anima Working Group
Internet-Draft
Intended status: Standards Track
Expires: July 17, 2020

M. Richardson
Sandelman Software Works
P. van der Stok
vanderstok consultancy
P. Kampanakis
Cisco Systems

January 14, 2020

# Constrained Voucher Artifacts for Bootstrapping Protocols
# draft-ietf-anima-constrained-voucher-06

## 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



## Copyright Notice

Table of Contents

## 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],

2. Addition to [I-D.ietf-ace-coap-est] of voucher transport requests over CoAP.

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

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.

## 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 have accurate 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 a 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 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.

| BRSKI | EST-coaps |
|---|---|
| /requestvoucher | /rv |
| /voucher_status | /vs |
| /enrollstatus | /es |
| /requestauditlog | /ra |

Table 1: BRSKI path to EST-coaps path

/requestvoucher, /voucher_status 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.

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

```
      SID Assigned to
 --------- --------------------------------------------------
      2501 data /ietf-constrained-voucher-request:voucher
      2502 data .../assertion
      2503 data .../created-on
      2504 data .../domain-cert-revocation-checks
      2505 data .../expires-on
      2506 data .../idevid-issuer
      2507 data .../last-renewal-date
      2508 data /ietf-constrained-voucher-request:voucher/nonce
      2509 data .../pinned-domain-cert
      2510 data .../prior-signed-voucher-request
      2511 data .../proximity-registrar-cert
      2512 data mity-registrar-sha256-of-subject-public-key-info
      2513 data .../proximity-registrar-subject-public-key-info
      2514 data .../serial-number
```

### 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@2019-09-01.yang"
module ietf-constrained-voucher-request {
  yang-version 1.1;

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

      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 "2019-09-01" {
      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.";
     }
```

```
        }
      }
    }
  }
  <CODE ENDS>
```

### 6.1.4. Example voucher request artifact

Below is 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].

```
  {
    2501: {
      +2 : "2016-10-07T19:31:42Z", / SID= 2503, created-on /
      +4 : "2016-10-21T19:31:42Z", / SID= 2505, expires-on /
      +1 : 2,                      / SID= 2502, assertion /
                                   /                "proximity" /
      +13: "JADA123456789",        / SID= 2514, serial-number /
      +5 : h'01020D0F',            / SID= 2506, idevid-issuer /
      +10: h'01020D0F',            / SID=2511, proximity-registrar-cert/
      +3 : true,                   / SID= 2504, domain-cert
                                                -revocation-checks/
      +6 : "2017-10-07T19:31:42Z", / SID= 2507, last-renewal-date /
      +12: h'01020D0F'             / SID= 2513, proximity-registrar
                                           -subject-public-key-info /
    }
  }
```

### 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                                   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
       +-- pinned-domain-subject-public-key-info?      binary
       +-- pinned-sha256-of-subject-public-key-info?   binary
```

### 6.2.2. SID values

```
      SID Assigned to
--------- --------------------------------------------------
     2451 data /ietf-constrained-voucher:voucher
     2452 data /ietf-constrained-voucher:voucher/assertion
     2453 data /ietf-constrained-voucher:voucher/created-on
     2454 data .../domain-cert-revocation-checks
     2455 data /ietf-constrained-voucher:voucher/expires-on
     2456 data /ietf-constrained-voucher:voucher/idevid-issuer
     2457 data .../last-renewal-date
     2458 data /ietf-constrained-voucher:voucher/nonce
     2459 data .../pinned-domain-cert
     2460 data .../pinned-domain-subject-public-key-info
     2461 data .../pinned-sha256-of-subject-public-key-info
     2462 data /ietf-constrained-voucher:voucher/serial-number
```

### 6.2.3. YANG Module

In the constrained-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@2019-09-01.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
   establis 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 "2019-09-01" {
    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";
        }
      }
    }
  }
}
<CODE ENDS>
```

### 6.2.4. Example voucher artifacts

Below a the CBOR serialization of 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].

