本，该脚本提取出访问令牌并将它传递给客户端。

2.3 原生程序

原生程序是作为原生代码运行在终端用户计算机或设备上的客户端（即，在user-agent之外运行或作为一个桌面程序）。这些客户端通常有能力与终端用户的user-agent交互（或嵌入user-agent），但是在这些交互如何影响终端用户体验的方式上受到限制。在很多情况下，原生程序无法直接从服务器接收回调请求（例如，防火墙、操作系统限制）。

基于不同的需求和期望的终端用户体验，原生程序客户端可以用不同的方式实现。原生程序客户端可以：

* 如第4节所述，通过启动一个外部user-agent来访问终端用户授权endpoint。客户端可以通过下面的方式捕获响应文本：提供一个具有自定义URI scheme{译者注：URI scheme就是一个URI里面的第一部分，即冒号前面的部分}的重定向URI（在操作系统上注册过以便调用客户端应用），或者提供一个指向在客户端控制下的服务器资源的重定向URI，这使得响应文本对客户端可见（例如，使用窗口标题或在user-agent外面可以访问到的其它位置）。
* 如第4节所述，通过嵌入一个user-agent来访问终端用户授权endpoint。客户端通过与嵌入的user-agent直接通信获取到响应文本。
* 提示终端用户输入密码，使用密码直接获得一个访问令牌。通常来讲，这是一种不推荐的方式，因为它将终端用户的密码直接交给了第三方客户端，而客户端不得不用明文存储密码。它还要求服务器支持基于密码的身份验证。

当在启动外部浏览器和嵌入的user-agent之间进行选择时，开发者应该考虑下列因素：

* 外部浏览器可能会提高完成比率，因为终端用户可能已经登录过了而不需要重新进行身份验证。
* 嵌入的user-agent通常能提供更好的用户流程，因为它不必切换上下文并打开新窗口。
* 嵌入的user-agent对安全提出了挑战，因为用户在一个无法辨别的窗口之中进行身份验证，而不像很多user-agent那样能提供可视化的保护。

2.4 自治态

自治客户端使用现成的信任关系或框架来建立授权。基于自治客户端的需求和他们所依赖的现存信任框架，自治客户端可以用不同的方式实现。自治客户端可以：

* 通过使用客户端私有证书与授权服务器进行验证，从而获得访问令牌。访问令牌的作用域局限于受客户端控制的受保护资源，或者其它资源拥有者与授权服务器预先约定的资源。
* 使用现存的某种访问许可，它被表达成授权服务器所支持的某种断言格式。使用断言需要客户端从一个断言发行方获得一个断言（如SAML[OASIS.saml-core-2.0-os]断言）或自己分发一个断言。断言的格式、获得断言的过程，以及验证断言的方法，由断言发行方和授权服务器定义，不在本规范的规定范围之内。

3. 客户端私有证书

当与授权服务器进行交互时，客户端使用一个私有证书集合来标识自己，这个证书集合包含一个客户端标识符和用于客户端身份验证的其它一些属性。客户端获得私有证书的方式不在本规范的规定范围之内，不过这通常都包含一个在授权服务器上注册的过程。

考虑到一些客户端的本质特性，在与客户端没有确立信任关系的前提下，授权服务器不应该对客户端密钥的私密性做出任何假设。授权服务器不应该向没有能力对密钥进行秘密保存的客户端分发密钥。

授权服务器可以使用任一合适的私有证书集合和验证机制来对客户端进行身份验证。客户端一定不能在一个请求中使用多个私有证书集合和验证机制。

3.1 客户端密码证书

客户端密码证书使用一个共享的对称密钥来验证客户端。客户端标识符和密码被包含在请求当中，使用[RFC2617]定义的HTTP Basic验证机制，将客户端标识符作为用户名（username）并将客户端密码作为密码（password）来传送。

