FIDO UAF 认证器元数据服务 v1.0

FIDO 联盟推荐标准 2014-12-08

当前版本:

https://fidoalliance.org/specs/fido-uaf-v1.0-ps-20141208/fido-uaf-authnr-metadata-service-v1.0-ps-20141208.html

之前版本:

https://fidoalliance.org/specs/fido-uaf-metadata-service-v1.0-rd-20141008.pdf

编写者:

罗尔夫•林德曼博士（Dr. Rolf Lindemann）, Nok Nok Labs, Inc.

贡献者:

布拉德•希尔（Brad Hill）, 贝宝（PayPal, Inc.）.

达维特•巴格达萨利安（Davit Baghdasaryan）, Nok Nok Labs, Inc.

摘要

FIDO UAF 认证器元数据协议定义了“认证器元数据”。元数据声明 包括可信锚要求的验证鉴证对象，同时元数据也描述了很多认证器其他的重要特征。

本文档描述的元数据服务定义了依赖方访问最新的元数据描述的基准方法。

文档状态

本章节描述了文档发布时的状态。本文档有可能会被其它文档所取代。当前 FIDO 联盟出版物的列表以及此技术报告的最新修订可在 FIDO 联盟规范索引 上找到。

网址：https://www.fidoalliance.org/specifications/.

本文档由 FIDO 联盟 作为推荐标准发布。如果您希望就此文档发表评论，请 联系我们 。欢迎所有评论。

本规范中某些元素的实现可能需要获得第三方知识产权的许可，包括（但不限于）专利权。FIDO 联盟及其成员，以及此规范的其他贡献者们不能，也不应该为任何识别或未能识别所有这些第三方知识产权的行为负责。

本 FIDO 联盟规范是“按原样”提供，没有任何类型的担保，包括但不限于，任何明确的或暗示的不侵权、适销性或者适合某一特定用途的担保。

本文档已经由 FIDO 联盟成员评审并签署成为推荐标准。这是一篇稳定的文档，可能会作为参考材料被其它文档引用。FIDO 联盟的作用是引起对规范的注意并促进其广泛的分发。

目录

1. 注释 3
    1.1 关键字 3
2. 概览 3
    2.1 范围 4
    2.2 结构细节 4
3. 元数据服务细节 5
    3.1 元数据 TOC 格式 6
        3.1.1 元数据 TOC 有效载荷条目结构 6
            3.1.1.1 MetadataTOCPayloadEntry 结构成员 6
        3.1.2 StatusReport 结构 8
            3.1.2.1 StatusReport 结构成员 9
        3.1.3 认证器状态枚举 10
        3.1.4 元数据 TOC 有效载荷结构 11
            3.1.4.1 MetadataTOCPayload 结构成员 11
        3.1.5 元数据 TOC 12
            3.1.5.1 举例 12
        3.1.6 元数据 TOC 对象处理规则 17
4. 注意事项 18

A. 参考文献 20
A.1 参考规范 20
A.2 参考资料 21

1. 注释

类型名称、属性名称和元素名称用代码形式书写。

字符串文本包含在双引号“”内，比如“UAF-TLV”。

公式中用 “|” 来表示按字节串联操作。

base64url(byte[8..64])代表 8-64 字节的数据用 base64url 形式编码。引用自*无填充*的"Base 64 Encoding with URL and Filename Safe Alphabet" [RFC4648]。

根据[WebIDL-ED],结构成员是可选的，除非他们被明确标注为 required。

WebIDL 的结构成员不得为空值。

除非特别声明，如果 WebIDL 的结构成员是 DOMString，则不得为空。

除非特别声明，如果 WebIDL 的结构成员是一个表单，则不得为一个空表单。

本文档中用到的 UAF 专用术语在 FIDO 术语表[FIDOGlossary]中均有定义。

此规范中的所有的图表、示例、注释都是非规范的。

> **注释**
>
> 特定的结构成员需要遵从 FIDO 协议的要求。本文档中以 required 为标示，标注了这些词汇在 WebIDL 的定义。关键词 required 是在开发中版本[WebIDL-ED]提出，如果使用执行 WebIDL 开发程序的解析器，则可删除在 WebIDL 中的关键词 required 并通过其他方式将这些字段填满。

1.1 关键字

本文档中的关键字：“必须”，“不得”，“要求”，“将”，“将不”，“应该”，“不应该”，“建议”，“可能”，“可选”都会按照[RFC2119]的描述来解释。

2. 概览

本节是非规范性的。

FIDO UAF 规范定义了 对于认证器元数据的表述[UAFAuthnrMetadata]。

元数据声明包括信任锚要求的验证鉴证对象，具体来说是 KeyRegistrationData 对象，也包括了认证器许多其他的重要特征，包括 AAID、支持的鉴别和注册断言方案和密钥保护标志。

