ANIMA Working Group
Internet-Draft
Intended status: Standards Track
Expires: 3 October 2025

K. Watsen
Watsen Networks
M. Richardson
Sandelman Software
M. Pritikin
Cisco Systems
T. Eckert
Futurewei Technologies Inc.
Q. Ma
Huawei

1 April 2025

A Voucher Artifact for Bootstrapping Protocols
draft-ietf-anima-rfc8366bis-14

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 defines an artifact format as a YANG-defined JSON or CBOR document that has been signed using a variety of cryptographic systems.

The voucher artifact is normally generated by the Pledge's manufacturer (i.e., the Manufacturer Authorized Signing Authority (MASA)).

This document updates RFC8366, includes a number of desired extensions into the YANG. The voucher request defined in RFC8995 is also now included in this document, as well as other YANG extensions needed for variants of BRSKI/RFC8995.

About This Document

This note is to be removed before publishing as an RFC.

Status information for this document may be found at https://datatracker.ietf.org/doc/draft-ietf-anima-rfc8366bis/.

Discussion of this document takes place on the anima Working Group mailing list (mailto:anima@ietf.org), which is archived at https://mailarchive.ietf.org/arch/browse/anima/. Subscribe at https://www.ietf.org/mailman/listinfo/anima/.

Source for this draft and an issue tracker can be found at https://github.com/anima-wg/voucher.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

This document defines a strategy to securely assign a candidate device (Pledge) to an owner using an artifact signed, directly or indirectly, by the Pledge's manufacturer, i.e., the Manufacturer Authorized Signing Authority (MASA). This artifact is known as the "voucher".

The voucher artifact is a JSON [RFC8259] document that conforms with a data model described by YANG [RFC7950]. It may also be serialized to CBOR [CBOR]. It is encoded using the rules defined in [RFC7951] or [RFC9254], and is signed using (by default) a CMS structure [RFC5652].

The primary purpose of a voucher is to securely convey a trust anchor that a Pledge can use to authenticate subsequent interactions. The trust anchor may be in the form of a certificate (the "pinned-domain-cert" attribute), a hash of a certificate, or it can be a raw public key (in constrained variations).

This trust anchor represents the authority of the owner of a network. Communicating this trust anchor securely to the Pledge is the job of the voucher artifact. The act of communicating this trust anchor is known as pinning the trust anchor, as the Pledge can then use the resulting anchor to authenticate other actors who are part of the network. The collection of all these actors is collectively known as the network domain. (This is not related to the domain name system, but rather the term is of mathematical origin)

A voucher may be useful in several contexts, but the driving motivation herein is to support secure onboarding mechanisms. This is accomplished by assigning an owner to the Pledge, enabling it to authenticate the network that it is connected to.

[RFC8366] originally defined just the voucher artifact, leaving the Voucher Request artifiact that is important to BRSKI to be defined in [BRSKI]. This document includes both Voucher and Voucher-Request, and therefore updates [BRSKI].

YANG is not easily extended except by updating the YANG definition. Since [RFC8366] was written, the pattern is to publish YANG modules as two documents: one with only the YANG module, and the other one with usage, motivation and further explanation. This allows the YANG module to be updated without replacing all of the context. This document does not follow that pattern, but future updates are may update only the YANG.

This document also introduces an experimental mechanism to support future extensions without requiring the YANG to be replaced. This includes both new IETF Standard mechanisms, as well as a facility for manufacturer private extensions.

The lifetimes of vouchers may vary. In some onboarding protocols, the vouchers may include a nonce restricting them to a single use, whereas the vouchers in other onboarding protocols may have an indicated lifetime. In order to support long lifetimes, this document recommends using short lifetimes with programmatic renewal, see Section 9.1.

Some onboarding protocols using the voucher artifact defined in this document include: [ZERO-TOUCH], [SECUREJOIN], and [BRSKI].

2. Terminology

This document uses the following terms:

(Voucher) Artifact: Used throughout to represent the voucher as instantiated in the form of a signed structure.

Bootstrapping: The process where a Pledge component obtains cryptographic key material to identify and trust future interactions within a specific domain network. Based on imprinted key material provided during manufacturing process (see imprinting).

Domain: The set of entities or infrastructure under common administrative control. The goal of the onboarding protocol is to enable a Pledge component to join a domain and obtain domain specific security credentials.

Imprint: The process where a device obtains the cryptographic key material to identify and trust future interactions generally as part of the manufacturing. This term is taken from Konrad Lorenz’s work in biology with new ducklings: "during a critical period, the duckling would assume that anything that looks like a mother duck is in fact their mother" [Stajano99theresurrecting]. An equivalent for a device is to obtain the fingerprint of the manufacturer’s root certification authority (root ca) certificate. A device that imprints on an attacker suffers a similar fate to a duckling that imprints on a hungry wolf. Imprinting is a term from psychology and ethology, as described in [imprinting].

Join Registrar (and Coordinator): A representative of the domain that is configured, perhaps autonomically, to decide whether a new device is allowed to join the domain. The administrator of the domain interfaces with a join registrar (and Coordinator) to control this process. Typically, a join registrar is "inside" its domain. For simplicity, this document often refers to this as just "registrar".

MASA (Manufacturer Authorized Signing Authority): The entity that, for the purpose of this document, issues and signs the vouchers for a manufacturer’s Pledges. In some onboarding protocols, the MASA may have an Internet presence and be integral to the onboarding process, whereas in other protocols the MASA may be an offline service that has no active role in the onboarding process.

malicious registrar: An on-path active attacker that presents itself as a legitimate registrar, but which is in fact under the control of an attacker.

Onboarding: Onboarding describes the process to provide necessary operational data to Pledge components and completes the process to bring a device into an operational state. This data may be configuration data, or also application specific cryptographic key material (application speciifc security credentials).

Owner: The entity that controls the private key of the "pinned-domain-cert" certificate conveyed by the voucher.

Pledge: The prospective component attempting to find and securely join a domain. When shipped or in factory reset mode, it only trusts authorized representatives of the manufacturer.

Registrar: See join registrar.

TOFU (Trust on First Use): Where a Pledge component makes no security decisions but rather simply trusts the first domain entity it is contacted by. Used similarly to [RFC7435]. This is also known as the "resurrecting duckling" model.

Voucher: A short form for Voucher Artifact. It refers to the signed statement from the MASA service that indicates to a Pledge the cryptographic identity of the domain it should trust. When clarity is needed, it may be preceeded by the type of the signature, such as CMS, JWS or COSE.

Voucher Data: The raw (serialized) representation of the YANG without any enclosing signature. Current formats include JSON and CBOR.

Voucher Request: A signed artifact sent from the Pledge to the Registrar, or from the Registrar to the MASA for Voucher acquisition.

Pledge Voucher Request (PVR): A signed artifact sent from the Pledge to the Registrar. It is a special form of Voucher Request.

Registrar Voucher Request (RVR): A signed artifact sent from the Registrar to the MASA. It is a special form of Voucher Request.

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

A voucher is a cryptographically protected statement to the Pledge device authorizing a zero-touch "imprint" on the join registrar of the domain. The specific information a voucher provides is influenced by the onboarding use case.

The voucher can impart the following information to the join registrar and Pledge:

Assertion Basis: Indicates the method that protects the imprint (this is distinct from the voucher signature that protects the voucher itself). This includes manufacturer-asserted ownership verification, assured logging operations, or reliance on Pledge behavior such as secure root of trust of measurement. The join registrar uses this information to make a determination as to whether to accept the Pledge into the network. Only some methods are normatively defined in this document. Other methods are left for future work.

Authentication of Join Registrar: Indicates how the Pledge can authenticate the join registrar. This document defines a mechanism to pin the domain certificate, or a raw public key. Pinning a symmetric key, or "CN-ID" or "DNS-ID" information (as defined in [RFC6125]) is left for future work.

Anti-Replay Protections: Time- or nonce-based information to constrain the voucher to time periods or bootstrap attempts.

A number of onboarding scenarios can be met using differing combinations of this information. All scenarios address the primary threat of an on-path active attacker (or MiTM) impersonating the registrar. This would gain control over the Pledge. The following combinations are "types" of vouchers:

	Assertion		Registrar ID		Validity	
Voucher Type	Logged	Verified	Trust Anchor	CN-ID or DNS-ID	RTC	Nonce
Audit	X		X			X
Nonceless Audit	X		X		X	
Owner Audit	X	X	X		X	X
Owner ID		X	X	X	X	
Bearer voucher	X		wildcard	wildcard	optional	opt

Table 1

NOTE: All voucher types include a ’Pledge ID serial-number’ (not shown here for space reasons).

Audit Voucher: An Audit Voucher is named after the logging assertion mechanisms that the registrar then "audits" to enforce local policy. The registrar mitigates a malicious registrar by auditing that an unknown malicious registrar does not appear in the log entries. This does not directly prevent a malicious registrar but provides a response mechanism that ensures the on-path-attack is unsuccessful. An advantage is that actual ownership knowledge (i.e., sales integration providing an indication of who purchased the device) is not required on the MASA service.

Nonceless Audit Voucher: An Audit Voucher without a validity period statement. Fundamentally, it is the same as an Audit Voucher except that it can be issued in advance to support network partitions or to provide a permanent voucher for remote deployments. Being issued in advance of the Pledge being online, the Pledge can not provide a nonce to be included for freshness. This compromise in reducing the freshness allows for the resulting voucher can be carried across air-gapped infrastructure. In addition, as there is no end to the validity period, the voucher can be used after the manufacturer (and it’s delegates) has gone out of business.

Ownership Audit Voucher: An Audit Voucher where the MASA service has verified the registrar as the authorized owner. The MASA service mitigates a MiTM registrar by refusing to generate Audit Vouchers for unauthorized registrars. The registrar uses audit techniques to supplement the MASA. This provides an ideal sharing of policy decisions and enforcement between the vendor and the owner.

Ownership ID Voucher: Named after inclusion of the Pledge’s CN-ID or DNS-ID within the voucher. The MASA service mitigates a MiTM registrar by identifying the specific registrar (via WebPKI) authorized to own the Pledge.

Bearer Voucher: A Bearer Voucher is named after the inclusion of a registrar ID wildcard. Because the registrar identity is not indicated, this voucher type must be treated as a secret and protected from exposure as any ’bearer’ of the voucher can claim the pledge device. This variation is included in the above description in order to clearly how other voucher types differ. This specification does not support bearer vouchers at this time. There are other specifications in the industry which are equivalent though. Publishing a nonceless bearer voucher effectively turns the specified pledge into a "TOFU" device with minimal mitigation against MiTM registrars. Bearer vouchers are therefore out of scope.

5. Changes since RFC8366

5.1. Attempts and motivation to extend RFC8366

[RFC8366] was published in 2018 during the development of [BRSKI], [ZERO-TOUCH] and other work-in-progress efforts. Since then the industry has matured significantly, and the in-the-field activity which this document supports has become known as _onboarding_ rather than _bootstrapping_.

