ICS 35.040
L 80

GM

中华人民共和国密码行业标准

GM/T 0068—2019

开放的第三方资源授权协议框架

Open third party resource authorization protocol framework

2019-07-12 发布　　2019-07-12 实施

国家密码管理局　　发 布

目　　次

前　言

本标准按照 GB/T 1.1—2009 给出的规则起草。

本标准参考国际互联网工程任务组(The Internet Engineering Task Force,简称 IETF)的 RFC 6749 文件《The OAuth2.0 Authorization Framework》进行制定。按照我国相关密码政策和法规,结合我国实际应用需求及产品生产厂商的实践经验,本标准在第三方应用程序身份鉴别部分增加了基于 SM2 国产密码算法的数字证书鉴别方法,在授权协议中的数据通信安全部分采用密码行业标准 GM/T 0024—2014《SSL　VPN　技术规范》中定义的安全通信协议取代 TLS 协议,在访问令牌的保护部分增加了采用 SM2、SM3、SM4 等国家密码管理局认可的算法对其进行签名和加密的规定。另外,本标准去除了RFC 6749文件中的安全考虑部分,将安全考虑部分涉及的应采用的安全措施具体化到本标准的各个章条,包括协议中传输的消息、端点、发放的令牌、第三方应用程序身份鉴别等部分。

本标准由密码行业标准化技术委员会提出并归口。

本标准的主要起草单位:中国科学院数据与通信保护研究教育中心、北京数字认证股份有限公司、中国科学院软件研究所、中国电子技术标准化研究院、北京信安世纪科技股份有限公司、普华诚信信息技术有限公司。

本标准主要起草人:刘丽敏、李敏、王鑫、江伟玉、高能、刘宗斌、荆继武、林雪焰、张立武、汪宗斌、彭佳、屠晨阳、刘泽艺、钱文飞、范科峰、郝春亮、梁佐泉。

引　言

在提供了资源互访接口的开放信息系统中，利用 Web、桌面、手机或其他智能设备应用程序实现互联已成为常态。为了实现信息资源共享、业务合作，用户可利用某个安全域中的应用程序（被称为第三方应用程序）访问另一个安全域中受保护的资源。为了确保受保护的资源只被资源拥有者许可的实体访问，需要对实体进行鉴别与授权。然而，在传统的授权模型中，资源拥有者通常需要将其身份凭证共享给访问者，这种方式带来了诸多安全隐患。本标准引入授权层，将第三方应用程序与资源拥有者的角色进行分离，在资源拥有者的授权下，授权实体向第三方应用程序发放不同于身份凭据的令牌方式，实现开放的第三方资源授权。

开放的第三方资源授权协议框架

1 范围

本标准规定了第三方资源授权协议的流程、不同类型的授权许可、协议各端点的功能要求以及系统实体之间传递消息的格式和参数要求等。

本标准适用于在互联网跨安全域应用场景中，身份鉴别与授权服务的开发、测试、评估和采购。

2 规范性引用文件

下列文件对于本文件的应用是必不可少的。凡是注日期的引用文件，仅注日期的版本适用于本文件。凡是不注日期的引用文件，其最新版本(包括所有的修改单)适用于本文件。

GB/T 15843.3—2008 信息技术 安全技术 实体鉴别 第3部分：采用数字签名技术的机制

GB/T 32905—2016 信息安全技术 SM3密码杂凑算法

GB/T 32907—2016 信息安全技术 SM4分组密码算法

GB/T 32918.2—2016 信息安全技术 SM2椭圆曲线公钥密码算法 第2部分：数字签名算法

GB/T 32918.4—2016 信息安全技术 SM2椭圆曲线公钥密码算法 第4部分：公钥加密算法

GM/T 0024—2014 SSL VPN技术规范

RFC 1867 HTML中基于表单的文件上传(Form-based File Upload in HTML)

RFC 2616 超文本传输协议HTTP1.1(Hypertext Transfer Protocol—HTTP/1.1)

RFC 2617 HTTP鉴别：基本访问鉴别和摘要访问鉴别(HTTP Authentication：Basic and Digest Access Authentication)

RFC 3986 统一资源标识符：通用语法(Uniform Resource Identifier (URI)：Generic Syntax)

RFC 6749 OAuth 2.0授权框架(The OAuth 2.0 Authorization Framework)

3 术语和定义

下列术语和定义适用于本文件。

3.1

访问令牌 access token

授权服务器发放的令牌，用于证明某实体具有访问特定范围内受保护资源的权限。

3.2

授权 authorization

授予访问者访问受保护资源的权限。

3.3

授权码 authorization code

授权服务器发放给第三方应用程序的凭据，表明资源拥有者同意第三方应用程序访问受保护资源，第三方应用程序可使用授权码获取访问令牌和刷新令牌。

3.4

授权端点 authorization endpoint

授权服务器上用于与资源拥有者交互的端点，用于接收资源拥有者的身份凭据和授权，以及返回授

权许可给第三方应用程序。

3.5

授权许可　authorization grant

授权服务器发放给第三方应用程序的凭据，表明资源拥有者同意第三方应用程序访问受保护资源。第三方应用程序不应使用该凭据直接访问受保护资源，应使用该凭据从授权服务器换取访问令牌。

3.6

授权服务器　authorization server

对第三方应用程序进行授权的服务器。授权服务器与资源服务器可以是同一实体，也可以是相互分离的两个实体。同一台授权服务器所发放的访问令牌可被多台资源服务器识别和接受。

3.7

端点　endpoint

授权服务器上用于接收请求消息、返回响应消息的接口。

3.8

终端用户　end-user

使用信息系统和系统资源的自然人。

3.9

重定向端点　redirection endpoint

第三方应用程序上用于接收授权响应的端点。

3.10

刷新令牌　refresh token

第三方应用程序从授权服务器获得的令牌。第三方应用程序使用该令牌向授权服务器重新请求新的访问令牌。

3.11

资源拥有者　resource owner

拥有受保护资源的实体，能够对受保护资源的访问进行授权。如果资源拥有者是自然人，则称为终端用户。

3.12

资源服务器　resource server

存储受保护资源的服务器，能够接收和响应对受保护资源的访问请求。

3.13

第三方应用程序　third party application

请求访问受保护资源的应用程序。

3.14

令牌　token

授权服务器发放给第三方应用程序的凭据，分为访问令牌和刷新令牌两种类型。

3.15

令牌端点　token endpoint

授权服务器上用于与第三方应用程序交互的端点，授权服务器使用该端点接收第三方应用程序发起的令牌请求，当授权服务器验证请求成功后，通过该端口返回令牌给第三方应用程序。

4　缩略语

下列缩略语适用于本文件。

HTTP　超文本传输协议(Hypertext Transfer Protocol)
OAuth　开放的第三方资源授权协议框架(Open Authorization Protocol Framework)
TLS　安全传输层协议(Transport Layer Security)
URI　统一资源标识(Uniform Resource Identifier)
URL　统一资源定位符(Uniform Resource Locator)

5　概述

5.1　协议流程

本标准定义的基本协议流程如图 1 所示。

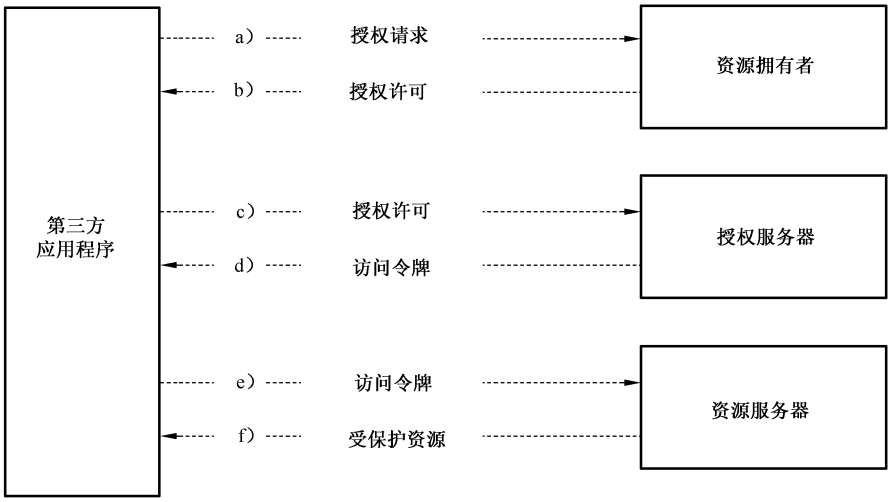

图 1　OAuth 基本协议流程图

图 1 所示的 OAuth 流程描述了第三方应用程序、资源拥有者、授权服务器、资源服务器等四种角色之间的交互，OAuth 参数说明参见附录 A 中的 A.1，协议基本流程如下：

a)　第三方应用程序向资源拥有者请求授权。第三方应用程序请求资源拥有者授权许可的方式有两种，一种是直接发送请求给资源拥有者，另一种是间接地通过授权服务器作为中介发送请求给资源拥有者。本标准推荐第三方应用程序使用授权服务器作为中介的方式来获取资源拥有者的授权许可，在图 2 和图 3 中给出了示意；

