

ICS 33.030

M 30

YD

中华人民共和国通信行业标准

YD/T 2912-2015

移动互联网应用编程接口的授权技术要求

Technical requirements of authorization framework for
mobile internet API

2015-07-14 发布

2015-10-01 实施

中华人民共和国工业和信息化部 发布

目 次

前 言	III
1 范围	1
2 规范性引用文件	1
3 术语、定义和缩略语	2
3.1 术语和定义	2
3.2 缩略语	3
4 概述	3
4.1 Autho4API的制定内容	3
4.2 抽象协议流程	4
5 需求	4
5.1 概述	4
5.2 高层次功能要求	4
5.3 安全需求	5
5.4 计费	7
5.5 管理和配置	7
5.6 可用性	7
5.7 互通性	7
5.8 隐私性	7
5.9 共享多服务提供商的情况	8
6 体系结构	8
6.1 相关性/依赖性	8
6.2 架构图	8
6.3 功能组件和接口/参考点定义	9
7 技术规范	10
7.1 客户端注册	10
7.2 协议端点	11
7.3 服务参数发现(参考性信息)	12
7.4 获取授权	14
7.5 发出访问令牌	25
7.6 更新访问令牌	25
7.7 访问受保护的资源	25
7.8 多业务提供商环境(消息环境)	27
7.9 安全考虑	33

8 OMNA考虑.....34

附录A（资料性附录） 部署场景.....35

附录B（资料性附录） 范围值（Scope Values）的定义.....38

附录B（资料性附录） “ 授予资源的URL前缀” 资源的定义.....44

参考文献.....49

前 言

本标准按照 GB/T 1.1-2009 给出的规则起草。

本标准参考了 OMA 引擎 Autho4API v1.0 规范 OMA-RD-Autho4API-V1_0-20120327-C 和 OMA-ER-Autho4API-V1_0-20120327-C，与这两个规范章节和内容的对应关系如下：

- 本标准中第3章，对应于OMA规范的OMA-ER-Autho4API-V1_0-20120327-C的第3章，增加了3个定义：机密类型客户端、公开类型客户端、RESTful；
- 本标准中第4章，对应于OMA规范的OMA-ER-Autho4API-V1_0-20120327-C的第4章，技术细节保持一致；
- 本标准中第5章，对应于OMA规范的OMA-RD-Autho4API-V1_0-20120327-C的第5章，技术细节保持一致；
- 本标准中第6章，对应于OMA规范的OMA-ER-Autho4API-V1_0-20120327-C的第6章，技术细节保持一致；
- 本标准中第7章，对应于OMA规范的OMA-ER-Autho4API-V1_0-20120327-C的第7章，技术细节保持一致；
- 本标准中第8章，对应于OMA规范的OMA-ER-Autho4API-V1_0-20120327-C的第8章，技术细节保持一致；
- 本标准中附录A，对应于OMA规范的OMA-ER-Autho4API-V1_0-20120327-C的附录C，技术细节保持一致；
- 本标准中附录B，对应于OMA规范的OMA-ER-Autho4API-V1_0-20120327-C的附录D，技术细节保持一致；
- 本标准中附录C，对应于OMA规范的OMA-ER-Autho4API-V1_0-20120327-C的附录E，技术细节保持一致。

请注意本文件的某些内容可能涉及专利。本文件的发布机构不承担识别这些专利的责任。

本标准由中国通信标准化协会提出并归口。

本标准起草单位：上海贝尔股份有限公司、中国联合网络通信集团有限公司、中国信息通信研究院、中国移动通信集团有限公司、南京爱立信熊猫通信有限公司。

本标准主要起草人：胡志远、孙群英、刘 镝、张 尼、董 慧、逢淑宁、王 静、刘 斐、李 钢。

移动互联网应用编程接口的授权技术要求

1 范围

本标准规定了移动互联网应用编程接口授权框架，该框架能使用户通过开放的网络能力API，尤其是RESTFul API，授权第三方应用（桌面、移动终端与web应用），以代表访问用户访问网络资源。

本标准适用于运营商为移动互联网应用访问网络资源提供安全授权机制。

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。

IETF RFC2045	因特网邮件协议之消息体（Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies）
IETF RFC2046	互联网邮件协议之媒体类型（Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types）
IETF RFC2119	RFC中关键词的解释(Key words for use in RFCs to Indicate Requirement Levels)
IETF RFC3986	统一资源标识（Uniform Resource Identifier (URI): Generic Syntax）
IETF RFC4234	ABNF语法规则（Augmented BNF for Syntax Specifications: ABNF）
IETF RFC4346	传输层安全（Transport Layer Security (TLS) Version 1.1）
IETF RFC5246	传输层安全（Transport Layer Security (TLS) Version 1.2）
IETF RFC2616	超文本协议（Hypertext Transfer Protocol -- HTTP/1.1）
IETF RFC2617	HTTP Digest认证（HTTP Authentication: Basic and Digest Access Authentication）
IETF RFC6749（2012）	OAuth2.0授权协议（The OAuth 2.0 Authorization Protocol）
IETF RFC6750	OAuth2.0承载令牌（The OAuth 2.0 Protocol: Bearer Tokens）
IETF RFC7009	令牌撤销(Token Revocation)
FIPS AES	高级加密算法标准（The specification of Advanced Encryption Standard）
FIPS AESMode	AES块加密模式（Recommendation of Block Cipher Modes of Operation）
GSMA RCSREQ	GSMA富媒体需求（Rich Communication Suite -- RCS API Detailed Requirements 1.1）
OMA AUTHO4API_RD_10	网络API授权框架需求（Authorization Framework for Network APIs Requirements）
OMA AUTHO4API_ER_10	网络API授权框架（Authorization Framework for Network APIs）

OMA OMNA_Autho4API	OMA 命名中心 Autho4API 范围值注册 (Open Mobile Naming Authority "Autho4API Scope Values Registry)
OMA OSE	OMA业务环境 (OMA Service Environment)
OMA SCRRULES	SCR规则和流程 (SCR Rules and Procedures)
OMA SEC_CF-V1_1	应用层公共安全功能 (Security Common Functions)
W3C HTML_4.01	HTML4.01规范 (HTML 4.01 Specification)

3 术语、定义和缩略语

3.1 术语和定义

下列术语和定义适用于本文件。

3.1.1

访问令牌 Access Token

用于访问受保护资源的信任凭证。在IETF RFC 6749 (2012)的1.4中有进一步的定义。

3.1.2

授权许可 Authorization Grant

代表资源拥有者授权 (访问其受保护资源) 的一种信任凭证, Autho4API 客户端可以凭借授权许可获取一个访问令牌。

3.1.3

更新令牌 Refresh Token

获取一个新的访问令牌的信任凭证。在IETF RFC 6749 (2012)的1.5中有进一步的定义。

3.1.4

资源拥有者 Resource Owner

许可访问受保护资源的实体, 例如终端用户。

3.1.5

表述性状态转移 REST Ful

REST(Representational State Transfer表述性状态转移)是一种针对网络应用的设计和开发方式, 指的是一组架构约束条件和原则, 可以降低开发的复杂性, 提高系统的可伸缩性。满足REST定义的约束条件和原则的应用程序或设计就是 RESTful。

3.1.6

范围 Scope

Autho4API客户端请求或以访问令牌的形式分发给Autho4API客户端的授权过程中资源访问权限, 由一系列范围值构成。

3.1.7

范围值 Scope Value

在网络API资源上的一系列已定义好的授权操作。

3.1.8

机密类型客户端 Confidential Client

能安全管理其认证凭证的机密性（如实现在安全服务器上且能限制访问其认证凭证的客户端）或能使用其他方式执行客户端的安全认证的客户端。

3.1.9

公开类型客户端 Public Client

不能确保其认证凭证的机密性（如执行在资源拥有者的设备上类似如本地应用或基于browser的应用）且不能使用其他方式执行客户端的安全认证的客户端。

3.2 缩略语

下列缩略语适用于本文件。

AES	Advanced Encryption Standard	高级加密算法标准
API	Application Program Interface	应用编程接口
Autho4API	Authorization Framework for Network APIs	网络API授权框架
CBC	Cipher Block Chaining	密码块链模式
GSMA	Global System for Mobile communications Association	全球移动通信系统联盟
HTTP	HyperText Transfer Protocol	超文本传输协议
JSON	JavaScript Object Notation	javascript 对象表示
MSISDN	Mobile Subscriber ISDN Number	移动用户ISDN号码
OMA	Open Mobile Alliance	开放移动联盟
OMNA	Open Mobile Naming Authority	开放移动命名方
OS	Operating System	操作系统
RCS	Rich Communication Suite	富通信组件
SMS	Short Message Service	短消息业务
TLS	Transport Layer Security	传输层安全
URI	Unified Resource Identifier	统一资源标识
URL	Uniform Resource Locator	统一资源位置
URN	Uniform Resource Name	统一资源命名
WAC	Wholesale Applications Community	应用程序社区
XML	eXtensible Markup Language	可扩展标记语言

4 概述

4.1 Autho4API 的制定内容

OMA RESTful 网络 API 结合基于 IETF OAuth2.0 协议的通用授权框架，以通过这些 API 实现授权第三方应用访问用户的网络资源。

授权框架，可允许网络资源拥有者通过 OMA RESTful API 开放资源，可基于资源拥有者的需求授权第三方应用(包括桌面、手机和互联网应用)代表该用户访问资源。

Autho4API 基于 IETF OAuth2.0 系列规范（IETF RFC 6749, IETF RFC 6750、IETF RFC 7009 制定，此外还定义了以下部分：

- 为授权范围值定义了 OMNA 注册。
- 第二通道，即在发送授权请求响应时，可选进行 HTTP 重定向。

- 多服务提供商提供相同服务环境下的部署场景。
- 根据发布的访问令牌解析资源服务器位置。
- 一次性访问令牌。
- 在以下方面的考虑：
 - 授权范围值定义；
 - 自包含访问令牌格式；
 - 服务发现；
 - 本地应用；
 - HTTP 重定向捕获机制。

4.2 抽象协议流程

Autho4API 的抽象流程见图 1。

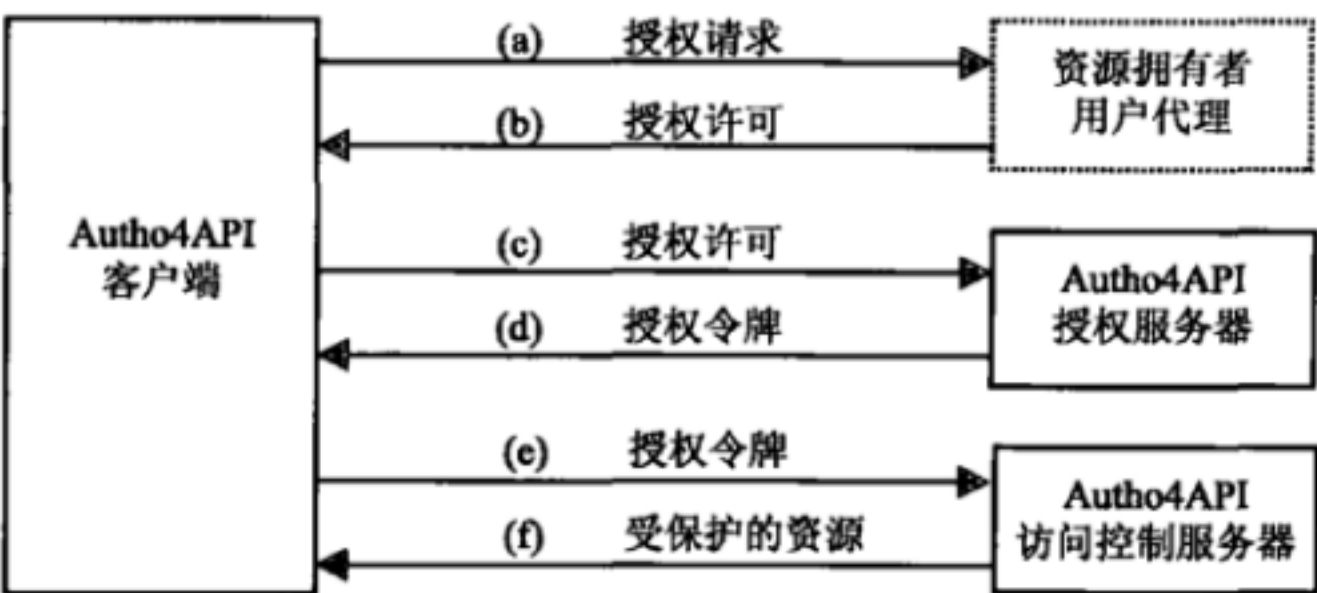


图 1 抽象协议流程

图 1 中的步骤描述如下：

- a) Autho4API 客户端请求资源拥有者的授权。
- b) Autho4API 客户端收到一个授权许可，授权许可是一个资源拥有者的授权许可凭证。
- c) Autho4API 客户端向授权服务器请求访问令牌，提交授权许可并请求 Autho4API 授权服务器认证。
- d) Autho4API 授权服务器认证 Autho4API 客户端，并验证授权许可，如果验证成功则为 Autho4API 客户端发布访问令牌。
- e) Autho4API 客户端向 Autho4API 访问控制服务器请求访问受保护的资源，并提交访问令牌。
- f) Autho4API 访问控制服务器验证访问令牌，如果验证成功则响应请求。

5 需求

5.1 概述

本节描述了有关引擎 Autho4API 的要求。该要求的主要部分来源于 GSMA 的 RCS API 中授权相关的要求，可参见文档 GSMA RCSREQ。

OMA 引擎 Autho4API 涵盖了参考授权的相关要求，除此之外致力于提供一个更通用的授权方法，以适用于广泛的 RESTful 网络 API。

为构建该框架，本文参考了 GSMA RCSREQ 中有关授权的可适用要求，还定义了附加要求。

5.2 高层次功能要求

本节描述的是有关引擎 Autho4API 的高层次功能要求，见表 1。

表 1 高层次功能要求

标号	描述	版本
Autho4API-HLF-001	授权框架不可阻止应用实例访问同一资源所有者的多个资源	Autho4API V1.0

5.3 安全需求

5.3.1 认证

(用户、应用或开发者有关的) 认证超出了引擎 Autho4API 的范围。

5.3.2 授权

5.3.2.1 通用授权

本节描述的是有关引擎 Autho4API 的通用授权要求。

应用于该引擎的 GSMA RCSGSMA RCSREQ 中的通用授权要求见表 2。

表 2 通用授权要求——来自 GSMA RCS

标号	描述	版本
Autho4API-RCAZ-001	授权框架应使拥有RESTful API公开的网络资源的用户, 能够授权第三方应用, 通过该RESTful API, 代表该用户访问这些资源	Autho4API V1.0
Autho4API-RCAZ-002	授权框架应支持网络端Web应用, 该应用通过用户的Web浏览器访问	Autho4API V1.0
Autho4API-RCAZ-003	授权框架应该支持客户端独立的安装在用户终端的Widget应用, 并支持在Web浏览器外运行	Autho4API V1.0
Autho4API-RCAZ-004	授权框架应该支持安装在用户终端的客户端本地应用	Autho4API V1.0
Autho4API-RCAZ-005	授权框架应通过引导用户到服务提供商门户, 来支持第三方应用启动授权请求	Autho4API V1.0
Autho4API-RCAZ-006	授权框架应使资源所有者能够对每一个请求的资源 and 业务操作进行授权或者拒绝访问	Autho4API V1.0
Autho4API-RCAZ-007	在用户授权第三方应用访问用户资源的情况下, 授权框架应能够向第三方应用提供代表用户授权的访问令牌	Autho4API V1.0
Autho4API-RCAZ-008	访问令牌应仅能由第三方应用用于限定的范围(资源操作), 该范围是由用户在有授权请求时进行授权	Autho4API V1.0
Autho4API-RCAZ-009	授权框架应支持在向RESTful API公开的资源发送请求时, 包含一个访问令牌(例如, 为了该请求的范围, 由第三方应用从服务提供商处获得)	Autho4API V1.0
Autho4API-RCAZ-010	发送到第三方应用的通知, 应在发放给第三方应用的授权的基础上, 进行过滤, 因此服务器不可对没有授权的应用发放通知	Autho4API V1.0

此外, OMA 还增加了一些通用授权要求, 见表 3。

表 3 通用授权请求

标号	描述	版本
Autho4API-AZ-001	依据请求的类型(例如计费), 授权框架应支持一次性访问令牌	Autho4API V1.0
Autho4API-AZ-002	授权框架应验证从第三方应用接收的访问令牌, 确保令牌没有过期并且其范围涵盖所请求的资源	Autho4API V1.0
Autho4API-AZ-003	授权框架可在资源所有者授权第三方应用访问他/她的网络资源之前, 方便地向资源所有者呈现第三方应用的可信赖性	Autho4API Future versions
Autho4API-AZ-004	授权框架应该能够告知第三方应用关于所分发的访问令牌的失效时间	Autho4API V1.0
Autho4API-AZ-005	授权框架应该支持访问令牌以特定格式进行分发和处理, 以及由第三方应用作为不透明信息处理。在这种特定的访问令牌格式中, 应该尽可能省略影响用户隐私的参数	Autho4API V1.0

5.3.2.2 OAuth 2.0 的特定授权

本节描述了有关引擎 Autho4API 的 OAuth 2.0 的特定授权要求。

应用于该引擎的 GSMA RCS RCSREQ 中的特定授权要求，见表 4。

表 4 OAuth 2.0 的特定授权要求 - 来自 GSMA RCS

标号	描述	版本
Autho4API-RCOAZ-001	正如 IETF RFC 6749 所指明的，授权框架应基于 IETF OAuth 2.0 构建	Autho4API V1.0
Autho4API-RCOAZ-002	授权框架应支持 OAuth 2.0 中的“授权代码工作流程”，在这里第三方应用是一个服务器端的网络应用	Autho4API V1.0
Autho4API-RCOAZ-003	授权框架应支持 OAuth 2.0 中的以下情况，第三方应用的类型可以是客户端安装的 Widget 应用或客户端的本地应用	Autho4API V1.0
Autho4API-RCOAZ-004	对于授权代码（“授权代码工作流程”）/访问令牌（“隐式授权工作流程”）向客户端安装的应用（Widget 或本地应用）的传递，授权框架应支持至少一种操作系统不可知和应用类型不可知的传送机制，这种机制不要求终端用户交互，比如人工输入授权代码	Autho4API V1.0
Autho4API-RCOAZ-005	不同于“授权代码工作流程”，授权框架可支持 OAuth 2.0 工作流程	Autho4API V1.0
Autho4API-RCOAZ-006	授权框架应支持 OAuth 2.0 中“授权服务器”和“资源服务器”的角色功能	Autho4API V1.0
Autho4API-RCOAZ-007	授权框架应把用户从 RESTful API 获取的资源作为 OAuth 2.0 的“被保护资源”	Autho4API V1.0
Autho4API-RCOAZ-008	当遵循授权代码工作流程时，每当用户授权时，授权框架应产生一个 OAuth 2.0 授权代码	Autho4API V1.0
Autho4API-RCOAZ-009	根据 OAuth 2.0，授权框架应支持访问令牌的授权代码的交互	Autho4API V1.0
Autho4API-RCOAZ-010	授权框架应将已验证用户的身份与已生成的授权代码和访问令牌进行绑定	Autho4API V1.0
Autho4API-RCOAZ-011	授权框架应能够通过从应用中接收访问令牌，来确定用户的身份（例如，MSISDN）	Autho4API V1.0
Autho4API-RCOAZ-012	根据 OAuth 2.0，授权框架应验证从应用中接收到的访问令牌	Autho4API V1.0

