Workgroup: ACME Working Group
Internet-Draft: draft-acme-device-attest-02
Published: 22 February 2024
Intended Status: Standards Track
Expires: 25 August 2024
Authors: B. Weeks
Google

Automated Certificate Management Environment (ACME) Device Attestation Extension

Abstract

This document specifies new identifiers and a challenge for the Automated Certificate Management Environment (ACME) protocol which allows validating the identity of a device using attestation.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on 25 August 2024.

Table of Contents

1. Introduction

The Automatic Certificate Management Environment (ACME) [RFC8555] standard specifies methods for validating control over identifiers, such as domain names. It is also useful to be able to validate properties of the device requesting the certificate, such as the identity of the device /and whether the certificate key is protected by a secure cryptoprocessor.

Many operating systems and device vendors offer functionality enabling a device to generate a cryptographic attestation of their identity, such as:

* Android Key Attestation

* Chrome OS Verified Access

* Trusted Platform Module

Using ACME and device attestation to issue client certificates for enterprise PKI is anticipated to be the most common use case. The following variances to the ACME specification are described in this document:

* Addition of permanent-identifier [RFC4043] and hardware-module [RFC4108] identifier types.

* Addition of the device-attest-01 challenge type to prove control of the permanent-identifier and hardware-module identifier types.

* The challenge response payload contains a serialized WebAuthn attestation statement format instead of an empty JSON object ({}).

* Accounts and external account binding being used as a mechanism to pre-authenticate requests to an enterprise CA.

This document does not specify the attestation verification procedures. Section 13 of [WebAuthn] gives some guidance, however verification procedures are complex and may require changes to address future security issues.

2. Conventions and Definitions

The key words "**MUST**", "**MUST NOT**", "**REQUIRED**", "**SHALL**", "**SHALL NOT**", "**SHOULD**", "**SHOULD NOT**", "**RECOMMENDED**", "**NOT RECOMMENDED**", "**MAY**", and "**OPTIONAL**" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

3. Permanent Identifier

A new identifier type, "permanent-identifier" is introduced to represent the identity of a device assigned by the manufacturer, typically a serial number. The name of this identifier type was chosen to align with [RFC4043], it does not prescribe the lifetime of the identifier, which is at the discretion of the Assigner Authority.

The identity along with the assigning organization can be included in the Subject Alternate Name Extension using the PermanentIdentifier form described in [RFC4043].

Clients **MAY** include this identifier in the certificate signing request (CSR). Alternatively if the server wishes to only issue privacy-preserving certificates, it **MAY** reject CSRs containing a PermanentIdentifier in the subjectAltName extension.

4. Hardware Module

A new identifier type, "hardware-module" is introduced to represent the identity of the secure cryptoprocessor that generated the certificate key.

If the server includes HardwareModule in the subjectAltName extension the CA **MUST** verify that the certificate key was generated

on the secure cryptoprocessor with the asserted identity and type. The key **MUST NOT** be able to be exported from the cryptoprocessor.

If the server wishes to issue privacy-preserving certificates, it **MAY** omit HardwareModule from the subjectAltName extension.

5. Device Attestation Challenge

The client can prove control over a permanent identifier of a device by providing an attestation statement containing the identifier of the device.

The device-attest-01 ACME challenge object has the following format:

type (required, string): The string "device-attest-01".

token (required, string): A random value that uniquely identifies the challenge. This value **MUST** have at least 128 bits of entropy. It **MUST NOT** contain any characters outside the base64url alphabet, including padding characters ("="). See [RFC4086] for additional information on randomness requirements.

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

A client fulfills this challenge by constructing a key authorization (Section 8.1 of [RFC8555]) from the "token" value provided in the challenge and the client's account key. The client then generates a WebAuthn attestation object using the key authorization as the challenge.

This specification borrows the WebAuthn *attestation object* representation as described in Section 6.5.4 of [WebAuthn] for encapsulating attestation formats, but with these modifications:

- The key authorization is used to form *attToBeSigned*. This replaces the concatenation of *authenticatorData* and *clientDataHash*. *attToBeSigned* is hashed using an algorithm specified by the attestation format.