The focus of [BRSKI] was onboarding of ISP and Enterprise owned wired routing and switching equipment, with IoT devices being a less important aspect. [ZERO-TOUCH] has focused upon onboarding of CPE equipment like cable modems and other larger IoT devices, again with smaller IoT devices being of less import.

Since [BRSKI] was published there is now a mature effort to do application-level onboarding of constrained IoT devices defined by The Thread and Fairhair (now OCF) consortia [fairhair]. The [cBRSKI] document has defined a version of [BRSKI] that is useable over constrained 802.15.4 networks using CoAP and DTLS, while [I-D.selander-ace-ake-authz] provides for using CoAP and EDHOC on even more constrained devices with very constrained networks.

[PRM] has created a new methodology for onboarding that does not depend upon a synchronous connection between the Pledge and the Registrar. This mechanism uses a mobile Registrar Agent that works to collect and transfer signed artifacts via physical travel from one network to another.

Both [cBRSKI] and [PRM] require extensions to the Voucher Request and the resulting Voucher. The new attribtes are required to carry the additional attributes and describe the extended semantics. In addition [cBRSKI] uses the serialization mechanism described in [YANGCBOR] to produce significantly more compact artifacts.

When the process to define [cBRSKI] and [PRM] was started, there was a belief that the appropriate process was to use the [RFC8040] _augment_ mechanism to further extend both the voucher request [BRSKI] and voucher [RFC8366] artifacts. However, [PRM] needs to extend an enumerated type with additional values and _augment_ can not do this, so that was initially the impetus for this document.

An attempt was then made to determine what would happen if one wanted to have a constrained version of the [PRM] voucher artifact. The result was invalid YANG, with multiple definitions of the core attributes from the [RFC8366] voucher artifact. After some discussion, it was determined that the _augment_ mechanism did not work, nor did it work better when [RFC8040] yang-data was replaced with the [RFC8791] structure mechanisms.

After significant discussion the decision was made to simply roll all of the needed extensions into this document.

5.2. Informational Model changes since RFC8366

This document therefore represents a merge of YANG definitions from [RFC8366], the voucher-request from [BRSKI], and then extensions to each of these from [cBRSKI], [CLOUD] and [PRM]. The difficulty with this approach is that the semantics of the definitions needed for the other documents is not included in this document, but rather in the respective other documents.

6. Signature mechanisms

Three signature systems have been defined for vouchers and voucher-requests.

[cBRSKI] defines a mechanism that uses COSE RFC9052, with the voucher data encoded using [RFC9254]. However, as the SID process requires up-to-date YANG, the SID values for this mechanism are presented in this document.

[jBRSKI] defines a mechanism that uses JSON [RFC8259] and [JWS].

The CMS mechanism first defined in [RFC8366] continues to be defined here.

6.1. CMS Format Voucher Artifact

The IETF evolution of PKCS#7 is CMS [RFC5652]. A CMS-signed voucher, the default type, contains a ContentInfo structure with the voucher content. An eContentType of 40 indicates that the content is a JSON-encoded voucher.

The signing structure is a CMS SignedData structure, as specified by Section 5.1 of [RFC5652], encoded using ASN.1 Distinguished Encoding Rules (DER), as specified in ITU-T X.690 [ITU-T.X690.2015].

To facilitate interoperability, Section 11.3 the media type "application/voucher-cms+json" and the filename extension ".vcj" were registered by [RFC8366].

The CMS structure MUST contain a ’signerInfo’ structure, as described in Section 5.1 of [RFC5652], containing the signature generated over the content using a private key trusted by the recipient. Normally, the recipient is the Pledge and the signer is the MASA. In the Voucher Request, the signer is the Pledge, or the Registrar. Within this document, the signer is assumed to be the MASA.

Note that Section 5.1 of [RFC5652] includes a discussion about how to validate a CMS object, which is really a PKCS7 object (cmsVersion=1). Intermediate systems (such the Bootstrapping Remote Secure Key Infrastructures [BRSKI] registrar) that might need to evaluate the voucher in flight MUST be prepared for such an older format. No signaling is necessary, as the manufacturer knows the capabilities of the Pledge and will use an appropriate format voucher for each Pledge.

The CMS structure SHOULD also contain all of 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 third parties cannot accurately audit the transaction without it.

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 a Pledge-provided nonce provides a freshness guarantee.

7. 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 JSON-encoded, and CMS signed encoding of the data defined in the YANG module Section 7.3.

This format is described here as a practical basis for some uses (such as in NETCONF), but more to clearly indicate what vouchers look like in practice. This description also serves to validate the YANG data model.

[RFC8366] defined a media type and a filename extension for the CMS-encoded JSON type. The media types for JOSE format vouchers is defined in [jBRSKI] and the COSE format voucher is defined in [cBRSKI].

The Media Type is used by the Pledge (to the Registrar) and from the Registrar (to the MASA) to signal what format of voucher is expected. Other aspects of the voucher, such as it being nonce-less or which kind of pinned anchor is used is not part of the Media type.

Only the format of voucher that is expected is signaled in the form of a (MIME) Media Content-Type in the HTTP Accept: header.

For vouchers stored/transferred via methods like a USB storage device (USB key), then the voucher format is usually signaled by a filename extension.

7.1. Tree Diagram

The following tree diagram illustrates a high-level view of a voucher document. The notation used in this diagram is described in [RFC8340]. Each node in the diagram is fully described by the YANG module in Section 7.3. Please review the YANG module for a detailed description of the voucher format.

```
module: ietf-voucher

  structure voucher:
    +-- voucher
    |  +-- created-on?                       yang:date-and-time
    |  +-- extensions*                       string
    |  +-- assertion?                        enumeration
    |  +-- serial-number                     string
    |  +-- idevid-issuer?                    binary
    |  +-- (pinning)?
    |  |  +--:(pinned-domain-cert)
    |  |  |  +-- pinned-domain-cert?         binary
    |  |  +--:(pinned-domain-pubk)
    |  |  |  +-- pinned-domain-pubk?         binary
    |  |  +--:(pinned-domain-pubk-sha256)
    |  |     +-- pinned-domain-pubk-sha256?  binary
    |  +-- domain-cert-revocation-checks?    boolean
    |  +-- last-renewal-date?                yang:date-and-time
    +-- (nonceless)?
    |  +--:(expires-on)
    |  |  +-- expires-on?             yang:date-and-time
    |  +--:(nonce)
    |     +-- nonce?                  binary
    +-- est-domain?                   ietf:uri
    +-- additional-configuration-url?  ietf:uri
```

7.2. Examples

This section provides voucher examples for illustration purposes. These examples conform to the encoding rules defined in [RFC8259].

The following example illustrates an ephemeral voucher (uses a nonce). The MASA generated this voucher using the 'logged' assertion type, knowing that it would be suitable for the Pledge making the request.


```
{
  "ietf-voucher:voucher": {
    "created-on": "2016-10-07T19:31:42Z",
    "assertion": "logged",
    "serial-number": "JADA123456789",
    "idevid-issuer": "base64encodedvalue==",
    "pinned-domain-cert": "base64encodedvalue==",
    "nonce": "base64encodedvalue=="
  }
}
```

The following example illustrates a non-ephemeral voucher (no nonce). While the voucher itself expires after two weeks, it presumably can be renewed for up to a year. The MASA generated this voucher using the 'verified' assertion type, which should satisfy all Pledges.


```
{
  "ietf-voucher:voucher": {
    "created-on": "2016-10-07T19:31:42Z",
    "expires-on": "2016-10-21T19:31:42Z",
    "assertion": "verified",
    "serial-number": "JADA123456789",
    "idevid-issuer": "base64encodedvalue==",
    "pinned-domain-cert": "base64encodedvalue==",
    "domain-cert-revocation-checks": true,
    "last-renewal-date": "2017-10-07T19:31:42Z"
  }
}
```


[jBRSKI], Section 8 contains examples of vouchers encoded in JSON, and signed with [JWS]. [cBRSKI], Section 9 contains examples of vouchers encoded in CBOR, and signed with [COSE].