例如（换行符只用于显示目的）：

POST /token HTTP/1.1

Host: server.example.com

Authorization: Basic czZCaGRSa3F0MzpnWDFmQmF0M2JW

Content-Type: application/x-www-form-urlencoded

grant\_type=authorization\_code&code=i1WsRn1uB1&

redirect\_uri=https%3A%2F%2Fclient%2Eexample%2Ecom%2Fcb

作为可选方式，客户端可以使用下列参数将密码包含在请求体（request body）中：

client\_id

必需参数。客户端标识符。

client\_secret

必需参数。客户端密钥。

例如（换行符只用于显示目的）：

POST /token HTTP/1.1

Host: server.example.com

Content-Type: application/x-www-form-urlencoded

grant\_type=authorization\_code&client\_id=s6BhdRkqt3&

client\_secret=gX1fBat3bV&code=i1WsRn1uB1&

redirect\_uri=https%3A%2F%2Fclient%2Eexample%2Ecom%2Fcb

授权服务器必须能够使用请求参数和HTTP Basic验证协议两种方式接受客户端私有证书。授权服务器可以支持更多适合于密码证书传输的验证机制。

3.2 客户端断言证书

客户端断言证书用于不宜使用密码（明文共享对称密钥）或密码无法为客户端验证提供足够安全性的情况。在这样的情况下，常见的做法是使用诸如HMAC或数字签名之类不需要发送明文密钥的其它机制。客户端断言证书提供了一种扩展机制，能够使用被授权服务器所支持的某种断言格式进行客户端身份验证。

使用断言需要客户端从一个断言发行方获得一个断言（如SAML[OASIS.saml-core-2.0-os]断言）或自己分发一个断言。断言的格式、获得断言的过程，以及验证断言的方法，由断言发行方和授权服务器定义，不在本规范的规定范围之内。

当使用客户端断言时，客户端传送下列参数：

client\_assertion\_type

必需参数。由授权服务器定义的断言格式。这个值必须是一个绝对URI。

client\_assertion

必需参数。客户端断言。

例如，客户端使用一个SAML 2.0断言发送如下访问令牌请求来验证自己（换行符只用于显示目的）：

POST /token HTTP/1.1

Host: server.example.com

Content-Type: application/x-www-form-urlencoded

grant\_type=authorization\_code&code=i1WsRn1uB1&

client\_assertion=PHNhbWxwOl[...omitted for brevity...]ZT4%3D&

client\_assertion\_type=

urn%3Aoasis%3Anames%sAtc%3ASAML%3A2.0%3Aassertion&

redirect\_uri=https%3A%2F%2Fclient%2Eexample%2Ecom%2Fcb

当使用一个客户端断言和一个授权码获得一个访问令牌的时候，需要一个机制在用于获取授权码的“client\_id”参数值和客户端断言之间完成映射。这个机制不在本规范的规定范围之内，但对于与授权码结合使用的任何客户端断言类型都必须明确指明。

对于那些使用客户端断言证书但不包含能够提供下列信息的HMAC或签名值的访问令牌请求，授权服务器必须拒绝响应：

* 指明这个断言是专门分发给当前接收endpoint来处理的（一般是通过一个包含接收endpoint标识符的audience或recipient值）。
* 标识出分发断言的实体（一般是通过一个issuer值）。
* 用一个绝对时间标识出断言在何时过期（一般是通过一个包含UTC日期/时间值的过期值）。授权服务器必须拒绝过期的断言。

4. 获得终端用户授权

在客户端能够访问一个受保护资源之前，它必须首先从终端用户那里获取授权。为了获得终端用户授权，客户端需要将终端用户引导到终端用户授权endpoint。一旦获得授权，终端用户的访问许可会被表示成一个授权码，客户端能够使用它去获取一个访问令牌。

在终端用户授权endpoint上，终端用户首先在授权服务器上完成身份验证，然后允许或者拒绝当前访问请求。授权服务器验证用户的方式（例如，用户名和密码登录，OpenID，会话cookie）和授权服务器获取终端用户授权的方式，以及是否使用诸如TSL之类的安全通道，不在本规范的规定范围之内。然而，授权服务器必须要首先验证终端用户的身份。

终端用户授权endpoint的位置能够在服务器文档中找到。终端用户授权endpoint的URI可以按照[RFC3986]第3节的定义包含一个查询参数部分，它们在添加其它参数时必须被保留。

既然对于终端用户授权endpoint的请求会导致用户身份验证和敏感信息的传输，授权服务器应该要求在向终端用户授权endpoint发送请求的时候使用诸如TLS之类的传输层安全机制。

