IETF
Internet-Draft
Intended status: Standards Track
Expires: 12 February 2026

K. Moriarty
SecurityBiaS

11 August 2025

ACME End User Client and Code Signing Certificates
draft-ietf-acme-client-14

Abstract

Automated Certificate Management Environment (ACME) core protocol addresses the use case of web server certificates for TLS. This document extends the ACME protocol to add 3 challenge types that may support service account authentication credentials, micro-service accounts credentials, device client, code signing, document signing certificates and keys.

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 12 February 2026.

Copyright Notice

Table of Contents

1. Introduction

ACME [RFC8555] is a mechanism for automating certificate management on the Internet. It enables administrative entities to prove effective control over resources like domain names, and automates the process of generating and issuing certificates.

The core ACME protocol defined challenge types specific to web server certificates with the possibility to create extensions, or additional challenge types for other use cases and certificate types. Client certificates, such as end user, device (including IoT), signature, and service authentication also benefit from automated management to ease the deployment and maintenance of these certificate types, thus the definition of this extension defining challenge types for end users and service accounts (e.g. cloud native containers, microservices). Use cases for digital signatures are increasingly becoming foundational for integrity protection, origin authentication and data provenance including the following that may benefit from automating the certificate and key management for functions such as code signing, document signing (e.g. PDF), and format signing (e.g. JWT, SPDX, XML).

2. End User Client Certificates

A client certificate used to authenticate an end user or (IoT) device may be used for mutual authentication in TLS, EAP-TLS, or messaging. The client certificate and key in this case may be stored in a browser, PKCS-#11 container, Key Management Interoperability Protocol (KMIP), or another key store. To obtain an end user client or microservice or container certificate and associated key pair, there are several possibilities to automate authentication of an identity credential intended to be tied to an end user, device, service, application, or container.

This draft adds challenge types to ACME for this purpose and includes:

- Public Key Infrastructure, x.509,
- FIDO Credentials/WebAuthn/Passkeys, and
- One Time Passwords

Existing credentials, for instance, FIDO or W3C WebAuthn may be used to automate the challenge required to obtain a certificate and key pair through ACME for use cases such as client certificates, code signing certificates, document signing certificates, and interactions between microservices in cloud native environments. User interaction may be required when these credentials are used to respond to a challenge and that requirement may be appropriate for the certificate type requested, yet still easing the certificate management burden.

FIDO uses a public and private key pair and does not perform identity proofing, identity proofing is a separate function and its requirement depends on the application and policy. FIDO authentication provides a different key pair to each authenticating entity and service pair using FIDO for authentication, which are generated at the client and registered by the server. This may require using the FIDO or WebAuthn credentials from a specific service for authentication to gain ACME issued certificates and keys. The key pairs for FIDO, WebAuthn, or Passkeys may be stored in a system-based or hardware-based (e.g. FIDO approved hardware token, hardware security module) key store.

One-time password (OTP) authentication is also an option providing multiple factors of authentication, designed with a human entering the OTP code manually. In cases where a higher assurance level is needed, OTP may be a good choice and many options exist today for OTP that could use an app on a phone for instance tied to an existing (or newly established) password. The OTP may be tied to an out-of-band

process and may be associated with a username/password and other accounts. It is possible that the OTP response is automated, but that eliminates the second factor if two-factor authentication was desired.

PKI may be fully automated to serve as the credential type in the challenge response for ACME.

One consideration is to understand if the use case could use FIDO credentials, WebAuthn, or Passkeys directly and not require an additional or new set of PKI keys for the service or end user authentication, meaning ACME client certificates and keys. FIDO Credentials provide a mechanism to have unique public key pair based access for client authentication to web sites and they are working on non-web. Identity proofing is intentionally decoupled from authentication in this model as that is in line with NIST 800-63r3 recommendations for privacy protections of the user. The credential in this case is authenticated and would be consistent for its use, but the identity proofing for that credential is not performed. Obviously, identity proofing is more important for some services, like financial applications where tying the user to the identity for access to financial information is important.

It is important to keep in mind that the challenge types defined may be used to have a human in the loop in order to establish PKI certificate and key pairs that may then be used for automated authentication. The issued certificate and key pairs can be stored in system based key stores or external hardware based key stores using PKCS 11.