这些特征可以用于[UAF protocol]中定义的关于何种认证器可以接受注册或鉴别的策略。

本文档描述的元数据服务定义了依赖方获取最新元数据描述的基准方法。

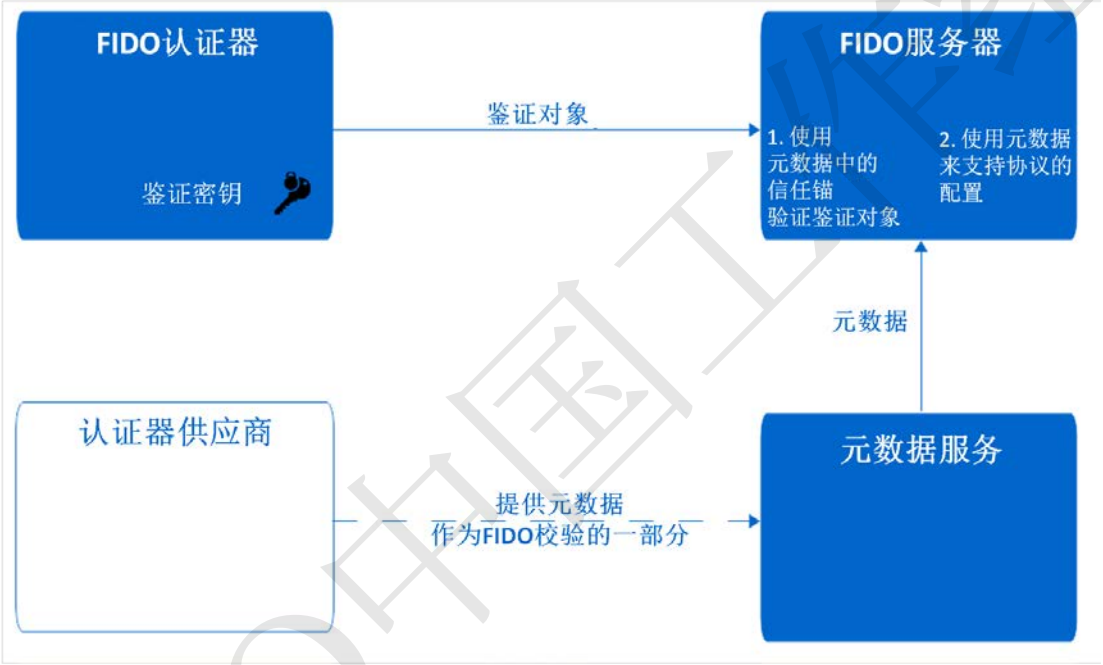

图 1 UAF 元数据服务架构概览

2.1 范围

本文档详细地描述了 FIDO 元数据服务的架构，定义了访问服务的结构和接口，同时定义了元数据相关的信息流并展示了设计方案背后的原理。

2.2 结构细节

元数据目录（TOC）文件包含 FIDO 联盟认证器元数据列表。

FIDO 服务器从 FIDO URL 下载并缓存元数据 TOC 文件到本地。

FIDO 服务器通过数字签名验证元数据 TOC 文件的完整性和真实性。然后，遍历每条记录，加载与依赖方相关的认证器 AAID 对应的元数据声明。

从 TOC 文件条目中特定的 URL 下载单个元数据声明，必要条目会被 FIDO 服务器缓存。

FIDO 服务器使用元数据 TOC 文件的相关条目中的哈希值验证元数据声明的完整性。

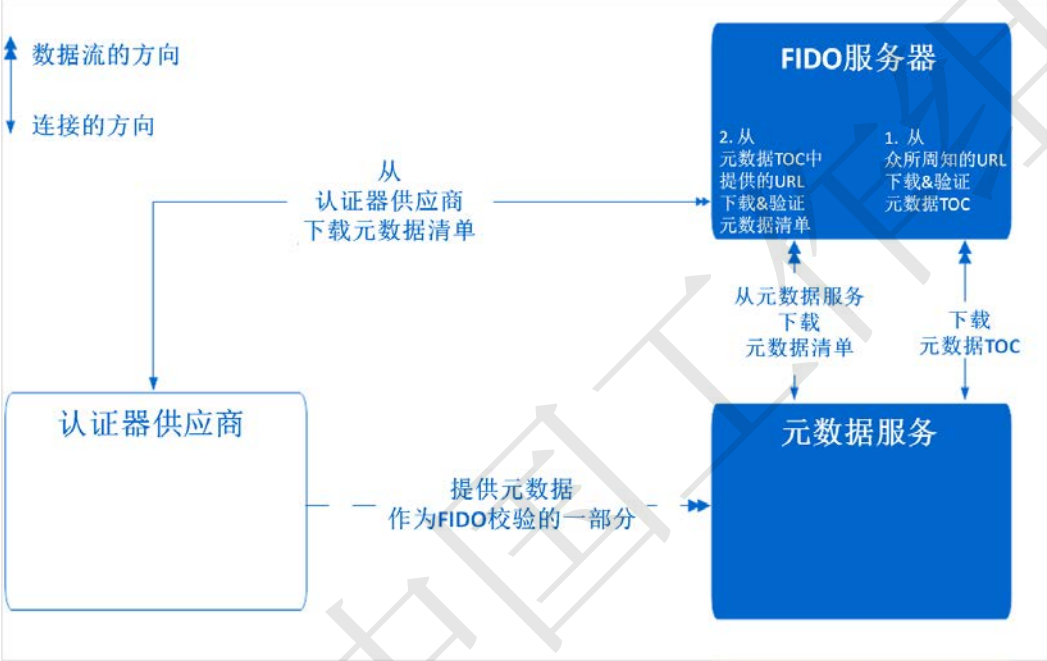

图 2 UAF 元数据服务架构

注释

单箭头表示网络连接的方向，双箭头表示数据流的方向。

注释

元数据 TOC 文件可以在 FIDO 联盟发布的已知 URL 上自由访问。

注释

依赖方决定元数据服务检查元数据目录更新的频率。

3. 元数据服务细节

本节是规范性的。

注释

依赖方决定是否使用元数据服务以及是否愿意接受某些认证器注册或鉴别。

依赖方亦可从认证器厂商或其他可靠来源直接获取元数据。

3.1 元数据 TOC 格式

注释

元数据服务可以让 FIDO 服务器访问元数据 TOC 对象（参见 Metadata TOC）。该对象是元数据“目录”，包括 AAID、用于下载的 URL 和单个元数据的哈希值。TOC 对象包含一个签名。

3.1.1 元数据 TOC 有效载荷条目结构

代表元数据 TOC 有效载荷条目：

WebIDL

```
dictionary MetadataTOCPayloadEntry{
    required AAID           aaid;
    required DomString      hash;
    required DomString      url;
    required StatusReport[] statusReports;
    required DomString      timOfLastStatusChange;
};
```

3.1.1.1 MetadataTOCPayloadEntry 结构成员

aaid类型为 required AAID

与元数据 TOC 有效载荷记录相关的认证器的 AAID。AAID 结构的定义参见 [UAFProtocol]。

hash类型为 required DOMString

base64url(string[1..512])

