

ICS 35.030

CCS L 80

备案号：



中华人民共和国密码行业标准

GM/T XXXX—XXXX

自动化证书管理协议

Automatic Certificate Management Protocol

（草案）

在提交反馈意见时，请将您知道的相关专利连同支持性文件一并附上。

XXXX-XX-XX 发布

XXXX-XX-XX 实施

国家密码管理局 发布

目 次

前 言	IV
引 言	V
1 范围	1
2 规范性引用文件	1
3 术语和定义	1
4 符号和缩略语	3
4.1 符号	3
4.2 缩略语	3
5 协议总体流程	3
5.1 账户创建阶段	4
5.2 证书签发阶段	4
5.3 证书撤销阶段	5
6 报文安全传输	6
6.1 概述	6
6.2 字符编码	6
6.3 HTTPS 请求	6
6.4 请求身份鉴别	6
6.5 GET 和 POST-as-GET 请求	8
6.6 请求 URL 完整性	8
6.7 重放保护	9
6.7.1 重放随机数	9
6.7.2 “随机数” (Nonce) JWS 头参数	10
6.8 速率限制	10
6.9 错误	10
7 证书管理协议	12
7.1 概述	12
7.2 资源	12
7.3 获取随机数	21
7.4 帐户管理	22
7.5 申请签发证书	28
7.6 用户身份核实	37
7.7 证书撤销	52
7.8 续订信息	54
7.8.1 获取续订信息	54
7.8.2 RenewalInfo 对象	54
7.8.3 订单对象的扩展	55
附 录 A （规范性） 编码及元数据规范	57

A.1 媒体类型: application/pem-certificate-chain	57
A.2 HTTP 挑战的已知 URI	57
A.3 Replay-Nonce 重放随机数 HTTP 头	57
A.4 “url” JWS 头参数	57
A.5 “nonce” JWS 头参数	58
A.6 ACME 的 URN 子命名空间(urn:ietf:params:acme)	58
A.7 新注册表	58
A.7.1 账户对象中的字段	58
A.7.2 订单对象中的字段	59
A.7.3 授权对象中的字段	59
A.7.4 RenewalInfo 对象中的字段	60
A.7.5 错误类型	60
A.7.6 资源类型	60
A.7.7 目录对象中“元”对象中的字段	61
A.7.8 主体标识类型	61
A.7.9 验证方法	61
附录 B (资料性) 典型应用场景	63
附录 C (资料性) 安全注意事项	64
C.1 概述	64
C.2 威胁模型	64
C.3 授权的完整性	65
C.4 拒绝服务注意事项	66
C.5 服务端请求伪造	66
C.6 CA 策略注意事项	67
附录 D (资料性) 操作注意事项	69
D.1 概述	69
D.2 密钥选择	69
D.3 DNS 安全	69
D.4 令牌熵(Token Entropy)	69
D.5 格式错误的证书链	70
参考文献	71

前 言

本文件按照GB/T 1.1—2020《标准化工作导则 第1部分：标准化文件的结构和起草规则》的规定起草。

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专利的责任。

本文件由密码行业标准化技术委员会提出并归口。

本文件起草单位：

本文件主要起草人：

引 言

数字证书是构建网络空间信任体系、保障数据传输安全的核心密码基础设施。随着网络与信息系统规模的不断扩大，数字证书的部署数量呈指数级增长，传统依赖人工交互的证书申请、验证、签发与部署模式，不仅流程繁杂、效率低下，且极易因人为疏漏导致证书过期，从而引发严重的安全事故与业务中断。

为解决上述问题，建立标准化的自动化证书管理机制已成为行业共识。国际互联网工程任务组(IETF)发布了 ACME (Automatic Certificate Management Environment) 协议 (RFC 8555)，以实现数字证书全生命周期的自动化管理。然而，原生的国际标准未包含对我国商用密码算法及传输层密码协议(TLCP)的原生支持。

本文件的制定旨在国际 ACME 协议框架的基础上，深度融合我国商用密码技术体系。本文件规范了基于商用密码算法（如 SM2、SM3）和密码协议（如 TLCP、基于 SM2 的 JWS 签名封装等）的自动化证书申请、域名/主体标识验证、证书签发与撤销等安全交互流程。

本文件在技术路线上充分论证了引入国密算法的必要性与合理性：一方面，在我国商用密码应用合规要求的背景下，数字证书的自动化管理必须支持SM2/SM3等国密算法；另一方面，本文件并非简单替换国际算法，而是在协议框架的各关键环节（身份鉴别、签名封装、证书格式等）系统性地融入国密技术体系，实现了与国际ACME协议在语法和语义层面的兼容。通过统筹设计账户管理、域名验证、证书签发与撤销等全流程的国密化改造，本文件为各级证书认证机构（CA）、云服务提供商、网络设备与安全网关厂商提供了统一的自动化证书管理技术规范。

本文件的发布与实施，对于推动商用密码在 HTTPS/TLCP 加密通信、零信任网络等广泛场景下的规模化、自动化应用具有重要意义。

自动化证书管理协议

1 范围

本文件规定了自动化完成域名验证和证书签发的交互协议，包括协议总体流程、报文安全传输和证书管理协议等。

本文件适用于证书认证系统中TLS证书的自动化管理，实现PKI系统自动化完成域名验证和签发TLS证书，并支持PKI中的其他扩展项验证，用于自动化签发其他类型的数字证书，如电子邮件证书、代码签名证书、文档签名证书和客户端证书等。本文件也适用于证书自动化的其他方面，实现在非自动化流程情况下的相关管理。

2 规范性引用文件

下列文件中的内容通过文中的规范性引用而构成本文件必不可少的条款。其中，注日期的引用文件，仅该日期对应的版本适用于本文件；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 25069-2022 信息安全技术 术语

GB/T 32918（所有部分）信息安全技术 SM2椭圆曲线公钥密码算法

GB/T 38636-2020 信息安全技术 传输层密码协议（TLCP）

GM/T 0125（所有部分）JSON Web 密码应用语法规范

GM/Z 4001 密码术语

RFC 3339 Date and Time on the Internet: Timestamps

RFC 6844 DNS Certification Authority Authorization (CAA) Resource Record

RFC 7515 JSON WebSignature(JWS)

RFC 7525 Recommendations for Secure Use of Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS)

RFC 7797 JSON WebSignature(JWS) Unencoded Payload Option

RFC 9773 ACME Renewal Information (ARI) Extension

3 术语和定义

GB/T 25069 和 GM/Z 4001 界定的以及下列术语和定义适用于本文件。

3.1

自动化证书管理环境 `automatic certificate management environment`; **ACME**

一种用于实现自动化验证域名/IP并自动执行证书签发的可扩展的协议框架。

3.2

ACME 客户端 `ACME client`

一个用于自动化向CA申请证书并自动部署已签发的TLS证书的客户端软件，一般安装在网站服务器或TLS网关上。

3.3

订单 order

客户端向服务端提交的、用于请求签发数字证书并跟踪其处理生命周期状态的资源对象。

3.4

授权 authorization

服务端赋予特定 ACME 帐户对某一主体标识（如域名、IP 地址、电子邮件地址等）拥有控制权的证明记录。

3.5

挑战 / 质询 challenge

服务端为验证客户端对特定主体标识的控制权，向客户端发出的一组需要按特定规则完成的验证任务或密码学操作要求。

3.6

主体标识 subject identifier

数字证书中所绑定的、用于标识证书所有者身份的属性字段，包括但不限于域名（DNS）、IP地址、电子邮件地址及设备唯一标识符等。

3.7

重放随机数 replay-nonce

由服务端生成的、具备足够熵值且不可预测的随机字符串，包含在 HTTP 响应头中，用于确保客户端后续请求的唯一性并抵御重放攻击。

3.8

证书签发请求 certificate signing request; CSR

客户端生成的一组包含申请者公钥、主体身份信息等内容的数据结构，需使用对应的私钥进行数字签名后提交给证书认证机构，用于申请数字证书。

3.9

账户密钥 account key

ACME客户端生成并持有的非对称密钥对，公钥用于向服务端注册账户并绑定账户身份，私钥用于对ACME请求进行数字签名。

3.10

令牌 token

由服务端在挑战对象中生成的随机值，客户端在完成挑战时以此令牌构造密钥授权字符串，供服务端验证客户端对主体标识的控制权。

3.11

密钥授权 key authorization

客户端基于令牌和账户密钥指纹构造的字符串，用于向服务端证明账户密钥持有者对主体标识的控制权。

3.12

证书链 certificate chain

由终端实体证书及其签发者的一个或多个中间CA证书按层级顺序排列构成的证书序列，用于验证终端实体证书的有效性。

3.13

JSON Web签名（JWS）

一种基于JSON数据结构，通过数字签名或消息鉴别码（MAC）保护内容的语法规则。JWS用于确保数据的完整性、真实性和不可否认性。

3.14

JSON Web密钥 (JWK)

一种使用JSON数据结构表示加密密钥的格式规范。JWK可表示公钥或私钥，包含密钥类型 (kty) 等参数，广泛用于JWS和JSON Web加密 (JWE) 等场景。

3.15

Base64url编码

一种将二进制数据编码为ASCII字符的编码方式，是标准Base64编码的URL安全变体。与标准Base64相比，将字符“+”替换为“-”、“/”替换为“_”，并省略填充字符“=”。该编码方式适用于在URL、文件名等环境中安全传输二进制数据。

4 符号和缩略语**4.1 符号**

$X||Y$: 比特串X和比特串Y的连接。

Base64url(*): URL安全的Base64编码

4.2 缩略语

下列缩略语适用于本文件。

ARI: 自动化证书管理续订信息 (ACME Renewal Info)

CA: 证书认证机构 (Certificate Authority)

CSR: 证书签发请求 (Certificate Signing Request)

JSON: JavaScript对象标记 (JavaScript Object Notation)

JWK: JSON Web密钥 (JSON Web Key)

JWS: JSON Web签名 (JSON Web Signature)

TLC: 传输层密码协议 (Transport Layer Cryptography Protocol)

URI: 统一资源标识符 (Uniform Resource Identifier)

URL: 统一资源定位符 (Uniform Resource Locator)

5 协议总体流程

自动化证书管理协议 (ACME) 采用 C/S 架构，客户端与服务端应通过安全传输协议进行通信，采用 JSON 格式封装交互报文。基于ACME的证书签发类似于传统的证书签发过程，其中用户创建一个帐户，请求一个证书，并证明对该证书绑定的域名的控制权，以便CA签发所请求的证书。除非特别说明，本文件中描述的客户端特指“ACME客户端”，服务端特指“ACME服务端”。

自动化证书管理框架如下图1所示，ACME客户端安装在网站服务器上，为网站自动化申请TLS证书、部署域名验证文件、获取证书和部署证书，并在证书到期前自动化续期证书。ACME服务端为ACME客户端提供身份鉴别、TLS证书申请和签发后下发证书服务，ACME服务端收到客户端提交的证书申请后，会把证书申请信息包括证书申请文件提交给证书认证系统 (CA)，CA在完成域名验证和相关所需的身份鉴别后会签发证书给ACME服务端，ACME服务端通知ACME客户端取回，ACME客户端取回后部署在网站上使用，从而完成整个证书管理流程。ACME客户端会定期检查已部署的TLS证书是否需要续期，若需要，则发起证书申请，重新走完整个流程。

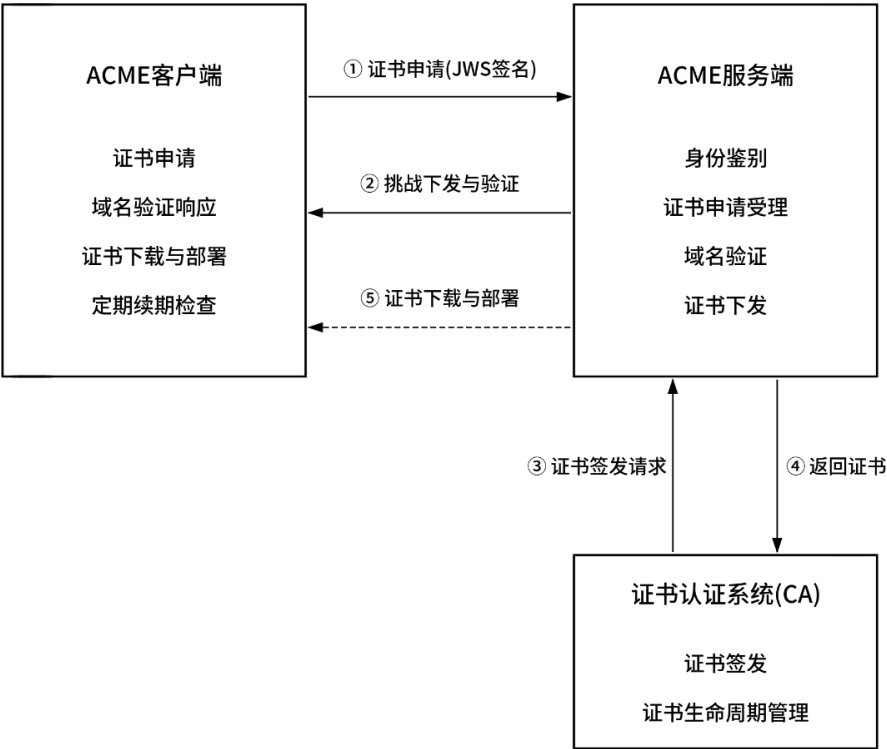


图 1 自动化证书管理框架

5.1 账户创建阶段

客户端应在服务端创建账户，具体流程见第7.4条。创建账户的流程见图2，客户端生成一个非对称密钥对并且请求一个新帐户，可选择提供联系信息、同意服务条款(ToS)和/或关联帐户使用另一个系统中的现有帐户。客户端使用生成的私钥对账户申请信息进行签名，以证明用户拥有私钥。

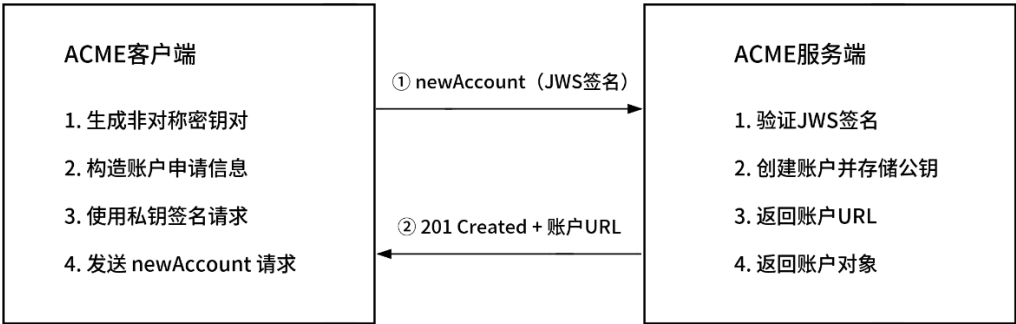


图 2 帐户创建流程

5.2 证书签发阶段

客户端在完成账户注册后，证书签发流程见图3，应按以下步骤申请证书：

- a) 提交证书订单，包括证书申请文件；
- b) 获取验证码并完成验证证书绑定的域名或其他主体标识的控制权；
- c) 等待证书签发，然后下载 CA 签发的证书。

客户端的订单描述了所需的证书字段、证书请求文件(CSR)以及一些 CSR 中未包含的附加字段，

服务端应告知客户端在签发证书之前应满足的要求列表。

例如：在大多数情况下，服务端会要求客户端证明它控制证书请求中的主体标识(如：域名)。可有很多不同的方式来验证不同类型的主体标识，服务端会选择一组可扩展的适合于正在请求验证的主体标识的验证方法。客户端通过一组响应来向服务端证明该客户端已经完成了哪些类型的验证准备，然后服务端验证该客户端是否已达到验证的要求。

一旦验证过程完成并且服务端确认了客户端满足其相关要求，服务端将签发请求的证书，并将其提供给客户端。

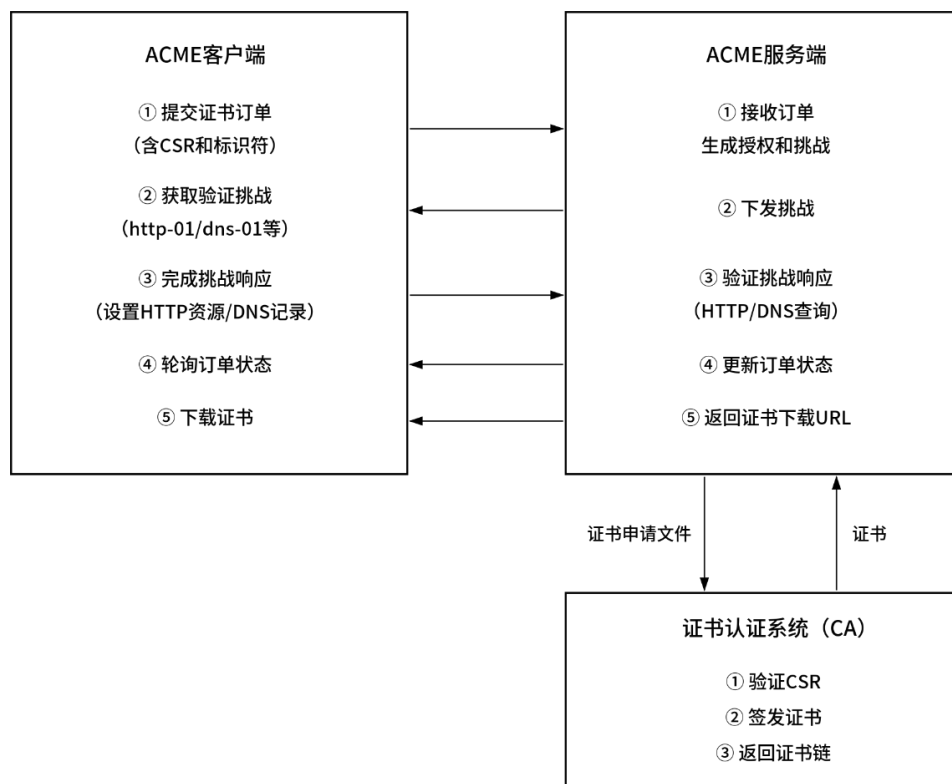


图 3 证书签发流程

5.3 证书撤销阶段

在签发证书之后，若需要撤销证书，客户端应发送一个带签名的撤销请求指示要撤销的证书，证书撤销流程见图 4。

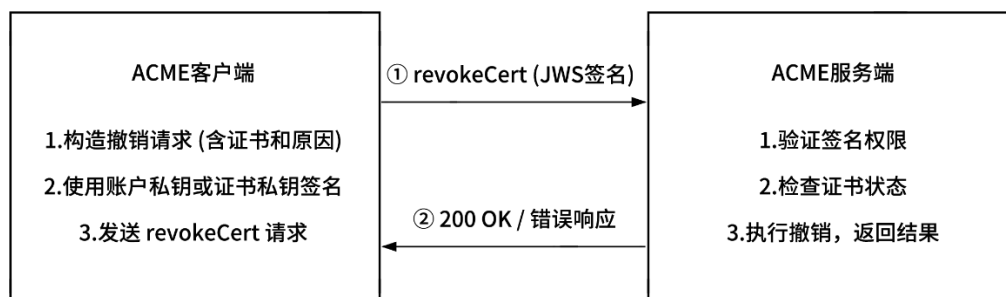


图 4 证书撤销流程

ACME 协议原则上具有足够的灵活性来处理不同类型的主体标识，本文件相关协议的范例主要以域名作为主体标识。将 ACME 用于其他主体标识的验证，将需要进一步规范以描述这些主体标识是如何在协议中编码的，以及服务端可能需要哪些类型的验证方法。

6 报文安全传输

6.1 概述

客户端和服务端之间的通信通过 HTTPS 完成，HTTPS 提供服务端身份验证和通信加密。与此同时，ACME 协议还利用 JSON Web 签名(JWS)为客户端发送到服务端的消息提供附加的安全属性。本文件对 JWS 进行了若干针对 ACME 场景的扩展，使其能为客户端请求提供身份认证、重放攻击防护以及 HTTPS 请求 URL 的完整性保护。

6.2 字符编码

客户端、ACME 服务端和域名验证服务器等实体通过 HTTP 发送的所有请求、响应报文以及密码杂凑算法的输入数据，应使用 UTF-8 字符集进行编码。此外，证书中的主体标识可能采用特定的编码格式。例如，包含非 ASCII 的域名(如中文域名：密.中国)应表示为 A 标签(如：xn--vct.xn--fiqs8s)而不是 U 标签。在执行任何此类编码转换之前，均应采用 UTF-8 编码。

6.3 HTTPS 请求

每个 ACME 功能都是由客户端向服务端发送一系列包含 JSON 消息的 HTTPS 请求来执行。JSON 消息应通过 HTTPS 来发送，本文件第 7 章描述了不同 ACME 功能使用的消息格式和发送消息的顺序。

在 ACME 使用的大多数 HTTPS 事务中，ACME 客户端是 HTTPS 客户端，ACME 服务端是 HTTPS 服务端。在验证挑战时，ACME 服务端和 ACME 客户端互为 HTTPS 服务端和客户端，协作完成“http-01”和“dns-01”等挑战/响应过程。

服务端应配置遵循 GB/T 38636-2020 的 TLCP 协议，或遵循 RFC 7525 的建议配置 TLS 协议。因为 ACME 协议本身包含防重放保护，支持 TLS1.3 的 ACME 服务端允许客户端发送早期数据(0-RTT)，并且 ACME 数据可不受限制地加载在 0-RTT 中。

客户端应在 HTTP 头部发送 User-Agent 字段，该字段除了包含 HTTP 客户端软件的名称和版本之外，还应包括 ACME 软件的名称和版本。

客户端应在 HTTP 头部发送 Accept-Language 头字段，启用错误消息的本地化语言。

作为可普遍访问的服务端，应使用跨源资源共享(CORS)，将 Access-Control-Allow-Origin 头字段设置为值“*”，以便基于浏览器的客户端能无限制访问。

ACME 使用的 JSON 消息中的二进制字段应按照 JSON Web 签名的配置，使用 URL 安全字符集进行 Base64url 编码，所有后缀包含“=”字符的编码结果应被认定为不正确编码而拒绝。

6.4 请求身份鉴别

所有具有非空主体的 ACME 请求，应按照 GM/T 0125 的规定将请求作为 JSON Web 签名(JWS)对象中的负载进行封装。除非特别说明，JWS 对象应包含携带 ACME 账户私钥的数字签名，服务端应在处理请求之前验证 JWS 签名。封装 JWS 的请求主体提供请求的身份鉴别。

作为 ACME 请求的主体发送的 JWS 对象应满足以下额外要求：

- a) JWS应采用扁平化JSON序列化；
- b) JWS不应包含多个签名；

- c) 不应使用RFC 7797标准的JWS Unencoded Payload Option;
- d) 不应使用RFC 7515标准的JWS Unprotected Header;
- e) JWS负载不应分离;
- f) JWS保护头部应包含以下字段:

- 1) “alg”: 该字段不应包含“none”或消息认证码(MAC)算法;
- 2) “nonce”: 在第6.7.1节中定义;
- 3) “url”: 请求目标 URL (定义见 6.6);
- 4) “jwk”或“kid”: JSON Web密钥或密钥标识符;

ACME 服务端应按照 GB/T 32918 以及 GM/T 0125 实现 SM2 签名算法。可兼容支持“ES256”签名算法, 使用“Ed25519”变体实现“EdDSA”签名算法。

“jwk”和“kid”字段是互斥的, 服务端应拒绝同时包含这两个字段的请求。

对于 newAccount 请求, 以及基于证书私钥来鉴别身份的 revokeCert 请求, 应包含“jwk”字段, 并且该字段应包含与 JWS 签名私钥相对应的公钥。

对于所有其他请求, 应使用 ACME 账号私钥来签名, 并且应包含“kid”字段, 该字段应包含 POST 到 newAccount 资源的 ACME 帐户 URL。