b)　资源拥有者(或者资源拥有者间接地通过授权服务器，如步骤 a)中所述)发放授权许可给第三方应用程序。该授权许可是资源拥有者授权的凭据，采用的类型为 7.1.1 所定义的四种类型之一。授权许可的类型取决于第三方应用程序请求授权许可的方式(即步骤 a)中描述的两种方式)和授权服务器所支持的授权许可类型(通常由授权服务器的服务文档提供)；

c)　第三方应用程序对授权服务器进行鉴别并呈递授权许可，向授权服务器请求访问令牌；

d)　授权服务器鉴别第三方应用程序的身份并验证授权许可的有效性，如果第三方应用程序身份鉴别通过且授权许可有效，则授权服务器向第三方应用程序发放访问令牌；

e)　第三方应用程序向资源服务器发送访问令牌及相关参数，用以请求访问受保护资源；

f)　资源服务器验证访问令牌的有效性。如果访问令牌有效，则将请求的资源返回给第三方应用程序。

基于上述协议流程，第三方应用程序可通过资源拥有者使用的用户代理(如浏览器)与资源拥有者进行交互，在某些特定情形下(见 7.5)，第三方应用程序可不与资源拥有者交互，直接使用第三方应用程

序的身份凭据获取授权服务器发放的访问令牌。如果第三方应用程序对受保护资源的访问超出访问令牌范围或有效期，则访问失效，第三方应用程序应重新开始协议流程或采用本标准规定的刷新机制进行访问令牌更新。

本标准中使用超文本传输协议 HTTP1.1(见 RFC 2616)进行通信。基于其他任何协议的开放的第三方资源授权协议框架，不属于本标准的讨论范围。

5.2 协议通道要求

对于授权码、访问令牌、刷新令牌、资源拥有者口令凭据以及第三方应用程序身份凭据等敏感的数据，不应采用明文传输方式，应采用 GM/T 0024—2014 定义的安全传输层协议进行传输。授权服务器应与第三方应用程序进行双向鉴别。由于〈state〉和〈scope〉参数(见 7.2.2)可能通过不安全信道传输，或被存储于不安全环境，因此，它们不应以明文形式包含第三方应用程序或资源拥有者的敏感信息。

5.3 协议端点

5.3.1 协议端点类型

授权过程使用了两个授权服务器上的端点：

a) 授权端点；

b) 令牌端点。

以及一个第三方应用程序的端点：重定向端点。

并非每一类授权许可类型的授权流程都使用了以上三种端点。

授权端点和令牌端点允许第三方应用程序在请求中添加〈scope〉参数，用该参数指明资源访问请求中请求的受保护资源访问范围。相应地，授权服务器通过在响应中包含〈scope〉参数来告知第三方应用程序其被发放的访问令牌的受保护资源访问范围。

〈scope〉参数的值以一组由空格分隔的大小写敏感的字符串表示，字符串的顺序不影响解析(“a b”等价于“b a”)。〈scope〉参数所包含字符串的含义由授权服务器定义，通常由授权服务器的服务文档提供。

根据授权服务器的策略或者是资源拥有者的参与，授权服务器可全部或部分地拒绝第三方应用程序所请求的受保护资源访问范围。如果所发放的访问令牌的受保护资源访问范围与第三方应用程序所请求的不同，授权服务器应在响应中包含〈scope〉参数，以告知第三方应用程序其实际被允许的受保护资源访问范围。

如果第三方应用程序在请求授权时省略了〈scope〉参数，授权服务器应使用预定义的默认值回应此请求，或者拒绝此请求。授权服务器应在其服务文档中对〈scope〉参数的要求和默认值进行说明。

端点使用 URI 进行标识。URI 的典型格式是：[方案(scheme)：][//主机名(authority)][路径(path)][？查询组件(query)][＃片段组件(fragment)]，查询组件和片段组件的编码格式应为“application/x-www-form-urlencoded”(见 RFC 1867)编码格式。

5.3.2 授权端点

授权服务器上用于与资源拥有者交互的端点，用于接收第三方应用程序的授权请求、资源拥有者的身份凭据和授权，以及返回授权许可给第三方应用程序。当授权服务器收到第三方应用程序的授权请求时，授权服务器应首先验证资源拥有者的身份。授权服务器鉴别资源拥有者的方式(例如，用户名口令登录，会话 cookie)不属于本标准的讨论范围。

授权端点 URI 通常由授权服务器的服务文档提供。

授权端点的 URI 可包含查询组件。当增加其他查询参数时，应保留该查询组件，该端点 URI 不应

包含片段组件。

授权服务器的授权端点应支持 HTTP GET 方法（见 RFC 2616），同时也可支持 HTTP POST 方法（见 RFC 2616）。

5.3.3 令牌端点

第三方应用程序呈递授权许可或刷新令牌给授权服务器的令牌端点，授权服务器验证请求后，令牌端点发放访问令牌给第三方应用程序。除隐式许可之外的其他授权方式中，都会用到令牌端点。

本标准不规定第三方应用程序获取令牌端点 URI 的方式（通常由授权服务器的服务文档提供）。

令牌端点 URI 可包含查询组件。当增加其他查询参数时，应保留该查询组件。令牌端点 URI 不应包含片段组件。

第三方应用程序在向授权服务器的令牌端点请求访问令牌时应使用 POST 方法。

当授权服务器的令牌端点收到有保密能力型的第三方应用程序或其他被授予了身份凭据的第三方应用程序的请求时，授权服务器应对第三方应用程序进行身份鉴别。

5.3.4 重定向端点

5.3.4.1 概述

授权服务器完成与资源拥有者的交互之后，将资源拥有者的用户代理重定向到第三方应用程序的重定向端点。

第三方应用程序在注册阶段或构造授权请求阶段确定其重定向端点 URI。

重定向端点 URI 应是绝对路径 URI（见 RFC 3986）。重定向端点 URI 可包含查询组件，在向重定向端点 URI 添加其他查询参数时，该组件应被保留。重定向端点 URI 不应包含片段组件。

5.3.4.2 重定向端点安全要求

任何与重定向端点的通信应使用 5.2 要求的安全通信协议。如果 5.2 要求的安全协议不可用，授权服务器在重定向之前，应当向资源拥有者发出此端点不安全的警告。

授权服务器应要求下列第三方应用程序在授权服务器上进行注册时（见 6.3）登记其重定向端点：

a） 无保密能力的第三方应用程序（见 6.1）；

b） 采用隐式许可类型（见 7.3）的有保密能力型的第三方应用程序。

授权服务器应要求所有类型的第三方应用程序在使用授权端点之前向授权服务器注册第三方应用程序的重定向端点。

授权服务器应要求第三方应用程序提供完整的重定向端点 URI。如果第三方应用程序无法实现注册完整的重定向端点 URI，授权服务器应要求第三方应用程序注册方案、主机名和路径等三部分（在第三方应用程序请求授权时只允许其变更重定向端点 URI 的查询组件）。

授权服务器允许第三方应用程序注册多个重定向端点。

如果第三方应用程序注册了多个重定向端点 URI，或者只注册了重定向端点 URI 的一部分，或者没有注册重定向端点 URI，第三方应用程序在发送授权请求时，应在请求中使用〈redirect_uri〉参数来标识该次请求所使用的重定向端点 URI。

当授权请求中包含重定向端点 URI 时，如果第三方应用程序注册过重定向端点 URI，授权服务器应采用 RFC 3986 第 6 节定义的比较和匹配方法，对收到的重定向端点 URI 和之前注册过的重定向端点 URI 进行比较和匹配。如果第三方应用程序注册了完整的 URI，则授权服务器应采用 RFC 3986 第 6.2.1节定义的简单字符串比较方法对两个重定向端点 URI 进行比较。

如果授权请求由于重定向端点 URI 丢失、无效或者不匹配而未通过验证，授权服务器应告知资源

拥有者这一错误，并且不得自动将用户代理重定向到未通过验证的重定向端点 URI。

发向第三方应用程序重定向端点的重定向请求通常会获得 HTML 文档的响应，该响应由用户代理处理。第三方应用程序不应在重定向请求的响应中包含任何第三方的脚本。第三方应用程序应从重定向请求的 URI 中解析出凭据并将用户代理再次重定向到另外的端点，以避免在 URI 或其他地方暴露凭据。

6 第三方应用程序及安全要求

6.1 第三方应用程序类型

根据第三方应用程序是否对其身份凭据具有保密能力，本标准定义了两种第三方应用程序的类型：

a） 有保密能力型

第三方应用程序有能力维持其凭据的机密性（例如，第三方应用程序运行在严格执行访问控制的安全服务器上），从而可通过提供安全的身份凭据来证明自己身份的真实性，或者第三方应用程序有能力通过其他的方式（本标准不作规定）证明自己身份的真实性。

b） 无保密能力型

第三方应用程序没有能力维持其凭据的机密性（例如，第三方应用程序运行在资源拥有者使用的设备上，本地应用或是基于浏览器的应用等），无法提供安全的身份凭据来证明自己身份的真实性，并且没有能力通过其他方式证明自己身份的真实性。

