Abstract

This document defines a strategy to securely assign a pledge to an owner, using an artifact signed, directly or indirectly, by the pledge’s manufacturer. This artifact is known as a "voucher".

This document builds upon the work in [RFC8366], encoding the resulting artifact in CBOR. Use with two signature technologies are described.

Additionally, this document explains how constrained vouchers may be transported as an extension to the [I-D.ietf-ace-coap-est] protocol.

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 January 5, 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 .......................... 3
2. Terminology ........................... 4
3. Requirements Language ................. 4
4. Survey of Voucher Types ................ 4
5. Discovery and URI ..................... 5
6. Artifacts ............................. 7
   6.1. Voucher Request artifact ........... 7
   6.1.1. Tree Diagram .................... 7
   6.1.2. SID values ..................... 8
   6.1.3. YANG Module .................... 9
   6.1.4. Example voucher request artifact .. 13
   6.2. Voucher artifact .................. 14
   6.2.1. Tree Diagram .................... 14
   6.2.2. SID values ..................... 15
   6.2.3. YANG Module .................... 15
   6.2.4. Example voucher artifacts ....... 18
   6.3. Signing voucher and voucher-request artifacts ... 19
       6.3.1. CMS signing .................. 19
       6.3.2. COSE signing ................. 20
7. Design Considerations .................. 21
8. Security Considerations ............... 21
8.1. Clock Sensitivity ................... 21
8.2. Protect Voucher PKI in HSM .......... 21
8.3. Test Domain Certificate Validity when Signing .... 21
9. IANA Considerations ................... 21
9.1. Resource Type Registry ............... 21
9.2. The IETF XML Registry ............... 21
9.3. The YANG Module Names Registry ...... 22
9.4. The RFC SID range assignment sub-registry .... 22
9.5. The SMI Security for S/MIME CMS Content Type Registry . 22
9.6. Media-Type Registry ................ 23
   9.6.1. application/voucher-cms+cbor .... 23
   9.6.2. application/voucher-cose+cbor .... 23
9.7. CoAP Content-Format Registry ........ 24
10. Acknowledgements .................... 24
11. Changelog ........................... 25
1. Introduction

Enrollment of new nodes into constrained networks with constrained nodes present unique challenges.

There are bandwidth and code space issues to contend. A solution such as [I-D.ietf-anima-bootstrapping-keyinfra] may be too large in terms of code space or bandwidth required.

This document defines a constrained version of [RFC8366]. Rather than serializing the YANG definition in JSON, it is serialized into CBOR ([RFC7049]).

This document follows a similar, but not identical structure as [RFC8366]. Some sections are left out entirely. Additional sections have been added concerning:

1. Addition of voucher-request specification as defined in [I-D.ietf-anima-bootstrapping-keyinfra],

The CBOR definitions for this constrained voucher format are defined using the mechanism describe in [I-D.ietf-core-yang-cbor] using the SID mechanism explained in [I-D.ietf-core-sid]. As the tooling to convert YANG documents into an list of SID keys is still in its infancy, the table of SID values presented here should be considered normative rather than the output of the pyang tool.

Two methods of signing the resulting CBOR object are described in this document:

1. One is CMS [RFC5652].
2. The other is COSE_Sign1 [RFC8152] objects.

2. Terminology

The following terms are defined in [RFC8366], and are used identically as in that document: artifact, imprint, domain, Join Registrar/Coordinator (JRC), Manufacturer Authorized Signing Authority (MASA), pledge, Trust of First Use (TOFU), and Voucher.

3. Requirements Language

In this document, the key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" are to be interpreted as described in BCP 14, RFC 2119 [RFC2119] and indicate requirement levels for compliant STuPiD implementations.

4. Survey of Voucher Types

[RFC8366] provides for vouchers that assert proximity, that authenticate the registrar and that include different amounts of anti-replay protection.

This document does not make any extensions to the types of vouchers.

Time based vouchers are included in this definition, but given that constrained devices are extremely unlikely to know the correct time, their use is very unlikely. Most users of these constrained vouchers will be online and will use live nonces to provide anti-replay protection.

[RFC8366] defined only the voucher artifact, and not the Voucher Request artifact, which was defined in [I-D.ietf-anima-bootstrapping-keyinfra].
This document defines both a constrained voucher and a constrained voucher-request. They are presented in the order voucher-request, followed by voucher response as this is the time order that they occur.

This document defines both CMS-signed voucher requests and responses, and COSE signed voucher requests and responses. The use of CMS signatures implies the use of PKIX format certificates. The pinned-domain-cert present in such a voucher, is the certificate of the Registrar.

The constrained voucher and constrained voucher request MUST be signed.

The use of the two signing formats permit the use of both PKIX format certificates, and also raw public keys (RPK). When RPKs are used, the voucher produced by the MASA pins the raw public key of the Registrar: the pinned-domain-subject-public-key-info in such a voucher, is the raw public key of the Registrar. This is described in the YANG definition for the constrained voucher.

5. Discovery and URI

This section describes the BRSKI extensions to EST-coaps [I-D.ietf-ace-coap-est] to transport the voucher between registrar, proxy and pledge over CoAP. The extensions are targeted to low-resource networks with small packets. Saving header space is important and the EST-coaps URI is shorter than the EST URI.

The presence and location of (path to) the management data are discovered by sending a GET request to "/.well-known/core" including a resource type (RT) parameter with the value "ace.est" [RFC6690]. Upon success, the return payload will contain the root resource of the EST resources. It is up to the implementation to choose its root resource; throughout this document the example root resource /est is used.

The EST-coaps server URIs differ from the EST URI by replacing the scheme https by coaps and by specifying shorter resource path names:

    coaps://www.example.com/est/short-name

Figure 5 in section 3.2.2 of [RFC7030] enumerates the operations and corresponding paths which are supported by EST. Table 1 provides the mapping from the BRSKI extension URI path to the EST-coaps URI path.
Table 1: BRSKI path to EST-coaps path

<table>
<thead>
<tr>
<th>BRSKI</th>
<th>EST-coaps</th>
</tr>
</thead>
<tbody>
<tr>
<td>/requestvoucher</td>
<td>/rv</td>
</tr>
<tr>
<td>/voucher-status</td>
<td>/vs</td>
</tr>
<tr>
<td>/enrollstatus</td>
<td>/es</td>
</tr>
<tr>
<td>/requestauditlog</td>
<td>/ra</td>
</tr>
</tbody>
</table>

/requestvoucher and /enrollstatus are needed between pledge and Registrar.

When discovering the root path for the EST resources, the server MAY return the full resource paths and the used content types. This is useful when multiple content types are specified for EST-coaps server. For example, the following more complete response is possible.

REQ: GET /.well-known/core?rt=ace.est*

RES: 2.05 Content

</est>; rt="ace.est"
</est/rv>; rt="ace.est/rv"; ct=TBD2 TBD3
</est/vs>; rt="ace.est/vs"; ct=50 60
</est/es>; rt="ace.est/es"; ct=50 60
</est/ra>; rt="ace.est/ra"; ct=TBD2 TBD3

The return of the content-types allows the client to choose the most appropriate one from multiple content types.

Port numbers, not returned in the example, are assumed to be the default numbers 5683 and 5684 for coap and coaps respectively (sections 12.6 and 12.7 of [RFC7252]). Discoverable port numbers MAY be returned in the <href> of the payload.

ct=TBD2 stands for Content-Format "application/voucher-cms+cbor, and ct=TBD3 stands for Content-Format "application/voucher-cose+cbor".

Content-Formats TBD2 and TBD3 are defined in this document.

The Content-Format ("application/json") 50 MAY be supported. Content-Formats ("application/cbor") 60, TBD2, and TBD3 MUST be supported.
6. Artifacts

This section describes the abstract (tree) definition as explained in [I-D.ietf-netmod-yang-tree-diagrams] first. This provides a high-level view of the contents of each artifact.