若客户端发送使用服务端不支持的算法签名的 JWS, 则服务端应返回状态代码为 400 (错误请求) 的错误, 错误类型为“urn:ietf:params:acme:error:badSignatureAlgorithm”。返回错误信息应包含一个“algorithms”字段, 其中包含支持的“alg”值数组。有关错误响应结构的更多详细信息, 请见第 6.9 节。

若服务端支持签名算法“alg”但不支持或选择拒绝公钥“jwk”, 则服务端应返回状态码为 400 (错误请求) 的错误, 错误类型为: “urn:ietf:params:acme:error:badPublicKey”。返回错误的详细信息应描述拒绝公钥的原因; 一些示例原因是:

- a) “alg”是“RS256”, 但模数“n”太小(例如: 512 位);
- b) “alg”是“ES256”, 但“jwk”不包含有效的 P-256 公钥;
- c) “alg”是“EdDSA”, “crv”是“Ed448”, 但服务端只支持“EdDSA”和“Ed25519”;
- d) 签名私钥被识别为已泄露。

由于 ACME 客户端请求在扁平化 JSON 序列中携带 JWS 对象, 因此它们应将 Content-Type 头字段设置为“application/jose+json”。若请求不满足此要求, 则服务端应返回状态码为 415 (不支持的媒体类型) 的响应。

JWS 请求的完整验证流程:

ACME 服务端收到客户端 POST 请求后, 应按以下步骤验证 JWS:

- a) 解析 JWS 对象, 确认其为扁平化 JSON 序列化格式, 且仅包含一个签名;
- b) 验证受保护头部是否包含“alg”、“nonce”、“url”三个必选字段;
- c) 验证“jwk”与“kid”字段互斥, 且至少存在其一;
 - 对于 newAccount 请求和 revokeCert 请求 (使用证书私钥签名), 应存在“jwk”字段;
 - 对于其他请求, 应存在“kid”字段;
- d) 验证“nonce”字段值是否在服务端已发布且未使用的随机数列表中;
- e. 验证“url”字段值与 HTTP 请求的 URL 是否完全一致【6.6 节】;
- f. 根据“alg”字段指定的算法验证 JWS 签名:
 - 若“alg”为“SM2”, 按 GM/T 0125 及本文件 6.4.1 节的规定执行验证;
 - 若“alg”为“ES256”等国际算法, 按 RFC 7515 的规定执行验证;
- g. 验证通过后, 根据“kid”字段 (或“jwk”字段中的公钥) 定位对应的 ACME 账户, 继续处理请求。

6.4.1 基于国密 SM2 算法的 JWS 封装与计算规范

为满足商用密码应用合规要求，当 ACME 客户端与服务端采用国密算法进行身份鉴别时，JWS 受保护头中的 alg 字段应设置为字符串 SM2。其签名计算与封装应严格遵循 GM/T 0125 的相关规范，并作如下具体约定：

- a) 底层算法调用：数字签名的生成与验证应符合 GB/T 32918.2 的规定；在此过程中使用的密码杂凑算法应符合 GB/T 32905（SM3 算法）的规定；
- b) 用户标识（ID）处理：在计算 SM2 签名的 Z_A 值（杂凑值）时，除通信双方存在预先协商的特殊约定外，应采用标准默认的用户标识（即 ASCII 字符串 1234567812345678）；
- c) 公钥参数封装：当使用 jwk 字段封装 SM2 公钥时，相关字段的取值应符合下列规定：
 - 1) “kty”字段：应设置为“EC”；
 - 2) “crv”字段：应设置为“SM2”，以标识 SM2 椭圆曲线；
 - 3) “x”字段：应为 SM2 公钥点的 X 坐标，以大端字节序表示为 32 字节，并使用 Base64url 编码；
 - 4) “y”字段：应为 SM2 公钥点的 Y 坐标，以大端字节序表示为 32 字节，并使用 Base64url 编码。
- d) 签名值编码格式：由 SM2 算法生成的签名值（包含 r 和 s 两个分量），应将 r 和 s 各转换为固定长度 32 字节（不足时在高位补零）的字节串，按 r||s 顺序拼接为 64 字节，随后采用 Base64url 格式进行编码。编码结果作为 JWS 的 signature 字段值，且不应包含任何回车、换行或填充字符（=）。

SM2 算法在 ACME 协议中应用于以下关键环节：

- a) 账户创建与身份鉴别：客户端生成 SM2 密钥对作为账户密钥，使用 SM2 私钥对 newAccount 请求进行 JWS 签名；服务端使用对应的 SM2 公钥验证签名；
- b) 证书签发请求：客户端在 CSR 中使用 SM2 算法生成签名证书和加密证书的密钥对，分别提交 csrSign 和 csrEncrypt 字段；
- c) 挑战验证：在 http-01 和 dns-01 挑战中，密钥授权（key authorization）的构造涉及对账户公钥的指纹计算。使用 SM2 算法时，指纹计算采用 SM3 密码杂凑算法；
- d) 证书撤销：客户端可使用 SM2 账户私钥或证书中 SM2 私钥对撤销请求进行签名。

6.5 GET 和 POST-as-GET 请求

由于 ACME 协议通过带有数字签名的 JWS 请求主体进行身份鉴别，因此无主体的请求（尤其是 GET 请求）将无法通过身份鉴别。除本节规定的特殊情况外，若服务端收到 GET 请求，应返回状态码 405（方法不允许）的错误响应，且错误类型为‘malformed’（格式错误）。

若客户端希望从服务端获取资源，客户端应发送带有 JWS 主体的 POST 请求（否则将使用 GET 完成）。如上所述，其中 JWS 的有效负载是一个零长度的八位字节字符串。换句话说，JWS 对象的“payload”字段应存在并设置为空字符串（“”）。

这种形式的请求称为“POST-as-GET”请求。在收到请求的有效载荷长度为零字节时（换言之不是 JSON 报文），服务端应验证发送者，并验证访问控制规则。否则，服务端应将此请求视为与对同一资源的 GET 请求具有相同的语义。

服务端应允许对服务目录（directory）和 newNonce 资源的 GET 请求（见第 7.2 节），以及对这些资源的 POST-as-GET 请求。这使得客户端能够引导进入 ACME 身份验证系统。

6.6 请求 URL 完整性

在部署中，为 HTTPS 终止 TLS 连接的过程与访问 HTTPS 服务端的过程不同，中间有一个“请求路由”层，这在部署中很常见。例如：服务端可能有一个 CDN 终止来自客户端的 TLS 连接，以便它可检查客户端请求以提供拒绝服务 (DoS) 保护。

这些中介还可更改请求中未在 HTTPS 请求中签名的值，例如：请求 URL 和头字段。ACME 使用 JWS 来提供完整性机制，以防止中介将请求 URL 更改为另一个 ACME URL。

如第 6.4 节所述，所有 ACME 请求对象在其受保护头中都带有一个“url”头参数。此头参数对客户端将请求定向到的 URL 进行编码。在 HTTP 请求中接收到此类对象时，服务端应将“url”头参数与请求 URL 进行比较。若两者不匹配，则服务端应拒绝未经授权的请求。

除目录资源外，所有 ACME 资源都使用服务端提供给客户端的 URL 进行寻址。在发送到这些资源的 POST 请求中，客户端应将“url”头参数设置为服务端提供的确切字符串（而不是对 URL 执行任何重新编码）。服务端应执行相应的字符串相等性检查，使用提供给客户端的 URL 字符串配置每个资源，并让资源检查请求在其“url”头参数中是否具有相同的字符串。若字符串相等性检查失败，服务端应拒绝未经授权的请求。

“url”头参数指定此 JWS 对象指向的 URL。在 JWS 的受保护头中应携带“url”头参数。“url”头参数的值应是表示目标 URL 的字符串。

6.7 重放保护

为保护 ACME 资源免受任何可能的重放攻击，ACME 的 POST 请求具有强制性的防重放机制。此机制基于服务端维护它已发布的随机数列表，并要求来自客户端的任何签名请求携带此类随机数。

ACME 服务端使用 HTTP 协议的 Replay-Nonce 头字段向客户端提供随机数，如第 6.7.1 节所述。服务端应在对 POST 请求的每个成功响应中包含一个 Replay-Nonce 头字段，并且也应在错误响应中提供它。

ACME 客户端发送的每个 JWS 都应在其受保护头中包含“nonce”头参数，其内容如第 6.7.2 节中所定义。作为 JWS 验证的一部分，ACME 服务端应验证“nonce”头的值是服务端先前在 Replay-Nonce 头字段中提供的值。一旦 Nonce 值出现在 ACME 请求中，服务端应认为它无效，就像它从未发出过的值一样。

当服务端因为 nonce 值不可接受（或未提供）而拒绝请求时，服务端应提供 HTTP 状态代码 400（错误请求），并指示 ACME 错误类型“urn:ietf:params:acme:error:badNonce”。错误类型为“badNonce”的错误响应，应包含一个带有新随机数的 Replay-Nonce 头字段，服务端将重试原始查询时接受该随机数（并且可能在其他请求中，根据服务端的随机数范围策略）。收到这样的响应后，客户端应使用新的随机数重试请求。

用于生成和跟踪随机数的精确方法取决于服务端。例如：服务端可为每个响应生成一个随机的 128 位值，保留已发布的随机数列表，并在使用随机数时从该列表中剔除随机数。

除了上面关于在“badNonce”响应中发出的随机数的约束，ACME 不限制服务端如何限定随机数。客户端可假设 Nonces 具有广泛的范围，例如：通过将单个 Nonces 池用于所有请求。但是，当重试响应“badNonce”错误时，客户端应使用错误响应中提供的随机数。服务端宜合理设置随机数的有效范围，以减少客户端因“badNonce”错误而重试的频率。

6.7.1 重放随机数

Replay-Nonce 头字段包含一个服务端生成的值，服务端可使用该值来检测未来客户端请求中未经授权的重放。服务端应生成 Replay-Nonce 头字段中提供的值，使得它们对于每条消息都是唯一的，并且概率很高，并且除了服务端之外的任何人都无法预测。例如：随机生成 Replay-Nonces 是可接受的。

Replay-Nonce头字段的值应是Base64url编码的字节串。客户端应忽略无效的Replay-Nonce值。
Replay-Nonce头字段遵循 ABNF如下：

Base64URL= ALPHA / DIGIT / “-” / “_”；

Replay-Nonce= 1*Base64url；

Replay-Nonce 头字段不应包含在HTTP请求消息中。

6.7.2 “随机数” (Nonce) JWS 头参数

“nonce”头参数提供了一个唯一值，使 JWS的验证程序能够识别何时发生重放。“nonce”头参数应携带在JWS的受保护头中。

“nonce”头参数的值应是字节串，并且采用Base64url编码格式。若“nonce”头参数的值根据此编码无效，则验证者应拒绝JWS，并且拒绝原因为“malformed”（格式错误）。

6.8 速率限制

ACME服务端可对资源的创建进行速率限制，以确保公平使用并防止滥用。一旦超过速率限制，服务端响应类型为“urn:ietf:params:acme:error:rateLimited”的错误。此外，服务端应发送一个有效的Retry-After头字段，指示当前请求何时可再次成功。若存在多个速率限制，则所有速率限制都允许使用完全相同的参数再次访问当前请求。

除了错误响应的人类可读的“详细信息”字段之外，服务端可使用“help”在Link头字段中发送一个或多个链接，指向有关被命中的特定速率限制的文档链接。

6.9 错误

ACME错误可在HTTP层和第6.7节中定义的验证对象中报告。ACME服务端可返回带有HTTP错误响应代码（4XX或5XX）的响应。例如：若客户端使用本文件中不允许的方法提交请求，则服务端可能会返回状态码405（方法不允许）。

当服务端以错误状态响应时，它应提供额外的信息。为促进对错误的自动响应，本文件定义了以下用于“类型”字段的标准标记（在ACME URN命名空间“urn:ietf:params:acme:error:”内），见表1：

表 1 错误信息

Type类型	描述
accountDoesNotExist	该请求指定了一个不存在的帐户
alreadyRevoked	该请求指定要撤销的证书已被吊销
badCSR	证书请求文件有问题（例如：由于短密钥）
badNonce	客户端发送了一个有问题的反重放随机数
badPublicKey	JWS使用了服务端不支持的公钥签名
badRevocationReason	服务端不接受客户端提供的撤销原因
badSignatureAlgorithm	JWS使用了服务端不支持的算法签名
caa	CAA记录禁止该CA签发证书
compound	“子问题”数组中指示了特定的错误情况
connection	服务端无法连接到请求验证的对象
dns	主体标识验证期间DNS查询出现问题
externalAccountRequired	该请求应包含“externalAccountBinding”字段的值
incorrectResponse	收到的响应验证码不正确
invalidContact	帐户的联系URL是无效的
malformed	请求消息格式错误
orderNotReady	该请求试图完成尚未准备好完成的订单
rateLimited	请求超出速率限制
rejectedIdentifier	服务端不会为此主体标识(域名)签发证书
serverInternal	服务端遇到内部错误
tls	服务端在验证域名期间收到TLS错误
unauthorized	客户端缺乏足够的授权
unsupportedContact	帐户的联系URL使用了不支持的协议
unsupportedIdentifier	主体标识类型不支持
userActionRequired	访问“实例”URL并执行所要求的操作
alreadyReplaced	请求指定了标记为已替换的前置任务证书

此列表并不详尽。服务端可能会返回错误，其“类型”字段被设置为一个不同于上面定义的 URI。对于未在适当的IANA注册表中列出的错误，服务端不应使用ACME URN命名空间（请参阅第A.6节）。客户端应显示所有错误的“详细信息”字段。

本文件后续条款均直接使用表1中的标记来指代错误类型，不再赘述完整的 URN。例如：‘badCSR 类型的错误’，即特指‘类型（type）’字段值为‘urn:ietf:params:acme:error:badCSR’的错误。

有时CA可能需要返回多个错误以响应请求。此外，CA可能需要将错误归因于特定的主体标识。例如：newOrder请求可能包含多个主体标识，CA无法为其签发证书。在这种情况下，ACME错误信息可能包含“子问题”字段，包含错误信息的JSON数组，每个错误信息可能包含一个“identifier”字段。若存在，“identifier”字段应包含一个ACME主体标识（第A.7节）。

“identifier”字段不应出现在ACME错误信息的顶层。它只能出现在子问题中。子问题不需要都具有相同的类型，也不需要匹配顶级类型。

ACME客户端可选择使用子问题的“identifier”字段作为提示，若省略该主体标识，操作将会成功。例如：若一个订单包含十个DNS域名，并且newOrder请求返回一个包含两个子问题（引用其中两个域名）的错误信息，则ACME客户端可选择提交另一个仅包含错误信息中未列出的其他八个域名的订单。

HTTP/1.1 403 Forbidden

Content-Type: application/problem+json

Link: <https://example.com/acme/directory>;rel="index"

```
{
  "type": "urn:ietf:params:acme:error:malformed",
  "detail": "Some of the identifiers requested were rejected",
  "subproblems": [
    {
      "type": "urn:ietf:params:acme:error:malformed",
      "detail": "Invalid underscore in DNS name \"_example.org\"",
      "identifier": {
        "type": "dns",
        "value": "_example.org"
      }
    },
    {
      "type": "urn:ietf:params:acme:error:rejectedIdentifier",
      "detail": "This CA will not issue for \"example.net\"",
      "identifier": {
        "type": "dns",
        "value": "example.net"
      }
    }
  ]
}
```

7 证书管理协议

7.1 概述

自动化证书管理功能应包含以下交互过程：

- a) 帐户创建；
- b) 申请证书；
- c) 主体标识(域名/IP)授权；
- d) 证书签发；
- e) 证书撤销。

7.2 资源

ACME的结构是基于HTTP的应用程序，具有以下类型的资源：

- a) 帐户资源，表示有关帐户的信息（第 7.2.2 节、第 7.4 节）；

- b) 订单资源，代表账户签发证书的请求（第 7.2.3 节）；
- c) 授权资源，代表账户对主体标识的授权（第 7.2.4 节）；
- d) 验证资源，代表验证以证明对主体标识的控制（第 7.6.2 节、第 0 节）；
- e) 证书资源，代表签发的证书（第 7.5 节）；
- f) “directory” 资源（第 7.2.1 节）；
- g) “newNonce” 资源（第 7.3 节）；
- h) “newAccount” 资源（第 7.4 节）；
- i) “newAuthz” 资源（第 7.5.2 节）
- j) “newOrder” 资源（第 7.5 节）；
- k) “revokeCert” 资源（第 7.7 节）；
- l) “keyChange” 资源（第 7.4.5 节）；
- m) “renewalInfo” 资源（第 7.8 节）。

服务端应提供“directory”和“newNonce”资源。

ACME为不同的管理功能使用不同的URL。每个功能都列在一个目录及其相应的URL中，因此客户端只应配置目录URL。这些URL由几个不同的链接关系连接[参考RFC 8288]。

“up”链接关系与验证资源一起使用，以指示验证所属的授权资源。对于某些媒体类型，它还用于证书资源中，以指示客户端可从中获取可用于验证原始资源中的证书的CA证书链的资源。

“index”链接关系存在于除目录之外的所有资源上并指示目录的URL。

图5说明了ACME服务端上资源之间的关系。大多数情况下，这些关系由资源的JSON表示中作为字符串提供的URL表示。引号中带有标签的行表示HTTP链接关系。

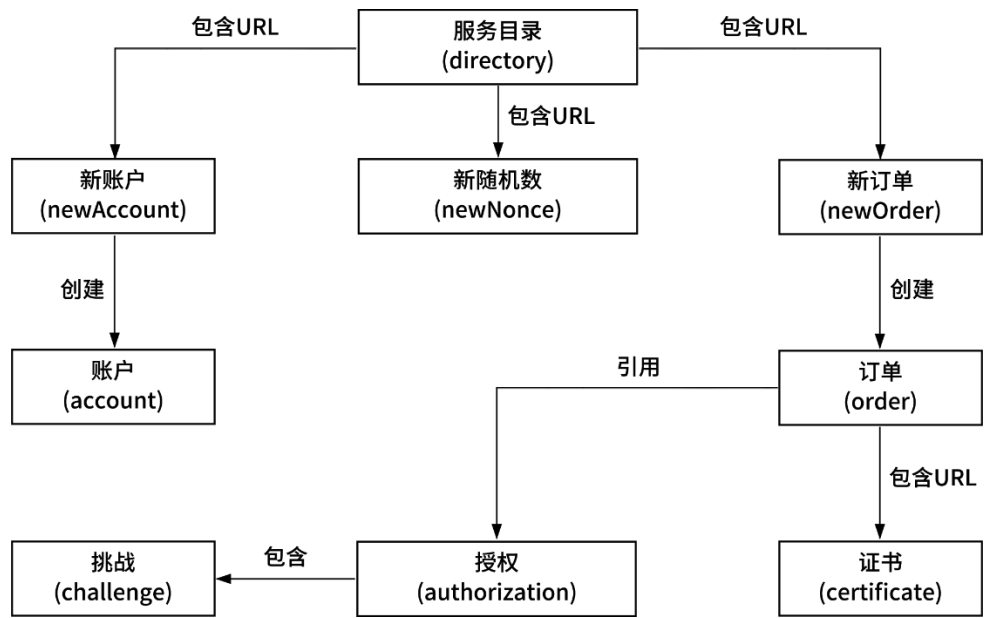


图 5 ACME 资源和关系

表2说明了在服务端上建立新帐户、证明对主体标识的控制、签发证书以及在签发后的某个时间获取更新证书所需的典型请求序列。“->”是指向已创建资源的Location头字段的助记符。

表 2 请求序列

Action	动作	要求	响应
--------	----	----	----

GET directory	获取目录	GET directory	200
GET nonce	获取随机数	HEAD newNonce	200
Create account	创建账户	POST newAccount	201 -> 帐户
Submit order	提交订单	POST newOrder	201 -> 订单
Fetch challenges	请求验证	POST-as-GET order's authorizationURLs	200
Respond to challenges	响应验证	POST authorization challengeURLs	200
Poll for status	轮询状态	POST-as-GET order	200
Finalize order	完成订单	POST order's finalizeURL	200
Poll for status	轮询状态	POST-as-GET order	200
Download certificate	下载证书	POST-as-GET order's certificateURL	200

本节的其余部分详细介绍了这些资源的结构以及ACME协议如何使用它们。

7.2.1 服务目录 (directory)

为帮助客户端为每个ACME操作配置正确的URL，ACME服务端提供了一个服务目录对象。这应是配置客户端所需的唯一URL。为一个JSON对象，其字段名称取自资源注册表(第A.7.6节)，其值是相应的URL，见表3。

表 3 目录对象

字段	URL in Value	URL值
newNonce	New Nonce	新随机数
newAccount	New account	新账户
newOrder	New order	新订单
newAuthz	New authorization	新授权
revokeCert	Revoke certificate	撤销证书
keyChange	Key change	密钥更改
renewalInfo	Renewal info	续期信息

服务目录的URL没有限制，除了它应与其他ACME服务端资源的URL不同，并且它不应与其他服务冲突。例如：

- a) 同时作为 ACME 和网站服务器的主机可能希望保留 HTML “首页” 的根路径 “/”，并将 ACME 目录放在路径 “/acme” 下；
- b) 仅作为 ACME 服务端的主机可将目录放在路径 “/” 下。

若ACME服务端没有实现预授权，服务端应省略目录的“newAuthz”字段。

该对象可另外包含一个“meta”字段。若存在，该字段应是一个JSON对象；对象中的每个字段都是与ACME服务端提供的服务相关的元数据项(metadata)。

以下元数据项（第A.7.7节）已定义，所有这些都是可选的：

- a) termsOfService（可选，字符串）：标识当前服务条款的URL；
- b) website（可选，字符串）：一个HTTP或HTTPS URL，用于提供有关ACME服务端的更多信息的网站；
- c) caaIdentities（可选，字符串数组）：用于ACME服务端识别自身的主机名，用于RFC 6844规定的CAA记录验证。每个字符串应代表相同的ASCII代码点序列，服务端希望在CAA issue或issuwild属性标签中将其视为“签发者域名”。这允许客户端以确定在配置CAA记录时使用正确签发者域名。

d) externalAccountRequired (可选, 布尔值): 若此字段存在并设置为“true”, 则CA要求所有newAccount请求包含一个“externalAccountBinding”字段, 将新账户与外部账户相关联。

客户端通过向目录 URL 发送 GET 请求来访问目录。在从目录向任何 URL 发出请求之前, 客户端应根据访问该目录对象时收到的 Cache-Control 头评估该目录对象是否仍然是新鲜的。若没有收到 Cache-Control 头, 则客户端应像收到“Cache-Control: no-cache”一样处理。若目录对象不再新鲜, 客户端应再次访问目录 (通过向目录 URL 发送另一个 GET 请求), 然后使用更新的目录对象。

HTTP/1.1 200 OK Content-Type: application/json

```
{
  "newNonce": "https://example.com/acme/new-nonce",
  "newAccount": "https://example.com/acme/new-account",
  "newOrder": "https://example.com/acme/new-order",
  "newAuthz": "https://example.com/acme/new-authz",
  "revokeCert": "https://example.com/acme/revoke-cert",
  "keyChange": "https://example.com/acme/key-change",
  "renewalInfo": "https://example.com/acme/renewal-info",
  "meta": {
    "termsOfService": "https://example.com/acme/terms/2017-5-30",
    "website": "https://www.example.com/",
    "caaIdentities": ["example.com"],
    "externalAccountRequired": false
  }
}
```