4.1 授权请求

为了将终端用户的user-agent引导到授权服务器，客户端将下列参数添加到终端用户授权endpoint URI的查询参数部分，并使用如[W3C.REC-html401-19991224]所定义的“application/x-www-form-urlencoded”格式构建起一个请求URI，如下定义：

response\_type

必需参数。请求的响应中：一个访问令牌、一个授权码，或两者都有。请求访问令牌参数值必须设为“token”，请求授权码参数值必须设为“code”，或者使用参数值为“code\_and\_token”同时请求两者。授权服务器可能拒绝提供这些响应类型中的一种或多种。

client\_id

必需参数。如第3节所述的客户端标识符。

redirect\_uri

必需参数，除非通过其它方式在客户端和授权服务器之间已经确定了一个重定向URI。这是当终端用户的授权步骤完成时授权服务器将要把user-agent重定向到的一个绝对URI。授权服务器应该要求客户端预先注册它们的重定向URI。

scope

可选参数。访问请求的作用域，以空格隔开的字符串列表来表示。“scope”参数的值由授权服务器定义。如果这个值包含多个空格隔开的字符串，那么它们的顺序不分先后，而且每个字符串都为请求的作用域增加一个新的访问范围。

state

可选参数。被客户端用来在请求和回调之间维护状态的值，对授权服务器来说是不透明的。授权服务器在将user-agent重定向回客户端时传回这个值。

客户端通过user-agent使用HTTP重定向响应，或者其它可用的方式，将终端用户引导到构建好的URI上。对于终端用户授权endpoint，授权服务器必须支持HTTP的“GET”方法，也可以支持使用“POST”方法。

例如，客户端引导终端用户的user-agent使用传输层安全机制发送下列HTTP请求（换行符只用于显示目的）:

GET /authorize?response\_type=code&client\_id=s6BhdRkqt3&

redirect\_uri=https%3A%2F%2Fclient%2Eexample%2Ecom%2Fcb HTTP/1.1

Host: server.example.com

如果客户端已经在授权服务器上预先注册了一个重定向URI，授权服务器必须保证收到的重定向URI与当前客户端标识符所对应的注册URI相匹配。授权服务器不应该将user-agent重定向到没有注册过的或不信任的URI，以避免endpoint被用作一个公开的转向器。如果没有可用的有效重定向URI，授权服务器应该将发生的错误报告给用户[[提供如何执行匹配操作的建议]]。

没有值的参数必须被当做它们在请求中不存在一样。授权服务器应该忽略识别不了的请求参数。

授权服务器对请求进行验证以保证所有必需参数都存在并有效。如果请求是无效的，授权服务器将使用重定向URI把user-agent重定向回客户端，并且URI后面加上适当的错误码，如4.3节所述。

授权服务器验证终端用户的身份并获得一个授权决定（通过询问用户或通过其它方式认可）。当一个决定被做出后，授权服务器将终端用户的user-agent引导到客户端提供的重定向URI，这个重定向或者使用HTTP重定向响应，或者通过终端用户user-agent的其它可用的方式。

4.2 授权响应

如果终端用户许可了访问请求，授权服务器会分发一个访问令牌，或一个授权码，或者两者都有，并且通过将下列参数添加到重定向URI将这些分发结果传递给客户端（如下所述）。

code

如果响应类型是“code”或“code\_and\_token”则是必需的，否则一定不能包含这个参数。表示由授权服务器产生的授权码。授权码应该在分发后迅速过期，以降低泄露风险。客户端一定不能重用同一个授权码。如果一个授权码被多次使用，授权服务器可能撤销之前基于这个授权码分发的所有令牌。授权码与客户端标识符和重定向URI相绑定。

access\_token

如果响应类型是“token”或“code\_and\_token”则是必需的，否则一定不能包含这个参数。表示由授权服务器分发的访问令牌。

token\_type

如果响应中包含一个访问令牌则是必需的。表示分发的令牌类型。令牌类型告诉客户端一个信息，即当访问一个受保护资源时访问令牌应该如何被使用，如6.1节所述。

expires\_in

可选参数。如果包含访问令牌参数，则表示访问令牌生命周期的秒数。例如，“3600”表示自响应被授权服务器产生的时刻起，访问令牌将在一小时后过期。

scope

可选参数。如果包含访问令牌参数，则表示访问令牌的作用域，表示为一个空格隔开的字符串列表。“scope”参数的值由授权服务器定义。如果这个值包含多个空格隔开的字符串，那么它们的顺序不分先后，而且每个字符串都为请求的作用域增加一个新的访问范围。如果请求到的作用域不同于客户端申请的作用域，授权服务器应该传回这个参数。

state

如果“state”参数在客户端授权请求中存在，则这个参数是必需的。需要精确地设置成从客户端接收到的值。

授权服务器在重定向URI上添加参数的方式取决于客户端在授权请求中请求的响应类型，由“response\_type”参数指定。

如果响应类型是“code”，授权服务器使用如[W3C.REC-html401-19991224]所定义的“application/x-www-form-urlencoded”格式添加参数到重定向URI的查询参数部分。