Then the assigned SID values are presented. These have been assigned using the rules in [I-D.ietf-core-sid], with an allocation that was made via the http://comi.space service.

6.1. Voucher Request artifact

6.1.1. Tree Diagram

The following diagram is largely a duplicate of the contents of [RFC8366], with the addition of proximity-registrar-subject-public-key-info, proximity-registrar-cert, and prior-signed-voucher-request.

prior-signed-voucher-request is only used between the Registrar and the MASA. proximity-registrar-subject-public-key-info replaces proximity-registrar-cert for the extremely constrained cases.
module: ietf-constrained-voucher-request

grouping voucher-request-constrained-grouping
  +-- voucher
  |     +-- created-on?
  |        |     yang:date-and-time
  |     +-- expires-on?
  |        |     yang:date-and-time
  |     +-- assertion
  |        |     enumeration
  |     +-- serial-number
  |        |     string
  |     +-- idevid-issuer?
  |        |     binary
  |     +-- pinned-domain-cert?
  |        |     binary
  |     +-- domain-cert-revocation-checks?
  |        |     boolean
  |     +-- nonce?
  |        |     binary
  |     +-- last-renewal-date?
  |        |     yang:date-and-time
  |     +-- proximity-registrar-subject-public-key-info?
  |        |     binary
  |     +-- proximity-registrar-sha256-of-subject-public-key-info?
  |        |     binary
  |     +-- proximity-registrar-cert?
  |        |     binary
  |     +-- prior-signed-voucher-request?
  |        |     binary

6.1.2. SID values
Base SID value for voucher request: 1001150.

SID Assigned to
--------- --------------------------------------------------
1001167 module ietf-constrained-voucher-request
1001168 module ietf-restconf
1001169 module ietf-voucher
1001170 module ietf-yang-types
1001171 data /ietf-constrained-voucher-request:voucher
1001154 data .../ietf-constrained-voucher-request:voucher
1001155 data .../assertion
1001156 data .../created-on
1001157 data .../domain-cert-revocation-checks
1001158 data .../expires-on
1001159 data .../idevid-issuer
1001160 data .../last-renewal-date
1001161 data .../nonce
1001162 data .../pinned-domain-cert
1001165 data .../prior-signed-voucher-request
1001166 data .../proximity-registrar-cert
1001163 data .../proximity-registrar-subject-public-key-info
1001164 data .../serial-number
1001172 data .../assertion
1001173 data .../created-on
1001174 data .../domain-cert-revocation-checks
1001175 data .../expires-on
1001176 data .../idevid-issuer
1001177 data .../last-renewal-date
1001178 data /ietf-constrained-voucher-request:voucher/nonce
1001179 data .../pinned-domain-cert
1001180 data .../prior-signed-voucher-request
1001181 data .../proximity-registrar-cert
1001182 data .../proximity-registrar-subject-public-key-info
1001183 data .../serial-number
1001150 data ietf-constrained-voucher-request
1001151 data ietf-restconf
1001152 data ietf-voucher
1001153 data ietf-yang-types

WARNING, obsolete definitions

6.1.3. YANG Module

In the constrained-voucher-request YANG module, the voucher is "augmented" within the "used" grouping statement such that one continuous set of SID values is generated for the constrained-
voucher-request module name, all voucher attributes, and the
constrained-voucher-request attribute. Two attributes of the voucher
are "refined" to be optional.

<CODE BEGINS> file "ietf-constrained-voucher-request@2018-09-01.yang"
module ietf-constrained-voucher-request {
    yang-version 1.1;

    namespace
    prefix "constrained";

    import ietf-restconf {
        prefix rc;
        description
            "This import statement is only present to access
            the yang-data extension defined in RFC 8040.";
        reference "RFC 8040: RESTCONF Protocol";
    }

    import ietf-voucher {
        prefix "v";
    }

    organization
        "IETF ANIMA Working Group";

    contact
        *WG Web:  <http://tools.ietf.org/wg/anima/>
        WG List:  <mailto:anima@ietf.org>
        Author:  Michael Richardson
            <mailto:mcr+ietf@sandelman.ca>
        Author:  Peter van der Stok
            <mailto: consultancy@vanderstok.org>
        Author:  Panos Kampanakis
            <mailto: pkampana@cisco.com>";
    description
        "This module defines the format for a voucher request,
        which is produced by a pledge to request a voucher.
        The voucher-request is sent to the potential owner’s
        Registrar, which in turn sends the voucher request to
        the manufacturer or delegate (MASA).

        A voucher is then returned to the pledge, binding the
        pledge to the owner. This is a constrained version of the
        voucher-request present in
draft-ietf-anima-bootstrap-keyinfra.txt."
This version provides a very restricted subset appropriate for very constrained devices.
In particular, it assumes that nonce-ful operation is always required, that expiration dates are rather weak, as no clocks can be assumed, and that the Registrar is identified by a pinned Raw Public Key.


revision "2018-09-01" {
  description
    "Initial version"
  reference
    "RFC XXXX: Voucher Profile for Constrained Devices"
}

rc:yang-data voucher-request-constrained-artifact {
  // YANG data template for a voucher.
  uses voucher-request-constrained-grouping;
}

// Grouping defined for future usage
grouping voucher-request-constrained-grouping {
  description
    "Grouping to allow reuse/extensions in future work."

  uses v:voucher-artifact-grouping {

    refine voucher/created-on {
      mandatory false;
    }

    refine voucher/pinned-domain-cert {
      mandatory false;
    }

    augment "voucher" {
      description "Base the constrained voucher-request upon the regular one";

      leaf proximity-registrar-subject-public-key-info {
        type binary;
        description
          "The proximity-registrar-subject-public-key-info replaces
the proximit-registrar-cert in constrained uses of the voucher-request.
The proximity-registrar-subject-public-key-info is the Raw Public Key of the Registrar. This field is encoded as specified in RFC7250, section 3. The ECDSA algorithm MUST be supported. The EdDSA algorithm as specified in draft-ietf-tls-rfc4492bis-17 SHOULD be supported. Support for the DSA algorithm is not recommended. Support for the RSA algorithm is MAY, but due to size is discouraged.;
}

leaf proximity-registrar-sha256-of-subject-public-key-info { type binary; description "The proximity-registrar-sha256-of-subject-public-key-info is an alternative to proximity-registrar-subject-public-key-info. and pinned-domain-cert. In many cases the public key of the domain has already been transmitted during the key agreement protocol, and it is wasteful to transmit the public key another two times. The use of a hash of public key info, at 32-bytes for sha256 is a significant savings compared to an RSA public key, but is only a minor savings compared to a 256-bit ECDSA public-key. Algorithm agility is provided by extensions to this specifications which define new leaf for other hash types.";
}

leaf proximity-registrar-cert { type binary; description "An X.509 v3 certificate structure as specified by RFC 5280, Section 4 encoded using the ASN.1 distinguished encoding rules (DER), as specified in ITU-T X.690.

The first certificate in the Registrar TLS server certificate_list sequence (see [RFC5246]) presented by the Registrar to the Pledge. This MUST be populated in a Pledge’s voucher request if the proximity assertion is populated.";
}

leaf prior-signed-voucher-request {
type binary;
description
  "If it is necessary to change a voucher, or re-sign and
  forward a voucher that was previously provided along a
  protocol path, then the previously signed voucher
  SHOULD be included in this field.

  For example, a pledge might sign a proximity voucher,
  which an intermediate registrar then re-signs to
  make its own proximity assertion. This is a simple
  mechanism for a chain of trusted parties to change a
  voucher, while maintaining the prior signature
  information.