```
{
  2451: {
    +2 : "2016-10-07T19:31:42Z", / SID = 2453, created-on /
    +4 : "2016-10-21T19:31:42Z", / SID = 2455, expires-on /
    +1 : 0,                      / SID = 2452, assertion /
                                 /                "verified" /
    +11: "JADA123456789",        / SID = 2462, serial-number /
    +5 : h'01020D0F',            / SID = 2456, idevid-issuer /
    +8 : h'01020D0F',            / SID = 2459, pinned-domain-cert/
    +3 : true,                   / SID = 2454, domain-cert
                                               -revocation-checks /
    +6 : "2017-10-07T19:31:42Z", / SID = 2457, last-renewal-date /
    +9 : h'01020D0F'             / SID = 2460, pinned-domain
                                         -subject-public-key-info /
  }
}
```

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 id-ct-animaJSONVoucher allocated by [RFC8366] indicates a voucher and voucher-request encoded in JSON, and the new value TBD1 indicates that the voucher and voucher-request are encoded in CBOR.

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 with Ed25519 is compulsory.

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

```
COSE_Sign1(
  [
    h'a10127',        # { "alg": EDdsa }
    {
      "kid" : h'789'  # hash(pblic key)
    },
    h'123', #voucher-request binary content
    h'456', #voucher-request binary public signature
  ]
)
```

Figure 1: cose-sign1 example

The [COSE-registry] specifies the integers that replace the strings and the mnemonics in Figure 1. The value of the "kid" parameter is an example value. Usually a hash of the public key is used to idientify the public key. The distribution of the public key and its hash is done out of band.

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:

```
URI: urn:ietf:params:xml:ns:yang:ietf-constrained-voucher
Registrant Contact: The ANIMA WG of the IETF.
XML: N/A, the requested URI is an XML namespace.

URI: urn:ietf:params:xml:ns:yang:ietf-constrained-voucher-request
Registrant Contact: The ANIMA WG of the IETF.
XML: N/A, the requested URI is an XML namespace.
```

### 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 the following registration is requested:

```
name:         ietf-constrained-voucher
namespace:    urn:ietf:params:xml:ns:yang:ietf-constrained-voucher
prefix:       vch
reference:    RFC XXXX

name:         ietf-constrained-voucher-request
namespace:    urn:ietf:params:xml:ns:yang:ietf-constrained
                                         -voucher-request
prefix:       vch
reference:    RFC XXXX
```

### 9.4. The RFC SID range assignment sub-registry

| Entry-point | Size | Module name | RFC Number |
|---|---|---|---|
| 2450 | 50 | ietf-constrained-voucher | [ThisRFC] |
| 2500 | 50 | ietf-constrained-voucher -request | [ThisRFC} |

Warning: These SID values are defined in [I-D.ietf-core-sid].

### 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:

| Decimal | Description | References |
|---|---|---|
| TBD1 | id-ct-animaCBORVoucher | [ThisRFC] |

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).

| Media type | mime type | Encoding | ID | References |
|---|---|---|---|---|
| application/voucher-cms+cbor | - - | CBOR | TBD2 | [This RFC] |
| application/voucher-cose+cbor | "COSE-Sign1" | CBOR | TBD3 | [This RFC] |

## 10. Acknowledgements

We are very grateful to Jim Schaad for explaining COSE and CMS choices. Also thanks to Jim Schaad for correctinging earlier version of the COSE Sign1 objects.

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

-06 New SID values assigned; regenerated examples

-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 requestvoucher 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-coap-est]
Stok, P., Kampanakis, P., Richardson, M., and S. Raza, "EST over secure CoAP (EST-coaps)", draft-ietf-ace-coap-est-18 (work in progress), January 2020.

[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-34 (work in progress), January 2020.

[I-D.ietf-core-sid]
Veillette, M., Pelov, A., and I. Petrov, "YANG Schema Item iDentifier (SID)", draft-ietf-core-sid-08 (work in progress), January 2020.

[I-D.ietf-core-yang-cbor]
Veillette, M., Petrov, I., and A. Pelov, "CBOR Encoding of Data Modeled with YANG", draft-ietf-core-yang-cbor-11 (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>.

[RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, DOI 10.17487/RFC3688, January 2004, <https://www.rfc-editor.org/info/rfc3688>.

[RFC5652] Housley, R., "Cryptographic Message Syntax (CMS)", STD 70, RFC 5652, DOI 10.17487/RFC5652, September 2009, <https://www.rfc-editor.org/info/rfc5652>.

[RFC6020] Bjorklund, M., Ed., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, DOI 10.17487/RFC6020, October 2010, <https://www.rfc-editor.org/info/rfc6020>.

[RFC7049] Bormann, C. and P. Hoffman, "Concise Binary Object Representation (CBOR)", RFC 7049, DOI 10.17487/RFC7049, October 2013, <https://www.rfc-editor.org/info/rfc7049>.

[RFC7950] Bjorklund, M., Ed., "The YANG 1.1 Data Modeling Language", RFC 7950, DOI 10.17487/RFC7950, August 2016, <https://www.rfc-editor.org/info/rfc7950>.

[RFC8152] Schaad, J., "CBOR Object Signing and Encryption (COSE)", RFC 8152, DOI 10.17487/RFC8152, July 2017, <https://www.rfc-editor.org/info/rfc8152>.

[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>.

## 12.2. Informative References

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

[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.

[RFC6690] Shelby, Z., "Constrained RESTful Environments (CoRE) Link Format", RFC 6690, DOI 10.17487/RFC6690, August 2012, <https://www.rfc-editor.org/info/rfc6690>.

[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>.

## Appendix A. EST messages to EST-coaps

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"
  }
```