第三方应用程序类型的认定取决于授权服务器的鉴别安全要求和授权服务器对第三方应用程序凭据暴露级别的接受程度（通常由授权服务器的服务文档提供）。授权服务器不应对第三方应用程序的类型进行假定。

第三方应用程序可能由一组分布式的组件共同实现，每个组件具有不同的第三方应用程序类型和安全上下文（例如，第三方应用程序同时具有基于服务器的有保密能力型组件和基于浏览器的无保密能力的组件）。对此类第三方应用程序的注册超出本标准的讨论范围，通常第三方应用程序运营商可将第三方应用程序的每个组件都注册在授权服务器上。

本标准主要涉及以下三种第三方应用程序实例：

——Web 应用

Web 应用是运行在一台 Web 服务器上的有保密能力型第三方应用程序。资源拥有者通过 HTML 用户界面来访问该应用，此 HTML 用户界面由其使用的设备中的用户代理来渲染。第三方应用程序身份凭据以及发放给第三方应用程序的所有访问令牌都存储在 Web 服务器上，对资源拥有者而言是不可获取且不可访问的。

——运行于用户代理上的应用

基于用户代理的应用是一类无保密能力型第三方应用程序，是从 Web 服务器下载到资源拥有者本地设备上、运行于用户代理（例如，Web 浏览器）中的代码。协议数据和凭据对于资源拥有者而言是可访问的。

——本地应用

本地应用是安装和运行在资源拥有者所使用的设备上的一类无保密能力型第三方应用程序。协议数据和凭据对于资源拥有者而言是可访问的。本标准假定该应用中所包含的任何第三方应用程序用于身份鉴别的凭据都可被提取出来。另一方面，动态发放的凭据（例如访问令牌和刷新令牌）应受到保护，不应被与该应用交互的恶意服务器获得，也不应受到同一设备上其他应用的威胁。

6.2 第三方应用程序标识符

第三方应用程序标识是授权服务器为注册的第三方应用程序发放的应用程序标识符。该标识符是字符串，授权服务器使用该字符串可唯一标识一个第三方应用程序。

本标准对第三方应用程序标识符的长度不作规定。授权服务器应规定该标识符的长度，并对其发放的任何长度的标识符进行详细记录。第三方应用程序不应对此标识的长度进行假定。

6.3 第三方应用程序注册要求

在协议进行之前，第三方应用程序的提供商需要在授权服务器上注册第三方应用程序的信息（例如重定向端点 URI、第三方应用程序类型等），建立第三方应用程序与授权服务器的信任关系。第三方应用程序提供商应使用授权服务器支持的注册方法（通常由授权服务器的服务文档提供）完成注册过程，具体注册方法不属于本标准的规定范围。

在注册第三方应用程序时，第三方应用程序提供商宜提供以下信息给授权服务器：

a） 第三方应用程序的类型（见 6.1）；

b） 指向第三方应用程序的重定向端点（见 5.3.4）；

c） 授权服务器所要求的其他信息（如，应用名称、网站和描述等）。

6.4 第三方应用程序身份鉴别

6.4.1 第三方应用程序鉴别方案

6.4.1.1 第三方应用程序口令凭据鉴别方案

当授权服务器使用基于口令凭据的鉴别方案对第三方应用程序进行鉴别时，授权服务器可使用 HTTP 摘要访问鉴别方案（digest access authentication scheme，见 RFC 2617）。第三方应用程序对其标识符、口令、授权服务器发送的 nonce 参数值（随机字符串，用于防止重放攻击）使用 SM3 算法（见 GB/T 32905—2016）进行杂凑运算后，再采用“application/x-www-form-urlencoded”（见 RFC 1867）编码格式进行编码，将编码后的值放在 HTTP 请求的主体部分，以 POST 请求的方式发送给授权服务器以进行身份鉴别。请求中包含如下参数：

a） 〈client_id〉[必选]

6.2 中所描述的第三方应用程序标识符。

b） 〈client_secret〉[必选]

第三方应用程序的口令。

本标准不推荐授权服务器使用口令对第三方应用程序进行鉴别。如果使用这种方案，第三方应用程序的口令应放在请求主体部分中进行传输（即应采用 POST 方式），不能包含在请求的 URI 中（即不应采用 GET 方式）。

授权服务器和第三方应用程序的交互应使用 SSL VPN 技术规范中规定的安全通信协议（见 5.2）。

由于鉴别此类第三方应用程序的方法涉及口令，授权服务器应确保所有涉及口令的端点能够抵御暴力攻击。

6.4.1.2 第三方应用程序数字证书鉴别方案

本标准推荐授权服务器使用基于 SM2 算法（见 GB/T 32918.2—2016 或 GB/T 32918.4—2016）的数字证书鉴别方案对第三方应用程序进行鉴别，推荐采用 GB/T 15843.3 中规范的相关鉴别方案。

对于访问安全要求较高的数据资源时，应使用基于 SM2 算法的数字证书鉴别方案。

6.4.1.3 其他鉴别方案

授权服务器可支持任何符合其安全要求的鉴别方案。当采用其他鉴别方案时，授权服务器应记录第三方应用程序标识符所对应的鉴别方案。

6.4.2 鉴别安全要求

有保密能力型的第三方应用程序和授权服务器之间应确立一种满足授权服务器安全要求（通常由授权服务器的服务文档提供）的身份鉴别方法，使得授权服务器可以安全地对第三方应用程序的身份进行鉴别。授权服务器通常在有保密能力型第三方应用程序注册时，授予第三方应用程序身份凭据（例如口令、数字证书），通过基于口令凭据或数字证书的鉴别方案对第三方应用程序进行身份鉴别。有保密能力型的第三方应用程序应确保其身份凭据的机密性。

无保密能力型的第三方应用程序可与授权服务器协商身份鉴别方法，但由于无保密能力型第三方应用无法保证凭据的机密性，授权服务器不能仅依赖该方法对第三方应用程序身份的真实性进行判定。

授权服务器不应发放第三方应用程序身份凭据给无保密能力型的第三方应用程序。但特殊设备上的本地应用如果具有对身份凭据的保密能力，授权服务器可发放第三方应用程序身份凭据给该类第三方应用程序。

为了防止假冒的第三方应用程序，授权服务器应对第三方应用程序的身份进行鉴别。当第三方应用程序身份鉴别流程无法实施时，授权服务器应采用其他方式来验证第三方应用程序的身份。例如，授权服务器可要求第三方应用程序注册重定向端点并将请求中的重定向端点 URI 与注册的重定向端点 URI 进行对比，或者要求资源拥有者参与确认第三方应用程序的身份（授权服务器鉴别资源拥有者的身份后，将第三方应用程序及其请求的受保护资源访问范围与生命周期提供给资源拥有者，由资源拥有者检查当前第三方应用程序的信息，并决定授权或拒绝该请求）。验证重定向端点 URI 的有效性并要求资源拥有者参与到鉴别流程中的方式，不足以验证第三方应用程序身份，但可以防止将凭据传递给假冒的第三方应用程序。

如下两种情况授权服务器不应自动处理（未与资源拥有者主动交互的情况下）重复的授权请求：

a） 未鉴别第三方应用程序；

b） 不能确认重复请求是来自真实的第三方应用程序，而不是假冒的第三方应用程序。

授权服务器应考虑与未授权第三方应用程序交互的安全性，并采取措施以规避所发放的凭据（如刷新令牌）存在的暴露风险。

第三方应用程序在同一个请求中只允许使用一种身份鉴别方案。

7 授权流程

7.1 授权许可

7.1.1 授权许可类型

授权许可可用于获取访问令牌和刷新令牌（可选）。本标准定义了四种授权许可类型——授权码许可、隐式许可、资源拥有者口令凭据许可和第三方应用程序身份凭据许可。

7.1.2 授权码许可

第三方应用程序不直接向资源拥有者请求授权，而是以授权服务器为中介向资源拥有者请求授权。第三方应用程序将资源拥有者的用户代理重定向到授权服务器，授权服务器与资源拥有者交互，对资源拥有者的身份进行鉴别，并征得资源拥有者授权后，将资源拥有者的用户代理重定向到第三方应用程

序，并在重定向消息中携带授权码。

授权码许可类型适合有保密能力型的第三方应用程序。授权码许可类型可用于获取访问令牌和刷新令牌。

7.1.3 隐式许可

隐式许可类型是授权码许可类型的简化版本，隐式许可类型不涉及授权码，授权服务器直接发送访问令牌（作为资源拥有者授权的结果）给第三方应用程序。由于第三方应用程序无需使用授权码换取访问令牌，而是直接获取访问令牌，因此该授权许可类型称为隐式许可。