  The pledge MUST ignore all prior voucher information
  when accepting a voucher for imprinting. Other
  parties MAY examine the prior signed voucher
  information for the purposes of policy decisions.
  For example this information could be useful to a
  MASA to determine that both pledge and registrar
  agree on proximity assertions. The MASA SHOULD
  remove all prior-signed-voucher-request information when
  signing a voucher for imprinting so as to minimize the
  final voucher size."

6.1.4. Example voucher request artifact

Below a CBOR serialization of the constrained-voucher-request is
shown in diagnostic CBOR notation. The enum value of the assertion
field is calculated to be zero by following the algorithm described
in section 9.6.4.2 of [RFC7950].
6.2. Voucher artifact

The voucher’s primary purpose is to securely assign a pledge to an owner. The voucher informs the pledge which entity it should consider to be its owner.

This document defines a voucher that is a CBOR encoded instance of the YANG module defined in Section 5.3 that has been signed with CMS or with COSE.

6.2.1. Tree Diagram

The following diagram is largely a duplicate of the contents of [RFC8366], with only the addition of pinned-domain-subject-public-key-info.
module: ietf-constrained-voucher

grouping voucher-constrained-grouping
  +-- voucher
    +-- created-on?
      |       yang:date-and-time
    +-- expires-on?
      |       yang:date-and-time
    +-- assertion
      +-- serial-number
      +-- idevid-issuer?
      +-- pinned-domain-cert?
      +-- domain-cert-revocation-checks?
      +-- nonce?
    +-- last-renewal-date?
      |       yang:date-and-time
    +-- pinned-domain-subject-public-key-info?
      +-- pinned-sha256-of-subject-public-key-info?

6.2.2. SID values

Base SID value for voucher request: 1001101.

SID Assigned to
-------------- ---------------------------------------------
1001115 module ietf-constrained-voucher
1001116 module ietf-restconf
1001117 module ietf-voucher
1001118 module ietf-yang-types
1001119 data /ietf-constrained-voucher:voucher
1001104 data .../ietf-constrained-voucher:voucher
1001105 data .../assertion
1001106 data .../created-on
1001107 data .../domain-cert-revocation-checks
1001108 data .../expires-on
1001109 data .../idevid-issuer
1001110 data .../last-renewal-date
1001111 data .../nonce
1001112 data .../pinned-domain-cert
1001113 data .../pinned-domain-subject-public-key-info
1001114 data .../serial-number

6.2.3. YANG Module

In the constraine-voucher YANG module, the voucher is "augmented" within the "used" grouping statement such that one continuous set of SID values is generated for the constrained-voucher module name, all
voucher attributes, and the constrained-voucher attribute. Two attributes of the voucher are "refined" to be optional.

<CODE BEGINS> file "ietf-constrained-voucher@2018-09-01.yang"
module ietf-constrained-voucher {  
yang-version 1.1;

namespace  
"urn:ietf:params:xml:ns:yang:ietf-constrained-voucher";
prefix "constrained";

import ietf-restconf {  
prefix rc;

description  
"This import statement is only present to access the yang-data extension defined in RFC 8040.";
reference "RFC 8040: RESTCONF Protocol";
}

import ietf-voucher {  
prefix "v";
}

organization  
"IETF ANIMA Working Group";

contact  
"WG Web: <http://tools.ietf.org/wg/anima/>  
WG List: <mailto:anima@ietf.org>  
Author: Michael Richardson <mailto:mcr+ietf@sandelman.ca>  
Author: Peter van der Stok <mailto: consultancy@vanderstok.org>  
Author: Panos Kampanakis <mailto: pkampana@cisco.com>";

description  
"This module defines the format for a voucher, which is produced by a pledge’s manufacturer or delegate (MASA) to securely assign one or more pledges to an ‘owner’, so that the pledges may establish a secure connection to the owner’s network infrastructure.

This version provides a very restricted subset appropriate for very constrained devices. In particular, it assumes that nonce-ful operation is always required, that expiration dates are rather weak, as no clocks can be assumed, and that the Registrar is identified by a pinned Raw Public Key."
The key words 'MUST', 'MUST NOT', 'REQUIRED', 'SHALL', 'SHALL NOT', 'SHOULD', 'SHOULD NOT', 'RECOMMENDED', 'MAY', and 'OPTIONAL' in the module text are to be interpreted as described in RFC 2119.

revision "2018-09-01" {
  description
  "Initial version";
  reference
  "RFC XXXX: Voucher Profile for Constrained Devices";
}

rc:yang-data voucher-constrained-artifact {
  // YANG data template for a voucher.
  uses voucher-constrained-grouping;
}

// Grouping defined for future usage
grouping voucher-constrained-grouping {
  description
  "Grouping to allow reuse/extensions in future work.";

  uses v:voucher-artifact-grouping {

    refine voucher/created-on {
      mandatory false;
    }

    refine voucher/pinned-domain-cert {
      mandatory false;
    }

    augment "voucher" {
      description "Base the constrained voucher
      upon the regular one";

      leaf pinned-domain-subject-public-key-info {
        type binary;
        description
          "The pinned-domain-subject-public-key-info replaces the
          pinned-domain-cert in constrained uses of
          the voucher. The pinned-domain-subject-public-key-info
          is the Raw Public Key of the Registrar.
          This field is encoded as specified in RFC7250,
          section 3.
          The ECDSA algorithm MUST be supported.
          The EdDSA algorithm as specified in
draft-ietf-tls-rfc4492bis-17 SHOULD be supported.
          Support for the DSA algorithm is not recommended.";
    }
  }
}
Support for the RSA algorithm is a MAY.

leaf pinned-sha256-of-subject-public-key-info {
  type binary;
  description "The pinned-hash-subject-public-key-info is a second alternative to pinned-domain-cert. In many cases the public key of the domain has already been transmitted during the key agreement process, and it is wasteful to transmit the public key another two times. The use of a hash of public key info, at 32-bytes for sha256 is a significant savings compared to an RSA public key, but is only a minor savings compared to a 256-bit ECDSA public-key. Algorithm agility is provided by extensions to this specifications which define new leaf for other hash types";
}

6.2.4. Example voucher artifacts

Below a the CBOR serialization of the the constrained-voucher is shown in diagnostic CBOR notation. The enum value of the assertion field is calculated to be zero by following the algorithm described in section 9.6.4.2 of [RFC7950].
The signing of the example is shown in Appendix B.1.

6.3. Signing voucher and voucher-request artifacts

6.3.1. CMS signing

The IETF evolution of PKCS#7 is CMS [RFC5652]. The CMS signed voucher is much like the equivalent voucher defined in [RFC8366].

A different eContentType of TBD1 is used to indicate that the contents are in a different format than in [RFC8366].

The ContentInfo structure contains a payload consisting of the CBOR encoded voucher. The [I-D.ietf-core-yang-cbor] use of delta encoding creates a canonical ordering for the keys on the wire. This canonical ordering is not important as there is no expectation that the content will be reproduced during the validation process.

Normally the recipient is the pledge and the signer is the MASA.

[I-D.ietf-anima-bootstrapping-keyinfra] supports both signed and unsigned voucher requests from the pledge to the JRC. In this specification, voucher-request artifact MUST be signed from the pledge to the registrar. From the JRC to the MASA, the voucher-request artifact MUST be signed by the domain owner key which is requesting ownership.

The considerations of [RFC5652] section 5.1, concerning validating CMS objects which are really PKCS7 objects (cmsVersion=1) applies.
The CMS structure SHOULD also contain all the certificates leading up to and including the signer’s trust anchor certificate known to the recipient. The inclusion of the trust anchor is unusual in many applications, but without it third parties can not accurately audit the transaction.

