

Network Working Group
Internet-Draft
Intended status: Informational
Expires: April 26, 2020

O. Friel
R. Barnes
Cisco
R. Shekh-Yusef
Avaya
October 24, 2019

ACME Integrations
draft-friel-acme-integrations-02

Abstract

This document outlines multiple advanced use cases and integrations that ACME facilitates without any modifications or enhancements required to the base ACME specification. The use cases include ACME integration with EST, BRSKI and TEAP.

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 April 26, 2020.

Copyright Notice

Copyright (c) 2019 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of

the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
2. Terminology	3
3. ACME Integration with EST	3
4. ACME Integration with BRSKI	6
5. ACME Integration with BRSKI Default Cloud Registrar	8
6. ACME Integration with TEAP	10
7. ACME Integration with TEAP-BRSKI	13
8. IANA Considerations	16
9. Security Considerations	16
10. Informative References	16
Appendix A. Comments	17
Authors' Addresses	17

1. Introduction

ACME [RFC8555] defines a protocol that a certificate authority (CA) and an applicant can use to automate the process of domain name ownership validation and X.509 (PKIX) certificate issuance. The protocol is rich and flexible and enables multiple use cases that are not immediately obvious from reading the specification. This document explicitly outlines multiple advanced ACME use cases including:

- o ACME integration with EST [RFC7030]
- o ACME integration with BRSKI
[I-D.ietf-anima-bootstrapping-keyinfra]
- o ACME integration with BRSKI Default Cloud Registrar
[I-D.friel-anima-brski-cloud]
- o ACME integration with TEAP [RFC7170]
- o ACME integration with TEAP-BRSKI [I-D.lear-eap-teap-brski]

The integrations with EST, BRSKI (which is based upon EST), and TEAP enable automated certificate enrolment for devices. ACME for subdomains [I-D.friel-acme-subdomains] outlines how ACME can be used by a client to obtain a certificate for a subdomain identifier from a certificate authority where client has fulfilled a challenge against a parent domain but does not need to fulfil a challenge against the explicit subdomain. This is a useful optimisation when ACME is used to issue certificates for large numbers of devices as it reduces the

domain ownership proof traffic (DNS or HTTP) and ACME traffic overhead, but is not a necessary requirement.

2. Terminology

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.

The following terms are used in this document:

- o BRSKI: Bootstrapping Remote Secure Key Infrastructures [I-D.ietf-anima-bootstrapping-keyinfra]
- o CA: Certificate Authority
- o CMC: Certificate Management over CMS
- o CSR: Certificate Signing Request
- o EST: Enrollment over Secure Transport [RFC7030]
- o FQDN: Fully Qualified Domain Name
- o RA: PKI Registration Authority
- o TEAP: Tunneled Extensible Authentication Protocol [RFC7170]

3. ACME Integration with EST

EST [RFC7030] defines a mechanism for clients to enroll with a PKI Registration Authority by sending CMC messages over HTTP. EST section 1 states:

"Architecturally, the EST service is located between a Certification Authority (CA) and a client. It performs several functions traditionally allocated to the Registration Authority (RA) role in a PKI."

EST section 1.1 states that:

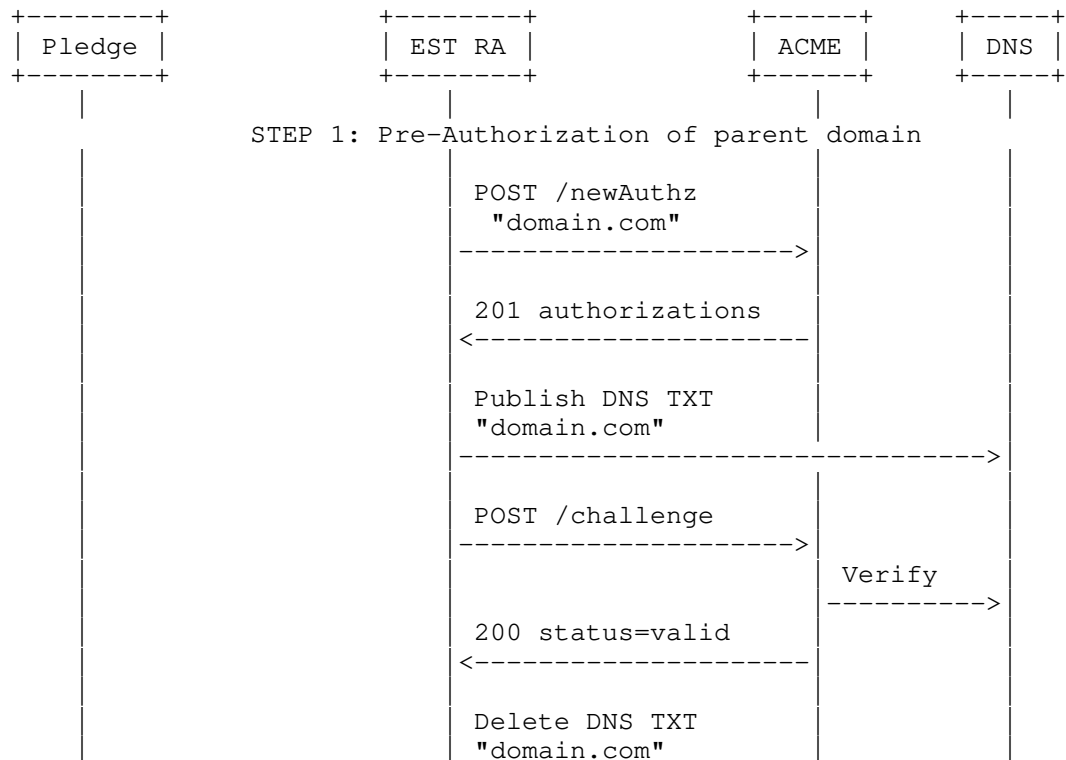
"For certificate issuing services, the EST CA is reached through the EST server; the CA could be logically "behind" the EST server or embedded within it."

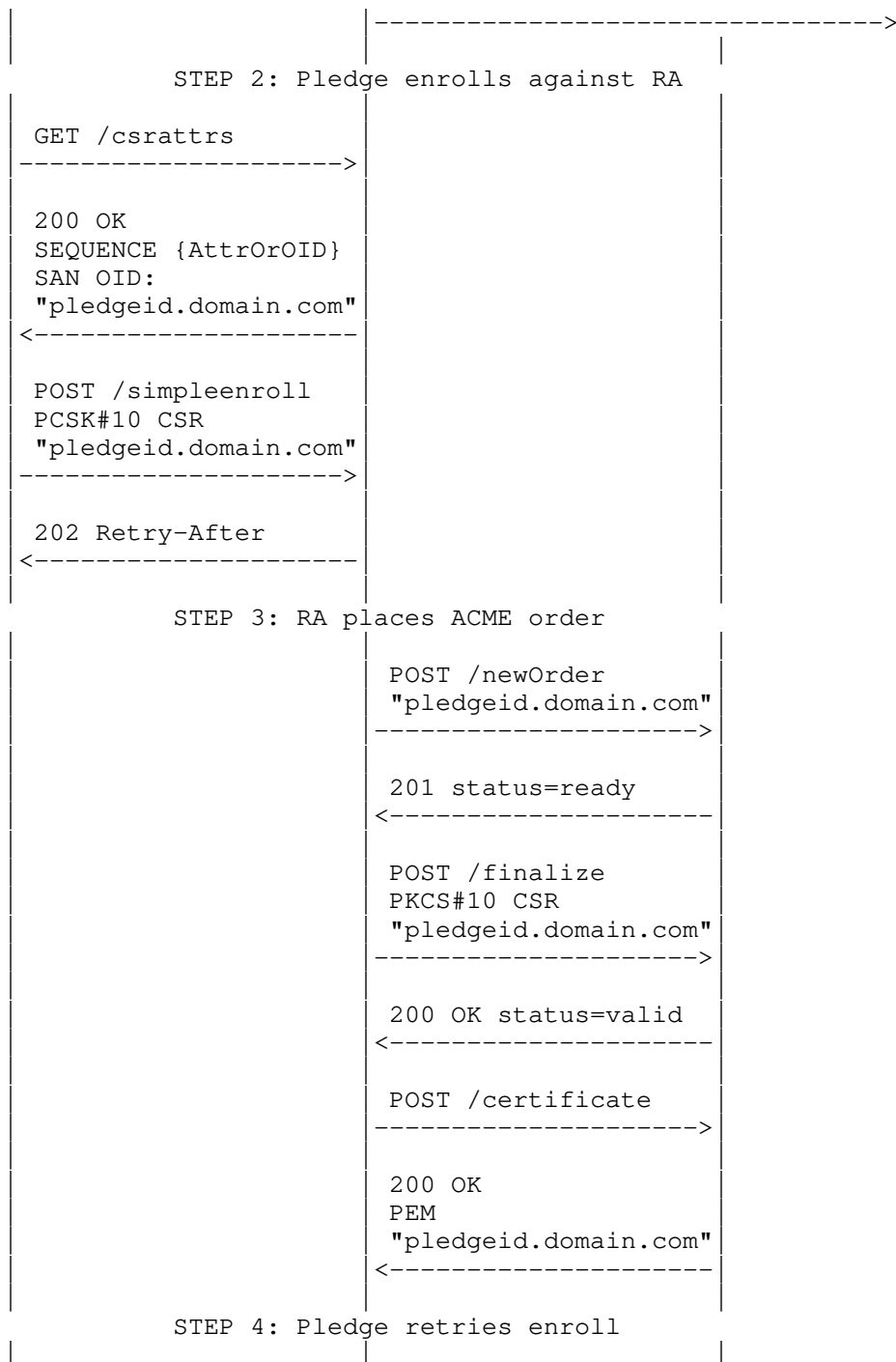
When the CA is logically "behind" the EST RA, EST does not specify how the RA communicates with the CA. EST section 1 states:

"The nature of communication between an EST server and a CA is not described in this document."

This section outlines how ACME could be used for communication between the EST RA and the CA. The example call flow leverages [I-D.friel-acme-subdomains] and shows the RA proving ownership of a parent domain, with individual client certificates being subdomains under that parent domain. This is an optimisation that reduces DNS and ACME traffic overhead. The RA could of course prove ownership of every explicit client certificate identifier.