如[UAFAuthnrMetadata]中定义，将 url 中可用的 JSON 编码的元数据表使用 UTF-8 编码，结果再使用 Base64URL 编码后的哈希值。必须使用与 JWTHeader（参见 Metadata TOC）中特定的签名算法相关的哈希算法。

> **注释**
>
> Base64url 编码的 UTF-8 编码的方法也被用于 JWT[JWT]以避免编码模糊。

url类型为 required DOMString

为某个认证器类型（通过自身 AAID 识别）编码元数据声明的统一资源定位符（URL），必须指向[UAFAuthnrMetadata]中定义的 base64url 编码的 UTF-8 编码的 JSON 编码的元数据声明。

encodedMetadataStatement = Base64url(utf8(JSONMetadataStatement))

> **注释**
>
> Base64url 编码的 UTF-8 编码的方法也被用于 JWT[JWT]以避免编码模糊。

statusReports类型为 required StatusReport 数组

适用于认证器的状态报告数组。

timeOfLastStatusChange 类型为 required DOMString

IOS-8601 格式化的状态报告数组设置为当前值时的日期。

例 1：UAF 元数据 TOC 有效载荷

```
{ "no": 1234, "next-update": "2014-03-31",
   "entries": [
      { "aaid": "1234#5678",
        "hash":
        "90da8da6de23248abb34da0d4861f4b30a793e198a8d5baa7f98f260db71acd
        4",
        "url": "https://fidoalliance.org/metadata/1234%x23abcd",
        "statusReports": [
                    { status: "FIDO_CERTIFIED", effectiveDate: "2014-01-04"}
                    ],
        "timeOfLastStatusChange": "2014-01-04"
```

```
        },
      { "aaid": "9876#4321",
        "hash":
        "785d16df640fd7b50ed174cb5645cc0f1e72b7f19cf22959052dd20b9541c64
        d",
        "url": "https://authnr-vendor-a.com/metadata/9876%x234321",
        "statusReports": [
                { status: "FIDO_CERTIFIED", effectiveDate: "2014-01-07"},
                { status: "UPDATE_AVAILABLE", effectiveDate: "2014-03-08",
                url: "https://example.com/update1234" }
                ],
      "timeOfLastStatusChange": "2014-02-19"
      }
  ]
}
```

注释

字符 # 是保留字符，不允许出现在 URL 中[RFC3986]。因此被十六进制值%x23 替换。

认证器厂商可以决定让元数据服务发布元数据声明或者自己发布元数据声明。认证器厂商可以限制对自己发布的元数据声明的访问。

3.1.2 StatusReport 结构

注释

包含 AuthenticatorStatus 和相关的附加数据（如果有）。

新的 StatusReport 条目将在固件更新时被添加用来报告现存已知问题。

最新的 StatusReport 记录必须反映“当前”的状态。例如，如果最新的条目有 USER_VERIFICATION_BYPASS 的状态，则对应此 AAID 的认证器的风险很有可能增加。如果最新的记录有状态 UPDATE_AVAILABLE，那么更新则要解决至少之前所有在 StatusReport 结构*报告*的问题。

WebIDL

```
dictionary StatusReport{
    required AuthenticatorStatus    status;
    DomString                       effectiveDate;
    DomString                       certificate;
    DomString                       url;
};
```

3.1.2.1 StatusReport 结构成员

status类型为 required AuthenticatorStatus

认证器状态。其他字段可能会根据本字段的值来设置。

effectiveDate类型为 DOMString

当状态码被设定后，若适用则根据 ISO-8601 对日期进行格式化。如果没有给定日期则状态默认为当前有效。

certificate类型为 DOMString

若适用为 Base64 编码的[RFC4648] (不是 base64url!) 与现在状态相关 DER [ITU-X690-2008] PKIX 证书值。

> 注释
>
> 例如与一组泄露认证器（ATTESTATION_KEY_COMPROMISE）相关的鉴证根证书（参见[UAFAuthnrMetadata]）。

url 类型为 DOMString

若适用为 HTTPS URL，在其上可能发现与当前状态相关的额外信息。

> 注释
>
> 例如在 UPDATE_AVAILABLE 状态下描述可用固件更新的链接，或在

USER_VERIFICATION_BYPASS 状态下描述一个可识别问题的链接。

3.1.3 认证器状态枚举

此枚举描述了认证器类型的状态，这些认证器类型是由 AAID 和一些潜在的附加信息（如特定的鉴证密钥）识别的。

WebIDL

```
enum AuthenticatorStatus {
    "FIDO_CERTIFIED",
    "NOT_FIDO_CERTIFIED",
    "USER_VERIFICATION_BYPASS",
    "ATTESTATION_KEY_COMPROMISE",
    "USER_KEY_REMOTE_COMPROMISE",
    "USER_KEY_PHYSICAL_COMPROMISE",
    "UPDATE_AVAILABLE",
    "REVOKED"
};
```

枚举描述	
FIDO_CERTIFIED	该认证器通过 FIDO 认证的。
NOT_FIDO_CERTIFIED	该认证器未通过 FIDO 认证的。
USER_VERIFICATION_BYPASS	表示恶意软件能绕过用户校验。这表明认证器能在没有经过用户同意甚至用户不知道的情况下使用。
ATTESTATION_KEY_COMPROMISE	表明认证器的鉴证密钥是被泄露的。如果知道则需要补充额外信息，包括密钥标识符和泄露的日期。
USER_KEY_REMOTE_COMPROMISE	表明认证器具有已识别的可泄露注册密钥的缺陷，该认证器不能继续被信任。这包括，能生成可预测密钥的弱熵，或者允许伪造、猜测、提取密钥或签名的旁路通道。
USER_KEY_PHYSICAL_COMPROMISE	该认证器的密钥保护机制具有已知的弱点，这使得设备的物理拥有者可以提取用户密钥。