The CMS structure MAY also contain revocation objects for any intermediate certificate authorities (CAs) between the voucher-issuer and the trust anchor known to the recipient. However, the use of CRLs and other validity mechanisms is discouraged, as the pledge is unlikely to be able to perform online checks, and is unlikely to have a trusted clock source. As described below, the use of short-lived vouchers and/or pledge provided nonce provides a freshness guarantee.

[EDnote: compulsory signing algorithms are ....]

In Appendix B.1 an example for the CMS signing of the voucher-request is shown.

6.3.2. COSE signing

The COSE-Sign1 structure is discussed in section 4.2 of [RFC8152]. The CBOR object that carries the body, the signature, and the information about the body and signature is called the COSE_Sign1 structure. It is used when only one signature is used on the body. Support for EDdsa 256 with Ed25519 is compulsory.

The supported COSE-sign1 object structure is shown in Figure 1.

COSE_Sign1(
    [
        h’a10126’, #{ "alg": EDdsa 256 }
        { "crv": Ed25519,
          "kty": OKP,
          "key_ops": "verify"
        },
        h’123’, #voucher-request binary content
        h’456’, #voucher-request binary public signature
    ]
)

Figure 1: The cose-sign1 structure.

The [COSE-registry] specifies the integers that replace the strings and the mnemonics in Figure 1. In Appendix C a binary cose-sign1 object is shown based on the voucher-request example of Section 6.1.4.
7. Design Considerations

The design considerations for the CBOR encoding of vouchers is much the same as for [RFC8366].

One key difference is that the names of the leaves in the YANG does not have a material effect on the size of the resulting CBOR, as the SID translation process assigns integers to the names.

8. Security Considerations

8.1. Clock Sensitivity

TBD.

8.2. Protect Voucher PKI in HSM

TBD.

8.3. Test Domain Certificate Validity when Signing

TBD.

9. IANA Considerations

9.1. Resource Type Registry

Additions to the sub-registry "CoAP Resource Type", within the "CoRE parameters" registry are specified below. These can be registered either in the Expert Review range (0-255) or IETF Review range (256-9999).

- ace.rt.rv needs registration with IANA
- ace.rt.vs needs registration with IANA
- ace.rt.es needs registration with IANA
- ace.rt.ra needs registration with IANA

9.2. The IETF XML Registry

This document registers two URIs in the IETF XML registry [RFC3688]. Following the format in [RFC3688], the following registration is requested:
9.3. The YANG Module Names Registry

This document registers two YANG modules in the YANG Module Names registry [RFC6020]. Following the format defined in [RFC6020], the following registration is requested:

| prefix:       | vch     |               |
| reference:    | RFC XXXX|               |

| prefix:       | vch     |               |
| reference:    | RFC XXXX|               |

9.4. The RFC SID range assignment sub-registry

<table>
<thead>
<tr>
<th>Entry-point</th>
<th>Size</th>
<th>Module name</th>
<th>RFC Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>1001100</td>
<td>50</td>
<td>ietf-constrained-voucher</td>
<td>[ThisRFC]</td>
</tr>
<tr>
<td>1001150</td>
<td>50</td>
<td>ietf-constrained-voucher-request</td>
<td>[ThisRFC]</td>
</tr>
</tbody>
</table>

Warning: These SID values will change when they transfer to the range 1000 - 59,999 allocated for SIDs in YANG modules defined in RFCs.

9.5. The SMI Security for S/MIME CMS Content Type Registry

This document registers an OID in the "SMI Security for S/MIME CMS Content Type" registry (1.2.840.113549.1.9.16.1), with the value:

<table>
<thead>
<tr>
<th>Decimal</th>
<th>Description</th>
<th>References</th>
</tr>
</thead>
<tbody>
<tr>
<td>TBD1</td>
<td>id-ct-animaCBORVoucher</td>
<td>[ThisRFC]</td>
</tr>
</tbody>
</table>

EDNOTE: should a separate value be used for Voucher Requests?
9.6. Media-Type Registry

This section registers the ‘application/voucher-cms+cbor’ media type and the ‘application/voucher-cose+cbor’ in the “Media Types” registry. These media types are used to indicate that the content is a CBOR voucher either signed with a cms structure or a COSE_Sign1 structure [RFC8152].

9.6.1. application/voucher-cms+cbor

Type name: application
Subtype name: voucher-cms+cbor
Required parameters: none
Optional parameters: none
Encoding considerations: CMS-signed CBOR vouchers are CBOR encoded.
Security considerations: See Security Considerations, Section
Interoperability considerations: The format is designed to be broadly interoperable.
Published specification: THIS RFC.
Applications that use this media type: ANIMA, 6tisch, and other zero-touch imprinting systems
Additional information:
  Magic number(s): None
  File extension(s): .vch
  Macintosh file type code(s): none
Person & email address to contact for further information: IETF ANIMA WG
Intended usage: LIMITED
Restrictions on usage: NONE
Author: ANIMA WG
Change controller: IETF
Provisional registration? (standards tree only): NO

9.6.2. application/voucher-cose+cbor
Type name: application
Subtype name: voucher-cose+cbor
Required parameters: none
Optional parameters: cose-type
Encoding considerations: COSE_Sign1 CBOR vouchers are COSE objects signed with one signer.
Security considerations: See Security Considerations, Section
Interoperability considerations: The format is designed to be broadly interoperable.
Published specification: THIS RFC.
Applications that use this media type: ANIMA, 6tisch, and other zero-touch imprinting systems
Additional information:
  Magic number(s): None
  File extension(s): .vch
  Macintosh file type code(s): none
Person & email address to contact for further information: IETF ANIMA WG
Intended usage: LIMITED
Restrictions on usage: NONE
Author: ANIMA WG
Change controller: IETF
Provisional registration? (standards tree only): NO

9.7. CoAP Content-Format Registry

Additions to the sub-registry "CoAP Content-Formats", within the "CoRE Parameters" registry are needed for two media types. These can be registered either in the Expert Review range (0-255) or IETF Review range (256-9999).

<table>
<thead>
<tr>
<th>Media type</th>
<th>mime type:</th>
<th>Encoding</th>
<th>ID</th>
<th>References</th>
</tr>
</thead>
<tbody>
<tr>
<td>application/voucher-cms+cbor</td>
<td></td>
<td>CBOR</td>
<td>TBD2</td>
<td>[This RFC]</td>
</tr>
<tr>
<td>application/voucher-cose+cbor</td>
<td>&quot;COSE-Sign1&quot;</td>
<td>CBOR</td>
<td>TBD3</td>
<td>[This RFC]</td>
</tr>
</tbody>
</table>

10. Acknowledgements

We are very grateful to Jim Schaad for explaining COSE and CMS choices.

Michel Veillette did extensive work on pyang to extend it to support the SID allocation process, and this document was among the first users.

We are grateful for the suggestions done by Esko Dijk.
11. Changelog

-04 voucher and request-voucher MUST be signed examples for signed request are added in appendix IANA SID registration is updated SID values in examples are aligned signed cms examples aligned with new SIDs

-03

Examples are inverted.

-02

Example of requestvoucherc with unsigned appllication/cbor is added attributes of voucher "refined" to optional CBOR serialization of vouchers improved Discovery port numbers are specified

-01

application/json is optional, application/cbor is compulsory Cms and cose mediatypes are introduced

12. References

12.1. Normative References

[I-D.ietf-ace-cbor-web-token]

[I-D.ietf-ace-coap-est]

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

[I-D.ietf-core-object-security]


12.2. Informative References

[COSE-registry]
IANA, ., "CBOR Object Signing and Encryption (COSE) registry", 2017,
<https://www.iana.org/assignments/cose/cose.xhtml>.

[duckling]

[I-D.ietf-netmod-yang-tree-diagrams]
Bjorklund, M. and L. Berger, "YANG Tree Diagrams", draft-ietf-netmod-yang-tree-diagrams-06 (work in progress),
February 2018.