7.3. YANG Module

```
<CODE BEGINS>
=============== NOTE: '\' line wrapping per RFC 8792 ================

module ietf-voucher {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-voucher";
  prefix vch;

  import ietf-yang-types {
    prefix yang;
    reference
      "RFC 6991: Common YANG Data Types";
  }
  import ietf-inet-types {
    prefix ietf;
    reference
      "RFC 6991: Common YANG Data Types";
  }
  import ietf-yang-structure-ext {
    prefix sx;
  }

  organization
    "IETF ANIMA Working Group";
```

```
     contact
       "WG Web:   <https://datatracker.ietf.org/wg/anima/>
        WG List:  <mailto:anima@ietf.org>
        Author:   Kent Watsen
                  <mailto:kent+ietf@watsen.net>
        Author:   Michael Richardson
                  <mailto:mcr+ietf@sandelman.ca>
        Author:   Max Pritikin
                  <mailto:pritikin@cisco.com>
        Author:   Toerless Eckert
                  <mailto:tte@cs.fau.de>
        Author:   Qiufang Ma
                  <mailto:maqiufang1@huawei.com>";
     description
       "This module defines the format for a voucher, which is
        produced by a pledge’s manufacturer or delegate (MASA)
        to securely assign a pledge to an ’owner’, so that the
        pledge may establish a secure connection to the owner’s
        network infrastructure.

        Copyright (c) 2024 IETF Trust and the persons identified as
        authors of the code.  All rights reserved.

        Redistribution and use in source and binary forms, with or
        without modification, is permitted pursuant to, and subject to
        the license terms contained in, the Revised BSD License set
        forth in Section 4.c of the IETF Trust’s Legal Provisions
        Relating to IETF Documents
        (https://trustee.ietf.org/license-info).

        This version of this YANG module is part of RFC XXXX
        (https://www.rfc-editor.org/info/rfcXXXX); see the RFC itself
        for full legal notices.

        RFCEDITOR: please replace XXXX with the RFC number assigned.

        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 (RFC 2119) (RFC 8174) when, and only when,
        they appear in all capitals, as shown here.";

     revision 2023-01-10 {
       description
         "updated to support new assertion enumerated type";
       reference
         "RFC XXXX Voucher Profile for Bootstrapping Protocols";
     }
```

```
     grouping voucher-artifact-grouping {
       description
         "Grouping to allow reuse/extensions in future work.";
       container voucher {
         description
           "A voucher assigns a pledge to an owner using
            the (pinned-domain-cert) value.";
         leaf created-on {
           type yang:date-and-time;
           description
             "A value indicating the date this voucher was created.
              This node is primarily for human consumption and auditing.
              Future work MAY create verification requirements based on
              this node.";
         }
         leaf-list extensions {
           type string {
             length "1..40";
           }
           description
             "A list of extension names that are used in this Voucher
              file.  Each name is registered with the IANA and
              described in an RFC.";
         }
         leaf assertion {
           type enumeration {
             enum verified {
               value 0;
               description
                 "Indicates that the ownership has been positively
                  verified by the MASA (e.g., through sales channel
                  integration).";
             }
             enum logged {
               value 1;
               description
                 "Indicates that the voucher has been issued after
                  minimal verification of ownership or control.  The
                  issuance has been logged for detection of
                  potential security issues (e.g., recipients of
                  vouchers might verify for themselves that unexpected
                  vouchers are not in the log).  This is similar to
                  unsecured trust-on-first-use principles but with the
                  logging providing a basis for detecting unexpected
                  events.";
             }
             enum proximity {
               value 2;
```

```
              description
                "Indicates that the voucher has been issued after
                 the MASA verified a proximity proof provided by the
                 device and target domain.  The issuance has been
                 logged for detection of potential security issues.";
            }
            enum agent-proximity {
              value 3;
              description
                "Mostly identical to proximity, but
                 indicates that the voucher has been issued
                 after the MASA has verified a statement that
                 a registrar agent has made contact with the device.";
            }
          }
          description
            "The assertion is a statement from the MASA regarding how
             the owner was verified.  This statement enables pledges
             to support more detailed policy checks.  Pledges MUST
             ensure that the assertion provided is acceptable, per
             local policy, before processing the voucher.";
        }
        leaf serial-number {
          type string;
          mandatory true;
          description
            "The serial-number of the hardware.  When processing a
             voucher, a pledge MUST ensure that its serial-number
             matches this value.  If no match occurs, then the
             pledge MUST NOT process this voucher.";
        }
        leaf idevid-issuer {
          type binary;
          description
            "The Authority Key Identifier OCTET STRING (as defined in
             Section 4.2.1.1 of RFC 5280) from the pledge's IDevID
             certificate.  Optional since some serial-numbers are
             already unique within the scope of a MASA.
             Inclusion of the statistically unique key identifier
             ensures statistically unique identification of the
             hardware.
             When processing a voucher, a pledge MUST ensure that its
             IDevID Authority Key Identifier matches this value.  If no
             match occurs, then the pledge MUST NOT process this
             voucher.
             When issuing a voucher, the MASA MUST ensure that this
             field is populated for serial-numbers that are not
             otherwise unique within the scope of the MASA.";
```

```
         }
         choice pinning {
           description
             "One of these attributes is used by the pledge to
              specify the registrar, and how the pledge would like
              the registrar’s identity to be pinned";
           leaf pinned-domain-cert {
             type binary;
             description
               "An X.509 v3 certificate structure, as specified by
                RFC 5280, using Distinguished Encoding Rules (DER)
                encoding, as defined in ITU-T X.690.

                This certificate is used by a pledge to trust a Public \
                                                                    Key
                Infrastructure in order to verify a domain certificate
                supplied to the pledge separately by the bootstrapping
                protocol.  The domain certificate MUST have this
                certificate somewhere in its chain of certificates.
                This certificate MAY be an end-entity certificate,
                including a self-signed entity.";
             reference
               "RFC 5280:
                  Internet X.509 Public Key Infrastructure Certificate
                  and Certificate Revocation List (CRL) Profile.
                ITU-T X.690:
                   Information technology - ASN.1 encoding rules:
                   Specification of Basic Encoding Rules (BER),
                   Canonical Encoding Rules (CER) and Distinguished
                   Encoding Rules (DER).";
           }
           leaf pinned-domain-pubk {
             type binary;
             description
               "The pinned-domain-pubk may replace the
                pinned-domain-cert in constrained uses of
                the voucher. The pinned-domain-pubk
                is the Raw Public Key of the Registrar.
                This field is encoded as a Subject Public Key Info block
                as specified in RFC7250, in 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-domain-pubk-sha256 {
             type binary;
```

```
            description
              "The pinned-domain-pubk-sha256 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
               specification which can define a new leaf for another
               hash type.";
          }
        }
        leaf domain-cert-revocation-checks {
          type boolean;
          description
            "A processing instruction to the pledge that it MUST (true)
             or MUST NOT (false) verify the revocation status for the
             pinned domain certificate.  If this field is not set, then
             normal PKIX behavior applies to validation of the domain
             certificate.";
        }
        leaf last-renewal-date {
          type yang:date-and-time;
          must '../expires-on';
          description
            "The date that the MASA projects to be the last date it
             will renew a voucher on. This field is merely
             informative; it is not processed by pledges.

             Circumstances may occur after a voucher is generated that
             may alter a voucher's validity period.  For instance,
             a vendor may associate validity periods with support
             contracts, which may be terminated or extended
             over time.";
        }
      }
      choice nonceless {
        description
          "Either a nonce must be present, or an expires-on header";
        leaf expires-on {
          type yang:date-and-time;
          description
            "A value indicating when this voucher expires.  The node is
             optional as not all pledges support expirations, such as
             pledges lacking a reliable clock.
```

```
              If this field exists, then the pledges MUST ensure that
              the expires-on time has not yet passed. A pledge without
              an accurate clock cannot meet this requirement.

              The expires-on value MUST NOT exceed the expiration date
              of any of the listed 'pinned-domain-cert' certificates.";
         }
         leaf nonce {
           type binary {
             length "8..32";
           }
           description
             "A value that can be used by a pledge in some bootstrapping
              protocols to enable anti-replay protection.  This node is
              optional because it is not used by all bootstrapping
              protocols.

              When present, the pledge MUST compare the provided nonce
              value with another value that the pledge randomly
              generated and sent to a bootstrap server in an earlier
              bootstrapping message.  If the value is present, but
              the values do not match, then the pledge MUST NOT process
              this voucher.";
         }
       }
       leaf est-domain {
         type ietf:uri;
         description
           "The est-domain is a URL from which the Pledge should
              continue doing enrollment rather than with the
              Cloud Registrar.
              The pinned-domain-cert contains a trust-anchor
              which is to be used to authenticate the server
              found at this URI.
             ";
       }
       leaf additional-configuration-url {
         type ietf:uri;
         description
           "The additional-configuration attribute contains a
              URL to which the Pledge can retrieve additional
              configuration information.
              The contents of this URL are manufacturer specific.
              This is intended to do things like configure
              a VoIP phone to point to the correct hosted
              PBX, for example.";
       }
     }
```

```
     // Top-level statement
     sx:structure voucher {
       uses voucher-artifact-grouping;
     }
   }
   <CODE ENDS>
```

7.4. ietf-voucher SID values

[RFC9148] explains how to serialize YANG into CBOR, and for this a series of SID values are required. While [CORESID] defines the management process for these values, due to the immaturity of the tooling around this YANG-SID mechanisms, the following values are considered normative. It is believed, however, that they will not change.

```
          SID Assigned to
   --------- --------------------------------------------------
        2451 data /ietf-voucher:voucher/voucher
        2452 data /ietf-voucher:voucher/voucher/assertion
        2453 data /ietf-voucher:voucher/voucher/created-on
        2454 data .../domain-cert-revocation-checks
        2455 data /ietf-voucher:voucher/voucher/expires-on
        2456 data /ietf-voucher:voucher/voucher/idevid-issuer
        2457 data /ietf-voucher:voucher/voucher/last-renewal-date
        2458 data /ietf-voucher:voucher/nonce
        2459 data /ietf-voucher:voucher/voucher/pinned-domain-cert
        2460 data /ietf-voucher:voucher/voucher/pinned-domain-pubk
        2461 data .../pinned-domain-pubk-sha256
        2462 data /ietf-voucher:voucher/voucher/serial-number
        2463 data .../additional-configuration-url
        2464 data /ietf-voucher:voucher/voucher/est-domain
        2465 data .../additional-configuration-url
        2466 data /ietf-voucher:voucher/est-domain
        2467 data /ietf-voucher:voucher/expires-on
        2468 data /ietf-voucher:voucher/voucher/extensions
```

WARNING, obsolete definitions

The "assertion" attribute is an enumerated type in [RFC8366], but no values were provided as part of the enumeration. This document provides enumerated values as part of the YANG module.

In the JSON serialization, the literal strings from the enumerated types are used so there is no ambiguity.

In the CBOR serialization, a small integer is used, and the following values are repeated here. The YANG module should be considered authoritative in the future. No IANA registry is provided or necessary because the YANG module (and this document) would be extended when there are new entries to make.

Integer	Assertion Type
0	verified
1	logged
2	proximity
3	agent-proximity

Table 2: CBOR integers for the "assertion" attribute enum

7.5. Voucher Extensions

An unstated assumption in [RFC8366] was that vouchers could be extended in proprietary ways by manufacturers. This allows for manufacturers to communicate new things from the MASA to the Pledge, and since both are under control of the same entity, it seemed perfectly fine, even though it would violate the strict definition of the YANG.

The JSON serialization of vouchers implicitely accomodates the above, since the voucher is just a map (or dictionary). Map keys are just strings, and creating unique strings is easy to do by including the manufacturer’s domain name.

In CBOR serialization [RFC9148]/[RFC9254], the situation is not so easy. The delta encoding for keys requires new keys to use the absolute Tag(47) for new entities. An extension might need to use the Private Use SID values, or acquire SID values for their own use.

Where the process has become complex is when making standard extensions, as has happened recently, resulting in this document. The processes which were anticipated to be useful, (the "augment" mechanism) turned out not to be the case, see Section 5.1.

Instead, a process similiar to what was done by [RFC8520] has been adopted. In this, extensions are listed in a leaf named "extensions". In JSON serialization, these extensions require a unique name, and this MUST be allocated by IANA. The name MUST be the same as the YANG module name. The "extensions" list attribute definied in this model allows for new standard extensions to be defined. Items within that list are strings (in JSON serialization), or integers (in CBOR serialization), as defined by the Voucher Extension Registry.

Extensions are full YANG modules, which are subject to the SID allocation process described in [RFC9254]. When an extension is serialized, the extension is placed in a sub-map in the value section. In JSON serialization, the key is the name of the extension, prefixed by the string "extension:" In CBOR serialization, the key is the SID which is allocated as the module SID. This will typically require the absolute Tag(47) to be applied to this key.

Note that this differs from the mechanism described in [RFC8520] in that a sub-map is not used. Instead keys are created by combining the module name and the attribute as a string. The [RFC8520] mechanism uses more bytes, but is also not translateable easily to CBOR.

As the Voucher Request artifact is created by augment on the voucher artifact, any extension defined for the voucher is also valid for Voucher Requests.

7.6. Manufacturer Private extensions

In addition to the above described extensions mechanism, a manufacturer might need to communicate content in the voucher (or in the voucher-request), which are never subject to standardization. While they can use the mechanism above, it does require allocation of a SID in order to do minimal sized encoding. Note that [RFC9254] does not require use of SIDs.