Three methods for ACME client authentication, not identity proofing, are proposed in the Challenge Type Section.

3. Pre-authorization

Additional challenge types are defined here for the verification of administrators at an organization requesting certificates with the extended key usage (EKU) for CodeSigning or Document Signing. Email is listed as possible in RFC8555 and may be used singularly or in combination as the ACME protocol allows for multiple pre-authorization challenges to be issued. Additional pre-authorization types are defined that provide a higher level of assurance to authorize a request.

4. Challenge Types

The challenge types defined in the following subsections are to authenticate individuals or holders of specific pre-issued credentials (users acting in roles for an organization). The challenge types can be used to obtain end user certificate types or as a pre-authorization challenge with certificate types such as digital signature with the EKU set for code signing or document signing. Please note that the pre-authorization challenge is also coupled with the account certificate in ACME for verification. The process for obtaining EV Code Signing Certificates typically requires authorization from one or more individuals in a role for the organization. The use of pre-issued secure credentials, at an assurance level appropriate for the certificate type being issued, provides a way to automate the issuance and renewal process.

4.1. One Time Password (OTP)

There are numerous one-time password technologies with slight variations between implementations. The response to the challenge is entered in the provided URL, offering flexibility to those using this challenge type to accommodate the specific requirements of their solution. Looking at 2 OTP solutions, the challenge response is provided via a tool or app without any user interaction of information required from the server to generate the challenge. The 2 solutions that operate in this manner include SecureID and Duo Security. If a challenge is required to generate the response to be provided in the URL, the token can supply the challenge.

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

token (required, string): A random value that uniquely identifies the challenge. OTP types and input vary between technologies. The token value will match the type expected for the pre-issued OTP credential. The user will be able to supply a response in the provided URL from this challenge. It MUST NOT contain any characters outside the base64url alphabet and MUST NOT include base64 padding characters ("=").

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

4.1.1. HMAC-Based One-Time Password (HOTP)

HOTP([RFC4226]) describes an algorithm for the generation of time-based password values.

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

token (required, string): The HOTP value. This SHOULD be the 6 digit representation.

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

4.1.2. Time-Based One-Time Password (TOTP)

TOTP([RFC6238]) describes an algorithm for the generation of time-based password values, an extension from HOTP.

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

token (required, string): The TOTP value. This SHOULD be the 6 digit representation.

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

4.1.3. Generic One Time Password (OTP)

There are numerous other one-time password technologies with slight variations between implementations. The response to the challenge is entered in the provided URL, offering flexibility to those using this challenge type to accommodate the specific requirements of their solution. Looking at 2 OTP solutions, the challenge response is provided via a tool or app without any user interaction of information required from the server to generate the challenge. The 2 solutions that operate in this manner include SecureID and Duo Security. If a challenge is required to generate the response to be provided in the URL, the token can supply the challenge.

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

token (required, string): A random value that uniquely identifies the challenge. OTP types and input vary between technologies. The token value will match the type expected for the pre-issued OTP credential. The user will be able to supply a response in the provided URL from this challenge. It MUST NOT contain any characters outside the base64url alphabet and MUST NOT include base64 padding characters ("=").

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

4.2. Public Key Cryptography

Certificates may be pre-issued and stored according to assurance level requirements for the purpose of identifying a user’s identity. If a higher assurance level is needed for a user serving in a specific role or for that individual, it is possible for identity proofing to occur in person using identifiers acceptable for the specified process and the private key stored appropriately for the required assurance level. PKCS#11 software or hardware tokens are both possible options. This model assumes that there may be multiple authorized users with different certificates that can be used for the authorization or pre-authentication challenge. As such, the user first provides the digital signature, so the account management can determine if one of the acceptable certificates was used to digitally sign the token.

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

token (required, string): A random value that uniquely identifies the challenge. The token for a certificate authentication challenge includes a value for the recipient to digitally sign using their private key and post to the provided URL. The ACME server then uses the digitally signed content to verify that the challenge was signed using authorized credentials (certificate issued and authorized for this challenge type). It MUST NOT contain any characters outside the base64url alphabet and MUST NOT include base64 padding characters ("=").