DOI 10.17487/RFC7030, October 2013,

Appendix A. EST messages to EST-coap

This section extends the examples from Appendix A of
[I-D.ietf-ace-coap-est]. The CoAP headers are only worked out for
the enrollstatus example.
A.1. enrollstatus

A coaps enrollstatus message can be:

GET coaps://[192.0.2.1:8085]/est/es

The corresponding coap header fields are shown below.

Ver = 1
T = 0 (CON)
Code = 0x01 (0.01 is GET)
Options
  Option (Uri-Path)
    Option Delta = 0xb   (option nr = 11)
    Option Length = 0x3
    Option Value = "est"
  Option (Uri-Path)
    Option Delta = 0x0   (option nr = 11)
    Option Length = 0x2
    Option Value = "es"
Payload = [Empty]

The Uri-Host and Uri-Port Options are omitted because they coincide
with the transport protocol destination address and port
respectively.

A 2.05 Content response with an unsigned voucher status (ct=60) will
then be:

  2.05 Content (Content-Format: application/cbor)

With CoAP fields and payload:
Ver=1
T=2 (ACK)
Code = 0x45 (2.05 Content)
Options
  Option1 (Content-Format)
  Option Delta = 0xC  (option nr 12)
  Option Length = 0x2
  Option Value = 60 (application/cbor)

Payload (CBOR diagnostic) =
{
  "version":"1",
  "Status": 1,   / 1 = Success, 0 = Fail   
  "Reason":"Informative human readable message",
  "reason-context": "Additional information"
}

Payload (binary) =
A46776657273696F66E6131665374617475730166526561736F6E7822
496E666F726D61746976652068756D616E207265616461626C65206D
6573736167656e726561736F6E74657874
7641646974696F66E666F726D6174696F6E

###voucher_status

A coaps voucher_status message can be:

GET coaps://[2001:db8::2:1]:61616]/est/vs ~~~~

A 2.05 Content response with a non signed CBOR voucher (ct=60) will then be:

  2.05 Content (Content-Format: application/cbor)
  Payload =
  A46776657273696F66E6131665374617475730166526561736F6E7822
  496E666F726D61746976652068756D616E207265616461626C65206D
  6573736167656e726561736F6E74657874
  7641646974696F66E666F726D6174696F6E

A.2. requestvoucher

Signed request-voucher-request payloads are sent from pledge to Registrar, as explained in Section 5.2 of [I-D.ietf-anima-bootstrapping-keyinfra].
A.2.1. signed requestvoucher

A CMS signed requestvoucher message from JRC to MASA is shown below. It would be CoAP POSTED to /est/rv.

```
POST coaps://[2001:db8::2:1]:61616]/est/rv
(Content-Format: application/voucher-cms+cbor)
```

The payload would be in binary, but is presented in base64 in this document.

```
MIIDugYJKoZIhvcNAQcCoIIDqzCCA6cCAQEXDTALBglghkgBZQMEAgEwCyYJ
KOZIHvcNAQcBoIICQCTCCAj0vsgHiocAMCAQICCH52Yde1TkYyNAAoGCCqGSM49
BAMCMF0xCzAJBgNVBAYTA1VTMQswCQYDVQQIDAJDQTEUMBIGA1UECgwLRXhh
bXBS2SBJbmFjxTbAUBgNVBAsMDWN1cnRpZmljYXRpb24xExARbgNVBAAMYmIw
Mi4xQIVOewIbN0MjMEYyOg12WhPoTk5OETeMyMzEwMzU5NTIaMFwx
CzAJBgNVBAYTA1VTMQswCQYDVQQIDAJDQTEUMBIGA1UEBwCNcmtvdwECEASBGNV
BAoMC2V4YW1wbGUwGyS5jMQwwCgYDVQQIDANJbQxZANBgNVBAUTBkdOMTlZ
NDBZMBMGykgSM49AeEGCCqGSM49AwEHA0IABMiO1fEcJeR+OsVix78tn9xJ
TwKx1hMGMA/FQv1DF+VjXVbVygN0kXf+ueQyVPoYdfYC+RUmGgKorI7Vj1
n9mgjYowgYcwCQYDVOR0TBA1wADADBgNVH4EgQUlmAnhxa/f9DnUtCsggd3
rWZdAqAwHwYDVv0jBBgwFoAIaAwF1uf1Rv/8gqQx0Nnwi8LSBbEwAwDgYVRO
AQk/BQDAGwGwMoGCAg1EdEQjMQCgHgYWkYWBQUHCAsgZeABgbrBgHEAaQ7
CgEEBAEACwQgCgYIKoZIzj0EAsIQWg1hAMCDBZbbSU1pF5xI6qXu1J9G
ptshQjNgRgGxYyTDMA2GAEAP3SYn0wGlzyXMqTNqC1n3yDuUGqHGeK
3m0kijXyggE/M1IIBQW1BTBMPF0xOJZARbNGVBAf1VTMQsCQYDVQQIDAJD
QTEUMBIGA1UECgwLRXhhXB2SBJbmMxJyAUBgNVBAsMDWN1cnRpZmljYXRp
b24xEzARbgNVBAAMYmIwMi4xQIVOewIbN0MjMEYyOg12WhPoTk5OETeMyMz
EwMzU5NTI1MFwx
CzAJBgNVBAYTA1VTMQswCQYDVQQIDAJDQTEUMBIGA1UEBwCNcmtvdwECEASBGNV
BAoMC2V4YW1wbGUwGyS5jMQwwCgYDVQQIDANJbQxZANBgNVBAUTBkdOMTlZ
NDBZMBMGykgSM49AeEGCCqGSM49AwEHA0IABMiO1fEcJeR+OsVix78tn9xJ
TwKx1hMGMA/FQv1DF+VjXVbVygN0kXf+ueQyVPoYdfYC+RUmGgKorI7Vj1
n9mgjYowgYcwCQYDVOR0TBA1wADADBgNVH4EgQUlmAnhxa/f9DnUtCsggd3
rWZdAqAwHwYDVv0jBBgwFoAIaAwF1uf1Rv/8gqQx0Nnwi8LSBbEwAwDgYVRO
AQk/BQDAGwGwMoGCAg1EdEQjMQCgHgYWkYWBQUHCAsgZeABgbrBgHEAaQ7
CgEEBAEACwQgCgYIKoZIzj0EAsIQWg1hAMCDBZbbSU1pF5xI6qXu1J9G
ptshQjNgRgGxYyTDMA2GAEAP3SYn0wGlzyXMqTNqC1n3yDuUGqHGeK
3m0kijXyggE/M1IIBQW1BTBMPF0xOJZARbNGVBAf1VTMQsCQYDVQQIDAJD
QTEUMBIGA1UECgwLRXhhXB2SBJbmMxJyAUBgNVBAsMDWN1cnRpZmljYXRp
b24xEzARbgNVBAAMYmIwMi4xQIVOewIbN0MjMEYyOg12WhPoTk5OETeMyMz
EwMzU5NTI1MFwx
CzAJBgNVBAYTA1VTMQswCQYDVQQIDAJDQTEUMBIGA1UEBwCNcmtvdwECEASBGNV
BAoMC2V4YW1wbGUwGyS5jMQwwCgYDVQQIDANJbQxZANBgNVBAUTBkdOMTlZ
NDBZMBMGykgSM49AeEGCCqGSM49AwEHA0IABMiO1fEcJeR+OsVix78tn9xJ
TwKx1hMGMA/FQv1DF+VjXVbVygN0kXf+ueQyVPoYdfYC+RUmGgKorI7Vj1
n9mgjYowgYcwCQYDVOR0TBA1wADADBgNVH4EgQUlmAnhxa/f9DnUtCsggd3
rWZdAqAwHwYDVv0jBBgwFoAIaAwF1uf1Rv/8gqQx0Nnwi8LSBbEwAwDgYVRO
AQk/BQDAGwGwMoGCAg1EdEQjMQCgHgYWkYWBQUHCAsgZeABgbrBgHEAaQ7
CgEEBAEACwQgCgYIKoZIzj0EAsIQWg1hAMCDBZbbSU1pF5xI6qXu1J9G
ptshQjNgRgGxYyTDMA2GAEAP3SYn0wGlzyXMqTNqC1n3yDuUGqHGeK
3m0kijXyggE/M1IIBQW1BTBMPF0xOJZARbNGVBAf1VTMQsCQYDVQQIDAJD
QTEUMBIGA1UECgwLRXhhXB2SBJbmMxJyAUBgNVBAsMDWN1cnRpZmljYXRp
b24xEzARbgNVBAAMYmIwMi4xQIVOewIbN0MjMEYyOg12WhPoTk5OETeMyMz
EwMzU5NTI1MFwx
CzAJBgNVBAYTA1VTMQswCQYDVQQIDAJDQTEUMBIGA1UEBwCNcmtvdwECEASBGNV
BAoMC2V4YW1wbGUwGyS5jMQwwCgYDVQQIDANJbQxZANBgNVBAUTBkdOMTlZ
NDBZMBMGykgSM49AeEGCCqGSM49AwEHA0IABMiO1fEcJeR+OsVix78tn9xJ
```