Instead, a manufacturer MAY use the manufacturer-private leaf to put any content they wish. In CBOR serialization, if a map is used, then it will be subject to delta encoding, so use of this leaf requires that the content are bstr-encoded [RFC8949], Section 3.1 (Major type 2). In JSON serialization, delta-encoding does not get in the way, and the manufacturer MAY use any encoding that is convenient for them, but base64URL encoding [RFC4648], Section 5 is RECOMMENDED.

8. Voucher Request Artifact

[BRSKI], Section 3 defined a Voucher-Request Artifact as an augmented artifact from the Voucher Artifact originally defined in [RFC8366]. That definition has been moved to this document, and translated from YANG-DATA [RFC8040] to the SX:STRUCTURE extension [RFC8791].

8.1. Tree Diagram

The following tree diagram illustrates a high-level view of a voucher request document. The notation used in this diagram is described in [RFC8340]. Each node in the diagram is fully described by the YANG module in Section 8.2.

```
module: ietf-voucher-request

  structure voucher:
    +-- voucher
    |  +-- created-on?
    |  |       yang:date-and-time
    |  +-- extensions*                                 string
    |  +-- assertion?                                  enumeration
    |  +-- serial-number                               string
    |  +-- idevid-issuer?                              binary
    |  +-- (pinning)?
    |  |  +--:(pinned-domain-cert)
    |  |  |  +-- pinned-domain-cert?                   binary
    |  |  +--:(pinned-domain-pubk)
    |  |  |  +-- pinned-domain-pubk?                   binary
    |  |  +--:(pinned-domain-pubk-sha256)
    |  |     +-- pinned-domain-pubk-sha256?            binary
    |  +-- domain-cert-revocation-checks?              boolean
    |  +-- last-renewal-date?
    |  |       yang:date-and-time
    |  +-- prior-signed-voucher-request?               binary
    |  +-- (registrar-identity)?
    |  |  +--:(proximity-registrar-cert)
    |  |  |  +-- proximity-registrar-cert?             binary
    |  |  +--:(proximity-registrar-pubk)
    |  |  |  +-- proximity-registrar-pubk?             binary
    |  |  +--:(proximity-registrar-pubk-sha256)
    |  |     +-- proximity-registrar-pubk-sha256?      binary
    |  +-- agent-signed-data?                          binary
    |  +-- agent-provided-proximity-registrar-cert?   binary
    |  +-- agent-sign-cert?                            binary
    +-- (nonceless)?
    |  +--:(expires-on)
    |  |  +-- expires-on?              yang:date-and-time
    |  +--:(nonce)
    |     +-- nonce?                   binary
    +-- est-domain?                    ietf:uri
    +-- additional-configuration-url?   ietf:uri
```

8.2. "ietf-voucher-request" Module

The ietf-voucher-request YANG module is derived from the ietf-voucher module.

```
<CODE BEGINS>
=============== NOTE: '\' line wrapping per RFC 8792 ================

module ietf-voucher-request {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-voucher-request";
  prefix vcr;

  import ietf-yang-structure-ext {
    prefix sx;
  }
  import ietf-voucher {
    prefix vch;
    description
      "This module defines the format for a voucher,
       which is produced by a pledge's manufacturer or
       delegate (MASA) to securely assign a pledge to
       an 'owner', so that the pledge may establish a secure
       connection to the owner's network infrastructure";
    reference
      "RFC XXXX: Voucher Artifact for
       Bootstrapping Protocols";
  }

  organization
    "IETF ANIMA Working Group";
  contact
    "WG Web:   <https://datatracker.ietf.org/wg/anima/>
     WG List:  <mailto:anima@ietf.org>
     Author:   Kent Watsen
               <mailto:kent+ietf@watsen.net>
     Author:   Michael Richardson
               <mailto:mcr+ietf@sandelman.ca>
     Author:   Max Pritikin
               <mailto:pritikin@cisco.com>
     Author:   Toerless Eckert
               <mailto:tte@cs.fau.de>
     Author:   Qiufang Ma
               <mailto:maqiufang1@huawei.com>";
  description
    "This module defines the format for a voucher request.
     It is a superset of the voucher itself.
     It provides content to the MASA for consideration
     during a voucher request.

     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 (RFC 2119) (RFC 8174) when, and only when,
        they appear in all capitals, as shown here.

         Copyright (c) 2024 IETF Trust and the persons identified as
         authors of the code.  All rights reserved.

         Redistribution and use in source and binary forms, with or
         without modification, is permitted pursuant to, and subject to
         the license terms contained in, the Revised BSD License set
         forth in Section 4.c of the IETF Trust's Legal Provisions
         Relating to IETF Documents
         (https://trustee.ietf.org/license-info).

         This version of this YANG module is part of RFC XXXX
         (https://www.rfc-editor.org/info/rfcXXXX); see the RFC itself
         for full legal notices.

        RFCEDITOR: please replace XXXX with the RFC number assigned.

         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 (RFC 2119) (RFC 8174) when, and only when,
         they appear in all capitals, as shown here.";

     revision 2023-01-10 {
       description
         "Initial version";
       reference
         "RFC XXXX: Bootstrapping Remote Secure Key Infrastructure";
     }

     grouping voucher-request-grouping {
       description
         "Grouping to allow reuse/extensions in future work.";
       uses vch:voucher-artifact-grouping {
         refine "voucher/created-on" {
           mandatory false;
         }
         refine "voucher/last-renewal-date" {
           description
             "A last-renewal-date field
              is not valid in a voucher request, and
              any occurrence MUST be ignored";
         }
         refine "voucher/domain-cert-revocation-checks" {
           description
             "The domain-cert-revocation-checks field
```

```
              is not valid in a voucher request, and
              any occurrence MUST be ignored";
         }
         refine "voucher/assertion" {
           mandatory false;
           description
             "Any assertion included in registrar voucher
              requests SHOULD be ignored by the MASA.";
         }
         augment "voucher" {
           description
             "Adds leaf nodes appropriate for requesting vouchers.";
           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 voucher request
                with a proximity-registrar-cert, and the registrar
                then includes it as the prior-signed-voucher-request
                field.  This is a simple mechanism for a chain of
                trusted parties to change a voucher request, while
                maintaining the prior signature information.

                The Registrar and MASA 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.";
           }
           choice registrar-identity {
             description
               "One of these three attributes will be used to pin the \
                                                  registrar identity";
             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.X690.1994].
```

```
             The first certificate in the Registrar TLS server
             certificate_list sequence  (the end-entity TLS
             certificate, see [RFC8446]) presented by the Registrar
             to the Pledge.
             This MUST be populated in a Pledge’s voucher request
             when a proximity assertion is requested.";
        }
        leaf proximity-registrar-pubk {
          type binary;
          description
            "The proximity-registrar-pubk replaces
             the proximity-registrar-cert in constrained uses of
             the voucher-request.
             The proximity-registrar-pubk 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, but due to
             size is discouraged.";
        }
        leaf proximity-registrar-pubk-sha256 {
          type binary;
          description
            "The proximity-registrar-pubk-sha256
             is an alternative to both
             proximity-registrar-pubk 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
             specification which may define a new leaf for another
             hash type.";
        }
      }
      leaf agent-signed-data {
        type binary;
        description
          "The agent-signed-data field contains a data artifact \
                                                          provided
           by the Registrar-Agent to the Pledge for inclusion \
```

```
                                                          into the
              voucher request.

              This artifact is signed by the Registrar-Agent and \
                                                          contains
              data, which can be verified by the pledge and the \
                                                         registrar.
              This data contains the pledge’s serial-number and a \
                                                        created-on
              information of the agent-signed-data.

              The format is intentionally defined as binary to allow
              the document using this leaf to determine the encoding.\
                                                                  ";
          }
          leaf agent-provided-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 X.690.
               The first certificate in the registrar TLS server
               certificate_list sequence (the end-entity TLS
               certificate; see RFC 8446) presented by the
               registrar to the registrar-agent and provided to
               the pledge.
               This MUST be populated in a pledge’s voucher-request
               when an agent-proximity assertion is requested.";
            reference
              "ITU X.690: Information Technology - ASN.1 encoding
               rules: Specification of Basic Encoding Rules (BER),
               Canonical Encoding Rules (CER) and Distinguished
               Encoding Rules (DER)
               RFC 5280: Internet X.509 Public Key Infrastructure
               Certificate and Certificate Revocation List (CRL)
               Profile
               RFC 8446: The Transport Layer Security (TLS)
               Protocol Version 1.3";
          }
          leaf agent-sign-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 X.690.
               This certificate can be used by the pledge,
```

```
                the registrar, and the MASA to verify the signature
                of agent-signed-data. It is an optional component
                for the pledge-voucher request.
                This MUST be populated in a registrar’s
                voucher-request when an agent-proximity assertion
                is requested.";
             reference
               "ITU X.690: Information Technology - ASN.1 encoding
                rules: Specification of Basic Encoding Rules (BER),
                Canonical Encoding Rules (CER) and Distinguished
                Encoding Rules (DER)
                RFC 5280: Internet X.509 Public Key Infrastructure
                Certificate and Certificate Revocation List (CRL)
                Profile";
           }
         }
       }
     }

     // Top-level statement
     sx:structure voucher {
       uses voucher-request-grouping;
     }
   }
   <CODE ENDS>
```

8.3. ietf-voucher-request SID values

[RFC9148] explains how to serialize YANG into CBOR, and for this a series of SID values are required. While [CORESID] defines the management process for these values, due to the immaturity of the tooling around this YANG-SID mechanisms, the following values are considered normative. It is believed, however, that they will not change.

SID	Assigned to
2501	data /ietf-voucher-request:voucher/voucher
2502	data /ietf-voucher-request:voucher/voucher/assertion
2503	data /ietf-voucher-request:voucher/voucher/created-on
2504	data .../domain-cert-revocation-checks
2505	data /ietf-voucher-request:voucher/voucher/expires-on
2506	data .../idevid-issuer
2507	data .../last-renewal-date
2508	data /ietf-voucher-request:voucher/nonce
2509	data .../pinned-domain-cert
2510	data .../prior-signed-voucher-request
2511	data .../proximity-registrar-cert
2512	data .../proximity-registrar-pubk-sha256
2513	data .../proximity-registrar-pubk
2514	data .../serial-number
2515	data .../agent-provided-proximity-registrar-cert
2516	data .../agent-sign-cert
2517	data .../agent-signed-data
2518	data .../pinned-domain-pubk
2519	data .../pinned-domain-pubk-sha256
2520	data .../additional-configuration-url
2521	data /ietf-voucher-request:voucher/est-domain
2522	data /ietf-voucher-request:voucher/expires-on
2523	data /ietf-voucher-request:voucher/voucher/extensions

WARNING, obsolete definitions

The "assertion" attribute is an enumerated type, and has values as defined above in Table 2.