- The *authData* field is unused and **SHOULD** be omitted.

A client responds with the response object containing the WebAuthn attestation object in the "attObj" field to acknowledge that the challenge can be validated by the server.

On receiving a response, the server constructs and stores the key authorization from the challenge's "token" value and the current client account key.

To validate a device attestation challenge, the server performs the following steps:

1. Perform the verification procedures described in Section 6 of [WebAuthn].

2. Verify that key authorization conveyed by *attToBeSigned* matches the key authorization stored by the server.

```
POST /acme/chall/Rg5dV14Gh1Q
Host: example.com
Content-Type: application/jose+json

{
  "protected": base64url({
    "alg": "ES256",
    "kid": "https://example.com/acme/acct/evOfKhNU60wg",
    "nonce": "SS2sSl1PtspvFZ08kNtzKd",
    "url": "https://example.com/acme/chall/Rg5dV14Gh1Q"
  }),
  "payload": base64url({
    "attObj": base64url(/* WebAuthn attestation object */),
  }),
  "signature": "Q1bURgJoEslbD1c5...3pYdSMLio57mQNN4"
}
```

6. Security Considerations

See Section 13 of [WebAuthn] for additional security considerations related to attestation statement formats, including certificate revocation.

Key attestation statements may include a variety of information in addition to the public key being attested. While not described in this document, the server **MAY** use any policy when evaluating this information. This evaluation can result in rejection of a certificate request that features a verifiable key attestation for the public key contained in the request. For example, an attestation statement may indicate use of an unacceptable firmware version.

7. IANA Considerations

7.1. ACME Identifier Types

The "ACME Validation Methods" registry is to be updated to include the following entries:

Label	Reference
permanent-identifier	RFC XXXX
hardware-module	RFC XXXX

Table 1

7.2. ACME Validation Method

The "ACME Validation Methods" registry is to be updated to include the following entry:

Label	Identifier Type	Reference
device-attest-01	permanent-identifier	RFC XXXX

Table 2

7.3. New Error Types

This document adds the following entries to the ACME Error Type registry:

Type	Description	Reference
badAttestationStatement	The attestation statement is unacceptable (e.g. not signed by an attestation authority trusted by the CA)	RFC XXXX

Table 3

7.4. Attestation statement formats

Section 2.1 of [RFC8809] describes registration of new attestation statement format types used when authenticating users via [WebAuthn]. This specification reuses the same format, but, because the context for use is different, a different registry is required. This section defines IANA registries for W3C Web Authentication (WebAuthn) attestation statement format identifiers and extension identifiers used in the context of a certificate request. This specification establishes two registries:

* the "WebAuthn Attestation Statement Format Identifiers for Certificate Request Protocols" registry

* the "WebAuthn Extension Identifiers for Certificate Request Protocols" registry

Any additional processes established by the expert(s) after the publication of this document will be recorded on the registry web page at the discretion of the expert(s), who may differ from the experts associated with the registry established by [RFC8809].

7.4.1. WebAuthn Attestation Statement Format Identifiers for Certificate Request Protocols

WebAuthn attestation statement format identifiers are strings whose semantic, syntactic, and string-matching criteria are specified in the "Attestation Statement Format Identifiers" (https://www.w3.org/TR/2019/REC-webauthn-1-20190304/#sctn-attstn-fmt-ids) section of [WebAuthn], along with the concepts of attestation and attestation statement formats.

Registered attestation statement format identifiers are those that have been added to the registry by following the procedure in Section 7.4.1.1.

Each attestation statement format identifier added to this registry **MUST** be unique amongst the set of registered attestation statement format identifiers.

Registered attestation statement format identifiers **MUST** be a maximum of 32 octets in length and **MUST** consist only of printable ASCII [RFC20] characters, excluding backslash and double quote, i.e., VCHAR as defined in [RFC5234] but without %x22 and %x5c. Attestation statement format identifiers are case sensitive and may not match other registered identifiers in a case-insensitive manner unless the designated experts determine that there is a compelling reason to allow an exception.