另外，OMA 还增加了一些特定授权要求，见表 5。

表 5 OAuth 2.0 的特定授权要求

标号	描述	版本
Autho4API-OAZ-001	不管那些用于规定 OAuth 2.0 访问令牌范围的值在何处定义，都应遵从 IETF RFC 6749	Autho4API V1.0
Autho4API-OAZ-002	不管那些用于规定 OAuth 2.0 访问令牌范围的值在何处定义，只要它们的定义与 IETF RFC 6749 保持一致性，授权框架就应能够接收和处理这些值	Autho4API V1.0
Autho4API-OAZ-003	授权框架应该制定指导方针或促进机制，用以管理那些用于规定 OAuth 2.0 访问令牌范围的值，从而支持这些通过标准定义的值以及由服务供应商定义的作为标准值扩展的值	Autho4API V1.0

5.3.3 数据完整性

数据完整性不属于本标准规定的内容。

5.3.4 机密性

凭证（比如，访问令牌）可以使用纯文本格式，并以 HTTP 请求和应答的方式传送。IETF OAuth2.0 见 IETF RFC 6749 中的要求在传输访问令牌时应支持 TLS。

应用于引擎 Autho4API 的机密性要求，见表 6。

表6 机密性要求

标号	描述	版本
Autho4API-CONF-001	当权限授予和访问令牌以明文传输时, 授权框架应支持其机密性保护	Autho4API v1.0
Autho4API-CONF-002	授权框架可支持在第三方应用和授权框架间传输的用户信息(例如, 用户email地址、MSISDN等)的机密性保护	Autho4API v1.0

5.4 计费

计费不属于本标准规定的内容。

5.5 管理和配置

本节描述了有关引擎 Autho4API 的管理和配置要求。

应用于本标准的 GSMA RCS 见 GSMA RCSREQ 中的要求, 见表 7。

表7 管理和配置要求——来自 GSMA RCS

标号	描述	版本
Autho4API-RCADM-001	授权框架应允许第三方应用从服务提供商(例如, 通过提取或动态发现)获得所需的参数, 用于请求用户授权和访问用户的网络资源	Autho4API V1.0
Autho4API-RCADM-002	授权框架可使资源所有者能够规定他的/她的权限授予的持续时间	Autho4API V1.0
Autho4API-RCADM-003	授权框架应该便于以下可能性, 检索先前已被授权的第三方应用的名单和哪些资源已被用户授权于每个第三方应用	Autho4API V1.0
Autho4API-RCADM-004	授权框架应该便于以下可能性, 让用户删除任何先前已被授权的第三方应用的权限	Autho4API V1.0

5.6 可用性

本节描述了有关引擎 Autho4API 的可用性要求。

应用于本标准中的 GSMA RCS 见 GSMA RCSREQ 中的要求, 见表 8。

表8 可用性要求——来自 GSMA RCS

标号	描述	版本
Autho4API-RCUSE-001	授权框架应支持以一个明确的授权对话框或用户同意请求的形式, 向资源所有者呈现第三方应用的授权请求	Autho4API V1.0
Autho4API-RCUSE-002	授权框架应该便于向资源所有者至少呈现第三方应用的身份、资源和这些请求授权资源上的操作	Autho4API V1.0
Autho4API-RCUSE-003	授权框架应该便于呈现资源所有者的首选语言和终端能力	Autho4API V1.0

5.7 互通性

不属于本标准规定的内容。

5.8 隐私性

本节描述了有关引擎 Autho4API 的隐私要求。

应用于本标准中的 GSMA RCS 见 GSMA RCSREQ 中的要求, 见表 9。

表9 隐私要求——来自 GSMA RCS

标号	描述	版本
Autho4API-RCPRV-001	授权框架不能将用于向服务供应商进行认证的的用户认证凭证透漏给第三方应用	Autho4API V1.0

另外, OMA 还增加了以下隐私要求, 见表 10。

表 10 隐私要求

标号	描述	版本
Autho4API-PRV-001	授权框架能将用户用来向服务供应商进行认证的用户身份信息（例如，MSISDN）透漏给第三方应用	Autho4API V1.0

5.9 共享多服务提供商的情况

本节描述了满足以下情况的要求，该情况下一个或一套 RESTful 网络 API 由多个服务提供商以共享的方式提供。在该部分内容中，共享的意思是应用开发者是与一个单独的整体建立关系，而不是和每个服务提供商分别建立关系。

应用到本标准中的相应要求，见表 11。

表 11 共享多服务提供商的要求

标号	描述	版本
Autho4API-MSP-001	授权框架应该支持客户端应用在多个服务提供商公开相同的RESTful网络API的情况下，无缝接入到特定的服务提供商公开的RESTful网络API，而无需客户端应用了解所选择的服务提供商的细节	Autho4API V1.0
Autho4API-MSP-002	在AUTHO4API-MSP-001所描述的情况中，应用应该能够获取访问权限，接着访问受保护的RESTful网络API资源，其中采用的流程与以下情况相同，在该情况中，应用单独处理服务提供商公开的RESTful网络API	Autho4API V1.0
Autho4API-MSP-003	在AUTHO4API-MSP-001描述的情况中，由于隐私规定的原因，在不知道资源者所有者的服务提供商的情况下，应用可能访问RESTful网络API	Autho4API V1.0
Autho4API-MSP-004	授权框架不应阻止特定的服务提供商依据其政策执行认证和令牌发放	Autho4API V1.0

6 体系结构

6.1 相关性/依赖性

Autho4API v1.0 引擎依赖于其他 OMA 引擎和扩展实体，包括以下内容：

- IETF OAuth 2.0 授权协议见 IETF RFC 6749：强制支持 OAuth2.0 获取访问令牌；
- IETF OAuth 2.0 协议：强制支持 OAuth2.0 承载访问令牌 IETF RFC 6750 的使用；
- IETF 令牌撤销见 IETF RFC 7009：可选支持 OAuth2.0 访问令牌和更新令牌的撤销；
- IETF OAuth 2.0 用户体验扩展：可选增强授权请求用户体验；
- OMA 推送见 OMAPUSH：可选支持通过第二通道“OMA 短信无连接推送”发送授权请求响应。

6.2 架构图

授权框架的架构图，见图 2。

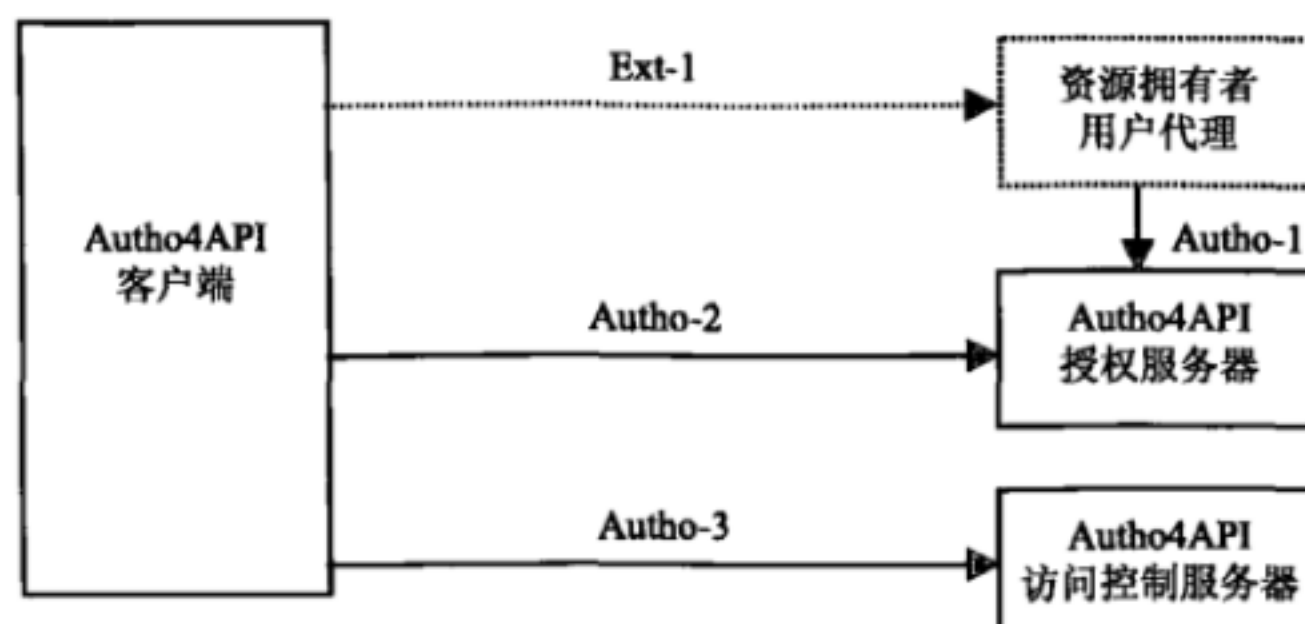


图 2 授权框架的架构

6.3 功能组件和接口/参考点定义

6.3.1 功能组件

6.3.1.1 内部功能组件

6.3.1.1.1 Autho4API 授权服务器

Autho4API 授权服务器的职责在于发放、验证和撤销访问令牌，使 Autho4API 客户端在资源拥有者的授权下，访问响应的网络资源。

Autho4API 授权服务器应能够断定访问令牌的有效性和正确性。

Autho4API 授权服务器应能够识别对资源拥有者的认证，认证应在授权过程之前进行。

必要时，Autho4API 授权服务器应在访问令牌发放或撤销之前认证 Autho4API 客户端。

注：Autho4API 授权服务器在授权过程之前认证资源拥有者的机制，不在本标准规范定义范围之内。

必要时(例如，在多服务提供商特定共享环境下)，本服务器应能够将授权请求和访问令牌请求路由到服务提供商实际指定的 Autho4API 授权服务器。

必要时，为了后续执行到服务提供商实际指定的 Autho4API 授权服务器的路由操作，本服务器应能够缓存授权码、访问令牌，以及与服务提供商指定的 Autho4API 授权服务器的关联。此外，它应能够指出到此授权服务器的重定向 URI，以便重定向资源拥有者用户代理回到本服务器，并能够缓存授权码或访问令牌。

Autho4API 授权服务器应能够与 Autho4API 客户端建立和使用安全通道，以保护他们之间交换的密钥信息的机密性。

6.3.1.1.2 Autho4API 访问控制服务器

Autho4API 访问控制服务器的职责在于，基于包含在资源访问请求中的访问令牌的有效性/正确性，对资源拥有者开放的受保护资源进行访问控制。Autho4API 访问控制服务器检查访问令牌有效性，例如在后端与 Autho4API 授权服务器协作。协作机制不在本标准规范定义范围之内。

必要时(例如，在多服务提供商特定共享环境下)，本服务器应能够将资源访问请求路由到服务提供商实际指定的访问控制服务器。

Autho4API 访问控制服务器应能够与 Autho4API 客户端建立和使用安全通道，以保护他们之间交换的密钥信息(例如，访问令牌)的机密性和防重放攻击。

6.3.1.1.3 Autho4API 客户端

Autho4API 客户端的职责在于：

- 在资源拥有者的授权下，获取访问令牌，以便能够访问资源拥有者的受保护的受保护的网络资源；
- 需要时，使用获得的访问令牌，访问资源拥有者的受保护资源；
- 在 Autho4API 授权服务器和 Autho4API 访问控制服务器之间建立和使用安全通道，以保护他们之间交换的密钥信息(例如，访问令牌、客户端密钥)的机密性和抵抗重放攻击。

6.3.1.2 外部功能组件(参考性信息)

对于明确请求资源拥有者授权的流程，资源拥有者用户代理充当 Autho4API 客户端的代理，将资源拥有者定向到 Autho4API 授权服务器，以进行对资源拥有者的认证和对 Autho4API 客户端的授权。资源拥有者用户代理模块是在认证和授权步骤中主要涉及的功能模块。

当授权码或访问令牌由 Autho4API 授权服务器通过 Autho-1 接口签发后，资源拥有者用户代理，能

够将授权许可或访问令牌发送给 Autho4API 客户端。

6.3.2 接口

6.3.2.1 接口 Autho-1

接口 Autho-1 由 Autho4API 授权服务器开放, 可以用于从使用用户代理的资源拥有者获得授权, 从而在其授权下访问资源拥有者的受保护的网路资源。获得的授权以授权许可或访问令牌的形式返回。

Autho-1 还能在授权流程之前认证资源拥有者。认证机制不在本标准定义范围之内。

Autho-1 还能向资源拥有者发送信息, 包括请求者的信息、被请求资源的信息和对资源请求操作的信息。

注: 此接口对应于 OAuth2.0 授权端点, 在 IETF RFC 6749 中定义。

6.3.2.2 接口 Autho-2

接口 Autho-2 由 Autho4API 授权服务器开放, 可以用于使用授权许可交换访问令牌和更新令牌(可选), 或者使用更新令牌交换访问令牌和更新令牌(可选)。

注: 在这个角色中, 此接口对应于 OAuth2.0 令牌端点, 在 IETF RFC 6749 中定义。

Autho-2 还能够用于, 撤销访问令牌和更新令牌。在这个角色中, 此接口对应于 OAuth2.0 令牌撤销端点, 在 IETF RFC7009 中定义。

Autho-2 还能通过特定访问令牌得到授权的资源的 URL 前缀。

Autho-2 应支持防重放攻击, 和保护 Autho4API 客户端和授权服务器之间交换的信息(例如, 访问令牌、客户端秘密)的机密性。

6.3.2.3 接口 Autho-3

接口 Autho-3 由 Autho4API 访问控制服务器开放, 可以用于在资源拥有者授权下访问受保护资源, 使用访问令牌作为授权凭证。

Autho-3 应支持防重放攻击, 和保护 Autho4API 客户端和访问控制服务器之间交换的信息(例如, 访问令牌)的机密性。

7 技术规范

7.1 客户端注册

7.1.1 客户端类型

Autho4API 客户端应为机密类型(例如, Web 服务器上执行的 Web 应用程序)或公开类型(例如, 本地应用程序), 类型在 IETF RFC 6749 的 2.1 节中定义。

Autho4API 客户端应在 Autho4API 授权服务器注册, 以发现根据客户端注册信息(例如, 客户端名称、重定向端点)发布的特定客户端标识符。此外, 机密类型的 Autho4API 客户端应从 Autho4API 授权服务器获取客户端认证凭证(例如, 客户端密钥), 以用于客户端认证。

本标准不考虑本地应用程序的安全。

7.1.2 客户端认证

机密类型的 Autho4API 客户端拥有一个客户端密钥, 可以使用 HTTP 基础认证机制来向 Autho4API 授权服务器进行认证, 定义见 IETF RFC 6749 (2012) 的 2.3.1。

Autho4API 授权服务器可以支持任何其他合适的满足安全需求的 HTTP 认证机制。本标准不详细考虑这些认证机制。

7.1.3 未注册的客户端

本标准不排除使用未注册的客户端，此类客户端定义见 IETF RFC 6749 的 2.4 节。但是，本标准不考虑此类客户端的使用。

7.2 协议端点

7.2.1 授权端点

7.2.1.1 概述

授权端点在 IETF RFC 6749 的 3.1 节中描述。授权端点嵌入在 Autho4API 授权服务器中，用于与资源拥有者进行交互，并通过 Autho-1 接口获取授权许可。

Autho4API 授权服务器能够在授权过程之前认证资源拥有者，本标准不定义所使用的认证机制。

Autho4API 授权服务器应支持 TLS 1.1（见 IETF RFC4346）和 TLS 1.2（见 IETF RFC5246）。

为了保障第二通道安全，需要考虑额外的传输层机制及其安全性。第二通道用于从授权端点向 Autho4API 客户端传输授权许可（见 7.4 节定义）。

7.2.1.2 重定向端点

IETF RFC 6749 中 3.1.2 节描述，Autho4API 授权服务器重定向资源拥有者用户代理回到 Autho4API 客户端重定向端点。例如，在客户端注册过程中，在 Autho4API 授权服务器中注册的绝对路径 URI。

Autho4API 客户端能够接收 HTTP 重定向响应（例如，对应客户端来说，重定向端点即是一个公共 HTTP URL）：

- 应在重定向端点要求使用 TLS 1.1（见 IETF RFC4346）或 TLS 1.2（见 IETF RFC5246）；
- 应支持在 3xx 范围内的 HTTP 状态码（特别包括，但不仅仅包括“302 Found”状态码）。

为了支持本地应用程序客户端，Autho4API 授权服务器可以支持非绝对路径 URI，也可以支持其他可选通道用于 Autho4API 客户端接收授权码，详见本标准 7.4.6 节的描述。

7.2.1.3 端点扩展

在由 Autho-1 接口发送的授权请求中，Autho4API 引擎能够特别支持授权端点扩展参数：

— IETF 用户体验扩展规范中定义的“显示”和“语言”参数。Autho4API 客户端和 Autho4API 授权服务器可以支持此扩展：

— 通过第二通道（在 7.4.7 节中定义）发送的请求响应中，“redirect_uri”参数的特定编码方式。

7.2.2 令牌端点

7.2.2.1 概述

令牌端点在 IETF RFC 6749 的 3.2 节描述。令牌端点嵌入在 Autho4API 授权服务器中，用于和 Autho4API 客户端进行交互，并通过 Autho-2 接口获得访问令牌。

必要时，Autho4API 授权服务器应认证 Autho4API 客户端，应在颁发访问令牌之前验证授权许可类型。

为了保证客户端认证凭证、授权许可和访问令牌的安全传输，Autho4API 授权服务器应支持 TLS 1.1 见 IETF RFC4346 并建议支持 TLS 1.2（见 IETF RFC5246）。

7.2.2.2 许可类型

关于授权许可见 IETF RFC 6749 的 1.3 节描述。

— Autho4API 客户端应至少支持授权码、隐式的、资源拥有者口令和客户端认证凭证许可类型中的一个；

— Autho4API 授权服务器应至少支持授权码和隐式许可类型，并可选支持资源拥有者口令和客户端认证凭证许可类型。

7.2.3 令牌撤销端点

Autho4API 客户端和 Autho4API 授权服务器可选支持令牌撤销功能；见 IETF RFC 7009 中定义。

如果支持在 IETF RFC 7009 中描述的令牌撤销端点，该令牌撤销端点由 Autho4API 授权服务器通过 Autho-2 发布，可让 Autho4API 客户端用于撤销访问令牌或更新令牌。此外：

— 当 Autho4API 客户端为机密类型，Autho4API 授权服务器应认证 Autho4API 客户端，并在撤销令牌之前，可选验证客户端是否被授权撤销指定的令牌；

— 当 Autho4API 客户端为公共类型，Autho4API 授权服务器应标识 Autho4API 客户端，并在撤销令牌之前，可选验证客户端是否被授权撤销指定的令牌；

— 为了提供传输层安全，Autho4API 授权服务器应支持 TLS 1.1（见 IETF RFC4346）和可选支持 TLS 1.2（见 IETF RFC5246）。

7.3 服务参数发现(参考性信息)

7.3.1 概述

OAuth 2.0 见 IETF RFC 6749 没有定义 Autho4API 客户端如何发现运行协议时需要的服务参数：

- 服务器端点 URL；
- 服务器能力，即支持的协议选项；
- 资源位置；
- 授权范围值及与资源访问的映射关系。

当这些参数固定并在网络 API 的服务文档中描述定义，Autho4API 客户端能够在配置之前静态的嵌入这些参数。

另外，Autho4API 客户端需要通过一些机制动态的发现未知参数，不仅仅为了正确运行协议，而且基于用户体验的考虑，为了提前确定是否能够在资源拥有者加入授权过程之前运行协议。