The binary payload is:


```
A46776657273696F6E6131665374617475730166526561736F6E7822
496E666F726D61746976652068756D616E207265616461626C65206D
6573736167656e726561736F6E2D636F6E74657874
764164646974696F6E616C20696E666F726D6174696F6E
```


The binary payload disassembles to the above CBOR diagnostic code.

## A.2. 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 status (ct=60) will then be:

```
2.05 Content (Content-Format: application/cbor)
Payload =
A46776657273696F6E6131666537461747573016652656173 6F6E7822
```

A46776657273696F6E613166653746174757301665265616736F6E7822
496E666F726D61746976652068756D616E207265616461626C65206D
6573736167656e726561736F6E2D636F6E74657874
764164646974696F6E616C20696E666F726D6174696F6E

## A.3. 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.3.1. 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.

MIIDugYJKoZIhvcNAQcCoIIDqzCCA6cCAQExDTALBglghkgBZQMEAgEwCwYJ
KoZIhvcNAQcBoIICQTCCAj0wggHioAMCAQICCH52Yde1TkYyMAoGCCqGSM49
BAMCMF0xCzAJBgNVBAYTAlVTMQswCQYDVQQIDAJDQTEUMBIGA1UECgwLRXhh
bXBsZSBJbmMxFjAUBgNVBAsMDWNlcnRpZmljYXRpb24xEzARBgNVBAMMCjgw
Mi4xQVIgQ0EwIBcNMTkwMTMxMTEyOTE2WhgPOTk5OTEyMzEyMzU5NTlaMFwx
CzAJBgNVBAYTAlVTMQswCQYDVQQIDAJDQTELMAkGA1UEBwwCTEExFDASBgNV
BAoMC2V4YW1wbGUgSW5jMQwwCgYDVQQLDANJb1QxDzANBgNVBAUTBld0MTIz
NDBZMBMGByqGSM49AgEGCCqGSM49AwEHA0IABMi0IfEcJeR+OsVxI78tn9xJ
TwKLw1HMgMA/FQv1DP+VjXVBnYGmokXf+ueQvpXPdfYC+RUmGPgWorI7Vjjl
n9mjgYowgYcwCQYDVR0TBAIwADAdBgNVHQ4EFgQUlmANhxa/f9DnUtCsdgd3
rWZdAqAwHwYDVR0jBBgwFoAUaNFlUflRv8gqQx0Nnwi8LSBbEWAwDgYDVR0P
AQH/BAQDAgWgMCoGA1UdEQQjMCGgHwYIKwYBBQUHCASgEzARBgkrBgEEAbQ7
CgEEBAECAwQwCgYIKoZIzj0EAwIDSQAwRgIhAMDYGZbSUH1pPzxI6qXulJG9
ptshQJnZgRfGOzYTdM2GAiEAp3SYn0wyGlzyXYMqTTNqCK1n3yDxUGQhGIoK
3m00kjYxggE/MIIBOwIBATBpMF0xCzAJBgNVBAYTAlVTMQswCQYDVQQIDAJD
QTEUMBIGA1UECgwLRXhhbXBsZSBJbmMxFjAUBgNVBAsMDWNlcnRpZmljYXRp
b24xEzARBgNVBAMMCjgwMi4xQVIgQ0ECCH52Yde1TkYyMAsGCWCGSAFlAwQC
AaBpMBgGCSqGSIb3DQEJAzELBgkqhkiG9w0BBwEwHAYJKoZIhvcNAQkFMQ8X
DTE5MDQwODEwNDgzNlowLwYJKoZIhvcNAQkEMSIEILEdCTOLs2Zy7w3LgvSZ
XZEadz3LbznoFBs6FMFN91RaMAoGCCqGSM49BAMCBEcwRQIgASjDsIpr0tW/
n6dRHqvvqsqgZlHbtFnErUbWfhS0KD4CIQDDUEqc5wTmRGf0adEQVQzqmIgh
MEgF10vqXv02gL1jLw==