7.4.1.1. Registering Attestation Statement Format Identifiers

WebAuthn attestation statement format identifiers are registered using the Specification Required policy (see Section 4.6 of [RFC8126]).

The "WebAuthn Attestation Statement Format Identifiers for Certificate Request Protocols" registry is located at https://www.iana.org/assignments/webauthn_for_certreq. Registration requests can be made by following the instructions located there or by sending an email to the webauthn-for-certreq-reg-review@ietf.org mailing list.

Registration requests consist of at least the following information:

* WebAuthn Attestation Statement Format Identifier:
  - An identifier meeting the requirements given in Section 7.4.1.
* Description:
  - A relatively short description of the attestation format.

*Specification Document(s):

  -Reference to the document or documents that specify the attestation statement format.

*Change Controller:

  -For Standards Track RFCs, list "IETF". For others, give the name of the responsible party. Other details (e.g., postal address, email address, home page URI) may also be included.

*Notes:

  -[optional]

Registrations **MUST** reference a freely available, stable specification, e.g., as described in Section 4.6 of [RFC8126]. This specification **MUST** include security and privacy considerations relevant to the attestation statement format.

Note that WebAuthn attestation statement format identifiers can be registered by third parties (including the expert(s) themselves), if the expert(s) determines that an unregistered attestation statement format is widely deployed and not likely to be registered in a timely manner otherwise. Such registrations still are subject to the requirements defined, including the need to reference a specification.

7.4.1.2. Registration Request Processing

As noted in Section 7.4.1.1, WebAuthn attestation statement format identifiers are registered using the Specification Required policy.

The expert(s) will clearly identify any issues that cause a registration to be refused, such as an incompletely specified attestation format.

When a request is approved, the expert(s) will inform IANA, and the registration will be processed. The IESG is the arbiter of any objection.

7.4.1.3. Initial Values in the WebAuthn Attestation Statement Format Identifiers for Certificate Request Protocols Registry

The initial values for the "WebAuthn Attestation Statement Format Identifiers for Certificate Request Protocols" registry have been populated with the values listed in the "WebAuthn Attestation Statement Format Identifier Registrations" (https://www.w3.org/TR/2019/REC-webauthn-1-20190304/#sctn-att-fmt-reg) section of

[WebAuthn]. Also, the Change Controller entry for each of those registrations is:

*Change Controller:

-W3C Web Authentication Working Group (public-webauthn@w3.org)

7.4.2. WebAuthn Extension Identifiers for Certificate Request Protocols

WebAuthn extension identifiers are strings whose semantic, syntactic, and string-matching criteria are specified in the "Extension Identifiers" (https://www.w3.org/TR/2019/REC-webauthn-1-20190304/#sctn-extension-id) section of [WebAuthn].

Registered extension identifiers are those that have been added to the registry by following the procedure in Section 7.4.2.1.

Each extension identifier added to this registry **MUST** be unique amongst the set of registered extension identifiers.

Registered extension identifiers **MUST** be a maximum of 32 octets in length and **MUST** consist only of printable ASCII characters, excluding backslash and double quote, i.e., VCHAR as defined in [RFC5234] but without %x22 and %x5c. Extension identifiers are case sensitive and may not match other registered identifiers in a case-insensitive manner unless the designated experts determine that there is a compelling reason to allow an exception.

7.4.2.1. Registering Extension Identifiers

WebAuthn extension identifiers are registered using the Specification Required policy (see Section 4.6 of [RFC8126]).

The "WebAuthn Extension Identifiers" registry is located at https://www.iana.org/assignments/webauthn. Registration requests can be made by following the instructions located there or by sending an email to the webauthn-for-certreq-reg-review@ietf.org mailing list.

Registration requests consist of at least the following information:

*WebAuthn Extension Identifier:

-An identifier meeting the requirements given in Section 7.4.2.

