ACME Working Group
Internet-Draft
Intended status: Standards Track
Expires: 27 September 2026

B. Weeks

G. Mallaya

S. Rajala

C. Bonnell

26 March 2026

Automated Certificate Management Environment (ACME) Device Attestation Extension
draft-ietf-acme-device-attest-02

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 27 September 2026.

Copyright Notice

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 (https://source.android.com/security/keystore/attestation)

* Chrome OS Verified Access (https://developers.google.com/chrome/verified-access/overview)

* Trusted Platform Module (https://trustedcomputinggroup.org/resource/trusted-platform-module-tpm-summary/)

* Managed Device Attestation for Apple Devices (https://support.apple.com/en-om/guide/deployment/dep28afbde6a/web)

Using ACME and device attestation to issue client certificates for enterprise PKI is to be a 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.

Efforts are underway within the Remote ATtestation ProcedureS (RATS) working group to define a set of standard formats and protocols for attestation. An explicit aim of this document is to support vendor specific formats and protocols that are widely deployed at publication time of this specification.

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. Additionally, the assigner of the identifier MAY also be specified. The name of this identifier type was chosen to align with [RFC4043]. This specification does not prescribe the lifetime of the identifier, which is at the discretion of the Assigner Authority.

Although [RFC4043] permits any valid UTF-8 string to be used as the identifier, this specification mandates that identifiers MUST NOT contain the forward-slash "/" (UTF-8: U+002F) character. This restriction is required to make the ABNF production rule for the permanent-identifier-value unambiguous.

3.1. Representation in Order resources

The identifier's value field contains a UTF-8 string representation of the identity of the device. In addition to the value being a valid UTF-8 string, the value MUST match the permanent-identifier-value production rule as defined in this ABNF [RFC5234] syntax:

```
''' assigner-value = ("0" / "1" / "2") 1_("." 1_DIGIT) device-identifier-value = 1*(%x00-2E / %x30-FF)

permanent-identifier-value = device-identifier-value ["/" assigner-value] '''
```

A valid permanent-identifier-value value is a UTF-8 string that contains an identity consisting of one or more characters without any forward-slash "/" (UTF-8: U+002F) characters. Optionally, a forward-slash "/" character and "dotted-decimal" object identifier identifying the assigner may follow the identity.

Example identifier without an assigner:

```
{ "type": "permanent-identifier", "value": "ABCDEF123456" }
```

Example identifier with an assigner:

```
{ "type": "permanent-identifier", "value": "ABCDEF123456/1.2.3.4" }
```

3.2. Representation in Certificate Signing Requests and X.509 Certificates

The identity is included in the Subject Alternative Name Extension using the identifierValue field of the PermanentIdentifier form described in [RFC4043]. Although [RFC4043] permits the requester to include the identifierValue in a serialNumber subject attribute, this specification mandates that the identifierValue field of the PermanentIdentifier MUST be present and MUST contain the identifier.

The value of the identifierValue field of the PermanentIdentifier MUST be an octet-for-octet match of the device-identifier-value value as encoded in the Order resource. If the assigner-value value is included in the identifier as encoded in the Order resource, then the assigner field of the PermanentIdentifier MUST be the encoding of the "dotted-decimal" object identifier encoded as the assigner-value value.

To ensure that the identifier as presented in the Order resource and CSR match, the Server MUST perform the logical equivalent of extracting the device-identifier-value and assigner-value values from the CSR and reconstructing the UTF-8 representation of the identifier. The Server MUST then ensure that the UTF-8 representation and the identifier presented in the Order resource are an octet-for-octet match and reject the Order otherwise.

[RFC8555] section 7.4 mandates that "The CSR MUST indicate the exact same set of requested identifiers as the initial newOrder request". However, there are some environments where the Server requires validation of the identifier but does not include the identifier in certificates due to privacy concerns. To support privacy-preserving certificates, Clients MAY omit this identifier in the certificate signing request (CSR). Similarly, if the Server wishes to 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 crypto-processor that generated the certificate key. The identity is modeled after the HardwareModuleName form described in [RFC4108]. It consists of two components: an OBJECT IDENTIFIER to represent the type of hardware module, and a serial number that identifies the specific hardware module.

Although [RFC4108] specifies that serial numbers can be represented as any sequence of bytes, this specification requires that serial numbers MUST be representable as valid UTF-8 strings consisting of at least one code point and MUST NOT contain a forward-slash "/" (UTF-8: U+002F) character. These restriction ensures that serial numbers can be included in hardware-module identifier string values and that the ABNF production rule for the value is unambiguous.