隐式许可类型适用于脚本语言（如 Javascript）实现的、内嵌于浏览器的第三方应用程序访问受保护资源的场景。隐式许可类型可用于获取访问令牌，不能获取刷新令牌。

7.1.4 资源拥有者口令凭据许可

资源拥有者口令凭据可作为授权许可，用于获取访问令牌。

资源拥有者口令凭据许可类型适用于资源拥有者与第三方应用程序之间存在高度互信的情况（例如，第三方应用程序是操作系统的一部分或者某个特权应用）。

该类型也适用于能够通过其他方式获取到资源拥有者口令凭据的第三方应用程序。

7.1.5 第三方应用程序身份凭据许可

第三方应用程序身份凭据（或其他形式的能够用于第三方应用程序身份鉴别的信息）可作为授权许可，用于获取访问令牌。

第三方应用程序身份凭据许可类型适用于以下场景：

a） 受保护资源由第三方应用程序控制的场景；

b） 经过协商（协商的方式本标准不作要求），授权服务器同意第三方应用程序访问受保护资源的场景。

7.2 授权码许可流程

7.2.1 协议流程

授权码许可流程见图 2，包括以下步骤：

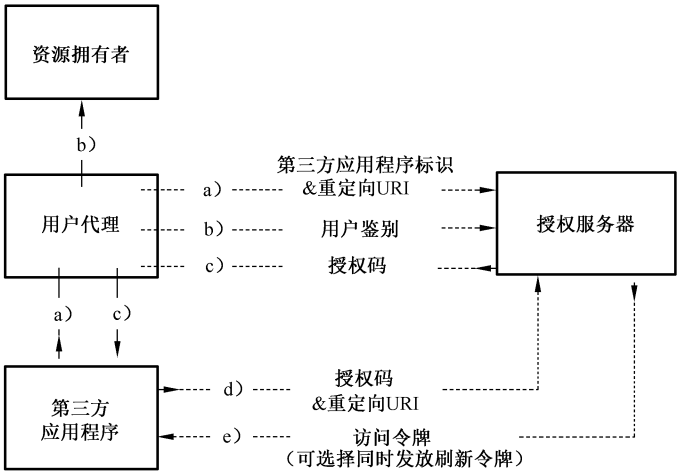

图 2 授权码协议流程

a） 第三方应用程序通过将资源拥有者的用户代理重定向到授权端点（图 2 中有两个 a），即表示

重定向的过程)向授权服务器发送授权请求(见 7.2.2)。授权请求中包含第三方应用程序标识符、申请的受保护资源访问范围、本地状态和第三方应用程序的重定向端点 URI(图 2 中仅标出了第三方应用程序标识符和重定向端点 URI 作为示例,通常请求中会包括更多参数),其中重定向端点 URI 用于第三方应用程序接收来自授权服务器的关于授权的结果。

b) 授权服务器(通过用户代理)鉴别资源拥有者并询问资源拥有者是否允许第三方应用程序的访问请求(图 2 中的两个 b)表示授权服务器通过资源拥有者的用户代理鉴别资源拥有者的过程)。

c) 假定资源拥有者同意了此次访问,授权服务器使用重定向端点 URI(见 5.3.4)将用户代理重定向到第三方应用程序(图 2 中有两个 c),即表示重定向的过程)。重定向端点 URI 中包含授权码和步骤 a)中由第三方应用程序提供的本地状态(图 2 中未表示出本地状态)。

d) 第三方应用程序向授权服务器的令牌端点发送访问令牌请求(见 7.2.4),该请求包含上一步骤中获得的授权码。在构造本次请求时,第三方应用程序应对授权服务器进行身份鉴别。同时第三方应用程序应在请求中包含其接收授权码的重定向端点 URI。

e) 授权服务器鉴别第三方应用程序,验证授权码的有效性,并确保收到的重定向端点 URI 与此前步骤 c)中的重定向端点 URI 相同。如果验证通过,授权服务器返回访问令牌(可同时返回刷新令牌)给第三方应用程序。

在授权码流程中,应满足以下安全要求:

a) 授权码应使用 GM/T 0024—2014 中描述的安全通信协议传输。

b) 在发放授权码时,如果授权服务器可以鉴别第三方应用程序,授权服务器应鉴别该第三方应用程序(见 6.4),以确保授权码发放给正确的第三方应用程序。

c) 授权码应为一次性有效,且授权码应具有较短的生命周期。如果授权服务器发现某个授权码被多次使用,授权服务器应撤销利用该授权码发放的所有访问令牌(或/和刷新令牌)。

d) 当使用授权码许可类型时,第三方应用程序通过〈redirect_uri〉参数指定重定向端点 URI。如果攻击者可以篡改重定向端点 URI 的值,将导致授权服务器把资源拥有者的用户代理重定向到攻击者控制的端点(重定向请求中包含授权码)。为了避免此类攻击,授权服务器应保证用于获取授权码的重定向端点 URI 与用授权码获取访问令牌时提供的重定向端点 URI 一致。授权服务器应要求无保密能力的第三方应用程序注册重定向端点 URI,宜要求有保密能力的第三方应用程序注册重定向端点 URI。如果请求中包含重定向端点 URI,授权服务器应根据注册值进行验证。

7.2.2 授权请求

第三方应用程序通过将以下参数添加到授权端点 URI 的查询组件,来构造授权请求:

a) 〈response_type〉[必选]

参数值应为“code”。

b) 〈client_id〉[必选]

6.2 中所描述的第三方应用程序标识符。

c) 〈redirect_uri〉[可选]

如 5.3.4 所述。

d) 〈scope〉[可选]

请求访问受保护资源的范围。

e) 〈state〉[推荐]

即 7.2.1 步骤(a)中的本地状态。开发者应确保该值的不可猜测性。授权服务器在将用户代理重定向到第三方应用程序时,在重定向请求中包含此值。第三方应用程序开发者宜使用该参数,以防止跨

站点请求伪造攻击。

授权服务器应验证授权请求的有效性，以确保所有必选参数都存在且有效。如果验证请求有效，授权服务器将对资源拥有者的身份进行鉴别并从资源拥有者处获得授权决定（通过直接询问资源拥有者的方式或其他方式）。

7.2.3 授权响应

7.2.3.1 正确响应

如果资源拥有者允许本次授权请求，授权服务器应给第三方应用程序发放授权码（通过重定向的方式）。授权服务器向重定向端点 URI 的查询组件添加如下参数（参数经过 UTF-8 编码后，再使用“application/x-www-form-urlencoded”格式编码）：

a) 〈code〉[必选]

由授权服务器产生的授权码。为降低泄露的风险，该授权码应在发放后短时间内失效。推荐授权码最长生命周期为 10 min。第三方应用程序不得重复使用该授权码。如果授权码被重复使用，授权服务器应拒绝该次请求并撤销此前基于该授权码所发放的所有访问令牌。授权服务器应将授权码与第三方应用程序标识符、重定向 URI 进行绑定。

b) 〈state〉

如果第三方应用程序的授权请求中含有此参数，则授权响应中也应包含此参数。该参数的值与第三方应用程序授权请求中的〈state〉值相同。

第三方应用程序应忽略未识别的响应参数。本标准对授权码的字符串长度不做定义。第三方应用程序不宜对字符串长度做出假定。授权服务器应在服务文档中说明其所有发放的参数值的长度。

7.2.3.2 出错响应

如果由于重定向端点 URI 丢失、无效或不匹配，或由于第三方应用程序标识符丢失或无效等原因出错，授权服务器应当告知资源拥有者这一错误，并且不得自动将用户代理重定向至无效的重定向端点 URI。

如果由于资源拥有者拒绝访问请求或是由于重定向端点 URI 有误之外的其他原因导致授权失败，授权服务器应向重定向端点 URI 的查询组件添加如下参数，以通知第三方应用程序出错原因：

a) 〈error〉[必选]

ASCII 码格式的错误代码，错误代码类型如下所示：

1) invalid_request　表示请求中丢失了必选参数，或存在无效参数，或重复包含了某个参数，或是其他形式的格式错误；
2) unauthorized_client　表示该第三方应用程序无权使用当前方法请求授权码；
3) access_denied　表示授权服务器拒绝访问请求（注：不区分资源拥有者拒绝请求和授权服务器拒绝请求，均采用相同的错误代码类型）；
4) unsupported_response_type　表示授权服务器不支持授权码许可类型；
5) invalid_scope　表示所请求的资源访问范围无效、未知或是形式不当；
6) server_error　表示授权服务器遇到了意外情况从而不能响应该请求（该错误代码是需要的，因为 HTTP 的 500 内部服务器错误代码不能通过 HTTP 重定向返回）；
7) temporarily_unavailable　表示由于暂时的过载或是服务器维护，授权服务器目前无法处理该请求（该错误代码是需要的，因为 HTTP 的 503 服务错误代码不能通过 HTTP 重定向返回）；