*Description:

-A relatively short description of the extension.

*Specification Document(s):

  -Reference to the document or documents that specify the extension.

*Change Controller:

  -For Standards Track RFCs, list "IETF". For others, give the name of the responsible party. Other details (e.g., postal address, email address, home page URI) may also be included.

*Notes:

  -[optional]

Registrations **MUST** reference a freely available, stable specification, e.g., as described in Section 4.6 of [RFC8126]. This specification **MUST** include security and privacy considerations relevant to the extension.

Note that WebAuthn extensions can be registered by third parties (including the expert(s) themselves), if the expert(s) determines that an unregistered extension is widely deployed and not likely to be registered in a timely manner otherwise. Such registrations still are subject to the requirements defined, including the need to reference a specification.

7.4.2.2. Registration Request Processing

As noted in Section 7.4.2.1, WebAuthn extension identifiers are registered using the Specification Required policy.

The expert(s) will clearly identify any issues that cause a registration to be refused, such as an incompletely specified extension.

When a request is approved, the expert(s) will inform IANA, and the registration will be processed. The IESG is the arbiter of any objection.

7.4.2.3. Initial Values in the WebAuthn Extension Identifiers Registry

The initial values for the "WebAuthn Extension Identifiers" registry have been populated with the values listed in the "WebAuthn Extension Identifier Registrations" https://www.w3.org/TR/2019/REC-webauthn-1-20190304/#sctn-extensions-reg section of [WebAuthn].

Also, the Change Controller entry for each of those registrations is:

*Change Controller:

-W3C Web Authentication Working Group (public-webauthn@w3.org)

8. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/rfc/rfc2119>.

[RFC4043] Pinkas, D. and T. Gindin, "Internet X.509 Public Key Infrastructure Permanent Identifier", RFC 4043, DOI 10.17487/RFC4043, May 2005, <https://www.rfc-editor.org/rfc/rfc4043>.

[RFC4086] Eastlake 3rd, D., Schiller, J., and S. Crocker, "Randomness Requirements for Security", BCP 106, RFC 4086, DOI 10.17487/RFC4086, June 2005, <https://www.rfc-editor.org/rfc/rfc4086>.

[RFC4108] Housley, R., "Using Cryptographic Message Syntax (CMS) to Protect Firmware Packages", RFC 4108, DOI 10.17487/RFC4108, August 2005, <https://www.rfc-editor.org/rfc/rfc4108>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/rfc/rfc8174>.

[RFC8555] Barnes, R., Hoffman-Andrews, J., McCarney, D., and J. Kasten, "Automatic Certificate Management Environment (ACME)", RFC 8555, DOI 10.17487/RFC8555, March 2019, <https://www.rfc-editor.org/rfc/rfc8555>.

[RFC8809] Hodges, J., Mandyam, G., and M. Jones, "Registries for Web Authentication (WebAuthn)", RFC 8809, DOI 10.17487/RFC8809, August 2020, <https://www.rfc-editor.org/rfc/rfc8809>.

[WebAuthn] Hodges, J., Jones, J., Jones, M. B., Kumar, A., and E. Lundberg, "Web Authentication: An API for accessing Public Key Credentials Level 2", April 2021, <https://www.w3.org/TR/webauthn-2/>.

Appendix A. Enterprise PKI

ACME was originally envisioned for issuing certificates in the Web PKI, however this extension will primarily be useful in enterprise PKI. The subsection below covers some operational considerations for an ACME-based enterprise CA.

A.1. External Account Binding

An enterprise CA likely only wants to receive requests from authorized devices. It is **RECOMMENDED** that the server require a value for the "externalAccountBinding" field to be present in "newAccount" requests.

If an enterprise CA desires to limit the number of certificates that can be requested with a given account, including limiting an account to a single certificate. After the desired number of certificates have been issued to an account, the server **MAY** revoke the account as described in Section 7.1.2 of [RFC8555].

Acknowledgments

TODO acknowledge.

Author's Address

Brandon Weeks
Google

Email: bweeks@google.com