7.2.2 账户对象

ACME帐户资源表示与帐户关联的一组元数据。帐户资源具有以下结构:

a) status (必需, 字符串): 此帐户的状态。可能的值为“valid(有效)”、“deactivated(停用)”和“revoked(撤销)”。值“deactivated”用于指示客户端发起的停用, 而“revoked”用于指示服务端发起的停用。请参阅第7.2.6节;

b) contact (可选, 字符串数组): 一组URL, 服务端可使用这些URL联系客户端以解决与此帐户相关的问题。例如: 服务端可能希望通知客户端关于服务端发起的撤销或证书过期。有关支持的URL方案的信息, 请参阅第7.4节;

c) termsOfServiceAgreed (可选, 布尔值): 在newAccount请求中包含此字段, 值为true, 表示客户端同意服务条款。客户端无法更新此字段;

d) externalAccountBinding (可选, 对象): 在newAccount请求中包含此字段表示一个已经存在的非ACME帐户的持有人批准将该帐户绑定到此ACME帐户。该字段不能由客户端更新 (见第7.4.4);

e) orders (必需, 字符串): 一个URL, 可通过 POST-as-GET请求从中获取此帐户提交的订单列表, 如第7.2.2节所述。

```
{
  "status": "valid",
  "contact": [
```

```

        "mailto:cert-admin@example.org",
        "mailto:admin@example.org"
    ],
    "termsOfServiceAgreed": true,
    "orders": "https://example.com/acme/orders/rzGoeA"
}

```

每个帐户对象都包含一个“订单”URL，可通过 POST-as-GET请求从中获取帐户创建的订单列表。请求的结果应是一个JSON对象，其“订单”字段是一个URL数组，每个URL标识一个属于该帐户的订单。服务端应包括待处理(pending order，指等待签发证书的订单)、就绪订单(ready order)、处理中订单(processing order)，并且不应包括在URL数组中无效的订单。服务端可返回一个不完整的列表，连同带有“下一个”链接关系的 Link 头字段，指示可获取更多条目的位置。

```

HTTP/1.1 200 OK
Content-Type: application/json
Link: <https://example.com/acme/directory>;rel="index"
Link: <https://example.com/acme/orders/rzGoeA?cursor=2>;rel="next"
{
  "orders": [
    "https://example.com/acme/order/T0locE8rfgo",
    "https://example.com/acme/order/4E16bbL5iSw",
    /* 为简洁起见，未显示更多 URL */
    "https://example.com/acme/order/neBHYLfw0mg"
  ]
}

```

7.2.3 订单对象

ACME订单对象表示客户对证书的请求，并用于跟踪该订单直至签发的进度。因此，该对象包含有关所请求证书的信息、服务端要求客户端完成的授权以及由此订单产生的任何证书。

ACME订单可分为快速订单和预验订单，快速订单下单时直接提交证书申请文件和选择域名验证方式，预验订单则是先完成域名验证后下单。

- a) status(必需, 字符串): 此订单的状态。可能的值是“pending(待处理)”、“ready(就绪)”、“processing(处理中)”、“valid(有效)”和“invalid(无效)”。请参阅第7.2.6节;
- b) expires(可选, 字符串): 服务端认为此订单无效的时间。对于状态字段中具有“pending(待处理)”或“valid(有效)”的对象，此字段是必需的;
- c) identifiers(必需, 对象数组): 订单所属的主体标识对象数组;
- 1) type(必需, 字符串): 主体标识的类型。本文件定义了“dns”主体标识类型。有关任何其他信息，请参阅第A.7.8节中定义的注册表;
- 2) value(必需, 字符串): 主体标识本身(一般为证书绑定的域名);
- 3) challengeType(可选, 字符串): 挑战类型，仅适用于快速订单;
- d) notBefore(可选, 字符串): 证书中notBefore字段的请求值;
- e) notAfter(可选, 字符串): 证书中notAfter字段的请求值;

- f) `csr` (可选, 字符串): CSR编码所请求国际算法证书的参数, CSR以DER格式的Base64url编码版本发送。仅适用于快速订单;
- g) `csrSign` (可选, 字符串): CSR编码所请求商密签名证书的参数, CSR以DER格式的Base64url编码版本发送。仅适用于快速订单;
- h) `csrEncrypt` (可选, 字符串): CSR编码所请求商密加密证书的参数, CSR以DER格式的Base64url编码版本发送。仅适用于快速订单;
- i) `csrSM2` (可选, 字符串): CSR编码所请求商密单密码用途证书的参数, CSR以DER格式的Base64url编码版本发送。仅适用于快速订单;
- j) `csrSignRSA` (可选, 字符串): CSR编码所请求国际算法签名证书的参数, CSR以DER格式的Base64url编码版本发送。仅适用于快速订单;
- k) `csrEncryptRSA` (可选, 字符串): CSR编码所请求国际算法加密证书的参数, CSR以DER格式的Base64url编码版本发送。仅适用于快速订单;
- l) `error` (可选, 对象): 处理订单时发生的错误(如果有)。该字段被构造为错误信息;
- m) `authorizations` (必需, 字符串): 对于待处理, 客户端应在签发所请求的证书之前完成的授权(见第7.6节), 包括客户端过去为指定的主体标识完成的未过期授权订单。所需的授权由服务端策略决定; 订单主体标识和所需授权之间可能不是1:1的关系。对于最终订单(处于“有效”或“无效”状态), 已完成授权。每个条目都是一个URL, 可使用POST-as-GET请求从中获取授权;
- n) `finalize` (可选, 字符串): 一个URL, 一旦满足所有订单的授权就可完成订单, 就应将CSR提交到该URL, 成功完成的结果将是订单URL的填充。快速订单无需调用;
- o) `certificate` (可选, 字符串): 为响应此订单而签发的证书的URL。此项保留供国际算法TLS证书用;
- p) `certificateSign` (可选, 字符串): 为响应此订单而签发的商密签名证书的URL;
- q) `certificateEncrypt` (可选, 字符串): 为响应此订单而签发的商密加密证书的URL;
- r) `certificateSM2` (可选, 字符串): 为响应此订单而签发的单密码用途证书的URL;
- s) `certificateSignRSA` (可选, 字符串): 为响应此订单而签发的国际算法签名证书的URL;
- t) `certificateEncryptRSA` (可选, 字符串): 为响应此订单而签发的国际算法加密证书的URL。

CSR是对客户端要签发的证书内容的签名请求进行编码。“`csr`”为国际算法证书的CSR, 可没有。“`csrSign`”为签名证书的CSR, “`csrEncrypt`”为加密证书的CSR, “`csrSignRSA`”为国际算法的签名证书的CSR, “`csrEncryptRSA`”为国际算法加密证书的CSR, TLS证书的这两个密钥用法的CSR应同时提交, 邮件证书可不同时提交。“`csrSM2`”为单用途商密证书的CSR。若只签发国际算法证书, 则只需提交“`csr`”; 若只签发商用密码算法证书, 则只需提交“`csrSign`”和“`csrEncrypt`”; 若同时签发国际算法证书和商用密码算法证书, 则应同时提交“`csr`”、“`csrSign`”和“`csrEncrypt`”或同时提交“`csrSignRSA`”、“`csrEncryptRSA`”、“`csrSign`”和“`csrEncrypt`”。若只提交“`csrSM2`”, 则只签发商用密码算法单用途证书。对于TLS证书, 这4种证书申请文件不能全部没有, 应至少有其中一个“`csr`”、“`csrSign`”和“`csrEncrypt`”、“`csrSM2`”。

```
{
  "status": "valid",
  "expires": "2016-01-20T14:09:07.99Z",
```

```

    "identifiers":[
      { "type": "dns", "value": "www.example.org", "challengeType": "http-01" },
      { "type": "dns", "value": "example.org", "challengeType": "dns-01" }
    ],
    "notBefore": "2016-01-01T00:00:00Z",
    "notAfter": "2016-01-08T00:00:00Z",
    "csr": "MIIBPTCBxAIBADBFBMQ...FS6aKdZeGsyoCo4H9P",
    "csrSign": "MIIBPTCBxAIBWghADBFBM...FS6aysCo4H9P",
    "csrEncrypt": "MIIBPTCBA88Dsers...FSWghsyCo4g788fH9P",
    "authorizations":[
      "https://example.com/acme/authz/PAniVnsZcis",
      "https://example.com/acme/authz/r4HqLzrSrpI"
    ],
    "finalize": "https://example.com/acme/order/T0locE8rfgo/finalize",
    "certificate": https://example.com/acme/cert/mAt3xBGaobw
    "certificateSign": "https://example.com/acme/cert/sign/mAt3xBGaobw"
    "certificateEncrypt": https://example.com/acme/cert/encrypt/mAt3xBGaobw
  }

```

newOrder请求中任何类型为“dns”的主体标识可有一个通配域名作为它的值。通配域名由一个星号字符后跟一个句点组成, 字符（“*.”）后跟一个域名, 定义为证书使用者可选名称扩展中使用。服务端为通配域名返回的授权不应在授权主体标识值中包含星号和句号（“*.”）前缀。返回的授权应包含可选的“wildcard”字段, 值为“true”。

“authorizations”和“identifiers”数组的元素在设置后是不可变的。创建后, 服务端不应更改任一数组的内容。若客户端观察到任一数组的内容发生变化, 则它应认为该顺序无效。

订单的“authorizations”数组应反映CA在决定签发证书时考虑的所有授权, 即使某些授权已在较早的订单或预授权交易中完成。例如: 若CA允许基于单个授权交易完成多个订单, 则它应在所有订单中反映该授权。

订单对象的‘authorizations’（授权）数组中包含授权 URL, 并不强制要求客户端采取进一步操作。该引用的授权可能已处于有效状态, 常见原因包括:

- a) 作为先前订单的一部分, 客户完成了授权;
- b) 服务端根据外部账号给客户端授权。

客户端应检查订单的“status”字段以确定他们是否需要采取任何行动。

7.2.4 授权对象

一个ACME授权对象表示服务端对一个帐户的授权, 以代表一个主体标识(如: 域名)。除了主体标识之外, 授权还包括几个元数据字段, 例如授权的状态(例如: “pending”, “valid”, “revoked”)以及使用哪些验证方式来验证对主体标识的拥有。

一个ACME授权资源的结构如下:

- a) identifier(必需, 对象): 账户被授权使用的主体标识;
 - 1) type (必需, 字符串): 主体标识的类型(见下文和第 A.7.8 节);
 - 2) value (必填, 字符串): 主体标识本身。

b) status (必需, 字符串): 此授权的状态。可能的值是“pending”, “valid”, “invalid”, “deactivated”, “expired”, and “revoked”(“待处理”、“有效”、“无效”、“已停用”、“已过期”和“已撤销”。请参阅第7.2.6节;

c) expires (可选, 字符串): 时间, 服务端认为此授权无效的时间。对于“status”字段是“valid”的对象, 此字段是必需的;

d) challenges (必需, 对象数组): 对于待处理授权, 客户端可完成的验证以证明拥有该主体标识; 对于有效授权, 验证已完成。对于无效授权, 已尝试验证但失败。每个数组条目都是一个对象, 其中包含完成验证所需的参数。客户端应尝试完成其中一项验证, 而服务端应考虑足以使授权有效的任何一项验证方法; 对于快速订单, 在创建订单时直接提交了“challengeType”, “challenges”只有一个挑战对象。

e) wildcard (可选, 布尔值): 对于包含通配符域名值的 DNS 标识符的newOrder请求而创建的授权, 此字段应存在且应为true。对于其他授权, 该字段应不存在。

通配域名在第 7.2.3 节中描述。

本规范定义的主体标识类型是一个合法注册的域名(类型: “dns”)。域名应以其在证书中出现的形式进行编码。服务端应验证以ASCII兼容编码前缀“xn-”开头的任何主体标识值是否已正确编码。授权对象中不应包含通配域名(以“*”作为第一个标签)。若授权对象传递了对包含通配域名的newOrder DNS标识符的基本域的授权, 则可选授权“wildcard”字段的值应存在且应为“true”。

第7.6节描述了域名验证的一组验证方法。

```
{
  "status": "valid",
  "expires": "2015-03-01T14:09:07.99Z",
  "identifier": {
    "type": "dns",
    "value": "www.example.org"
  },
  "challenges": [
    {
      "url": "https://example.com/acme/chall/prV_B7yEyA4",
      "type": "http-01",
      "status": "valid",
      "token": "DGyRejmCefe7v4NfDGDKfA",
      "validated": "2014-12-01T12:05:58.16Z"
    }
  ],
  "wildcard": false
}
```

7.2.5 挑战对象

ACME挑战对象表示服务端以特定方式验证客户端是否拥有主体标识。与上面列出的其他对象不同, 挑战对象没有单一的标准结构。挑战对象的内容取决于所使用的验证方法。第7.6节描述了挑战对象的一般结构和一组初始验证方法。

7.2.6 状态变化

每个ACME对象类型在其生命周期中都经过一个简单的状态机。对象的“status”字段指示对象当前处于哪种状态。

挑战对象以“pending”状态创建。当客户端响应挑战（见第7.6.2.2节）且服务端开始尝试验证挑战完成情况时，挑战对象将转换为“processing”（正在处理）状态。在“processing”状态下，服务端可多次尝试完成验证（见第7.6.2.3节），且客户端重试请求不会导致状态更改。若验证成功，挑战进入“valid”（有效）状态；若发生错误，挑战进入“invalid”（无效）状态，状态转换见图6。对于快速订单，挑战对象直接进入“processing”状态。

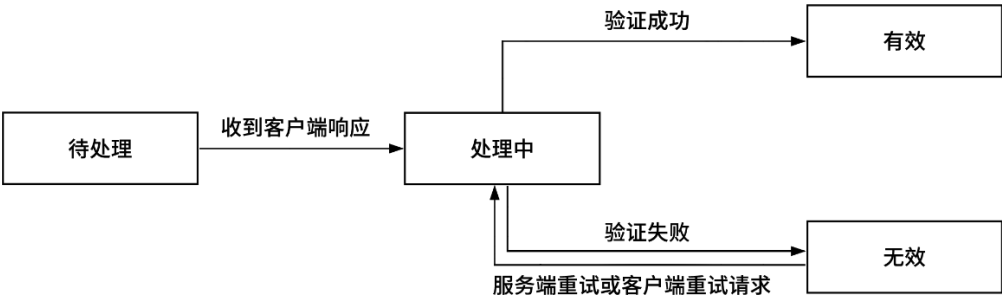


图 6 挑战对象的状态转换

授权对象在“pending”状态下创建。若授权中列出的挑战之一转换为“valid”状态，则授权也更改为“valid”状态。若客户端尝试完成验证但失败，或者若在授权仍在等待期间出现错误，则授权将转换为“invalid”状态。一旦授权处于“valid”状态，它就会过期（“expired”）、被客户端停用（“deactivated”，见第7.6.2.4节）或被服务端撤销（“revoked”），见图。

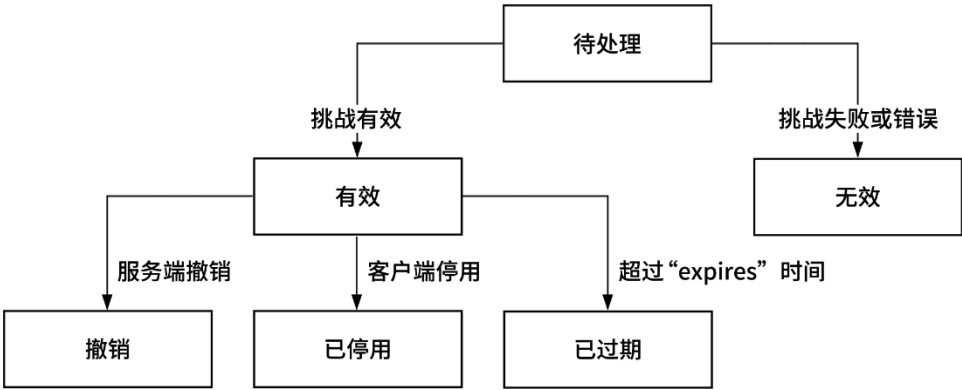


图 7 授权对象的状态转换

订单对象在“pending”状态下创建。一旦订单对象中列出的所有授权都处于“valid”状态，订单就会转换为“ready”状态。在客户端向订单的“finalize” URL提交证书申请并且CA开始证书的签发处理后，订单进入“processing”状态。一旦证书签发，订单就进入“valid”状态。若在这些阶段中的任何一个地方发生错误，订单将进入“invalid”状态。若订单过期或其授权之一的最终状态而不是“valid”（“expired”，“revoked”，“deactivated”（“过期”、“撤销”、“停用”）），则订单也会进入“invalid”状态，见图。对于快速订单，直接进入“processing”状态。

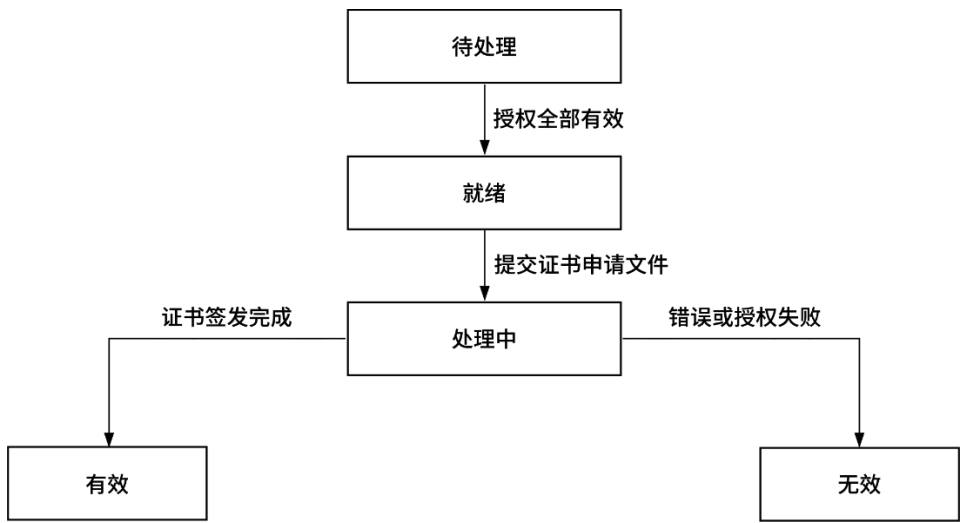


图 8 订单对象的状态转换

帐户对象是在“valid”状态下创建的，因为在成功的newAccount请求之后不需要进一步的操作来创建帐户。若帐户被客户端停用或被服务端终止，它会移动到相应的状态，见图。

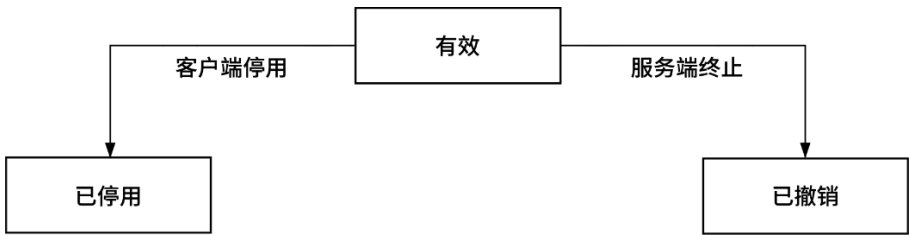


图 9 帐户对象的状态转换

其中一些状态可能永远不会出现在“status”字段中，具体取决于服务端行为。例如：同步操作中的服务端永远不会显示订单处于“processing”状态。立即删除过期授权的服务端永远不会显示“expired”状态的授权。

7.3 获取随机数

在向服务端发送POST请求之前,ACME客户端应有一个新的抗重放随机数以放入JWS的“nonce”头中。在大多数情况下，客户端会从之前的请求中提取随机数。然而，客户端有时可能需要获取一个新的随机数，例如：在它对服务端的第一次请求时，或者若现有的随机数不再有效。

为获取新的随机数，客户端向服务端的新Nonce资源发送 HEAD 请求。服务端的响应应包括一个Replay-Nonce头字段，其中包含一个新的随机数并且应有状态码 204（无内容）。服务端还应响应对该资源的GET请求，返回一个空主体（同时仍然提供Replay-Nonce头），状态代码为204（无内容）。

```
HEAD /acme/new-Nonce HTTP/1.1
Host: example.com
HTTP/1.1 200 OK
Replay-Nonce: oFvn1FP1wIhRlYS2jTaXbA
Cache-Control: no-store
Link: <https://example.com/acme/directory>;rel="index"
```

来自newNonce资源的代理缓存响应可能会导致客户端重复接收相同的随机数,从而导致“badNonce”错误。服务端应在对newNonce资源的响应中包含带有“no-store”指令的Cache-Control头字段,以防止缓存此资源。

7.4 帐户管理

本条规定了帐户的创建、查找、更新、密钥更换及停用等管理机制。

客户端通过向服务端的 newAccountURL发送POST请求来在服务端上创建一个新帐户。请求的主体是一个存根帐户对象,其中包含以下字段的某些子集:

- a) contact (可选, 字符串数组): 与第 7.2.2 节中定义的相应服务端字段含义相同。
- b) termsOfServiceAgreed (可选, 布尔值): 与第 7.2.2 节定义的相应服务端字段含义相同。
- c) onlyReturnExisting (可选, 布尔值): 若此字段的值为“true”, 则服务端不应创建新帐户 (若尚不存在)。这允许客户端根据帐户密钥查找帐户 URL (请参阅第 7.4.1 节);
- d) externalAccountBinding (可选, 对象): 与第 7.2.2 节定义的相应服务端字段含义相同。

```
POST /acme/new-account HTTP/1.1
Host: example.com
Content-Type: application/jose+json
{
  "protected": Base64url({
    "alg": "SM2",
    "jwk": {...},
    "nonce": "6S8Iq0GY7eL2lsGoTZYifg",
    "url": "https://example.com/acme/new-account"
  }),
  "payload": Base64url({
    "termsOfServiceAgreed": true,
    "contact": [
      "mailto:cert-admin@example.org",
      "mailto:admin@example.org"
    ]
  }),
  "signature": "RZPOnYoPs1PhjszF...-nh6X1qt0FPB519I"
}
```

服务端应忽略客户端发送的帐户对象中“orders”字段中提供的任何值,以及它不识别的任何其他字段。若将来指定新的字段,这些字段的规范应描述它们是否可由客户端提供。服务端不应在生成的帐户对象中反映“onlyReturnExisting”字段或任何无法识别的字段。这允许客户端检测服务端何时不支持扩展字段。

服务端应验证“contact”字段中的联系URL是否有效并被服务端支持。若服务端验证联系URL,它应支持“mailto”方案。客户端不应在包含“hfields”的“contact”字段中提供“mailto”URL,或在“to”组件中提供多个“addr-spec”。若服务端遇到不符合这些条件的“mailto”联系URL,则它应认为无效而拒绝它。

若服务端拒绝使用不支持的方案的联系URL，它应返回一个类型为“unsupportedContact”的错误，以及错误的描述和服务端认为可接受的联系URL类型。若服务端拒绝使用支持的方案但值是无效的联系URL，则服务端应返回类型为“invalidContact”的错误。

若服务端希望要求客户端同意使用ACME服务的条款，它应在目录对象的“meta”字段的“termsOfService”子字段中指示可访问此类条款的URL，并且服务端应拒绝没有将“termsOfServiceAgreed”字段设置为“true”的newAccount请求。默认情况下，客户不应自动同意条款。相反，他们应需要一些用户交互来同意条款。

服务端创建一个帐户并存储用于验证JWS的公钥（即 JWS头的“jwk”元素）以验证来自该帐户的将来请求。服务端在201（已创建）响应中返回此帐户对象，并在Location头字段中包含帐户URL。帐户URL用作JWS中的“kid”值，用于验证此帐户的后续请求（请参阅第6.4节）。帐户URL还用于请求对此帐户进行管理操作，如下所述。

```
HTTP/1.1 201 Created
Content-Type: application/json
Replay-Nonce: D8s4D2mLs8Vn-goWuPQeKA
Link: <https://example.com/acme/directory>;rel="index"
Location: HTTPS://example.com/acme/acct/evOfKhNU60wg
{
  "status": "valid",
  "contact": [
    "mailto:cert-admin@example.org",
    "mailto:admin@example.org"
  ],
  "orders": "https://example.com/acme/acct/evOfKhNU60wg/orders"
}
```