UPDATE_AVAILABLE	当设备软件或固件有可用更新时。应该提供额外信息，包括一个用户获取更新的 URL 和更新发布的日期。 当这段代码被使用时，必须更新元数据声明[UAFAuthnrMetadata]中的 authenticatorVersion字段。 如果更新修正了很严重的安全问题，例如之前的状态报告条目包含下列状态码 USER_VERIFICATION_BYPASS,ATTESTATION_KEY_COMPROMISE, USER_KEY_REMOTE_COMPROMISE, USER_KEY_PHYSICAL_COMPROMISE, REVOKED。 **注释** 依赖方可能需要告知用户可用的固件更新。
REVOKED	FIDO 联盟决定该认证器由于种种原因不应该被信任，例如已知其具是虚假产品或者包含故意留有的后门。

3.1.4 元数据 TOC 有效载荷结构

代表元数据 TOC 有效载荷。

WebIDL

```
dictionary MetadataTOCPayload{
    required Number                     no;
    required DomString                  nextUpdate;
    required MetadataTOCPayloadEntry[]  entries;
};
```

3.1.4.1 MetadataTOCPayload 结构成员

no类型为 required Number

该 UAF 元数据 TOC 有效载荷的序列号。序列号必须是连续且严格单调递增的。例如，后续的 TOC 的 no 值以 1 为单位递增。

nextUpdate类型为 required DOMString

ISO-8601 格式化的下次提供最新更新的日期。

entries类型为 required MetadataTOCPayloadEntry 数组

0 或多个 MetadataTOCPayloadEntry 对象的列表。

3.1.5 元数据 TOC

元数据目录(TOC) 是一个 JSON Web Token (参见[JWT]和[JWS])。
它包含三个元素：

- UTF-8 编码的 JWT 报头的 base64url 编码，无填充（见后面的例子）。
- UTF-8 编码的 UAF 元数据 TOC 有效载荷的 base64url 编码，无填充（参见 Metadata TOC Format开头的例子）。
- 通过将被签名的有效载荷计算的 JWS 签名[JWS]的 base64url 编码，同样也没有填充。例如

tbsPayload = EncodedJWTHeader | "." | EncodedMetadataTOCPayload

TOC 的三个元素通过句号 (".")连接起来。

MetadataTOC = EncodedJWTHeader | "." | EncodedMetadataTOCPayload | "." | EncodedJWSSignature

与 JWT 报头中（例如报头是“ES256”的情况下为 SHA256）特定的签名算法相关的哈希算法必须用来计算元数据声明的哈希值(参见 Metadata TOC Payload Entry Dictionary)。

3.1.5.1 举例

本节是非规范性的。

例 2: 经过编码的元数据声明

eyAiQUFJRCI6ICIxMjM0IzU2NzgiLA0KICAiQXR0ZXN0YXRpb25Sb290Q2VydGlmaWNhdGUiOiAiTUlJQ1BUQ0NBZU9nQXdJQkFnSUpBT3VleHZVM095MndNQW9HQ0NxR1NNNDlCQU1DTUhzeElEQWVCZ05WQkFNTQ