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

4.3. WebAuthn or Public/Private Key Pairs

W3C's WebAuthn/Passkeys use raw public/private key pairs that are issued specific to a service. If WebAuthn or public/private key pairs (PPKP) are selected as the challenge type, the account and credential issuance will have to occur prior to use of this challenge type. The WebAuthn or public/private key pair credentials would be specific to the certificate management account and would be created by the client, then registered with the service as occurs with normal WebAuthn registration of credentials. As with normal WebAuthn and public/private key pairs, the token or challenge is digitally signed to prove possession of the private key.

Passkeys need to be pre-registered with the ACME service since they are origin bound.

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

token (required, string): A random value that uniquely identifies the challenge. This challenge will operate much in the same way as the certificate challenge as the operations are largely the same. The user will be able to supply a response in the provided URL from this challenge. It MUST NOT contain any characters outside the base64url alphabet and MUST NOT include base64 padding characters ("=").

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

5. Security Considerations

The additional challenge types may be used for any certificate type as ACME's design allows for that flexibility. Careful consideration is recommended when selecting the appropriate challenge types for certificate and key pairs. In the case of code or document signing certificate and key pairs, policy for an issuing organization may require processes involving challenges to more than one person in the

organization with specific challenge types required. The challenge type may be tied back to specific authentication levels, for instance the authentication levels defined in NIST SP 800-63. The authentication process may also be tied to roles in the organization as opposed to individuals. Policy and application of policy are outside the scope of this document.

Policy considerations for microservices accounts and other automated service accounts may include the required authentication level to be used for the initial challenge type. The policy should also consider what is acceptable for subsequent key renewal and if automation is possible with other PKI or WebAuthn keys for instance. The security levels of the service may dictate the appropriate and required authentication levels for both the initial challenge type and the subsequent challenge types for renewals in advance of certificate and key pair expiration dates.

The challenge types may or may not be tied to identity proofing processes and is to be determined by the certificate authority (CA) policy. The policies may vary based on the CA as well as the type of certificate being issued or the assurance level of the certificate, or other factors that are outside the scope of this document.

As with the TLS certificates defined in the core ACME document <xref target="RFC8555"/>, identity proofing for ACME issued end user client, device client, IoT device client, service accounts, code signing certificates, and document signing certificates is a separate process outside of the automation of ACME. Identity proofing may be an out-of-band process, if needed, and for this draft is likely tied to the credentials used for the defined challenge types.

Identity proofing for these certificate types present some challenges for process automation. NIST SP 800-63 r3 [NIST800-63r3] serves as guidance for identity proofing further detailed in NIST SP 800-63A [NIST800-63A] that may occur prior to the ability to automate certificate management via ACME or may obviate the need for it weighing end user privacy as a higher concern and allowing for credential issuance to be decoupled from identity proofing (IAL1). This is noted as a security and privacy consideration and is not in scope for this document.

6. IANA Considerations

This memo includes no request to IANA, yet.

7. Contributors

Thank you to reviewers and contributors who helped to improve this document. Thank you to Thomas Peterson who added the one-time password types, HOTP and TOTP. Thank you to Tim Hollebeek for your early review and added text specific to EV certificate issuance and one time use code signing certificates. Thank you to Andrei Popov and Deb Cooley for your reviews and suggestions made in -04. Thank you to those who reviewed the CAA text removed in version -05 including: Carl Mehner, Roland Shoemaker, Ben Schwartz, and Ryan Sleevi. Posted WG version. -02 updates authors email address. Updates included in version 11 correspond to comments on list from Dean Saxe and Richard Wang. Version 13 includes updates from Richard Wang on a correction for the extended key usage and Nancy Cam-Winget on adding IoT device. Draft 14 includes the updates as requested during and following the recorded IETF123 session where feedback was provided by Q. Misell to move the CSR options to an appendix and Mike Ounsworth requested moving and reducing the identity proofing text, which is now int he security considerations section and has been reduced.

8. References

8.1. 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/info/rfc2119>.

[RFC4226] M'Raihi, D., Bellare, M., Hoornaert, F., Naccache, D., and O. Ranen, "HOTP: An HMAC-Based One-Time Password Algorithm", RFC 4226, DOI 10.17487/RFC4226, December 2005, <https://www.rfc-editor.org/info/rfc4226>.