7.4.1 查找给定密钥的帐户 URL

若服务端收到一个用密钥签名的newAccount请求，而它已经有一个以提供的帐户密钥注册的帐户，则它应返回一个状态代码为200（OK）的响应，并在Location头字段中提供该帐户的URL。此响应的主体表示此请求之前服务端上存在的帐户对象；应忽略请求对象中的任何字段。这允许具有帐户密钥但没有相应帐户URL的客户端恢复帐户URL。

若客户端希望找到现有帐户的URL，并且不希望创建帐户（若帐户不存在），则它应通过使用有效载荷具有“onlyReturnExisting”字段设置为“true”（{“onlyReturnExisting”: true}）。若客户端发送这样的请求并且帐户不存在，则服务端应返回状态代码为400（错误请求）的错误响应，错误类型为“urn:ietf:params:acme:error:accountDoesNotExist”。

7.4.2 帐户更新

若客户端希望在将来更新此信息，它会向帐户URL发送一个包含更新信息的POST请求。服务端应忽略对“orders”字段、“termsOfServiceAgreed”字段（见第7.4.3节）、“status”字段（第7.4.6节允许的除外）或它不识别的任何其他字段的任何更新。若服务端接受更新，服务端应返回一个带有200（OK）状态码的响应和结果帐户对象。

例如：要更新上述帐户中的联系信息，客户端可发送以下请求：

```
POST /acme/acct/evOfKhNU60wg HTTP/1.1
```

```
Host: example.com
Content-Type: application/jose+json
{
  "protected": Base64url({
    "alg": "SM2",
    "kid": "https://example.com/acme/acct/evOfKhNU60wg",
    "nonce": "ax5RnthDqp_Yf4_HZnFLmA",
    "url": "https://example.com/acme/acct/evOfKhNU60wg"
  }),
  "payload": Base64url({
    "contact": [
      "mailto:certificates@example.org",
      "mailto:admin@example.org"
    ]
  }),
  "signature": "hDXzvcj8T6fbFbmn...rDzXzzvzpRy64N0o"
}
```

7.4.3 服务条款的变更

如上所述，客户端可通过将其帐户对象中的“termsOfServiceAgreed”字段设置为“true”来指示其同意CA的服务条款。

若服务端在客户端最初同意后更改了它的服务条款，并且服务端不愿意在没有明确同意新条款的情况下处理请求，则服务端应返回一个错误响应，状态代码为403（禁止），错误类型为“urn:ietf:params:acme:error:userActionRequired”。此响应应包含一个Link头字段，其中包含链接关系“terms-of-service”和最新的服务条款URL。

返回的错误信息还应包含一个“instance”字段，指示客户端应指示用户（操作员）访问的URL，以便获取有关如何同意条款的说明。

```
HTTP/1.1 403 Forbidden
Replay-Nonce: T81bdZroZ2ITWSondpTmA
Link: <https://example.com/acme/directory>;rel="index"
Link: <https://example.com/acme/terms/2017-6-02>;rel="terms-of-service"
Content-Type: application/problem+json
Content-Language: en
{
  "type": "urn:ietf:params:acme:error:userActionRequired",
  "detail": "Terms of service have changed",
  "instance": "https://example.com/acme/agreement/?token=W8Ih3PswD-8"
}
```

7.4.4 外部账户绑定

服务端可能要求“externalAccountBinding”字段的值出现在“newAccount”请求中。这可用于将ACME帐户与非ACME系统（例如CA客户数据库）中的现有帐户相关联。

为启用ACME帐户绑定，运行ACME服务端的CA应使用ACME之外的某种机制为ACME客户端提供MAC密钥和密钥标识符。密钥标识符应是ASCII字符串。MAC密钥应以Base64url编码的形式提供，以最大限度地提高非ACME供应系统和ACME客户端之间的兼容性。

ACME客户端然后计算绑定JWS以指示外部帐户持有人对ACME帐户密钥的批准。此JWS的有效负载是正在注册的ACME帐户密钥，采用JWK形式。JWS的受保护头应满足以下条件：

- a) “alg” 字段应指明基于 MAC 算法；
- b) “kid” 字段应包含 CA 提供的密钥标识符；
- c) “nonce” 字段不应存在；
- d) “url” 字段应设置为与外部 JWS 相同的值。

JWS 的“签名”字段将包含使用 CA 提供的 MAC 密钥计算的 MAC 值。

```
POST /acme/new-account HTTP/1.1
Host: example.com
Content-Type: application/jose+json
{
  "protected": Base64url({
    "alg": "SM2",
    "jwk": /* account key */,
    "nonce": "K60BWPrMQG9SDxBDS_xtSw",
    "url": "https://example.com/acme/new-account"
  }),
  "payload": Base64url({
    "contact": [
      "mailto:cert-admin@example.org",
      "mailto:admin@example.org"
    ],
    "termsOfServiceAgreed": true,
    "externalAccountBinding": {
      "protected": Base64url({
        "alg": "SM2",
        "kid": /* key identifier from CA */,
        "url": "https://example.com/acme/new-account"
      }),
      "payload": Base64url(/* same as in "jwk" above */),
      "signature": /* MAC using MACkey from CA */
    }
  }),
  "signature": "5TWiqIYQfIDfALQv...x9C2mg8JGPx15bI4"
```

若CA要求newAccount请求包含一个“externalAccountBinding”字段，应在目录对象的“meta”字段的“externalAccountRequired”子字段中提供值“true”。若CA收到一个没有“externalAccountBinding”字段的新账户请求，则它应回复一个“externalAccountRequired”类型的错误。

当CA收到包含“externalAccountBinding”字段的新Account请求时，它决定是否验证绑定。若CA不验证绑定，则它不应在结果帐户对象（若有）中反映“externalAccountBinding”字段。为验证帐户绑定，CA应采取以下步骤：

验证该字段的值是否为格式正确的JWS；

验证JWS保护字段是否满足上述条件；

检索与“kid”字段中的密钥标识符对应的MAC密钥；

验证JWS上的MAC是否使用该MAC密钥进行验证；

验证JWS的有效负载是否表示与用于验证外部JWS相同的密钥（即外部 JWS的“jwk”字段）。

若所有这些检查都通过并且CA创建了一个新帐户，则CA可认为该新帐户与MAC密钥对应的外部帐户相关联。CA返回的帐户对象应包含一个“externalAccountBinding”字段，其值与请求中的字段相同。若这些检查中的任何一个失败，则CA应拒绝newAccount请求。

7.4.5 账户密钥更换

客户端可能需要更改与帐户关联的公钥，以便从已发生的密钥泄露事件中恢复，或主动降低未察觉的密钥泄露所带来的安全影响。

要更改与帐户关联的密钥，客户端会向服务端发送一个请求，其中包含新旧密钥的签名。新密钥的签名包含了账户URL和旧密钥，表示新密钥持有者请求从旧密钥持有者手中接管账户。旧密钥的签名包含了该请求及其签名，并表示旧密钥持有者同意更换请求。

要创建此请求对象，客户端首先构造一个keyChange对象，描述要更新的帐户及其帐户密钥：

- a) account（必需，字符串）：正在修改的帐户的URL。该字段的内容应是Location头字段中提供的准确字符串，以响应创建帐户的newAccount请求；
- b) oldKey（必需，JWK）：旧密钥的JWK表示。

然后，客户端将keyChange对象封装在一个内部JWS中，并使用请求的新帐户密钥进行签名。这个内部JWS成为外部JWS的有效负载，即ACME请求的主体。

外部JWS应满足ACME JWS请求主体的正常要求（请参阅第6.4节）。内部JWS应满足正常要求，但有以下区别：

- a) 内部JWS应有一个“jwk”头参数，包含新密钥对的公钥；
- b) 内部JWS应具有与外部JWS相同的“url”头参数；
- c) 内部JWS应省略“nonce”头参数。

此更新具有来自旧密钥和新密钥的签名，以便服务端可验证两个密钥的持有者是否都同意更改。签名是嵌套的，以保留POST消息上的所有签名都由一个密钥签名的属性。内部JWS有效地代表了新密钥持有者从旧密钥持有者手中接管帐户的请求。外部JWS代表当前账户持有人同意此请求。

POST /acme/key-change HTTP/1.1

Host: example.com Content-Type: application/jose+json

```
{
  "protected": Base64url({
    "alg": "SM2",
    "kid": "https://example.com/acme/acct/evOfKhNU60wg",
    "nonce": "S9Xa0cxP5McpnTcWPIhYuB",
    "url": "https://example.com/acme/key-change"
  }),
  "payload": Base64url({
    "protected": Base64url({
```

```

    "alg": "SM2",
    "jwk": /* new key */,
    "url": "https://example.com/acme/key-change"
  }),
  "payload": Base64url({
    "account": "https://example.com/acme/acct/evOfKhNU60wg",
    "oldKey": /* old key */
  }),
  "signature": "Xe8B94RD30Azj2ea...8BmZIRtcSKPSd8gU"
}),
"signature": "5TWiqIYQfIDfALQv...x9C2mg8JGPx15bI4"
}

```

在收到keyChange请求时，除了典型的JWS验证之外，服务端还应执行以下步骤：

- a) 验证 POST 请求属于当前活动的帐户，如第 6 章所述；
- b) 检查 JWS 的负载是否是格式正确的 JWS 对象（“内部 JWS”）；
- c) 检查内部 JWS 的受 JWS 保护的头部是否有“jwk”字段；
- d) 检查内部 JWS 是否使用其“jwk”字段中的密钥进行验证；
- e) 检查内部 JWS 的负载是否是格式正确的 keyChange 对象（如上所述）；
- f) 检查内部和外部 JWS 的“url”参数是否相同；
- g) 检查 keyChange 对象的“account”字段是否包含与旧密钥匹配的帐户的 URL 即外部 JWS 中的“kid”字段）；
- h) 检查 keyChange 对象的“oldKey”字段是否与相关帐户的帐户密钥相同；
- i) 检查不存在其帐户密钥与内部 JWS 的“jwk”头参数中的密钥相同的帐户。

若所有这些检查都通过，则服务端通过用新公钥替换旧帐户密钥来更新相应的帐户，并返回状态代码 200（OK）。否则，服务端以错误状态代码和描述错误信息作为响应。若有提供新密钥的现有帐户，则服务端应使用状态代码 409（冲突）并在 Location 头字段中提供该帐户的 URL。

更改帐户的帐户密钥不应对该帐户产生任何其他影响。例如：服务端不应根据帐户密钥的更改使挂起的订单或授权交易无效。

7.4.6 账户停用

客户可通过将签名更新发布到状态字段为“deactivated”的帐户 URL 来停用帐户。当帐户密钥被泄露或停用时，客户可能希望这样做。已停用的帐户将无法再请求签发证书或访问与该帐户相关的资源，例如订单或授权。若服务端从已停用的帐户接收到 POST 或 POST-as-GET，服务端应返回状态代码为 401（未授权）的错误响应，错误类型为“urn:ietf:params:acme:error:unauthorized”。

```

POST /acme/acct/evOfKhNU60wg HTTP/1.1
Host: example.com
Content-Type: application/jose+json
{
  "protected": Base64url({
    "alg": "SM2",
    "kid": "https://example.com/acme/acct/evOfKhNU60wg",
    "nonce": "ntuJWWSic4WVNSqeUmshgg",

```

```

    "url": "https://example.com/acme/acct/evOfKhNU60wg"
  }},
  "payload": Base64url({
    "status": "deactivated"
  }),
  "signature": "earzVLd3m5M4xJzR...bVTqn7R08AKOVf3Y"
}

```

服务端应验证请求是否由帐户密钥签名。若服务端接受停用请求，它会回复200（OK）状态代码和帐户对象的当前内容。

一旦帐户被停用，服务端不应接受该帐户密钥授权的进一步请求。服务端应取消任何由帐户密钥授权的未决操作，例如证书订单。服务端可采取不同的动作来响应帐户停用，例如：删除与该帐户相关的数据或向该帐户的联系人发送邮件。服务端不应撤销由停用帐户签发的证书，因为这可能会导致使用这些证书的服务端的操作中断。ACME不提供重新激活已停用帐户的方法。

7.5 申请签发证书

客户端通过向服务端的新Order资源发送POST请求来开始证书签发过程。POST的主体是一个JWS对象，其JSON负载是第7.2.3节中定义的订单对象的子集，包含描述要签发的证书的字段：

- a) identifiers（必需，对象数组）：客户端希望为其提交订单的主体标识对象数组；
 - 1) type（必需，字符串）：主体标识的类型；
 - 2) value（必需，字符串）：主体标识本身；
- b) notBefore（可选，字符串）：证书中 notBefore 字段的请求值；
- c) notAfter（可选，字符串）：证书中 notAfter 字段的请求值；

POST /acme/new-order HTTP/1.1

Host: example.com

Content-Type: application/jose+json

```

{
  "protected": Base64url({
    "alg": "SM2",
    "kid": "https://example.com/acme/acct/evOfKhNU60wg",
    "nonce": "5XJ1L31EkMG7tR6pA00c1A",
    "url": "https://example.com/acme/new-order"
  }),
  "payload": Base64url({
    "identifiers": [
      { "type": "dns", "value": "www.example.org" },
      { "type": "dns", "value": "example.org" }
    ],
    "notBefore": "2016-01-01T00:04:00+04:00",
    "notAfter": "2016-01-08T00:04:00+04:00"
  }),
  "signature": "H6ZXtGjTZyUnPeKn...wEA4Tk1Bdh3e454g"
}

```

若申请电子邮件证书,则应把identifiers的“type”设置为“email”,“value”设置为申请邮件证书的电子邮件地址,如:

```
"identifiers":[
  { "type": "email", "value": "alexey@example.org", "challengeType": "email-http-00" }
]
```

CSR是对客户端要签发的证书内容的签名请求进行编码。CSR应指示与初始newOrder请求完全相同的一组请求标识符。“dns”类型的请求标识符应出现在CSR的主题名称的commonName部分或出现在CSRsubjectAltName 扩展的 extensionRequest 属性中,或两者都有。“email”类型的请求标识符应出现在CSR的主题名称的E字段或CSRsubjectAltName 扩展的 extensionRequest 属性中,或两者都有。(这些标识符可以任何排序顺序出现。)定义新标识符类型的规范应指定这些标识符可出现在证书签名请求中的什么位置。

若服务端不能按照指定的方式满足请求,则服务端应返回一个错误,并且它不应签发包含与请求内容不同的证书。若服务端要求以某种方式修改请求,它应使用适当的错误类型和描述来指示所需的更改。

若服务端同意签发所请求的证书,它会以201(已创建)响应进行响应。此响应的主体是一个订单对象,它反映了客户端的请求以及客户端在签发证书之前应完成的任何授权。服务器在 Location 头字段中返回一个订单URL。

```
HTTP/1.1 201 Created
Replay-Nonce: MYAuvOpaoIiywTezizk5vw
Link: <https://example.com/acme/directory>;rel="index"
Location:HTTPS://example.com/acme/order/T0locE8rfgo
{
  "status": "pending",
  "expires": "2016-01-05T14:09:07.99Z",
  "notBefore": "2016-01-01T00:00:00Z",
  "notAfter": "2016-01-08T00:00:00Z",
  "identifiers":[
    { "type": "dns", "value": "www.example.org", "challengeType": "http-01" },
    { "type": "dns", "value": "example.org", "challengeType": "dns-01" }
  ],
  "authorizations":[
    "https://example.com/acme/authz/PAniVnsZcis",
    "https://example.com/acme/authz/r4HqLzrSrpI"
  ],
  "finalize": "https://example.com/acme/order/T0locE8rfgo/finalize"
}
```

服务端返回的订单对象代表一个承诺,若客户端在“expires”时间之前满足服务端的要求,则服务端将同意根据请求完成订单并签发请求的证书。在 order 对象中,“authorizations”数组中引用的任何状态为“pending”的授权代表客户端应在服务端签发证书之前完成的授权事务。若客户端未能在“expires”时间之前完成所需的操作,则服务端应将订单状态更改为“invalid”并且可删除订单资

源。客户端不对返回的订单对象中“identifiers”或“authorizations”元素的排序顺序做出任何假设。

一旦客户端认为已满足服务端的要求，应向订单资源的最终确定 URL 发送 POST 请求。POST 正文应包含 CSR：

- a) csr（可选，字符串）：CSR 编码所请求国际算法证书的参数，CSR 以 DER 格式的 Base64url 编码版本发送；
- b) csrSign（可选，字符串）：CSR 编码所请求商密签名证书的参数，CSR 以 DER 格式的 Base64url 编码版本发送；
- c) csrEncrypt（可选，字符串）：CSR 编码所请求商密加密证书的参数，CSR 以 DER 格式的 Base64url 编码版本发送；
- d) csrSM2（可选，字符串）：CSR 编码所请求商密单密码用途证书的参数，CSR 以 DER 格式的 Base64url 编码版本发送；
- e) csrSignRSA（可选，字符串）：CSR 编码所请求国际算法签名证书的参数，CSR 以 DER 格式的 Base64url 编码版本发送；
- f) csrEncryptRSA（可选，字符串）：CSR 编码所请求国际算法加密证书的参数，CSR 以 DER 格式的 Base64url 编码版本发送。

```
POST /acme/order/T0locE8rfgo/finalize HTTP/1.1
Host: example.com
Content-Type: application/jose+json
{
  "protected": Base64url({
    "alg": "SM2",
    "kid": "https://example.com/acme/acct/evOfKhNU60wg",
    "nonce": "5XJlL3lEkMG7tR6pA00c1A",
    "url": "https://example.com/acme/new-order"
  }),
  "payload": Base64url({
    "csr": "MIIBPTCBxAIBADBFMQ...FS6aKdZeGsysoCo4H9P",
    "csrSign": "MIIBPTCBxAIBWghADBFM...FS6aysoCo4H9P",
    "csrEncrypt": "MIIBPTCBA88Dsers...FSWghsyCo4g788fH9P"
  }),
  "signature": "uOrUfIIk5RyQ...nw62Aylcl6AB"
}
```

若CA不愿意签发与提交的CSR相对应的证书，则完成订单的请求将导致错误。例如：

- a) 若 CSR 和订单主体标识不同；
- b) 若帐户未获取 CSR 中指示的主体标识的授权；
- c) 若 CSR 请求的扩展项 CA 不愿意包含。

在这种情况下，服务端返回的错误信息应使用错误代码“badCSR”并在其“详细信息”字段中描述 CSR 被拒绝的具体原因。在返回这样的错误之后，服务端应让订单处于“ready”状态，以允许客户端提交一个新的带有修改后的 CSR 的最终确定请求。

若订单未处于“ready”状态，则完成订单的请求将导致错误。在这种情况下，服务端应返回403（禁止）错误以及错误类型为“orderNotReady”。然后，客户端应向订单资源发送POST-as-GET请求以获取其当前状态。订单状态将指示客户端应采取的行动（见下文）。

若完成订单的请求成功，服务端将返回 200（OK）和更新的订单对象。订单状态将指示客户端应采取的行动：

- a) “invalid”：不签发证书。考虑放弃此订购流程；
- b) “pending”：服务端认为客户端没有满足要求。检查“authorizations”数组以查找仍未决的条目；
- c) “ready”：服务端认为已满足要求，正在等待最终确定。客户端可提交最新请求；
- d) “processing”：正在签发证书。在响应的 Retry-After 头字段中给定的时间（若有）之后发送 POST-as-GET 请求；
- e) “valid”：服务端已签发证书并将其 URL 提供给订单的“certificate”字段。客户端可下载证书。根据用户是否提供了商密签名证书和加密证书的证书申请文件，会同时返回商密证书下载 URL “certificateSign”和“certificateEncrypt”或“certificateSM2”。若 CA 生成加密密钥对，会在返回商密证书下载 URL 的同时返回密钥数字信封的下载 URL “keyEnvelopEncrypt”。

HTTP/1.1 200 OK

Replay-Nonce: CGf81JWBsq8QyIgPCi9Q9X

Link: <https://example.com/acme/directory>;rel="index"

Location:HTTPS://example.com/acme/order/T0locE8rfgo

```
{
  "status": "valid",
  "expires": "2016-01-20T14:09:07.99Z",
  "notBefore": "2016-01-01T00:00:00Z",
  "notAfter": "2016-01-08T00:00:00Z",
  "identifiers":[
    { "type": "dns", "value": www.example.org },
    { "type": "dns", "value": "example.org" }
  ],
  "authorizations":[
    "https://example.com/acme/authz/PAniVnsZcis",
    "https://example.com/acme/authz/r4HqLzrSrpI"
  ],
  "finalize": "https://example.com/acme/order/T0locE8rfgo/finalize",
  "certificate": "https://example.com/acme/cert/mAt3xBGaobw",
  "certificateSign": "https://example.com/acme/cert/sign/mAt3xBGaobw",
  "certificateEncrypt": "https://example.com/acme/cert/encrypt/mAt3xBGaobw",
  "keyEnvelopEncrypt": "https://example.com/acme/cert/encrypt/mAt3xBGaobw",
}
```

7.5.1 快速订单签发证书

对于快速订单，没有预验证过程，客户端通过向服务端的新Order资源发送POST请求开始域名验证和证书签发过程，应同时指定挑战类型和提交证书签发请求CSR。POST的主体是一个JWS对象，其JSON负载是第7.2.3节中定义的订单对象的子集，包含描述要签发的证书的字段：

- a) identifiers（必需，对象数组）：客户端希望为其提交订单的主体标识对象数组；
 - 1) type（必需，字符串）：主体标识的类型；
 - 2) value（必需，字符串）：主体标识本身；
 - 3) challengeType（必需，字符串）：挑战类型
- b) notBefore（可选，字符串）：证书中 notBefore 字段的请求值，按照 RFC 3339 规定的日期格式；
- c) notAfter（可选，字符串）：证书中 notAfter 字段的请求值，按照 RFC 3339 规定的日期格式。
- d) csr（可选，字符串）：CSR 编码所请求国际算法证书的参数，CSR 以 DER 格式的 Base64url 编码版本发送；
- e) csrSign（可选，字符串）：CSR 编码所请求商密签名证书的参数，CSR 以 DER 格式的 Base64url 编码版本发送；
- f) csrEncrypt（可选，字符串）：CSR 编码所请求商密加密证书的参数，CSR 以 DER 格式的 Base64url 编码版本发送；
- g) csrSM2（可选，字符串）：CSR 编码所请求商密单密码用途证书的参数，CSR 以 DER 格式的 Base64url 编码版本发送；
- h) csrSignRSA（可选，字符串）：CSR 编码所请求国际算法签名证书的参数，CSR 以 DER 格式的 Base64url 编码版本发送；
- i) csrEncryptRSA（可选，字符串）：CSR 编码所请求国际算法加密证书的参数，CSR 以 DER 格式的 Base64url 编码版本发送。

POST /acme/new-order HTTP/1.1

Host: example.com

Content-Type: application/jose+json

```
{
  "protected": Base64url({
    "alg": "SM2",
    "kid": "https://example.com/acme/acct/evOfKhNU60wg",
    "nonce": "5XJ1L31EkMG7tR6pA00c1A",
    "url": "https://example.com/acme/new-order"
  }),
  "payload": Base64url({
    "identifiers": [
      { "type": "dns", "value": "www.example.org", "challengeType": "http-01" },
      { "type": "dns", "value": "example.org", "challengeType": "dns-01" }
    ],
    "csr": "MIIBPTCBxAIBADBFBMQ...FS6aKdZeGsysoCo4H9P",
    "csrSign": "MIIBPTCBxAIBWghADBFBM...FS6aysoCo4H9P",
    "csrEncrypt": "MIIBPTCBA88Dsers...FSWghsyCo4g788fH9P",
    "notBefore": "2016-01-01T00:04:00+04:00",
  })
}
```

```

    "notAfter": "2016-01-08T00:04:00+04:00"
  }
  "signature": "H6ZXtGjTZyUnPeKn...wEA4Tk1Bdh3e454g"
}

```

若申请电子邮件证书，则应把identifiers的“type”设置为“email”，“value”设置为申请邮件证书的电子邮件地址，如：

```

"identifiers": [
  { "type": "email", "value": "alexey@example.org", "challengeType": "email-http-00" }
]