0KRjFOaGJYQnNaU0JCZEhSbGMzUmhkR2x2YmlCU2IyOTBNUll3RkFZR
FZRUUtEQTFHU1VSUElFRnNiR2xoYm1ObA0KTVJFd0R3WURWUVFM
REFoVlFVWWdWRmRITERFU01CQUdBMVVFQnd3SlVHRnNieUJCYkhS
dk1Rc3dDUVlEVlFRSQ0KREFKRFFURUxNQWtHQTFVRUJoTUNWVk13
SGhjTk1UUXdOakU0TVRNek16TXlXaGNOTkRFeE1UQXpNVE16TXpNe
Q0KV2pCN01TQXdIZ1lEVlFRRERCZFRZVzF3YkdVZ1FYUjBaWE4wWV
hScGIyNGdVbTl2ZERFV01CUUdBMVVFQ2d3Tg0KUmtsRVR5QkJiR3hw
WVc1alpURVJNQThHQTFVRUN3d0lWVUZHSUZSWFJ5d3hFakFRQmdO
VkJBY01DVkJoYkc4Zw0KUVd4MGJ6RUxNQWtHQTFVRUNBd0NRMEV
4Q3pBSkJnTlZCQVlUQWxWVE1Ga3dFd1lIS29aSXpqMENBUVlJS29aSQ0
KemowREFRY0RRZ0FFSDhodjJEMEhYYTU5L0JtcFE3UlplaEwvRk1HekZ
kMVFCZzl2QVVwT1ozYWpudVE5NFBSNw0KYU16SDMzblVTQnI4ZkhZ
RHJxT0JiNThweEdxSEpSeVgvNk5RTUU0d0hRWURWUjBPQkJZRUZQb0h
BM0NMaHhGYg0KQzBJdDd6RTR3OGhrNUVKL01COEdBMVVkSXdRW
U1CYUFGUG9IQTNDTGh4RmJDMEl0N3pFNHc4aGs1RUovTUF3Rw0KQ
TFVZEV3UUZNQU1CQWY4d0NnWUlLb1pJemowRUF3SURTQUF3UlFJa
EFKMDZRU1h0OWloSWJFS1lLSWpzUGtyaQ0KVmRMSWd0ZnNiRFN1N
0VySmZ6cjRBaUJxb1lDWmYwK3pJNTVhUWVBSGpJekE5WG02M3JydUF
4Qlo5cHM5ejJYTg0KbFE9PSIsDQogICJEZXNjcmlwdGlvbiI6ICJGSURPIEF
sbGlhbmNlIFNhbXBsZSBVQUYgQXV0aGVudGljYXRvciIsDQogICJVc2Vy
VmVyaWZpY2F0aW9uTWV0aG9kcyI6IDIsDQogICJWYWxpZEF0dGFjaG1l
bnRUeXBlcyI6IDEsDQogICJLZXlQcm90ZWN0aW9uIjogNiwNCiAgIk1hdG
NoZXJQcm90ZWN0aW9uIjogMiwNCiAgIlNlY3VyZURpc3BsYXkiOiA0LA
0KICAiU2VjdXJlRGlzcGxheUNvbnRlbnRUeXBlcyI6IFsiaW1hZ2UvcG5nIl0
sDQogICJTZWN1cmVEaXNwbGF5UE5HQ2hhcmFjdGVyaXN0aWNzIjogW
1swLDAsMSw2NCwwLDAsMSwyMjQsMTYsMiwwLDAsMF1dLA0KICAia
XNTZWNvbmRGYWN0b3JPbmx5IjogImZhbHNlIiwNCiAgIkljb24iOiAiZGF
0YTppbWFnZS9wbmc7YmFzZTY0LGlWQk9SdzBLR2dvQUFBQU5TVWhF
VWdBQUFFOEFBQUF2Q0FZQUFBQ2l3SmZjQUFBQUFYTlNSMElBcnM0
YzZRQUFBQVJuUVUxQkFBQ3gNCmp3djhZUVVBQUFBSmNFaFpjd0FBR
HNNQUFBN0RBY2R2cUdRQUFBYWhTVVJCVkdoRDdacjVieFJsR01mOU
t6VEI4QU0vWUVoRTJXN3ANClFaY1dLS0JjbFNwSEFUbEVMQVJFN2tO
RUNDQTNGa1dLMENLS1NDRklzS0JjZ1ZDRFdHTkVTZEFZaWR3Z2dnSk
JpUmlNaEZjLzR3eTgNCjg4NHp1OU5kbG5HVGZaSlAybjNuTysrODg5MzN
mdmVCQngrUHFDekprVFV2QmJMbXBVRFd2QlRJbXBjQ1NadlhMQ2RY
OVIwNVNrMTkNCmJiNWF0ZjU5OWZHKy9lckE1NDFxNDdhUDFMTFZh
OVNJeVZOVWk4SWk4ZDVrR1RzaTMwTkZ2N2FpOW43UVpQTXdiZHlz
MmVyVTJYTXENClVkeTgrWmNhTm1HaW1FOHlYTjNSVWQzYTE4bkY
wZlVsb3ZaKzBDVHpXcGQyVmorZU9tMWJFeXk2RHg0aTVwVU1HV3Zlb
zUwNnEyMjcNCmR0dVdCSXVmZnI2b1dwVjBGUE5MaG93MTc1MU5tMj
FMdlBIM3JWdFdqZno2NkxmcWw4dFg3RlJsOVlGU1hzbVNzZWI5Y2VPR2
JZazcNCk1OVWNHUGc4WnNiTWU5cmZRVWFhVi9KTVg5c3FkekRDU3Z
wMGtaSG1UWmc5eDdiTEhjTW5UaGIxNmVKK21WZlFxOHlhVVpRTkc2N
GkNClhaKzAva3E2dU9aRk8wUXRhdGRXS2ZYblJROTlCajkxUjVPSUZuaz
U0ak4wbWtVaXFsTzNYRFcrTWwrOThtS0I2dFc3cldwWmNQYysNCjB6Zz
R0THJZbFVjODZFNmVHRGpJTXViVnBjdXNlYXJmZ0lZR1JrNmJyaFpWc
i9KY0h6b29MNzU1MGplZExFeG9wV2NBcGkyWlVxaHUNCjdKTHZyVnN
RVTgxemt6T1BlZW1NUll2VnVRc1g3UGJpRFFZNUp2Wm9uZnRLKzFWW
ThIOXV0eDUzMGgwb2Iram1SWXFqNm91YVl2RWUNCm5XL1dsWWpw
OGN3Yk1tNjgydFB3cVcxUjR0ai8yU0gxM0lSSllsNG1vWnZYcGlTcURyN2