例如，授权服务器通过发送下列HTTP响应将终端用户的user-agent进行重定向：

HTTP/1.1 302 Found

Location: https://client.example.com/cb?code=i1WsRn1uB1

如果响应类型是“code”或“code\_and\_token”，授权服务器使用如[W3C.REC-html401-19991224]所定义的“application/x-www-form-urlencoded”格式添加参数到重定向URI的分段参数部分。

例如，授权服务器通过发送下列HTTP响应将终端用户的user-agent进行重定向（URI换行符只用于显示目的）：

HTTP/1.1 302 Found

Location: http://example.com/rd#access\_token=FJQbwq9&

token\_type=example&expires\_in=3600

客户端应该忽略无法识别的响应参数。从授权服务器接收到的令牌和其它参数值的大小，本规范未作定义。客户端应该避免对参数值大小做任何假设。服务器应该对它们所分发的任何参数值的期望大小做出文档说明。

4.3 错误响应

如果终端用户拒绝了访问请求，或者由于除了缺少或无效重定向URI之外的其它原因而导致请求失败，授权服务器使用如[W3C.REC-html401-19991224]所定义的“application/x-www-form-urlencoded”格式添加下列参数到重定向URI的查询参数部分以通知客户端：

error

必需参数。如4.3.1节所述的一个错误码。

error\_description

可选参数。提供额外信息的一段人类可读的文字，用来帮助理解和解决发生的错误。

error\_uri

可选参数。指明了一个人类可读的网页URI，带有关于错误的信息，用来为终端用户提供与错误有关的额外信息。

state

如果“state”参数在客户端授权请求中存在，则这个参数是必需的。需要精确地设置成从客户端接收到的值。

例如，授权服务器通过发送下列HTTP响应将终端用户的user-agent进行重定向：

HTTP/1.1 302 Found

Location: https://client.example.com/cb?error=access\_denied

如果由于缺少或无效重定向URI而导致请求失败，授权服务器应该通知终端用户这个错误，而一定不能将终端用户的user-agent重定向到这个无效的重定向URI。

4.3.1 错误码

授权服务器在错误响应中包含下列错误码之一：

invalid\_request

请求缺少某个必需参数，包含一个不支持的参数或参数值，或者格式不正确。

invalid\_client

提供的客户端标识符是无效的。

unauthorized\_client

客户端没有权限使用该请求的响应类型。

redirect\_uri\_mismatch

提供的重定向URI与预先注册的值不匹配。

access\_denied

终端用户或授权服务器拒绝了请求。

unsupported\_response\_type

请求的响应类型不为授权服务器所支持。

invalid\_scope

请求的作用域是无效的、未知的，或格式不正确的。

[[增加扩展错误码的机制]]

5. 获取访问令牌

客户端通过在授权服务器上验证并出示它的访问许可（表示成授权码、资源拥有者私有证书、断言或刷新令牌的形式）来获取一个访问令牌。

既然对于令牌endpoint的请求会导致在HTTP请求和响应中传输明文证书，授权服务器必需要求在向令牌endpoint发送请求的时候使用传输层安全机制。服务器必需支持[RFC5246]所定义的TLS 1.2，并且可能支持额外的传输层安全机制。

客户端通过向令牌endpoint发送一个HTTP POST请求来获取一个访问令牌。令牌endpoint的位置能够在服务器文档中找到。令牌endpoint URI可能包含一个查询参数部分。

客户端通过在请求中添加客户端私有证书与授权服务器进行验证，如第3节所述。当客户端标识符不重要的时候（例如匿名客户端），或当客户端标识符通过其它方式确定的时候（例如使用一个断言访问许可），授权服务器可能允许不经验证的访问令牌请求。