9. Design Considerations

9.1. Renewals Instead of Revocations

The lifetimes of vouchers may vary. In some onboarding protocols, the vouchers may be created and consumed immediately, whereas in other onboarding solutions, there may be a significant time delay between when a voucher is created and when it is consumed. In cases when there is a time delay, there is a need for the Pledge to ensure that the assertions made when the voucher was created are still valid.

A revocation artifact is generally used to verify the continued validity of an assertion such as a PKIX certificate, web token, or a "voucher". With this approach, a potentially long-lived assertion is paired with a reasonably fresh revocation status check to ensure that the assertion is still valid. However, this approach increases solution complexity, as it introduces the need for additional protocols and code paths to distribute and process the revocations.

Addressing the shortcomings of revocations, this document recommends instead the use of lightweight renewals of short-lived non-revocable vouchers. That is, rather than issue a long-lived voucher, where the 'expires-on' leaf is set to some distant date, the expectation is for the MASA to instead issue a short-lived voucher, where the 'expires-on' leaf is set to a relatively near date, along with a promise (reflected in the 'last-renewal-date' field) to reissue the voucher again when needed. Importantly, while issuing the initial voucher may incur heavyweight verification checks ("Are you who you say you are?" "Does the Pledge actually belong to you?"), reissuing the voucher should be a lightweight process, as it ostensibly only updates the voucher's validity period. With this approach, there is only the one artifact, and only one code path is needed to process it; there is no possibility of a Pledge choosing to skip the revocation status check because, for instance, the OCSP Responder is not reachable.

While this document recommends issuing short-lived vouchers, the voucher artifact does not restrict the ability to create long-lived voucher, if required; however, no revocation method is described.

Note that a voucher may be signed by a chain of intermediate CAs leading up to the trust anchor certificate known by the Pledge. Even though the voucher itself is not revocable, it may still be revoked, per se, if one of the intermediate CA certificates is revoked.

9.2. Voucher Per Pledge

The solution described herein originally enabled a single voucher to apply to many Pledges, using lists of regular expressions to represent ranges of serial-numbers. However, it was determined that blocking the renewal of a voucher that applied to many devices would be excessive when only the ownership for a single Pledge needed to be blocked. Thus, the voucher format now only supports a single serial-number to be listed.

10. Security Considerations

10.1. Clock Sensitivity

An attacker could use an expired voucher to gain control over a device that has no understanding of time. The device cannot trust NTP as a time reference, as an attacker could control the NTP stream.

There are three things to defend against this: 1) devices are required to verify that the expires-on field has not yet passed, 2) devices without access to time can use nonces to get ephemeral vouchers, and 3) vouchers without expiration times may be used, which will appear in the audit log, informing the security decision.

This document defines a voucher format that contains time values for expirations, which require an accurate clock in order to be processed correctly. Vendors planning on issuing vouchers with expiration values must ensure that devices have an accurate clock when shipped from manufacturing facilities and take steps to prevent clock tampering. If it is not possible to ensure clock accuracy, then vouchers with expirations should not be issued.

10.2. Protect Voucher PKI in HSM

Pursuant the recommendation made in Section 6.1 for the MASA to be deployed as an online voucher signing service, it is RECOMMENDED that the MASA’s private key used for signing vouchers is protected by a hardware security module (HSM).

10.3. Test Domain Certificate Validity When Signing

If a domain certificate is compromised, then any outstanding vouchers for that domain could be used by the attacker. The domain administrator is clearly expected to initiate revocation of any domain identity certificates (as is normal in PKI solutions).

Similarly, they are expected to contact the MASA to indicate that an outstanding (presumably short lifetime) voucher should be blocked from automated renewal. Protocols for voucher distribution are RECOMMENDED to check for revocation of domain identity certificates before the signing of vouchers.

10.4. YANG Module Security Considerations

The YANG module specified in this document defines the schema for data that is subsequently encapsulated by a CMS signed-data content type, as described in Section 5 of [RFC5652]. As such, all of the YANG modeled data is protected from modification.

Implementations should be aware that the signed data is only protected from external modification; the data is still visible. This potential disclosure of information doesn’t affect security so much as privacy. In particular, adversaries can glean information such as which devices belong to which organizations and which CRL Distribution Point and/or OCSP Responder URLs are accessed to validate the vouchers. When privacy is important, the CMS signed-data content type SHOULD be encrypted, either by conveying it via a mutually authenticated secure transport protocol (e.g., TLS [RFC5246]) or by encapsulating the signed-data content type with an enveloped-data content type (Section 6 of [RFC5652]), though details for how to do this are outside the scope of this document.

The use of YANG to define data structures, via the ’yang-data’ statement, is relatively new and distinct from the traditional use of YANG to define an API accessed by network management protocols such as NETCONF [RFC6241] and RESTCONF [RFC8040]. For this reason, these guidelines do not follow template described by Section 3.7 of [YANG-GUIDE].

11. IANA Considerations

11.1. The IETF XML Registry

This document registers two URIs in the "IETF XML Registry" [RFC3688].

IANA has registered the following:

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

This reference should be updated to point to this document.

11.2. The YANG Module Names Registry

This document registers two YANG module in the "YANG Module Names" registry [RFC6020].

IANA has registred the following:

    name: ietf-voucher
    namespace: urn:ietf:params:xml:ns:yang:ietf-voucher
    prefix: vch

    reference: :RFC 8366

This reference should be updated to point to this document.

11.3. The Media Types Registry

IANA has registered the media type: voucher-cms+json, and this registration should be updated to point to this document.

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

IANA has registered the OID 1.2.840.113549.1.9.16.1.40, id-ct-animaJSONVoucher. This registration should be updated to point to this document.

11.5. Extensions Registry

IANA is asked to create a registry of extensions as follows:

Registry name: Voucher Extensions Registry
Registry policy: First Come First Served
Reference: an optional document
Extension name: UTF-8-encoded string, not to exceed 40 characters.
Extension SID: the module SID value as allocated

Each extension MUST follow the rules specified in this specification. As is usual, the IANA issues early allocations in accordance with [RFC7120].

Note that the SID module value is allocated as part of a [RFC9595] process. This may be from a SID range managed by IANA, or from any other MegaRange. Future work may allow for PEN based allocations. IANA does not need to separately allocate a SID value for this column.

Extension name strings for standards track documents are single words, given by the YANG Module Name. They do not contain dots. For vendor proprietary extensions, the string SHOULD be made unique by putting the extension name in the form a FQDN [RFC5822], such as "fuubar.example.com"

Vendor proprietary extensions do not need to be registered with IANA, but vendors MAY do so.

Designated Experts should review for standards track documents for clarity, but the process is essentially tied to WG and IESG process: There are no choices in the extension names (which is the YANG module), or SID (which is from another IANA process). For non-standard track extensions, the Designated Expert should review whatever document is provided, if any. The stability of the reference may be of concern. The Desigtnated Expert should determine if the work overlaps existing efforts; and if so suggest merging. However, as registration is optional, the designated expert should not block any registrations.

12. References

12.1. Normative References

[BRSKI] Pritikin, M., Richardson, M., Eckert, T., Behringer, M., and K. Watsen, "Bootstrapping Remote Secure Key Infrastructure (BRSKI)", RFC 8995, DOI 10.17487/RFC8995, May 2021, <https://www.rfc-editor.org/rfc/rfc8995>.

[cBRSKI] Richardson, M., Van der Stok, P., Kampanakis, P., and E. Dijk, "Constrained Bootstrapping Remote Secure Key Infrastructure (cBRSKI)", Work in Progress, Internet-Draft, draft-ietf-anima-constrained-voucher-27, 3 March 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-anima-constrained-voucher-27>.

[CLOUD] Friel, O., Shekh-Yusef, R., and M. Richardson, "BRSKI Cloud Registrar", Work in Progress, Internet-Draft, draft-ietf-anima-brski-cloud-14, 3 March 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-anima-brski-cloud-14>.

[CORESID] Veillette, M., Ed., Pelov, A., Ed., Petrov, I., Ed., Bormann, C., and M. Richardson, "YANG Schema Item iDentifier (YANG SID)", RFC 9595, DOI 10.17487/RFC9595, July 2024, <https://www.rfc-editor.org/rfc/rfc9595>.

[ITU-T.X690.2015] International Telecommunication Union, "Information Technology - ASN.1 encoding rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER)", ITU-T Recommendation X.690, ISO/IEC 8825-1, August 2015, <https://www.itu.int/rec/T-REC-X.690/>.

[jBRSKI] Werner, T. and M. Richardson, "JWS signed Voucher Artifacts for Bootstrapping Protocols", Work in Progress, Internet-Draft, draft-ietf-anima-jws-voucher-16, 15 January 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-anima-jws-voucher-16>.

[PRM] Fries, S., Werner, T., Lear, E., and M. Richardson, "BRSKI with Pledge in Responder Mode (BRSKI-PRM)", Work in Progress, Internet-Draft, draft-ietf-anima-brski-prm-18, 11 February 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-anima-brski-prm-18>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/rfc/rfc2119>.

[RFC5652] Housley, R., "Cryptographic Message Syntax (CMS)", STD 70, RFC 5652, DOI 10.17487/RFC5652, September 2009, <https://www.rfc-editor.org/rfc/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/rfc/rfc6020>.

[RFC7120] Cotton, M., "Early IANA Allocation of Standards Track Code Points", BCP 100, RFC 7120, DOI 10.17487/RFC7120, January 2014, <https://www.rfc-editor.org/rfc/rfc7120>.

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

[RFC7951] Lhotka, L., "JSON Encoding of Data Modeled with YANG", RFC 7951, DOI 10.17487/RFC7951, August 2016, <https://www.rfc-editor.org/rfc/rfc7951>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/rfc/rfc8174>.

[RFC8259] Bray, T., Ed., "The JavaScript Object Notation (JSON) Data Interchange Format", STD 90, RFC 8259, DOI 10.17487/RFC8259, December 2017, <https://www.rfc-editor.org/rfc/rfc8259>.

[RFC8791] Bierman, A., Bj旦rklund, M., and K. Watsen, "YANG Data Structure Extensions", RFC 8791, DOI 10.17487/RFC8791, June 2020, <https://www.rfc-editor.org/rfc/rfc8791>.

[RFC9148] van der Stok, P., Kampanakis, P., Richardson, M., and S. Raza, "EST-coaps: Enrollment over Secure Transport with the Secure Constrained Application Protocol", RFC 9148, DOI 10.17487/RFC9148, April 2022, <https://www.rfc-editor.org/rfc/rfc9148>.

[RFC9254] Veillette, M., Ed., Petrov, I., Ed., Pelov, A., Bormann, C., and M. Richardson, "Encoding of Data Modeled with YANG in the Concise Binary Object Representation (CBOR)", RFC 9254, DOI 10.17487/RFC9254, July 2022, <https://www.rfc-editor.org/rfc/rfc9254>.