RYdFFIeGEvUEszLytCV3NLMWRUZ0h1NlYNCjh0UUozYndGa3dwRnJVT
1E1MHMxcjNsZXZtOHpaY3ExNytCQmF3N0s4bEVLNXF6a1llYXJrOUE4c
DdQM0d6REsrbmQzRFFvdys2VUMNCjhTVk44Mml1djM4aW03TnRhWHR
WMUNWcTZSZ3c0cGtzbWJkaTNidTJEZTdZZmFCQnhjcWZ2cVByVWpG
UU5UUTIybGZkVVZWVDY4clQNCkpLRjVEblNtVWpnZHFnNG1TUzlwb
XNmREpSM0c2VG9IMGlXOWFWN0xXTEhZWEtsbFREdDBMVEF0a1lJY
WFtcDFRalZ2Kyt1eUdVeFYNCmRKMEROVlhTbStiMXFSeHBsODRkZGZ
YMUxwMU8vZDY5dHNvZDB2czVoR3JlOXh1OG8rZnBMUjFjR2hOVEQ2
WjU3QzlLTVdYZWZKZE8NClo5NGJiOW9xZDFST25TN3FJVFR6SGltTXF
pdmJPM2cwRGRWeWszV1FCaEJ6dEszNVlLTmRPbmM4TzNhY1M2ZkRa
RmdLYVhMc0VKcDUNCnJkcmxpQnFwODljSmNzL203VHZzMHJrakdmTj
RiMGtQb1puM1VKdUlPcm5aMjJ5UDFmbXZVeCtPNWdTcWViVjFtK3pTd
VlOVmhxN1QNCldiRGlMVnZsanBsTGxvcDZDTFhQKzJxdHZHTElMLzF2
aW1JU2RNQmd6U29GWnl1NlRxZCtqenhnc1BhVjlCQ3FlZS9OallrNnY2bEs
NCjljd2lVYy9TVHRmMUhEcE0zYjU5Mnk3aDNUaHg1b3pLNjlITHBZV3V
BBd2FxUzVjdjI2cTdjZWI4ZWZWWWFSZVAzaUZVOHpqMWtuU3cNClpYS
E1tbkNqWTBPZ2FsbzdVUWZTQ00zcVFRcjJIL1hGUDdzc1h4NDVZbDkxQ
nllQ2VwNG1vWm9IKzFmRzN4RDR0VDd4OGt3eWo4bncNCmI5ZXYyNlY
wQjZkKzdINHpLdnVkQUg1MzdGanF5ek9IZEpuSEV1em1YcS9XanhPYnZ
OTWJ2N25oeXdzWDJhVnNXdEM4KzQ4YUxlYXANCkU3cDV3S1ppMEE
yQVFSVjVudlI0RSt1SmMrYjYxa0FwcUlueEJnbWQvNFY1UVAvbXQxOEh
EQzdzUkhmdG1ldTVsbWhWMHJuL0FMWDINCjMyYnFkNEJGbkR4N1Zp
MWNXUzJ1ZmYwSWJCNDdxZXh4bVVqOVF1dFlqdXBkM3RZRDZhYldC
Qk1yaCthcE5iT0tyTkYxK3VnQ2E0cmkNClhHZndNUFB0VmlhdmhVM1lNT
0FBbnVVYi9SMDdMMHlPU2VPYWRFODhBcHNYRkdmZjMweW5obEpn
TTUxQ1U2dk45RXpnbnB2SEJGVXkNCmlWcmFlUGl3SjUzREY1WlRabm9
tRU5nODVrTlVkMm9KaTJXcHI0T21ta2ZONHg0ekhmaVZGYzhEdjhOenVo
TnFPaWRpbEd2QTZER3UNCmVad083OEFBUW42Y2lFazYrcnc1VmN2anZ
xTkRZUE9vSVV3YUtTaHJ4QXVYTGxrSDRhWXVHZk1ZRGMxMFdGNV
RhMzFoUEpPZmNVaHINClUvSmxJTmk2YzZlbFJZZEJwbzYrK1lmang2M
WxHTmZSbTRNRDVySjFqM0ZvR0huakRTQk5hcllVZ01MeU1zektwYjd0W
HBvSGZQczgNCmgzV3AxTHpOZk5rNTRYeEMxd0RHVW1ZelhZZWZoNn
ovY0t0Vm00RUJ4YTlWUUdEellyM0xyVU1SakhFS2trN3phRktZUUEyaEdR
VTENCnorODVORldwWERya3ozdngxMEdxeFE2QnplTmJvQms1bjhrNG5l
YlJoK2sxaFdmeFRGMEQxRXlXVXM1bnYrZGdRcUtheHp1Q2RFMGkNCn
NIbDAyTlE4YWgwbVhyMTJMYTNtMGY5d2lrOSt3TE5UTVkvODZNUG8
4eWkzMU9meG1UNlBXb3FHOStEWnVrWW5hNTZtU1p0NVdXU3kNCjVx
VkExcndVeUpxWEFsbnpraWFpL2dIU0Q3UmtUeWlob2dBQUFBQkpSVTV
FcmtKZ2dnPT0iLA0KICAiQXNzZXJ0aW9uU2NoZW1lIjogIlVBRlYxVExW
IiwNCiAgIkF1dGhlbnRpY2F0aW9uQWxnb3JpdGhtIjogMSwNCiAgIkF0dGV
zdGF0aW9uVHlwZXMiOiBbMTYzOTFdLA0KICAiVVBWIjogW1sxLDBdX
Q0KfQ0K

例 3：JWT 报头

{"typ":"JWT",

"alg":"ES256"

"x5t#S256":"7231962210d2933ec993a77b4a7203898ab74cdf974ff02d2de3f1ec7cb9de68"}

为了生成 tbs 有效载荷，首先需要 base64url 编码的 JWT 报头（无填充）。

例 4：经过编码的 JWT 报头

eyJ0eXAiOiJKV1QiLAogImFsZyI6IkVTMjU2IiwKICJ4NXQjUzI1NiI6IjcyMzE5NjIyMTBkMjkzM2VjOTkzYTc3YjRhNzIwMzg5OGFiNzRjZGY5NzRmZjAyZDJkZTNmMWVjN2NiOWRlNjgifQ

然后必须加上句号(".")，对 EncodedMetadataTOCPayload 进行 base64url 编码（按 Metadata TOC Format 部分的举例来进行）。

例 5：tbs 有效载荷

eyJ0eXAiOiJKV1QiLAogImFsZyI6IkVTMjU2IiwKICJ4NXQjUzI1NiI6IjcyMzE5NjIyMTBkMjkzM2VjOTkzYTc3YjRhNzIwMzg5OGFiNzRjZGY5NzRmZjAyZDJkZTNmMWVjN2NiOWRlNjgifQ.eyAibm8iOiAxMjM0LCAibmV4dC11cGRhdGUiOiAiMzEtMDMtMjAxNCIsDQogICJlbnRyaWVzIjogWw0KICAgeyAiYWFpZCI6ICIxMjM0IzU2NzgiLCANCiAgICAgImhhc2giOiAiOTBkYThkYTZkZTIzMjQ4YWJiMzRkYTBkNDg2MWY0YjMwYTc5M2UxOThhOGQ1YmFhN2Y5OGYyNjBkYjcxYWNkNCIsIA0KICAgICAidXJsIjogImh0dHBzOi8vZmlkb2FsbGhbmNlLm9yZy9tZXRhZGF0YS8xMjM0JXgyM2FiY2QiLCANCiAgICAgInN0YXR1cyI6ICJmaWRvQ2VydGlmaWVkIg0KICAgICAidGltZU9mTGFzdFN0YXR1c0NoYW5nZSI6ICIiLA0KICAgICAidXJsIjogImh0dHBzOi8vZmlkb2FsbGhbmNlLm9yZy9tZXRhZGF0YS8xMjM0JXgyMzQzMjEiLA0KICAgICAic3RhdHVzIjogImZpZG9DZXJ0aWZpZWQiDQogICAgICJ0aW1lT2ZMYXN0U3RhdHVzQ2hhbmdlIjogIjIwMTQtMDItMTkiLA0KICAgICAiY2VydGlmaWNhdGlvbkRhdGUiOiAiMjAxNC0wMS0wNyIgfQ0KICBdD