客户端通过在HTTP请求的entity-body中使用“application/x-www-form-urlencoded”格式包含下列参数，来构建请求：

grant\_type

必需参数。在请求中所包含的访问许可类型。它的值必须是“authorization\_code”、“password”、“refresh\_token”、“client\_credentials”或一个用来标识被授权服务器所支持的断言类型的绝对URI。

scope

可选参数。访问请求的作用域，表达为一个由空格隔开的字符串列表。“scope”参数的值由授权服务器定义。如果这个值包含多个空格隔开的字符串，那么它们的顺序不分先后，而且每个字符串都为请求的作用域增加一个新的访问范围。如果使用的访问许可已经代表了一个许可作用域（例如，授权码、断言），那么请求的作用域必须等于或少于之前许可的作用域，如果缺少这个参数就认为是等于之前的许可作用域。

另外，对于5.1节列出的某个访问许可类型，客户端必须包含合适的参数。

没有值的参数必须被当做它们在请求中不存在一样。授权服务器应该忽略识别不了的请求参数。

5.1 访问许可类型

客户端使用一个授权码、资源拥有者密码证书、客户端私有证书、刷新令牌或断言来请求一个访问许可。

5.1.1 授权码

客户端使用“authorization\_code”访问许可类型和下列参数传入授权码：

code

必需参数。从授权服务器接收到的授权码。

redirect\_uri

必需参数。在最初请求中使用的重定向URI。

例如，客户端通过如第3节所述的“client\_secret”参数包含客户端私有证书，并使用传输层安全机制，来发送下列HTTP请求（换行符只用于显示目的）：

POST /token HTTP/1.1

Host: server.example.com

Content-Type: application/x-www-form-urlencoded

grant\_type=authorization\_code&client\_id=s6BhdRkqt3&

client\_secret=gX1fBat3bV&code=i1WsRn1uB1&

redirect\_uri=https%3A%2F%2Fclient%2Eexample%2Ecom%2Fcb

授权服务器必须：

* 验证客户端私有证书（如果存在）并保证它们与授权码匹配。
* 验证授权码和重定向URI都是有效的，并且与存储的关联关系相匹配。

如果请求有效，授权服务器分发一个成功响应，如5.2节所述。

5.1.2 资源拥有者密码证书

客户端使用“password”访问许可类型和下列参数传入资源拥有者密码证书：[[增加对于用户名和密码的国际化考虑]]

username

必需参数。资源拥有者的用户名。

password

必需参数。资源拥有者的密码。

例如，客户端通过如第3节所述的“client\_secret”参数包含客户端私有证书，并使用传输层安全机制，来发送下列HTTP请求（换行符只用于显示目的）：

POST /token HTTP/1.1

Host: server.example.com

Content-Type: application/x-www-form-urlencoded

grant\_type=password&client\_id=s6BhdRkqt3&

client\_secret=47HDu8s&username=johndoe&password=A3ddj3w

授权服务器必须验证客户端私有证书（如果存在）和终端用户私有证书，而且如果发现有效则必须发布一个访问令牌响应，如5.2节所述。

5.1.3 客户端私有证书

客户端可以仅仅使用它的客户端私有证书来请求一个访问令牌，即使用“client\_credentials”访问许可类型。当省略一个显式的访问许可时，客户端是在请求访问它所控制的受保护资源，或另一个资源拥有者之前与授权服务器约定好的受保护资源（约定方式不在本规范的规定范围之内）。

5.1.4 刷新令牌

客户端使用“refresh\_token”访问许可类型和下列参数传入刷新令牌：

refresh\_token

必需参数。与待刷新的访问令牌相关联的刷新令牌。

例如，客户端通过如第3节所述的“client\_secret”参数包含客户端私有证书，并使用传输层安全机制，来发送下列HTTP请求（换行符只用于显示目的）：

POST /token HTTP/1.1

Host: server.example.com

Content-Type: application/x-www-form-urlencoded

grant\_type=refresh\_token&client\_id=s6BhdRkqt3&

client\_secret=8eSEIpnqmM&refresh\_token=n4E9O119d

授权服务器必须验证客户端私有证书（如果存在），验证刷新令牌是否有效，以及验证资源拥有者的授权是否仍然有效。如果请求有效，授权服务器则发布一个访问令牌响应，如5.2节所述。授权服务器可以发布一个新的刷新令牌，在这种情况下，客户端必须丢弃旧的刷新令牌并且用新的访问令牌替换。