The call flow illustrates the client calling the EST /csrattrs API before calling the EST /simpleenroll API. This enables the EST server to indicate to the client what attributes it expects the client to include in the CSR request send in the /simpleenroll API. For example, EST servers could use this mechanism to tell the client what fields to include in the CSR Subject and Subject Alternative Name fields.





```

POST /simpleenroll
PCSK#10 CSR
"pledgeid.domain.com"
----->

200 OK
PKCS#7
"pledgeid.domain.com"
<-----

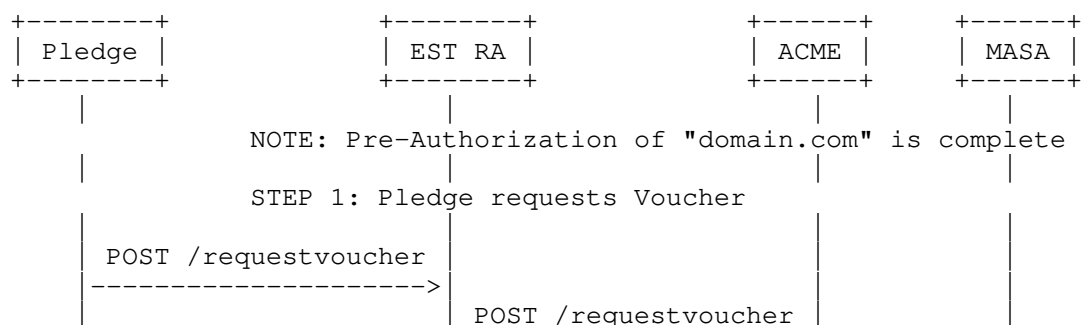
```

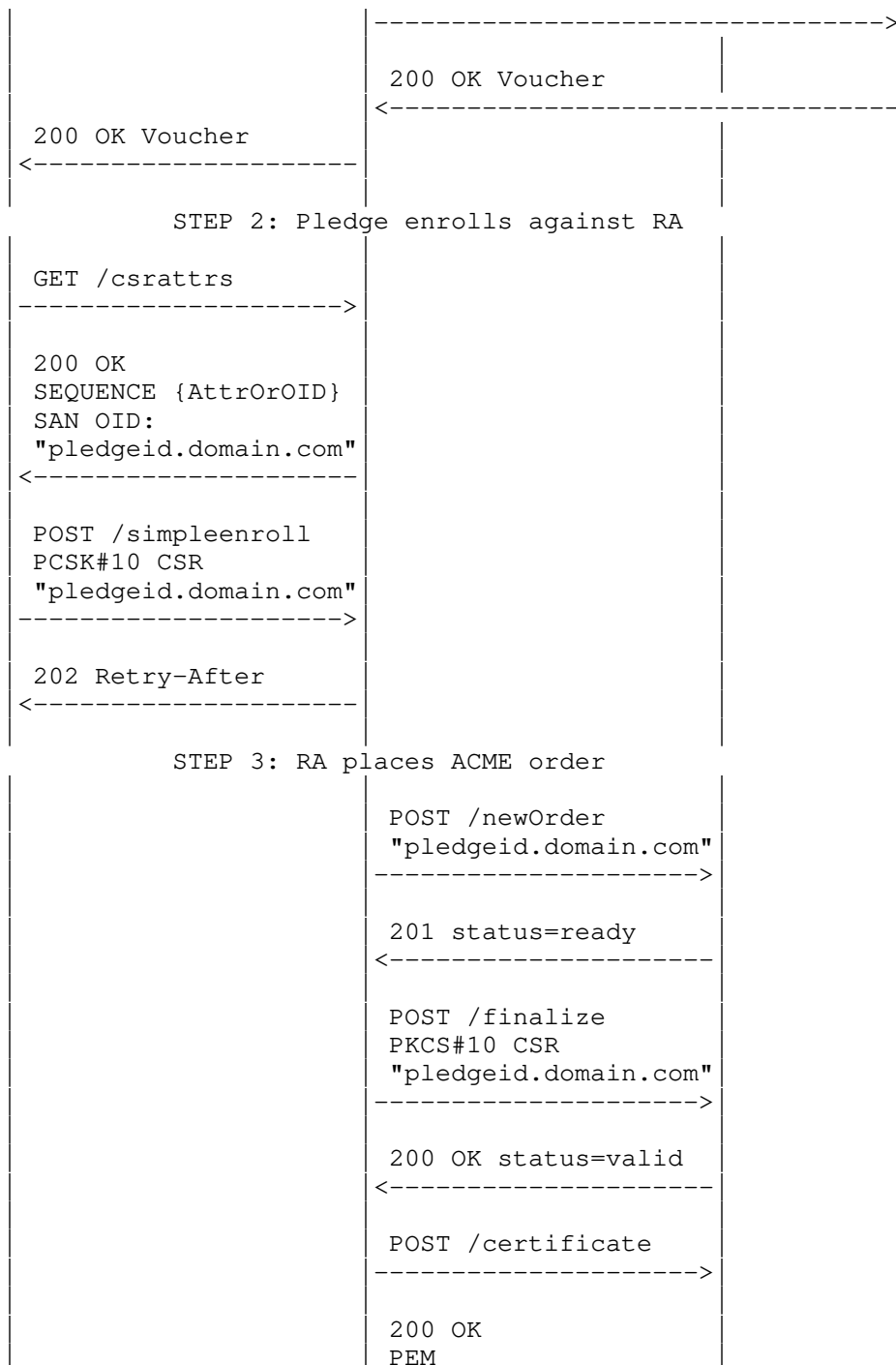
4. ACME Integration with BRSKI

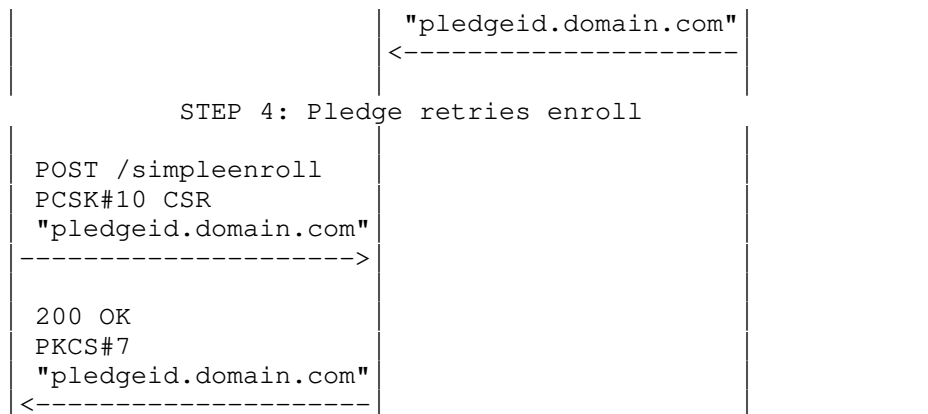
BRSKI [I-D.ietf-anima-bootstrapping-keyinfra] is based upon EST [RFC7030] and defines how to autonomically bootstrap PKI trust anchors into devices via means of signed vouchers. EST certificate enrollment may then optionally take place after trust has been established. BRSKI voucher exchange and trust establishment are based on EST extensions and the certificate enrollment part of BRSKI is fully based on EST. Similar to EST, BRSKI does not define how the EST RA communicates with the CA. Therefore, the mechanisms outlined in the previous section for using ACME as the communications protocol between the EST RA and the CA are equally applicable to BRSKI.

The following call flow shows how ACME may be integrated into a full BRSKI voucher plus EST enrollment workflow. For brevity, it assumes that the EST RA has previously proven ownership of a parent domain and that pledge certificate identifiers are a subdomain of that parent domain. The domain ownership exchanges between the RA, ACME and DNS are not shown. Similarly, not all BRSKI interactions are shown and only the key protocol flows involving voucher exchange and EST enrollment are shown.

Similar to the EST section above, the client calls EST /csrattrs API before calling the EST /simpleenroll API. This enables the server to indicate what fields the pledge should include in the CSR that the client sends in the /simpleenroll API.