7.3.2 服务参数

本节描述了 Autho4API 客户端应以某种方法发现参数，以确保适当的运行协议流程，直到发起受保护资源访问请求。

7.3.2.1 引导过程

在配置阶段，Autho4API 客户端尚不知道所有使用授权框架需要的参数，但至少需要提供一些可以后续用于获取这些参数的信息，例如：服务发现端点位置。

7.3.2.2 授权范围值（规范性信息）

7.3.2.2.1 定义

一个使用独立的网络 API 规范，它的授权框架可定义授权访问值。

附录 B 提供了为指定网络 API 定义授权范围值的考虑。

7.3.2.2.2 命名和注册

开放移动命名权威机构 OMA OMNA_Autho4API 维持着应用于网络 API 的 Autho4API 授权范围值的

注册信息，网络 API 使用 Autho4API 引擎作为它的授权框架。

此注册信息中的授权范围值名称应是区分大小写的字符串，其要求如下：

— 首先，符合在 IETF RFC 6749 中 3.3 章节中指定的“scope-token”元素，特别定义了授权范围值许可的字符集；

— 其次，符合以下语法，以确保一致性：

{ScopeValue} ::= {{OMAScopeValue} | {ExtOrgScopeValue} |
{UnregisteredScopeValue}}
{OMAScopeValue} ::= oma_{ApiType}_{ApiIdentification}.{Token}
{ExtOrgScopeValue} ::= {ExtOrgPrefix}_{Label}
{UnregisteredScopeValue} ::= x_{Label}

具体说明见表 12。

表 12 授权范围值名称说明

名称	描述
ApiType	网络 API 类型。字符串格式，其中不能包含 “_” 字符
ApiIdentification	网络 API 标识。包含在 {ApiType} 定义范围中。字符串格式，其中不能包含 “_” 和 “.” 字符
Token	由 {ApiType}_{ApiIdentification} 识别的 API，作用在资源集合上的操作集标识。字符串格式
ExtOrgPrefix	外部组织定义的前缀。推荐用于识别该组织自身身份。字符串格式，其中不能包含 “_” 字符
Label	部分授权访问值前缀范围内的网络 API 中，作用在资源集合上的操作集标识。字符串格式

由 OMA 规范定义的 Autho4API 授权范围值应遵循 {OMAScopeValue} 语法，并应向 OMNA Autho4API 授权范围值登记处注册。

定义 Autho4API 授权范围值的外部组织，推荐向 OMNA Autho4API 授权范围值登记处注册一个授权范围值前缀，并适当的将它的授权范围值定义发布到公开可用的规范。

授权范围值前缀 “x_” 可以用于自由定义 Autho4API 授权范围值，此类授权范围值不会向 OMNA 注册。同样的，这些授权范围值在用于 Autho4API 端点时，不能保证是无冲突的。

7.3.2.3 Autho4API 授权服务器参数

为了在接口 Autho-1、Autho-2 处与 Autho4API 授权服务器正常地交互，Autho4API 客户端需要得知以下服务器相关参数：

对于包含授权请求的协议流程所需参数有：

- 授权端点的位置；
- 支持/要求的授权端点扩展参数；
- 支持的第二通道，用以传送授权请求响应，如果适用，同时需要支持的响应加密算法；
- 支持的客户端类型；
- 支持的响应类型；
- 支持的传输层机制；
- 要求的客户端环境所具备的能力，例如，在它们参与用户认证过程的情况下所具备的能力。

对于包含访问令牌请求的协议流所需参数有：

- 令牌端点位置；
- 支持/要求的令牌端点扩展参数；

- 支持/首选的客户端认证方法;
- 支持的授权许可类型;
- 支持/首选的传输层机制;
- 已发布的访问令牌种类(承载令牌等)。

当支持令牌撤销特征属性时:

- 令牌撤销端点的位置;
- 支持/首选的传输层机制。

此处应注意:当部署 Autho4API 客户端时,在以下情况可能不能静态认知给定端点(例如,授权端:

- 服务提供商为追求灵活性,随时间改变端点 URL;
- 服务提供商暴露多个授权端点(例如,被暴露的每个端点都针对一个专有的用户认证方法、都针对常用 API);
- 服务提供商的聚集为每个服务提供商都暴露一个端点,在这种情况下客户端部署是不为服务商所知的。

7.3.2.4 Autho4API 访问控制服务器参数

为了在接口 Autho-3 处与 Autho4API 访问控制服务器进行正常的交互,Autho4API 客户端需要获悉服务器相关参数:已发布访问令牌的类型,并且对于承载令牌还需要支持/首选的访问令牌包含方法。

7.3.3 发现机制

本标准不定义 Autho4API 客户端如何获取与 Autho4API 授权服务器、Autho4API 访问控制服务器进行交互所需要的参数的相关机制。

7.4 获取授权

7.4.1 授权代码流程

7.4.1.1 基于 OAuth2.0 的实现

7.4.1.1.1 概述

该章节描述了 Autho4API 客户端利用授权码许可类型从资源拥有者获得授权访问资源的过程,在 IETF RFC 6749 的 4.1 节描述。

IETF RFC 6749 的 4.1 节中的说明应当遵从相关流程中各种不同的角色:Autho4API 客户端、资源拥有者的用户代理与 Autho4API 授权服务器。

7.4.1.1.2 详细协议流程(信息)

通过 OMAAutho4API 实体与接口将 IETF RFC 6749 中 4.1 节图 3 具体流程映射到下面图 3 的流程。

步骤1) Autho4API 客户端重定向资源拥有者的用户代理至 Autho4API 授权服务器端点。该步骤与 IETF RFC 6749 的 4.1 节中步骤 A 对应,在 IETF RFC 6749 的 4.1.1 节中详细描述。关于 Autho4API 客户端如何重定向至资源拥有者用户代理的过程不在本标准讨论范围内,因为它通过接口 Ext-1 实现的。

步骤2) 资源拥有者认证通过并准许 Autho4API 客户端访问其资源。该步骤与 IETF RFC 6749 中 4.1 节中的步骤 B 对应。关于第 2 步如何执行,本标准不作讨论。

步骤3) Autho4API 授权服务器应答步骤 1 中的请求,重定向资源拥有者用户代理至步骤 1 提供的重定向 URI;资源拥有者用户代理发送相应的 HTTP GET 请求至已接收到的 HTTP 302 响应头地址指定的

URI。该步骤对应 IETF RFC 6749 的 4.1 节中的步骤 c。该步骤的具体细节见 IETF RFC 6749 的 4.1.2 节。关于资源所有者用户代理如何执行重定向至 Autho4API 客户端不属于本标准标准的研究范围，因为它通过接口 Ext-1 实现的。

步骤4) Autho4API 客户端发送访问令牌请求至 Autho4API 授权服务器。该步骤与 IETF RFC 6749 的 4.1 节的步骤 D 对应，具体细节见 IETF RFC 6749 中 4.1.3 节。

步骤5) Autho4API 授权服务器做出回应，将访问令牌与可选的更新令牌发送至 Autho4API 客户端。该步骤与 IETF RFC 6749 中 4.1 节对应，具体细节详见 IETF RFC 6749 的 4.1.4 节。

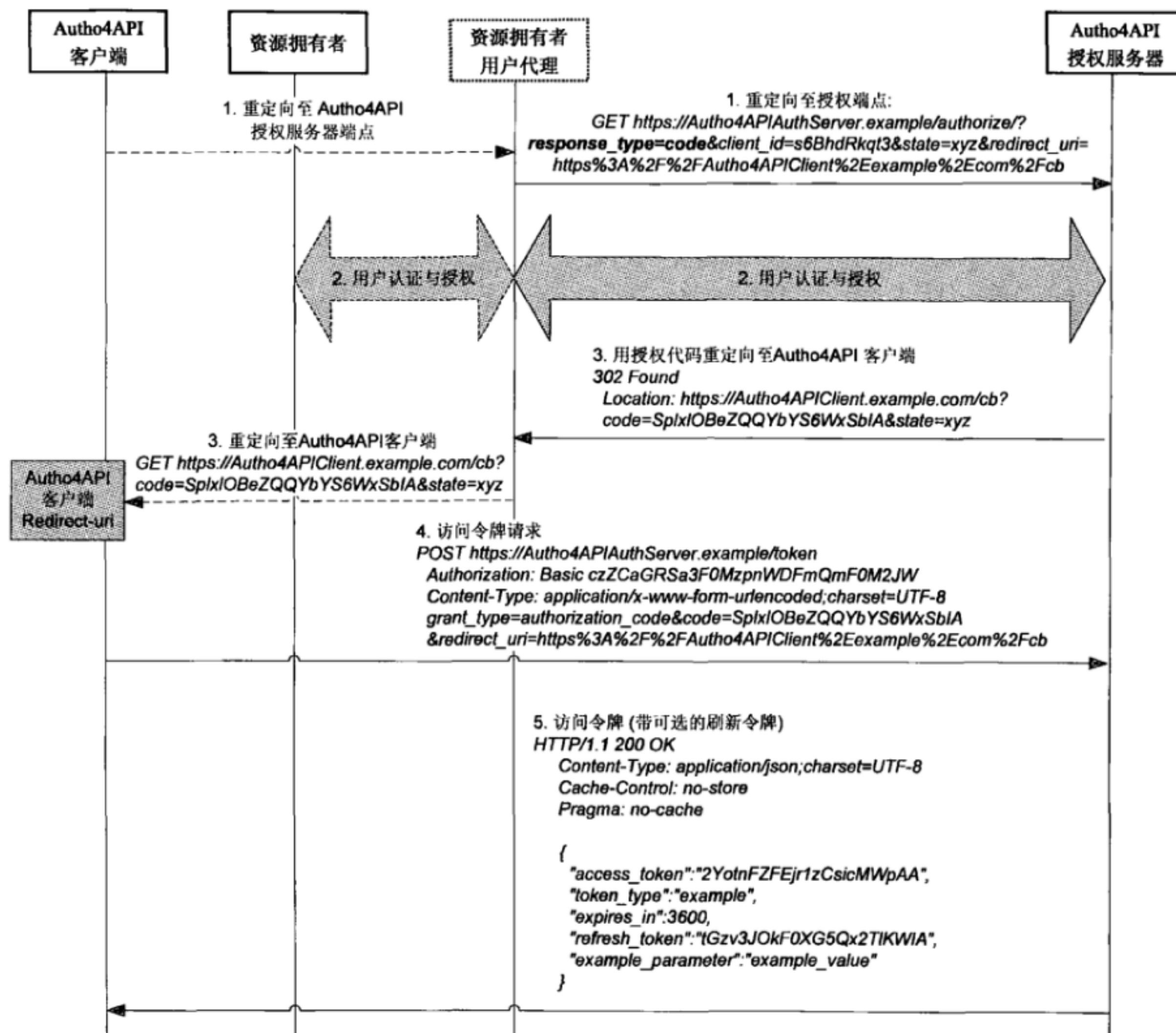


图3 利用授权代码许可类型获取授权——详细协议流程

7.4.1.2 对本地应用的支持（信息）

客户端侧安装的应用（例如，本地代码应用）虽然一般情况下不能接收传入的 HTTP 请求（包括传输授权请求响应的 HTTP 重定向），但利用 7.4.6 节中指定的策略仍然支持授权码流程。

7.4.2 隐式许可流程

7.4.2.1 基于 OAuth2.0 的实现

该段描述了 Autho4API 客户端利用隐式许可类型（Implicit Grant type）获取资源拥有者的授权，访问受保护资源的过程，见 IETF RFC 6749 中 4.2 节所述。

IETF RFC 6749.2 节的说明应当遵从相关流程中的各种不同的角色：Autho4API 客户端、资源拥有者的用户代理与 Autho4API 授权服务器。

通过 OMAAutho4API 实体与接口，将 IETF RFC 6749 的 4.2 节图 4 具体流程映射到下面图 4 的流程。

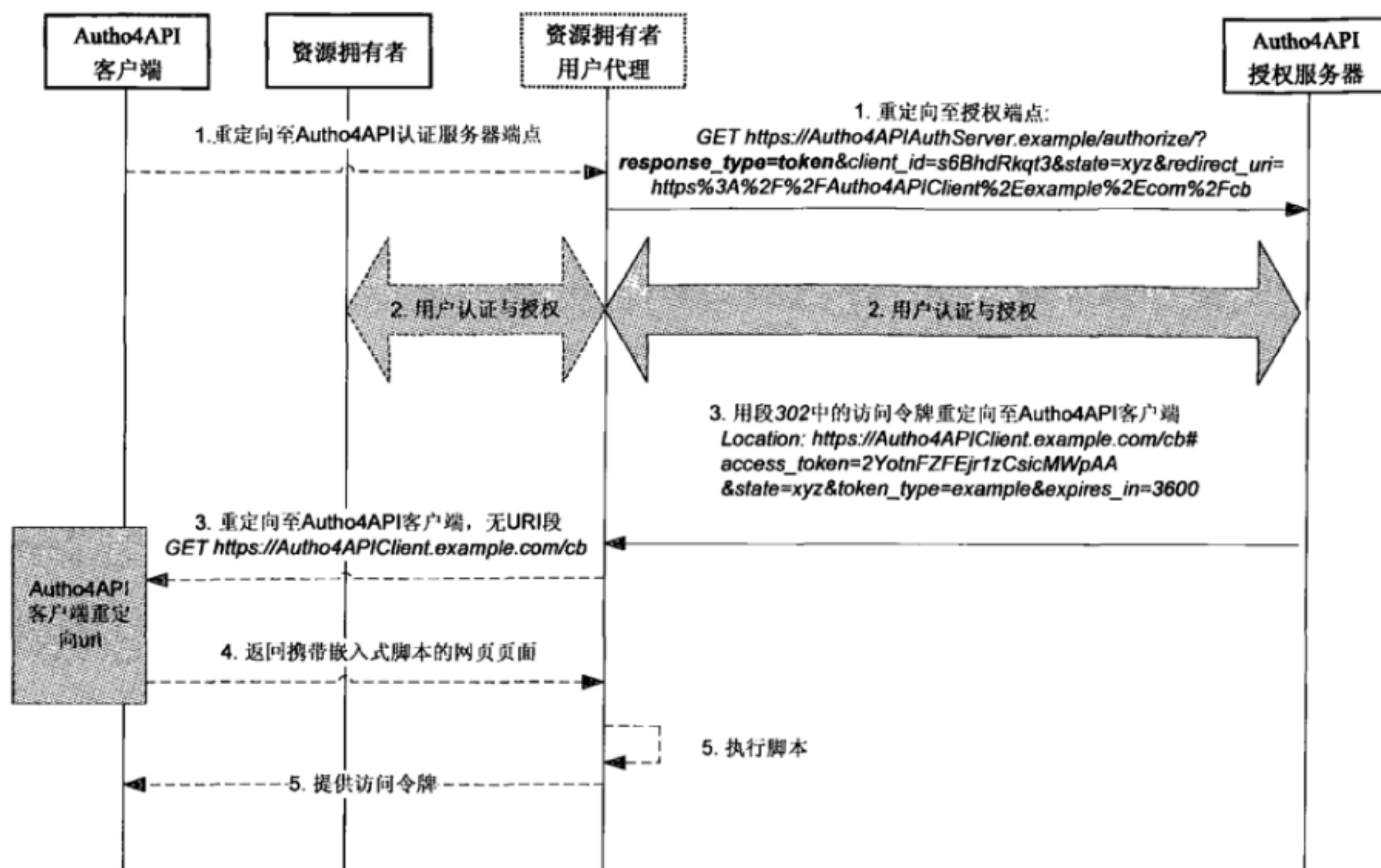


图 4 利用隐式许可类型获取授权：详细协议流程

步骤1) Autho4API 客户端重定向资源拥有者用户代理至 Autho4API 授权服务器端点。该步骤对应 IETF RFC 6749 中 4.2 节的步骤 A，详细细节参见 IETF RFC 6749 中 4.2.1。关于 Autho4API 客户端如何重定向资源拥有者用户代理不属于本标准研究范围，因为它通过接口 Ext-1 实现的。

步骤2) 资源拥有者认证通过并准许 Autho4API 客户端访问其资源。该步骤对应 IETF RFC 6749 中 4.2 节中的步骤 B。关于如何详细执行第 2 步不属于本标准研究范围。

步骤3) Autho4API 授权服务器应答步骤 1 中的请求，并重定向资源拥有者用户代理至步骤 1 提供的重定向 URI；URI 段提供访问令牌。资源拥有者用户代理发送相应的 HTTP GET 请求至接收到 HTTP 302 响应的地址头内指定的 URI。该步骤对应 IETF RFC 6749 的 4.2 节中的步骤 C、D 的内容，该步详细描述参见 IETF RFC 6749 的 4.2.2。关于资源拥有者用户代理如何执行 Autho4API 客户端重定向不属于本标准研究范围。

步骤4) Autho4API 客户端返回一个网页（典型的是一个带嵌入式脚本的 HTML 文档），该网页能够访问完整的重定向 URI 包括资源拥有者用户代理保留的段，并提取段中包含的访问令牌（和其它参数）。该步骤对应 IETF RFC 6749 中 4.2 节步骤 E。该步骤具体细节不属于本标准研究范围，因为它通过接口 Ext-1 实现的。

步骤5) 资源拥有者用户代理在本地执行 Autho4API 客户端网络主机提供的脚本，提取访问令牌并将其发送至 Autho4API 客户端。该步骤对应 IETF RFC 6749 的 4.2 节中的步骤 F 和 G，其具体细节不属于本标准研究范围，因为它通过接口 Ext-1 实现的。

7.4.2.2 对本地应用的支持（信息）

客户端侧安装的应用（例如，内部代码应用），虽然通常不能接收传入的 HTTP 请求（包括携带授权请求响应的 HTTP 重定向），但利用本标准 7.4.6 节中指定的策略仍然支持隐式许可流程。

7.4.3 资源拥有者密码凭证流程

Autho4API 客户端与授权服务器可以支持资源拥有者密码凭证流程，如 IETF RFC 6749 中 4.3 节定义。Autho4API 客户端获取资源拥有者凭证的相关机制不属于本标准研究范围。

7.4.4 客户端凭证流程

Autho4API 客户端与授权服务器可以支持客户端凭证流程，见 IETF RFC 6749 的 4.4 节所定义。关于 Autho4API 客户端获取客户端凭证的方法（或其它支持的认证方式）不属于本标准研究范围。

7.4.5 其它流程

本标准通过 IETF RFC 6749 的 4.5 节定义的许可类型扩展，可以支持其它 IETF 定义的 OAuth2.0 流程。

7.4.6 本地应用支持

7.4.6.1 概述

ietf RFC 6749 定义了一些流程（授权代码流程、隐式许可流程），使得位于资源拥有者认证与授权步骤之后的协议响应可以通过将 HTTP 重定向至客户端的公共 HTTP URL 来传递。

然而客户端侧安装的应用（如本地代码应用）通常不能接收传入的 HTTP 请求（包括由 HTTP 重定向导致的 HTTP 请求），因此不能接收对授权请求的协议响应。

7.4.2、7.4.3 和 7.4.4 指定了一些策略，这些策略能够使得本地应用仍然支持基于 HTTP-重定向的基本流程。这些策略（在 IETF RFC 6749 的 9 节中概括的那些策略）可分为两类：