5.1.5 断言

客户端使用一个绝对URI（由授权服务器定义）作为“grant\_type”参数的值指定断言格式，并添加下列参数，来传入一个断言：

assertion

必需参数。断言。

例如，客户端使用传输层安全机制来发送下列HTTP请求，且客户端验证通过断言来完成（换行符只用于显示目的）：

POST /token HTTP/1.1

Host: server.example.com

Content-Type: application/x-www-form-urlencoded

grant\_type=urn%3Aoasis%3Anames%3Atc%3ASAML%3A2.0%3Aassertion&

assertion=PHNhbWxwOl[...omitted for brevity...]ZT4%3D

授权服务器必须验证客户端私有证书（如果存在）和断言，而且如果发现有效则必须发布一个访问令牌响应，如5.2节所述。授权服务器不应该分发一个刷新令牌（而是应该要求客户端使用相同的或新的断言）。

授权服务器应该分发具有有限生命周期的访问令牌，并且要求客户端使用仍然有效的同一个断言来请求新的访问令牌，从而完成对令牌的刷新。

5.2 访问令牌响应

在接收到并验证过来自客户端的一个有效且经授权的访问令牌请求之后，授权服务器分发访问令牌和可选的刷新令牌，并且通过一个200（OK）状态码在HTTP响应的entity body中添加下列参数来构造响应：

令牌响应包含下列参数：

access\_token

必需参数。由授权服务器分发的访问令牌。

token\_type

必需参数。分发的令牌类型。令牌类型告诉客户端一个信息，即当访问一个受保护资源时访问令牌应该如何被使用，如6.1节所述。

expires\_in

可选参数。访问令牌生命周期的秒数。例如，“3600”表示自响应被授权服务器产生的时刻起，访问令牌将在一小时后过期。

refresh\_token

可选参数。用来获取新的访问令牌的刷新令牌，如5.1.4节所述使用相同的终端用户访问许可。当访问许可类型是一个断言或一个客户端私有证书集合时，授权服务器不应该分发一个刷新令牌。

scope

可选参数。访问令牌的作用域，表示为一个空格隔开的字符串列表。“scope”参数的值由授权服务器定义。如果这个值包含多个空格隔开的字符串，那么它们的顺序不分先后，而且每个字符串都为请求的作用域增加一个新的访问范围。如果请求到的作用域不同于客户端申请的作用域，授权服务器应该传回这个参数。

参数包含在HTTP响应的entity body中，使用[RFC4627]定义的“application/json”媒体类型。通过在最高结构层次上添加每个参数，将它们序列化成一个JSON结构。参数名和字符串值都表示成JSON字符串。数字值表示成JSON数字。

在任何包含令牌、密钥或其它敏感信息的响应中，授权服务器必须在“Cache-Control”响应头部字段中传入一个“no-store”的值。

例如：

HTTP/1.1 200 OK

Content-Type: application/json

Cache-Control: no-store

{

"access\_token":"SlAV32hkKG",

"token\_type":"example",

"expires\_in":3600,

"refresh\_token":"8xLOxBtZp8"

}

客户端应该忽略无法识别的响应参数。从授权服务器接收到的令牌和其它参数值的大小，本规范未作定义。客户端应该避免对参数值大小做任何假设。服务器应该对它们所分发的任何参数值的期望大小做出文档说明。

5.3 错误响应

如果令牌请求是无效的或未经授权的，授权服务器通过在HTTP响应的entity body中添加下列参数并使用“application/json”媒体类型来构造响应：

error

必需参数。如4.3.1节所述的一个错误码。

error\_description

可选参数。提供额外信息的一段人类可读的文字，用来帮助理解和解决发生的错误。

error\_uri

可选参数。指明了一个人类可读的网页URI，带有关于错误的信息，用来为终端用户提供与错误有关的额外信息。

例如：

HTTP/1.1 400 Bad Request

Content-Type: application/json

Cache-Control: no-store

{

"error":"invalid\_request"

}

如果客户端通过“Authorization”请求头部字段使用HTTP验证机制这种方式提供了无效的私有证书，那么授权服务器必须用HTTP 401（Unauthorized）状态码进行响应。否则，授权服务器应该用HTTP 400（Bad Request）状态码进行响应。