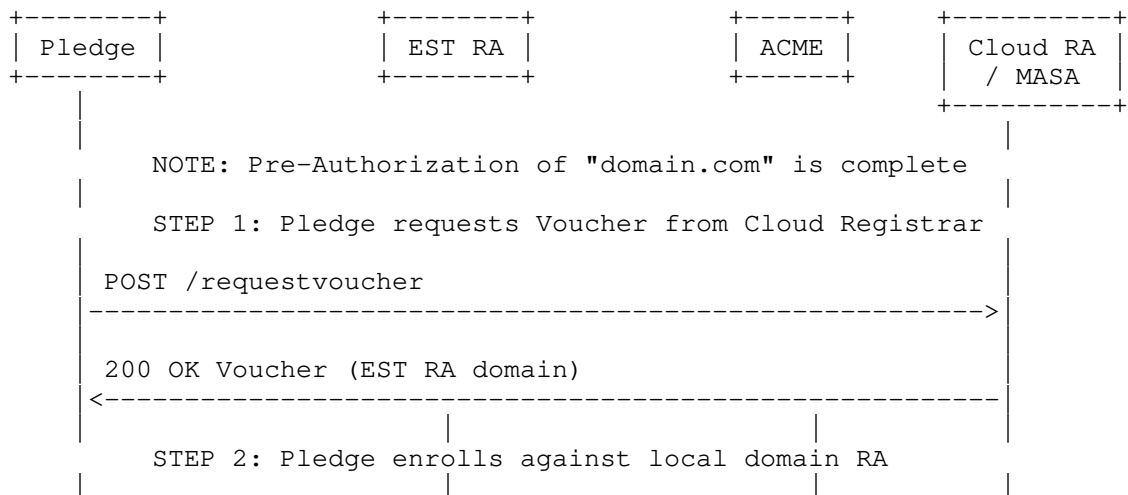


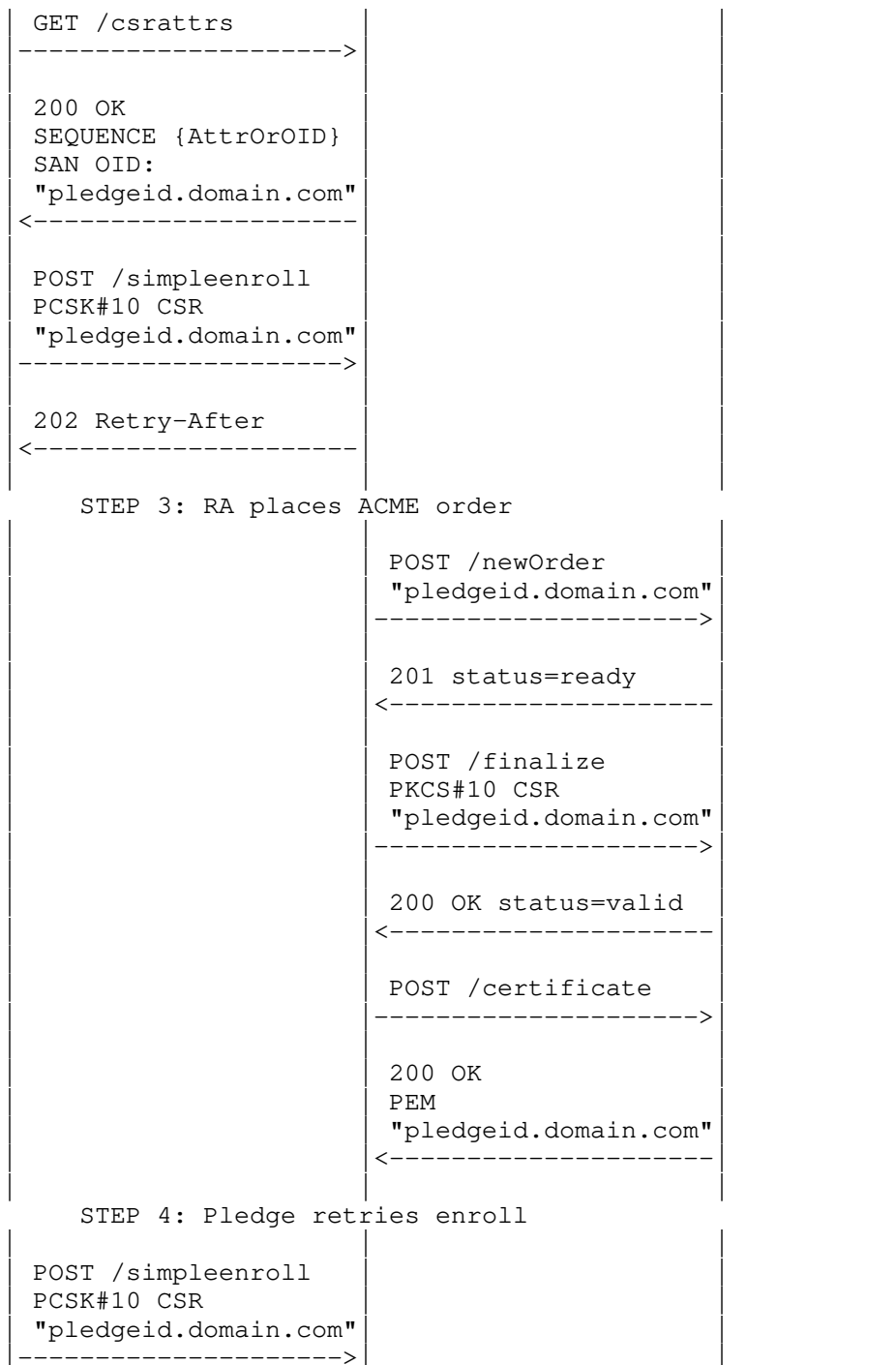


5. ACME Integration with BRSKI Default Cloud Registrar

BRSKI Cloud Registrar [I-D.friel-anima-brski-cloud] specifies the behaviour of a BRSKI Cloud Registrar, and how a pledge can interact with a BRSKI Cloud Registrar when bootstrapping. Similar to the local domain registrar BRSKI flow, ACME can be easily integrated with a cloud registrar bootstrap flow.

BRSKI cloud registrar is flexible and allows for multiple different local domain discovery and redirect scenarios. In the example illustrated here, the extension to [RFC8366] Vouchers which is defined in [[TODO ID-TBD]] and allows the specification of a bootstrap DNS domain is leveraged. This extension allows the cloud registrar to specify the local domain RA that the pledge should connect to for the purposes of EST enrollment.





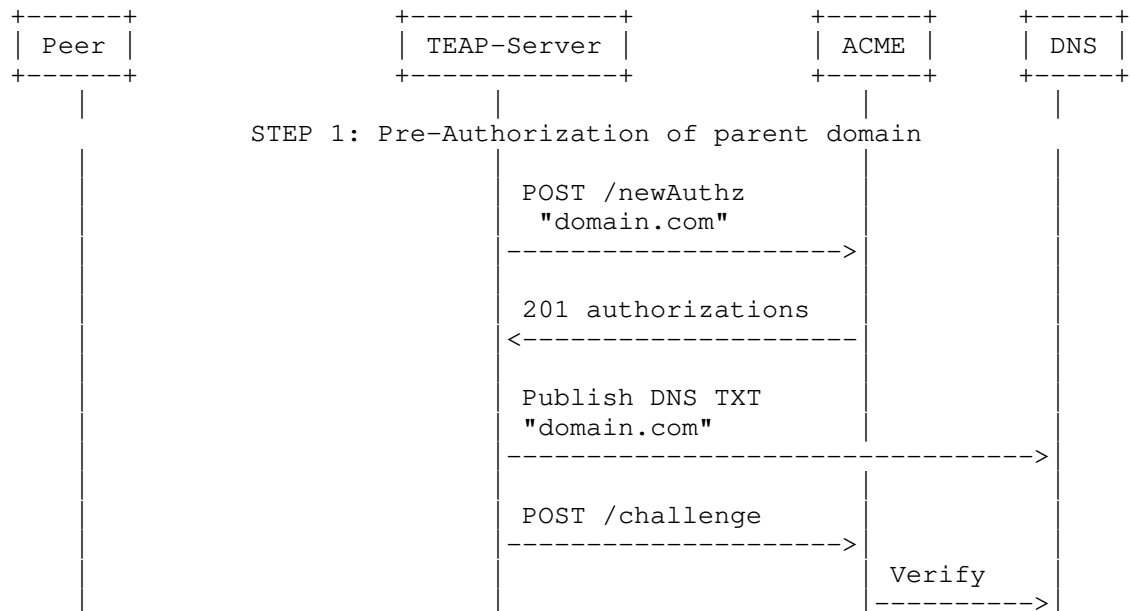
200 OK PKCS#7 "pledgeid.domain.com" <-----			
---	--	--	--