Qp9DQo

最后加上另外一个句号(".")，其后是 base64url 编码后的签名。

例 6：JWT

eyJ0eXAiOiJKV1QiLAogImFsZyI6IkVTMjU2IiwKICJ4NXQjUzI1NiI6IjcyMzE5NjIyMTBkMjkzM2VjOTkzYTc3YjRhNzIwMzg5OGFiNzRjZGY5NzRmZjAyZDJkZTNmMWVjN2NiOWRlNjgifQ.eyAibm8iOiAxMjM0LCAibmV4dC11cGRhdGUiOiAiMzEtMDMtMjAxNCIsDQogICJlbnRyaWVzIjogWw0KICAgeyAiYWFpZCI6ICIxMjM0IzU2NzgiLCANCiAgICAgImhhc2giOiAiOTBkYThkYTZkZTIzMjQ4YWJiMzRkYTBkNDg2MWY0YjMwYTc5M2UxOThhOGQ1YmFhN2Y5OGYyNjBkYjcxYWNkNCIsIA0KICAgICAidXJsIjogImh0dHBzOi8vZmlkb2FsbGlhbmNlLm9yZy9tZXRhZGF0YS8xMjM0JXgyM2FiY2QiLCANCiAgICAgInN0YXR1cyI6ICJmaWRvQ2VydGlmaWVkIg0KICAgICAidGltZU9mTGFzdFN0YXR1c0NoYW5nZSI6ICIiLA0KICAgICAiY2VydGlmaWNhdGlvbkRhdGUiOiAiMjAxNC0wMS0wNCIgfSwNCiAgICAgImF1dGhlbnRpY2F0b3JEYXRhIjogIjc4NWQxNmRmNjQwZmQ3YjUwZWQxNzRjYjU2NDVjYzBmMWU3MmI3ZjE5Y2YyMjk1OTA1MmRkMjBiOTU0MWM2NGQiLA0KICAgICAidXJsIjogImh0dHBzOi8vYXV0aG5yLXZlbmRvci1hLmNvbS9tZXRhZGF0YS85ODc2JXgyMzQzMjEiLA0KICAgICAic3RhdHVzIjogImZpZG9DZXJ0aWZpZWQiDQogICAgICJ0aW1lT2ZMYXN0U3RhdHVzQ2hhbmdlIjogIjIwMTQtMDItMTkiLA0KICAgICAiY2VydGlmaWNhdGlvbkRhdGUiOiAiMjAxNC0wMS0wNyIgfQ0KICBdDQp9DQo.

AP-qoJ3VPzj7L6lCE1UzHzJYQnszFQ8d2hJz51sPASgyABK5VXOFnAHzBTQRRkgwGqULy6PtTyUVzKxM0HrvoyZq

注释

分行只是为了显示需要。

上面例子中的签名是通过下面的椭圆曲线签名密钥计算得出的。

例 7：用于计算签名的椭圆曲线签名密钥

x: d4166ba8843d1731813f46f1af32174b5c2f6013831fb16f12c9c0b18af3a9b4

y: 861bc2f803a2241f4939bd0d8ecd34e468e42f7fdccd424edb1c3ce7c4dd04e

d: 3744c426764f331f153e182d24f133190b6393cea480a8eec1c722fce161fe2d

3.1.6 元数据 TOC 对象处理规则

FIDO 服务器必须遵守下面的处理规则：

1. FIDO 服务器必须能在合适的时间从熟知的 URL 下载最新的元数据 TOC 对象。Metadata TOC 的 nextUpdate 字段指定了应该下载的最新日期。
2. 如果 x5u属性出现在 JWT 报头中，那么：
    1) FIDO 服务器必须验证 x5u 属性指定的 URL 与用于下载元数据 TOC 的 URL 是同源的。如果不同源，FIDO 服务器应该忽略这个文件（为了防止从恶意网站加载对象）。
    2) FIDO 服务器必须从 x5u 属性指定的 URL 下载证书（链）[JWS]。根据[RFC5280]，证书链必须被验证以正确地链接到元数据 TOC 签名的可信锚，证书链中的所有证书必须检查撤销。
    3) 如果证书链不能通过验证或者如果一个证书链被撤销，FIDO 服务器应该忽略这个文件。
3. 如果 x5u 属性缺失，元数据 TOC 签名可信锚被认为是 TOC 签名证书链。
4. 用 TOC 签名证书链来验证元数据 TOC 对象的签名（正如上一步决定的）。如果签名无效则 FIDO 服务器应该忽略这个文件。如果这个文件的序列号(no)小于或等于本地缓存的最后的元数据 TOC 对象的序列号，该文件也应该忽略。
5. 按要求将验证对象写入本地缓存。
6. 遍历各个条目（类型为 MetadataTOCPayloadEntry）。对每个条目：
    1) 如果 AAID 不与依赖方相关（例如不接受任何策略），忽略该条目。

2) 从 url 字段指定的 URL 下载元数据声明。认证器供应商可能需要通过鉴别来提供对数据的访问。合格的 FIDO 服务器应该按[RFC2617]中定义的，支持 HTTP Basic、HTTP Digest 鉴别方案。

3) 通过与本地缓存条目对比 timeOfLastStatusChange 和 statusReport 字段的值，来检查通过 AAID 识别的认证器类型的状态报告是否改变。更新缓存条目的状态。对于状态报告中指出的缺少凭证或有已知安全问题的认证器，由依赖方来规定它们的行为。然而，REVOKED 状态表明与该认证器存在明显的安全问题。

> 注释
>
> 对不合格状态的认证器应该作出相应的标记。在注册请求和鉴别请求中的建立注册和鉴别策略时都需要该信息[UAFProtocol]。

4) 计算从 URL 下载的元数据声明的哈希值（Base64url 编码没有对 UTF-8 编码进行填充），与元数据 TOC 对象中的 hash 字段中的哈希值进行对比验证。如果哈希值不匹配，忽略下载的元数据声明。

5) 根据下载的元数据声明更新缓存的元数据声明。

4. 注意事项

本节是非规范性的。

本节描述了设计元数据服务的关键注意事项。

对认证器元数据的需求

当定义合适的认证器的策略时，用通用方式描述必要的认证器特征比单独列出认证器 AAID 要好。元数据声明提供了这样的信息。认证器元数据提供验证鉴证对象所需的可信锚。