5.3.1 错误码

授权服务器在错误响应中传回下列错误码之一：

invalid\_request

请求缺少某个必需参数，包含一个不支持的参数或参数值，参数重复，包含多个私有证书，使用了多种验证客户端的机制，或者请求格式不正确。

invalid\_client

提供的客户端标识符是无效的，客户端验证失败，客户端不包含私有证书，提供了多个客户端私有证书，或使用了不支持的证书类型。

unauthorized\_client

经过验证的客户端没有权限使用提供的访问许可类型。

invalid\_grant

提供的访问许可是无效的、过期的或已撤销的（例如，无效的断言，过期的授权令牌，错误的终端用户密码证书，或者不匹配的授权码和重定向URI）。

unsupported\_grant\_type

包含的访问许可——它的类型或其它属性——不被授权服务器所支持。

invalid\_scope

请求的作用域是无效的、未知的、格式不正确的，或超出了之前许可的作用域。

[[增加扩展错误码的机制]]

6. 访问受保护资源

客户端通过向资源服务器出示一个访问令牌来访问受保护资源。资源服务器必须验证访问令牌，保证它没有过期并且它的作用域覆盖了请求的资源。被资源服务器用来验证访问令牌的方式不在本规范的规定范围之内，但是它通常需要在资源服务器和授权服务器之间进行交互或配合。

客户端利用访问令牌来验证资源服务器的方式取决于由授权服务器分发的访问令牌类型。

6.1 访问令牌类型

[[增加令牌类型的解释，可能包含指定其它规范的链接]]

6.2 WWW-Authenticate响应头部字段

如果对于受保护资源的请求不包含验证证书，包含一个无效的访问令牌，或格式不正确，那么资源服务器必须包含一个HTTP “WWW-Authenticate”响应头部字段。这个“WWW-Authenticate”头部字段使用[RFC2617]定义的框架，如下所示：

challenge = "OAuth2" [ RWS 1#param ]

param = scope /

error / error-desc / error-uri /

( token "=" ( token / quoted-string ) )

scope = "scope" "=" <"> scope-v \*( SP scope-v ) <">

scope-v = 1\*quoted-char

quoted-char = ALPHA / DIGIT /

"!" / "#" / "$" / "%" / "&" / "'" / "(" / ")" /

"\*" / "+" / "-" / "." / "/" / ":" / "<" / "=" /

">" / "?" / "@" / "[" / "]" / "^" / "\_" / "`" /

"{" / "|" / "}" / "~" / "\" / "," / ";"

error = "error" "=" quoted-string

error-desc = "error\_description" "=" quoted-string

error-uri = "error\_uri" = <"> URI-reference <">

“scope”属性是一个空格隔开的作用域值的列表，表明了为了访问受保护资源所需要访问令牌作用域。“scope”属性一定不能出现多次。

如果对于受保护资源的请求包含一个访问令牌并且验证失败了，那么资源服务器应该包含“error”属性来向客户端提供为何访问请求被拒绝的原因。参数值在6.2.1中描述。另外，资源服务器可能包含“error\_description”属性来提供一个人类可读的解释，或者包含“error-uri”属性用一个绝对URI来指定一个用于解释错误的人类可读的网页。“error”、“error\_description”和“error-uri”属性一定不能出现多次。

例如，对于一个缺少验证的受保护资源请求的响应：

HTTP/1.1 401 Unauthorized

WWW-Authenticate: OAuth2

对于一个使用过期访问令牌尝试验证的受保护资源请求的响应：

HTTP/1.1 401 Unauthorized

WWW-Authenticate: OAuth2

error="invalid\_token",

error\_description="The access token expired"

6.2.1 错误码

当一个请求失败时，资源服务器使用恰当的HTTP状态码（典型的如400、401或403）进行响应，并且在响应中包含下列错误码之一：

invalid\_request

请求缺少某个必需参数，包含一个不支持的参数或参数值，参数重复，使用多种方式包含访问令牌，或者请求格式不正确。资源服务器应该使用HTTP 400（Bad Request）状态码进行响应。

invalid\_token

提供的访问令牌是过期的、已撤销的、格式不正确的，或由于其它原因是无效的。资源服务器应该使用HTTP 401（Unauthorized）状态码进行响应。客户端可能请求一个新的访问令牌并重试受保护资源请求。

insufficient\_scope

请求需要比访问令牌所提供的权限更高的权限。资源服务器应该使用HTTP 403（Forbidden）状态码进行响应并且包含“scope”属性，带上访问该受保护资源必需的作用域。