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)
```


MIIDuwYJKoZIhvcNAQcCoIIDrDCCA6gCAQExDTALBglghkgBZQMEAgEwCwYJ
KoZIhvcNAQcBoIICQTCCAj0wggHioAMCAQICCH52Yde1TkYyMAoGCCqGSM49
BAMCMF0xCzAJBgNVBAYTAlVTMQswCQYDVQQIDAJDQTEUMBIGA1UECgwLRXhh
bXBsZSBJbmMxFjAUBgNVBAsMDWNlcnRpZmljYXRpb24xEzARBgNVBAMMCjgw
Mi4xQVIgQ0EwIBcNMTkwMTMxMTEyOTE2WhgPOTk5OTEyMzEyMzU5NTlaMFwx
CzAJBgNVBAYTAlVTMQswCQYDVQQIDAJDQTELMAkGA1UEBwwCTEExFDASBgNV
BAoMC2V4YW1wbGUgSW5jMQwwCgYDVQQLDANJb1QxDzANBgNVBAUTBld0MTIz
NDBZMBMGByqGSM49AgEGCCqGSM49AwEHA0IABMi0IfEcJeR+OsVxI78tn9xJ
TwKLw1HMgMA/FQv1DP+VjXVBnYGmokXf+ueQvpXPdfYC+RUmGPgWorI7Vjjl
n9mjgYowgYcwCQYDVR0TBAIwADAdBgNVHQ4EFgQUlmANhxa/f9DnUtCsdgd3
rWZdAqAwHwYDVR0jBBgwFoAUaNFlUflRv8gqQx0Nnwi8LSBbEWAwDgYDVR0P
AQH/BAQDAgWgMCoGA1UdEQQjMCGgHwYIKwYBBQUHCASgEzARBgkrBgEEAbQ7
CgEEBAECAwQwCgYIKoZIzj0EAwIDSQAwRgIhAMDYGZbSUH1pPzxI6qXulJG9
ptshQJnZgRfGOzYTdM2GAiEAp3SYn0wyGlzyXYMqTTNqCK1n3yDxUGQhGIoK
3m00kjYxggFAMIIBPAIBATBpMF0xCzAJBgNVBAYTAlVTMQswCQYDVQQIDAJD
QTEUMBIGA1UECgwLRXhhbXBsZSBJbmMxFjAUBgNVBAsMDWNlcnRpZmljYXRp
b24xEzARBgNVBAMMCjgwMi4xQVIgQ0ECCH52Yde1TkYyMAsGCWCGSAFlAwQC
AaBpMBgGCSqGSIb3DQEJAzELBgkqhkiG9w0BBwEwHAYJKoZIhvcNAQkFMQ8X
DTE5MDQwODA3MzQxMFowLwYJKoZIhvcNAQkEMSIEIP2rKa+J8LVdwYEmB2he
uxsz05As0zoAAYkeyNqsh4fiMAoGCCqGSM49BAMCBEgwRgIhALOd2FKbe9FG
kN4Pg7FIgF+//cQv/N+v7tDZMzGBAFN0AiEAu5BI0oQ4o0wZcrDyKoU2GbeX
hlG/g+OgTUftYMJ32so=


## A.4. requestauditing

A coaps requestauditing message contains the signed CBOR voucher :

```
    POST coaps://[2001:db8::2:1]:61616/est/ra
    (Content-Format: application/voucher-cms+cbor)
    Payload =
TO BE FILLED
```

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.