```

CSR是对客户端要签发的证书内容的签名请求进行编码。CSR应指示与初始newOrder请求完全相同的一组请求标识符。“dns”类型的请求标识符应出现在CSR的主题名称的commonName部分或出现在CSRsubjectAltName 扩展的 extensionRequest 属性中，或两者都有。“email”类型的请求标识符应出现在CSR的主题名称的E字段 或CSRsubjectAltName 扩展的 extensionRequest 属性中，或两者都有。（这些标识符可以任何排序顺序出现。）定义新标识符类型的规范应指定这些标识符可出现在证书签名请求中的什么位置。

若服务端不能按照指定的方式满足请求，则服务端应返回一个错误，并且它不应签发包含与请求内容不同的证书。若服务端要求以某种方式修改请求，它应使用适当的错误类型和描述来指示所需的更改。

若服务端同意签发所请求的证书，它会以201（已创建）响应进行响应。此响应的主体是一个订单对象，它反映了客户端的请求以及客户端在签发证书之前应完成的任何授权。服务器在 Location 头字段中返回一个订单URL。

```

HTTP/1.1 201 Created
Replay-Nonce: MYAuvOpaoIiywTezizk5vw
Link: <https://example.com/acme/directory>;rel="index"
Location: HTTPS://example.com/acme/order/T0locE8rfgo
{
  "status": "processing",
  "expires": "2016-01-05T14:09:07.99Z",
  "notBefore": "2016-01-01T00:00:00Z",
  "notAfter": "2016-01-08T00:00:00Z",
  "identifiers": [
    { "type": "dns", "value": "www.example.org", "challengeType": "http-01" },
    { "type": "dns", "value": "example.org", "challengeType": "dns-01" }
  ],
  "authorizations": [
    "https://example.com/acme/authz/PAniVnsZcis",
    "https://example.com/acme/authz/r4HqLzrSrpI"
  ],
}

```

若CA不愿意签发与提交的CSR相对应的证书，则创建订单的请求将导致错误。例如：

- a) 若 CSR 和订单主体标识不同；
- b) 若帐户未获取 CSR 中指示的主体标识的授权；

c) 若 CSR 请求的扩展项 CA 不愿意包含。

在这种情况下，服务端返回的错误信息应使用错误代码“badCSR”并在其“详细信息”字段中描述 CSR 被拒绝的具体原因。在返回这样的错误之后，服务端应让订单处于“invalid”状态，客户端应提交一个新的带有修改后的 CSR 重新创建证书订单。

客户端完成授权后，客户端应向订单资源发送 POST-as-GET 请求以获取其当前状态。订单状态将指示客户端应采取的行动（见下文）。

若完成订单的请求成功，服务端将返回 200 (OK) 和更新的订单对象。订单状态将指示客户端应采取的行动：

- a) “invalid”：不签发证书。考虑放弃此订购流程；
- b) “processing”：正在验证挑战或者签发证书。在响应的 Retry-After 头字段中给定的时间（若有）之后再次发起验证挑战；
- c) “valid”：服务端已签发证书并将其 URL 提供给订单的“certificate”字段。客户端可下载证书。根据用户是否提供了商密签名证书和加密证书的证书申请文件，会同时返回商密证书下载 URL “certificateSign”和“certificateEncrypt”或“certificateSM2”。若 CA 生成加密密钥对，会在返回商密证书下载 URL 的同时返回密钥数字信封的下载 URL “keyEnvelopEncrypt”。

HTTP/1.1 200 OK

Replay-Nonce: CGf81JWBsq8QyIgPCi9Q9X

Link: <https://example.com/acme/directory>;rel="index"

Location: HTTPS://example.com/acme/order/T0locE8rfgo

```
{
  "status": "valid",
  "expires": "2016-01-20T14:09:07.99Z",
  "notBefore": "2016-01-01T00:00:00Z",
  "notAfter": "2016-01-08T00:00:00Z",
  "identifiers": [
    { "type": "dns", "value": "www.example.org", "challengeType": "http-01" },
    { "type": "dns", "value": "example.org", "challengeType": "dns-01" }
  ],
  "authorizations": [
    "https://example.com/acme/authz/PAniVnsZcis",
    "https://example.com/acme/authz/r4HqLzrSrpI"
  ],
  "finalize": "https://example.com/acme/order/T0locE8rfgo/finalize",
  "certificate": "https://example.com/acme/cert/mAt3xBGaobw",
  "certificateSign": "https://example.com/acme/cert/sign/mAt3xBGaobw",
  "certificateEncrypt": "https://example.com/acme/cert/encrypt/mAt3xBGaobw",
  "keyEnvelopEncrypt": "https://example.com/acme/cert/encrypt/mAt3xBGaobw",
}
```

7.5.2 预授权

上述订购流程假定授权对象是响应证书订购而被动创建的。一些服务端可能还希望使客户端能够在特定签发的上下文之外主动获取标识符的授权。例如：为一组虚拟主机托管的域名的客户端可能希望

在创建任何虚拟服务端之前获取授权，并且仅在虚拟服务端启动时创建证书。

在某些情况下，运行ACME服务端的CA可能有一个完全外部的非ACME进程，用于授权客户端为标识符签发证书。在这些情况下，CA应为其ACME服务端提供与这些授权相对应的授权对象，并在客户端提交的任何订单中反映它们已经有效。

若CA希望在ACME中允许预授权，它可通过添加字段“newAuthz”和newAuthz资源的URL在其目录中提供“new authorization”资源。

为请求对标识符的授权，客户端向newAuthz资源发送一个POST请求，指定请求授权的标识符。

identifier(required, object): 出现在生成的授权对象中的标识符:

a) type (必需, 字符串): 标识符的类型;

b) value (必需, 字符串): 标识符本身。

```
POST /acme/new-authz HTTP/1.1
Host: example.com
Content-Type: application/jose+json
{
  "protected": Base64url({
    "alg": "SM2",
    "kid": "https://example.com/acme/acct/evOfKhNU60wg",
    "nonce": "uQpSjlRb4vQVCjVYAyyUWg",
    "url": "https://example.com/acme/new-authz"
  }),
  "payload": Base64url({
    "identifier": {
      "type": "dns",
      "value": "example.org"
    }
  }),
  "signature": "nuSDISbWG8mMgE7H...QyVUL68yzf3Zawps"
}
```

快速订单模式不支持预授权。

由于预授权请求中的标识符与授权对象中包含的标识符应完全一致，因此，预授权机制不适用于授权签发包含通配域名的证书。

在处理授权请求之前，服务端应确定它是否愿意为标识符签发证书。例如：服务端应检查标识符是否属于受支持的类型。服务端还可能根据已知高价值域名的黑名单检查名称。若服务端不愿意为这些域名签发证书，它应返回一个状态码为403（禁止）的错误，以及一个描述拒绝原因的错误信息。

若服务端愿意继续，它会根据客户端提交的输入构建一个待定授权对象：

“identifier” 客户端提交的标识符；

“status” 应是“pending”除非服务端有关于客户端授权状态的带外信息；

“challenges” 服务端策略为此标识符选择的验证方式。

服务端为此授权分配一个新的URL，并返回一个201（已创建）响应，其中Location标头字段中包含授权URL，正文中包含JSON授权对象。客户端然后按照第9.6节中描述的过程完成授权过程。

若服务器收到授权对象已存在的标识符的 newAuthz 请求，无论是由 ACME 服务器上的 CA 配置

创建的,还是由处理来自客户端的先前 newAuthz 请求的 ACME 服务器创建的,服务器将返回 200 (OK) 响应 Location 标头字段中的现有授权 URL 和正文中的现有 JSON 授权对象。

7.5.3 下载证书

要下载签发的证书,客户端只需向证书 URL 发送一个 POST-as-GET 请求。

证书的默认格式是 application/pem-certificate-chain (见第 A.1 节)。

服务端可提供一个或多个 Link 关系 “alternate” 头字段。每个这样的字段应表达一个以相同的最终用户证书开始的证书链。这可用于表达各种信任锚的路径。客户端可获取这些备选方案并使用他们自己的启发式方法来决定哪个是最佳的。

以下示例中,下载国际算法证书应使用: POST /acme/cert/mAt3xBGaobw HTTP/1.1, 下载国际算法的加密证书应使用: POST /acme/cert/rsaEncrypt/mAt3xBGaobw HTTP/1.1, 下载商用密码算法的签名证书应使用: POST /acme/cert/sign/mAt3xBGaobw HTTP/1.1, 下载商用密码算法的加密证书应使用: POST /acme/cert/encrypt/mAt3xBGaobw HTTP/1.1, 下载商用密码算法的单证书应使用: POST /acme/cert/sm2/mAt3xBGaobw HTTP/1.1。证书链叠加方式不变,若有多级中级根证书,则按层级关系叠加。

```
POST /acme/cert/mAt3xBGaobw HTTP/1.1
Host: example.com
Content-Type: application/jose+json
Accept: application/pem-certificate-chain
{
  "protected": Base64url({
    "alg": "SM2",
    "kid": "https://example.com/acme/acct/evOfKhNU60wg",
    "nonce": "uQpSjlRb4vQVCjVYAyyUWg",
    "url": "https://example.com/acme/cert/mAt3xBGaobw"
  }),
  "payload": "",
  "signature": "nuSDISbWG8mMgE7H...QyVUL68yzf3Zawps"
}
HTTP/1.1 200 OK
Content-Type: application/pem-certificate-chain
Link: <https://example.com/acme/directory>;rel="index"
-----BEGIN CERTIFICATE-----
[End-entity certificate contents]
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
[Issuer certificate contents]
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
[Other certificate contents]
-----END CERTIFICATE-----
```

证书资源代表一个单一的、不可变的证书。若客户希望获取更新的证书，则客户会启动一个新的订单流程来请求。

因为证书资源在签发完成后是不可变的，所以服务端可通过添加 Expires 和 Cache-Control 头字段来启用资源缓存，指定一个未来时间点。这些头字段与证书的有效期无关。

ACME 客户端可通过在其请求中 Accept 头字段来请求其他格式。例如：客户端可使用媒体类型“application/pkix-cert”或“application/pkcs7-mime”来请求 DER 格式的用户证书。服务端对替代格式的支持是可选的。对于只能表示单个证书的格式，服务端宜提供一个或多个指向一个或多个签发者的“Link:rel=up”头字段，以便 ACME 客户端可构建 TLS 中定义的证书链。

7.6 用户身份核实

7.6.1 主体标识授权

主体标识授权过程通过建立帐户授权来管理给定主体标识的证书。CA 通常依靠多种方式来验证申请具有给定主体标识的证书的最终用户是否实际控制该主体标识，此过程向服务端保证两件事：

- a) 客户端控制帐户密钥对的私钥；并且
- b) 客户端控制待验证的主体标识(域名)。

可重复此过程以将多个主体标识与一个帐户相关联（例如：以请求具有多个主体标识的证书(多域证书)）或将多个帐户与一个主体标识相关联（例如：以允许多个实体管理证书）。

挑战/响应为服务端提供了保证，即帐户持有人就是控制主体标识的最终用户。对于每种类型的挑战，应是这样的情况，为使一个用户成功完成挑战，该用户应：

- a) 持有用于响应挑战的帐户密钥对的私钥；以及
- b) 控制等待验证的主体标识。

附录 C 记录了本文件中定义的挑战如何满足这些要求。新的挑战应记录他们是如何应对的。

ACME 使用可扩展的挑战/响应框架进行主体标识验证。服务端在发送给客户端的授权对象中提出一组挑战（作为“challenges”数组中的对象），客户端通过在 POST 请求中向挑战 URL 发送响应对象来进行响应。

本节描述了一组初始的挑战类型。挑战类型的定义包括：

- a) 挑战对象的内容；
- b) 响应对象的内容；
- c) 服务端如何使用挑战和响应来验证对主体标识的控制。

挑战对象都包含以下基本字段：

- a) type（必需，字符串）：对象中编码的挑战类型；
- b) url（必需，字符串）：可提交响应的 URL；
- c) status（必需，字符串）：此挑战的状态。可能的值是“pending”、“processing”、“valid”和“invalid”；
- d) validated（可选，字符串）：服务端验证此挑战的时间。若“status”字段为“valid”，则此字段是必需的；
- e) error（可选，对象）：服务端验证挑战时发生的错误（若有的话），结构为错误报告。可使用第 6.9.1 节子问题来指示多个错误。有错误的挑战对象应具有等于“processing”或“invalid”的状态。

所有附加字段均由挑战类型指定。若服务端将挑战的“status”设置为“invalid”，它还应包括“error”字段以帮助客户端诊断挑战失败的原因。

不同的挑战允许服务端获取对主体标识不同方面控制的证明。在一些挑战中，比如 HTTP 和 DNS，

客户端直接证明它有能力做与主体标识相关的某些事情。选择在何种情况下向客户端提供哪些挑战是服务端策略的问题。

本节中描述的主体标识验证挑战都与域名验证有关。若未来 ACME 扩展到支持其他类型的主体标识，则需要有新的挑战类型，他们应指定它们适用于哪些类型的主体标识。

7.6.2 授权验证管理

7.6.2.1 授权请求

授权资源由服务端创建，以响应帐户密钥持有者提交的 newOrder 请求；他们的 URL 在对这些请求的响应中提供给客户端。授权对象隐式绑定到用于签署请求的帐户密钥。

当客户端从服务端收到订单 newOrder 请求响应时，它通过向指定的 URL 发送 POST-as-GET 请求来下载授权资源。

```
POST /acme/authz/PAniVnsZcis HTTP/1.1
Host: example.com
Content-Type: application/jose+json
{
  "protected": Base64url({
    "alg": "SM2",
    "kid": "https://example.com/acme/acct/evOfKhNU60wg",
    "nonce": "uQpSjlRb4vQVCjVYAyyUWg",
    "url": "https://example.com/acme/authz/PAniVnsZcis"
  }),
  "payload": "",
  "signature": "nuSDISbWG8mMgE7H...QyVUL68yzf3Zawps"
}
```

提供资源的路径由固定前缀 “/.well-known/” 组成，后面是 pki-validation/ 或 acme-challenge/，加验证文件或验证值。资源的值应是密钥授权的 ASCII 表示。

```
HTTP/1.1 200 OK
Content-Type: application/json
Link: <https://example.com/acme/directory>;rel="index"
{
  "status": "pending",
  "expires": "2016-01-02T14:09:30Z",
  "identifier": {
    "type": "dns",
    "value": "www.example.org"
  },
  "challenges": [
    {
      "type": "http-01",
```

```

    "url": "https://example.com/acme/chall/prV_B7yEyA4",
    "token": "DGyRejmCefe7v4NfDGDkFA"
    "tokenType": "HTTP"
    "tokenPath": "/.well-known/pki-validation/W725G084H61CF8A5C8.txt"
  },
  {
    "type": "dns-01",
    "url": "https://example.com/acme/chall/Rg5dV14Gh1Q",
    "token": "DGyRejmCefe7v4NfDGDkFA",
    "tokenType": "TXT",
    "tokenPath": "W725G084H61CF8A5C8"
  }
  {
    "type": "email-http-00",
    "url": "https://example.com/acme/chall/ABprV_B7yEyA4f",
    "from": "acme-email-challenge@example.com",
    "token": "DGyRejmCefe7v4NfDGDkFA"
  }
  {
    "type": "email-dns-01",
    "url": "https://example.com/acme/chall/ABprV_B7yEyA4f",
    "from": "acme-email-challenge@example.com",
    "token": "DGyRejmCefe7v4NfDGDkFA"
  }
]
}

```

7.6.2.2 授权挑战的响应

为证明对主体标识的控制并获取授权，客户端应根据挑战类型提供所需的挑战响应，并向服务端表明它已准备好等待验证。对于快速订单，则无需响应授权的挑战 URL，服务端会在下单后定时验证挑战。

客户端通过向挑战 URL（而非授权 URL）发送 POST 请求向服务器表明它已准备好进行挑战验证，其中 POST 请求的主体是一个 JWS 对象，其 JSON 有效负载是一个响应对象。对于本文档中定义的所有挑战类型，响应对象是空 JSON 对象（“{}”）。

例如：若客户端要响应上述授权中的“http-01”验证，它将发送以下请求：

```

POST /acme/chall/prV_B7yEyA4 HTTP/1.1
Host: example.com
Content-Type: application/jose+json
{
  "protected": Base64url({
    "alg": "SM2",
    "kid": "https://example.com/acme/acct/evOfKhNU60wg",
    "nonce": "Q_s3MWoqT05TrdkM2MTDcw",

```

```

    "url": "https://example.com/acme/chall/prV_B7yEyA4"
  }},
  "payload": Base64url({}),
  "signature": "9cbg5J01Gf5YLjjz...SpkUfcdPai9uVYYQ"
}

```

服务端通过客户端提供的响应对象更新其验证方法，从而更新授权信息。服务端应忽略响应对象中与当前验证方式无关的任何字段。虽然本文件规定的挑战未定义任何特定的响应字段，但服务端处理机制应兼容未来规范可能的字段扩展。服务端响应 200（OK）状态码，其消息主体为更新后的挑战对象。

若客户端的响应由于任何原因无效或者没有向服务端提供适当的信息来完成验证，则服务端应返回一个 HTTP 错误。在收到这样的错误时，客户端应撤消为完成验证而采取的任何操作，例如：删除已提供给网站服务器的文件。

服务端在完成其中一项验证或全部失败后被称为“finalize(完成)”授权。这是通过为授权设置为“valid”或“invalid”状态来完成的，对应于该帐户是否被授权拥有主体标识。若最终状态是“valid”，则服务端应包含一个“expires”字段。当完成验证后，服务端不再尝试其他方式的验证，并且可修改“expires”字段。服务端不应停止状态为“invalid”的其他验证方式。

通常，验证过程需要一些时间，因此客户端应轮询授权资源以查看何时完成。对于客户端可知道服务端完成验证的验证方法（例如：通过查看来自服务端的 HTTP 或 DNS 请求），客户端在看到来自服务端的验证请求之前不应开始轮询。

为检查授权状态，客户端向授权 URL 发送 POST-as-GET 请求，服务端以当前授权对象进行响应。在验证仍在进行时，服务端响应轮询请求时应返回 200（OK）响应，并且可包含一个 Retry-After 头字段以向客户端建议轮询间隔。

```

POST /acme/authz/PAniVnsZcis HTTP/1.1
Host: example.com
Content-Type: application/jose+json
{
  "protected": Base64url({
    "alg": "SM2",
    "kid": "https://example.com/acme/acct/evOfKhNU60wg",
    "nonce": "uQpSjlRb4vQVCjVYAyyUWg",
    "url": "https://example.com/acme/authz/PAniVnsZcis"
  })),
  "payload": "",
  "signature": "nuSDISbWG8mMgE7H...QyVUL68yzf3Zawps"
}

```

```

HTTP/1.1 200 OK
Content-Type: application/json
Link: <https://example.com/acme/directory>;rel="index"
{
  "status": "valid",
  "expires": "2018-09-09T14:09:01.13Z",

```

```

    "identifier": {
      "type": "dns",
      "value": "www.example.org"
    },
    "challenges": [
      {
        "type": "http-01",
        "url": "https://example.com/acme/chall/prV_B7yEyA4",
        "status": "valid",
        "validated": "2014-12-01T12:05:13.72Z",
        "token": "IlirfxKKXAsHtmzK29Pj8A"
      }
    ]
  }
}

```

7.6.2.3 重试挑战

ACME 挑战通常要求客户端设置一些服务端可查询的网络可访问资源，以验证客户端是否控制主体标识。对于快速订单，服务端会在下单后定时验证挑战，直到挑战验证成功或者订单超时。若服务端支持，客户端也可响应授权的挑战 URL，让服务端立刻验证。

客户端不应反复提交验证请求，除非客户端确信服务端没有正确完成验证。若服务端的初始验证查询失败，服务端应在一段时间后重试查询，以解决设置响应（如 DNS 记录或 HTTP 资源）的延迟。精确的重试计划取决于服务端，但服务端操作员应牢记该计划试图适应的操作场景。鉴于重试旨在解决 HTTP 或 DNS 配置中的传播延迟等问题，通常没有任何理由比每 5 秒或 10 秒更频繁地重试。当服务端还在尝试时，挑战的状态仍然是“processing”；一旦服务端放弃，它只会被标记为“invalid”。

服务端应通过挑战中的“error”字段和 Retry-AfterHTTP 头字段向客户端提供有关其重试状态的信息，以响应对挑战资源的请求。服务端应在每次失败的验证查询后向挑战中的“error”字段添加一个条目。服务端应将 Retry-After 头字段设置为服务端下一次验证查询之后的时间，因为挑战的状态直到那个时间才会改变。

客户端可通过在新的 POST 请求（使用新的随机数等）中重新发送对挑战的响应来明确请求重试。从而客户端在状态发生变化时请求重试（例如：在更新防火墙规则之后）。服务端应在收到这样的 POST 请求后立即重试请求。为避免通过客户端发起的重试导致拒绝服务攻击拦截，服务端应对此类请求进行速率限制。

7.6.2.4 停用授权

若客户端希望放弃为主体标识签发证书的授权，则它可通过向每个授权 URL 发送带有静态对象{"status": "deactivated"}的 POST 请求来请求服务端停用与其关联的每个授权。

```

POST /acme/authz/PAniVnsZcis HTTP/1.1
Host: example.com
Content-Type: application/jose+json
{
  "protected": Base64url({
    "alg": "SM2",

```

```

    "kid": "https://example.com/acme/acct/evOfKhNU60wg",
    "nonce": "xWCM9lGbIyCgue8di6ueWQ",
    "url": "https://example.com/acme/authz/PAniVnsZcis"
  }),
  "payload": Base64url({
    "status": "deactivated"
  }),
  "signature": "srX9Ji7Le9bjszhu...WTFdtuj0bzMtZcx4"
}

```

服务端应验证请求是否由与拥有授权的帐户对应的帐户密钥签名。若服务端接受停用，它应回复 200 (OK) 状态代码和授权对象的更新内容。

服务端不应将停用的授权对象视为可签发证书。

7.6.3 域名控制权验证

7.6.3.1 HTTP 挑战

通过 HTTP 验证，ACME 事务中的客户端通过证明它可在该域名下可访问的服务器上提供 HTTP 资源来证明其对域名的控制。ACME 服务端要求客户端在特定路径提供文件，并以特定字符串作为其内容。

由于域名可能解析为多个 IPv4 和 IPv6 地址，因此服务端将自行决定连接到 DNS A 和 AAAA 记录中找到的至少一个主机。服务端采用指定的 HTTP 或 HTTPS 方式完成挑战验证，若采用 HTTPS 方式，则可不验证 TLS 证书是否可信、是否过期，但是应验证主体标识与证书域名是否匹配。

- a) type (必需，字符串)：字符串 “http-01”；
- b) token (必需，字符串)：唯一标识挑战的随机值。这个值应至少有 128 位的熵。它不应包含 Base64URL 字母表之外的任何字符，并且不应包含 Base64 填充字符 (“=”)。
- c) tokenType (必需，字符串)：字符串 “HTTP” 或 “HTTPS”；
- d) tokenPath (必需，字符串)：验证文件名或验证文件路径名

```

{
  "type": "http-01",
  "url": "https://example.com/acme/chall/prV_B7yEyA4",
  "status": "pending",
  "token": "LoqXcYV8q5ONbJQxbmR7SCTNo3tiAXDfowyjxAjEuX0"
  "tokenType": "HTTP"
  "tokenPath": "/well-known/pki-validation/W725G084H61CF8A5C8.txt"
}

```

客户端通过根据挑战中提供的 “token” 值、“tokenType” 类型和 “tokenPath” 值，客户端将 “token” 值作为待验证的域名在 HTTP 服务器上的 “tokenPath” 地址上的资源来完成此挑战。