[[增加扩展错误码的机制]]

如果请求中缺少任何验证信息（即客户端没有意识到验证是必需的或尝试使用一个不支持的验证方法），那么资源服务器不应该包含一个错误码和其它错误信息。

例如：

HTTP/1.1 401 Unauthorized

WWW-Authenticate: OAuth2

7. 扩展

7.1 定义新的客户端证书类型

[[待定]]

7.2 定义新的Endpoint参数

希望在终端用户授权endpoint和令牌endpoint上定义新的请求和响应参数的应用，应该使用下列两种方式之一：在参数注册表中注册（遵从9.1节描述的流程），或使用“x\_”参数名前缀。

使用“x\_”参数名前缀的参数必须局限于那些不会被广泛应用的针对厂商特性的扩展，并且特定于授权服务器使用场景的实现细节。所有其它参数必须被注册，并且一定不能使用“x\_”参数名前缀。

参数名必须遵从param-name的ABNF，并且参数值语法必须被规范地定义（例如，使用ABNF，或者引用到现存参数的语法）。

param-name = 1\*name-char

name-char = "-" / "." / "\_" / DIGIT / ALPHA

7.3 定义新的头部字段参数

希望在OAuth “WWW-Authenticate”头部字段中定义新参数的应用必须在参数注册表中进行注册，遵从9.1节描述的流程。

参数名必须遵从param-name的ABNF并且不能以“x\_”开头。参数值必须遵从param-value的ABNF并且语法必须被规范地定义（例如，使用ABNF，或者引用到现存参数的语法）。

param-value = quoted-value | quoted-string

7.4 定义新的访问许可类型

断言访问许可类型允许授权服务器接受其它未规定的访问许可。希望定义其它访问许可类型的应用能够通过利用新的或现存的断言类型和格式进行实现。

8. 安全考虑

[[待定]]

9. IANA事项

9.1 OAuth参数注册表

本文档设立参数注册表。

用于终端用户授权endpoint的请求、终端用户授权endpoint的响应、令牌endpoint的请求、令牌endpoint的响应或“WWW-Authenticate”头部字段的多余参数，在一个或多个“指派专家”（由IESG或他们的代理机构指定）的指导下进行注册，遵从所需的规范（使用[RFC5226]的术语）。然而，为了允许在发布之前对值进行分配，“指派专家”们一旦认同这样的一个规范能够发布，可能就立即批准注册。

注册请求应该发送给[待定]@ietf.org邮件组进行评审和讨论，使用恰当的标题（如“Request for parameter: example”）。[[RFC-EDITOR注解：邮件组名称应该在与IESG和IANA磋商后确定。建议名称：oauth-ext-review。]]

在14天之内，“指派专家”们会批准或拒绝注册请求，并将结果告知邮件组和IANA。拒绝的决定应该包含一个解释，并且，如果可行的话，应该包含如何进行修改的建议。在21天以上未确定的注册请求会交由IESG处理（使用iesg@iesg.org邮件组）。

9.1.1 注册模板

Parameter name: 请求的名称（例如“example”）。

Parameter usage location: 参数能够被使用的位置。可能的位置包括：终端用户授权endpoint的请求、终端用户授权endpoint的响应、令牌endpoint的请求、令牌endpoint的响应或“WWW-Authenticate”头部字段。

Change controller: 对于标准轨道的RFC，写明“IETF”。对于其它规范，使用负责机构的名称。其它细节（例如，邮编地址，电子邮件地址，主页URI）也可以包含。

Specification document(s): 对规定参数的文档引用，可取的方式是包含一个能够获取到文档拷贝的URI。对于相关章节的标示也可以包含，但不是必需的。

Related information: 可选地，对于包含更多相关信息的额外文档的引述。

9.1.2 例子

下面是对于本规范定义的“scope”参数的参数注册请求：

Parameter name: scope

Parameter usage location: The end-user authorization endpoint

request, the end-user authorization endpoint response, the token

endpoint request, the token endpoint response, and the

"WWW-Authenticate" header field.

Change controller: IETF

Specification document(s): [[ this document ]]

Related information: None

附录 翻译略。