A 2.04 Changed response returning CBOR voucher signed with a cms structure(ct=TBD2) will then be:

```
2.04 Changed (Content-Format: application/voucher-cms+cbor)
```
A.3. requestauditing

A COAPS requestauditing message contains the signed CBOR voucher:
A 2.05 Content response returning a log of the voucher (ct=60) will then be:
2.05 Content (Content-Format: application/cbor)

Payload =
{
  "version":"1",
  "events":[
    {
      "date":"<date/time of the entry>",
      "domainID": "<domainID extracted from voucher-request>",
      "nonce": "<any nonce if supplied (or the exact string ‘NULL’)>",
      "assertion": "<the value from the voucher assertion leaf>",
      "truncated": "<the number of domainID entries truncated>",
    },
    {
      "date": "<date/time of the entry>",
      "domainID": "<anotherDomainID extracted from voucher-request>",
      "nonce": "<any nonce if supplied (or the exact string ‘NULL’)>",
      "assertion": "<the value from the voucher assertion leaf>"
    }
  ],
  "truncation": {
    "nonced duplicates": "<total number of entries truncated>",
    "nonceless duplicates": "<total number of entries truncated>",
    "arbitrary": "<number of domainID entries removed entirely>"
  }
}

[EDNOTE: Change JSON to CBOR; Serialize CBOR payload to binary]

Appendix B. Signed voucher-request examples

B.1. CMS signed voucher-request example

The voucher-request example, visualized in CBOR diagnostic notation in Section 6.1.4 is shown as a hexadecimal dump of the binary file.

A11A000F46C2A90274323031362D31302D30375431393A33313A34325A0
474323031362D3102D32315431393A33313A34325A01020d6d4A414441
313233343536373839054010200F0A44010202D0F03F50674323031372
D31302D30375431393A33313A34325A0c4401020D0F

The voucher-request example has been signed by using the WT1234 certificate and key pair shown in Appendix C of [I-D.ietf-ace-coap-est]. The CMS signing of the binary voucher-request leads to a binary signed voucher-request, shown with a hexadecimal representation shown in the payload of the request part of Appendix A.2.1 and Appendix A.3.
The breakdown of the CMS signed binary voucher-request file is visualized below:

**CMS_ContentInfo:**
- `contentType`: pkcs7-signedData (1.2.840.113549.1.7.2)
  - `version`: 1
  - `digestAlgorithms`:
    - `algorithm`: sha256 (2.16.840.1.101.3.4.2.1)
    - `parameter`: <ABSENT>
  - `encapContentInfo`:
    - `eContentType`: pkcs7-data (1.2.840.113549.1.7.1)
    - `eContent`: <ABSENT>
  - `certificates`:
    - `d.certificate`:
      - `cert_info`:
        - `version`: 2
        - `serialNumber`: 9112578475118446130
        - `signature`:
          - `algorithm`: ecdsa-with-SHA256 (1.2.840.10045.4.3.2)
          - `parameter`: <ABSENT>
          - `issuer`: C=US, ST=CA, O=Example Inc, OU=certification, CN=802.1AR CA
        - `validity`:
          - `notBefore`: Jan 31 11:29:16 2019 GMT
          - `notAfter`: Dec 31 23:59:59 9999 GMT
        - `subject`: C=US, ST=CA, L=LA, O=example Inc, OU=IoT/serialNumber=Wt1234
        - `key`:
          - `algorithm`: id-ecPublicKey (1.2.840.10045.2.1)
          - `parameter`: OBJECT:prime256v1 (1.2.840.10045.3.1.7)
          - `public_key`:
            - (0 unused bits)
            - 0000 - 04 c8 b4 21 f1 1c 25 e4-7e 3a c5 71 23 bf
            - 000e - 2d 9f dc 49 4f 02 8b c3-51 cc 80 c0 3f 15
            - 001c - 0b f5 0c ff 95 8d 75 41-9d 81 a6 a2 45 df
            - 002a - fa e7 90 be 95 cf 75 f6-02 f9 15 26 18 f8
            - 0038 - 16 a2 b2 3b 56 38 e5 9f-d9
        - `issuerUID`: <ABSENT>
        - `subjectUID`: <ABSENT>
      - `extensions`:
        - `object`: X509v3 Basic Constraints (2.5.29.19)
          - `critical`: BOOL ABSENT
          - `value`:
            - 0000 - 30
            - 0002 - <SPACES/NULS>
        - `object`: X509v3 Subject Key Identifier (2.5.29.14)
critical: BOOL ABSENT
value:
  0000 – 04 14 96 60 0d 87 16 bf-7f d0 e7 52 d0
  000d – ac 76 07 77 ad 66 5d 02-a0

object: X509v3 Authority Key Identifier (2.5.29.35)
critical: BOOL ABSENT
value:
  0000 – 30 16 80 14 68 d1 65 51-f9 51 bf c8 2a
  000d – 43 1d 0d 9f 08 bc 2d 20-5b 11 60

object: X509v3 Key Usage (2.5.29.15)
critical: TRUE
value:
  0000 – 03 02 05 a0

object: X509v3 Subject Alternative Name (2.5.29.17)
critical: BOOL ABSENT
value:
  0000 – 30 21 a0 1f 06 08 2b 06-01 05 05 07 08
  000d – 04 a0 13 30 11 06 09 2b-06 01 04 01 b4
  001a – 3b 0a 01 04 04 01 02 03-04

sig_alg:
  algorithm: ecdsa-with-SHA256 (1.2.840.10045.4.3.2)
  parameter: <ABSENT>
signature: (0 unused bits)
  0000 – 30 46 02 21 00 c0 d8 19-96 d2 50 7d 69 3f 3c
  000f – 48 ea a5 ee 94 91 bd a6-db 21 40 99 d9 81 17
  001e – c6 3b 36 13 74 cd 86 02-21 00 a7 74 98 9f 4c
  002d – 32 1a 5c f2 5d 83 2a 4d-33 6a 08 ad 67 df 20
  003c – f1 50 64 21 18 8a 0a de-6d 34 92 36
crls: <EMPTY>
signerInfos:
  version: 1
d.issuerAndSerialNumber:
  issuer: C=US, ST=CA, O=Example Inc, OU=certification, CN=802.1AR CA
  serialNumber: 9112578475118446130
digestAlgorithm:
  algorithm: sha256 (2.16.840.1.101.3.4.2.1)
  parameter: <ABSENT>
signedAttrs:
  object: contentType (1.2.840.113549.1.9.3)
  value.set:
    OBJECT:pkcs7-data (1.2.840.113549.1.7.1)
  object: signingTime (1.2.840.113549.1.9.5)
value.set:
  UTCTIME: Jul  3 08:53:30 2019 GMT