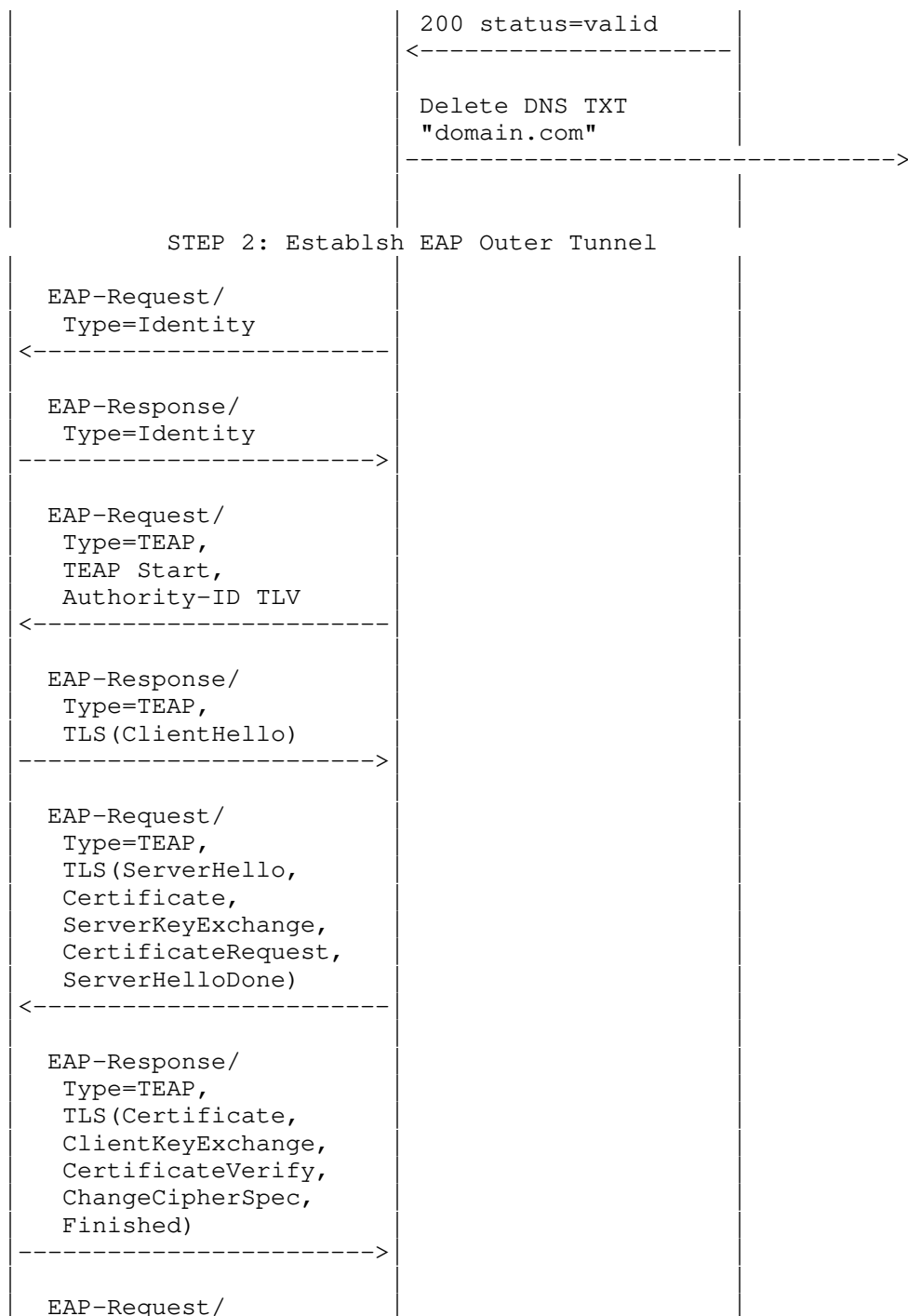
6. ACME Integration with TEAP

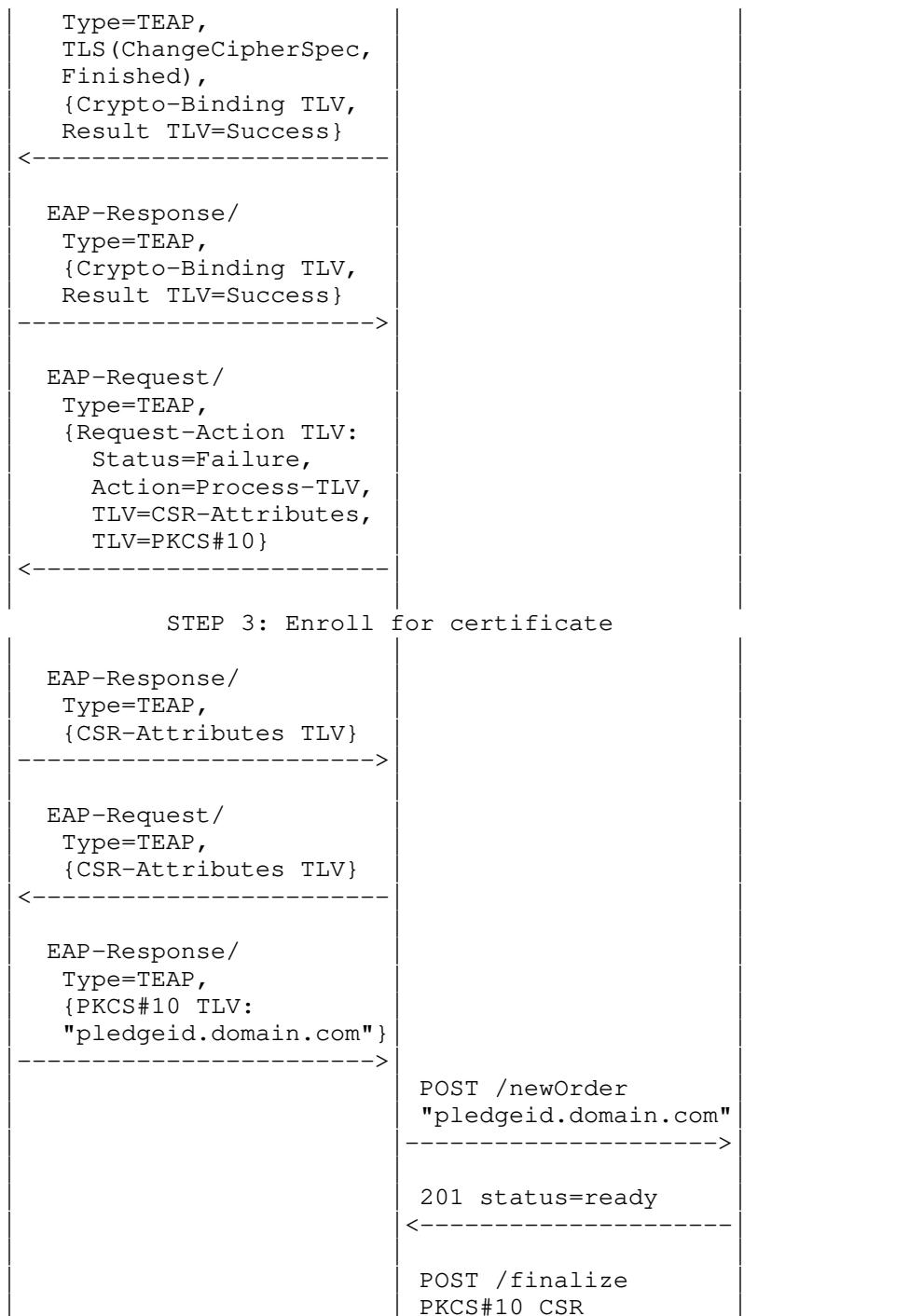
TEAP [RFC7170] defines a tunnel-based EAP method that enables secure communication between a peer and a server by using TLS to establish a mutually authenticated tunnel. TEAP enables certificate provisioning within the tunnel. TEAP does not define how the TEAP server communicates with the CA.

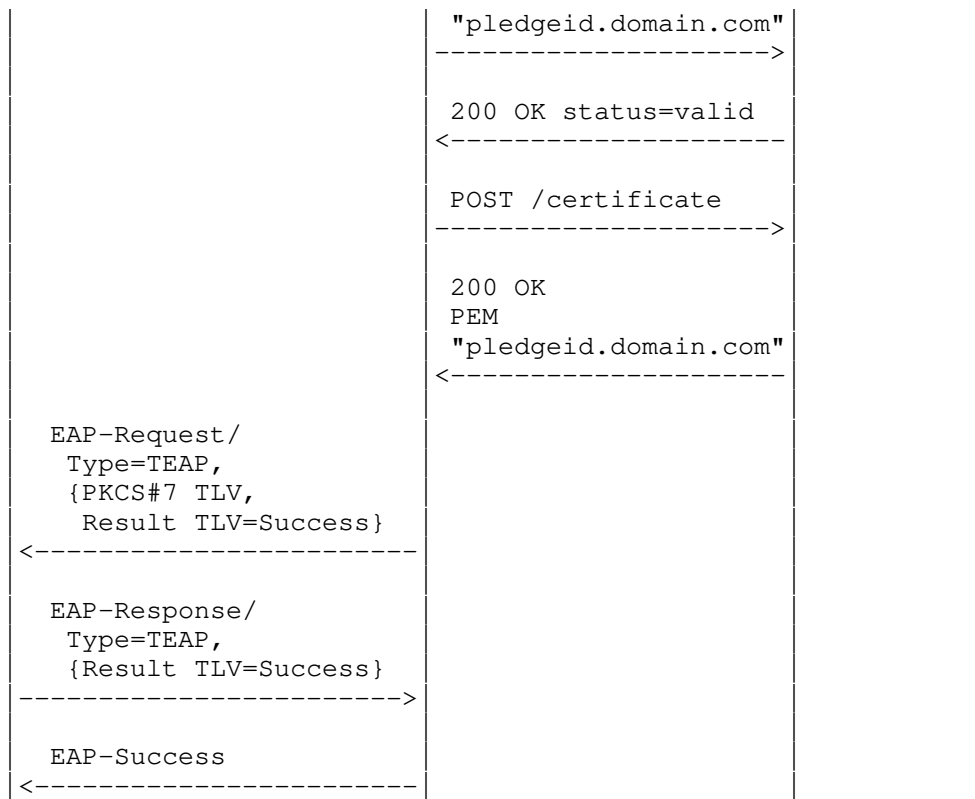
This section outlines how ACME could be used for communication between the TEAP server and the CA. The example call flow leverages [I-D.friel-acme-subdomains] and shows the TEAP server proving ownership of a parent domain, with individual client certificates being subdomains under that parent domain.

The example illustrates the TEAP server sending a Request-Action TLV including a CSR-Attributes TLV instructing the peer to send a CSR-Attributes TLV to the server. This enables the server to indicate what fields the peer should include in the CSR that the peer sends in the PKCS#10 TLV. For example, the TEAP server could instruct the peer what Subject or SAN entries to include in its CSR.







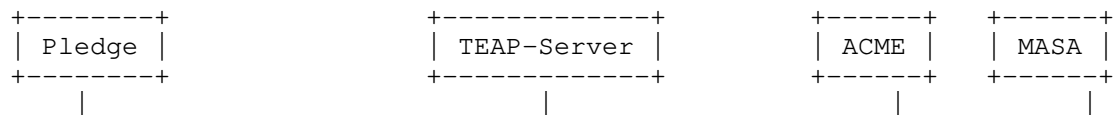


7. ACME Integration with TEAP-BRSKI

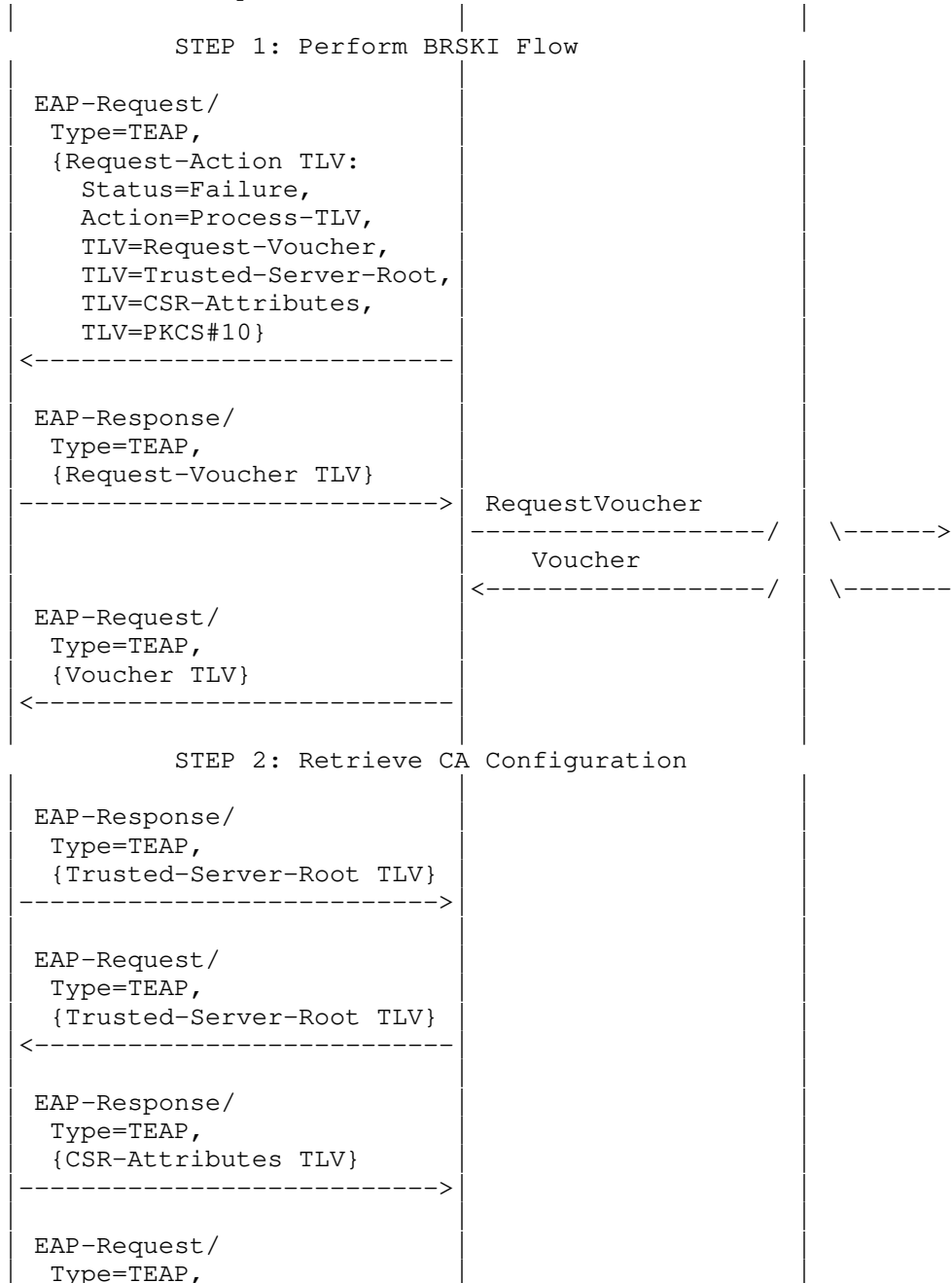
TEAP-BRSKI [I-D.lear-eap-teap-brski] defines how to execute BRSKI at layer 2 inside a TEAP tunnel. Similar to the TEAP proposal in the previous section, BRSKI-TEAP leverages the existing TEAP PKXS#10 and PKCS#7 mechanisms for certificate enrollment, and does not define how the TEAP server communicates with the CA.