```
a11909c5a90274323031362d31302d30375431393a33313a34325a0474323031
362d31302d32315431393a33313a34325a01020d6d4a414441313233343536
373839054401020d0f0a4401020d0f03f50674323031372d31302d30375431
393a33313a34325a0c4401020d0f
```


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.3.1 and Appendix A.4.

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)
  d.signedData:
    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:
            algor:
              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

These examples are from the https://minerva.sandelman.ca/ reference code, using the unit test case key pairs, with a flow between pledge ("reach" code), JRC ("fountain") code, and MASA ("highway") code. This example comes from the spec/files/product/00-D0-E5-F2-00-03 directory.

### 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 = h
ighway-test.example.com CA
    Validity
      Not Before: Feb 14 17:05:09 2019 GMT
      Not After : Dec 31 00:00:00 2999 GMT
    Subject: serialNumber = 00-D0-E5-F2-00-03
    Subject Public Key Info:
      Public Key Algorithm: id-ecPublicKey
        Public-Key: (256 bit)
        pub:
          04:82:c4:28:5b:7c:f0:37:18:c7:90:14:dc:cb:f4:
          4d:7e:b0:00:ed:c0:de:bd:4d:25:55:4e:35:f9:d5:
          6a:57:14:b4:94:af:ce:6d:53:c8:60:c2:ce:53:3f:
          2c:1b:42:f1:c0:8b:5f:c1:7b:3d:f3:29:54:87:46:
          86:a4:0c:8b:b7
        ASN1 OID: prime256v1
        NIST CURVE: P-256
    X509v3 extensions:
      X509v3 Subject Key Identifier:
        C8:A3:87:72:82:82:E6:EA:90:D0:E1:81:BC:C7:51:08:78:0
F:D7:52
      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
     30:65:02:31:00:b2:9a:7a:1a:74:20:8f:e9:e0:5d:fc:af:d6:
     4a:80:1f:66:e3:bf:17:2e:3e:07:87:39:be:65:bd:94:57:71:
     1f:df:e5:fc:4d:ef:96:8a:3a:03:5b:d1:ca:a1:72:55:a3:02:
     30:13:43:08:a4:af:c8:28:19:d2:a0:93:3d:ed:53:fa:09:7d:
     76:9c:b7:0b:95:2b:8f:2f:b4:fa:87:02:ec:b4:2d:19:92:5b:
     b2:bb:79:04:63:6e:17:0e:79:8a:65:f5:a3
```

### C.1.2. Device private key


```
-----BEGIN EC PRIVATE KEY-----
MHcCAQEEIA1sa0l4bkj/rJxPUN1bKSBNYolVVzx+T28wo60cYpuaoAoGCCqGSM49
AwEHoUQDQgAEgsQoW3zwNxjHkBTcy/RNfrAA7cDevU0lVU41+dVqVxS0lK/ObVPI
YMLOUz8sG0LxwItfwXs98ylUh0aGpAyLtw==
-----END EC PRIVATE KEY-----
```

### C.1.3. Registrar Certificate


-----BEGIN CERTIFICATE-----
MIIB0TCCAVagAwIBAgIBAjAKBggqhkjOPQQDAzBxMRIwEAYKCZImiZPyLGQBGRYC
Y2ExGTAXBgoJkiaJk/IsZAEZFglzYW5kZWxtYW4xQDA+BgNVBAMMNyM8U3lzdGVt
VmFyaWFibGU6MHgwMDAwMDAwNGY5MTFhMD4gVW5zdHJ1bmcgRm91bnRhaW4gQ0Ew
HhcNMTcxMTA3MjM0NTI4WhcNMTkxMTA3MjM0NTI4WjBDMRIwEAYKCZImiZPyLGQB
GRYCY2ExGTAXBgoJkiaJk/IsZAEZFglzYW5kZWxtYW4xEjAQBgNVBAMMCWxvY2Fs
aG9zdDBZMBMGByqGSM49AgEGCCqGSM49AwEHA0IABJZlUHI0up/l3eZf9vCBb+lI
noEMEgc7Ro+XZCtjAI0CD1fJfJR/hIyyDmHWyYiNFbRCH9fyarfkzgX4p0zTizqj
DTALMAkGA1UdEwQCMAAwCgYIKoZIzj0EAwMDaQAwZgIxALQMNurf8tv50lROD5DQ
XHEOJJNW3QV2g9QEdDSk2MY+AoSrBSmGSNjh4olEOhEuLgIxAJ4nWfNw+BjbZmKi
IiUEcTwHMhGVXaMHY/F7n39wwKcBBSOndNPqCpOELl6bq3CZqQ==
-----END CERTIFICATE-----


### C.1.4. Registrar private key


-----BEGIN EC PRIVATE KEY-----
MHcCAQEEIFZodk+PC5Mu24+ra0sbOjKzan+dW5rvDAR7YuJUOC1YoAoGCCqGSM49
AwEHoUQDQgAElmVQcjS6n+Xd5l/28IFv6UiegQwSBztGj5dkK2MAjQIPV8l8lH+E
jLIOYdbJiI0VtEIf1/Jqt+TOBfinTNOLOg==
-----END EC PRIVATE KEY-----