- 通过 HTTP 重定向与设备侧机制相结合的方式传递响应，具体细节参见 7.4.6.2；
- 通过第二通道（即代替 HTTP 重定向的另一种通道）的方式传递响应，具体细节参见 7.4.7。

7.4.6.2 通过 HTTP 重定向的方式传输响应（信息）

该章节描述了具体方法：授权请求响应由 Autho4API 授权服务器借助 HTTP 重定向以及一些附加机制（通过设备路由至应用）传输到设备。可能有以下几种情况：

— Autho4API 客户端嵌入用户代理。该方法使得 Autho4API 客户端通过嵌入用户代理本身能捕捉 HTTP 重定向，见 IETF RFC 6749 中解释的。

利用嵌入用户代理可能面临安全威胁：资源拥有者在一个未经标识的窗口进行认证，该窗口并不具备用大多数外部用户代理提供的视觉保护，因此降低了资源拥有者对某种攻击威胁的保护（例如，点击劫持与钓鱼攻击）。

— 运行本地 Web 服务器。该方法使得 Autho4API 客户端调用一个外部用户代理，然后重定向至本地 Web 服务器，该服务器作为外部通用模块或嵌入到 Autho4API 客户端运行。这样，Autho4 API 客户端就可以与本地 Web 服务器进行交互获取响应。这种机制会引起几个问题：

- 当应用位于本地 Web 服务器之外的情况下，恶意应用可以与本地 Web 服务器进行交互拦截处理过的响应；
- 多种应用利用这种机制安装在设备上，因此这些应用的重定向 URI 共享同一个域（即本地主机），这样，需要特殊维护与协调（通过分配特定的端口、路径、查询参数），才能避免重定向 URI 值发生冲突。

— WAC Webview 设备 API: 通过 WAC2.1 中定义的设备 API, 在 Web 运行环境中正在执行的应用程序可以加载 (通过控制 web 运行环境) 受限浏览器到授权端点, 然后就收到被 web 运行环境捕获的来自 HTTP 重定向受限浏览器的授权请求响应。该获取机制的特征如下:

- 安全。因为: 第一, 用于请求和响应的通道为同一个 (用户代理浏览内容被 Web 运行环境控制); 第二, 响应仅返回至发起流程的应用实例。

- 对需要在网络运行环境下执行的本地应用 (如 Widget) 有所限制。

— 用 OS 注册 URI 组件重定向 URI: 通过这种方法, 同服务提供商一起预注册的重定向 URI 包含许多注册在操作系统上的组件 (算法, 以及其他模块例如授权与路径), 通常是在应用程序安装时注册的。无论本地浏览器什么时候正在打开一个 URL 去匹配这些组件, 本地浏览器都会处理应用程序的 URL。这种运行在操作系统上的机制存在一些负面影响:

- 该机制属于操作系统专有;
- 当同一应用的很多实例 (共享同一重定向 URI) 同时驻留在设备上时, 会引起路由问题;
- 引起安全问题, 恶意应用可以注册同一 URI 组件, 拦截重定向有效净荷。

7.4.7 用于传递响应的第二通道

7.4.7.1 概述

该章节详细描述了通过辅助通道 (即非 HTTP 重定向通道) 从 Autho4API 授权服务器传输至 Autho4API 客户端的授权请求响应的方法。

本地 Autho4API 客户端或其他类型的客户端 (公共或私密客户端) 可以见 IETF RFC 6749 使用这些方法。

7.4.7.2 传送方法

— 通过资源拥有者交互的响应显示。响应显示在资源拥有者的用户代理上 (即显示到浏览器中), 资源拥有者复制该响应至 Autho4API 客户端。这意味着:

资源拥有者希望执行交互过程, 当响应内容很长和/或包含特殊字符 (如访问令牌) 时, 用户体验会很差。

— 响应通过自动客户端检索传递给用户代理。Autho4API 客户端利用本标准研究范围外的取回方法, 将响应传递给资源拥有者用户代理。一种方法是响应显示在资源拥有者的用户代理的窗口标题上, Autho4API 客户端能够读取该标题。

— 通过与资源拥有者的文本短消息交互传递响应。该响应以文本短消息净荷的形式传递, 资源拥有者能够读取这个响应, 并将该响应传送至 Autho4API 客户端。这意味着:

- 资源拥有者希望执行这个交互。当响应内容很长和/或包含特殊字符 (如访问令牌) 时, 用户体验会很差;

- Autho4API 授权服务器或者发送短消息的实体能够利用某种方法 (具体方法细节不属于本标准研究范围) 知悉资源拥有者的 MSISDN。

— 利用 OMA 无连接短消息推送功能 (OMAPUSH) 传递响应。该响应被传输至设备上的 OMA 推送客户端, 然后 OMA 推送客户端通过推送应用寻址方法将响应路由至 Autho4API 客户端。这说明:

- 接受推送响应的资源拥有者的设备正好是应用和 Autho4API 客户端运行的设备;

● 下载应用与 Autho4API 客户端（即没有随着 OMA 推送客户端预安装）等这些应用不具备应用 ID（即应用分配代码）的情况下，OMA 推送客户端通常需要：

- 接受应用 ID 的动态注册；
- 支持纯 URI 格式的应用 ID，如果“INST”参数可以用来区分相同应用程序安装在装置中的多个实例，则该格式包含一个查询模块。

● Autho4API 授权服务器或发送无连接推送短消息的实体能够利用某种方法（该方法细节不属于本标准研究范围），获悉资源拥有者的 MSISDN。

7.4.7.3 预注册重定向端点

Autho4API 客户端意图请求利用一个或多个第二通道传输授权请求响应，这将为每个拟使用的第二通道注册一个重定向端点，该重定向 URI 需具有以下格式：

`http://{authorizationServer}/autho4apiSecondaryChannel/{channel}`

表 13 规定了 URI 变量的值。

表 13 URI 变量的值

名称	描述
授权服务器	授权服务器的授权端点URL的授权方IETF RFC3986。 该方可作为可选项利用某种机制（该机制具体细节不属于本标准研究范围）为Autho4API客户端动态发现授权端点的环境提供注册时段
通道	要使用的第二通道的特定类型将使用以下变量中的某一个： <ul style="list-style-type: none"> • ‘sms_text’，通过文本短消息传递请求响应 • ‘push_over_sms’，利用短消息无连接推送传递请求响应 • ‘browser_title’，通过设置浏览器窗口标题的响应值的自动客户端检索机制将请求响应传递给用户代理 • ‘browser_display’，在浏览器页面显示请求响应，以便于与资源拥有者的进一步交互

当某个辅助通道为短消息无连接推送时，Autho4API 客户端将不仅仅注册推送应用 ID 的‘app-id-base’部分，这已在 7.4.7.3 节中定义。

Autho4API 客户端中将不会注册其他由“redirect_uri”参数定义的 URI 查询参数，因为它们要么依赖于实例（例如变量“inst”），要么是授权请求相关的（例如，“加密”）。

如果提交注册的重定向 URI 具备以下前缀的话 Autho4API 授权服务器将为潜在用途的辅助通道检测 Autho4API 客户端请求：

`http://{authorizationServer}/autho4apiSecondaryChannel/`

在这种情况下，如果以下情况为真，那么 Autho4API 授权服务器将拒绝重定向端点注册。

- Autho4API 授权服务器不支持第二通道功能；
- 通道后续 URI 前缀缺失、未知或不支持。

7.4.7.4 授权请求“redirect_uri”参数

Autho4API 客户端通过第二通道接收来自授权请求的响应，将构建授权请求的“redirect_uri”参数如下：

- a) 通过利用第二通道选择由 Autho4API 客户端预注册的重定向 URI；
- b) 如果在注册时不知道该参数，设定这个 URI 授权模块为授权服务器授权端点的授权模块；
- c) 如果请求特殊通道，应为该 URI 添加一个查询模块，其查询参数定义见表 14。

表 14 参数定义

名称	种类/变量	可选项	描述
app-id-base	纯 URI 定义于 IETF RFC3986, 或代表应用分配代码的字符串	可选	<p>该参数将在通道设置为‘push_over_sms’的情况下包含, 否则, 该参数不会被添加。</p> <p>当一个字符串代表应用分配代码时:</p> <p>它相当于应用ID分配代码OMAPUSH的十六进制表示 (被用于路由无连接短消息推送至Autho4API客户端), 该代码在客户端注册时是已知的, 前缀为“0x”。</p> <p>例如: 0x909A</p> <p>当纯URI格式时:</p> <ul style="list-style-type: none"> • 它相当于应用ID部分OMAPUSH (被用于路由无连接推送短消息至Autho4API客户端), 该部分在客户端注册阶段是已知的。 • 若参数“inst”没有在授权请求中传输, Autho4API授权服务器会设定OMA推送应用ID为‘app-id-base’ URI, 后面附加 ‘inst’ 查询参数。 <p>例如</p> <p>app-id-base=http://example_app_id.com inst=qwertyasdf</p> <p>Resulting Application-ID: http://example_app_id.com?inst=qwertyasdf</p> <ul style="list-style-type: none"> • 若参数“inst”在授权请求内被传输, Autho4API授权服务器设置OMA推送应用ID为‘app-id-base’ URI与‘inst’查询参数
Inst	字符串	是	<p>在通道被设定为‘push_over_sms’时, 该参数可能被包含, 在纯URI格式下还可能包含查询参数‘app-id-base’; 否则, 将不被包含。</p> <p>该参数对应应用ID见OMAPUSH的一部分, 该ID针对于已安装应用的实例, 因此在客户端注册时未知。</p> <p>该参数用于客户端平台, 该客户端平台已安装应用动态注册一个推送应用ID至OMA推送客户端</p>
Encryption	字符串	是	<p>该参数的引入是为了对通过第二通道传输的响应请求加密。该参数的值说明了Autho4API授权服务器利用对称密钥加密响应的算法</p> <p>加密算法将采用高级加密标准 (AES) 算法, 在FIPS AES中描述, 密钥大小为128位、192位或256位, 加密操作模式密码块链接CBC, 在FIPS AESMode中描述。</p> <p>参数‘encryption’的ABNF定义为:</p> <p>encryption = “AES_” (“128” “192” “256”) “_CBC”</p>
encryption_key	字符串	是	<p>当参数‘encryption’被引入时, 该参数将出现。</p> <p>该参数的值标识了对称密钥用于响应加密。</p> <p>加密密钥通过Autho4API客户端被随机生成, 且每个授权请求的都不相同。</p> <p>参数‘encryption_key’的ABNF定义为:</p> <p>encryption_key = 32 (HEX) 48 (HEX) 64(HEX)</p>
encryption_IV	字符串	是	<p>当引入参数‘encryption’时, 该参数会出现。</p> <p>根据CBC加密模式, 该参数的值表示了需要被加密响应的初始向量</p> <p>每一个授权请求的初始向量由Autho4API客户端产生。</p> <p>‘encryption_IV’参数的ABNF定义为:</p> <p>参数‘encryption_IV’的ABNF定义为:</p> <p>encryption_IV = 32 (HEX)</p>

支持第二通道功能的 Autho4API 授权服务器支持“encryption”, “encryption_key”与“encryption_IV”等

参数。

支持第二通道“短消息无连接推送”的 Autho4API 授权服务器支持“app-id-base”与“inst”等参数。

参数‘redirect_uri’样本值为：

`http://exampleServiceProviderAuthServer.com/autho4apiSecondaryChannel/sms_text`

`http://exampleServiceProviderAuthServer.com/autho4apiSecondaryChannel/push_over_sms?app-id-base=http%3A%2F%2Fexample_app_id.com&encryption=AES_128_CBC&encryption_key=63cab7040953d051cd60e0e7ba70e18c&encryption_IV=6353e08c0960e104cd70b751bacad0e7`

`http://exampleServiceProviderAuthServer.com/autho4apiSecondaryChannel/push_over_sms?app-id-base=http%3A%2F%2Fexample_app_id.com&inst=qwertyasdf`

`http://exampleServiceProviderAuthServer.com/autho4apiSecondaryChannel/push_over_sms?app-id-base=0x909A`

`http://exampleServiceProviderAuthServer.com/autho4apiSecondaryChannel/browser_title`

7.4.7.5 通过第二通道传递的响应请求

Autho4API 客户端希望通过第二通道接收授权请求响应，需确保以下前提：

- a) 通过 Autho4API 授权服务器指定该通道潜在用途完成重定向端点的注册。
- b) 在利用 OMA 无连接短消息推送传递响应的情况下，通过 OMA 推客户端完成应用 ID 的注册（如果适用，如果在纯 URI 格式下则包括参数‘inst’部分）。

Autho4API 客户端会按如下方式构建授权请求：

- a) 参数“redirect_uri”应按 7.4.7 节定义的值结构被包含。另外：

通过使用上节提到的加密相关的 URI 查询参数，Autho4API 客户端可能要求第二通道的响应加密。是否强制 Autho4API 客户端请求响应加密是实现决策，这取决于 Autho4API 客户性质和所使用的第二通道。根据 Autho4API 客户端特性利用第二通道的情况，Autho4API 客户端可以要求通过第二通道利用加密相关 URI 查询参数响应加密，决定是否强制 Autho4API 客户端请求响应加密。

- b) 参数“state”在 IETF RFC 6749 中定义：

— 当“通过与资源所有者交互的响应显示”与“通过与资源所有者交互的文本短消息传送响应”的情况出现时，参数“state”不应该被引入；

— 当“通过自动客户端取回方式传送响应至用户代理”与“用短消息无连接推送传送响应”的情况出现时，参数“state”应该被包含。

- c) 其他服务器相关授权端点扩展（本标准未定义）可以被包含。

- d) 其他要求的通过指定通道传输的适当响应可以被包含。

通过短消息（OMA 无连接短消息推送或文本短消息）请求响应（以明文或密文形式）被传输，为了传输的可靠性，应当适配一个单独的短消息，大小不应该超过 4 段的短消息，因为这是 OMA 推客户端需要重新组装的最小尺寸。为确保这一点，Autho4API 授权服务器可以通过本标准以外的方法指示 Autho4API 客户端使用一个不超过给定长度的状态参数。

7.4.7.6 请求处理与错误处理

该章节详细介绍了当 Autho4API 客户端在授权请求中通过 7.4.7.3 节定义的“redirect_uri”指定的结构请求使用第二通道时的 Autho4API 授权服务器的错误处理机制。

如果通道参数“redirect_uri”将存在以下前缀，Autho4API 授权服务器会利用第二通道探测 Autho4API 客户端请求：

http://{authorizationServer}/autho4apiSecondaryChannel/

其中{authorizationServer}授权端点 URL 的授权模块与发送的授权请求内容相匹配。

此时，如有如下情形，则 Autho4API 授权服务器会产生一个“非法重定向 URI”错误：

- 不支持第二通道功能；
- 支持第二通道功能，但满足如下情形之一：
 - “redirect_uri”不是一个有效 URI；
 - “redirect_uri”不符合 7.4.7.3 节中规定的语法；
 - Autho4API 客户端已经注册，但是“redirect_uri”的值（不包括其中的查询模块）不是该客户端注册的重定向端点；

● Autho4API 客户端已经注册，“redirect_uri”的值（不包括其中的查询模块）也与该客户端注册的重定向端点匹配，第二通道方式为 OMA 无连接短消息推送，但请求（不包含“ins”参数）中提供的应用 ID 不是该客户端注册的值；

- Autho4API 客户端没有注册，但是 Autho4API 授权服务器不支持指明的第二通道。

Autho4API 授权服务器产生“无效的重定向 URI”错误时，应当遵从 IETF RFC 6749 中 3.1.2.4 定义的错误处理。

如果没有以上错误，也没有其他在 IETF RFC 6749 中规定的错误，Autho4API 授权服务器会验证“redirect_uri”中的查询模块。如果存在：当查询模块包含未知的密钥或值（在 7.4.7.3 中指定的），或者不支持一些密钥或值（如 AES 的密钥长度），Autho4API 服务器会产生一个“无效请求”错误。

只要授权请求失败，Autho4API 授权服务器应当在所有情况下通过解释性的网页告知资源拥有者。

7.4.7.7 响应编码

该节规定了 Autho4API 授权服务器会如何回复 Autho4API 客户端，这将随使用的第二通道和协议流变化而变化。

对于“与资源拥有者交互的回复展示”和“与资源拥有者交互通过文本短信传递回复”：

如果授权请求失败，则 Autho4API 授权服务器不会通过第二通道传送回复。

否则，通过第二通道传送的回复应当按照如下方法构造：

a) 用 W3C HTML_4.01 中定义的“application/x-www-form-urlencoded”编码方法来编码特定流的参数值。参数如下：

— 对于授权代码流程：“code”

例如：vc1234

— 对于隐式许可流程：“access_token”

例如：AAACa8r8lZA4ABAAP0vw1sSG

b) 如果授权请求中有 7.4.7.3 节中定义的加密参数，则将先前的字符串用这些参数加密，将结果用 Base64 编码，否则传输纯文本的回复。

c) 将页面或文本信息中的结果字符串与可选的文本信息放在一起，指导资源拥有者如何处理回复。

对于“通过自动客户端检索向用户代理传递响应”和“通过 OMA 的无连接短信息推动发送传递回

复”的情形，回复应当按照如下方式构造：

a) 建立可用的参数（键/值对）列表，包含在回复中。这些参数包括流相关的参数以及其他服务器相关的参数（没有在该标准中定义）。适用的流相关的参数是：

— 对于授权码流程：IETF RFC 6749 中 4.1.2 节（如果成功回复）和 4.1.2.1 节（如果错误回复）定义的 OAuth 2.0 参数

Example: code=vc1234&state=xyz

Example: code=1vc456

Example: error=access_denied

— 对于隐式授权流程：IETF RFC 6749 中 4.2.2 节（如果成功回复）和 4.2.2.1 节（如果错误回复）定义的 OAuth 2.0 参数。

Example: access_token=AAACa8r8lZA4ABAAp0vw1sSG&token_type=Bearer&expires_in=3600&state=xyz

b) 用 W3C HTML_4.01 中定义的“application/x-www-form-urlencoded”编码格式来编码表单数据集。

c) 如果在授权请求中提供了 7.4.7.3 节中定义的加密参数，则使用这些参数对先前的字符串进行加密。

对于“通过自动客户端取回方式传递响应给用户代理”的情况：如果响应是加密的，那么用 Base64 对结果数据编码，否则以明文方式来回复。

对于“通过 OMA 短消息无连接推送传送响应”的情况：

a) 将推送信息构造为每个 OMAPUSH，回复包含在内容体中；

b) 如果回复是加密的，就将正文类型的头设置为“application/octet-stream”见 IETF RFC2045 和 IETF RFC2046，否则设置为“text/plain”；

c) 将 X-Wap-Application-Id 的头设置为 Autho4API 客户端的“app-id-base”值。如果是纯 URI 格式，且在授权请求中提供了该值，则在后面添加“inst”查询参数。