〈error〉参数的值不应包含％x20-21/％x23-5B/％x5D-7E 集合之外的字符。

b) 〈error_description〉[可选]

终端用户可读的 ASCII 文本，提供附加信息以帮助第三方应用程序的开发者理解出现的错误。

c) 〈error_uri〉[可选]

用于标识网页，该网页含有终端用户可读的、关于该错误更多信息。该参数的值应遵循 URI-reference 语法，并且不得包含%x20-21/%x23-5B/%x5D-7E 集合之外的字符。

d) 〈state〉

如果第三方应用程序的授权请求中含有此参数，则响应中也应包含此参数。响应中该参数的值与授权请求中的值相同。

7.2.4 访问令牌请求

第三方应用程序向令牌端点发送请求，以获取访问令牌。第三方应用程序应在请求的主体部分添加如下参数(参数经过 UTF-8 编码后，再使用“application/x-www-form-urlencoded”格式编码)：

a) 〈grant_type〉[必选]

参数值应为 “authorization_code”。

b) 〈code〉[必选]

从授权服务器获得的授权码。

c) 〈redirect_uri〉

如果第三方应用程序在 7.2.2 的授权请求中包含此参数，则在该请求中也应包含此参数。应确保该参数两次的值相同。

d) 〈client_id〉

如果第三方应用程序未按照 6.4 的要求被授权服务器鉴别身份，则第三方应用程序应在该请求中添加此参数。

如果第三方应用程序的类型为有保密能力型，或该第三方应用程序被发放了第三方应用程序身份凭据(或被其他鉴别要求所约束)，该第三方应用程序应如 6.4 所述被授权服务器鉴别身份。

授权服务器应：

——要求对有保密能力型的第三方应用程序或任何被发放过第三方应用程序身份凭据(或被其他鉴别要求所约束)的第三方应用程序进行身份鉴别；

——如果需要鉴别该第三方应用程序，则执行鉴别流程；

——确保发放授权码给正确的第三方应用程序，即通过了身份鉴别的有保密能力型第三方应用程序，或是请求中〈client_id〉参数所标识的无保密能力型第三方应用程序；

——验证授权码是否有效；

——如果第三方应用程序在 7.2.2 中定义的授权请求中包含〈redirect_uri〉参数，则授权服务器应确保该请求中也包含此参数，且确保该参数两次的值相同。

7.2.5 访问令牌响应

如果访问令牌请求是有效的，授权服务器应按照 8.2.2 所述发放访问令牌和刷新令牌(可选的)。如果授权服务器对发起访问令牌请求的第三方应用程序的身份鉴别失败，或对访问令牌请求的验证失败，授权服务器应返回 8.2.3 所述的出错响应。

7.3 隐式许可流程

7.3.1 协议流程

隐式许可流程见图 3，包括以下步骤：

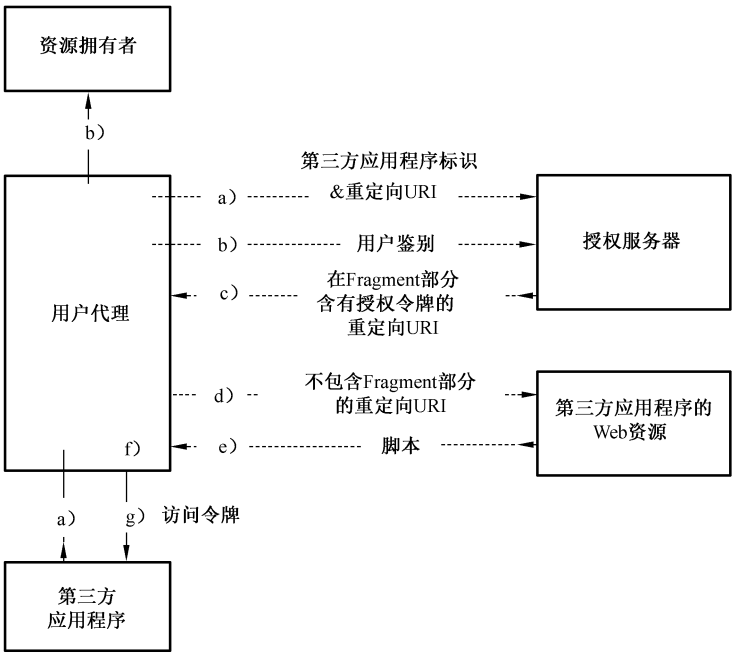

图 3　隐式许可流程

a）　第三方应用程序通过将资源拥有者的用户代理重定向到授权端点(图 3 中有两个 a)，即表示重定向的过程)发送授权请求(见 7.3.2)。第三方应用程序发送的请求中包含第三方应用程序标识符、申请的受保护资源访问范围、本地状态和第三方应用程序的重定向端点 URI(图 3 中仅标出了第三方应用程序标识符和重定向端点 URI 作为示例，通常请求中会包括更多参数)，该重定向端点 URI 用于第三方应用程序接收来自授权服务器的关于授权的结果。

b）　授权服务器(通过用户代理)鉴别资源拥有者并询问资源拥有者是否允许第三方应用程序的访问请求(图 3 中的两个 b)表示授权服务器通过资源拥有者的用户代理鉴别资源拥有者的过程)。

c）　假定资源拥有者同意了此次访问，授权服务器使用第三方应用程序在注册时提供的重定向端点 URI(该重定向端点 URI 是图 3 中的第三方应用程序 Web 资源端点 URI)向资源拥有者的用户代理发送重定向请求，并在重定向端点 URI 的片段组件包含访问令牌。

d）　用户代理根据重定向请求，使用重定向端点 URI(不包括片段组件)向第三方应用程序的 Web 资源发送请求。用户代理将重定向端点 URI 中的片段组件保留在本地。

e）　第三方应用程序的 Web 资源返回网页(通常是带有内嵌脚本的 HTML 文档)，该网页能够读取用户代理本地保留的完整的重定向 URI，并能够把片段组件所包含的访问令牌(以及其他参数)提取出来。

f）　资源拥有者的用户代理在本地执行步骤 e)中返回的脚本，用于提取访问令牌。

g）　步骤 e)中返回的脚本使用用户代理将提取出来的访问令牌发送给第三方应用程序。

隐私许可流程中，应进行如下安全考虑：

a）　隐式许可流程不包含对第三方应用程序的身份鉴别流程。在某些特定情况下，授权服务器可验证第三方应用程序的身份。例如授权服务器通过验证请求中的重定向端点 URI 与第三方应用程序注册时提供的重定向端点 URI 是否一致，从而验证第三方应用程序的身份。

b）　由于授权服务器所发放的访问令牌会被编码到重定向端点 URI 中，因此访问令牌可能暴露给资源拥有者或其他能够访问资源拥有者用户代理的应用程序。

c) 由于隐式许可流程减少了使用授权码获取访问令牌的环节，因此该流程可提升某些第三方应用程序（例如浏览器内嵌的应用程序）的响应能力和效率。然而，若能使用授权码许可流程和隐式许可流程等多类流程，应在隐式许可流程带来的性能提升和其带来的安全性问题等两方面进行权衡。

7.3.2 授权请求

第三方应用程序通过将以下参数添加到授权端点 URI 的查询组件，来构造授权请求的 URI：

a) 〈response_type〉[必选]

参数值应为“token”；

b) 〈client_id〉[必选]

6.2 中所描述的第三方应用程序标识符；

c) 〈redirect_uri〉[可选]

如 7.2.2 所述；

d) 〈scope〉[可选]

如 7.2.2 所述；

e) 〈state〉[推荐]

即 7.2.1 步骤(a)中的本地状态。开发者应确保该值的不可猜测性。授权服务器在将用户代理重定向回第三方应用程序时，在重定向请求中包含此值。第三方应用程序开发者宜使用该参数，以防止跨站点请求伪造攻击。

授权服务器应验证授权请求的有效性，以确保所有必选参数都存在且有效。如果验证请求有效，授权服务器将对资源拥有者的身份进行鉴别并从资源拥有者处获得授权决定（通过直接询问资源拥有者的方式或其他方式）。

7.3.3 访问令牌响应

7.3.3.1 正确响应

如果资源拥有者允许本次授权请求，授权服务器应给第三方应用程序发放访问令牌。授权服务器向重定向端点 URI 的片段组件添加如下参数（参数经过 UTF-8 编码后，再使用“application/x-www-form-urlencoded”格式编码）：

a) 〈access_token〉[必选]

授权服务器发放的访问令牌；

b) 〈token_type〉[必选]

8.1 描述的令牌类型；

c) 〈expires_in〉[推荐]