### C.1.5. MASA Certificate


-----BEGIN CERTIFICATE-----
MIIB3zCCAWSgAwIBAgIEG5lfVDAKBggqhkjOPQQDAjBdMQ8wDQYDVQQGEwZDYW5h
ZGExEDAOBgNVBAgMB09udGFyaW8xEjAQBgNVBAsMCVNhbmRlbG1hbjEkMCIGA1UE
AwwbaGlnaHdheS10ZXN0LmV4YW1wbGUuY29tIENBMB4XDTE5MDIxMjIyMjI0MVoX
DTIxMDIxMTIyMjI0MVowXzEPMA0GA1UEBhMGQ2FuYWRhMRAwDgYDVQQIDAdPbnRh
cmlvMRIwEAYDVQQLDAlTYW5kZWxtYW4xJjAkBgNVBAMMHWhpZ2h3YXktdGVzdC5l
eGFtcGxlLmNvbSBNQVNBMFkwEwYHKoZIzj0CAQYIKoZIzj0DAQcDQgAEqgQVo0S5
4kT4yfkbBxumdHOcHrpsqbOpMKmiMln3oB1HAW25MJV+gqi4tMFfSJ0iEwt8kszf
WXK4rLgJS2mnpaMQMA4wDAYDVR0TAQH/BAIwADAKBggqhkjOPQQDAgNpADBmAjEA
vVXlmw77/F6VKeOBsxU1qpMYogS+RHKyUX1NbevR1cEQOrI5e1c/xcywow7nmUa6
AjEA9n9EfbcU+tFnatQRw0uu5vuamFb6hSEuXEhM8D/ymz+uiCCnrvly/1v5eGjP
D0jJ
-----END CERTIFICATE-----


### C.1.6. MASA private key


-----BEGIN EC PRIVATE KEY-----
MHcCAQEEIFhdd0eDdzip67kXx72K+KHGJQYJHNy8pkiLJ6CcvxMGoAoGCCqGSM49
AwEHoUQDQgAEqgQVo0S54kT4yfkbBxumdHOcHrpsqbOpMKmiMln3oB1HAW25MJV+
gqi4tMFfSJ0iEwt8kszfWXK4rLgJS2mnpQ==
-----END EC PRIVATE KEY-----

### C.2. COSE signed requestvoucher with registrar certificate pinned

EDNote: These examples do not parse

This voucher request has been signed by the pledge, 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
D1E49970A5130302D44302D45352D46322D30302D303307765F715674477
738565342626C65394D34557036354C770C5901D4308201D030820157A00
30201020204228ECD27300A06082A8648CE3D040302306E31123010060A0
 ** KNOWN TO BE BAD, NOT YET VALIDATED **
0340F4E6D0F9F702553FA53BE572ACF0EED858275B6AC75994332FB25FB3
A54411E9FA02E6F75FD1AADB7EA9A61F5409E02303E615E75C8F07432A59
0C8D48798BEDA1EB49E5E7D8E0EA118BD17A02D02F0313D144816002F756
B528ABD1B0ADB749D', h'96B82530AC57650346C2BFFB5A6CC16B28F16F
ACFE5A2FD1BCF3D5F5D62733F7F7812D67D43BE1CF9906E356FB0C2BDD36
777FD7DBAE22B8CEB07D51D8F55AD3'])
```


This is the raw binary, shown as hex dump:


```
0oRBoKBZAhyhGgAPRsKlAWlwcm94aW1pdHkCwRpdHkmXClEwMC1EMC1FNS1G
Mi0wMC0wMwd2X3FWdEd3OFZTQmJsZTlNNFVwNjVMdwxZAdQwggHQMIIBV6AD
 ** KNOWN TO BE BAD, NOT YET VALIDATED **
NA9ObQ+fcCVT+lO+VyrPDu2FgnW2rHWZQzL7Jfs6VEEen6Aub3X9Gq236pph
9UCeAjA+YV51yPB0MqWQyNSHmL7aHrSeXn2ODqEYvRegLQLwMT0USBYAL3Vr
Uoq9GwrbdJ1YQJa4JTCsV2UDRsK/+1pswWso8W+s/lov0bzz1fXWJzP394Et
Z9Q74c+ZBuNW+wwr3TZ3f9fbriK4zrB9Udj1WtM=
```


### C.3. COSE signed parboiled requestvoucher

These example do not parse

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:

18([h'A0', {}, h'A11A000F46C2A5016970726F78696D69747902C11A5
9DD3BFD0A5130302D44302D45352D46322D30302D303307765F715674477
738565342626C65394D34557036354C770B590266D28441A0A059021CA11
A000F46C2A5016970726F78696D69747902C11A5D1E49970A5130302D443
 ** KNOWN TO BE BAD, NOT YET VALIDATED **
AADB7EA9A61F5409E02303E615E75C8F07432A590C8D48798BEDA1EB49E5
E7D8E0EA118BD17A02D02F0313D144816002F756B528ABD1B0ADB749D584
096B82530AC57650346C2BFFB5A6CC16B28F16FACFE5A2FD1BCF3D5F5D62
733F7F7812D67D43BE1CF9906E356FB0C2BDD36777FD7DBAE22B8CEB07D5
1D8F55AD3', h'EAE868ECC176883766C5DC5BA5B8DCA25DAB3C2E56A551
CE5705B793914348E1F93C2B81E88CCBE28E90800F66945EFBBECE4F741D
0EDE18EB1008EF7E9A279C'])

This is the raw binary, encoded in base64:

0oRBoKBZAq6hGgAPRsKlAWlwcm94aW1pdHkCwRpZ3Tv9ClEwMC1EMC1FNS1G
Mi0wMC0wMwd2X3FWdEd3OFZTQmJsZTlNNFVwNjVMdwtZAmbShEGgoFkCHKEa
AA9GwqUBaXByb3hpbWl0eQLBGl0eSZcKUTAwLUQwLUU1LUYyLTAwLTAzB3Zf
 ** KNOWN TO BE BAD, NOT YET VALIDATED **
U75XKs8O7YWCdbasdZlDMvsl+zpUQR6foC5vdf0arbfqmmH1QJ4CMD5hXnXI
8HQypZDI1IeYvtoetJ5efY4OoRi9F6AtAvAxPRRIFgAvdWtSir0bCtt0nVhA
lrglMKxXZQNGwr/7WmzBayjxb6z+Wi/RvPPV9dYnM/f3gS1n1Dvhz5kG41b7
DCvdNnd/19uuIrjOsH1R2PVa01hA6uho7MF2iDdmxdxbpbjcol2rPC5WpVHO
VwW3k5FDSOH5PCuB6IzL4o6QgA9mlF77vs5PdB0O3hjrEAjvfponnA==

## 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
A0E7130302D44302D45352D46322D30302D30330B765F715674477738565
342626C65394D34557036354C770C7902744D49494230544343415661674
177494241674942416A414B42676771686B6A4F50515144417A42784D524
9774541594B435A496D695A50794C47514247525943593245784754441584
2676F4A6B69614A6B2F49735A41455A46676C7A5957356B5A57787459573
4785144412B42674E5642414D4D4E794D3855336C7A64475674566D46796
1574669624755364D4867774D4441774D4441774E4759354D5446684D443
4675657357A64484A31626D6367526D3931626E5268615734675130457774
868634E4D5463784D5441334D6A4D304E5449345768634E4D546B784D544
1334D6A4D304E544934576A42444D5249774541594B435A496D695A50794
C475142475259435932457847544415842676F4A6B69614A6B2F49735A414
55A46676C7A5957356B5A57787459573478456A415142674E5642414D4D4
357787659324673614739734A6444425A4D424D4742797147534D343941674
54743437147534D34394177454841304941424A5A6C5548493075702F6C3
3655A6639764342622B6C496E6F454D45676337526F2B585A43746A41493
0434431664A664A522F68497979446D48577959694E46625243483966796
172666B7A67583470307A54697A716A4454414C4D416B474131556445775
1434D41417743675949B6F5A497A6A304541774D44615141775A6749784
14C514D4E75726638747635306C524F443544515848454F4A4A4E5733515
632673951456444536B324D592B416F537242536D47534E6A68346F6C454
F6845754C674978414A346E57664E772B426A625A6D4B694969554563547
7484D68475658614D48592F46376E333977774B634242534F6E644E50714
3704F454C6C36627133435A71513D3D', h'7468FB16A4035FDAF510DBF5
A88F67B6FB849CFBA8B094B77AD5248900E4BCD6E892FE74B39AB787637B
121944BED4D1CB4B8DC8F59212EAC2AD20469C71C1F6'])
```