4.1. Representation in Order resources

The identifier's value field contains a UTF-8 string representation of the identity of the hardware module. In addition to the value being a valid UTF-8 string, the value MUST match the hardware-module-value production rule as defined in this ABNF [RFC5234] syntax:

''' hw-type-value = ("0" / "1" / "2") 1_("." 1_DIGIT) hw-serial-num-value = 1*(%x00-2E / %x30-FF)

hardware-module-value = hw-serial-num-value ["/" hw-type-value] '''

A valid hardware-module-value value is a UTF-8 string that contains a serial number consisting of one or more characters without any forward-slash "/" (UTF-8: U+002F) characters. Optionally, a forward-slash "/" character and "dotted-decimal" object identifier identifying the hardware type may follow the serial number.

Example identifier with the type of the hardware module represented using the OBJECT IDENTIFIER "1.2.3.4" and a serial number of "ABCD":

{ "type": "hardware-module", "value": "ABCD/1.2.3.4" }

Example identifier with no type specified and a serial number of "ABCD":

{ "type": "hardware-module", "value": "ABCD" }

4.2. Representation in Certificate Signing Requests and X.509 Certificates

The hardware module identity is included in the Subject Alternate Name Extension using the HardwareModuleName form described in [RFC4108]. The HardwareModuleName is encoded as an otherName with the OID id-on-hardwareModuleName (1.3.6.1.5.5.7.8.4) and consists of:

* hwType: An OBJECT IDENTIFIER that identifies the type of hardware module

* hwSerialNum: An OCTET STRING containing the hardware module serial number

The value of the hwSerialNum field of the HardwareModuleName MUST be an octet-for-octet match of the hw-serial-num-value value as encoded in the Order resource. If the hw-type-value value is included in the identifier as encoded in the Order resource, then the hwType field of the HardwareModuleName MUST be the encoding of the "dotted-decimal" object identifier encoded as the hw-type-value value.

To ensure that the identifier as presented in the Order resource and CSR match, the Server MUST perform the logical equivalent of extracting the hw-serial-num-value and hw-type-value values from the CSR and reconstructing the UTF-8 representation of the identifier. The Server MUST then ensure that the UTF-8 representation and the identifier presented in the Order resource are an octet-for-octet match and reject the Order otherwise.

[RFC8555] section 7.4 mandates that "The CSR MUST indicate the exact same set of requested identifiers as the initial newOrder request". However, there are some environments where the Server requires validation of the identifier but does not include the identifier in certificates due to privacy concerns. To support privacy-preserving certificates, Clients MAY omit this identifier in the certificate signing request (CSR). Similarly, if the Server wishes to issue privacy-preserving certificates, it MAY reject CSRs containing a HardwareModuleName in 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.

```
{
  "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"
}
```

The webauthn payload MAY contain any identifiers registered in "WebAuthn Attestation Statement Format Identifiers" and any extensions registered in "WebAuthn Extension Identifiers" [IANA-Webauthn], [RFC8809].

6. Operational Considerations

Although this document focuses guidance on implementing new type and challenge for certificate issuance using ACME, it does not define a New Protocol, a Protocol Extension, or an architecture.

6.1. 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.

6.1.1. 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].

7. Security Considerations

Please reference [RFC8555] for other 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.

The "token" value MUST have at least 128 bits of entropy. It MUST NOT contain any characters outside the base64url alphabet, including padding characters ("="). See [I-D.ietf-tls-rfc8446bis], Appendix C.1 for additional information on randomness requirements.

8. IANA Considerations

8.1. ACME Identifier Types

The "ACME Identifier Types" registry is to be updated to include the following entries:

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

Table 1

8.2. ACME Validation Method

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

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

Table 2

8.3. 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

9. References

9.1. Normative References

[I-D.ietf-tls-rfc8446bis]
Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", Work in Progress, Internet-Draft, draft-ietf-tls-rfc8446bis-14, 13 September 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-tls-rfc8446bis-14>.

[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>.

[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>.

[RFC5234] Crocker, D., Ed. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, DOI 10.17487/RFC5234, January 2008, <https://www.rfc-editor.org/rfc/rfc5234>.

[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/>.

9.2. Informative References

[IANA-Webauthn] "IANA Registries for Web Authentication (WebAuthn)", n.d., <https://www.iana.org/assignments/webauthn/webauthn.xhtml>.

Acknowledgments

We thank the participants on the ACME Working Group mailing list for their insightful feedback and comments. In particular, the authors extend sincere appreciation to Mike Ounsworth, Deb Cooley, Aaron Gable, Richard Barnes, and Herman Slatman for their reviews and suggestions, which greatly improved the quality of this document.

Authors' Addresses

Brandon Weeks
Email: me@brandonweeks.com

Ganesh Mallaya
Email: ganesh.mallaya@appviewx.com

Sven Rajala
Email: sven.rajala@keyfactor.com

Corey Bonnell
Email: corey.bonnell@digicert.com