7.4.7.8 详细协议流程（信息）

当授权请求响应通过第二通道传送时，图 3 中的授权码通用流程更改成如图 5 所示。

a) Autho4API 客户端定向资源拥有者用户代理至 Autho4API 授权服务器端点，参数 redirect_uri 被设置为：

http://exampleServiceProviderAuthServer.com/autho4apiSecondaryChannel/sms_text，如 7.4.7.3 节定义，请求授权响应以文本短消息形式传送。其余参数在图 3 的步骤 1 中描述。

b) 资源拥有者认证通过并准许 Autho4API 客户端访问其资源。该步骤和图 3 中的步骤 2 描述相同。关于步骤 2 具体如何执行不在本标准考虑范围之内。

c) Autho4API 授权服务器通过第二通道发送授权响应。关于步骤 2 具体如何执行不在本标准考虑范围之内，这牵涉到其他实体，如 SMSC 等。

d) Autho4API 客户端向 Autho4API 授权服务器发送访问令牌请求。该步骤即为图 3 中的步骤 4。

e) Autho4API 授权服务器回应 Autho4API 客户端，向其提供访问令牌和可选的更新令牌。该步骤即为图 3 中的步骤 5。

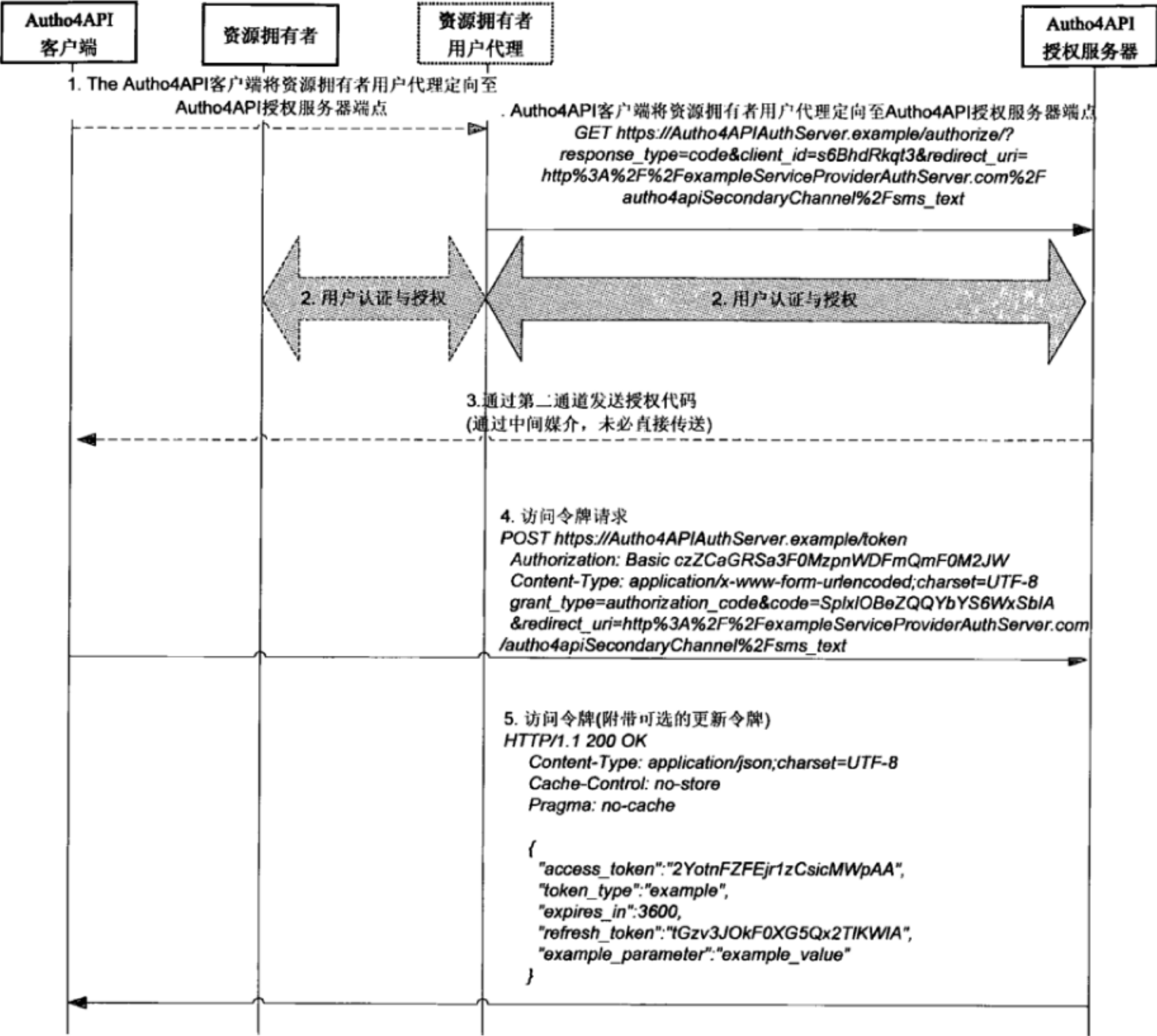


图5 利用授权码许可类型与第二通道获得授权——详细协议流程图

当授权请求响应通过第二通道传送时, 图4中的通用隐式许可流程更改成如图6所示。

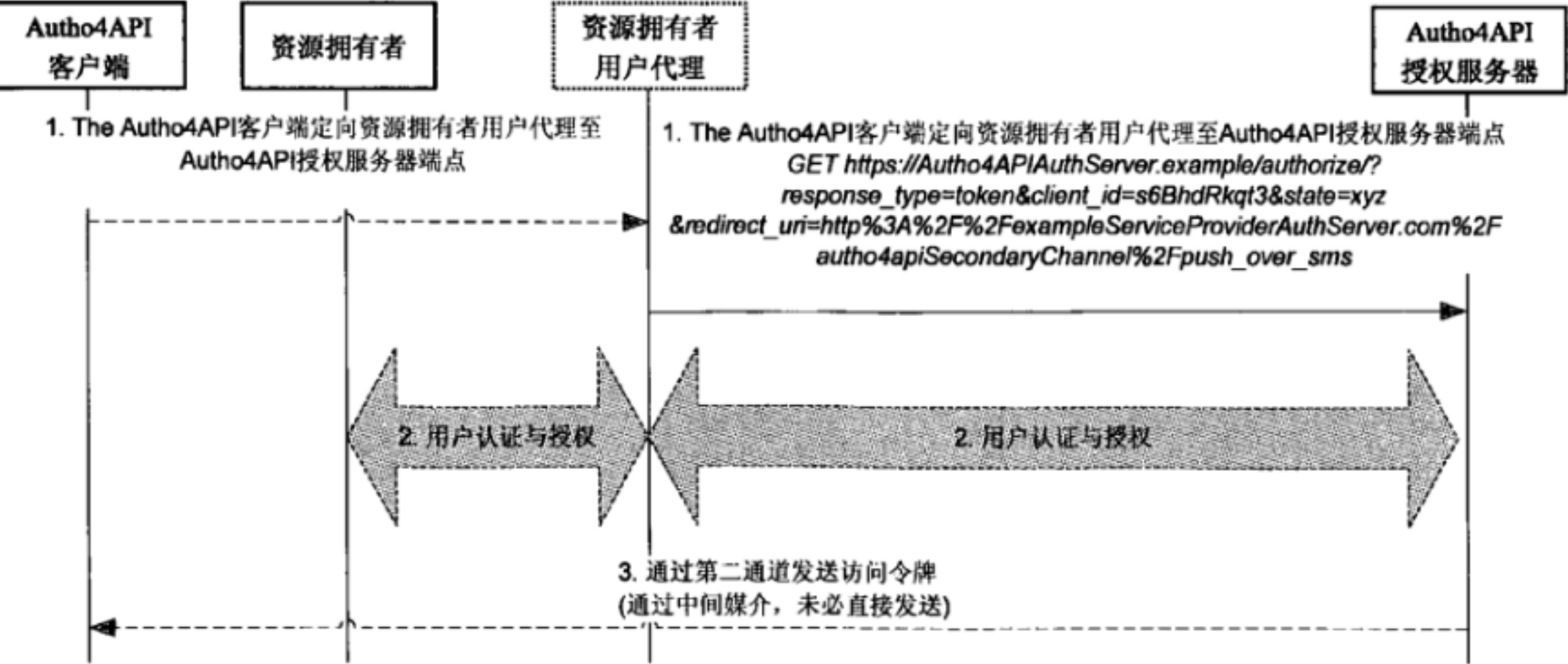


图6 利用隐式许可和第二通道获得授权——详细协议流程

a) Autho4API 客户端定向资源拥有者用户代理至 Autho4API 授权服务端点, 参数 `redirect_uri` 设置为:

`http://exampleServiceProviderAuthServer.com/autho4apiSecondaryChannel/push_over_sms`

如 7.4.7.3 节描述的, 要求的参数 `redirect_uri`, 请求以短消息无连接推送方式传送授权响应。其他参数如图 4 步骤 1 所描述。

b) 资源拥有者认证通过并准许 Autho4API 客户端访问其资源。该步骤和图 4 中的步骤 2 描述相同。关于步骤 2 具体如何执行不在本标准考虑范围之内。

c) Autho4API 授权服务器通过第二通道发送访问令牌响应。关于步骤 3 具体如何执行不在本标准考虑范围之内, 这涉及到其他实体, 如 SMSC 等。

7.4.7.9 安全性考虑

依赖于环境 (设备、操作系统), 以及授权请求响应是否能正确传送到 Autho4API 客户端正在运行的相同设备上等因素, 第二通道存在安全隐患。

注意: 因为第二通道是否安全取决于环境因素, 因此, 第二通道是否安全不在本标准的讨论范围之内。

授权码流程:

— 若认为第二通道不安全, 而 Autho4API 客户端根据 IETF RFC 6749 的 2.1 节判断为可信的, 则第二通道可以任意使用, 这是因为 (由于客户端认证的强制性) 仅获取授权码不足以交换访问令牌。然而, 为了增强传输的安全性, Autho4API 客户端应当要求响应以 7.4.7 节定义的方式加密, 以确保信道的安全;

— 若认为第二通道不安全, Autho4API 客户端根据 IETF RFC 6749 的 2.1 节描述也为公开的, Autho4API 客户端应要求响应以 7.4.7 节定义的方式加密, 以确保信道的安全。

隐式授权流程:

— 若认为第二通道不安全, Autho4API 客户端应要求响应以 7.4.7 节定义的方式加密, 以确保信道的安全。

当使用 Autho4API 客户端可以不通过资源拥有者的交互而找回响应这一方法时, 强烈推荐 Autho4API 客户端使用 IETF RFC 6749 的 4.1 节中的“状态”参数来验证第二通道收到的响应是否与待处理的认证请求匹配, 以此可以避免跨站点的伪造攻击。

7.5 发出访问令牌

在响应 Autho-2 接口上 Autho4API 客户端发送的访问令牌请求时, Autho4API 授权服务器按照 IETF RFC 6749 第 5 部分的规定应该返回成功或者失败的访问令牌响应。

7.6 更新访问令牌

已获得访问令牌和更新令牌的 Autho4API 客户端, 之后当该访问令牌到期后, 可以通过 Autho-2 接口向 Autho4API 授权服务器的令牌端点发送重更新请求, 详细过程可参考 IETF RFC 6749 第 6 部分。

7.7 访问受保护的资源

7.7.1 概述

7.7.1.1 访问令牌有效性

Autho4API 访问控制服务器在响应 Autho4API 客户端的请求之前, 应先检查资源请求中的访问令牌的有效性, 可参考 IETF RFC 6749 第 7 章。在以下任何情况为真时, Autho4API 访问控制服务器将判定

访问令牌无效并禁止资源访问:

- 访问令牌是一次性访问令牌,并且在访问令牌范围内的受保护资源的访问请求中已成功使用过一次;
- 访问令牌应符合完整保护格式,并且检测为格式改变;
- 访问令牌应符合结构格式,并且检测到格式不正确;
- 访问令牌不能被 Autho4API 授权服务器识别,此授权服务器和 Autho4API 访问控制服务器有信任关系;
- 访问令牌已经到期;
- 访问令牌或者已经获得访问令牌的更新令牌权限被撤销;
- 访问令牌没有涵盖资源请求的范围。

对于承载令牌 (bearer token): 如果受保护资源的请求没有包含任何访问令牌或者不包含有效访问令牌, Autho4API 访问控制服务器将向 Autho4API 客户端返回按照 IETF RFC 6750 第三部分构造的错误响应。

Autho4API 访问控制服务器是怎样协同 Autho4API 授权服务器执行访问令牌有效性检查的超出本标准的范围,可行的方法如下:

- 安全性: Autho4API 访问控制服务器从不接受无效的访问令牌,如当访问令牌刚被 Autho4API 授权服务器撤销权限;
- 可靠性: Autho4API 访问控制服务器从不拒绝有效的访问令牌,如 Autho4API 授权服务器刚发出访问令牌;
- 可扩展性: Autho4API 访问控制服务器能够处理大量的资源请求 (无论它们是没有令牌、包含无效令牌或者包含有效访问令牌的请求)。

一种最安全可靠的方法是: Autho4API 访问控制服务器每次收到资源访问请求时把访问令牌的有效性检查交给 Autho4API 授权服务器 (这一点在 Autho4API 访问控制服务器不能缓存访问令牌有效性时尤其有用),其它更多的可扩展性方法 (如, Autho4API 访问控制服务器能在某些情况下自己决定提交的访问令牌的有效或无效) 将提供相同的安全等级。

7.7.1.2 一次性访问令牌

IETF RFC 6749 定义的 OAuth 2.0 访问令牌能够被客户端无限期的用于访问受保护的资源,直到访问令牌到期或者被撤销访问权限。到期时间可以通过访问令牌响应时发出的访问令牌所附带的参数 “expires_in” 中明确的传递给客户端。

另一方面 IETF 访问令牌响应参数没有规定访问令牌的使用次数。对于某些资源类型 (如支付交易), 访问令牌只能使用一次从安全角度讲是很关键的。Autho4API 授权服务器和访问控制服务器互相同意在后端执行一次使用,但客户端仍然尝试多次重用已发布的访问令牌,这必然对访问控制服务器造成不必要的过载并且增加客户端重新发起认证过程的总时间。

实现一次性使用而无需创建新的访问令牌响应扩展的一种方式是在发放一次性访问令牌时按照惯例规定用于发放一次性访问令牌的网络 API 的范围值。无论何时网络 API 定义这些范围值时,Autho4API 引擎将按以下方式支持一次访问令牌功能:

- 在第一次协议流请求的范围参数中, Autho4API 客户端会只包括此范围值 (即它将不包括多范围值);
- Autho4API 授权服务器不会同时发放一次性访问令牌和更新令牌;

— Autho4API 访问控制服务器在一次性访问令牌被成功地用于访问受保护资源后，会立即使一次性访问令牌永久无效；

— Autho4API 客户端在用一次性访问令牌成功的访问受保护资源后，将立即会丢弃一次性访问令牌。

Autho4API 引擎可以使用本标准没有定义的其他方法支持一次性令牌访问功能。

7.7.1.3 自包含的访问令牌格式

本标准不限制用于构建访问令牌的可能格式，它能够表示一个用于重新获取授权信息的标识符或其它自包含授权信息本身或者一些访问令牌的属性（到期时间，一次性访问令牌指示等），例如增加 Autho4API 访问控制服务器令牌有效性检查的可扩展性，后一种情况下遵循以下规定：

— Autho4API 授权服务器发起自包含的访问令牌时：

- 会使用一种格式保护访问令牌可能携带的任何敏感信息（如与资源所有者隐私相关）的保密性；
- 会使用一种格式保护访问令牌的完整性。

— Autho4API 访问控制服务器检查在受保护资源请求中第一次提交自包含访问令牌的有效性，不会根据访问令牌包含的唯一信息判定它的有效性，而是联合 Autho4API 授权服务器确定它的有效性。

7.7.2 访问令牌类型

Autho4API 引擎会支持 IETF RFC 6750 中定义的非匿名访问令牌类型。

7.7.3 承载令牌

关于发送承载令牌进行授权请求访问受保护资源的方法：

— Autho4API 客户端应该支持“授权”请求消息头域发送访问令牌的方法，参考 IETF RFC 6750 第 2.1 节。Autho4API 客户端可以支持 IETF RFC 6750 定义的其他发送方法；

— Autho4API 访问控制服务器应支持“授权”请求消息头域发送访问令牌的方法，参考 IETF RFC 6750 中第 2.1 节。Autho4API 访问控制服务器可以支持 IETF RFC 6750 中定义的其他发送方法。

关于受保护资源请求中发送承载令牌的传输层安全：Autho4API 访问控制服务器应支持 TLS 1.1（见 IETF RFC4346）和 TLS 1.2（见 IETF RFC5246）。

7.8 多业务提供商环境（消息环境）

7.8.1 概述

本节介绍 Autho4API 引擎多业务提供商环境下的部署选项。这些环境呈现以下特点：

- 一个以上业务提供商提供一组相同的受保护资源；
- 每个资源所有者属于某个业务提供商；
- 一个特定的 Autho4API 客户端可以被隶属于任何业务提供商的资源所有者使用。

有些环境拥有上述特征，它们的共同点是 Autho4API 引擎会使 Autho4API 客户端获得访问任何资源所有者的受保护资源的授权成为可能，无论资源所有者属于哪个业务提供商。

正如在下面的章节中描述的，不同的环境分别依赖于涉及的不同 Autho4API 实体的含义，这些场景可以技术解决。

这些场景既然不是相互排斥的，就可以对它们进行组合。

7.8.2 Autho4API 客户端发现具体的 Autho4API 授权服务器

这种场景呈现除多业务提供商环境下一般特点以外的以下特点：

- 多个为资源所有者服务的 Autho4API 授权服务器属于一个或者一组业务提供商；

— Autho4API 客户端使用发现过程确定 Autho4API 授权服务器服务的相应资源拥有者的 URL 超出了本标准的范围。

如 7.4 节描述,一旦 Autho4API 授权服务器已经确定 Autho4API 客户端成功获得授权,将把重新获取 URL 的授权请求作为发现过程的一部分。

7.8.3 Autho4API 客户端通过单 Autho4API 授权服务器请求授权

7.8.3.1 单个 Autho4API 授权服务器服务多个业务提供商

这种情况下呈现除多业务提供商环境下一般特点以外的以下特点:

单个 Autho4API 授权服务器生成和管理资源拥有者从全部业务提供商那里获取的所有访问令牌。

在这种情况下,Autho4API 客户端将按 7.4 节描述的获得授权,以单个 Autho4API 授权服务器的授权请求为目标。

注:在这种情况下,Autho4API 客户端可以使用发现机制以确定单个 Autho4API 授权服务器的 URL,这超出了本标准的范围。

7.8.3.2 多个 Autho4API 授权服务器服务多个服务提供商

7.8.3.2.1 概述

这种情况下呈现除多业务提供商环境下一般特点以外的以下特点:

— 多个 Autho4API 授权服务器,分别生成和管理属于特定服务提供商的资源拥有者的访问令牌;

— Autho4API 客户端向单个 Autho4API 授权服务器发出授权请求作为入口点,即共享 Autho4API 授权服务器,相应地,Autho4API 授权服务器服务每个特定的资源拥有者;

— Autho4API 客户端在完成 OAuth 流程之前不需要知道服务提供商及相应的资源拥有者的细节。

在这种情况下,Autho4API 客户端接下来如 7.4 节所述的获得授权,以单个 Autho4API 授权服务器的授权请求作为入口点。

注 1:多服务提供商环境的部署场景图如 C.1.所述。

注 2:在这种情况下,Autho4API 客户端可以使用发现机制以确定作为入口点的(共享)Autho4API 授权服务器的 URL,超出了本标准的范围。

7.8.3.2.2 客户端注册

在这种情况下,Autho4API 客户端如 7.1 节所述在(共享)Autho4API 授权服务器注册。

在 Autho4API 客户端注册时,(共享)Autho4API 授权服务器在不同服务提供商的 Autho4API 授权服务器上注册成为 Autho4API 客户端,需要考虑以下情况:

如果 Autho4API 客户端在(共享)Autho4API 授权服务器注册一个重定向 URI,在每个服务提供商的 Autho4API 授权服务器上注册的重定向 URI 将会是带以下查询参数的(共享)Autho4API 授权服务器的 URI 并且包含 Autho4API 客户端注册的重定向 URI,见表 15。

表 15 客户端注册中重定向 URI

名称	类型/值	可选的	描述
client_redirect_uri	xsd:anyuri	不可选	(共享) Autho4API 授权服务器上的 Autho4API 客户端重定向 URL 注册

示例:

https://sharedAuthServerURI.com/?client_redirect_uri=https%3A%2F%2FclientURI.com

7.8.3.2.3 获得授权

7.8.3.2.3.1 概述

获得授权要考虑以下事项：

— 为 Autho4API 客户端提供授权和令牌端点的（共享）Autho4API 授权服务器应能够缓存授权码和访问令牌：

- 当使用合法的授权码的时候，缓存授权码，从而提供令牌端点的 Autho4API 授权服务器能够重定向访问令牌请求到业务提供商合适的 Autho4API 授权服务器。

- 缓存访问令牌，从而作为资源请求入口点的 Autho4API 访问控制服务器能够将它们重定向到服务提供商合适的 Autho4API 访问控制服务器。

— 为 Autho4API 客户端提供授权和令牌端点的（共享）Autho4API 授权服务器应能够重定向资源拥有者的用户代理到服务提供商的 Autho4API 授权服务器。

注：要执行此步骤，提供授权和令牌端口的 Autho4API 授权服务器会发现最终用户的服务提供商，这超出本标准范围。

— 向 Autho4API 客户端提供授权和令牌端点的（共享）Autho4API 授权服务器应能够标识一个重定向 URI，该 URI 是资源拥有者的用户代理在用户认证和授权后返回的。只要 Autho4API 客户端把它包含在最初的授权请求，（共享）Autho4API 授权服务器就包括资源拥有者的用户代理重定向到一个服务提供商的 Autho4API 授权服务器的 URI。仅当 Autho4API 客户端包含它到初始认证请求中时，重定向 URI 将包含在资源所有者代理重定向到服务提供商的 Autho4API 授权服务器的（共享）Autho4API 授权服务器中。在这种情况下，`redirect_uri` URI 是带有‘`client_redirect_uri`’查询参数的（共享）Autho4API 授权服务器的 URI，这个查询参数包含 Autho4API 客户端的重定向 URI，它与 7.8.2.2.1 所述的先前注册所匹配。

— 为 Autho4API 客户端提供授权和令牌端点的 Autho4API 授权服务器应能够缓存以下元素之间的映射：

- 当使用合法授权码时，授权码包含服务提供商的某个 Autho4API 授权服务器；
- 包含服务提供商的某个 Autho4API 授权服务器的访问令牌。

7.8.3.2.3.2 详细的协议流程—授权码

一般的流程修改如图 7 所示。

步骤1) Autho4API 客户端把资源拥有者用户代理指定给（共享）Autho4API 授权服务器的授权端点。

步骤2) （共享）Autho4API 授权服务器发现资源拥有者的服务提供商。有多种方式可以发现服务提供商，如通过网络认证或者其它方法。如果没有一种方式可行，资源拥有者可以选择自己的服务提供商。

步骤3) 一旦确定了服务提供商，（共享）Autho4API 授权服务器把资源拥有者用户代理重定向到服务提供商 X 端点的 Autho4API 授权服务器。（共享）Autho4API 授权服务器通过‘`redirect_uri`’参数提供 URI，这样资源拥有者用户代理以后可以重定向回（共享）Autho4API 授权服务器。提供的 URI 是（共享）Autho4API 授权服务器的 URI，包括 Autho4API 客户端在第一步中标识的查询参数‘`client_redirect_uri`’，‘`client_redirect_uri`’包含‘`redirect_uri`’。

步骤4) 资源拥有者通过身份验证并且允许 Autho4API 客户端访问资源。

步骤5) 服务提供商 X 的 Autho4API 授权服务器响应第三步的请求时把资源拥有者用户代理重定向到第三步中提供的重定向 URI；资源拥有者用户代理发送相应的 HTTP GET 请求到接收到 HTTP 302 响应消息头标识的 URI。

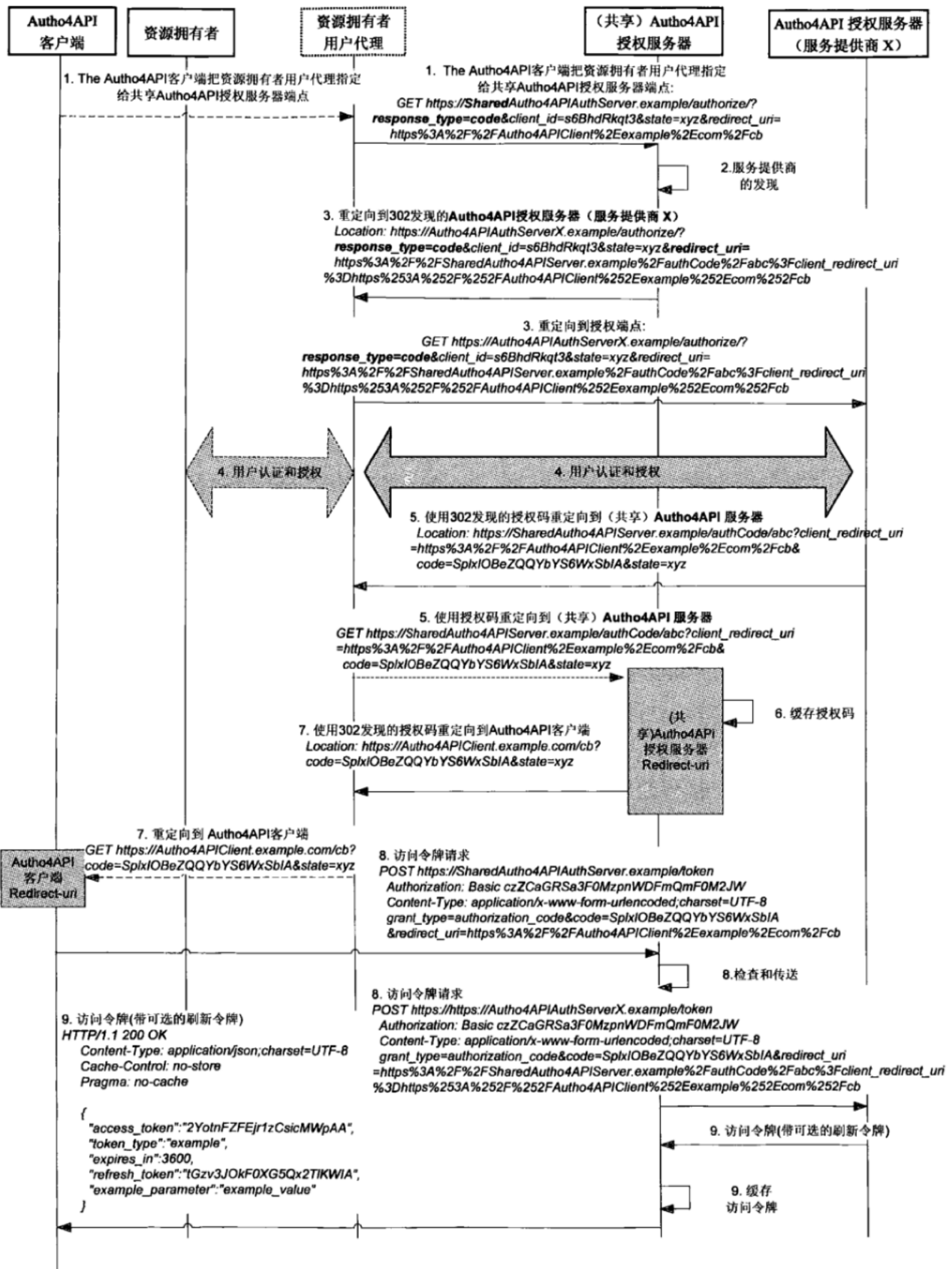


图 7 使用授权码授权形式获得授权——多个 Autho4API 授权服务器服务多个服务提供商的详细协议流程

步骤6) (共享) Autho4API 授权服务器缓存接收到的授权码并且把它与服务提供商 X 关联起来。

步骤7) (共享) Autho4API 授权服务器 (响应第五步中的 GET 请求把资源拥有者用户代理重定向到通过 ‘client_redirect_uri’ 查询参数中接收到的重定向 URI, 此参数跟 Autho4API 客户端在第一步中提供的 ‘redirect_uri’ 参数匹配; 资源拥有者用户代理发送相应的 HTTP GET 请求到接收到 HTTP 302 响应消息头标识的 URI。

步骤8) Autho4API 客户端发送一个访问令牌请求到 (共享) Autho4API 授权服务器。(共享) Autho4API 授权服务器检查授权码的有效性, 它缓存在第六步中并把访问令牌请求分给服务提供商 X 的授权服务器。假如 Autho4API 客户端在访问令牌请求时 ‘包含 redirect_uri’ 参数, (共享) Autho4API 授权服务器会包含访问令牌请求的 ‘redirect_uri’ 参数, 然后把它发送到服务提供商 X 的 Autho4API 授权服务器。在这种情况下, 该 redirect_uri URI 会是带有 ‘client_redirect_uri’ 查询参数的 (共享) Autho4API 授权服务器的 URI, 该查询参数包含 Autho4API 客户端访问令牌请求时提供的 redirect_uri (和 7.8.2.2.1 节所述已注册的及第三步中使用的相匹配)。

步骤9) 服务提供商 X 的 Autho4API 授权服务器响应 Autho4API 客户端, 提供访问令牌和可选的更新令牌。如果有令牌, (共享) Autho4API 授权服务器缓存访问令牌、更新令牌把它们和服务提供商 X 关联起来并且转发响应到 Autho4API 客户端。

7.8.3.2.3.3 详细的协议流程-隐式授权

对于隐私授权, 一般的流程修改如图 8 所。

步骤1) Autho4API 客户端把资源拥有者用户代理指定给 (共享) Autho4API 授权服务器授权端点。

步骤2) (共享) Autho4API 授权服务器发现资源拥有者的服务提供商。有多种方式可以发现服务提供商, 如通过网络认证或者其它方法。如果没有一种方式可行, 资源拥有者被要求选择自己的服务提供商。

步骤3) 一旦确定了服务提供商, (共享) Autho4API 授权服务器把资源拥有者用户代理重定向到服务提供商 X 端点的 Autho4API 授权服务器。(共享) Autho4API 授权服务器通过 ‘redirect_uri’ 参数提供 URI, 这样资源拥有者用户代理以后可以重定向回到 (共享) Autho4API 授权服务器。提供的 URI 是包含查询参数 ‘client_redirect_uri’ 的 (共享) Autho4API 授权服务器的 URI, 该查询参数包含 Autho4API 客户端在第 1 步中指定的 ‘redirect_uri’。

步骤4) 资源拥有者通过身份验证并且允许 Autho4API 客户端访问资源。

步骤5) 服务提供商 X 的 Autho4API 授权服务器响应第 3 步中的请求把资源拥有者用户代理重定向到第 3 步中提供的重定向 URI; 访问令牌作为 URI 片段通过 URI 提供。源拥有者用户代理发送相应的 HTTP GET 请求到接收到 HTTP 302 响应消息头标识的 URI, 不包含 URI 片段。

步骤6) (共享) Autho4API 授权服务器返回到 Web 页面 (典型的是个嵌入脚本的 HTML 文档)。

步骤7) 资源拥有者用户代理执行 Web 托管的 Autho4API 授权服务器提供的脚本, 指示资源拥有者用户代理向 web 服务器发送 URI 片段并且重定向到通过 ‘client_redirect_uri’ 查询参数收到的 URI, 和 Autho4API 客户端在第 1 步中提供的 redirect_uri 相匹配。

a) 资源拥有者用户代理发送访问令牌到 (共享) Autho4API 授权服务器; (共享) Autho4API 授权服务器缓存访问令牌并且与服务提供商 X 关联。

b) 资源拥有者用户代理发送相应的 HTTP GET 请求到脚本中指定的 URI。GET 请求, 该请求不包括

先前的 URI 片段, 并保留在资源拥有者用户代理中。

注: 重定向不能被 HTTP 302 执行, 但在 Java 脚本中容易执行。

步骤8) Autho4API 客户端返回网页 (通常是一个 HTML 文件和一个嵌入式脚本), 它能够访问完整重定向 URI 包括资源拥有者用户代理保留的片段, 并且提取包含在片段中的访问令牌 (和其它参数)。

步骤9) 资源拥有者用户代理本地执行 web 托管的 Autho4API 客户端资源提供的脚本, 提取访问令牌并且传递给 Autho4API 客户端。

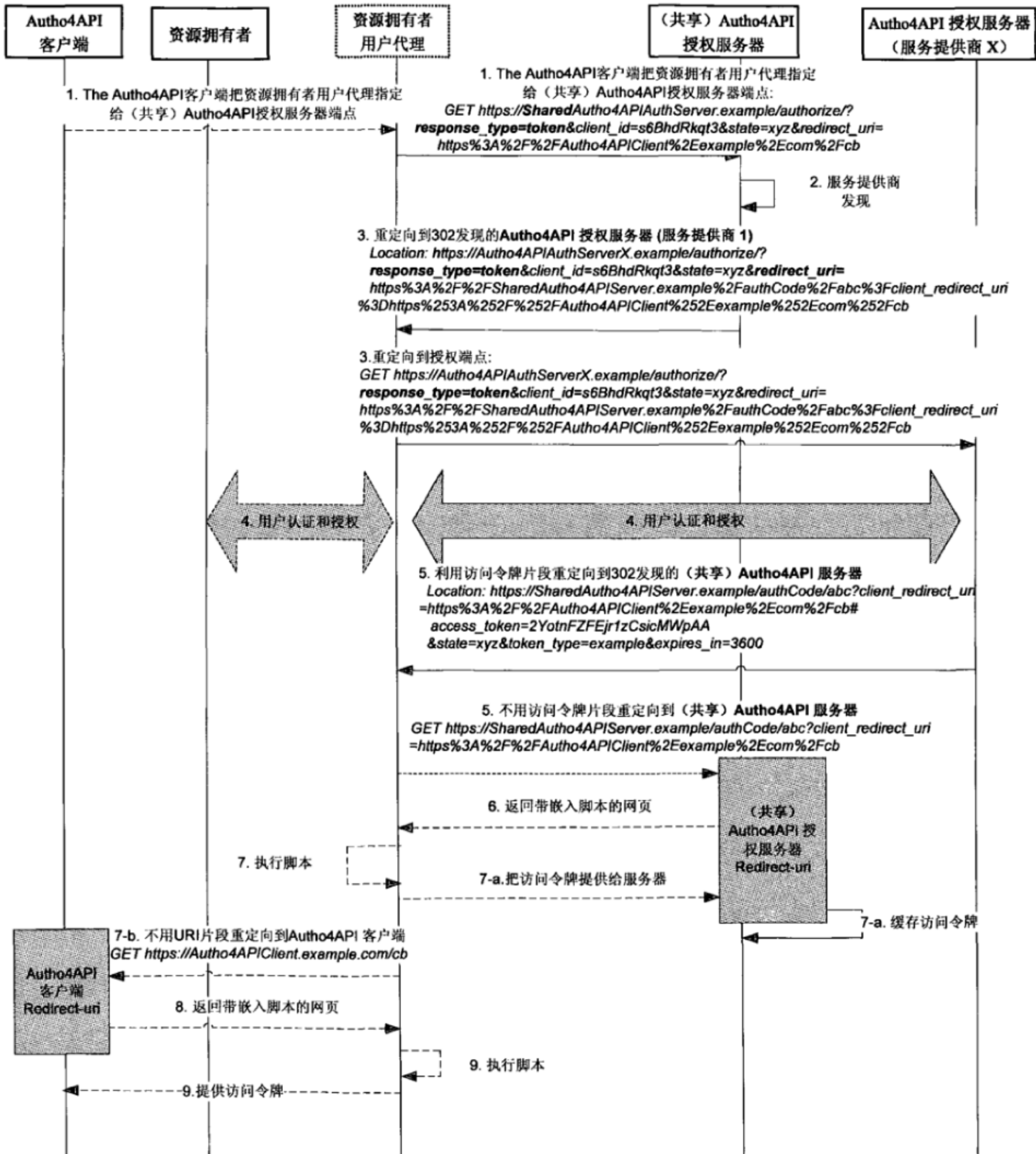


图 8 使用隐式授权形式获取授权——多个 Autho4API 授权服务器服务多个服务提供者的详细协议流程

7.8.3.2.4 访问受保护资源

7.8.3.2.4.1 概述

访问受保护资源要考虑以下事项：

— 作为资源请求入口点的 Autho4API 访问控制服务器，根据缓存的访问令牌，向服务提供商发出访问令牌的 Autho4API 访问控制服务器转发资源请求。

— 另外，如 7.8.2 中所述，作为资源请求入口点的 Autho4API 授权服务器提供资源服务器重定向端点：

- Autho4API 客户端发送包含已发布访问令牌的请求；
- 根据缓存的访问令牌，作为资源请求入口点的 Autho4API 授权服务器，响应访问令牌有效端点的请求，即发放访问令牌的业务提供商的 Autho4API 访问控制服务器的端点。

7.8.3.2.4.2 资源服务器重定向端点

此端点由 Autho4API 授权服务器提供，Autho4API 客户端使用此端点查询服务提供商授权给访问令牌访问权限的资源的端点。

提供的端点通过 URL 提供 REST 资源，该 URL 遵守 OMA REST 通用资源 URL 规定见 OMA REST_NetAPI_Common，资源名称是‘URL Prefixes for Granted resources’，在附录 C 中有定义。

REST 资源通过以下方式被 Autho4API 客户端访问：

— Autho4API 客户端使用 HTTPS 协议向 Autho4API 授权服务器提供的资源服务器重定向端点发送 GET 请求。

— Autho4API 客户端包含授权请求头访问令牌，它是在获得授权步骤中获取的。

— Autho4API 授权服务器检查已接收的访问令牌和 Autho4API 访问控制服务器的关联性（例，相应服务提供商的 Autho4API 访问控制服务器），并且以特定访问控制服务器的端点和访问每个端点的资源列表作为响应：

- 返回的端点是 URL 前缀指的是服务器的根，即：URL 前缀的形式为：主机名+端口+基础路径；例如：http://example.com/ServerX/ResourceExposingPlatform
- 每个返回端点的可访问资源通过使用“范围”参数有选择的匹配，此参数在 Autho4API 客户端获取授权时指定；
- 为了访问实际的资源，Autho4API 客户端采用提供的 URL 前缀并且追加需要的部分。对于 OMA 网络能力开放 API，这意味着 URL 前缀匹配{serverRoot}变量。

7.9 安全考虑

本标准是建立在 IETF RFC 6749 第 10 条和 IETF RFC 6750 第 4 条所描述的安全考虑基础之上，并从以下几个方面扩展它们：

- 验证 Autho4API 客户端和资源的所有者，以减轻伪装攻击；
- 验证 Autho4API 授权服务器，以保证端点的真实性；
- 确保授权码、访问令牌、更新令牌和客户认证凭证的安全传输；
- 确保当辅助通道可用时授权请求响应的安全传输；
- 当选择特定的辅助通道传送机制时客户端侧的安全挑战；
- 多业务提供商环境的特殊的安全挑战。