访问令牌的生命周期，以秒为单位。例如，“3600”表示该访问令牌将在响应发出 1 小时后失效。如果本参数被省略，授权服务器应通过其他方式提供失效时间，或公布缺省值；

d) 〈scope〉

如果该参数的值与第三方应用程序授权请求中的〈scope〉参数值相同，则该参数是可选的；其他情况下，该参数是必选的；

e) 〈state〉

如果第三方应用程序的授权请求中含有此参数，则授权响应中也应包含此参数。该参数的值与第三方应用程序授权请求中的〈state〉值相同。

在隐式许可类型的流程中，授权服务器不得发放刷新令牌。

第三方应用程序应忽略未识别的响应参数。本标准对授权码的字符串长度不做定义。第三方应用程序不宜对字符串长度做出假定。授权服务器应在其文档中说明其发放所有参数值的长度。

7.3.3.2 出错响应

如果由于重定向端点 URI 丢失、无效或不匹配，或由于第三方应用程序标识符丢失或无效等原因出错，授权服务器应当告知资源拥有者这一错误，并且不得自动将用户代理重定向至无效的重定向端点 URI。

如果由于资源拥有者拒绝访问请求或是由于重定向端点 URI 有误之外的其他原因导致未授权，授权服务器将通过向重定向端点 URI 的片段组件添加如下参数的方式通知第三方应用程序出错原因：

a) 〈error〉[必选]

ASCII 码格式的错误代码，错误代码类型如下所示：

1) invalid_request 表示请求中丢失了必选参数，存在无效参数，或重复包含了某个参数，或是其他形式的格式错误；
2) unauthorized_client 表示该第三方应用程序无权使用当前方法请求授权码；
3) access_denied 表示资源拥有者或授权服务器拒绝了访问请求；
4) unsupported_response_type 表示授权服务器不支持授权码许可类型；
5) invalid_scope 表示所请求的资源访问范围无效、未知或是形式不当；
6) server_error 表示授权服务器遇到了意外情况从而不能响应该请求(该错误代码是需要的，因为 HTTP 的 500 内部服务器错误代码不能通过 HTTP 重定向返回)；
7) temporarily_unavailable 表示由于暂时的过载或是服务器维护，授权服务器目前无法处理该请求(该错误代码是需要的，因为 HTTP 的 503 服务错误代码不能通过 HTTP 重定向返回)；

〈error〉参数的值不应包含％x20-21/％x23-5B/％x5D-7E 集合之外的字符。

b) 〈error_description〉[可选]

终端用户可读的 ASCII 文本，提供附加信息以帮助第三方应用程序的开发者理解出现的错误。

c) 〈error_uri〉[可选]

用于标识网页，该网页含有终端用户可读的、关于该错误更多信息。该参数的值应遵循 URI-reference 语法，并且不得包含％x20-21/％x23-5B/％x5D-7E 集合之外的字符。

d) 〈state〉

如果第三方应用程序的授权请求中含有此参数，则响应中也应包含此参数。响应中该参数的值与授权请求中的值相同。

7.4 资源拥有者口令凭据许可流程

7.4.1 协议流程

资源拥有者口令凭据许可类型流程见图 4，包括以下步骤：

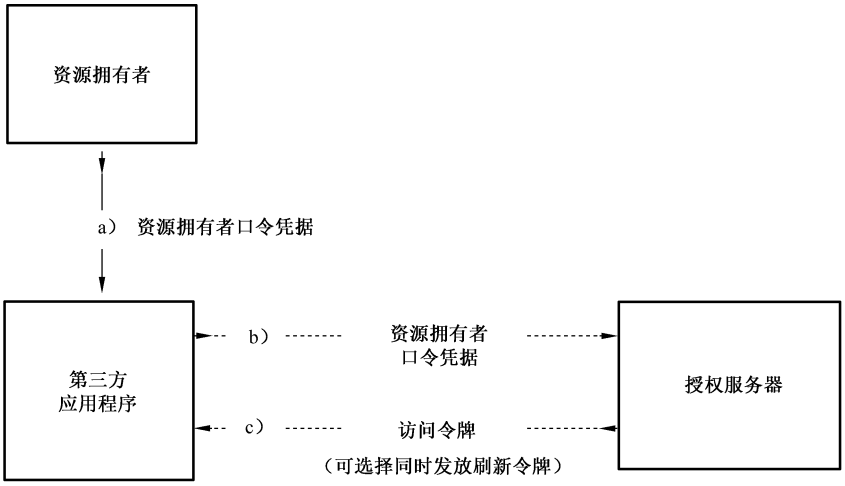

图 4 资源拥有者口令凭据许可流程

a） 资源拥有者向第三方应用程序提供其用户名和口令；

b） 第三发应用程序向授权服务器提供资源拥有者的用户名和口令，请求访问令牌。发送请求时，第三方应用程序应鉴别授权服务器的身份；

c） 授权服务器鉴别第三方应用程序的身份并验证资源拥有者用户名口令的有效性，如果有效，授权服务器发放访问令牌。

在资源拥有者口令凭据许可流程中，应满足以下安全要求：

a） 资源拥有者的口令凭据仅在换取访问令牌时使用。第三方应用程序无需存储资源拥有者的口令凭据。该授权许可类型一定程度上降低了第三方应用程序存储用户名和口令的风险，但没有降低将高特权凭据暴露给第三方应用程序的风险。

b） 由于资源拥有者不能控制授权流程（当资源拥有者把口令凭据交给第三方应用程序时，资源拥有者的参与即结束），第三方应用程序可能获得比资源拥有者要求的更大受保护资源访问范围的访问令牌。授权服务器应根据授权许可类型来设定访问令牌的受保护资源访问范围和生命周期。

c） 授权服务器应当谨慎采用此种类型的授权许可，只有当资源拥有者与第三方应用程序之间高度互信且无法采用其他类型的授权许可流程时，才允许使用该流程。

7.4.2 授权请求和授权响应

第三方应用程序获取资源拥有者口令凭据的方式不在本标准的讨论范围内。第三方应用程序在获得访问令牌后应销毁资源拥有者口令凭据。

7.4.3 访问令牌请求

第三方应用程序向令牌端点发送请求，在该请求的主体部分添加如下参数（参数经过 UTF-8 编码后，再使用“application/x-www-form-urlencoded”格式编码）：

a） 〈grant_type〉[必选]

参数值应为“password”；

b） 〈username〉[必选]

资源拥有者的用户名；

c） 〈password〉[必选]

资源拥有者的口令；

d) 〈scope〉[可选]

7.2.2 所描述的请求访问受保护资源的范围。

如果第三方应用程序的类型为有保密能力型，或该第三方应用程序被发放了第三方应用程序身份凭据(或被其他鉴别要求所约束)，该第三方应用程序应如 7.3.2 所述被授权服务器鉴别身份。

授权服务器应：

——要求对有保密能力型的第三方应用程序或任何被发放过第三方应用程序身份凭据(或被其他鉴别要求所约束)的第三方应用程序进行身份鉴别；

——如果需要鉴别该第三方应用程序，则执行鉴别流程；

——用口令验证算法验证资源拥有者的口令凭据。

由于访问令牌请求使用了资源拥有者的口令凭据，授权服务器应使令牌端点具有抵御暴力攻击的能力(例如，使用速度限制或产生告警等)。

7.4.4 访问令牌响应

如果访问令牌请求是有效的，授权服务器应按照 8.2.2 所述发放访问令牌和刷新令牌(可选的)。如果授权服务器对发起访问令牌请求的第三方应用程序的身份鉴别失败，或对访问令牌请求的验证失败，授权服务器应返回 8.2.3 所述的出错响应。

7.5 第三方应用程序身份凭据许可流程

7.5.1 协议流程

第三方应用程序身份凭据许可流程见图 5，包括以下步骤：

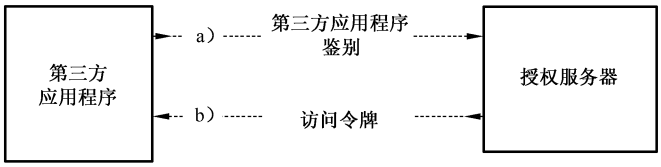

图 5 第三方应用程序凭据许可流程

a) 第三方应用程序对授权服务器进行身份鉴别，并向授权服务器的令牌端点发送访问令牌请求；

b) 授权服务器鉴别第三方应用程序的身份，如果身份鉴别通过，则发放访问令牌。

在第三方应用程序身份凭据许可流程中，应满足以下安全要求：

a) 授权服务器应要求第三方应用程序提前注册，注册完成后，授权服务器将生成的第三方应用程序身份凭据发放给第三方应用程序。第三方应用程序的身份凭据应能抵抗猜测攻击。

b) 仅有保密能力型的第三方应用程序能够使用第三方应用程序身份凭据许可类型。

7.5.2 授权请求和响应

第三方应用程序鉴别结果被用作授权许可，无需额外的授权请求。

7.5.3 访问令牌请求

第三方应用程序向令牌端点发送请求，在该请求的主体部分添加如下参数(参数经过 UTF-8 编码后，再使用“application/x-www-form-urlencoded”格式编码)：

a) 〈grant_type〉[必选]

参数值应为“client_credentials”；

b) 〈scope〉[可选]