This is the raw binary, encoded in base64:


```
0oRBoKBZArmhGgAPRoylBWZsb2dnZWQGwRpdHkmaDnEwMC1EMC1FNS1GMi0w
MC0wMwt2X3FWdEd3OFZTQmJsZTlNNFVwNjVMdwx5AnRNSUlCMFRDQ0FWYWdB
d0lCQWdJQkFqQUtCZ2dxaGtqT1BRUURBekJ4TVJJd0VBWUtDWkltaVpQeUxH
UUJHUllDWTJFeEdUQVhCZ29Ka2lhSmsvSXNaQUVaRmdsellXNWtaV3h0WVc0
eFFEQStCZ05WQkFNTU55TThVM2x6ZEdWdFZtRnlhV0ZpYkdVNk1IZ3dNREF3
TURBd05HWTVNVEZoTUQ0Z1ZXNXpkSEoxYm1jZ1JtOTFibTJoYVc0Z1EwRXdI
aGNOTVRjeE1UQTNNak0wTlRJNFdoY05NVGt4TVRBM01qTTBOVEk0V2pCRE1S
SXdFQVlLQ1pJbWlaUHlMR1FCR1JZQ1kyRXhHVEFYQmdvSmtpYUprL0lzWkFF
WkZnbHpZVzVrWld4dFlXNHhFakFRQmdOVkJBTU1DV3h2WTJGc2FHOXpkREJa
TUJNR0J5cUdTTTQ5QWdFR0NDcUdTTTQ5QXdFSEEwSUFCSlpsVUhJMHVwL2wz
ZVpmOXZDQmIrbElub0VNRWdjN1JvK1haQ3RqQUkwQ0QxZkpmSlIvaEl5eURt
SFd5WWlORmJSQ0g5ZnlhcmZremdYNHAwelRpenFqRFRBTE1Ba0dBMVVkRXdR
Q01BQXdDZ1lJS29aSXpqMEVBd01EYVFBd1pnSXhBTFFFNTnVyZjh0djUwbFJP
RDVEUVhIRU9KSk5XM1FWMmc5UUVkRFNrMk1ZK0FvU3JCU21HU05qaDRvbEVP
aEV1TGdJeEFKNG5XZk53K0JqYlptS2lJaVVFY1R3SE1oR1ZYYU1IWS9GN24z
OXd3S2NCQlNPbmROUHFDcE9FTGw2YnEzQ1pxUT09WEB0aPsWpANf2vUQ2/Wo
j2e2+4Sc+6iwlLd61SSJAOS81uiS/nSzmreHY3sSGUS+1NHLS43I9ZIS6sKt
IEacccH2
```

### C.5. COSE signed request voucher

The headecimal dump of request-voucher shown in Appendix B.1 is used for this example. The cose sign1 object of Figure 1 provides the structure of the cose sign1 example. To identify the public key a hash of the public key is made using murmur3 with seed 42.

```
public key:
84 44 de 3a b7 b5 a0 2f 20 ed e7 80 1a f0 76 d6 52 0a e5 c8 a1
04 41 61 b2 64 57 fe 0e ae 08 4d
```

murmur3 hash of public key: 0x4727e669  or  1193797225

Using the values "kid" = 4, EDdsa = -8, "alg" = 2, COSE_Sign1 = 18, and using the murmur3 hash value for the "kid" parameter, the CBOR object is:


```
18([h'A10127', {4: 1193797225},
  h'A11909C5A90274323031362D31302D30375431393A33313A34325A04743
  23031362D31302D32315431393A33313A34325A01020D6D4A414441313233
  343536373839054401020D0F0A4401020D0F03F50674323031372D31302D3
  0375431393A33313A34325A0C4401020D0F',
h'955D82A26B7C0869EE8FE5A09EE3D68DDFFE8FE39E3BCADFA80F2F9A6E13F
  F0349A2CA131C8F6A9AAF7780BAB671F63CBB158EC17322323C1AB82B1CDC
  B62A06'])
```


The corresponding hexadecimal dump is:


```
d28443a10127a1041a4727e669586ca11909c5a90274323031362d31302d
30375431393a33313a34325a0474323031362d31302d32315431393a3331
3a34325a01020d6d4a414441313233343536373839054401020d0f0a4401
020d0f03f50674323031372d31302d30375431393a33313a34325a0c4401
020d0f5840955d82a26b7c0869ee8fe5a09ee3d68ddffe8fe39e3bcadfa8
0f2f9a6e13ff0349a2ca131c8f6a9aaf7780bab671f63cbb158ec1732232
3c1ab82b1cdcb62a06
```


## 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