[RFC6238] M'Raihi, D., Machani, S., Pei, M., and J. Rydell, "TOTP: Time-Based One-Time Password Algorithm", RFC 6238, DOI 10.17487/RFC6238, May 2011, <https://www.rfc-editor.org/info/rfc6238>.

[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/info/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/info/rfc8555>.

[RFC9336] Ito, T., Okubo, T., and S. Turner, "X.509 Certificate General-Purpose Extended Key Usage (EKU) for Document Signing", RFC 9336, DOI 10.17487/RFC9336, December 2022, <https://www.rfc-editor.org/info/rfc9336>.

[RFC7030] Pritikin, M., Ed., Yee, P., Ed., and D. Harkins, Ed., "Enrollment over Secure Transport", RFC 7030, DOI 10.17487/RFC7030, October 2013, <https://www.rfc-editor.org/info/rfc7030>.

8.2. Informative References

[I-D.ietf-acme-ip]
Shoemaker, R. B., "Automated Certificate Management Environment (ACME) IP Identifier Validation Extension", Work in Progress, Internet-Draft, draft-ietf-acme-ip-08, 1 October 2019, <https://datatracker.ietf.org/doc/html/draft-ietf-acme-ip-08>.

8.3. URL References

[NIST800-63r3]
US National Institute of Standards and Technology, "https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-63-3.pdf".

[NIST800-63A]
US National Institute of Standards and Technology, "https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-63a.pdf".

[NIST800-63B]
US National Institute of Standards and Technology, "https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-63b.pdf".

[NIST800-63C]
US National Institute of Standards and TechnologyCSR, "https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-63c.pdf".

Appendix A. CSR Definitions Independent of Challenge Types

A.1. CodeSigning Certificates

The process to retrieve a code signing certificate is similar to that of a web server certificate, with differences primarily in the CSR request and the resulting certificate properties. The storage and access of a code signing certificate must be protected and is typically done through hardware, a hardware security module (HSM), which likely has a PKCS#11 interface. A code signing certificate may either be a standard one or an extended validation (EV) certificate.

For automation purposes, the process described in this document will follow the standard process and any out-of-band preprocessing can increase the level of the issued certificate if the CA offers such options and has additional identity proofing mechanisms (in-band or out-of-band).

Strict vetting processes are necessary for many code signing certificates to provide a high assurance on the signer. In some cases, issuance of a standard CodeSigning certificate will be appropriate and no additional "challenges" [RFC8555 Section 8] will be necessary. In this case, the standard option could be automated very similarly to web server certificates with the only changes being in the CSR properties. However, this may not apply to all scenarios, such as those requiring EV certificates with the possibility for required out-of-band initial authentication and identity proofing.

EV code signing certificates have a distinct set of requirements from EV web certificates. In particular, they don’t have associated domain names, nor is CAA checking done. The code signing certificate links a public key to an organization, not a domain. CAs may choose different methods to enable the use of ACME for EV code signing certificates. The intent of this work is to provide additional authentication challenge types that may enable their automation process.

Organization validation is required for standard code signing certificates from most issuers. The CSR is used to identify the organization from the included domain name in the request. The resulting certificate, however, instead contains the organization’s name and for EV certificates, other identifying information for the organization. For EV certificates, this could require that the domain is registered with the Certificate Authority provider, listed in CAA [RFC6844], and administrators for the account are named with provided portal access for certificate issuance and management options.

While ACME allows the client to directly establish an account with a CA, an initial out-of-band process for this step may assist with the additional requirements for EV certificates and assurance levels typically required for code signing certificates. For standard certificates, with a recommendation for additional vetting through extended challenge options to enable ACME to establish the account directly. In cases where code signing certificates are used heavily for an organization, having the portal access replaced with ACME authenticated client access with extra challenges for authentication may be an option to automate the functionality.

To improve the vetting process, ACME's optional use of CAA [RFC6844] with the Directory "meta" data "caaIdentities" ([RFC8555] Section 9.7.6) assists with the validation that a CA may have issue certificates for any particular domain and is RECOMMENDED for use with code signing certificates for this additional level of validation checking on issued certificates.

As noted in RFC8555, "the external account binding feature (see Section 7.3.4) can allow an ACME account to use authorizations that have been granted to an external, non-ACME account. This allows ACME to address issuance scenarios that cannot yet be fully automated, such as the issuance of "Extended Validation" certificates."