提供资源的路径由固定前缀 “/.well-known/” 组成，后跟挑战值。资源的值应是密钥授权的 ASCII 表示。

GET /.well-known/pki-validation/W725G084H61CF8A5C8.txt

或

```
GET /.well-known/acme-challenge/LoqXcYV8q50NbJQxbmR7SCTNo3tiAXDfowyjxAjEuX0
Host: example.org
HTTP/1.1 200 OK
Content-Type: application/octet-stream
LoqXcYV8...jxAjEuX0.9jg46WB3...fm2lmqTI
```

（在上面，“...”表示密钥授权中的令牌和 JWK 指纹已被截断，没有完整显示。）

客户端使用空对象({})进行响应，以确认服务端可验证挑战。

```
POST /acme/chall/prV_B7yEyA4 HTTP/1.1
Host: example.com
Content-Type: application/jose+json
{
  "protected": Base64url({
    "alg": "SM2",
    "kid": "https://example.com/acme/acct/evOfKhNU60wg",
    "nonce": "UQI1PoRi50uXzXuX7V7wL0",
    "url": "https://example.com/acme/chall/prV_B7yEyA4"
  }),
  "payload": Base64url({}),
  "signature": "Q1bURgJoEslbD1c5...3pYdSMLio57mQNN4"
}
```

收到响应后，服务端根据挑战“tokenType”类型和“tokenPath”值来请求 HTTP 服务器验证授权。给定挑战/响应对，服务端通过验证资源是否按预期提供的“token”来验证客户端对域名的控制。

a) 通过填充 URL 模板“{schema}://{domain}/{tokenPath}”构建 URL，其中：

- schema 根据 tokenType，设置为 http 或者 https；
- domain 字段设置为待验证的域名或 IP 地址；
- tokenPath 字段设置为获取挑战 token 的网址；

b) 验证生成的 URL 是否格式正确。

c) 使用 HTTP GET 请求 URL，此请求应发送到 HTTP 服务器上的 TCP 80 或 443 端口。

d) 验证响应正文是否为格式正确的密钥授权。服务端应忽略主体末尾的空白字符。

e) 验证 HTTP 服务器提供的“token”是否与服务端存储的“token”相匹配。

访问 URL 时，服务端应遵循重定向。例如：客户端可能会使用重定向，以便可由中央证书管理服务端提供响应。有关重定向的安全注意事项，请参阅第 C.3 节。

若以上所有验证都成功，则验证成功。若请求失败，或者正文没有通过这些检查，则它就失败了。

一旦挑战完成，即挑战的“status”字段为值“valid”或“invalid”，客户端应取消为该挑战提供的资源。

由于 Token 同时出现在 ACME 服务端发送的请求及响应的密钥授权中，可能存在直接将 Token 从请求复制到响应的潜在客户实现机制。客户端应避免此种行为以防止引发跨站脚本（XSS）漏洞，而应针对每个挑战进行明确配置。若客户端仍采用从请求复制 Token 至响应的机制，则应验证请求中的 Token 是否符合相关语法规则（例如：仅包含 Base64url 字符集中的字符）。

7.6.3.2 DNS 挑战

当被验证的主体标识是域名时，客户端可通过提供包含特定验证域名的指定值的 TXT 资源记录来证明对该域名的控制。

- a) type（必需，字符串）：字符串 “dns-01”；
- b) token（必需，字符串）：唯一标识挑战的随机值。这个值应至少有 128 位的熵。它不应包含 Base64URL 字母表之外的任何字符，并且不应包含 Base64 填充字符（“=”）。
- c) tokenType：域名解析类型，只能是“TXT”或“CNAME”，不能为其他值；
- d) tokenPath：域名解析 DNS 记录

```
{
  "type": "dns-01",
  "url": "https://example.com/acme/chall/Rg5dV14Gh1Q",
  "status": "pending",
  "token": "evaGxfADs6pSRb2LAv9IZf17Dt3juxGJ-PCt92wr-oA"
  "tokenType": "TXT"
  "tokenPath": "W725G084H61CF8A5C8"
}
```

客户端通过根据挑战中提供的“token”值、“tokenType”值和“tokenPath”值设置 DNS 记录来完成此挑战。

提供给 DNS 记录包含此摘要的 Base64url 编码。客户端通过在被验证的域名前加上标签为“tokenPath”值前加下划线“_”为 DNS 记录来构建验证域名授权，然后提供一个“tokenType”的记录类型。例如：若要验证的域名是“www.example.org”，则客户端将提供以下 DNS 记录：

```
_W725G084H61CF8A5C8.www.example.org. 300 IN TXT "evaGxfADs6p...92wr-oA"
```

客户端使用空对象({})进行响应，以确认服务端可验证挑战。

```
POST /acme/chall/Rg5dV14Gh1Q HTTP/1.1
Host: example.com
Content-Type: application/jose+json
{
  "protected": Base64url({
    "alg": "SM2",
    "kid": "https://example.com/acme/acct/evOfKhNU60wg",
    "nonce": "SS2sS11PtspvFZ08kNtzKd",
    "url": "https://example.com/acme/chall/Rg5dV14Gh1Q"
  }),
  "payload": Base64url({}),
  "signature": "Q1bURgJoEslbD1c5...3pYdSMLio57mQNN4"
}
```

收到响应后，服务端根据挑战“token”值、“tokenType”值和“tokenPath”值来获取 DNS 记录验证授权。

为验证 DNS 挑战，服务端执行以下步骤：

- a) 查询指定域名记录类型的 DNS 记录；
- b) 验证其中一个 DNS 记录的内容是否与哈希值匹配。

若以上所有验证都成功，则验证成功。若未找到 DNS 记录，或者 DNS 记录和响应负载未通过这些检查，则验证失败。

一旦挑战完成，即挑战的“status”字段值为“valid”或“invalid”，客户端应取消为该挑战提供的资源记录。

7.6.3.3 Email 挑战

此方法用于电子邮件证书自动化管理。通过 Email 验证，ACME 事务中的客户端通过证明它可接收该邮箱中的验证码来证明邮件客户端能对此邮箱的控制。ACME 服务端要求客户端能提交特定验证码。

type（必需，字符串）：字符串“email-http-00”；

token（必需，字符串）：唯一标识挑战的随机值。这个值应至少有 128 位的熵。它不应包含 Base64URL 字母表之外的任何字符，并且不应包含 Base64 填充字符（“=”）。

```
{
  "type": "email-http-00",
  "url": "https://example.com/acme/chall/ABprV_B7yEyA4f",
  "from": "acme-email-challenge@example.com",
  "token": "DGyRejmCefe7v4NfDGDkFA"
}
```

挑战对象的“token”字段（来自“challenges”数组）是响应挑战的第二段 token，命名为 token-part2，ACME 服务端还会生成另一个 token，命名为 token-part1，这个 token 通过上述电子邮件账户发送到用户邮箱中。这封“挑战”邮件的主题为“ACME: <token-part1>”，其中 <token-part1> 是 Base64url 编码的令牌的形式。ACME 服务器应为每个 S/MIME 签发请求（授权请求）生成一个新 token，token-part1 应包含至少 128 位熵。

“挑战”邮件应具有以下结构：

1. 主题头字段的语法如下：“ACME: <token-part1>”，其中前缀“ACME:”后有一个空格，然后是 <token-part1>，这是 ACME 令牌的 Base64url 编码的第一部分，应解码后至少为 128 位。由于有 78 位字节行长限制，主题行可折叠，因此 <token-part1> 中的空格（若有）应被忽略。主题字段的编码允许使用“UTF-8”和“US-ASCII”字符集；其他字符集不能使用。US-ASCII 字符集可使用。

2. “From”头字段的电子邮件地址应与挑战对象的“from”字段中指定的电子邮件地址相同。

3. “To”头字段的电子邮件地址应是请求生成 S/MIME 邮件证书的用户的电子邮件地址。

4. 消息可包含 Reply-To 和/或 CC 头字段。

5. 消息应包含值为“auto-generated”的“自动提交”头字段。为帮助调试（对于某些实现，为简化自动处理），自动提交头字段应包括“type=acme”参数。它可包括“自动提交”头字段的语法所允许的其他可选参数。

6. 为证明挑战信息的真实性，它应使用域名密钥识别邮件（DKIM）进行签名或者 S/MIME 签名。

- 若使用 DKIM 签名，则生成的 DKIM-Signature 头字段应包含以下内容的“h=”标签，至少是发件人、发件人、回复、收件人、抄送、主题、日期、回复中、参考、Message-ID、Auto-Submitted、Content-Type 和 Content-Transfer-Encoding 头字段。DKIM-Signature 头字段的“h=”标记还应包括 resent-date、resent-from、resent-to、resent-cc、list-id、list-help、list-unsubscribe、List-Subscribe、List-Post、

List-Owner、List-Archive 和 List-Unsubscribe-Post 头字段。DKIM-Signature 头字段的“d=”标记中的域名应与“挑战”邮件的发件人标题字段。

- 若使用 S/MIME 签名，则与签署者对应的证书应具有 rfc822Name subjectAltName 扩展名的值等于“挑战”邮件的“发件人”标题字段电子邮件地址。

7. 挑战消息的正文不用于自动处理，因此它可是任何媒体类型。通常，它是 text/plain 或 text/html，包含人类可读的信息。若使用 S/MIME 签名来证明挑战邮件的真实性，应是 multipart/signed 或“application/pkcs7-mime;smime-type=signed-data;”媒体类型。无论哪种方式，它应使用 S/MIME 头保护。

符合此规范的电子邮件客户端，可检测特定的“挑战”邮件，未通过上述验证应忽略此“挑战”邮件。为帮助调试，此类失败的验证应被记录下来。

以下为 ACME ‘挑战’邮件的示例（为简明起见，该示例省略了与 DKIM 和 S/MIME 签名相关的头字段）。

```
Auto-Submitted: auto-generated; type=acme
Date: Sat, 5 Dec 2020 10:08:55 +0100
Message-ID: <A2299BB.FF7788@example.com>
From: acme-email-challenge@example.com
To: alexey@example.org
Subject: ACME: LgYemJLy3F1LDkiJrdIGbEzyFJy0yf6vBdyZ1TG3sME=
Content-Type: text/plain
MIME-Version: 1.0
```

这是针对电子邮件地址“alexey@example.org”自动生成的 ACME 挑战。若您没有请求为此电子邮件地址生成 S/MIME 证书，请忽略此邮件。若您确实请求了，您的电子邮件客户端可能能够自动处理此请求，或者您可能需要将主题 ACME：后面的验证码粘贴到外部程序中

ACME 邮件客户端串联“token-part1”（通过电子邮件接收）和“token-part2”（通过 HTTPS 接收）授权来完成挑战。

ACME 邮件客户端从邮件中提取 token-part1，从 HTTPS 响应中获取 token-part2。串联 token-part1、token-part2，格式为：response = token-part1 + token-part2

```
POST /acme/chall/prV_B7yEyA4 HTTP/1.1
Host: example.com
Content-Type: application/jose+json
{
  "protected": Base64url({
    "alg": "SM2",
    "kid": "https://example.com/acme/acct/evOfKhNU60wg",
    "nonce": "UQI1PoRi5OuXzXuX7V7wL0",
    "url": "https://example.com/acme/chall/prV_B7yEyA4"
  }),
```

```

    "payload": Base64url({
      "response": "LoqXcYV8q50NbJQxbmR7SCTNo3tiAXDfowyjxAjEuX0"
    }),
    "signature": "Q1bURgJoEslbD1c5...3pYdSMLio57mQNN4"
  }

```

ACME 服务端收到响应后，根据发出的“token-part1”和“token-part2”构建并存储密钥授权，并比较是否与收到的“response”一致，一致则表示验证成功。若不一致，那就是验证失败。

一旦挑战完成，即挑战的“status”字段为值“valid”或“invalid”，客户端应取消为该挑战提供的资源。

7.6.3.4 DNS-Email 挑战

此方法用于电子邮件证书自动化管理。为应对待验证邮箱无法收到 ACME 服务端发送的“token-part1”邮件，可选此方式完成挑战。

当被验证的标识符是 email 时，客户端可通过提供包含待验证邮箱的域名的指定值的 TXT 资源记录来证明对该邮箱的控制。

type（必需，字符串）：字符串“email-dns-01”；
 token（必需，字符串）：唯一标识挑战的随机值。这个值应至少有 128 位的熵。它不应包含 Base64URL 字母表之外的任何字符，并且不应包含 Base64 填充字符（“=”）。

```

{
  "type": "email-dns-01",
  "url": "https://example.com/acme/chall/Rg5dV14Gh1Q",
  "status": "pending",
  "token": "evaGxfADs6pSRb2LAv9IZf17Dt3juxGJ-PCt92wr-oA"
}

```

客户端通过根据挑战中提供的“token”值来完成此挑战。

提供给 DNS 的记录包含此摘要的 Base64url 编码。客户端通过在被验证的域名前加上标签“_acme-email-challenge”来构建邮箱验证，然后提供一个 TXT 记录，其值就是“token”值。例如：若要验证的邮箱是“alexey@example.org”，则客户端将提供以下 DNS 记录，其中@前的邮箱用户名加“.”格式显构成域名 RR：

```
_acme-email-challenge.alexey.example.org. 300 IN TXT "gfj9Xq...Rg85nM"
```

客户端使用空对象({})进行响应，以确认服务端可验证挑战。

```

POST /acme/chall/Rg5dV14Gh1Q HTTP/1.1
Host: example.com
Content-Type: application/jose+json
{
  "protected": Base64url({
    "alg": "SM2",
    "kid": "https://example.com/acme/acct/evOfKhNU60wg",

```

```

    "nonce": "SS2sS11PtspvFZ08kNtzKd",
    "url": "https://example.com/acme/chall/Rg5dV14Gh1Q"
  }},
  "payload": Base64url({}),
  "signature": "Q1bURgJoEslbD1c5...3pYdSMLio57mQNN4"
}

```

收到响应后，服务端根据挑战“token”值来验证授权。

c) 为验证 DNS 挑战，服务端执行以下步骤：查询验证域名的 TXT 记录；

d) 验证其中一个 TXT 记录的内容是否与挑战“token”值匹配。

若以上所有验证都成功，则验证成功。若未找到 DNS 记录，或者 DNS 记录和响应负载未通过这些检查，则验证失败。

一旦挑战完成，即挑战的“status”字段值为“valid”或“invalid”，客户端应取消为该挑战提供的资源记录。

7.6.3.5 扩展主体标识类型

在许多实际应用场景下，需要证书的设备或应用并不具备一个有效的域名或电子邮件地址，而可能需要采用其它类型的主体标识来标识自身。尤其是在入驻式场景下，需要采用更多类型的主体标识来区分不同的设备、用户或终端。

因而，在 dns 类型和 email 主体标识的基础上，本规范还推荐采用已签发的证书、密钥对、一次性口令、sip 号码、phone、唯一标识符（如 id、VIN 码、IMEI 码）、permanent-identifier 及 hardware-module 等内容作为主体标识。

验证不同类型主体标识时，所采用的挑战类型并不相同。服务端应当根据提交申请的主体标识类型，为客户端返回其支持类型的挑战对象。同时为区分签发不同的数字证书类型，可在证书请求文件（CSR）中包含 KeyUsage（DigitalSignature）和 ExtendedKeyUsage（CodeSigning）设置为正确的值，以便 CA 能够识别该请求是签发何种类型证书，如代码签名证书的增强密钥用法应设置为 OID “1.3.6.1.5.5.7.3.3”，文档签名证书应设置为 OID “1.3.6.1.5.5.7.3.36”。CN 应设置为用户帐户的预期注册域。

7.6.3.6 扩展挑战类型

7.6.3.6.1 一次性口令（OTP）挑战

一次性口令技术种类繁多，不同实现方式略有不同。质询响应输入到提供的 URL 中，为使用此种质询类型的用户提供了灵活性，以满足其解决方案的特定需求。以下两种 OTP 解决方案通过工具或应用程序提供质询响应，无需用户与服务器交互生成质询所需的信息。采用这种方式运行的两种解决方案包括 SecureID 和 Duo Security。若需要质询来生成 URL 中提供的响应，则令牌可提供质询。

type（必需，字符串）：字符串“otp-01”。

token（必需，字符串）：用于唯一标识此质询的随机值。OTP 类型和输入因技术而异。token 值将与预签发的 OTP 凭证的预期类型匹配。用户将能够在提供的 URL 中提供此质询的响应。响应不应包含任何 Base64url 字母表以外的字符，也不应包含 Base64 填充字符（“=”）。

```

{
  "type": "otp-01",
  "url": "https://example.com/acme/chall/WrV_H87EyD3",

```

```

    "status": "pending",
    "token": "challenge"
  }

```

7.6.3.6.2 基于 HMAC 的一次性口令 (HOTP)

HOTP([RFC4226]) 描述了一种基于时间的密码值生成算法。

type (必需, 字符串): 字符串 “hotp-01”。

token (必需, 字符串): HOTP 值。应为 6 位数字表示。

```

{
  "type": "hotp-01",
  "url": "https://example.com/acme/chall/WrV_H87EyD3",
  "status": "pending",
  "token": "123456"
}

```

7.6.3.6.3 基于时间的一次性口令 (TOTP)

TOTP([RFC6238]) 描述了一种基于时间的密码值生成算法, 它是 HOTP 的扩展。

type (必需, 字符串): 字符串 “totp-01”。

token (必需, 字符串): TOTP 值。应为 6 位数字表示。

```

{
  "type": "totp-01",
  "url": "https://example.com/acme/chall/WrV_H87EyD3",
  "status": "pending",
  "token": "123456"
}

```

7.6.3.6.4 公钥密码挑战

基于已签发的公钥证书及私钥, 用于识别用户身份。若某个特定角色的用户或个人需要更高的保证级别, 则可使用指定流程可接受的标识符以及根据所需保证级别正确存储的私钥进行身份鉴证。公钥密码学标准#11 (PKCS#11) 软件或硬件令牌都是可行的选择。此模型假设可能有多个授权用户持有不同的证书, 可用于授权或预身份鉴别质询。因此, 用户首先提供数字签名, 以便帐户管理员可确定是否使用了其中一个可接受的证书对令牌进行数字签名。

type (必需, 字符串): 字符串 “cert-01”。

token (必需, 字符串): 用于唯一标识质询的随机值。证书身份鉴别质询的 token 包含一个值, 接收者可使用其私钥进行数字签名并发布到提供的 URL。然后, ACME 服务器使用数字签名的内容来验证质询是否使用授权凭证(针对此质询类型签发并授权的证书)进行了签名。token 不应包含 Base64url 字母表以外的任何字符, 也不应包含 Base64 填充字符 (“=”)。

```

{
  "type": "cert-01",
  "url": "https://example.com/acme/chall/WrV_H87EyD3",
  "status": "pending",
  "token": "Some challenge to digitally sign"
}

```

7.6.3.6.5 WebAuthn 或 公钥/私钥对

W3C 的 WebAuthn/Passkeys 使用针对特定服务签发的原始公钥/私钥对。若选择 WebAuthn 或公钥/私钥对 (PPKP) 作为质询类型, 则应在使用此质询类型之前设立帐户和凭证。WebAuthn 或公钥/私钥对凭证将特定于证书管理帐户, 并由客户端创建, 然后像常规 WebAuthn 凭证注册一样在服务中注册。与常规 WebAuthn 和公钥/私钥对一样, 令牌或质询也经过数字签名, 以证明私钥的所有权。

密钥应预先向 ACME 服务注册。

type (必需, 字符串): 字符串 “ppkp-01”。

token (必需, 字符串): 用于唯一标识此质询的随机值。此质询的操作方式与证书质询大致相同, 操作大致相同。用户将能够通过此质询提供的 URL 提供响应。响应不应包含任何 Base64url 字母表以外的字符, 也不应包含 Base64 填充字符 (“=”)。

```
{
  "type": "ppkp-01",
  "url": "https://example.com/acme/chall/WrV_H87EyD3",
  "status": "pending",
  "token": "Some challenge to sign"
}
```

7.6.3.6.6 SSO 挑战

利用 Single Sign-on 单点登录技术, 可用于验证某些特定类型的身份。服务端可基于可信的 IDP (Identity Provider), 使用 IDP 验明申请者的身份, 并获取该申请者在 IDP 中对应的身份信息。

在这种解决方案中, 提供一种 sso 类型的挑战, 表明基于 SSO 技术可完成相应的身份验证。不同的 IDP 可能支持不同类型的身份信息, 而这种身份信息可用作申请者所申请证书的身份——即, 订单中主体标识的类型。服务端可自定义主体标识的类型, 本文推荐可使用邮箱 email、sip 号码、phone、唯一标识符 (如 id、VIN 码、IMEI 码) 等作为主体标识。

SSO 挑战, 其对应的主体标识为: e-mail/sip/phone/id/vin/imei。SSO 挑战的类型为 sso-01。

SSO 挑战对象示例

```
{
  "type": "sso-01",
  "url": "https://example.com/acme/chall/prV_B7yEyA4",
  "status": "pending",
  "sso-url": "https://example.org/acme/start-sso/",
  "sso-provider": "https://identity-provider/sso/"
}
```

ACME 客户端通过 sso-url 开始进行 SSO 挑战。可同时存在多个 sso-01 挑战, 客户端根据 sso-provider 选择对应的挑战。

sso-url 允许客户端发起 SSO 流程。建议在用户浏览器中执行此流程。服务器应接受 sso-url 的 GET 请求。在收到此类请求时, 应将客户端重定向到适当的 SSO 提供方, 以便用户可得到适当的身份验证。当 SSO 提供方成功完成用户身份验证时, 它将客户端重定向回 CA, 重定向到 CA 在启动 SSO 流程时提供的位置, 并包含相应的证明。然后, 服务器应根据 SAML 或 Open ID 规范指定方式, 验证提供方证明是否有效。此外, 服务器应验证 IDP 对于挑战验证的身份信息的证明是否有效。

针对不同类型的申请, 如邮箱 email、sip 号码、id、phone、设备唯一标识 (VIN 码、IMEI 码) 等字段的主体标识, ACME 在验证提供方身份的时候, 应根据主体标识获取用户身份, 验证其是否与申请证书

的字段是否一致。同时，在申请证书的CSR中，在指定的字段中标识出用户身份（如subject的CommonName/CN或者subjectAltName/SAN），而不包括其它字段。

7.6.3.6.7 设备身份挑战

设备身份挑战添加主体标识类型：（1）permanent-identifier；（2）hardware-module。对应的挑战类型为device-attest-01。

设备挑战对象示例：

```
{
  "type": "device-attest-01",
  "url": "https://example.com/acme/chall/Rg5dV14Gh1Q",
  "token": "evaGxfADs6pSRb2LAv9IZf17Dt3juxGJ-PCt92wr-oA"
}
```

1) 客户端执行挑战。客户端通过从服务端挑战中获取的“token”值和客户端的账户密钥构建密钥授权来完成挑战。客户端使用密钥授权作为挑战生成WebAuthn证明对象。

2) 客户端响应挑战。

客户端响应挑战的示例如下：

```
POST /acme/chall/Rg5dV14Gh1Q
Host: example.com
Content-Type: application/jose+json
{
  "protected": Base64url({
    "alg": "SM2",
    "kid": "https://example.com/acme/acct/evOfKhNU60wg",
    "nonce": "SS2sS11PtspvFZ08kNtzKd",
    "url": "https://example.com/acme/chall/Rg5dV14Gh1Q"
  }),
  "payload": Base64url({
    "attObj": Base64url(/* WebAuthn attestation object */),
  }),
  "signature": "Q1bURgJoEs1bD1c5...3pYdSMLio57mQNN4"
}
```

3) 服务端校验挑战

服务端按照如下流程验证挑战：

- a. 执行WebAuthn验证流程；
- b. 验证_attToBeSigned_传送的密钥授权是否与服务器存储的密钥授权相匹配。

7.6.3.6.8 公钥 PK 挑战

公钥 PK 挑战添加三种类型主体标识：pk、csr 和 selfsign-cert。对应的挑战类型为 pk-01。公钥 pk 挑战可与支持执行公钥验证的身份提供商 CMS（例如支持 aPAKE/opaque 协议）交互。其核心是验证证书请求者对最终证书公钥所对应私钥的控制权，以杜绝在完成订单（finalize）阶段篡改证书申请文件公钥的可能性。