[RFC9595] Veillette, M., Ed., Pelov, A., Ed., Petrov, I., Ed., Bormann, C., and M. Richardson, "YANG Schema Item iDentifier (YANG SID)", RFC 9595, DOI 10.17487/RFC9595, July 2024, <https://www.rfc-editor.org/rfc/rfc9595>.

[ZERO-TOUCH] Watsen, K., Farrer, I., and M. Abrahamsson, "Secure Zero Touch Provisioning (SZTP)", RFC 8572, DOI 10.17487/RFC8572, April 2019, <https://www.rfc-editor.org/rfc/rfc8572>.

12.2. Informative References

[CBOR] Internet Standard 94, <https://www.rfc-editor.org/info/std94>. At the time of writing, this STD comprises the following:

Bormann, C. and P. Hoffman, "Concise Binary Object Representation (CBOR)", STD 94, RFC 8949, DOI 10.17487/RFC8949, December 2020, <https://www.rfc-editor.org/info/rfc8949>.

[COSE] Internet Standard 96, <https://www.rfc-editor.org/info/std96>. At the time of writing, this STD comprises the following:

Schaad, J., "CBOR Object Signing and Encryption (COSE): Structures and Process", STD 96, RFC 9052, DOI 10.17487/RFC9052, August 2022, <https://www.rfc-editor.org/info/rfc9052>.

Schaad, J., "CBOR Object Signing and Encryption (COSE): Countersignatures", STD 96, RFC 9338, DOI 10.17487/RFC9338, December 2022, <https://www.rfc-editor.org/info/rfc9338>.

[fairhair] Open Connectivity Foundation, "Fairhair Specification", 1 November 2019, <https://openconnectivity.org/developer/specifications/fairhair/>.

[I-D.selander-ace-ake-authz] Selander, G., Mattsson, J. P., Vuini, M., Richardson, M., and A. Schellenbaum, "Lightweight Authorization for Authenticated Key Exchange.", Work in Progress, Internet-Draft, draft-selander-ace-ake-authz-05, 18 April 2022, <https://datatracker.ietf.org/doc/html/draft-selander-ace-ake-authz-05>.

[imprinting] Wikipedia, "Wikipedia article: Imprinting", February 2018, <https://en.wikipedia.org/w/index.php?title=Imprinting_(psychology)&oldid=825757556>.

[JWS] Jones, M., Bradley, J., and N. Sakimura, "JSON Web Signature (JWS)", RFC 7515, DOI 10.17487/RFC7515, May 2015, <https://www.rfc-editor.org/rfc/rfc7515>.

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

[RFC4648] Josefsson, S., "The Base16, Base32, and Base64 Data Encodings", RFC 4648, DOI 10.17487/RFC4648, October 2006, <https://www.rfc-editor.org/rfc/rfc4648>.

[RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, DOI 10.17487/RFC5246, August 2008, <https://www.rfc-editor.org/rfc/rfc5246>.

[RFC5822] "Not Issued", RFC 5822, <https://www.rfc-editor.org/rfc/rfc5822>.

[RFC6125] Saint-Andre, P. and J. Hodges, "Representation and Verification of Domain-Based Application Service Identity within Internet Public Key Infrastructure Using X.509 (PKIX) Certificates in the Context of Transport Layer Security (TLS)", RFC 6125, DOI 10.17487/RFC6125, March 2011, <https://www.rfc-editor.org/rfc/rfc6125>.

[RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011, <https://www.rfc-editor.org/rfc/rfc6241>.

[RFC7435] Dukhovni, V., "Opportunistic Security: Some Protection Most of the Time", RFC 7435, DOI 10.17487/RFC7435, December 2014, <https://www.rfc-editor.org/rfc/rfc7435>.

[RFC8040] Bierman, A., Bjorklund, M., and K. Watsen, "RESTCONF Protocol", RFC 8040, DOI 10.17487/RFC8040, January 2017, <https://www.rfc-editor.org/rfc/rfc8040>.

[RFC8340] Bjorklund, M. and L. Berger, Ed., "YANG Tree Diagrams", BCP 215, RFC 8340, DOI 10.17487/RFC8340, March 2018, <https://www.rfc-editor.org/rfc/rfc8340>.

[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/rfc/rfc8366>.

[RFC8520] Lear, E., Droms, R., and D. Romascanu, "Manufacturer Usage Description Specification", RFC 8520, DOI 10.17487/RFC8520, March 2019, <https://www.rfc-editor.org/rfc/rfc8520>.

[SECUREJOIN] Richardson, M., "6tisch Secure Join protocol", Work in Progress, Internet-Draft, draft-ietf-6tisch-dtsecurity-secure-join-01, 25 February 2017, <https://datatracker.ietf.org/doc/html/draft-ietf-6tisch-dtsecurity-secure-join-01>.

[Stajano99theresurrecting] Stajano, F. and R. Anderson, "The Resurrecting Duckling: Security Issues for Ad-Hoc Wireless Networks", 1999, <https://www.cl.cam.ac.uk/research/dtg/www/files/publications/public/files/tr.1999.2.pdf>.

[YANG-GUIDE] Bierman, A., "Guidelines for Authors and Reviewers of Documents Containing YANG Data Models", BCP 216, RFC 8407, DOI 10.17487/RFC8407, October 2018, <https://www.rfc-editor.org/rfc/rfc8407>.

[YANGCBOR] Veillette, M., Ed., Petrov, I., Ed., Pelov, A., Bormann, C., and M. Richardson, "Encoding of Data Modeled with YANG in the Concise Binary Object Representation (CBOR)", RFC 9254, DOI 10.17487/RFC9254, July 2022, <https://www.rfc-editor.org/rfc/rfc9254>.

Appendix A. Examples

A.1. Key pairs associated with examples

The following voucher request has been produced using the IDevID public (certificate) and private key. They are included so that other developers can match the same output.

The private RSA key:


```
-----BEGIN EC PRIVATE KEY-----
MHcCAQEEIBHNh6r8QRevRuo+tEmBJeFjQKf6bpFA/9NGoltv+9sNoAoGCCqGSM49
AwEHoUQDQgAEA6N1Q4ezfMAKmoecrfb0OBMc1AyEH+BATkF58FsTSyBxs0SbSWLx
FjDOuwB9gLGn2TsTUJumJ6VPw5Z/TP4hJw==
-----END EC PRIVATE KEY-----
```


The IDevID certificate (public key):


```
-----BEGIN CERTIFICATE-----
MIIBrzCCATWgAwIBAgIEHxj+5zAKBggqhkjOPQQDAjAmMSQwIgYDVQQDDBtoaWdo
d2F5LXRlc3QuZXhhbXBsZS5jb20gQ0EwIBcNMjEwNDI3MTgyOTMwWhgPMjk5OTEy
MzEwMDAwMDBaMBwxGjAYBgNVBAUTETAwLUQwLUU1LUYyLTAwLTAyMFkwEwYHKoZI
zj0CAQYIKoZIzj0DAQcDQgAEA6N1Q4ezfMAKmoecrfb0OBMc1AyEH+BATkF58FsT
SyBxs0SbSWLxFjDOuwB9gLGn2TsTUJumJ6VPw5Z/TP4hJ6NZMFcwHQYDVR0OBBYE
FEWIzJaWAGQ3sLojZWRkVAgGbFatMAkGA1UdEwQCMAAwKwYIKwYBBQUHASAEHxYd
aGlnaHdheS10ZXN0LmV4YW1wbGUuY29tOjk0NDMwCgYIKoZIzj0EAwIDaAAwZQIw
YirbvjT3G8uF3iaOQwD5DYjId6jdPAhAVLzsPbbccCvDf8oZIZqgq8VRjqrfNt6L
AjEAsl1Z+EfH7QOXqMDHqIH6qIbtZ2Q3UXpunKOCTW2tvPM1np1qom1/fyUcA+/w
uptx
-----END CERTIFICATE-----
```