The ACME challenge object, [RFC8555] Section 7.1.5 is RECOMMENDED for use for Pre-authorization ([RFC8555] Section 7.4.1). Additional challenge types are added to provide higher levels of security for this issuance verification step. The use of OTP, FIDO credentials (public/private key pairs), or validation from a certificate issued at account setup time are defined in Section 8. Pre-Authorization.

ACME provides an option for notification of the operator via email or SMS upon issuance/renewal of a certificate after the domain has been validated as owned by the requestor. This option is RECOMMENDED due to the security considerations of code signing certificates as a way to limit or reduce the possibility of a third-party gaining access to a code signing certificate inappropriately. The development of additional challenge types is included in this document to support this for pre-authorization, which would better match the security considerations for this certificate type.

Since DNS is used to identify the organization in the request, the identifier "type" ([RFC8555]Section 7.4) is set to dns, not requiring any additions to the ACME protocol for this type of certificate. The distinction lies in the CSR, where the values are set to request a CodeSigning certificate for a client certificate. [Question: Is it helpful to define an identifier for the administrator or for the developer to distinguish the certificate type in ACME and not just the CSR?]

KeyUsage (DigitalSignature) and ExtendedKeyUsage (CodeSigning) in the CSR MUST be set to the correct values for the CA to see the request is for a code signing certificate. The Extended Key Usage SHOULD be set to show this is a CodeSigning certificate, using OID "1.3.6.1.5.5.7.3.3". The CN MUST be set to the expected registered domain with the CA account.

An advantage of ACME is the ability to automate rollover to allow for easy management of short expiry times on certificates. The lifetime of CodeSigning certificates is typically a year, and automation could allow for shorter expiry times becoming feasible. However, lifetimes are less of an issue for code signing certificates than other certificate types. However, there is a legitimate case for "one signature per certificate." Automation might be helpful in this case if patches or software updates were frequent or to minimize the knowledge needed for the organization using this method.

Automation of storage to a hardware security module (HSM), which typically requires authentication is intentionally left out-of-scope since standards exist for key storage and this document is limited to adding new challenge types.

A.2. Document Signing

The challenge type may also be used to obtain a document signing certificate. The CSR must be configured with the correct information to obtain this certificate type.

KeyUsage (DigitalSignature) and ExtendedKeyUsage (DocumentSigning) in the CSR MUST be set to the correct values for the CA to see the request is for a Document Signing certificate. The Extended Key Usage SHOULD be set to show this is a client certificate, using OID "1.3.6.1.5.5.7.3.36". The CN MUST be set to the expected registered domain with the CA account. [RFC9336]

A.3. Micro-Services, Service, or Automated Authentication Accounts

For service accounts, including microservices, it may be desirable to establish PKI based certificate and key pairs for use to perform the automated authentication functions necessary. As such, a challenge type paired with ACME can be supported to request the appropriate certificate type by defining the characteristics required in the CSR to be presented to the CA. The challenge type may be completely automated as in the TLS server use case or may require user interaction with the defined challenge types in this document. The challenge types could be used as the challenge to obtain certificates of a defined lifetime, with a certificate update process following the same requirements or presentation of a challenge type that can be fully automated.

The CSR must be set to the appropriate values in order to obtain the desired certificate type and key pairs for use with the service, server, application, or microservice. ACME can be used for issuance of any of the possible certificate and key types by setting the CST values appropriately for the use case.

KeyUsage (DigitalSignature) and ExtendedKeyUsage (serverAuth) in the CSR MUST be set to the correct values for the CA to see the request is for a server authentication certificate and associated key pair. The Extended Key Usage SHOULD be set to show this is a server certificate, using OID "1.3.6.1.5.5.7.3.1". The CN MUST be set to the expected registered domain with the CA account.

If other properties are necessary for a particular use case, the challenge types defined in this document may be used in combination with the values asserted in the CSR to obtain the correct certificate type if supported by the certificate authority and its policies.

Appendix B. Open Issues

Note to RFC Editor: please remove this appendix before publication as an RFC.

Author’s Address

Kathleen M. Moriarty
SecurityBiaS
Arlington,
United States of America
Email: Kathleen.Moriarty.ietf@gmail.com