该流程与 SSO 挑战相似，不同点在于（1）认证申请者在 CMS 上的身份验证是基于验证公钥身份，

例如可采用更安全的 aPAKE/opaque 协议；（2）CMS 中存储了用户的身份公钥，因而，能够校验最终提交的 CSR 中的公钥是否真实代表用户、未被篡改。

新添加的三种主体标识类型中，pk 表示使用一个特定的公钥申请证书，未指定身份；selfsign-cert 和 csr 除了指定申请证书的公钥外，还指定了新申请证书的身份。这里支持多种类型的身份，包括 dns、ip、email、id/uuid、phone、sip 号码、设备标识符（如 VIN 码、IMEI 码）。

pk-01 挑战对象示例：

```
{
  "type": "pk-01",
  "url": "https://example.com/acme/chall/prV_B7yEyA4",
  "status": "pending",
  "pk_url": "https://example.org/acme/start-pk/",
  "pk_provider": "https://identity-provider/pk/"
}
```

ACME 客户端通过 pk-url 开始进行公钥验证挑战。可同时存在多个 pk-01 挑战，客户端根据 pk_provider 选择对应的挑战。

服务器应接受 pk_url 的 GET 请求。在收到此类请求时，应将客户端重定向到适当的 IDP 身份提供方，以使用户可得到适当的身份验证，在 IDP 中保存有用用户信息和用户公钥信息。公钥验证（即验证公钥对应私钥的控制权）支持多个方式：1）挑战公钥签名并验签；2）Opaque/AKE；3）NIZK/DLEQ proof。客户端在公钥验证时应指定 IDP 提供的一种验证协议。当 IDP 身份提供方成功完成用户公钥验证后，它将客户端重定向回 ACME 服务端，重定向到客户端在启动 pk 流程时提供的位置，并包含相应的证明。用户的证明信息包含用户注册公钥、用户信息和用户公钥对应私钥的密文。对于使用 pk 申请证书的订单，应检验 pk 与用户公钥是否一致；对于使用 selfsign-cert (csr) 申请证书的订单，还应检验订单中包含的身份信息是否与用户信息一致。

在挑战完成后，finalize 阶段还应对最终提交的 CSR 订单检验。校验其中的公钥与身份是否与申请的订单一致。

证书撤销

要请求撤销证书，客户端向 ACME 服务端的 revokeCertURL 发送 POST 请求。POST 的主体是一个 JWS 对象，其 JSON 负载包含要撤销的证书：

- a) certificate（必需，字符串）：要撤销的证书，采用 DER 格式的 Base64url 编码版本；（注：由于该字段使用 Base64url，不包含 PEM 头，所以与 PEM 不同。）
- b) reason(optional, int)：撤销原因。若未设置此字段，服务端在生成 OCSP 响应和 CRL 时应忽略 reasonCode CRL 条目扩展。服务端可禁止用户使用 reasonCodes 的子集。若请求包含不允许的 reasonCode，则服务端应使用错误类型“URN:ietf:params:acme:error:badRevocationReason”拒绝它，错误信息应指出允许哪些原因代码。

撤销请求不同于其他 ACME 请求，因为它们可使用帐户密钥对或证书中的密钥对进行签名。使用帐户密钥对其签名的示例：

```
POST /acme/revoke-cert HTTP/1.1
Host: example.com
Content-Type: application/jose+json
{
```

```

    "protected": Base64url({
      "alg": "SM2",
      "kid": "https://example.com/acme/acct/evOfKhNU60wg",
      "nonce": "JHb54aT_KTXBWQ0zGYkt9A",
      "url": "https://example.com/acme/revoke-cert"
    }),
    "payload": Base64url({
      "certificate": "MIIEDTCCAvegAwIBAgIRAP8...",
      "reason": 4
    }),
    "signature": "Q1bURgJoEslbD1c5...3pYdSMLio57mQNN4"
  }

```

使用证书密钥对其签名的示例：

POST /acme/revoke-cert HTTP/1.1

Host: example.com

Content-Type: application/jose+json

```

{
  "protected": Base64url({
    "alg": "SM2",
    "jwk": /* certificate's public key */,
    "nonce": "JHb54aT_KTXBWQ0zGYkt9A",
    "url": "https://example.com/acme/revoke-cert"
  }),
  "payload": Base64url({
    "certificate": "MIIEDTCCAvegAwIBAgIRAP8...",
    "reason": 1
  }),
  "signature": "Q1bURgJoEslbD1c5...3pYdSMLio57mQNN4"
}

```

在撤销证书之前，服务端应验证用于签署请求的密钥是否有权撤销证书。服务端应至少考虑以下授权给给定证书的帐户：

- a) 签发证书的帐户；
- b) 拥有所有主体标识授权的帐户。

若使用与证书中公钥相对应的私钥签名，服务端应认为撤销请求有效。

若撤销成功，服务端会返回状态码 200(OK)。若撤销失败，服务端会返回一个错误。例如：若证书已被撤销，服务端将返回状态代码为 400（错误请求）的错误响应，错误类型为“urn:ietf:params:acme:error:alreadyRevoked”。

HTTP/1.1 200 OK

Replay-Nonce: IXVHDyxIRGcTEOVsblhPzw

Content-Length: 0

```

Link: <https://example.com/acme/directory>;rel="index"
- - 或 - -
HTTP/1.1 403 Forbidden
Replay-Nonce: lXfyFzi6238tfPQRwgfmPU
Content-Type: application/problem+json
Content-Language: en
Link: <https://example.com/acme/directory>;rel="index"
{
  "type": "urn:ietf:params:acme:error:unauthorized",
  "detail": "No authorization provided for name example.org"
}

```

7.8 续订信息

续订信息("renewalInfo")资源允许客户端查询 ACME 服务器以了解关于何时续订证书的建议，并允许客户端在完成续订（或以其他乐意的方式更换证书）时通知服务端。

7.8.1 获取续订信息

要请求证书的建议续订信息，客户端将 GET 请求发送到服务端的续订信息 URL 下的路径。

路径是相关证书的唯一标识符，由 Base64url 编码的授权密钥标识符(AKI)的 keyIdentifier 字段、句点字符“.”以及 DER 编码的序列号字段的 Base64url 编码（不带标记和长度字节）。所有尾随的“”空字符=应从唯一标识符的两个部分中剥离。

例如，某张用户证书的授权密钥标识符为：69885B6B87464041E1B37B847BA0AE2CDE01C8D4，证书序列号为：0087654321，前者 Base64url 编码为：aYhba4dGQEHhs3uEe6CuLN4ByNQ，后者 Base64url 编码为 AIdlQyE：则客户端将按如下方式发出续订请求：

```

GET /renewal-info/aYhba4dGQEHhs3uEe6CuLN4ByNQ.AIdlQyE HTTP/1.1
Host: example.com
Accept: application/json

```

7.8.2 RenewalInfo 对象

ACME RenewalInfo 对象的结构如下：

suggestedWindow（对象，必需）：一个 JSON 对象，包含两个键“start”和“end”，其值为按照 RFC 3339 规定的格式编码的时间戳，它绑定了 CA 建议续订证书的时间窗口。

explanationURL（字符串，可选）：一个指向解释建议的续订窗口页面的 URL。例如，它可是解释 CA 负载均衡策略的页面，也可是记录哪些证书受到大规模撤销事件影响的页面。符合的客户端应将此 URL 提供给其操作员（若有）。

```

HTTP/1.1 200 OK
Content-Type: application/json
Retry-After: 21600
{
  "suggestedWindow": {
    "start": "2021-01-03T00:00:00Z",

```

```

    "end": "2021-01-07T00:00:00Z"
  },
  "explanationURL": "https://example.com/docs/ari"
}

```

服务器应包括一个 Retry-After 头参数，指示 ACME 服务器建议的轮询间隔。符合规定的客户端应在重试后周期过去后再次查询续订信息 URL，因为服务器可能会提供不同的 suggestedWindow。

符合的客户应在建议的续订窗口内，选择时间尝试续订。建议使用以下算法选择续订时间：

- a) 在建议的窗口中选择统一随机时间。
- b) 若选择了过去的时间，立即尝试续订。
- c) 否则，若客户端可安排自己在选定的时间尝试续订，执行此操作。
- d) 否则，若选择的时间在客户端下一次正常唤醒时间之前，立即尝试续订。
- e) 否则，睡眠到下一个正常唤醒时间，重新检查 ARI，然后返回步骤 a)。

在所有情况下，尝试续订都取决于客户端现有的错误回退和重试间隔。

客户端可能会发现他们需要增加运行频率，以更频繁地检查 ARI。这些客户端应存储有关故障的信息，以便增加其运行频率不会导致在重试失败的情况下没有适当回退。存储的典型信息应包括：给定订单的失败数量（由订单上的名称集定义）和最近失败的时间。

若客户端没有收到响应或响应格式错误（例如：结束时间戳在开始时间戳之前），它应自己确定何时续订证书，并可能以适当的指数回退行为重试续订信息请求。

7.8.3 订单对象的扩展

为传达有关证书请求为以前证书的续订的信息，将向 Order 对象添加一个新字段：

replaces（字符串，可选）：唯一字符串，识别此订单是先前颁发的证书续期。此唯一标识符的构造方式与请求路径相同，用于 GET 请求，如第 7.8.1 节所述。

若是续期订单，则客户端应在 newOrder 请求中包含此字段。但若 ACME 服务端尚未在其目录中通告资源来表明支持此协议，则不应包含此字段。

```

POST /new-order HTTP/1.1
Host: example.com
Content-Type: application/jose+json

{
  "protected": Base64url({
    "alg": "SM2",
    "kid": "https://acme.example.com/acct/evOfKhNU60wg",
    "nonce": "5XJlL3lEkMG7tR6pA00c1A",
    "url": "https://example.com/new-order"
  }),
  "payload": Base64url({
    "identifiers": [
      { "type": "dns", "value": "example.com" }
    ],
    "replaces": "aYhba4dGQEHhs3uEe6CuLN4ByNQ.AId1QyE"
  })
}

```

```

    }),
    "signature": "H6ZXtGjTZyUnPeKn...wEA4Tk1Bdh3e454g"
  }

```

ACME 服务端应检查已识别的证书和 newOrder 请求是否对应于同一 ACME 帐户，它们是否至少共享一个标识符，以及已识别的证书尚未标记为“无效”的其他订单替换。超出此范围的相应检查（例如要求精确标识符匹配）取决于服务端策略。若这些检查中的任何一个失败，则服务端**应**拒绝 newOrder 请求。若服务器拒绝请求，因为已识别的证书已标记为已替换，则**应**返回 HTTP 409（冲突），其中包含类型为“alreadyReplaced”的问题文档（见第 7.4 节）。

若服务器接受带有“replaces”字段的 newOrder 请求，则它**应**在响应和相应 Order 对象的后续请求中反映该字段。

此替换信息可能有多种用途，包括但不限于：

- 授予在已识别的前身证书的建议续订窗口内到达的 newOrder 请求更高的优先级，或者允许它们绕过速率限制（若服务器的策略使用）；
- 跟踪受合规事件影响的证书的更换情况，以便在更换后立即撤销；和
- 将根据同一合同颁发的证书与外部帐户绑定确定的实体绑定在一起。

附录 A
(规范性)
编码及元数据规范

A.1 媒体类型: application/pem-certificate-chain

此类文件包含一个或多个使用 PEM 文本编码的证书。此文件中证书的文本编码应使用严格的编码并且不应包含解释性文本。这种格式的 ABNF 如下,其中“stricttextualmsg”的含义见 RFC 7468 第 3 节:

certchain = stricttextualmsg * (stricttextualmsg)
为提供与 TLS 的简单互操作,第一个证书应是用户证书。每个后续证书都应直接证明它前面的证书。因为证书验证要求独立分发受信任的根证书,所以表示信任锚的根证书可从链中省略。
以下内容已在“Media Types”注册表中注册:
类型名称: application;
子类型名称: pem-certificate-chain;
必需参数: 无;
可选参数: 无;
编码注意事项: 7bit;
安全考虑: 携带证书及其关联的证书链。此媒体类型不包含活动内容;
互操作性注意事项: 无;
已发布规范: 本文件
使用该媒体类型的应用: ACME 客户端和服务端、HTTP 服务器、其他应配置证书链的应用。

A.2 HTTP 挑战的已知 URI

以下值已在“Well-Known URIs”注册表中注册:
URI 后缀: acme-challenge 或 pki-validation;
变更控制者: IETF;
规范文档: 第 7.6.3.1 节;
相关信息: 不适用。

A.3 Replay-Nonce 重放随机数 HTTP 头

见表 A.1, 以下值已在“Message Headers”注册表中注册:

表 A.1 已注册的 HTTP 头

头字段名称	协议	状态	参考
Replay-Nonce	http	standard	第6.7.1节

A.4 “url” JWS 头参数

- “JSON Web Signature and Encryption Header Parameters”注册表中已注册以下值:
- a) 头参数名称: “url”;
 - b) 头部参数说明: URL;
 - c) 头参数使用位置: JWE、JWS;
 - d) 变更负责人: IESG;

e) 规范文档：第 6.6 节。

A.5 “nonce” JWS 头参数

“JSON Web Signature and Encryption Header Parameters”注册表中已注册以下值：

- a) 头参数名称：“nonce”；
- b) 头部参数说明：Nonce；
- c) 头参数使用位置：JWE、JWS；
- d) 变更负责人：IESG；
- e) 规范文档：第 6.7.1 节。

A.6 ACME 的 URN 子命名空间(urn:ietf:params:acme)

注册表名称：acme；
规范：本文件；
存储库：HTTP://www.iana.org/assignments/acme；
指标值：无需转换。

A.7 新注册表

IANA 创建了以下注册表：

- a) ACME 帐户对象字段（第 A.7.1 节）；
- b) ACME 订单对象字段（第 A.7.2 节）；
- c) ACME 授权对象字段（第 A.7.3 节）；
- d) ACME 错误类型（第 A.7.4 节）；
- e) ACME 资源类型（第 A.7.6 节）；
- f) ACME 目录元数据字段（第 A.7.7 节）；
- g) ACME 主体标识类型（第 A.7.8 节）；
- h) ACME 验证方法（第 A.7.9 节）。

A.7.1 账户对象中的字段

“ACME Account Object Fields”注册表列出了为在 ACME 帐户对象中使用而定义的字段名称。标记为“configurable”的字段可能包含在 newAccount 请求中，见表。

模板：

- a) 字段名：在 JSON 对象中用作字段名的字符串；
- b) 字段类型：要提供的值的类型，如：string, boolean, array of string；
- c) 请求：值“none”或请求对象中允许该字段的请求类型列表，取自以下值：
 - 1) “new”：对“newAccount”URL 的请求；
 - 2) “account”：对帐户 URL 的请求。
- d) 参考：定义该字段的地方；
- e) 初始内容：9.1.2 节定义了字段和描述。

表 A.2 ACME 帐户对象字段

字段名称	字段类型	请求	参考
status	string	new, account	本文件
contact	Array of string	new, account	本文件

externalAccountBinding	object	new	本文件
termsOfServiceAgreed	boolean	new	本文件
onlyReturnExisting	boolean	new	本文件
orders	string	new	本文件

A. 7. 2 订单对象中的字段

“ACME Order Object Fields”注册表列出了为在 ACME 订单对象中使用而定义的字段名称。标记为“可配置”的字段可能包含在 newOrder 请求中，见表 A. 3。

模板：

- a) 字段名：在 JSON 对象中用作字段名的字符串；
- b) 字段类型：要提供的值的类型，如 string, boolean, array of string；
- c) 可配置：布尔值，指示服务端是否应接受客户端提供的值；
- d) 参考：定义该字段的地方；
- e) 初始内容：第 7. 2. 3 节定义了字段和描述。

表 A. 3 ACME 订单对象字段

字段名	字段类型	可配置	参考
status	string	false	本文件
expires	string	false	本文件
identifiers	array of object	true	本文件
notBefore	string	true	本文件
notAfter	string	true	本文件
error	string	false	本文件
authorizations	array of string	false	本文件
finalize	string	false	本文件
certificate	string	false	本文件
certificateSign	string	false	本文件
certificateEncrypt	string	false	本文件
certificateSM2	string	false	本文件
replaces	string	true	本文件

A. 7. 3 授权对象中的字段

“ACME Authorization Object Fields”注册表列出了为在 ACME 授权对象中使用而定义的字段名称。标记为“可配置”的字段可能包含在 newAuthz 请求中，见表。

模板：

- a) 字段名：在 JSON 对象中用作字段名的字符串；
- b) 字段类型：要提供的值的类型，如 string, boolean, array of string；
- c) 可配置：布尔值，指示服务端是否应接受客户端提供的值；
- d) 参考：定义该字段的地方；
- e) 初始内容：第 7. 2. 4 节定义了字段和描述。

表 A. 4 ACME 授权对象字段

字段名	字段类型	可配置	参考
identifier	object	true	本文件
status	string	false	本文件
expires	string	false	本文件
challenges	array of object	false	本文件
wildcard	boolean	false	本文件

A. 7. 4 RenewalInfo 对象中的字段

“ACME RenewalInfo Object Fields”注册表列出了为在 ACME 续订信息对象中使用而定义的字段名称，见表。

模板：

- a) 字段名：在 JSON 对象中用作字段名的字符串；
- b) 字段类型：要提供的值的类型，如 string, object；
- c) 参考：定义该字段的地方；

表 A. 5 ACME 续订对象字段

字段名	字段类型	参考
suggestedWindow	object	本文件
explanationURL	string	本文件

A. 7. 5 错误类型

“ACME Error Types”注册表列出了在 ACME 错误报告的 “type” 字段中提供的 URN 值中使用的值。

模板：

- a) 类型：要包含在该错误的 URN 中的标签，位于 “urn:ietf:params:acme:error:” 之后；
- b) 描述：人类可读的错误描述；
- c) 参考：定义错误的地方；
- d) 第 6. 9 节表格中的类型和描述，设置 Reference 字段指向本规范。

A. 7. 6 资源类型

“ACME Resource Types”注册表列出了 ACME 服务端可能在其目录对象中列出的资源类型。

模板：

- a) 字段名：在目录对象中用作字段名的值；
- b) 资源类型：字段标注的资源类型；
- c) 参考：定义资源类型的地方。
- d) 初始内容，见表：

表 A. 6 ACME 资源类型

字段名	资源类型	参考
newNonce	NewNonce	本文件
newAccount	New account	本文件
newOrder	New order	本文件
newAuthz	New authorization	本文件

revokeCert	Revoke certificate	本文件
keyChange	Key change	本文件
meta	Metadata object	本文件

A. 7. 7 目录对象中“元”对象中的字段

“ACME Directory Metadata Fields”注册表列出了为在 ACME 目录对象的“meta”字段中包含的 JSON 对象中使用而定义的字段名称。

模板：

- a) 字段名：在 JSON 对象中用作字段名的字符串；
- b) 字段类型：要提供的值的类型，如 string, boolean, array of string；
- c) 参考：定义该字段的地方。
- d) 初始内容：第 7. 2. 1 节定义了字段和描述，见表。

表 A. 7 目录对象中“元”对象中的字段

字段名	字段类型	参考
termsOfService	string	本文件
website	string	本文件
caaIdentities	array of string	本文件
externalAccountRequired	boolean	本文件

A. 7. 8 主体标识类型

“ACME Identifier Types”注册表列出了可出现在 ACME 授权对象中的主体标识类型。

模板：

- a) 标签：要放入主体标识对象的“type”字段中的值；
- b) 参考：定义主体标识类型的地方。

初始内容，见表：

表 A. 8 ACME 授权对象主体标识类型

标签	参考
dns	本文件

A. 7. 9 验证方法

“ACME Validation Methods”注册表列出了 CA 可验证主体标识控制的方式的标识符。每个方法的条目应指定它是否对应于 ACME 挑战类型。“Identifier Type”字段应包含在“ACMEIdentifier Types”注册表的标签列中。

模板：

- a) 标签：此验证方法的标识符；
- b) 主体标识类型：此方法适用的主体标识类型；
- c) ACME：若验证方法对应于 ACME 挑战类型，则为“Y”；否则为“N”；
- d) 参考：定义验证方法的地方。

该注册表还可能包含保留条目（例如：为避免冲突）。此类条目应将“ACME”字段设置为“N”并

将“Identifier Type”设置为“RESERVED”。
初始内容，见表：

表 A.9 验证主体标识

标签	主体标识类型	ACME	参考
http-01	dns	Y	本文件
dns-01	dns	Y	本文件
tls-sni-01	RESERVED	N	本文件
tls-sni-02	RESERVED	N	本文件

在评估此注册表中的分配请求时，指定专家应确保所注册的方法具有清晰、可互操作的定义，并且不与现有验证方法重叠。也就是说，客户端和服务端不可能遵循同一组操作来完成两种不同的验证方法。

值“tls-sni-01”和“tls-sni-02”被保留，因为它们在某些情况下不安全而被删除。

验证方法不必为注册而与 ACME 兼容。例如：一个 CA 可能希望注册一个验证方法以支持它与 CAA[ACME-CAA]的 ACME 扩展一起使用。

附录 B

(资料性)

典型应用场景

ACME 的典型应用是为 Web 网站获取 TLS 证书 (HTTPS)。在这种情况下,网站服务器已经配置启用了 一个或多个域名,证书签发的过程是旨在验证此网站服务器是否真正部署启用了这些域名。

DV 证书验证通常用于检查与域名控制权相关的属性——即证书签发方在完全在线的交互过程中可直接观测到的属性。因此,在典型场景下,证书的申请、验证与签发等所有步骤均可通过互联网 IP 协议自动交互执行,无需额外的人工干预。

在 ACME 之前,在部署 HTTPS 服务器时,服务器操作员通常会要求向 CA 机构申请证书。若操作员改为使用 ACME 部署 HTTPS 服务器,会是这样的:

- a) ACME 客户端会提示操作员输入网站服务器使用的域名;
- b) ACME 客户端向操作员提供可获取证书的 CA 列表。ACME 客户端可能会提示操作员支付证书费用;
- c) 操作员选择一个 CA;
- d) 在后台,ACME 客户端联系 CA 并请求它为此域名签发证书;
- e) CA 验证客户端控制请求的域名,通过让 ACME 客户端执行一些只能通过控制域名来完成的操作。例如:CA 可能要求请求 example.org 的客户端提供 example.org 下的 DNS 记录或一个可访问的 HTTP 资源(<http://example.org>);
- f) 一旦完成 CA 要求的域名验证,CA 就会签发证书,ACME 客户端自动下载并安装它,可通过电子邮件、短信等方式通知操作员;
- g) ACME 客户端定期联系 CA 以获取更新证书,或任何其他应保留的内容,以确保网站服务器功能正常且其这些凭据是最新的。

这样,部署使用 CA 签发的证书比自己签发一张自签证书还容易。此外,维护 CA 签发的证书也只需要最少的人工干预。ACME 与 HTTPS 服务器如此紧密的结合使得服务器在证书签发后能立即自动部署证书,这使得网站管理员免于许多耗时的证书管理工作。

在我国商用密码应用合规要求下,ACME 协议的国密化改造具有以下典型应用场景:

- a) 政务网站 HTTPS/TLCP 加密:政府网站和政务服务平台需使用国密算法实现 HTTPS/TLCP 加密通信。ACME 客户端可自动向支持国密的 CA 申请 SM2 算法 TLS 证书,并自动部署到网站服务器;
- b) 企业内部 PKI 系统:企业可部署支持国密 ACME 协议的内部 CA,为内部信息系统自动签发和管理 SM2 算法证书;
- c) 云服务平台:云服务提供商可通过国密 ACME 协议为租户提供自动化证书管理服务,降低证书管理成本;
- d) 物联网设备认证:物联网设备可通过 ACME 协议自动申请设备证书,实现设备身份的自动化管理。