The Certification Authority that created the IDevID:

```
=============== NOTE: '\' line wrapping per RFC 8792 ================

Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number: 1016146354 (0x3c9129b2)
        Signature Algorithm: sha256WithRSAEncryption
        Issuer: CN = highway-test.example.com CA
        Validity
```

```
               Not Before: Apr  5 19:36:57 2021 GMT
               Not After : May  6 05:36:57 2021 GMT
           Subject: CN = highway-test.example.com CA
           Subject Public Key Info:
               Public Key Algorithm: rsaEncryption
                   Public-Key: (3072 bit)
                   Modulus:
                       00:b4:7b:27:42:49:9f:ed:85:47:74:ff:f6:50:cd:
                       5d:22:1a:64:38:22:f8:09:d2:d6:f3:60:d8:98:7f:
                       e5:84:52:1e:d9:ce:96:b4:dc:a6:43:74:67:27:d9:
                       9d:42:7d:bf:1a:43:92:9b:d1:dd:34:9b:41:d2:e3:
                       d5:59:b3:40:fc:b3:c9:e1:58:84:3f:87:f7:06:45:
                       25:26:4c:bf:a1:45:72:a0:0a:5b:86:41:d7:8e:be:
                       d3:38:b5:aa:66:69:bd:3a:fd:e9:b5:b8:a2:79:c4:
                       f0:a5:3c:9e:91:94:32:1e:9c:b0:7f:25:46:5b:76:
                       1d:86:23:85:b0:62:45:5c:a8:6f:fb:c5:26:e1:dd:
                       a8:f2:68:ab:c5:8c:b4:58:b4:2e:96:49:fa:fe:d2:
                       ea:a5:11:68:c2:8d:f4:58:ab:30:bd:dd:1b:29:97:
                       00:18:6f:59:40:9c:3a:2a:e4:96:25:bb:12:f4:1a:
                       11:72:6d:31:f6:b4:e1:cc:d8:9a:0c:aa:a8:aa:a4:
                       64:e3:f1:06:1c:c0:09:df:62:ba:04:cb:70:b0:c4:
                       f7:ca:35:22:ea:a9:c7:52:e1:ce:27:fb:6c:52:39:
                       b7:22:b3:5d:97:cb:0a:9f:75:a3:af:16:ef:e6:b2:
                       1b:6a:c3:0b:1d:15:fd:b8:d8:e7:8a:f6:f4:99:1c:
                       23:97:4b:80:e9:79:a3:85:16:f8:dd:bd:77:ef:3a:
                       3c:8e:e7:75:56:67:36:3a:dd:42:7b:84:2f:64:2f:
                       13:0e:fa:b0:3b:11:13:7e:ae:78:a6:2f:46:dd:4b:
                       11:88:e4:7b:19:ab:21:2d:1f:34:ba:61:cd:51:84:
                       a5:ec:6a:c1:90:20:70:e3:aa:f4:01:fd:0c:6e:cd:
                       04:47:99:31:70:79:6c:af:41:78:c1:04:2a:43:78:
                       84:8a:fe:c3:3d:f2:41:c8:2a:a1:10:e0:b7:b4:4f:
                       4e:e6:26:79:ac:49:64:cf:57:1e:2e:e3:2f:58:bd:
                       6f:30:00:67:d7:8b:d6:13:60:bf
                   Exponent: 65537 (0x10001)
           X509v3 extensions:
               X509v3 Basic Constraints: critical
                   CA:TRUE
               X509v3 Key Usage: critical
                   Certificate Sign, CRL Sign
               X509v3 Subject Key Identifier:
                   33:12:45:B7:1B:10:BE:F3:CB:64:E5:4C:50:80:7C:9D:88:\
                                                                65:74:40
               X509v3 Authority Key Identifier:
                   33:12:45:B7:1B:10:BE:F3:CB:64:E5:4C:50:80:7C:9D:88:\
                                                                65:74:40
       Signature Algorithm: sha256WithRSAEncryption
       Signature Value:
           05:37:28:85:37:39:71:87:ec:5c:f0:51:19:55:4a:b7:e0:2a:
```

```
            e6:61:30:d4:e2:2b:ad:7a:db:12:fc:8a:a6:6e:15:82:80:10:
            fa:5d:67:60:e8:54:14:e3:89:d6:4e:60:89:98:5b:ab:fe:32:
            26:aa:02:35:68:4e:c6:2e:ce:08:36:d1:ea:a0:97:3d:76:38:
            6e:9d:4b:6f:33:d2:fa:c2:7e:b0:59:bc:75:97:17:d1:1b:c5:
            c4:58:ae:7b:7e:87:e5:87:2b:8b:6b:10:16:70:7c:c8:65:c7:
            d0:62:5d:f3:b5:06:af:03:8b:32:dd:88:f0:07:2b:5d:61:58:
            61:35:54:a6:ce:95:81:a2:6e:fa:b5:aa:25:e1:41:53:9d:e7:
            4b:7e:93:88:79:6b:dd:a3:6e:9a:0d:bd:85:b4:2d:66:b9:cc:
            01:13:f1:b5:d5:91:cc:86:5e:a7:c8:4a:8f:4d:9d:f8:17:31:
            32:7d:50:d5:c2:79:a0:41:a0:69:83:33:16:14:35:26:10:3b:
            23:eb:60:d9:28:68:99:d5:55:61:89:b5:35:5d:8b:fe:b1:96:
            32:69:3e:8b:c2:a2:4e:e1:d8:76:04:3c:87:91:5d:66:9e:81:
            a5:bf:18:2e:3e:39:da:4f:68:57:46:d2:1d:aa:81:51:3b:33:
            72:da:e9:7d:12:b6:a1:fc:c7:1d:c1:9c:bd:92:e8:1b:d2:06:
            e8:0b:82:2a:4f:23:5a:7a:fa:7b:86:a0:d7:c1:46:e7:04:47:
            77:11:cd:da:7c:50:32:d2:6f:fd:1e:0a:df:cf:b1:20:d2:86:
            ce:40:5a:27:61:49:2f:71:f5:04:ac:eb:c6:03:70:a4:70:13:
            4a:af:41:35:83:dc:55:c0:29:7f:12:4f:d0:f1:bb:f7:61:4a:
            9f:8d:61:b0:5e:89:46:49:e3:27:8b:42:82:5e:af:14:d5:d9:
            91:69:3d:af:11:70:5b:a3:92:3b:e3:c8:2a:a4:38:e5:88:f2:
            6f:09:f4:e5:04:3b
-----BEGIN CERTIFICATE-----
MIIELTCCApWgAwIBAgIEPJEpsjANBgkqhkiG9w0BAQsFADAmMSQwIgYDVQQDDBto
aWdod2F5LXRlc3QuZXhhbXBsZS5jb20gQ0EwHhcNMjEwNDA1MTkzNjU3WhcNMjEw
NTA2MDUzNjU3WjAmMSQwIgYDVQQDDBtoaWdod2F5LXRlc3QuZXhhbXBsZS5jb20g
Q0EwggGiMA0GCSqGSIb3DQEBAQUAA4IBjwAwggGKAoIBgQC0eydCSZ/thUd0//ZQ
zV0iGmQ4IvgJ0tbzYNiYf+WEUh7Zzpa03KZDdGcn2Z1Cfb8aQ5Kb0d00m0HS49VZ
s0D8s8nhWIQ/h/cGRSUmTL+hRXKgCluGQdeOvtM4tapmab06/em1uKJ5xPClPJ6R
lDIenLB/JUZbdh2GI4WwYkVcqG/7xSbh3ajyaKvFjLRYtC6WSfr+0uqlEWjCjfRY
qzC93RsplwAYb1lAnDoq5JYluxL0GhFybTH2tOHM2JoMqqiqpGTj8QYcwAnfYroE
y3CwxPfKNSLqqcdS4c4n+2xSObcis12XywqfdaOvFu/mshtqwwsdFf242OeK9vSZ
HCOXS4DpeaOFFvjdvXfvOjyO53VWZzY63UJ7hC9kLxMO+rA7ERN+rnimL0bdSxGI
5HsZqyEtHzS6Yc1RhKXsasGQIHDjqvQB/QxuzQRHmTFweWyvQXjBBCpDeISK/sM9
8kHIKqEQ4Le0T07mJnmsSWTPVx4u4y9YvW8wAGfXi9YTYL8CAwEAAaNjMGEwDwYD
VR0TAQH/BAUwAwEB/zAOBgNVHQ8BAf8EBAMCAQYwHQYDVR0OBBYEFDMSRbcbEL7z
y2TlTFCAfJ2IZXRAMB8GA1UdIwQYMBaAFDMSRbcbEL7zy2TlTFCAfJ2IZXRAMA0G
CSqGSIb3DQEBCwUAA4IBgQAFNyiFNzlxh+xc8FEZVUq34CrmYTDU4iutetsS/Iqm
bhWCgBD6XWdg6FQU44nWTmCJmFur/jImqgI1aE7GLs4INtHqoJc9djhunUtvM9L6
wn6wWbx1lxfRG8XEWK57foflhyuLaxAWcHzIZcfQYl3ztQavA4sy3YjwBytdYVhh
NVSmzpWBom76taol4UFTnedLfpOIeWvdo26aDb2FtC1mucwBE/G11ZHMhl6nyEqP
TZ34FzEyfVDVwnmgQaBpgzMWFDUmEDsj62DZKGiZ1VVhibU1XYv+sZYyaT6LwqJO
4dh2BDyHkV1mnoGlvxguPjnaT2hXRtIdqoFROzNy2ul9Erah/McdwZy9kugb0gbo
C4IqTyNaevp7hqDXwUbnBEd3Ec3afFAy0m/9Hgrfz7Eg0obOQFonYUkvcfUErOvG
A3CkcBNKr0E1g9xVwCl/Ek/Q8bv3YUqfjWGwXolGSeMni0KCXq8U1dmRaT2vEXBb
o5I748gqpDjliPJvCfTlBDs=
-----END CERTIFICATE-----
```

The private key for the Certification Authority that created the IDevID:


```
-----BEGIN RSA PRIVATE KEY-----
MIIG5AIBAAKCAYEAtHsnQkmf7YVHdP/2UM1dIhpkOCL4CdLW82DYmH/lhFIe2c6W
tNymQ3RnJ9mdQn2/GkOSm9HdNJtB0uPVWbNA/LPJ4ViEP4f3BkUlJky/oUVyoApb
hkHXjr7TOLWqZmm9Ov3ptbiiecTwpTyekZQyHpywfyVGW3YdhiOFsGJFXKhv+8Um
4d2o8mirxYy0WLQulkn6/tLqpRFowo30WKswvd0bKZcAGG9ZQJw6KuSWJbsS9BoR
cm0x9rThzNiaDKqoqqRk4/EGHMAJ32K6BMtwsMT3yjUi6qnHUuHOJ/tsUjm3IrNd
l8sKn3Wjrxbv5rIbasMLHRX9uNjnivb0mRwjl0uA6XmjhRb43b137zo8jud1Vmc2
Ot1Ce4QvZC8TDvqwOxETfq54pi9G3UsRiOR7GashLR80umHNUYSl7GrBkCBw46r0
Af0Mbs0ER5kxcHlsr0F4wQQqQ3iEiv7DPfJByCqhEOC3tE9O5iZ5rElkz1ceLuMv
WL1vMABn14vWE2C/AgMBAAECggGAAUF6HHP2sOhkfuPpCtbi9wHIALv9jdPxuu/J
kgYRysHnhQxy7/85CO8eaKCS/4twcPZXZs4nA96wro73RRCCOz/k/7Rl9yszBNAm
WgXer3iUO5jW2jBLF6ssPRDGhr/lmSt7HNCUENTV99BcKhcl4iCk+b2Ap9JCklRc
8cU9Rk/Ft7K/eoLYUhd4Wn+IIbXfPRx2qp89Erj0SaZDNPq79BY9wiRS09iyfkiX
/wRoJwsOLrSfunQYDOdlSs+XAs+NKeKmB6chmPhP+sYTXx+zFj+36NRjq2dxkYSH
hB9peJ5yzTDhLQpagV5D36VXQsqHawvgEu6cQAfcZ4Iqmnura7zYBysfk4YzzizO
rsc9rYGP10UO5W0EpKR/IcNfMGwtDbHe1/7z+0JSVDe/ldht8YrwX3ogd5rNbhlf
lUE+D7rof8E8g6Uz4TWI8dpMDaXCzjgz6q2iiW770R5xCphLFbuNh/SnbkYNYNEo
k8AN+Fx+w3EO7Cg4aaETB76iNXVBAoHBAOibavF4IYurjni39Z/6vIhO31F7VdNj
x9gZ9Om6MmZNFSbU8PLyoQEyI46ygf8TO/BSfiHyUMncohmXWsoUXiFZV412aVqk
HgZg+MWsKuYuTmGk/CouYQzd7RtrLl8TpPncXhsJIZ48ppcVGnMHnWZmTLj/Kqf6
oDfsI7QhZy8fUxgIJ3vWoC5zFeQYzXpID4PKkn6mXczt6YiQHFJuvqVjpflVh9WZ
leIhCBxoI76j1uU3ZiOEWfkmxSWddIPyIwKBwQDGobnHJ1lIJeny/KaHBVt8OECV
wEH6lAxp4jcxYgQCbPVGJzNs+BstjOiY+UDrG2MVyJ+dj+yS2lfDBJcyzo/mE/ox
0odGpKJ9MVk4Mb4m543Jllgb9ZQmJmKzJipqpRetmXV22QB0sJyaYL4M3zroqw17
tEf6HH1vmc9XQwACJOrlm+k41djutwmuCE2JYoNbLdcrCgdfO06Z3bhNkknbrrFD
OrB40xx1H5u38kDU7ifieQ4jvUEWk6a5+sIR+rUCgcEAyp+AJEJyblmObShKhgaE
LvUN4cvfcppL3rqVtvhkqOrizwXVsryadhE4GjjztsAJiYpCp82OhJl2d3Z6NuhR
KxnJg8gvdC7cnM/iRUd5wzN5QePXaeMm1W+I+UZ/iYDySFmnfEOTDmVk9N0EQknS
2f2pPcnBXbybzrscSvCCEvFlj9yikGTg+jV0T1MvwyJ8qWBQBpVjxn1E3poyobgo
yKeqUC0qe24ju2zsxNoOsSXFr7x3c976BWi5ec/UTJAjAoHAYZ+GwRzTwqPvsZ7+
8Yluh0TWaUNOqistVrT5z2mO8uo+OjZ2De563Q5OGzEV+PdC4afy2uurqBlr3Mta
zHu9OaVD6EzCc7PisIkagoXgIRrZEuSzdTpjj8R56fauDjAJzSaJFtpcYP2UWkOF
5KmqOEQpokzeu0xZUgpUX1zsmiEu2Z6hJ2/i6KBJP6GRCh7C1INZJywMp39siC7y
sB1f83qOYK5toVSQvffE/skvl/dc3vAERQh0/vWekfVugIupAoHBAJj9U/aFU/c5
Kc/94hmeR6TljINMSn0EI9nlJ5FkY2BDmzgeAD9/kNBbPHRjIyMa5Ow7rHO4Lt09
U837yytEcbmErNzMuBhOX+nirXXq1Dp5LMNkHP3gnPy0XC2Cu5m2vH/qbFhIlRER
1GXCxBrWOzovXFu090oIjOhwCbxt7GWZH/GMUUJGXJb+s1CzQNz1qiXKng7XpluA
S9jVch5pKqmWvDYYrBXmmCe9Ju0RnBCgOIuGUiCPjEFAy+myLdgQ0A==
-----END RSA PRIVATE KEY-----
```