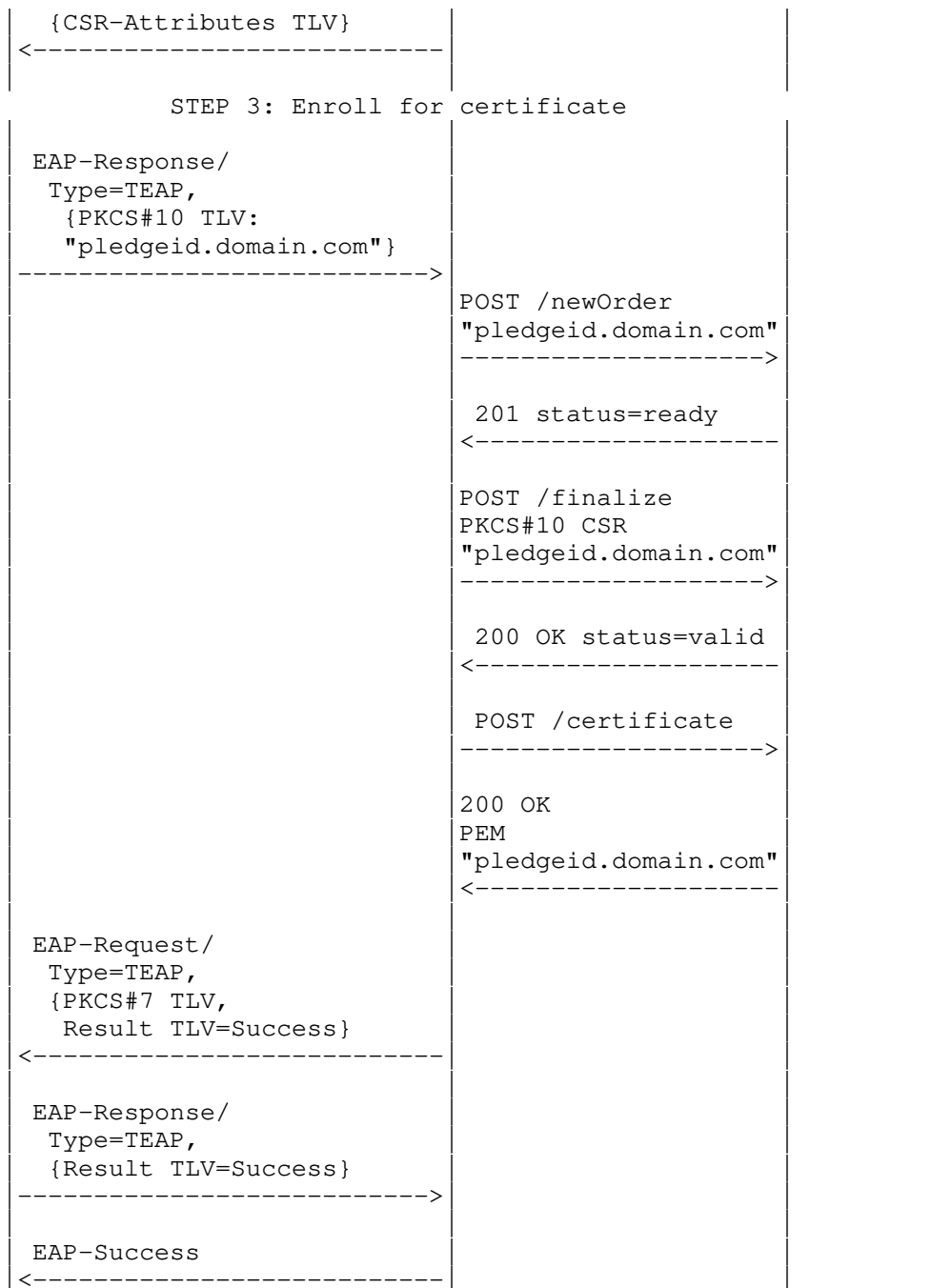
This section outlines how ACME could be used for communication between the TEAP server and the CA, and how this fits in with the TEAP-BRSKI proposal.

Similar to baseline TEAP, the TEAP server can use the CSR-Attributes TLV to tell the peer what attributes to include in its CSR request.



NOTE: Pre-Authorization of "domain.com" is complete and EAP outer tunnel is established as outlined in the previous section





8. IANA Considerations

[todo]

9. Security Considerations

[todo]

10. Informative References

[I-D.friel-acme-subdomains]

Friel, O., Barnes, R., and T. Hollebeek, "ACME for Subdomains", draft-friel-acme-subdomains-00 (work in progress), October 2019.

[I-D.friel-anima-brski-cloud]

Friel, O., Shekh-Yusef, R., and M. Richardson, "BRSKI Cloud Registrar", draft-friel-anima-brski-cloud-01 (work in progress), October 2019.

[I-D.ietf-anima-bootstrapping-keyinfra]

Pritikin, M., Richardson, M., Eckert, T., Behringer, M., and K. Watsen, "Bootstrapping Remote Secure Key Infrastructures (BRSKI)", draft-ietf-anima-bootstrapping-keyinfra-28 (work in progress), September 2019.

[I-D.lear-eap-teap-brski]

Lear, E., Friel, O., Cam-Winget, N., and D. Harkins, "Bootstrapping Key Infrastructure over EAP", draft-lear-eap-teap-brski-04 (work in progress), September 2019.

[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>>.

[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>>.

[RFC7170] Zhou, H., Cam-Winget, N., Salowey, J., and S. Hanna, "Tunnel Extensible Authentication Protocol (TEAP) Version 1", RFC 7170, DOI 10.17487/RFC7170, May 2014, <<https://www.rfc-editor.org/info/rfc7170>>.

- [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>>.
- [RFC8366] Watsen, K., Richardson, M., Pritikin, M., and T. Eckert, "A Voucher Artifact for Bootstrapping Protocols", RFC 8366, DOI 10.17487/RFC8366, May 2018, <<https://www.rfc-editor.org/info/rfc8366>>.
- [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>>.

Appendix A. Comments

Authors' Addresses

Owen Friel
Cisco

Email: ofriel@cisco.com

Richard Barnes
Cisco

Email: rlb@ipv.sx

Rifaat Shekh-Yusef
Avaya

Email: rifaat.ietf@gmail.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: April 28, 2022

O. Friel
R. Barnes
Cisco
T. Hollebeek
DigiCert
M. Richardson
Sandelman Software Works
October 25, 2021

ACME for Subdomains
draft-friel-acme-subdomains-06

Abstract

This document outlines how ACME can be used by a client to obtain a certificate for a subdomain identifier from a certification authority. The client has fulfilled a challenge against a parent domain but does not need to fulfill a challenge against the explicit subdomain as certificate policy allows issuance of the subdomain certificate without explicit subdomain ownership proof.

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 April 28, 2022.

Copyright Notice

Copyright (c) 2021 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents

carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
2. Terminology	2
3. ACME Workflow and Identifier Requirements	4
4. ACME Issuance of Subdomain Certificates	5
4.1. ACME Challenge Type	6
4.2. Authorization Object	6
4.3. Pre-Authorization	7
4.4. New Orders	8
4.5. Directory Object Metadata	10
5. Illustrative Call Flow	10
6. IANA Considerations	16
6.1. Authorization Object Fields Registry	16
6.2. Directory Object Metadata Fields Registry	16
7. Security Considerations	17
7.1. ACME Server Policy Considerations	18
8. Informative References	18
Appendix A. CA Browser Forum Baseline Requirements Extracts . .	19
Authors' Addresses	20

1. Introduction

ACME [RFC8555] defines a protocol that a certification authority (CA) and an applicant can use to automate the process of domain name ownership validation and X.509v3 (PKIX) [RFC5280] certificate issuance. This document outlines how ACME can be used to issue subdomain certificates, without requiring the ACME client to explicitly fulfill an ownership challenge against the subdomain identifiers - the ACME client need only fulfill an ownership challenge against a parent domain identifier.

2. Terminology

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.

The following terms are defined in DNS Terminology [RFC8499] and are reproduced here:

- o Label: An ordered list of zero or more octets that makes up a portion of a domain name. Using graph theory, a label identifies one node in a portion of the graph of all possible domain names.
- o Domain Name: An ordered list of one or more labels.
- o Subdomain: "A domain is a subdomain of another domain if it is contained within that domain. This relationship can be tested by seeing if the subdomain's name ends with the containing domain's name." (Quoted from [RFC1034], Section 3.1) For example, in the host name "nnn.mmm.example.com", both "mmm.example.com" and "nnn.mmm.example.com" are subdomains of "example.com". Note that the comparisons here are done on whole labels; that is, "ooo.example.com" is not a subdomain of "oo.example.com".
- o Fully-Qualified Domain Name (FQDN): This is often just a clear way of saying the same thing as "domain name of a node", as outlined above. However, the term is ambiguous. Strictly speaking, a fully-qualified domain name would include every label, including the zero-length label of the root: such a name would be written "www.example.net." (note the terminating dot). But, because every name eventually shares the common root, names are often written relative to the root (such as "www.example.net") and are still called "fully qualified". This term first appeared in [RFC0819]. In this document, names are often written relative to the root.

The following terms are defined in the CA/Browser Forum Baseline Requirements [CAB] version 1.7.1 and are reproduced here:

- o Authorization Domain Name (ADN): The Domain Name used to obtain authorization for certificate issuance for a given FQDN. The CA may use the FQDN returned from a DNS CNAME lookup as the FQDN for the purposes of domain validation. If the FQDN contains a wildcard character, then the CA MUST remove all wildcard labels from the left most portion of requested FQDN. The CA may prune zero or more labels from left to right until encountering a Base Domain Name and may use any one of the intermediate values for the purpose of domain validation
- o Base Domain Name: The portion of an applied-for FQDN that is the first domain name node left of a registry-controlled or public suffix plus the registry-controlled or public suffix (e.g. "example.co.uk" or "example.com"). For FQDNs where the right-most domain name node is a gTLD having ICANN Specification 13 in its registry agreement, the gTLD itself may be used as the Base Domain Name.