这些安全方面有详细的部分分别描述它们的功能。

8 OMNA 考虑

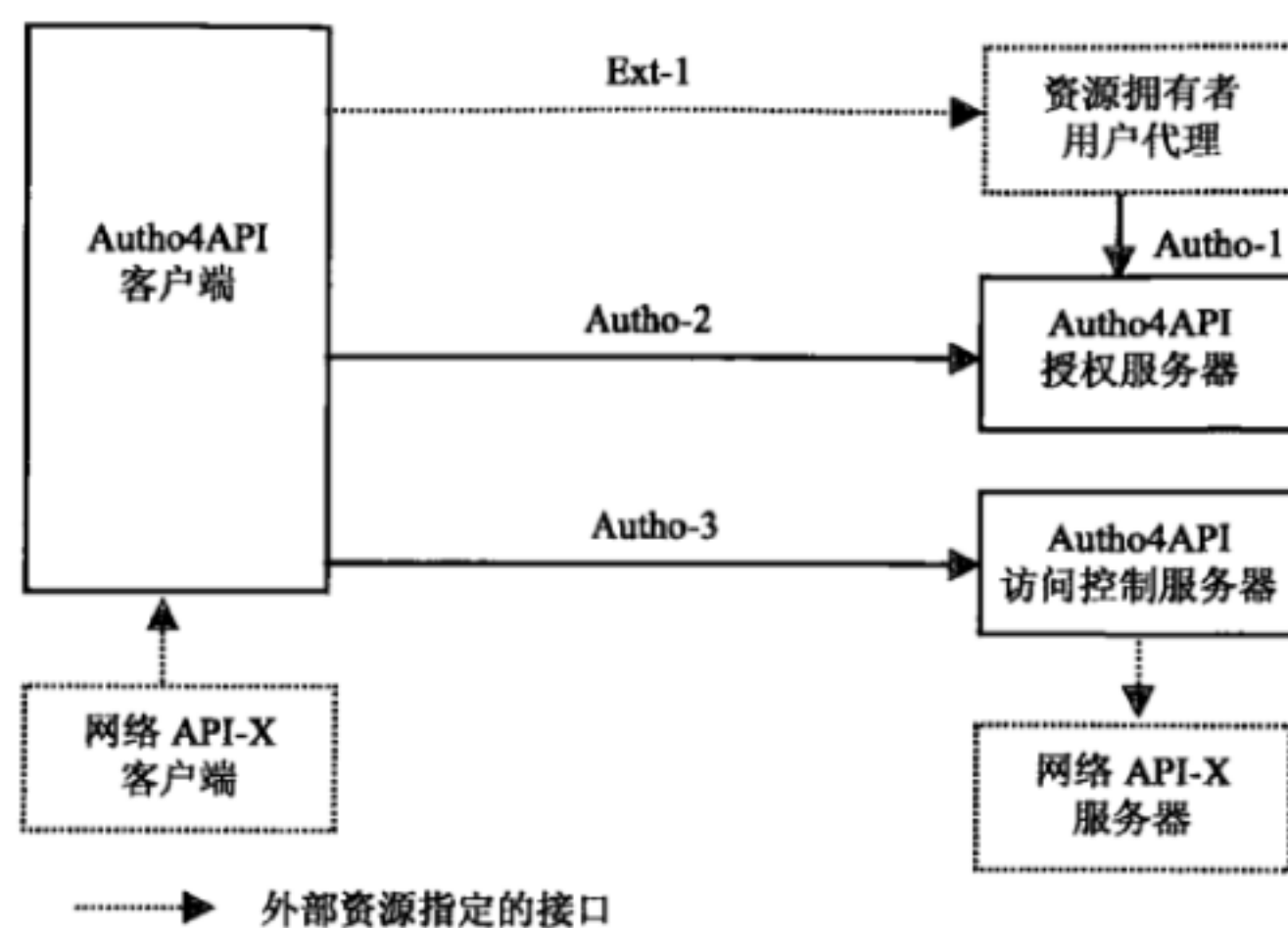
Autho4API 1.0 包括以下 OMNA 项目：

- a) 基于 URN 的命名空间标识符：urn:oma:xml:rest:autho:重定向端点:1（参见附录 C）；
- b) OMNA 注册表：OMNA Author4API 范围值注册表（参见附录 B）。

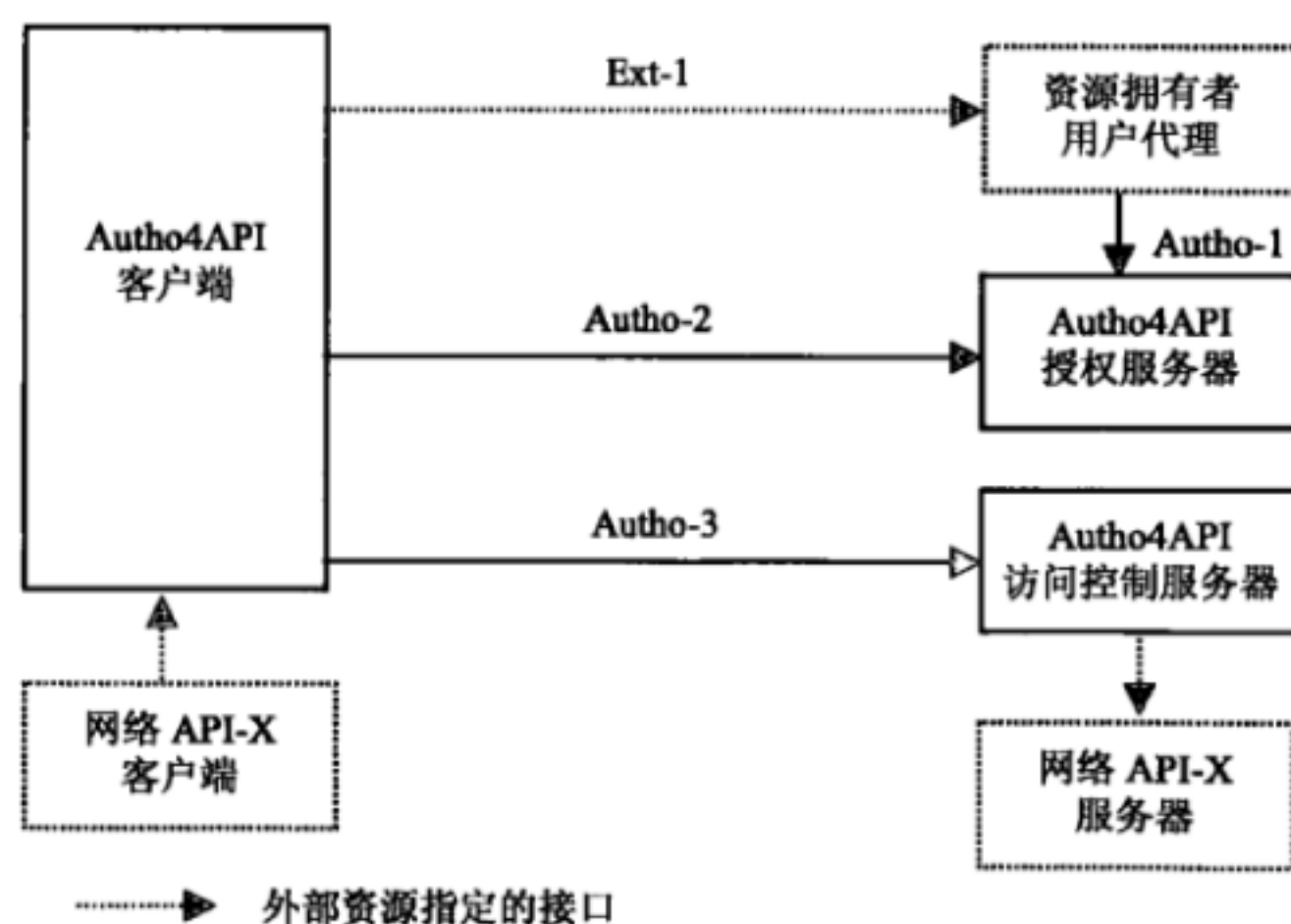
附录 A
(资料性附录)
部署场景

A.1 通用部署场景

图 A.1 和 A.2 给出了可能的部署场景。



图A.1 Autho-3 由 Autho4API 触发

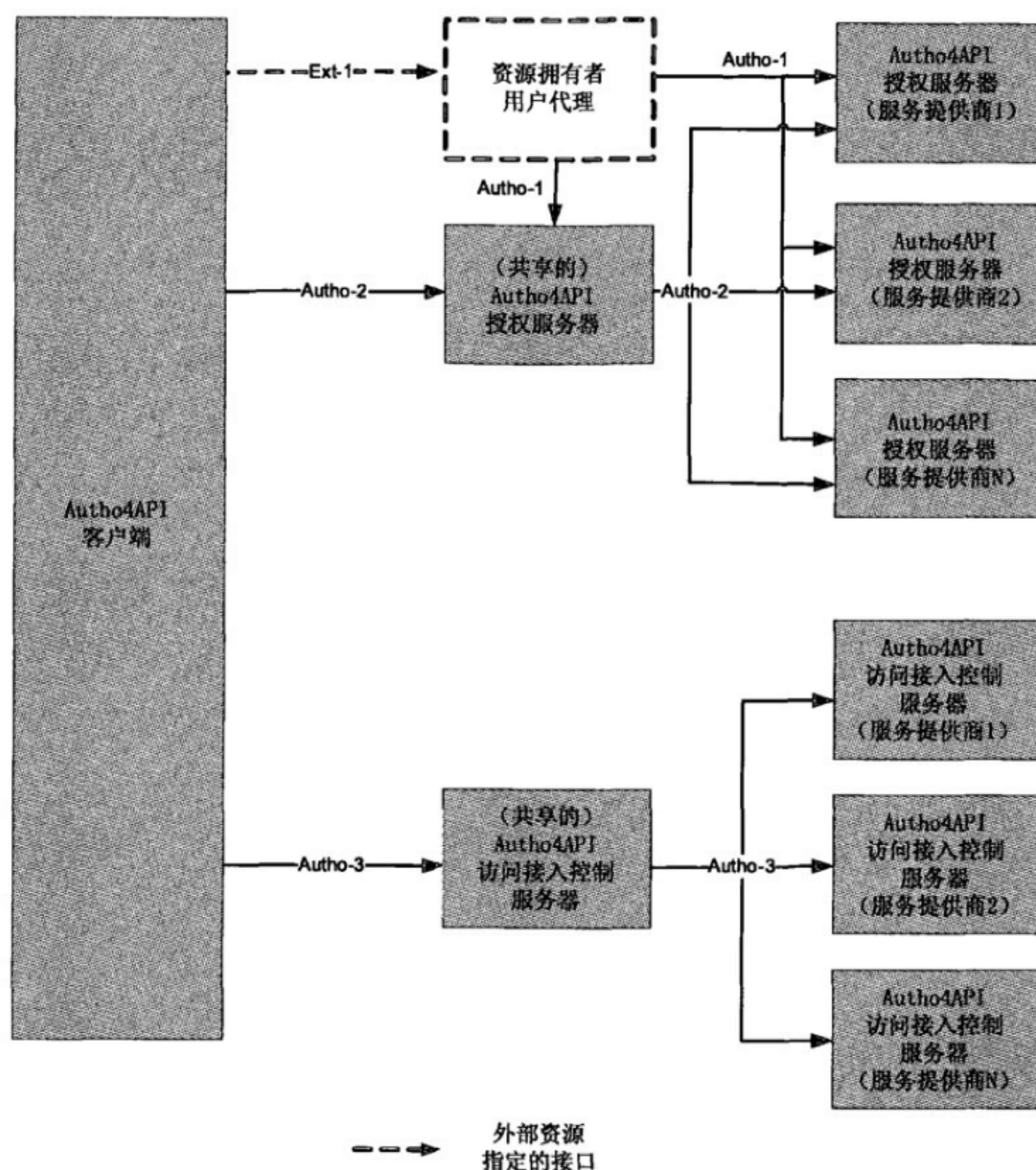


图A.2 Autho-3 由 Net API-X 触发

A.2 多服务提供商场景: Autho4API客户不知道具体服务提供商的情形

A.2.1 多服务提供商场景: Autho4API客户不知道具体服务提供商的部署场景

图A.3给出了Autho4API客户不知道具体服务提供商的部署场景。



图A.3 多个服务提供商的应用场景：Autho4API 不知道具体提供商的情形

A.2.2 上面部署场景的注意事项

这种应用场景不用改变Autho4API在6.2节描述的架构。正如图2里显示及6.3节功能描述的那样，Autho4API授权服务器实体和Autho4API访问控制服务器实体将执行不同的职能，如下所述：

— (共享) Autho4API 授权服务器：是一个具有较少功能的 Autho4API 授权服务器，这些功能只在这种场景中需要被支持：

- **Autho-1 接口的使用**：(共享) Autho4API 授权服务器在与资源所有者用户代理之间进行通信时表现为，Autho4API 授权服务器有意将资源所有者用户代理重定向到某些服务提供商的 Autho4API 授权服务器，再将资源所有者用户代理重定向回到自身，以便缓存授权码/访问令牌，之后将资源所有者用户代理重定向到 Autho4API 客户端。

- **Autho-2 接口的使用**：(共享) Autho4API 授权服务器在与 Autho4API 客户端之间进行通信时表现为 Autho4API 授权服务器，同时，在与某些服务提供商的 Autho4API 授权服务器进行通信时，表现为 Autho4API 客户端。

— 某些服务提供商的 Autho4API 授权服务器：表现为一个通常的 Autho4API 授权服务器。

— (共享) Autho4API 接入控制服务器: 是一个具有较少功能的 Autho4API 接入控制服务器, 这些功能只在这种场景中需要被支持。

Autho-3 接口的使用: (共享) Autho4API 接入控制服务器在与 Autho4API 客户端之间进行通信时表现为 Autho4API 接入控制服务器, 同时, 在与某些服务提供商的 Autho4API 接入控制服务器进行通信时, 表现为 Autho4API 客户端。

— 某些服务提供商的 Autho4API 接入控制服务器: 表现为一个通常的 Autho4API 接入控制服务器。

这种应用场景假定, 在(共享) Autho4API 授权服务器和参与的不同服务提供商的授权服务器是相互信任的关系, 并且(共享) Autho4API 访问控制服务器和那些服务提供商的 Autho4API 访问控制服务器也是相互信任的关系。

附 录 B

(资料性附录)

范围值 (Scope Values) 的定义

B.1 概述

本附录提供了一些Network API范围值 (Scope Value) 定义的一个导则, 这些网络API将会在Autho4API授权框架中使用到。

对于一个给定的Network API Scope值的定义, 应指定如下内容。

- Scope 值的语法和语义;
- 如果需要, 指定 Scope 值的注册过程;
- 构造 URL 端点, Autho4API 客户端发送第一条协议请求 (这条协议请求根据相应流程, 可以是Autho-1 上的授权请求, 或者是 Autho-2 上的令牌请求) 到此端点, 某些特殊情况下, 这个 URL 端点是受限的, 或者不指向一个给定的服务提供商和/或网络 API (机构、API、简介、版本等);
- 在客户端请求中省略 “scope” 参数的语义;
- Downscoping 处理顺序;
- 当 “scope” 参数中包含多个 Scope 值, 且其中至少有一个值是定义来用于一次性访问令牌的处理, 指定返回的 scope;
- Scope 值定义 (专用、包含、重载) 之间的关系;
- 是否支持 Downscoping, 如果支持的话, 在 “包含” 定义的情况下 Downscoping 如何应用;
- 对于每一个 Scope 值:
 - 映射到网络 API 资源的授权操作;
 - 这个 Scope 值是否用于处理一次性的访问令牌;
 - 在过期时间方面的建议。

B.2 注意事项

B.2.1 访问范围 (scope) 的表示

由分发的访问令牌代表的实际访问权限范围 (scope) 具有如下功能特点:

- 标识服务提供商, 此服务提供商通过网络 API 开放其网络资源。
- 在服务提供商的 scope 中, 标识资源拥有者。
- 在服务提供商的 scope 中, 标识 Autho4API 客户端。
- 网络 API 标识, 例如:
 - 定义此 API 的机构;
 - API 本身 (区别于此机构定义的其他 API);
 - API 的简介 (如果适用的话);
 - 版本。
- 在 Network API 的 scope 中, 标识 (REST 操作的) 资源。

在访问范围 scope 定义中, 可以发挥作用的第一个元素是第一个协议请求所访问的 URL 端点, 这个 URL 端点是:

— 对于引入了明确资源所有者授权的流程（如授权码流程），授权请求通过 Autho-1 接口被发送到授权 URL 端点；或者

— 对于其他流程，请求令牌通过 Autho-2 接口被发送到令牌 URL 端点。

这个 URL 在本附录中简称“endpoint URL”，它可以绑定一些 scope 信息（如服务提供商和网络 API 标识），例如：

— 当 Autho4API 客户端通过发现机制获得 URL 端点时，动态的绑定一些 scope 信息，如在客户端侧提供的或由服务器侧判断的服务提供商或者网络 API 标识。

— 当 URL 端点有公开的构造时，静态的绑定用于传送服务提供商或网络 API 的 URL 组件等。

对于给定的 URL 端点，Autho4API 客户端可以通过协议中第一个请求来缩小访问范围 scope：

— OAuth 的“scope”参数，这个参数列出了大量 Scope 值；

— 一些端点扩展参数。

端点 URL 绑定越多的 scope 信息，则需要传输的 Scope 值就越少。一些相反的例子如下：

对于一个不绑定任何 scope 信息的 URL 端点（例如被多个服务提供商共享的一个 URL，每个服务提供商提供多 API），第一个协议请求需要通过 Scope 值和扩展参数来传输服务提供商和网络 API 的标识（机构、API、版本）。

对于一个绑定了很多 scope 信息的 URL 端点（例如一个 URL 指定一个服务提供商，一个服务提供商只提供一个版本的单一 API），第一个协议请求只需要包含简单的 Scope 值来标识资源，甚至对于非常简单的 API，Scope 值完全不需要。

B.2.2 语法和语义

B.2.2.1 范围（Scope）值的语法

Scope 值在 scope 参数中被编码为一个由空格分隔的列表，scope 参数可以被一个编码的百分数表示，位于：

- 授权请求中；
- 访问请求中；
- 明确授权流程中的令牌响应中；
- 在对受保护资源请求后的失败响应中，此时访问令牌表示 Scope 是不够的。

scope 参数还可以表示为一个 JSON 字符串，位于：

- 在除了明确授权流程以外的其他流程中的令牌响应中。

因此，对构造 Scope 值的字符组，理论上不存在限制，Scope 值可以是：

- 任意文本值；
- 符合众所周知的文本格式的文本值（URN、URI、JSON object 等等）。

超长的 Scope 值可能会导致传输问题，特别是当 scope 参数作为 URI 查询参数的时候（例如，授权请求）。

B.2.2.2 范围（Scope）值的语法

IETF RFC 6749 中不限制在 Scope 值中所传输内容的语法：

— 通常它包含一个字符串令牌，按照惯例来标识一系列对某些网络资源的授权操作。它也可以包含资源的 scope 信息，如网络 API 标识。

— 另外, 为了避免字符串令牌语义的公开, 一个 Scope 值可以包含用来标识被授权访问请求的资源 URI, 但这种技术也有它的不足之处:

- 对于一个开放许多资源的 API, scope 参数的大小会增加得非常快 (例如, 包含许多资源 URI 的列表);
- 对与这个资源 URI, 不能给予高于“完全访问权限”的其他访问权限。

B.2.3 重载和冲突定义

B.2.3.1 重载Scope值

对于给定的一个API, 在资源 (操作上) 和Scope值之前的映射可以是:

- 专用: 每一个资源最多映射到一个 Scope 值;
- 重载: 一些资源被映射到几个 Scope 值;
- 包含: 一些资源被映射到几个 Scope 值, 但是在这种情况下, Scope 值之间有一个严格的超集/子集关系。

这些映射模型都是可用的, 带有保留值的重载 Scope 表示在 downscoping 功能方面的限制 (见 B.2.5 节)。

B.2.3.2 冲突的范围 (Scope) 值

冲突范围 (Scope) 值在如下情况会发生碰撞:

- 不同的 API (被相同或不同机构定义的) 定义了相同的 Scope 值;
- 对于这些不同的 API, 它们的授权请求使用相同的端点。

在这种情况下, Autho4API 授权服务器可能会错误的授予资源访问权限, 而这些资源并不是客户端实际所请求的。

B.2.3.3 API版本

另一种冲突情形会发生在如下情况下:

- 在 Autho4API 客户端和服务端中部署了同一 API 的不同版本;
- 一些 Scope 值在一个 API 的不同版本中被重用, 但是 Scope 值和资源的映射发生了改变;
- Scope 值不传输 API 版本信息;
- URL 端点没有被绑定到特定的 API 版本。

在这种情况下, Autho4API 授权服务器可能会授予过少或过多的资源访问权限, 这些资源访问权限少于或多于客户端实际所请求的。

B.2.4 省略请求Scope

在第一个请求协议中缺少Scope, 可以得到如下语义:

- a) 不允许服务提供商授权 scope 参数加入到这个 URL 端点;
- b) 允许请求 Scope 拥有对这个 API 的某些资源的基本访问能力, 这要依据所发布的约定;
- c) 允许请求 Scope 拥有对这个 API 的所有资源的完全访问能力。

允许缺失的 Scope 可以与正常定义的 Scope 共存。作为一个例子, 在上述第 3 种情形的请求中, 可能发生 downscoping, 即响应中可能包含一个 scope 参数, 此参数中列出了正常定义的 Scope 值。

允许缺失的Scope可能会限制URL端点来传输Scope信息 (例如, 用于一个单一API的授权请求的URL)。

范围缩编

B.2.4.1 规则

在包括特定授权范围的 Autho4API 客户端请求发生以后，会发生以下一个或者两个步骤：

- 授权服务器能基于比如安全注意事项，缩小所请求的授权范围；
- 当资源拥有者被提示授权时，可以缩窄请求范围。

根据授权范围值如何定义，范围缩编操作可以为 Autho4API 客户端带来：

- 当每个授权范围值映射到资源的功能组时，授出较少的功能性访问；
- 当每个授权范围值映射到相同资源的特定特权操作时，授出较少的特权访问。

因此，范围缩编可以帮助减少泄露的承载令牌造成的损失。

Autho4API 授权服务器可以在传递访问令牌的响应当中包含一个‘授权范围’参数，从而向客户端发出消息以通知范围缩编已经发生。这个响应是：

- 对于隐式的授权流程，在授权请求之后的令牌响应；
- 对于其它流程，在令牌请求之后的令牌响应。

注：因此对于授权码流程，在授权请求（其中包括所请求的授权范围）的响应中不包含范围缩编通知，而是稍后包含在令牌请求中。

B.2.4.2 处理规则

Autho4API 授权服务器的范围缩编是发生在资源拥有者的范围缩编之前或者之后会影响最终用户的体验。以 Autho4API 客户端请求“读写”访问范围和服务器打算将范围缩编为“读”为例：

- 服务器在资源拥有者授权之前进行范围缩编；然后资源拥有者被提示给应用授予“读”访问权限；或者
- 资源拥有者给应用授予“读写”访问权限；然后服务器将此授权范围缩编为只“读”，稍后资源拥有者可能会提出疑问为什么应用程序只能对资源进行只读访问。

B.2.4.3 返回的授权范围值

IETF RFC 6749 中不会将缩编而来的授权范围限制为所请求授权范围值的精确子集。另一方面，范围缩编受授权范围值之间相互关联的方式约束。特别地：

- 在排他性定义的情况下，较窄的授权范围可以由请求中所列出值的精确子集组成（如请求“短信彩信”，返回“短信”）；
- 在包含性定义的情况下，较窄的授权范围可以由其他没有在请求中列出的值组成（如请求“通讯”，返回“短信”）；
- 在重叠定义的情况下，不能消息通知较窄的授权范围，授权请求不得被拒绝。

在任何情况下，了解返回的授权范围对于 Autho4API 客户非常关键，以便进一步访问资源时表现一致。特别地，试图使用授权范围不足的访问令牌对受保护的资源进行访问，会导致很差的终端用户体验和 Autho4API 服务器过度超载。

B.2.5 授权范围值的特征

B.2.5.1 有效期管理

通过在令牌响应中加入“expires_in”参数，Autho4API 授权服务器可以选择性地向 Autho4API 客户通知发出的访问令牌有一定的寿命。因此，支持此参数的客户可以预见访问令牌何时失效和避免发送含

过期令牌的资源请求。

这个访问令牌还另外发布给由 Autho4API 客户端在授权过程中协商的一系列给定的授权范围值，这意味着每一个所“授予”的授权范围值都附加了相同的有效期。

最后，Autho4API 授权服务器可能已经在内部定义了一些规则用于计算已发布的访问令牌的有效期，例如这基于：

- Autho4API 客户端的类型（公开的或机密的）；
- 授权批准的类型（隐式的、授权码等）；
- Autho-1 上的终端身份验证方法（如适用），因为有些比其他更安全；
- Autho-2 上的客户端身份验证方法（如适用）；
- 这是第一次发出访问令牌，或是访问令牌更新；
- 资源和对这些资源的操作，即授权范围值。

根据 Autho4API 的授权服务器策略，每一个访问令牌可以被发布为可变的有效期，这取决于在授权过程中由 Autho4API 客户端协商的授权范围值列表。例如，如果此列表包含：

- 为重要访问权限定义的授权范围值，访问令牌将被发布为短暂的生命周期（如几分钟）；
- 为非关键访问权限定义的授权范围值，访问令牌将会被发布为长生命周期（如几周）；
- 为重要访问权限定义的授权范围值和为非关键访问权限定义的授权范围值，访问令牌很可能会发布为短暂的生命周期（例如，出于安全原因）。

最后一点提出了一个问题，即使是 Autho4API 客户端只打算以后访问非关键资源，到期的访问令牌也不会很快被使用。因此：

- 当更新令牌不可用（如隐式的授权流程）或者没有被 Autho4API 授权服务器发布，资源拥有者将被频繁地用于进行授权更新；
- 当使用更新令牌时，Autho4API 授权服务器将被频繁地请求更新访问令牌。

此问题是由 OAuth2.0 协议的现有限制引起的：授权批准只可以用于交换单一的访问令牌。

网络 API 的一个可能的变通方案是为每个授权范围值指定一些数量级的访问令牌生命周期，这样使 Autho4API 客户端在每一组具有相似生命周期的授权范围值中请求一个特定的授权。然而如本节前面所述，由 Autho4API 授权服务器选择的最终到期时间可能会考虑其他参数，而不考虑授权范围值。

B.2.5.2 一次性访问令牌

通过设计，访问令牌可以多次使用，直到其到期或被撤销，OAuth2.0 协议并没有定义任何令牌响应参数为发布的访问令牌声明有限的使用次数。

但仍可能通过文件明确指出一个授权范围值只被定义为发布一次性访问令牌。Autho4API 客户端考虑到这一指示，可避免尝试多次使用所获得的访问令牌。

在实践中，当授权范围值用于发布一次性访问令牌时，一个 Autho4API 客户端将在‘scope’参数中只列出一个授权范围值。但当客户端在‘scope’参数中列出如下项目时，值得阐明 Autho4API 授权服务器授予什么授权范围：

- 为一次性访问令牌定义的几个授权范围值；
 - 为一次性访问令牌定义的一个授权范围值，以及为多次访问令牌定义的多个授权范围值。
- 一些可能的行为可能会拒绝请求，或者可能会将范围缩编为一次性访问令牌定义的一个授权范围值。

B.2.6 授权范围值的颗粒度

对于一个资源的访问粒度，理论上，一个给定的网络 API 的授权范围值定义可以从完全没有范围值定义，到为每个 API 资源上的每个允许操作定义一个范围值。在实践中，颗粒度需要一个权衡：

— 如果访问的粒度太粗（即授权范围值太少），以下将可能不可用：

- 当 API 包含功能上明显分组的资源时，有效的特征驱动的范围缩编；

- 当 API 恰好对相同的资源定义了不同的访问权限时，有效的安全驱动的范围缩编。特别是，Autho4API 授权服务器可能会强制将 Autho4API 客户端的安全驱动范围缩编认定为“公共的”，而不是“机密的”（按照 IETF RFC 6749 定义）；

- 访问令牌有效期的精细化管理（见 B.2.6.1 节）。

— 如果访问颗粒度过细（即授权范围值过多），那么：

- 在提交给资源拥有者的授权请求作为一个检查开关选项代表每一个授权范围值的情况下，由于用户界面的复杂性，资源拥有者可能会不堪重负；

- Autho4API 客户端开发被复杂化。

附录 C

(资料性附录)

“授予资源的 URL 前缀”资源的定义

C.1 资源概述

本附录用来提供对“被授予资源的 URL 前缀”资源的深入理解，定义如下的[OMA REST_NetAPI_Common 建议。本附录指明了资源、数据结构的定义，以及对资源所允许的所有操作的定义。

表 C.1 给出了对“被授予资源的 URL 前缀”资源及其表现的数据类型和所允许的 HTTP 方法的详细综述。

目的：为授予给一个给定访问令牌的资源找回 URL 前缀

表 C.1 “被授予资源的 URL 前缀”资源及其表现的数据类型和所允许的 HTTP 方法

资源	URL 基础URL: https://{serverRoot}/ autho4api/{version}	数据结构	HTTP verbs			
			GET	PUT	POST	DELETE
被授予资源 的URL前缀	/resourcesURLPrefixes	RedirectEndpointList	为授予给一个访问 令牌的资源读取一 个或多个URL前缀	无	无	无

C.2 数据类型

C.2.1 XML命名空间

“被授予资源的 URL 前缀”资源的命名空间是：

urn:oma:xml:rest:autho:redirectEndpoint:1

本标准中使用的“xsd”命名空间是指在 XMLSchema1, XMLSchema2 中定义的 XML Schema 数据类型。本标准中使用的“common”命名空间是指在 OMA REST_NetAPI_Common 中定义的数据类型。“xsd”和“common”两个名称的使用在语义上并不重要。

需要使用 7.4 节定义的资源服务器重定向端点的应用应使用如下命名空间：
urn:oma:xml:rest:autho:redirectEndpoint:1。

C.2.2 结构

C.2.2.1 概述

本节定义了“被授予资源的 URL 前缀”资源中使用的数据结构。

一些数据结构可以被实例化为所谓的根元素。

C.2.2.2 类型：RedirectEndpointList

端点清单见表 C.2。

表 C.2 端点清单

元素	类型	可选	描述
端点	端点[1..未绑定的]	否	一组与已提供的访问令牌相关的端点

一个名为 redirectEndpointList 的 RedirectEndpointList 类型的根元素，被允许出现在响应主体中。

C.2.2.3 类型：Endpoint

个别端点见表 C.3。

表 C.3 个别端点

元素	类型	可选	描述
url	xsd:anyURI	否	URL前缀参考开放资源的根服务器，即，URL前缀有这种形式：hostname+port+base path。 例如：http://example.com/ServerX/ResourceExposingPlatform
授权范围	xsd:string [0..unbounded]	是	授权范围值列表参考授予指定的访问令牌的资源，该令牌在前述参数所指明的URL前缀中， 如果这个参数没有出现，这意味着前述参数所指明的URL前缀公开了所有授予给访问令牌的资源。如果提供了超过一个端点，该“scope”参数应出现在每一个端点中

C.3 详细的资源规范

C.3.1 概述

以下内容适用于本附录定义的资源，与表示格式（即 XML、JSON 格式）无关：

— 在 URL 变量中的保留字符（由波形括号中的名称表示的下述 URL 的部分）应根据 IETF RFC3986 进行百分比编码。请注意，无论 URL 是用作请求 URL 或用在资源的表示里（如在“端点”元素里），这个规则总是适用。

— 对于有主体的响应，以下情况是适用的：根据在 REST_NetAPI_Common 中指定的内容类型协商的结果，服务器应在响应主体中返回 JSON 或 XML 编码参数。JSON 表示的产生和处理应遵循在 HTTP 请求/响应中的 JSON 编码规则，正如 REST_NetAPI_Common 中所规定的。

C.3.2 资源：‘授出资源的URL前缀’

C.3.2.1 概述

使用的资源是：

https://{serverRoot}/autho4api/{version}/resourcesURLPrefixes/

此资源是用作对服务提供者公开的已授予访问令牌的资源的端点进行轮询。

C.3.2.2 Request URL变量

以下请求 URL 变量对于所有 HTTP 命令是通用的，见表 C.4。

表 C.4 URL 变量

名字	描述
serverRoot	服务器基本url：主机名+端口+基本路径。端口和基本路径是可选的。例如： example.com/exampleAPI 注：ServerRoot将和在章节7.2.1和7.2.2中分别定义的授权ServerRoot和令牌端点相同
Version版本	API客户端想使用的版本，该变量的值为1

参见 C.3 节中关于 URL 变量中转义的保留字符的声明。

C.3.2.3 响应编码和错误处理

对于 HTTP 响应编码，参见 OMA REST_NetAPI_Common。

当提出请求一个已过期的、撤销的、不能识别的、或者无效的访问令牌时，根据 IETF RFC 6750 的 2.4.1 节中承载令牌的定义，应响应 HTTP 401（未授权的），和错误代码‘invalid_token’。

C.3.2.4 GET

C.3.2.4.1 概述

此操作是用作对服务提供者公开的已授予访问令牌的资源的端点进行可靠的检索。

Request URL 参数: 无。

对端点的轮询请求应包含访问令牌, 采取与第 7.7 节中规定的访问受保护资源相同的方式。

C.3.2.4.2 例1: 检索授予给定的Token. XML访问请求资源的端点

C.3.2.4.2.1 Request

以下是一个对此端点进行请求的例子:

```
GET exampleAPI/autho4api/v1/resourcesURLPrefixes HTTP/1.1
Accept: application/xml
Host: server.example.com
Authorization: Bearer vF9df4qmT
```

C.3.2.4.2.2 Response

以下是一个该端点对先前的请求有效响应的例子:

```
HTTP/1.1 200 OK
Content-Type: application/xml
Content-Length: nnnn
Date: Thu, 18 Nov 2011 02:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<autho:redirectEndpointList xmlns:autho="urn:oma:xml:rest:autho:redirectEndpoint:1">
  <endpoint>
    <url>https://Autho4APIAccessControlServer_1.serviceproviderExample.com</url>
    <scope>oma_rest_messaging.in_regist</scope>
    <scope>oma_rest_messaging.in_subscr</scope>
  </endpoint>
  <endpoint>
    <url>https://Autho4APIAccessControlServer_2.serviceproviderExample.com</url>
    <scope>oma_rest_messaging.out</scope>
  </endpoint>
</autho:redirectEndpointList>
```

C.3.2.4.3 例2: 检索为给定的无效访问令牌授予资源的端点

C.3.2.4.3.1 Request

以下是一个对此端点进行请求的例子:

```
GET exampleAPI/autho4api/v1/resourcesURLPrefixes HTTP/1.1
```



```
Accept: application/xml
Host: server.example.com
Authorization: Bearer FakevF9dft4qmT
```

C.3.2.4.3.2 Response

以下是一个该端点对先前的请求有效响应的例子：

```
HTTP/1.1 401Unauthorized
Date: Thu, 18 Nov 2011 02:51:59 GMT

WWW-Authenticate: Bearer realm=" server.example.com ",
                  error="invalid_token",
                  error_description="The Access Token is not valid"
```

C.3.2.4.4 例3：检索为给定的Token. JSON访问请求授予资源的端点。

C.3.2.4.4.1 Request

以下是一个对此端点进行请求的例子：

```
GET exampleAPI/autho4api/v1/resourcesURLPrefixes HTTP/1.1
Accept: application/json
Host: server.example.com
Authorization: Bearer vF9dft4qmT
```

C.3.2.4.4.2 Response

以下是一个该端点对先前的请求有效响应的例子：

```
HTTP/1.1 200 OK
Content-Type: application/json
Content-Length: nmmn
Date: Thu, 18 Nov 2011 02:51:59 GMT

{"redirectEndpointList": {"endpoint": [
  {
    "scope": [
      "oma_rest_messaging.in_regist",
      "oma_rest_messaging.in_subscr"
    ],
    "url": "https://Autho4APIAccessControlServer_1.serviceproviderExample.com"
  },
  ...
]
},
...
}
```



```
{  
    "scope": "oma_rest_messaging.out",  
    "url": "https://Autho4APIAccessControlServer_2.serviceproviderExample.com"  
}  
}}
```

C.3.2.5 PUT

资源所不允许的类函数。返回的 HTTP 错误状态是 405。服务器也应按照 IETF RFC 2616 的第 14.7 条的规定在响应中包含‘Allow: GET’字段。

C.3.2.6 POST

资源所不允许的类函数。返回的 HTTP 错误状态是 405。服务器也应按照 IETF RFC 2616 的第 14.7 条的规定在响应中包含‘Allow: GET’字段。

C.3.2.7 DELETE

资源所不允许的类函数。返回的 HTTP 错误状态是 405。依据 IETF RFC 2616 的 14.7 节，服务器也应该在响应中包含 ‘Allow: GET’ 字段。

参 考 文 献

OMA AUTHO4API_SUP_Resource_ServerOMA Autho4API 的 XML schema (XML schema for Autho4API for the URL Prefixes for Granted Resources)

OMADICT OMA 词典规范 (Dictionary for OMA Specifications)

OMAPUSHOMA 推送 (OMA Push)

OMA REST_NetAPI_Common OMA RESTful 网络 API 公共定义 (Common definitions for OMA RESTful Network APIs)

WAC2.1 WAC 规范 2.1 (WAC Specifications 2.1)

W3C XMLSchema1 W3C 建议 XML 语义结构 (W3C Recommendation, XML Schema Part 1: Structures Second Edition,)

W3C XMLSchema2 W3C 建议 XML 数据类型 (W3C Recommendation, XML Schema Part 2: Data types Second Edition)

中 华 人 民 共 和 国
通 信 行 业 标 准
移动互联网应用编程接口的授权技术要求
YD/T 2912-2015

*

人民邮电出版社出版发行
北京市丰台区成寿寺路1号邮电出版大厦
邮政编码：100164
北京康利胶印厂
版权所有 不得翻印

*

开本：880×1230 1/16 2015年12月第1版
印张：3.75 2015年12月北京第1次印刷
字数：96千字

15115·831

定价：40元

本书如有印装质量问题，请与本社联系 电话：(010)81055492