附录 C
(资料性)
安全注意事项

C.1 概述

ACME 是一种用于管理证明主体标识/密钥绑定的证书的协议。因此，ACME 的首要安全目标是确保此过程的完整性，即确保由证书证明的绑定是正确的，并且只有授权用户才能管理证书。ACME 通过账户密钥识别客户端，因此这个总体目标分解为两个更精确的目标：

- a) 只有控制主体标识的用户才能获取该主体标识的授权；
- b) 一旦授权，一个账户密钥的授权不能被另一个账户不当使用。

本附录描述了构成 ACME 基础的威胁模型，以及 ACME 在该模型下实现上述安全目标的方式；同时，对 ACME 服务端可能面临的拒绝服务（DoS）攻击风险及其他安全注意事项进行了说明。

C.2 威胁模型

作为 Internet 上的一项服务，ACME 广泛存在于 Internet 威胁模型中。在分析 ACME 时，考虑 ACME 服务端沿着两个“通道”与其他 Internet 主机交互是很有用的（见图 C.1）：

- a) 一个 ACME 通道，通过它交换 ACME HTTPS 请求；
- b) 一个验证通道，ACME 服务端通过该通道执行其他请求以验证客户端对主体标识的控制。

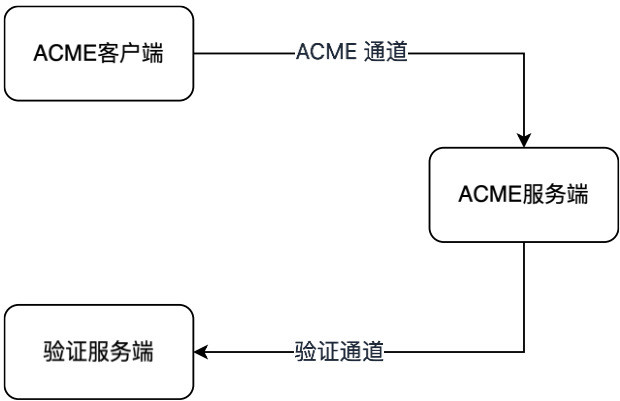


图 C.1 ACME 使用的通信通道

在实践中，这些渠道面临的风险并不是完全独立的，但在大多数情况下它们是不同的。例如：每个通道使用不同的通信模式：ACME 通道将包含到 ACME 服务端的入站 HTTPS 连接和验证通道出站 HTTP 或 DNS 请求。

从广义上讲，ACME 旨在防止任何单个通道上的主动和被动攻击者。当攻击者可同时利用 ACME 通道和其中一个通道时，会出现一些漏洞（如下所述）。

在 ACME 通道上，除网络层攻击外，还需防范应用层中间人（MitM）攻击及对协议本身的滥用行为。针对应用层中间人攻击的保护机制，主要用于防范诸如 CDN 和具备 TLS-MitM 功能的中间设备等潜在威胁节点。为防止 ACME 协议被滥用，需确保已获取验证通道访问权限的攻击者，无法通过伪装成合法的 ACME 客户端来骗取非法授权。

ACME 不防止 MitM 在 ACME 通道上进行其他类型的滥用。例如：此类攻击者可发送伪造的“badSignatureAlgorithm”错误响应，将客户端降级为服务端支持的最低质量签名算法。存在于所有连接（例如 CDN）上的 MitM 可通过多种方式导致拒绝服务条件。

C.3 授权的完整性

ACME 允许任何人通过注册账户密钥并使用该账户密钥发送 newOrder 请求来请求对主体标识的挑战，见图。因此，授权过程的完整性取决于主体标识验证挑战，以确保挑战只能由(1)持有帐户密钥对的私钥和(2)控制相关主体标识的人完成。

验证响应需要绑定到帐户密钥对，以避免 ACME HTTPS 请求上的 MitM 可将合法域名持有者的帐户密钥切换为他选择的帐户密钥的情况。例如：若 CA 在其 ACME 接口前使用 CDN 或第三方反向代理，就会出现此类 MitM。这种 MitM 的攻击可能具有以下形式：

- a) 合法域名持有者注册账户密钥对 A；
- b) MitM 注册账户密钥对 B；
- c) 合法域名持有者发送使用账户密钥 A 签名的 newOrder 请求；
- d) MitM 抑制合法请求，但发送使用帐户密钥 B 签名的相同请求；
- e) ACME 服务端发出挑战，MitM 将挑战转发给合法域持有者；
- f) 合法域持有者提供验证响应；
- g) ACME 服务端执行验证查询并查看合法域名持有者提供的响应；
- h) 因为挑战是为响应用户密钥B签名的消息而发出的,ACME 服务端将授权授予账户密钥B(MitM)而不是账户密钥 A（合法域持有者）。

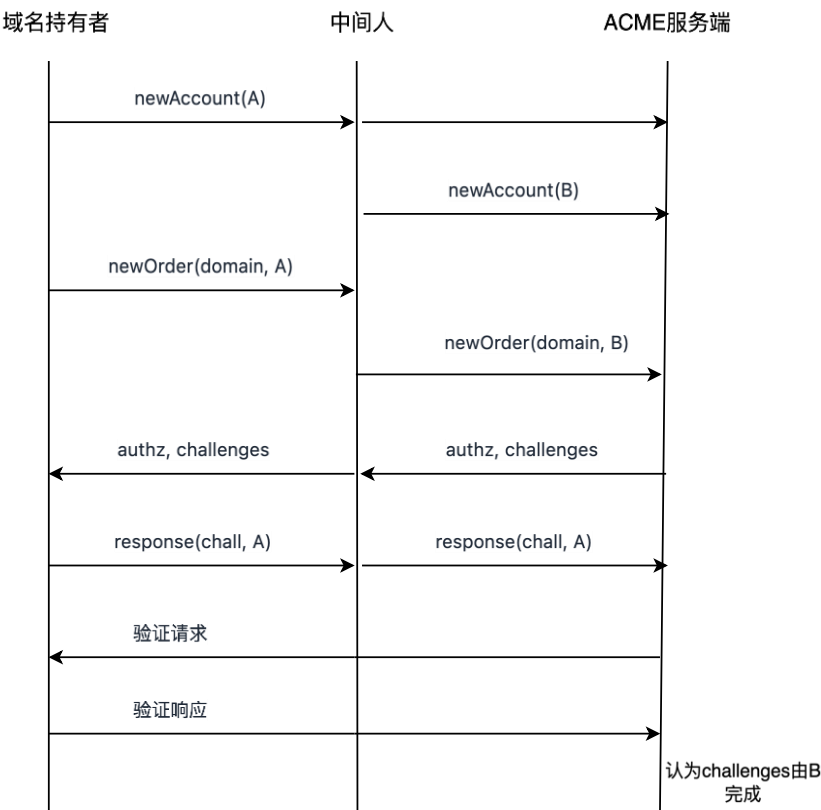


图 C.2 利用没有帐户密钥绑定的验证方法的中间人攻击

所有挑战都通过密钥授权在帐户私钥和服务端进行的验证查询之间进行绑定。密钥授权反映了帐户公钥，并通过验证通道在验证响应中提供给服务端。

挑战与主体标识的关联通常是通过要求客户端执行一些只有有效控制主体标识的人才能执行的操作来完成的。针对本文件中的挑战，操作如下：

- a) HTTP: 在域名的网站服务器上的 well-known 下提供文件;
- b) DNS: 为域名提供 DNS 资源记录。

有几种方法可违反这些假设, 包括错误配置和攻击。例如: 在允许非管理用户写入 well-known 的网站服务器上, 任何用户都可通过响应 HTTP 挑战来声明拥有该网站服务器的主机名。同样, 若可用于 ACME 验证的服务端被恶意行为者破坏, 则该恶意行为者可使用该访问权限通过 ACME 获取证书。

使用托管服务提供商是 ACME 验证的一个特殊风险。若域名的所有者将 DNS 或网站服务器的操作外包给托管服务提供商, 则托管服务提供商无法采取任何措施来防止篡改。在外界看来, 托管商提供的专区或网站才是真材实料。

更有限的委托形式也可能导致非预期方获取成功完成验证的能力。例如: 假设 ACME 服务端在 HTTP 验证中遵循 HTTP 重定向, 并且网站运营商提供了一个包罗万象的重定向规则, 将对未知资源的请求重定向到不同的域。然后重定向的目标可使用它通过 HTTP 验证获取证书, 因为主服务端不知道验证路径。

DNS 是所有这些挑战的一个常见漏洞点。可为域名提供虚假 DNS 记录的实体可直接攻击 DNS 挑战, 并且可提供虚假 A/AAAA 记录以指示 ACME 服务端将其 HTTP 验证查询发送到攻击者选择的远程服务端。ACME 服务端可应用几种不同的缓解措施:

- a) 始终使用 DNSSEC 验证解析器查询 DNS (增强启 DNSSEC 的区域的安全性);
- b) 从多个 DNS 服务查询 DNS 以应对本地攻击者;
- c) 对 DNS 路径外攻击者应用缓解措施, 例如: 向请求[DNS 0x20]添加熵或仅使用 TCP。

鉴于这些考虑, ACME 验证过程使得 ACME 通道上的任何攻击者或验证通道上的被动攻击者不可能劫持授权过程来授权攻击者选择的密钥。

只能看到 ACME 通道的攻击者需要说服验证服务端提供一个响应来授权攻击者的帐户密钥, 但这可通过绑定对用于请求挑战的帐户密钥的验证响应。验证通道上的被动攻击者可观察到正确的验证响应, 甚至可重播它, 但该响应只能与生成它的帐户密钥一起使用。

验证通道上的主动攻击者可通过执行正常的 ACME 事务并为他自己的帐户密钥提供验证响应来破坏 ACME 过程。上述托管服务提供商带来的风险是一个特例。

攻击者还可利用 Internet 路由协议中的漏洞来访问验证通道。为使此类攻击更加困难, 建议服务端执行 DNS 查询并从网络中的多个点建立 HTTP 连接。由于路由攻击通常是局部的或取决于攻击者的位置, 迫使攻击者攻击多个点 (服务端的验证点) 或特定点 (DNS/HTTP 服务器) 使得在路由上攻击破坏 ACME 验证变得更加困难。

C.4 拒绝服务注意事项

作为在 HTTPS 上运行的协议, 基于 TCP 和基于 HTTP 的 DoS 缓解的标准注意事项也适用于 ACME。

在应用层, ACME 要求服务端执行一些可能代价高昂的操作。主体标识验证事务需要 ACME 服务端与潜在的攻击者控制的服务端建立出站连接, 并且证书签发可能需要与加密硬件进行交互。

此外, 攻击者还可通过向受害域名提交授权请求, 使 ACME 服务端向其选择的域名发送验证请求。

所有这些攻击都可通过应用适当的速率限制来减轻。更靠近前端的问题, 如 POST 正文验证, 可使用 HTTP 请求限制来解决。对于验证和证书请求, 还有其他可设置速率限制的标识符。例如: 服务端可能会限制任何个人帐户密钥可签发证书的速率, 或者可在 DNS 的给定子树中请求验证的速率。为防止攻击者通过创建新账户来规避这些限制, 服务端应限制账户注册的速度。

C.5 服务端请求伪造

当攻击者可使服务端对攻击者选择的 URL 执行 HTTP 请求时, 就会出现服务端请求伪造 (SSRF) 攻击。在 ACMEHTTP 挑战验证过程中, ACME 服务端对攻击者可在其中选择域名的 URL 执行 HTTPGET 请求。此请求是在服务端验证客户端控制域名之前发出的, 因此任何客户端都可对任何域名进行查询。

一些 ACME 服务端实现包括来自验证服务端响应的信息（以便于调试）。此类实施使攻击者能够从 ACME 服务端可访问的任何网站服务器中提取此信息，即使 ACME 客户端无法访问它也是如此。例如：ACME 服务端可能能够访问防火墙后面的服务端，防火墙会阻止 ACME 客户端访问。

似乎 SSRF 通过此通道的风险受到攻击者只能控制 URL 的域名而不是路径这一事实的限制。但是，若攻击者首先将域名设置为他们控制的域名，则他们可向服务端发送 HTTP 重定向（例如：302 响应），这将导致服务端查询任意 URL。

为进一步限制 SSRF 风险，ACME 服务端运营商应确保验证查询只能发送到公共互联网上的服务端，而不是服务端运营商内部网络中的网站服务。由于攻击者可自己向这些公共服务端发出请求，因此除了匿名层之外，他无法通过对 ACME 的 SSRF 攻击获取任何额外的收益。

C.6 CA 策略注意事项

ACME 对证书签发的控制主要集中在验证证书申请人是否控制他声明的主体标识。然而，在签发证书之前，CA 可能需要执行许多其他检查，例如：

- a) 客户是否同意订户协议；
- b) 声明的主体标识在语法上是否有效；
- c) 对于域名：
 - 1) 若最左边的标签是“*”，则是否应用了适当的检查；
 - 2) 该名称是否在公共后缀列表中；
 - 3) 域名是否为高价值域名；
 - 4) 该域名是否为已知的网络钓鱼域名。
- d) 证书申请文件中的密钥是否足够强大；
- e) 证书申请文件是否使用可接受的算法签名；
- f) 证书签发机构授权(CAA)记录是否授权或禁止签发。

使用 ACME 自动签发证书的 CA 应确保他们的服务端在签发证书之前执行所有必要的检查。若 ACME 用于签发应验证网站身份的 IV/OV/EV 证书，则 CA 机构应已经现有的 CA 标准对网站完成身份鉴别后才签发证书。

使用 ACME 允许客户端同意服务条款的 CA 应记住，ACME 客户端可自动执行此协议，可能不是真人用户。

ACME 不指定服务端如何构造它用于寻址资源的 URL。若服务端操作员使用第三方可预测的 URL，这可能会泄露有关服务端上存在哪些 URL 的信息，因为攻击者可探测对 URL 的 POST-as-GET 请求是返回 404（未找到）还是 401（未经授权）。

例如，假设 CA 使用具有可猜测字段的高度结构化 URL：

- a) 账户：https://example.com/:accountID;
- b) 订单：https://example.com/:accountID/:domainName;
- c) 授权：https://example.com/:accountID/:domainName;
- d) 证书：https://example.com/:accountID/:domainName;

根据该方案，若 CA 不允许，攻击者可探查哪些域名与哪些帐户相关联，这可能允许域名之间的所有权关联。

为避免泄露这些相关性，CA 应分配具有不可预测组件的 URL。例如：CA 可能会为来自独立命名空间的每种资源类型分配 URL，为每种资源使用不可预测的 ID：

- a) 账户：https://example.com/acct/:accountID;
- b) 订单：https://example.com/order/:orderID;
- c) 授权：https://example.com/authz/:authorizationID;

d) 证书：<https://example.com/cert/:certID>。

这样的方案只会泄漏资源的类型，隐藏上面示例中显示的其他相关性。

附录 D

（资料性）

操作注意事项

D.1 概述

基于 ACME 的 CA 运营方在配置其服务时应牢记运营现实中出现的某些因素。有关示例，请见下面的小节。

D.2 密钥选择

ACME 依赖于两种不同类别的密钥对：

- a) 账户密钥对，用于验证账户持有人；
- b) 证书密钥对，用于签署和验证 CSR（其公钥包含在证书中）。

账户密钥对的私钥被泄露比证书对应的私钥被泄露具有更严重的后果。虽然证书密钥对的泄露允许攻击者在证书的生命周期内模拟证书中指定的实体，但账户密钥对的泄露允许攻击者完全控制受害者的 ACME 帐户并采取任何行动合法账户持有人可在 ACME 范围内采取：

- a) 使用现有授权签发证书；
- b) 撤销现有证书；
- c) 访问和更改帐户信息（例如联系人）；
- d) 更改账户的账户密钥对，锁定合法账户持有人。

因此，建议每个帐户密钥对仅用于单个 ACME 帐户的身份验证。例如：证书中不应包含帐户密钥对的公钥。若 ACME 客户端收到的帐户创建或密钥更新请求中，使用了已知在别处使用过的帐户密钥，客户端应返回错误。客户端应为每次帐户创建或更新操作生成全新的账户密钥。依据第 7.4.1 节的规定，服务端在任何情况下均不应使用重复的密钥来创建帐户。

ACME 客户端和服务端应验证在最终请求中提交的 CSR 不包含任何已知帐户密钥对的公钥。特别是，当服务端接收到 finalize 请求时，服务端应验证 CSR 中的公钥与用于验证该请求的帐户密钥对的公钥不同。这确保了使用证书的协议中的漏洞（例如：在 TLS[JSS15]中签署 oracle）不会导致 ACME 帐户受到损害。因为 ACME 帐户由他们的账户密钥对唯一标识（见第 7.4.1 节），服务端不应允许跨多个帐户重用账户密钥对。

D.3 DNS 安全

如上所述，针对 ACME 服务端的 DNS 伪造攻击可能导致服务端做出有关域名控制的错误决策，从而错误签发证书。服务端应通过 TCP 执行 DNS 查询，这比通过 UDP 的 DNS 提供更好的抵抗某些伪造攻击的能力。

基于 ACME 的 CA 通常需要进行 DNS 查询，例如：验证对域名的控制权。由于此类验证的安全性最终取决于 DNS 数据的真实性，因此应采取一切可能的预防措施来保护 CA 进行的 DNS 查询。因此，建议基于 ACME 的 CA 通过 DNSSEC 验证存根或递归解析器进行所有 DNS 查询。这为选择使用 DNSSEC 的域名提供了额外的保护。

若基于 ACME 的 CA 信任解析器和访问它的网络路由的每个组件，则它只能使用解析器。因此，建议基于 ACME 的 CA 在其受信网络中运行自己的 DNSSEC 验证解析器，并将这些解析器用于 CAA 记录查找和所有记录查找以改进挑战方案（A、AAAA、TXT 等）。

D.4 令牌熵（Token Entropy）

http-01 和 dns-01 验证方法强制使用随机令牌值来唯一标识挑战。对于以下安全属性，令牌的值

应包含至少 128 位的熵。首先，ACME 客户端不应能够影响 ACME 服务端对令牌的选择，因为这可能允许攻击者为新的验证请求重用域名所有者先前的挑战响应。其次，熵要求使 ACME 客户端更难实现“简单”的验证服务端，该服务端无需针对每个挑战进行配置即可自动回复挑战。

D.5 格式错误的证书链

ACME 以广泛使用的格式提供证书链，通俗地称为 PEM。当前的一些软件将允许通过连接两个对象的文本编码来在一个 PEM 文件中配置私钥和证书。在 ACME 的上下文中，此类软件可能容易受到密钥替换攻击。恶意 ACME 服务端可能导致客户端使用其选择的私钥，方法是将私钥包含在响应证书 URL 查询而返回的 PEM 文件中。

当处理“application/pem-certificate-chain”类型的文件时，客户端应验证该文件只包含编码证书。若发现除证书以外的任何内容（即，若字符串“-----BEGIN”后面跟着除“CERTIFICATE”之外的任何内容），则客户端应拒绝该文件，认为该文件无效。

参 考 文 献

- [1] IETF. Automatic Certificate Management Environment (ACME): RFC 8555 [S/OL]. (2019-03). <<https://www.rfc-editor.org/rfc/rfc8555>>
- [2] IETF. Internet X.509 Public Key Infrastructure Operational Protocols: FTP and HTTP: RFC 2585 [S/OL]. (1999-05). <<https://www.rfc-editor.org/info/rfc2585>>.
- [3] IETF. HTTP Over TLS: RFC 2818 [S/OL]. (2000-05). <<https://www.rfc-editor.org/info/rfc2818>>.
- [4] IETF. PKCS #9: Selected Object Classes and Attribute Types Version 2.0: RFC 2985 [S/OL]. (2000-11). <<https://www.rfc-editor.org/info/rfc2985>>.
- [5] IETF. PKCS #10: Certification Request Syntax Specification Version 1.7: RFC 2986 [S/OL]. (2000-11). <<https://www.rfc-editor.org/info/rfc2986>>.
- [6] IETF. UTF-8, a transformation format of ISO 10646: RFC 3629 [S/OL]. (2003-11). <<https://www.rfc-editor.org/info/rfc3629>>.
- [7] IETF. Uniform Resource Identifier (URI): Generic Syntax: RFC 3986 [S/OL]. (2005-01). <<https://www.rfc-editor.org/info/rfc3986>>.
- [8] IETF. Randomness Requirements for Security: RFC 4086 [S/OL]. (2005-06). <<https://www.rfc-editor.org/info/rfc4086>>.
- [9] IETF. The Base16, Base32, and Base64 Data Encodings: RFC 4648 [S/OL]. (2006-10). <<https://www.rfc-editor.org/info/rfc4648>>.
- [10] IETF. Augmented BNF for Syntax Specifications: ABNF: RFC 5234 [S/OL]. (2008-01). <<https://www.rfc-editor.org/info/rfc5234>>.
- [11] IETF. Secure/Multipurpose Internet Mail Extensions (S/MIME) Version 3.2 Message Specification: RFC 5751 [S/OL]. <<https://www.rfc-editor.org/info/rfc5751>>.
- [12] ETF. The 'mailto' URI Scheme: RFC 6068 [S/OL]. (2010-10). <<https://www.rfc-editor.org/info/rfc6068>>.
- [13] IETF. URI Template: RFC 6570 [S/OL]. (2012-03). <<https://www.rfc-editor.org/info/rfc6570>>.
- [14] ETF. Textual Encodings of PKIX, PKCS, and CMS Structures: RFC 7468 [S/OL]. (2015-04). <<https://www.rfc-editor.org/info/rfc7468>>.
- [15] ETF. JSON Web Algorithms (JWA): RFC 7518 [S/OL]. (2015-05). <<https://www.rfc-editor.org/info/rfc7518>>.
- [16] IETF. JSON Web Key (JWK) Thumbprint: RFC 7638 [S/OL]. (2015-09). <<https://www.rfc-editor.org/info/rfc7638>>.
- [17] IETF. The JavaScript Object Notation (JSON) Data Interchange Format: RFC 8259 [S/OL]. (2017-12). <<https://www.rfc-editor.org/info/rfc8259>>.
- [18] IETF. Web Linking: RFC 8288 [S/OL]. (2017-10). <<https://www.rfc-editor.org/info/rfc8288>>.
- [19] IETF. The Transport Layer Security (TLS) Protocol Version 1.3: RFC 8446 [S/OL]. (2018-08). <<https://www.rfc-editor.org/info/rfc8446>>.
- [20] IETF. CAA Record Extensions for Account URI and ACME Method Binding: draft-ietf-acme-caa-06 [R/OL]. (2019-01).

- [21] IETF. ACME IP Identifier Validation Extension: draft-ietf-acme-ip-05 [R/OL]. (2019-02).
- [22] IETF. ACME Identifiers and Challenges for Telephone Numbers: draft-ietf-acme-telephone-01 [R/OL]. (2017-10).
- [23] CA/Browser Forum. CA/Browser Forum Baseline Requirements [S/OL]. (2018-09). <<https://cabforum.org/baseline-requirements-documents/>>.
- [24] IETF. Use of Bit 0x20 in DNS Labels to Improve Transaction Identity: draft-vixie-dnsext-dns0x20-00 [R/OL]. (2008-03).
- [25] IETF. Privacy Enhancement for Internet Electronic Mail: Part I: Message Encryption and Authentication Procedures: RFC 1421 [S/OL]. (1993-02). <<https://www.rfc-editor.org/info/rfc1421>>.
- [26] IETF. X.509 Internet Public Key Infrastructure Online Certificate Status Protocol – OCSP: RFC 6960 [S/OL]. (2013-06). <<https://www.rfc-editor.org/info/rfc6960>>.
- [27] IETF. Threat Model for BGP Path Security: RFC 7132 [S/OL]. (2014-02). <<https://www.rfc-editor.org/info/rfc7132>>.
- [28] W3C. Cross-Origin Resource Sharing [S/OL]. (2014-01). <<http://www.w3.org/TR/2014/REC-cors-20140116>>.
-