- o **Certification Authority (CA):** An organization that is responsible for the creation, issuance, revocation, and management of Certificates. The term applies equally to both Roots CAs and Subordinate CAs
- o **Domain Namespace:** The set of all possible Domain Names that are subordinate to a single node in the Domain Name System

The following additional terms are used in this document:

- o **Certification Authority (CA):** An organization that is responsible for the creation, issuance, revocation, and management of Certificates. The term applies equally to both Roots CAs and Subordinate CAs
- o **CSR:** Certificate Signing Request
- o **Parent Domain:** a domain is a parent domain of a subdomain if it contains that subdomain, as per the [RFC8499] definition of subdomain. For example, for the host name "nnn.mmm.example.com", both "mmm.example.com" and "example.com" are parent domains of "nnn.mmm.example.com".

3. ACME Workflow and Identifier Requirements

A typical ACME workflow for issuance of certificates is as follows:

1. client POSTs a newOrder request that contains a set of "identifiers"
2. server replies with a set of "authorizations" and a "finalize" URI
3. client sends POST-as-GET requests to retrieve the "authorizations", with the downloaded "authorization" object(s) containing the "identifier" that the client must prove that they control, and a set of associated "challenges", one of which the client must fulfil
4. client proves control over the "identifier" in the "authorization" object by completing one of the specified challenges, for example, by publishing a DNS TXT record
5. client POSTs a CSR to the "finalize" API
6. server replies with an updated order object that includes a "certificate" URI

7. client sends POST-as-GET request to the "certificate" URI to download the certificate

ACME places the following restrictions on "identifiers":

- o [RFC8555] section 7.1.3: The authorizations required are dictated by server policy; there may not be a 1:1 relationship between the order identifiers and the authorizations required.
- o [RFC8555] section 7.1.4: the only type of "identifier" defined by the ACME specification is an FQDN: "The only type of identifier defined by this specification is a fully qualified domain name (type: "dns"). The domain name MUST be encoded in the form in which it would appear in a certificate."
- o [RFC8555] section 7.4: the "identifier" in the CSR request must match the "identifier" in the newOrder request: "The CSR MUST indicate the exact same set of requested identifiers as the initial newOrder request."
- o [RFC8555] section 8.3: the "identifier", or FQDN, in the "authorization" object must be used when fulfilling challenges via HTTP: "Construct a URL by populating the URL template ... where the domain field is set to the domain name being verified"
- o [RFC8555] section 8.4: the "identifier", or FQDN, in the "authorization" object must be used when fulfilling challenges via DNS: "The client constructs the validation domain name by prepending the label "_acme-challenge" to the domain name being validated."

ACME does not mandate that the "identifier" in a newOrder request matches the "identifier" in "authorization" objects.

4. ACME Issuance of Subdomain Certificates

As noted in the previous section, ACME does not mandate that the "identifier" in a newOrder request matches the "identifier" in "authorization" objects. This means that the ACME specification does not preclude an ACME server processing newOrder requests and issuing certificates for a subdomain without requiring a challenge to be fulfilled against that explicit subdomain.

ACME server policy could allow issuance of certificates for a subdomain to a client where the client only has to fulfill an authorization challenge for a parent domain of that subdomain. This allows a flow where a client proves ownership of, for example,

"example.org" and then successfully obtains a certificate for "sub.example.org".

ACME server policy is out of scope of this document, however some commentary is provided in Section 7.1.

Clients need a mechanism to instruct the ACME server that they are requesting authorization for a Domain Namespace subordinate to a given ADN, as opposed to just requesting authorization for an explicit ADN identifier. Clients need a mechanism to do this in both newAuthz and newOrder requests. ACME servers need a mechanism to indicate to clients that authorization objects are valid for an entire Domain Namespace. These are described in this section.

4.1. ACME Challenge Type

ACME for subdomains is restricted for use with "dns-01" challenges. If a server policy allows a client to fulfill a challenge against a parent ADN of a requested certificate FQDN identifier, then the server MUST issue a "dns-01" challenge against that parent ADN.

4.2. Authorization Object

ACME [RFC8555] section 7.1.4 defines the authorization object. When ACME server policy allows authorization for Domain Namespaces subordinate to an ADN, the server indicates this by including the "domainNamespace" flag in the authorization object for that ADN identifier:

domainNamespace (optional, boolean): This field MUST be present and true for authorizations where ACME server policy allows certificates to be issued for any Domain Name in the Domain Namespace subordinate to the ADN specified in the 'identifier' field of the authorization object.

The following example shows an authorization object for the ADN "example.org" where the authorization covers the Domain Namespace subordinate to "example.org".

```
{
  "status": "valid",
  "expires": "2015-03-01T14:09:07.99Z",

  "identifier": {
    "type": "dns",
    "value": "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"
    }
  ],

  "domainNamespace": true
}
```

If the "domainNamespace" field is not included, then the assumed default value is false.

4.3. Pre-Authorization

The standard ACME workflow has authorization objects created reactively in response to a certificate order. ACME also allows for pre-authorization, where clients obtain authorization for an identifier proactively, outside of the context of a specific issuance. With the ACME pre-authorization flow, a client can pre-authorize for a parent ADN once, and then issue multiple newOrder requests for certificates with identifiers in the Domain Namespace subordinate to that ADN.

ACME [RFC8555] section 7.4.1 defines the "identifier" object for newAuthz requests. One additional field for the "identifier" object is defined:

domainNamespace (optional, boolean): An ACME client sets this flag to indicate to the server that it is requesting an authorization for the Domain Namespace subordinate to the specified ADN identifier value

Clients include the flag in the "identifier" object of newAuthz requests to indicate that they are requesting a Domain Namespace authorization. In the following example newAuthz payload, the client

is requesting pre-authorization for the Domain Namespace subordinate to "example.org".

```
"payload": base64url({
  "identifier": {
    "type": "dns",
    "value": "example.org",
    "domainNamespace": true
  }
})
```

If the server is willing to allow a single authorization for the Domain Namespace, and there is not an existing authorization object for the identifier, then it will create an authorization object and include the "domainNamespace" flag with value of true. If the server policy does not allow creation of Domain Namespace authorizations subordinate to that ADN, the server can create an authorization object for the indicated identifier, and include the "domainNamespace" flag with value of false. In both scenarios, handling of the pre-authorization follows the process documented in ACME section 7.4.1.

4.4. New Orders

Clients need a mechanism to optionally indicate to servers whether or not they are authorized to fulfill challenges against parent ADNs for a given identifier FQDN. For example, if a client places an order for an identifier "foo.bar.example.org", and is authorized to update DNS TXT records against the parent ADNs "bar.example.org" or "example.org", then the client needs a mechanism to indicate control over the parent ADNs to the ACME server.

This can be achieved by adding an optional field "domainNamespace" to the "identifiers" field in the order object:

domainNamespace (optional, string): This is the parent ADN of a Domain Namespace that the requested identifier belongs to. The client MUST have DNS control over the parent ADN.

This field specifies the ADN of the Domain Namespace that the client has DNS control over, and is capable of fulfilling challenges against. Based on server policy, the server can choose to issue a challenge against any parent domain of the identifier in the Domain Namespace up to and including the specified "domainNamespace", and create a corresponding authorization object against the chosen identifier.

In the following example `newOrder` payload, the client requests a certificate for identifier `"foo.bar.example.org"` and indicates that it can fulfill a challenge against the parent ADN and the Domain Namespace subordinate to `"bar.example.org"`. The server can then choose to issue a challenge against either `"foo.bar.example.org"` or `"bar.example.org"` identifiers.

```
"payload": base64url({
  "identifiers": [
    { "type": "dns",
      "value": "foo.bar.example.org",
      "domainNamespace": "bar.example.org"  }
  ],
  "notBefore": "2016-01-01T00:04:00+04:00",
  "notAfter": "2016-01-08T00:04:00+04:00"
})
```

In the following example `newOrder` payload, the client requests a certificate for identifier `"foo.bar.example.org"` and indicates that it can fulfill a challenge against the parent ADN and the Domain Namespace subordinate to `"example.org"`. The server can then choose to issue a challenge against any one of `"foo.bar.example.org"`, `"bar.example.org"` or `"example.org"` identifiers.

```
"payload": base64url({
  "identifiers": [
    { "type": "dns",
      "value": "foo.bar.example.org",
      "domainNamespace": "example.org"  }
  ],
  "notBefore": "2016-01-01T00:04:00+04:00",
  "notAfter": "2016-01-08T00:04:00+04:00"
})
```

If the client is unable to fulfill authorizations against parent ADNs, the client should not include the `"domainNamespace"` field.

Server `newOrder` handling generally follows the process documented ACME section 7.4. If the server is willing to allow Domain Namespace authorizations for the ADN specified in `"domainNamespace"`, then it creates an authorization object against that ADN and includes the `"domainNamespace"` flag with a value of `true`. If the server policy does not allow creation of Domain Namespace authorizations against that ADN, then it can create an authorization object for the indicated identifier value, and include the `"domainNamespace"` flag with value of `false`.

4.5. Directory Object Metadata

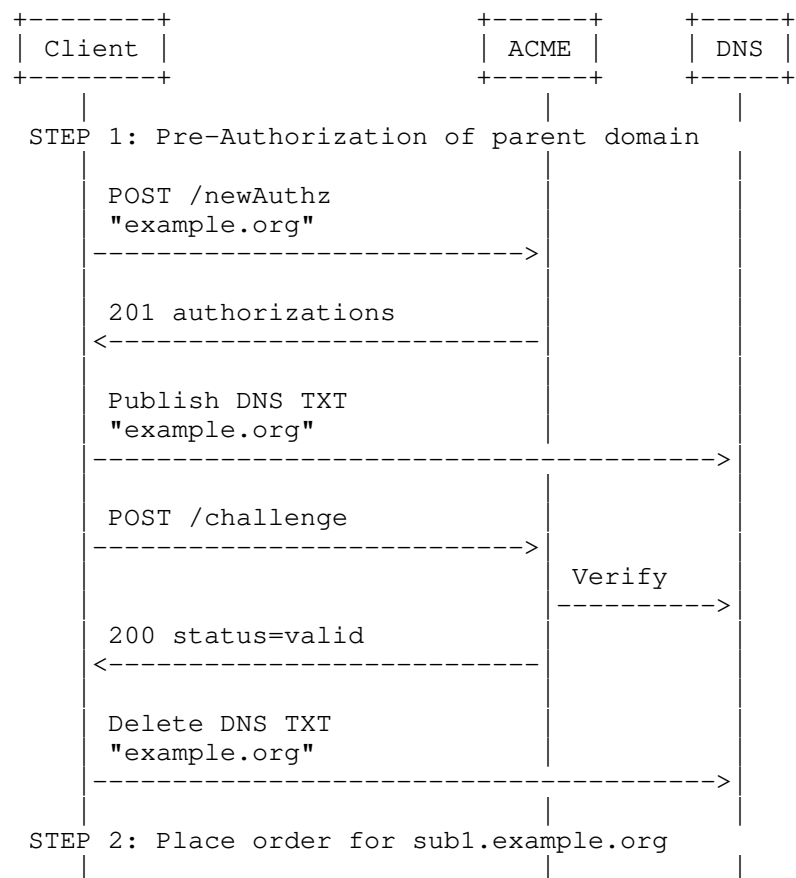
An ACME server can advertise support for authorization of Domain Namespaces by including the following boolean flag in its "ACME Directory Metadata Fields" registry:

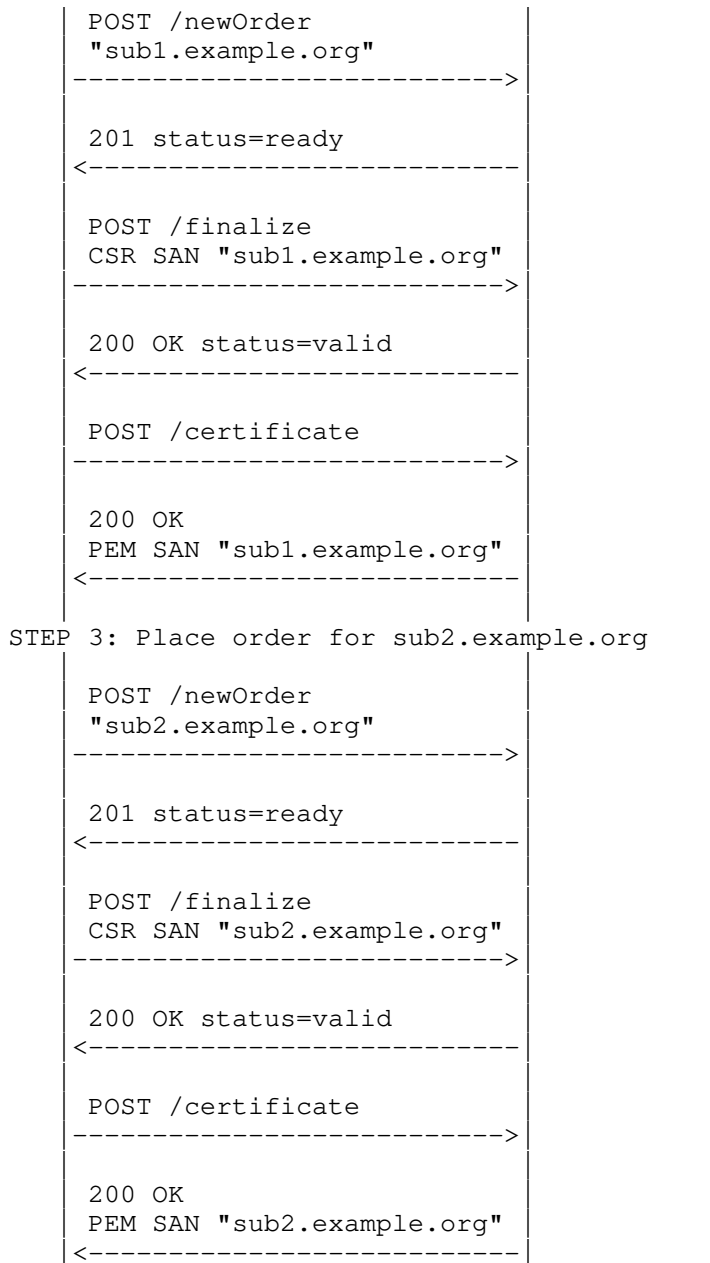
`domainNamespace` (optional, bool): Indicates if an ACME server supports authorization of Domain Namespaces.

If not specified, then no default value is assumed. If an ACME server supports authorization of Domain Namespaces, it can indicate this by including this field with a value of "true".

5. Illustrative Call Flow

The call flow illustrated here uses the ACME pre-authorization flow using DNS-based proof of ownership.





- o STEP 1: Pre-authorization of Domain Namespace

The client sends a newAuthz request for the parent ADN of the Domain Namespace including the "domainNamespace" flag in the identifier object.

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

```
{
  "protected": base64url({
    "alg": "ES256",
    "kid": "https://example.com/acme/acct/evOfKhNU60wg",
    "nonce": "uQpSjlRb4vQVCjVYAyyUWg",
    "url": "https://example.com/acme/new-authz"
  }),
  "payload": base64url({
    "identifier": {
      "type": "dns",
      "value": "example.org",
      "domainNamespace": true
    }
  }),
  "signature": "nuSDISbWG8mMgE7H...QyVUL68yzf3Zawps"
}
```

The server creates and returns an authorization object for the identifier including the "domainNamespace" flag. The object is initially in "pending" state. Once the client completes the challenge, the server will transition the authorization object and associated challenge object status to "valid".

```
{
  "status": "pending",
  "expires": "2015-03-01T14:09:07.99Z",

  "identifier": {
    "type": "dns",
    "value": "example.org"
  },

  "challenges": [
    {
      "url": "https://example.com/acme/chall/prV_B7yEyA4",
      "type": "http-01",
      "status": "pending",
      "token": "DGyRejmCefe7v4NfDGDKfA",
      "validated": "2014-12-01T12:05:58.16Z"
    }
  ],

  "domainNamespace": true
}
```

- o STEP 2: The client places a newOrder for "sub1.example.org"

The client sends a newOrder request to the server and includes the subdomain identifier. Note that the identifier is in the Domain Namespace that has been pre-authorized in step 1. The client does not need to include the "domainNamespace" field in the "identifier" object as it has already pre-authorized the Domain Namespace.

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

{
  "protected": base64url({
    "alg": "ES256",
    "kid": "https://example.com/acme/acct/evOfKhNU60wg",
    "nonce": "5XJ1L3lEkMG7tR6pA00clA",
    "url": "https://example.com/acme/new-order"
  }),
  "payload": base64url({
    "identifiers": [
      { "type": "dns", "value": "sub1.example.org" }
    ],
    "notBefore": "2016-01-01T00:04:00+04:00",
    "notAfter": "2016-01-08T00:04:00+04:00"
  }),
  "signature": "H6ZXtGjTZyUnPeKn...wEA4Tk1Bdh3e454g"
}
```

As an authorization object already exists for the parent ADN of the Domain Namespace, the server replies with an order object with a status of "valid" that includes a link to the existing "valid" authorization object.

```
HTTP/1.1 201 Created
Replay-Nonce: MYAuvOpaoIiywTezizk5vw
Link: <https://example.com/acme/directory>;rel="index"
Location: https://example.com/acme/order/TolocE8rfgo

{
  "status": "valid",
  "expires": "2016-01-05T14:09:07.99Z",

  "notBefore": "2016-01-01T00:00:00Z",
  "notAfter": "2016-01-08T00:00:00Z",

  "identifiers": [
    { "type": "dns", "value": "sub1.example.org" }
  ],

  "authorizations": [
    "https://example.com/acme/authz/PAniVnsZcis"
  ],

  "finalize": "https://example.com/acme/order/Tolocrfgo/finalize"
}
```

The client can proceed to finalize the order and download the certificate for "sub1.example.org".

- o STEP 3: The client places a newOrder for "sub2.example.org"

The client sends a newOrder request to the server and includes the subdomain identifier. Note that the identifier is in the Domain Namespace that has been pre-authorized in step 1. The client does not need to include the "domainNamespace" field in the "identifier" object as it has already pre-authorized the Domain Namespace.

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

```
{
  "protected": base64url({
    "alg": "ES256",
    "kid": "https://example.com/acme/acct/evOfKhNU60wg",
    "nonce": "5XJ1L3lEkMG7tR6pA00clA",
    "url": "https://example.com/acme/new-order"
  }),
  "payload": base64url({
    "identifiers": [
      { "type": "dns", "value": "sub2.example.org" }
    ],
    "notBefore": "2016-01-01T00:04:00+04:00",
    "notAfter": "2016-01-08T00:04:00+04:00"
  }),
  "signature": "H6ZXtGjTZyUnPeKn...wEA4Tk1Bdh3e454g"
}
```

As an authorization object already exists for the parent ADN of the Domain Namespace, the server replies with an order object with a status of "valid" that includes a link to the existing "valid" authorization object.


```
HTTP/1.1 201 Created
Replay-Nonce: MYAuvOpaoIiywTezizk5vw
Link: <https://example.com/acme/directory>;rel="index"
Location: https://example.com/acme/order/T0locE8rfgo