object: messageDigest (1.2.840.113549.1.9.4)
value.set:
  OCTET STRING:
    0000 - d4 b0 5c dd c8 b4 91 28-4a 18 ca 25 9d
    000d - be d0 60 23 cf ad a0 aa-c2 95 ac e9 3f
    001a - 0b 4f 44 9e 25
    0020 - <SPACES/NULS>

signatureAlgorithm:
  algorithm: ecdsa-with-SHA256 (1.2.840.10045.4.3.2)
  parameter: <ABSENT>

signature:
  0000 - 30 46 02 21 00 e5 e1 7f-23 c3 aa 14 9f 35 64
  000f - 1e c4 4a 0f 68 fe b0 16-3b e6 7c 06 51 af bf
  001e - 5a a0 99 59 e0 28 1f 02-21 00 b4 07 2f 7c c4
  002d - f9 26 0c 6d 47 a7 93 56-de b8 da f7 23 f0 af
  003c - 2b 59 16 cc 36 63 e7 91-89 39 df df

unsignedAttrs:
  <EMPTY>

Appendix C.  COSE examples

C.1.  Device, Registrar and MASA keys

This first section documents the public and private keys used in the
subsequent test vectors below.  These keys come from test code and
are not used in any production system, and should only be used only
to validate implementations.

C.1.1.  Device IDevID certificate
Certificate:
  Data:
    Version: 3 (0x2)
    Serial Number: 787697345 (0x2ef34ec1)
    Signature Algorithm: ecdsa-with-SHA256
    Issuer: C = Canada, ST = Ontario, OU = Sandelman, CN = highway-test.example.com CA
    Validity
      Not Before: Feb 14 17:05:09 2019 GMT
      Not After : Dec 31 00:00:00 2999 GMT
    Subject Public Key Info:
      Public Key Algorithm: id-ecPublicKey
      Public-Key: (256 bit)
      pub:
        b:f4:
        4d:7e:b0:00:ed:c0:de:bd:4d:25:55:4e:35:f
        9:d5:
        3:3f:
        7:46:
        86:a4:0c:8b:b7
      ASN1 OID: prime256v1
      NIST CURVE: P-256
    X509v3 extensions:
      X509v3 Subject Key Identifier:
        :08:78:OF:7D:52
      X509v3 Basic Constraints:
        CA:FALSE
      X509v3 Subject Alternative Name:
        othername:<unsupported>
        1.3.6.1.4.1.46930.2:
        ..highway-test.example.com:9443
    Signature Algorithm: ecdsa-with-SHA256
    d6:
    30:65:02:31:00:b2:9a:7a:1a:74:20:8f:e9:e0:5d:fc:af:
    71:
    02:
    7d:
    5b:
C.1.2. Device private key

```
-----BEGIN EC PRIVATE KEY-----
MHcCAQEEIA1sa0l4bkj/rJxPUNibKSBNYo1VVzx+T28wo60cYpuaoAoGCCqGSM49
AwEHOuQDQgAEGsQcO3zwNnjhkBtCy/RNfrAA7cDevU01VU41+dVq/VX0s1K/ObVPI
YMLOUz8sG0LxwtfwXs98y1U0aGpAyLtw==
-----END EC PRIVATE KEY-----
```

C.1.3. Registrar Certificate

```
-----BEGIN CERTIFICATE-----
MIIB0TCCAVagAwIBAgIBAjAKBggqhkjOPQQDAzBxMRIwEAYKCZIxmiZPyLQGBRGC
Y2ExGTAXBgoJkiaJk/IsZA2FglzYW5kZXUwYQ4xQDA+BgNVBAMM8U3lzdGVtVmFyaWFs
-----BEGIN EC PRIVATE KEY-----
MHcCAQEEIFZodk+PC5Mu24+ra0sb0ojKzan+dW5rvDAR7YuUOC1YoAoGCCqGSM49
AwEHOuQDQgAElmVQcjS6n+Xd5l/28IFv6UieqQwSBztGj5dkK2MAjQIPV8181H+E
jLIOydbJ110VtEI1/Jqt+TOBfinTNOLOg==
-----END EC PRIVATE KEY-----
```

C.1.4. Registrar private key

```
-----BEGIN CERTIFICATE-----
MIIB3zCCAWSgAwIBAgIEG5lfVDAKBggqhkjOPQDAjbdMQ8wDQYDVQGEwZDYW5h
ZGEkdXDAoBgnVBqMB09udGywa8xeJaqBqNVBAwMCVhbmR1bgIhjEkMCIGAUE
AwwbdaHdhesi10ZNMv4YlwbGU0yZ9tIENMBMB4XDTE5MDIwMi4yMj10MVox
DTIxMDixM1YxMj0MVwuxEMPA0GAIUEhMq2FyYWRhMRAwDgYDVQIDA8DpbnRh
clm1VRIwEAYDVQQLDA1TY5kZXUwY4xkJJAhkgNVBAMHWhp22h3YXktcGVzdC51
eGFTcGxlXNvbSBNQVWMBFkEwEyYHkoZj0CAQYIKoZj0DAcQDQgAEDqQVo0s5
4K74ykb8xumdHcprgsb0rMKmm1n3oB1HA25MVJ+qq14MFJSj0iEw8kszf
WXX4rLqjS2mpnMQA4dDYZV0TAQHBAlwADAKbgqkhj0PQDQAgNpADBmAES
-----END CERTIFICATE-----
```

C.1.5. MASA Certificate

```
-----BEGIN CERTIFICATE-----
MIB3zCCAWSgAwIBAgIEG5lfVDAKBggqhkjOPQDAjbdMQ8wDQYDVQGEwZDYW5h
ZGEkdXDAoBgnVBqMB09udGywa8xeJaqBqNVBAwMCVhbmR1bgIhjEkMCIGAUE
AwwbdaHdhesi10ZNMv4YlwbGU0yZ9tIENMBMB4XDTE5MDIwMi4yMj10MVox
DTIxMDixM1YxMj0MVwuxEMPA0GAIUEhMq2FyYWRhMRAwDgYDVQIDA8DpbnRh
clm1VRIwEAYDVQQLDA1TY5kZXUwY4xkJJAhkgNVBAMHWhp22h3YXktcGVzdC51
eGFTcGxlXNvbSBNQVWMBFkEwEyYHkoZj0CAQYIKoZj0DAcQDQgAEDqQVo0s5
4K74ykb8xumdHcprgsb0rMKmm1n3oB1HA25MVJ+qq14MFJSj0iEw8kszf
WXX4rLqjS2mpnMQA4dDYZV0TAQHBAlwADAKbgqkhj0PQDQAgNpADBmAES
-----END CERTIFICATE-----
```
C.1.6. MASA private key

-----BEGIN EC PRIVATE KEY-----
MHcCAQEEIFhdd0eDdzp67kXx72K+KHGJQYJHNy8pklJL6CvxxMGoAoGCCqGSM49
AwEHoUQDQgAEgqVQo0S54kT4yfkbbBxumdHOCrpsbQOpMKm1Mn3oB1HAWZ5MJV+
gq14tMFJSJo1Ewt8kszfWXK4rLgJS2mnpQ==
-----END EC PRIVATE KEY-----

C.2. COSE signed requestvoucher with registrar certificate pinned

This voucher request has been signed by the pledge, using the private
key given above, and has been sent to the JRC over CoAPS. This
example uses the proximity-registrar-cert mechanism to request a
voucher that pins the certificate of the registrar.