元数据服务提供了访问元数据声明的标准化的方法。

完整性和真实性

元数据声明包括相关的安全信息。一些垂直业务出于审计的目的，可能需要记录认证器策略和用于验证鉴证对象的可信锚。

重要的是要有一种强大的方法来验证和证明元数据声明的真实性、完整性和新鲜度。可用单数字签名来保护元数据 TOC 对象的完整性和真实性，同时通过将加密哈希值列入到元数据 TOC 对象的方法来保护单个元数据声明的完整性和真实性。这种方法可以促使元数据声明和元数据 TOC 对象通过使用标准内容分发网络更加灵活地分发。

组织影响

认证器供应商能够委托元数据服务完整地发布元数据声明。即使认证器供应商选择自己发布元数据声明也无需过多工作，因为元数据声明可以像普通的文档一样在网站发布。FIDO 联盟对 FIDO 认证过程有控制权，并将接收元数据作为认证过程的一部分。在元数据服务中，已知的认证器列表需要定期更新、签名和发布。需要生成单个签名来保护元数据 TOC 对象的完整性和真实性。

性能影响

元数据 TOC 对象和元数据声明可以通过 FIDO 服务器进行缓存。

更新策略可以由依赖方指定。

元数据 TOC 对象包括下次更新的日期。在 FIDO 鉴别或 FIDO 注册操作过程中，对 FIDO 服务器*没有额外的影响*。

更新元数据 TOC 对象和元数据声明能够异步执行。这减少了元数据服务的可用性要求和 FIDO 服务器的负载。

由于不包含个别元数据声明，元数据 TOC 对象本身相对来说是比较小的，所以下载元数据 TOC 对象不会产生过多的数据流量。

与元数据 TOC 对象相比，单个的元数据声明的更新频率较低。只有修改过的元数据声明需要通过 FIDO 服务器下载。

非公开的元数据声明

一些认证器供应商可能只想对其认购客户提供元数据声明的访问权限。

这些认证器供应商可以在有访问保护的 URL 上发布元数据声明。元数据 TOC 对象中包括访问 URL 和元数据声明的加密哈希值。

高安全性环境

一些高安全性环境可能只信任内部权威策略。这种环境下的 FIDO 服务器被限制在只能从专有的信任源使用元数据 TOC 对象。元数据服务是大多数依赖方的基准。

认证器的扩展信息

一些依赖方在接受认证器之前，可能需要知道认证器额外的扩展信息。由于策略配置是由依赖方控制的，所以很可能只接受那些可提供扩展信息并满足要求的认证器。

A. 参考文献

A.1 参考规范

[JWS]

M. Jones *JSON Web Signature (JWS)*. Internet-Draft (Work in progress.)

URL: http://tools.ietf.org/html/draft-ietf-jose-json-web-signature

[JWT]

M. Jones; J. Bradley; N. Sakimura. *JSON Web Token (JWT)*. 6 July 2012. Internet Draft.

URL: http://tools.ietf.org/html/draft-ietf-oauth-json-web-token-01

[RFC4648]

S. Josefsson, *The Base16, Base32, and Base64 Data Encodings (RFC 4648)*, IETF, October 2006, URL: http://www.ietf.org/rfc/rfc4648.txt

[RFC5280]

D. Cooper, S. Santesson, s. Farrell, S.Boeyen, R. Housley, W. Polk; *Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile*, IETF, May 2008, URL: http://www.ietf.org/rfc/rfc5280.txt

[UAFAuthnrMetadata]

B. Hill, D. Baghdasaryan, J. Kemp, *FIDO UAF Authenticator Metadata Statements v1.0*. FIDO Alliance Proposed Standard. URLs:

HTML: fido-uaf-authnr-metadata-v1.0-ps-20141208.html

PDF: fido-uaf-authnr-metadata-v1.0-ps-20141208.pdf

[WebIDL-ED]

Cameron McCormack, *Web IDL*, W3C. Editor's Draft 13 November 2014. URL: http://heycam.github.io/webidl/

A.2 参考资料

[FIDOGlossary]

R. Lindemann, D. Baghdasaryan, B. Hill, J. Hodges, *FIDO Technical Glossary*. FIDO Alliance Proposed Standard. URLs:

HTML: fido-glossary-v1.0-ps-20141208.html

PDF: fido-glossary-v1.0-ps-20141208.pdf

[ITU-X690-2008]

X.690: Information technology - ASN.1 encoding rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER), (T-REC-X.690-200811). International Telecommunications Union, November 2008

URL: http://www.itu.int/rec/T-REC-X.690-200811-I/en

[RFC2119]

S. Bradner. *Key words for use in RFCs to Indicate Requirement Levels.* March 1997. Best Current Practice. URL: https://tools.ietf.org/html/rfc2119

[RFC2617]

J. Franks; P. Hallam-Baker; J. Hostetler; S. Lawrence; P. Leach; A. Luotonen; L. Stewart. *HTTP Authentication: Basic and Digest Access Authentication*. June 1999. Draft Standard.

URL: https://tools.ietf.org/html/rfc2617

[RFC3986]

T. Berners-Lee; R. Fielding; L. Masinter. *Uniform Resource Identifier (URI): Generic Syntax*. January 2005. Internet Standard.

URL:https://tools.ietf.org/html/rfc3986

[UAFProtocol]

R. Lindemann, D. Baghdasaryan, E. Tiffany, D. Balfanz, B. Hill, J. Hodges, *FIDO UAF Protocol Specification v1.0*. FIDO Alliance Proposed Standard. URLs:

HTML: fido-uaf-protocol-v1.0-ps-20141208.html

PDF: fido-uaf-protocol-v1.0-ps-20141208.pdf

[WebIDL]

Cameron McCormack. *Web IDL*. 19 April 2012. W3C Candidate Recommendation. URL: http://www.w3.org/TR/WebIDL/