{
  "status": "valid",
  "expires": "2016-01-05T14:09:07.99Z",

  "notBefore": "2016-01-01T00:00:00Z",
  "notAfter": "2016-01-08T00:00:00Z",

  "identifiers": [
    { "type": "dns", "value": "sub1.example.org" }
  ],

  "authorizations": [
    "https://example.com/acme/authz/PAniVnsZcis"
  ],

  "finalize": "https://example.com/acme/order/R0ni7rdde/finalize"
}
```

The client can proceed to finalize the order and download the certificate for "sub2.example.org".

6. IANA Considerations

6.1. Authorization Object Fields Registry

The following field is added to the "ACME Authorization Object Fields" registry defined in ACME [RFC8555].

Field Name	Field Type	Configurable	Reference
domainNamespace	boolean	false	RFC XXXX

6.2. Directory Object Metadata Fields Registry

The following field is added to the "ACME Directory Metadata Fields" registry defined in ACME [RFC8555].

Field Name	Field Type	Reference
domainNamespace	boolean	RFC XXXX

7. Security Considerations

This document documents enhancements to ACME [RFC8555] that optimize the protocol flows for issuance of certificates for subdomains. The underlying goal of ACME for Subdomains remains the same as that of ACME: managing certificates that attest to identifier/key bindings for these subdomains. Thus, ACME for Subdomains has the same two security goals as ACME:

1. Only an entity that controls an identifier can get an authorization for that identifier
2. Once authorized, an account key's authorizations cannot be improperly used by another account

ACME for Subdomains makes no changes to:

- o account or account key management
- o ACME channel establishment, security mechanisms or threat model
- o Validation channel establishment, security mechanisms or threat model

Therefore, all Security Considerations in ACME in the following areas are equally applicable to ACME for Subdomains:

- o Threat Model
- o Integrity of Authorizations
- o Denial-of-Service Considerations
- o Server-Side Request Forgery
- o CA Policy Considerations

Some additional comments on ACME server policy are given in the following section.

7.1. ACME Server Policy Considerations

The ACME for Subdomains and the ACME specifications do not mandate any specific ACME server or CA policies, or any specific use cases for issuance of certificates. For example, an ACME server could be used:

- o to issue Web PKI certificates where the ACME server must comply with CA/Browser Forum [CAB] Baseline Requirements.
- o as a Private CA for issuance of certificates within an organisation. The organisation could enforce whatever policies they desire on the ACME server.
- o for issuance of IoT device certificates. There are currently no IoT device certificate policies that are generally enforced across the industry. Organizations issuing IoT device certificates can enforce whatever policies they desire on the ACME server.

ACME server policy could specify whether:

- o issuance of subdomain certificates is allowed based on proof of ownership of a parent domain
- o issuance of subdomain certificates is allowed, but only for a specific set of parent domains
- o whether DNS based proof of ownership, or HTTP based proof of ownership, or both, are allowed

ACME server policy specification is explicitly out of scope of this document. For reference, extracts from CA/Browser Forum Baseline Requirements are given in the appendices.

8. Informative References

- [CAB] CA/Browser Forum, "Baseline Requirements for the Issuance and Management of Publicly-Trusted Certificates", n.d., <<https://cabforum.org/wp-content/uploads/CA-Browser-Forum-BR-1.7.1.pdf>>.
- [RFC0819] Su, Z. and J. Postel, "The Domain Naming Convention for Internet User Applications", RFC 819, DOI 10.17487/RFC0819, August 1982, <<https://www.rfc-editor.org/info/rfc819>>.

- [RFC1034] Mockapetris, P., "Domain names - concepts and facilities", STD 13, RFC 1034, DOI 10.17487/RFC1034, November 1987, <<https://www.rfc-editor.org/info/rfc1034>>.
- [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>>.
- [RFC5280] Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 5280, DOI 10.17487/RFC5280, May 2008, <<https://www.rfc-editor.org/info/rfc5280>>.
- [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>>.
- [RFC8499] Hoffman, P., Sullivan, A., and K. Fujiwara, "DNS Terminology", BCP 219, RFC 8499, DOI 10.17487/RFC8499, January 2019, <<https://www.rfc-editor.org/info/rfc8499>>.
- [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>>.

Appendix A. CA Browser Forum Baseline Requirements Extracts

The CA/Browser Forum Baseline Requirements [CAB] allow issuance of subdomain certificates where authorization is only required for a parent domain. Baseline Requirements version 1.7.1 states:

- o Section: "1.6.1 Definitions": Authorization Domain Name: The Domain Name used to obtain authorization for certificate issuance for a given FQDN. The CA may use the FQDN returned from a DNS CNAME lookup as the FQDN for the purposes of domain validation. If the FQDN contains a wildcard character, then the CA MUST remove all wildcard labels from the left most portion of requested FQDN. The CA may prune zero or more labels from left to right until encountering a Base Domain Name and may use any one of the intermediate values for the purpose of domain validation.
- o Section: "3.2.2.4.6 Agreed-Upon Change to Website": Once the FQDN has been validated using this method, the CA MAY also issue Certificates for other FQDNs that end with all the labels of the

validated FQDN. This method is suitable for validating Wildcard Domain Names.

- o Section: "3.2.2.4.7 DNS Change": Once the FQDN has been validated using this method, the CA MAY also issue Certificates for other FQDNs that end with all the labels of the validated FQDN. This method is suitable for validating Wildcard Domain Names.

Authors' Addresses

Owen Friel
Cisco

Email: ofriel@cisco.com

Richard Barnes
Cisco

Email: rlb@ipv.sx

Tim Hollebeek
DigiCert

Email: tim.hollebeek@digicert.com

Michael Richardson
Sandelman Software Works

Email: mcr+ietf@sandelman.ca

IETF
Internet-Draft
Intended status: Standards Track
Expires: February 6, 2021

K. Moriarty
Dell Technologies
August 5, 2020

ACME End User Client and Code Signing Certificates
draft-moriarty-acme-client-05

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 support end user client, device client, and code signing certificates.

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 February 6, 2021.

Copyright Notice

Copyright (c) 2020 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
2. Identity Proofing for Client Certificates	2
3. End User Client Certificates	3
4. CodeSigning Certificates	5
5. Pre-authorization	8
6. Challenge Types	8
6.1. One Time Password (OTP)	8
6.1.1. HMAC-Based One-Time Password (HOTP)	9
6.1.2. Time-Based One-Time Password (TOTP)	9
6.1.3. Generic One Time Password (OTP)	9
6.2. Public Key Cryptography	10
6.3. WebAuthn or Public/Private Key Pairs	11
7. Security Considerations	11
8. IANA Considerations	12
9. Contributors	12
10. References	12
10.1. Normative References	12
10.2. Informative References	13
10.3. URL References	13
Appendix A. Change Log	14
Appendix B. Open Issues	14
Author's Address	14

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 and code signing may also benefit from automated management to ease the deployment and maintenance of these certificate types, thus the definition of this extension defining challenge types specific to that usage.

2. Identity Proofing for Client Certificates

As with the TLS certificates defined in the core ACME document [<xref target="RFC8555"/>](#), identity proofing for ACME issued end user client, device client, and code 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). Using this guidance, a CA might select from the identity proofing levels to assert claims on the issued certificates as described in NIST SP 800-63 r3 [NIST800-63r3].

The certificate issuing CA may make this choice by certificate type issued. Once identity proofing has been performed, in cases where this is part of the process, and certificates have been issued, NIST SP 800-63 r3 [NIST800-63r3] includes recommendations for authentication or in the context of ACME, management of issuance for subsequent client, device, or code-signing certificates:

If federations and assertions are used for authorizing certificate issuance, NIST SP 800-63 C [NIST800-63C] may be referenced for guidance on levels of assurance.

Existing PKI certification authorities (CAs) tend to use a set of ad hoc protocols for certificate issuance and identity verification. For each certificate usage type, a basic process will be described to obtain an initial certificate and for the certificate renewal process. If higher assurance levels are desired, the guidance from NIST SP 800-63 r3 [NIST800-63r3] may be useful and out-of-band identity proofing options are possible options for pre-authorization challenges or notifications.

3. End User Client Certificates

A client certificate used to authenticate an end user may be used for mutual authentication in TLS, EAP-TLS, or messaging. The client certificate in this case may be stored in a browser, PKCS-#11 container, KMIP (possible, but just code signing and device client certificates in practice), or another key container. To obtain an end user client certificate, there are several possibilities to automate authentication of an identity credential intended to be tied to an end user.

[We need to determine if it is important in ACME to define an automated method that tests the identity or the user or to just have consistent credentials for the authentication challenges. The credentials may be distributed through an out-of-band method that involves identity proofing.]

[Several authentication options with identity proofing are intentionally provided for review and discussion by the ACME working group.]

A trusted federated service that ties the user to an email address with a reputation of the user attached to the email may be possible. One such example might be the use of a JWT signed OAuth token.

Risk based authentication used for identity proofing with red herring questions is a third option that could utilize public information on individuals to authenticate. This would be similar to the signup process used in some financial applications.

Existing credentials - for instance, FIDO. FIDO uses a public key pair and does not perform identity proofing. FIDO authentication provides a different key pair to each service using FIDO for authentication, which are generated at the client and registered by the server. This may require using the FIDO credentials from a specific service for authentication to gain ACME issued credentials (not advised based on how FIDO credentials are supposed to be used). Are there instances where the same provider would issue both sets of credentials? You wouldn't want to expose your FIDO credentials to a different party, that's why each service has their own. Would you set up a mechanism to get FIDO credentials to then obtain a certificate? (What use cases would this be necessary? When do you need a certificate where you already have a specific public/private key pair?) This can be defined as an auth type, but should it be?

One-time password (OTP) authentication is a secure option. 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.

One consideration is to understand if the use case could just use FIDO and not create anything new (ACME client certificates). FIDO provides 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 it's 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. However, is automated identity proofing important for any user certificate or should it remain

decoupled where it could be automated by a service offering or is there a need for a standardized mechanism to support it for user certificates?

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

4. 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 similar 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 chose 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 for 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.

[For standard certificates, is it worth defining SMS and email for the challenge? Obviously, EV needs more, so a few choices are suggested in this revision.]

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.

Questions for reviewers:

[Is there interest to set a specific or default challenge object for CodeSigning Certificates? Or should this be left to individual CAs to decide and differentiate? The current challenge types defined in RFC8555 include HTTPS (provisioning HTTP resources) and DNS (provisioning a TXT resource record). Use of DNS may be possible, but the HTTP resource doesn't necessarily make sense. Since the

process to retrieve an EV CodeSigning certificate usually requires proof of the organization and validation from one of 2 named administrators, some other challenge type like public/private key pairs or OTP may be needed as defined challenge types. An organization may want to tie this contact to a role rather than a person and that consideration should be made in the design as well as implementation by organizations.]

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. 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. Additional types may be added if agreed upon by the working group.

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 Enhanced Key Usage SHOULD be set to show this is a client certificate., using OID "1.3.6.1.5.5.7.3.2". 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 or two, but 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 an HSM, which typically requires authentication is intentionally left out-of-scope.

5. Pre-authorization

Additional challenge types are defined here for the verification of administrators at an organization requesting CodeSigning certificates. SMS and email are 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.

6. 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 challenges with certificate types such as the Code Signing Certificate. 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.

6.1. 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 acomodate 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"
}
```

6.1.1. 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"
}
```

6.1.2. 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"
}
```

6.1.3. 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 acomodate 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"
}
```

6.2. 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 receipt 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"
}
```

6.3. WebAuthn or Public/Private Key Pairs

W3C's WebAuthn uses 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.

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"
}
```

7. Security Considerations

This will likely be full of considerations and is TBD for this revision until challenge types are settled.

8. IANA Considerations

This memo includes no request to IANA, yet.

9. 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.

10. References

10.1. 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>>.
- [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>>.
- [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>>.

10.2. Informative References

[I-D.ietf-acme-ip]
Shoemaker, R., "ACME IP Identifier Validation Extension",
draft-ietf-acme-ip-08 (work in progress), October 2019.

10.3. URL References

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

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

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

Appendix A. Change Log

Note to RFC Editor: if this document does not obsolete an existing RFC, please remove this appendix before publication as an RFC.

02 draft added subsections contributed from Thomas Peterson on HOTP and TOTP.

Appendix B. Open Issues

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

Author's Address

Kathleen M. Moriarty
Dell Technologies
176 South Street
Hopkinton
US

EMail: Kathleen.Moriarty@dell.com