7.2.2 所描述的请求访问受保护资源的范围。

授权服务器应如 6.4 所述鉴别第三方应用程序的身份。

7.5.4 访问令牌响应

如果访问令牌请求是有效的，授权服务器按照 8.2.2 所述发放访问令牌，响应中不得包含刷新令牌。如果授权服务器对发起访问令牌请求的第三方应用程序的身份鉴别失败，或对第三方应用程序请求的验证失败，则授权服务器返回 8.2.3 所述的出错响应。

8 令牌

8.1 令牌类型

8.1.1 访问令牌

访问令牌是授权服务器发放给第三方应用程序用于访问受保护资源的凭据。访问令牌使用字符串表示，表明第三方应用程序拥有了资源拥有者的授权。访问令牌对于第三方应用程序而言通常是不易解读的。访问令牌中给出了受保护资源的访问范围和访问有效期，访问范围和访问有效期由资源拥有者授权同意，并由资源服务器和授权服务器实施。

访问令牌可作为提取授权信息的标识符；也可自身包含授权信息，访问令牌中包含的授权信息可通过某种方式得到验证（例如，包含数据和签名）。本标准要求授权服务器应先采用 SM3 算法（见 GB/T 32905—2016）对访问令牌的内容进行杂凑运算，再使用 SM2 算法（见 GB/T 32918.2—2016），最后使用 SM4 算法（见 GB/T 32907—2016），将最后签名加密处理过的令牌发送出去。访问令牌包含了第三方应用程序请求受保护资源所需的必要信息。

通常每一种访问令牌类型都应有对应的规范文档。如果第三方应用程序无法理解某访问令牌的类型，第三方应用程序不得使用该访问令牌。在访问令牌类型的定义中，应规定访问令牌响应中除8.2.2中定义的参数外的其他参数，同时应规定访问令牌的验证方法。

在传输和存储访问令牌时，应保证访问令牌（以及其他机密的访问令牌属性）的机密性，并只在授权服务器、访问令牌规定范围的资源所在的资源服务器和访问令牌对应的第三方应用程序等三者中共享。访问令牌的传输应使用 GM/T 0024—2014 定义的安全传输层协议。

当使用隐式许可流程进行授权时，访问令牌在 URI 片段中传输，但该传输方式可能将访问令牌暴露给未授权方。授权服务器应确保未授权方不能伪造和修改访问令牌，并且难以通过猜测访问令牌以产生有效的访问令牌。

第三方应用程序应根据需求请求最低的受保护资源访问范围的访问令牌。授权服务器应根据第三方应用程序的身份，确定访问令牌的受保护资源访问范围，该访问范围不应高于第三方应用程序请求的受保护资源访问范围。

本标准不规定资源服务器验证访问令牌有效性的方法。

为了防止访问令牌猜测攻击，授权服务器应保证攻击者猜测产生访问令牌的可能性应不高于 2^{-128}，宜不高于 2^{-160}。

8.1.2 刷新令牌

刷新令牌是第三方应用程序重新获取访问令牌的凭据。刷新令牌由授权服务器发放给第三方应用程序，用于在当前的访问令牌作废或是过期时换取新的访问令牌，或是换取具有同等（或更小）作用域的另一个访问令牌（访问令牌的生命周期和权限可小于资源拥有者的授权范围）。如果授权服务器决定发放刷新令牌给第三方应用程序，刷新令牌与访问令牌应同时发放。

刷新令牌使用字符串表示，代表资源拥有者的授权。刷新令牌对于第三方应用程序而言通常是不易解读的。刷新令牌是提取授权信息的标识，刷新令牌中不能包含授权信息。与访问令牌不同，刷新令牌只能用于第三方应用程序与授权服务器的交互，不应发送给资源服务器。

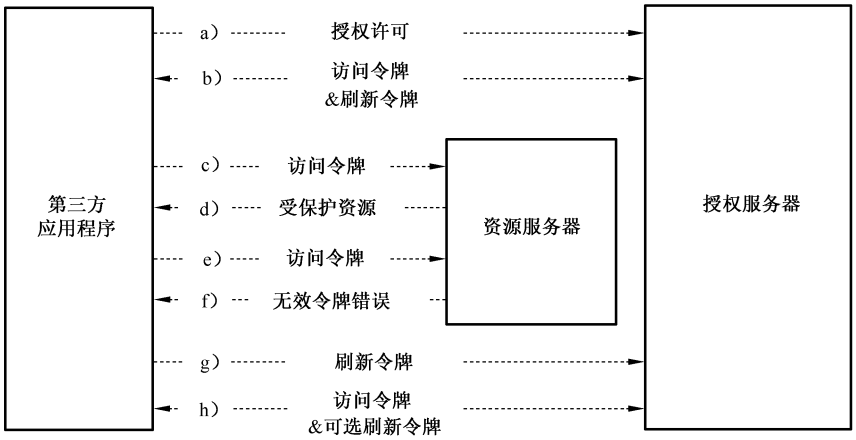

图 6　使用刷新令牌的协议流程

图 6 中所示流程具体步骤如下：

a）　第三方应用程序鉴别授权服务器的身份并呈递授权许可，以此来请求访问令牌和刷新令牌。

b）　授权服务器鉴别第三方应用程序的身份并验证授权许可的有效性。如果身份鉴别通过且授权许可有效，则发放访问令牌和刷新令牌。

c）　第三方应用程序向资源服务器呈递访问令牌，请求访问受保护资源。

d）　资源服务器验证访问令牌的有效性。如果访问令牌有效，则受理该请求。

e）　重复执行步骤 c)和 d)，直到访问令牌过期。在访问令牌过期的情况下，执行步骤 g)，否则继续请求访问受保护资源。

f）　如果访问令牌无效，资源服务器返回访问令牌无效的出错响应。

g）　第三方应用程序鉴别授权服务器的身份并呈递刷新令牌，以此来请求新的访问令牌。授权服务器是否需要再次鉴别第三方应用程序的身份取决于第三方应用程序的类型和授权服务器的策略(通常由授权服务器的服务文档提供)。

h）　授权服务器鉴别第三方应用程序的身份并验证刷新令牌的有效性，如果刷新令牌有效，则向第三方应用程序发放新的访问令牌(或者同时发放新的刷新令牌[可选的])。

本标准不对步骤 c)、d)、e)、f)的具体内容和实现进行规定。

在传输和存储刷新令牌时，应保证刷新令牌的机密性，并只在授权服务器、刷新令牌对应的第三方应用程序两者中共享。授权服务器应维护刷新令牌和接收刷新令牌的第三方应用程序之间的绑定关系。刷新令牌的传输应使用 GM/T 0024—2014 定义的安全传输层协议。

在鉴别第三方应用程序的身份时(见 6.3)，授权服务器应验证刷新令牌和第三方应用程序身份间的绑定关系。当无法鉴别第三方应用程序身份时，授权服务器应采用其他方式来检测刷新令牌是否被滥用。例如，授权服务器可采用刷新令牌环，刷新令牌环中新刷新令牌的发放对应每一个访问令牌刷新请求。过期的刷新令牌失效但仍由授权服务器保存。如果刷新令牌被盗用，且随后被攻击者和合法第三方应用程序使用，合法第三方应用程序将通过出示已失效的刷新令牌来向授权服务器证明自己的合法性。

授权服务器应确保未授权方不能构造、修改刷新令牌，同时确保未授权方不能根据对刷新令牌的猜测生成有效的刷新令牌。

为了防止刷新令牌猜测攻击，授权服务器应保证攻击者猜测产生刷新令牌的可能性应不高于

2^{-128}，宜不高于 2^{-160}。

8.2 访问令牌发放

8.2.1 访问令牌发放流程

如果访问令牌请求是有效的，授权服务器按照 8.2.2 所述发放访问令牌和刷新令牌（可选的）。如果授权服务器对发起访问令牌请求的第三方应用程序的身份鉴别失败，或对第三方应用程序的访问令牌请求验证失败，则授权服务器返回 8.2.3 所述的出错响应。

8.2.2 成功响应

授权服务器将如下参数添加到状态码是 200 的 HTTP 响应中，构造成功响应来发放访问令牌和刷新令牌（可选的）：

a) 〈access_token〉[必选]

授权服务器发放的访问令牌；

b) 〈token_type〉[必选]

8.1.1 描述的访问令牌类型；

c) 〈expires_in〉[推荐]

访问令牌的生命周期，以秒为单位。例如，“3600”表示该访问令牌将在响应发出 1 小时后过期。如果本参数被省略，授权服务器应当通过其他方式（例如公布缺省值）提供过期时间；

d) 〈refresh_token〉[可选]；

如 8.1.2 所述。

e) 〈scope〉

如果该参数的值与第三方应用程序访问令牌请求中的 scope 参数值相同，则该参数是可选的；其他情况下，该参数是必选的。

对于任何包含令牌等敏感信息的响应，授权服务器应在 HTTP 响应（见 RFC 2616）的头部中包含值为“no-store”的“Cache-Control”字段，以及值为“no-cache”的“Pragma”字段。