This is the CBOR diagnostic format, folded to 60 characters:

18([h'A0', {}, h'A11A000F46C2A5016970726F78696D69747902C11A5
D1E49970A130302D4432D45352D46322D3032D303307765F715674477
738565342626C65394D34557036354C770C5901D4308201D303820157A00
30201020204228ED27300A06082A8648CE3D040302306E31123010060A0
992268993F22C64011960263613119301706A0992268993E22C6401191
6097361E664656C6D61E331D303B0603350403C3666F756E7461696E2
D746573742E6578616D70C652E636F660A20556E737472756E6720466F7
56E7416969E20526F674204341301E170D313903043136313835341315
A170D3139030351373034353431315A305331123010060A0992268993F22
C64011960263613119301706A0992268993E22C6401191609736161E646
56C6D61E312230200635504030C19666F756E7461696E2D746573742E6
586107D06C652E636F6D3059301306072A8648CE3D020106082A8648CE3
D0301073420049665507234BA9F5E5DDE65FF6F0816FE9489E810C12D73
B468F97642B6300D020F57C97C94F748F8CB20E616C9888D15B4421FD7F
26AB7E74CE05F8A74CD38B3A300A06082A8648CE3D0403023067003064023
0340F4E600F9F702553FA53BE572ACF0ED858275B6AC7594332FB25FB3
A54411E9F0A2ED6F75F1DAADB7EA9A61F5409E02303E615E75C8F70432A59
0CB48D798BEDA1E49E5ED8E0E118BD17A02D02F031D144B16020F756
B528ABD010ADB749D', h'96B82530AC57650346C2BFF5BA6CC16B28F16F
ACFE5A2FD1BCFD5F5D62733F7F7812D67D43BE1CF9906E356FB0C2BD36
77FD7DBEAE22B8CEB07D51D8F55AD3')

This is the raw binary, encoded in base64:
This voucher request has been signed by the JRC using the private key from Appendix C.1.4. Contained within this voucher request is the pledge voucher request above.

This is the CBOR diagnostic format, folded to 60 characters: 0oRBoKBZAhyhGgAPRsKlAWlwc94aW1pdHkCwRpdHkmXC1eMClFNS1GMi0wMC0wMwd2X3FwEd3OFZTqmJsZTlNFFVwNjVmdwxZAdQwggHQMiIBV6ADAgEEAwqio0nMAoGCCqGSM49BAMCMG4xExAQBgoJkiaJk/IsZAEZFgjJYTEZMBcGcmSmJomTSixxARKwCwXNhbmRlbG1hbjE9MDsGAIUEAw0Zm91bnRha8tdGVzdC5eGFtcGxlLmNvbQogVW5zdHJ1bmcgRm91bnRha8tM9vdCBDQTAeFw0xOTA0MTYxODU0MTFaFw0xOTA1MTcwNDU0MTFaMFMrEjAQBgoJkiaJk/IsZAEZFgjJYTEZMBcGcmSmJomTSixxARKwCwXNhbmRlbG1hbjE1MCAGA1UEAwwZZm91bnRha8tdGVzdC5eGFtcGxlLmNvbTBZMBMGByqGSM49AgEGCCqGSM49AwEHA0IABJZl1UI0up/13ezJ9vCBBb+1IinoEMEgc7Ro+XZCtjAI0CdIrfJfJR/hIyyDmHwyYNFbRCH9fyrarkqX4p0zTizowCgYIKoZIzj0EAwIDZwAwZAIwNA90bQ+fcCVT+10+VyrFdu2FgmW2rHWZQzL7JFs6VEEn6Aub3X9Gq236pph9UCeAja+YV51yPB0MqWQyNShmL7aHrSeXn2DqEyvRegQLwMT0USBYAL3VrUoq9GwrbdJ1YQJa4JTCv2UDRsK/+1pswWso8W+s/lov0bzz1fWXJzP394EtZ9Q74c+ZBmNW+wwr3T23f9fbriK4zrB9Udj1WtM=

C.3. COSE signed parboiled requestvoucher

This voucher request has been signed by the JRC using the private key from Appendix C.1.4. Contained within this voucher request is the pledge voucher request above.
This is the raw binary, encoded in base64:

```
0oRBoKBZAq6hGgAPRsKlAWlwcm94aW1pdHkCwRpZ3Tv9C1EwMc1EMC1FNS1G
Mi0wMC0wMDw2X3FwEd3OFTZQmJaZTl1NWFvWjVMdwtZAmbSHEGoFKCHKEa
AA9GwqUBaXYb3hpbW1eqLGBl0eS2ckKUTAWLUwLU1LYyLaTLAzB3Zf
cV2Rz3c4VlNYmXlOU00VXAZ2Nux3DFkBDCCAdA_wgFXoAMCAQ1C8CB0Scw
CygYIoEzi0EAwIwbrjEMBACGcmSmJ0t8ixkARwAmNhMKwfwYWKCZImlzPy
LQBGKRYJcZfu2ZgSwbfWGmTowwYDvQD3Rmb3VudG9iZ01ZXN0LmY4Y1wL
buGvYu29tCl1BrvbNoUnvZyGBg3VudG9iZ01ZXN0LmY4Y1wL
```

Richardson, et al.  Expires January 5, 2020  [Page 41]
C.4. COSE signed voucher

The resulting voucher is created by the MASA and returned via the JRC to the Pledge. It is signed by the MASA’s private key Appendix C.1.6 and can be verified by the pledge using the MASA’s public key.

This is the CBOR diagnostic format, folded to 60 characters:

```
18((h’A0’, {}, h’A11A000F468CA505666C6F6767656406C11A5D1E499
A0E7130302D4302D45352D46322D30302D3030B765F715674477738565
342626C65394D34557036354C770C7902744D949423054434315661674
177494241674942416A414B42676771686B6A4F50515144417A42784D524
9774541594B435A496D695A50794C475142752594359325478475441584
2676F4A6B6914A6B2F49735A41455A46676C7A5957356B5A57787459573
4785144112B42674E5642414D4D4E794D38553336C7A64475674566246976
1574666924755364D4867774D4441774E4759354D546684D443
4675657357A6448A31626D6367526D393126E526861573467513054774
868634E4D5463784D5413346A4D304E5449345768634E4D546B784D544
1334D6A4D304E544934576A42444DS4977451594B435A496D695A50794
C7541247525943593245784754415842676F4A6B6914A6B2F49735A414
55A46676C7A5957356B5A57787459573478456A415142674E5642144D4
357786593246736147379A644425A4D424D47297147534D34914674
54743437147534D34914774548143044124A5A6C55489307502F6C3
365A6639764342622B6C496E654D4454D45676337526F2B858A4376A41493
0434431664A664A522F68497979446D48577959694E5662524348396796
172666B7A67583470307A54697A716A4454114CD416B47413155455775
1434D411743675948B6F5A4977A63A04541774D46615141775A6749784
14C514D4E752638747635306C524F443544515848454F4A44AE7353515
63267395145644436B324D592B416F537242536D475346A683466C454
F6845754C67497841A3A46E57664E772B426A625A6D4B94969554563547
7484D684568614D48592F46376E3339777774B634242534F6E64507144
3704F454C6C36627133435A71513D3D’, h’7468F16A4035FDF510DBF5
A88F67B6F8B49CFBA8B094B77AD524890E4BC6DE892FE74B39AB787637B
121944BED4D1CB4B8DC8F59212EAC2AD20469C71C1F6’)})
```

This is the raw binary, encoded in base64:

```
```
Authors’ Addresses

Michael Richardson
Sandelman Software Works

Email: mcr+ietf@sandelman.ca

Peter van der Stok
vanderstok consultancy

Email: consultancy@vanderstok.org

Panos Kampanakis
Cisco Systems

Email: pkampana@cisco.com