第三方应用程序应忽略未识别的响应参数。本标准对授权码的字符串长度不做定义。第三方应用程序不宜对字符串长度做出假定。授权服务器应在服务文档中说明其发放的所有参数值的长度。

8.2.3 出错响应

授权服务器返回状态码为 400（Bad Request）的 HTTP 响应，并在响应中包含如下参数：

a) 〈error〉[必选]

ASCII 格式的错误代码，有以下类型：

1) invalid_request 表示该请求缺失了必选参数，或包含了不被支持的参数值（但如果是授权许可是不支持的授权许可类型，返回错误码应为 invalid_grant），或重复包含了某一参数，或包含了多个凭据，或使用了超过一种的第三方应用程序身份鉴别的方式等；
2) invalid_client 表示鉴别第三方应用程序身份失败（例如，未知的第三方应用程序，或访问令牌请求过程中没有包含 6.4 定义的第三方应用程序身份鉴别，或第三方应用程序身份鉴别方式不被支持）。授权服务器可返回状态码为 401 的 HTTP 响应，用以表明支持哪些 HTTP 鉴别方案。如果第三方应用程序是通过请求头部的“Authorization”字段进行身份鉴别，但身份鉴别失败，授权服务器应返回状态码为 401 的 HTTP 响应，并在头部中包含与第三方应用程序使用的鉴别方案匹配的“www-Authentication”头字段；
3) invalid_grant 表示第三方应用程序提供的授权许可（例如，授权码，资源拥有者口令凭

据)或刷新令牌是无效的,或过期的,或被撤销的,或与授权请求中提供的重定向端点URI不匹配,或是发放给另外的第三方应用程序的;

4) unauthorized_client 表示授权服务器不允许该第三方应用程序使用当前授权许可类型;

5) unsupported_grant_type 表示授权服务器不支持当前使用的授权许可类型;

6) invalid_scope 表示所请求的访问受保护资源范围无效、未知、格式有误或是超出了资源拥有者授权的范围。

〈error〉参数的值不应包含%x20-21/%x23-5B/%x5D-7E集合之外的字符。

b) 〈error_description〉[可选]

终端用户可读的ASCII文本,提供附加信息以帮助第三方应用程序的开发者理解出现的错误。

c) 〈error_uri〉[可选]

用于标识包含有终端用户可读的、关于该错误更多信息的网页。该参数的值应遵循URI-reference语法,并且不应包含%x20-21/%x23-5B/%x5D-7E集合之外的字符。

8.3 访问令牌刷新

如果第三方应用程序接收到授权服务器发放的刷新令牌,则第三方应用程序可利用刷新令牌向令牌端点发送请求以获得新的访问令牌。第三方应用程序在访问令牌刷新请求的主体部分添加如下参数(参数经过UTF-8编码后,再使用"application/x-www-form-urlencoded"格式编码):

a) 〈grant_type〉[必选]

参数值应为"refresh_token";

b) 〈refresh_token〉[必选]

授权服务器发放给第三方应用程序的刷新令牌;

c) 〈scope〉[可选]

7.2.2所描述的请求受保护资源访问范围。请求中的受保护资源访问范围不得超出资源拥有者最初许可的受保护资源访问范围,如果该参数被省略,则缺省为资源拥有者最初许可的受保护资源访问范围。

授权服务器在发放刷新令牌时,应将刷新令牌与被发放该刷新令牌的第三方应用程序绑定。随后在接收到访问令牌刷新请求时,授权服务器应:

——对有保密能力的第三方应用程序或任何被发放过第三方应用程序身份凭据(或被其他鉴别要求所约束)的第三方应用程序进行身份鉴别;

——如果需要鉴别该第三方应用程序,则执行鉴别流程,并确保刷新令牌发放给通过身份鉴别的第三方应用程序;

——验证刷新令牌。

如果访问令牌刷新请求是有效的,授权服务器按照8.2.2所述发放访问令牌。如果对当前请求的第三方应用程序的身份鉴别失败或刷新令牌验证失败,授权服务器应返回8.2.3所述的出错响应。

授权服务器可在发放新的访问令牌的同时发放新的刷新令牌,在此种情况下,第三方应用程序应丢弃旧的刷新令牌,代之以新的刷新令牌。在向第三方应用程序发放了新的刷新令牌后,授权服务器可撤销旧的刷新令牌。如果授权服务器发放了新的刷新令牌,新发放的刷新令牌的受保护资源访问范围应与第三方应用程序在访问令牌刷新请求中的受保护资源访问范围相同。

9 受保护资源访问

9.1 受保护资源访问流程

第三方应用程序向资源服务器呈递访问令牌来访问受保护的资源。资源服务器应验证访问令牌的有效性,确保访问令牌没有过期,且访问令牌的受保护资源访问范围包括第三方应用程序请求访问的资源。资源服务器验证访问令牌的方法本标准不作规定。

9.2 成功响应

当访问受保护资源的请求成功时，资源服务器应告知第三方应用程序，并允许第三方应用程序访问受保护的资源。成功响应的详细内容本标准不作规定。

9.3 出错响应

如果访问受保护资源的请求失败，资源服务器应当告知第三方应用程序这一错误。资源服务器告知第三方应用程序请求错误的方式本标准不作规定。

开发者在设计令牌验证方案时，应定义一种向第三方应用程序返回错误状态码的机制，规范协议参与方可以理解的、达成一致的错误码及错误码值（参见 A.2）。设计的令牌验证方案可将有效的错误值集合限定为本标准所列出值的子集。如果错误码通过参数返回，参数名应为“error”。

其他现有的能够用于本标准令牌验证的方案，可将其错误值与注册表中的值进行绑定。

新设计的令牌验证方案可选择〈error_description〉和〈error_uri〉参数来返回错误信息，但不应与本标准中〈error_description〉和〈error_uri〉参数的使用互相冲突。

附 录 A
（资料性附录）
协议参数说明

A.1 OAuth 参数说明

本标准对 OAuth 参数进行说明，包括授权端点请求、授权端点响应、令牌端点请求或者令牌端点响应中的关键参数；

a） 参数名称：client_id
参数使用的地方：授权请求，令牌请求；

b） 参数名称：client_secret
参数使用的地方：令牌请求；

c） 参数名称：response_type
参数使用的地方：授权请求；

d） 参数名称：redirect_uri
参数使用的地方：授权请求，令牌请求；

e） 参数名称：scope
参数使用的地方：授权请求，授权响应，令牌请求，令牌响应；

f） 参数名称：state
参数使用的地方：授权请求，授权响应；

g） 参数名称：code
参数使用的地方：授权响应，令牌请求；

h） 参数名称：error_description
参数使用的地方：授权响应，令牌响应；

i） 参数名称：error_uri
参数使用的地方：授权响应，令牌响应；

j） 参数名称：grant_type
参数使用的地方：令牌请求；

k） 参数名称：access_token
参数使用的地方：授权响应，令牌响应；

l） 参数名称：token_type
参数使用的地方：授权响应，令牌响应；

m） 参数名称：expires_in
参数使用的地方：授权响应，令牌响应；

n） 参数名称：username
参数使用的地方：令牌请求；

o） 参数名称：password
参数使用的地方：令牌请求；

p） 参数名称：refresh_token
参数使用的地方：令牌请求，令牌响应；

q） 响应类型名称：token

参数使用的地方：令牌请求，令牌响应。

A.2 OAuth 错误码说明

错误码应该包含错误码名称和错误码值，协议应统一规范一致的错误码。

错误码名称：所请求的名称（如 example）。错误码值必须在%x20-21/%x23-5B/%x5D-7E 字符集中，字符集为 ASCⅡ码形式。

错误使用的地方：可能的地方有授权码许可出错响应（见 7.2.3.2），隐式许可出错响应（见 7.3.3.2），访问令牌发放出错响应（见 8.2.3）和受保护资源访问出错响应（见 9.3）。

相关协议扩展：与错误码一同使用的扩展许可类型、访问令牌类型或扩展参数的名称。

参 考 文 献

[1] OAuth-HTTP-MAC. Hammer-Lahav, E., Ed., "HTTP Authentication: MAC Access Authentication", Work in Progress, February 2012.

[2] OAuth-SAML2. Campbell, B. and C. Mortimore, "SAML 2.0 Bearer Assertion Profiles for OAuth 2.0", Work in Progress, September 2012.

[3] OAuth-THREATMODEL. Lodderstedt, T., Ed., McGloin, M., and P. Hunt, "OAuth 2.0 Threat Model and Security Considerations", Work in Progress, October 2012.

[4] OAuth-WRAP. Hardt, D., Ed., Tom, A., Eaton, B., and Y. Goland, "OAuth Web Resource Authorization Profiles", Work in Progress, January 2010.

[5] RFC 5849. Hammer-Lahav, E., "The OAuth 1.0 Protocol", April 2010.

[6] RFC 2246. T. Dierks,C. Allen, " The TLS Protocol Version 1.0",January 1999.