The MASA certificate that signs the voucher:

```
-----BEGIN CERTIFICATE-----
MIIBcDCB9qADAgECAgQLhwoxMAoGCCqGSM49BAMCMCYxJDAiBgNVBAMMG2hpZ2h3
YXktdGVzdC5leGFtcGxlLmNvbSBDQTAeFw0yMTA0MTMyMTQwMTZaFw0yMzA0MTMy
MTQwMTZaMCgxJjAkBgNVBAMMHWhpZ2h3YXktdGVzdC5leGFtcGxlLmNvbSBNQVNB
MFkwEwYHKoZIzj0CAQYIKoZIzj0DAQcDQgAEqgQVo0S54kT4yfkbBxumdHOcHrps
qbOpMKmiMln3oB1HAW25MJV+gqi4tMFfSJ0iEwt8kszfWXK4rLgJS2mnpaMQMA4w
DAYDVR0TAQH/BAIwADAKBggqhkjOPQQDAgNpADBmAjEArsthLdRcjW6GqgsGHcbT
YLoyczYl0yOFSYcczpQjeRqeQVUkHRUioUi7CsCrPBNzAjEAhjxns5Wi4uX5rfkd
nME0Mnj1z+rVRwOfAL/QWctRwpgEgSSKURNQsXWyL52otPS5
-----END CERTIFICATE-----
```


The private key for MASA certificate signs the voucher:


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


A.2. Example CMS signed voucher request

```
MIIGjQYJKoZIhvcNAQcCoIIGfjCCBnoCAQExDTALBglghkgBZQMEAgEwggOl
BgkqhkiG9w0BBwGgggOWBIIDknsiaWV0Zi12b3VjaGVyLXJlcXVlc3Q6dm91
Y2hlciI6eyJhc3NlcnRpb24iOiJwcm94aW1pdHkiLCJjcmVhdGVkLW9uIjoi
MjAyMi0wNy0xMFQxNzowODoxOC41OTgtMDQ6MDAiLCJzZXJpYWwtbnVtYmVy
IjoiMDAtRDAtRTUtRjItMDAtMDIiLCJub25jZSI6IjR2VHNwcFMyQ2VxQnpo
RWRvaWZNMmciLCJwcm94aW1pdHktcmVnaXN0cmFyLWNlcnQiOiJNSUlDRURD
Q0FaYWdBd0lCQWdJRVlGYTZaVEFLQmdncWhrak9QUVFEQWpCdE1SSXdFQVlL
Q1pJbWlaUHlMR1FCR1JZQ1kyRXhHVEFYQmdvSmtpYUprL0lzWkFFWkZnbHpZ
VzVrWld4dFlXNHhQREE2QmdOVkJBTU1NMlp2ZFc1MFlXbHVMWFJsYzNRdVpY
aGhiWEJzWlM1amIyMGdWVzV6ZEhKMWJtY2dSbTkxYm5SaGFXNGdVbTl2ZENC
RFFUQWVGdzB5TVRFeE1qUXhPVFF6TURWYUZ3MHlNekV4TWpReE9UUXpNRFZh
TUZNeEVqQVFCZ29Ka2lhSmsvSXNaQUVaRmdKallURVpNQmNHQ2dtU0pvbVQ4
aXhrQVJrV0NYTmhibVJsYkcxaGJqRWlNQ0FHQTFVRUF3d1pabTkxYm5SaGFX
NHRkR1Z6ZEM1bGVHRnRjR3hsTG1OdmJUQlpNQk1HQnlxR1NNNDlBZ0VHQ0Nx
R1NNNDlBd0VIQTBJQUJKWmxVSEkwdXAvbDNlWmY5dkNCYitsSW5vRU1FZ2M3
Um8rWFpDdGpBSTBDRDFmSmZKUi9oSXl5RG1IV3lZaU5GYlJDSDlmeWFyZmt6
Z1g0cDB6VGl6cWpQakE4TUNvR0ExVWRKUUVCL3dRZ01CNEdDQ3NHQVFVRkJ3
TWNCZ2dyQmdFRkJRY0RBZ1lJS3dZQkJRVUhBd0V3RGdZRFZSMFBBUUgvQkFR
REFnZUFNQW9HQ0NxR1NNNDlCQU1DQTJnQU1HVUNNUUNkU1pSSjgzTU5SQ3ph
Myt2T0JhMDFoNHFadjJsS2hkK0RmaEI0WURodkdwa1dvbFplSEh3TmI3QXRC
Q010YlV3Q01Ib054b2lrK3hXN0F0MWhYRWhwMy9NY1hpQWR6blpicFZxK3hK
RVppaFhVMzZJQmp2WWdXREY5aXZxeEpwRGJ5dz09In19oIIBszCCAa8wggE1
oAMCAQICBB8Y/ucwCgYIKoZIzj0EAwIwJjEkMCIGA1UEAwwbaGlnaHdheS10
ZXN0LmV4YW1wbGUuY29tIENBMCAXDTIxMDQyNzE4MjkzMFoYDzI5OTkxMjMx
MDAwMDAwWjAcMRowGAYDVQQFExEwMC1EMC1FNS1GMi0wMC0wMjBZMBMGByqG
SM49AgEGCCqGSM49AwEHA0IABAOjdUOHs3zACpqHnK329DgTHNQMhB/gQE5B
efBbE0sgcbNEm0li8RYwzrsAfYCxp9k7E1CbpielT8OWf0z+ISejWTBXMB0G
A1UdDgQWBBRFiMyWlgBkN7C6I2VkZFQIBmxWrTAJBgNVHRMEAjAAMCsGCCsG
AQUFBwEgBB8WHWhpZ2h3YXktdGVzdC5leGFtcGxlLmNvbTo5NDQzMAoGCCqG
SM49BAMCA2gAMGUCMGIq27409xvLhd4mjkMA+Q2IyHeo3TwIQFS87D223HAr
w3/KGSGaoKvFUY6q3zbeiwIxALJdWfhHx+0Dl6jAx6iB+qiG7WdkN1F6bpyj
gk1trbzzNZ6daqJtf38lHAPv8LqbcTGCAQQwggEAAgEBMC4wJjEkMCIGA1UE
AwwbaGlnaHdheS10ZXN0LmV4YW1wbGUuY29tIENBAgQfGP7nMAsGCWCGSAFl
AwQCAaBpMBgGCSqGSIb3DQEJAzELBgkqhkiG9w0BBwEwHAYJKoZIhvcNAQkF
MQ8XDTIyMDcxMDIxMDgxOFowLwYJKoZIhvcNAQkEMSIEIFc4jO6OnilTLkM/
fcc9p5au4ANjvJvjRXsAKK6+RcTvMAoGCCqGSM49BAMCBEcwRQIhAOjoOdgh
Sr+Hk2r2APsfs1+QJba0uRf/+zXA70yb6mRCAiB9aS6Wj8kBcWEvvfsDue41
KWo0ukOBQxdPGpJqg+GAMw==
```


A.3. Example CMS signed voucher from MASA

```
MIIGPQYJKoZIhvcNAQcCoIIGLjCCBioCAQExDTALBglghkgBZQMEAgEwggOU
BgkqhkiG9w0BBwGgggOFBIIDgXsiaWV0Zi12b3VjaGVyOnZvdWNoZXIiOnsi
YXNzZXJ0aW9uIjoibG9nZ2VkIiwiY3JlYXRlZC1vbiI6IjIwMjItMDctMTBU
MTc6MDg6MTguNzIwLTA0OjAwIiwic2VyaWFsLW51bWJlciI6IjAwLUQwLUU1
LUYyLTAwLTAyIiwibm9uY2UiOiI0dlRzcHBTMkNlcUJ6aEVkb2lmTTJnIiwi
cGlubmVkLWRvbWFpbi1jZXJ0IjoiTUlJQ0VEQ0NBWmFnQXdJQkFnSUVZRmE2
WlRBS0JnZ3Foa2pPUFFRREFqQnRNUkl3RUFZS0NaSW1pWlB5TEdRQkdSWUNZ
MkV4R1RBWEJnb0praWFKay9Jc1pBRVpGZ2x6WVc1a1pXeHRZVzR4UERBNkJn
TlZCQU1NTTJadmRXNTBZV2x1TFhSbGMzUXVaWGhoYlhCc1pTNWpiMjBnVlc1
emRISjFibWNnUm05MWJuUmhhVzRnVW05dmRDQkRRVEFlRncweU1URXhNalF4
T1RRek1EVmFGdzB5TXpFeE1qUXhPVFF6TURWYU1GTXhFakFRQmdvSmtpYUpr
L0lzWkFFWkZnSmpZVEVaTUJjR0NnbVNKb21UOGl4a0FSa1dDWE5oYm1SbGJGJH
MWhiakVpTUNBR0ExVUVBd3daWm05MWJuUmhhVzR0ZEdWemRDNWxlR0Z0Y0d4
bExtTnZiVEJaTUJNR0J5cUdTTTQ5QWdFR0NDcUdTTTQ5QXdFSEEwSUFCSlps
VUhJMHVwL2wzZVpmOXZDQmIrbElub0VNRWdjN1JvK1haQ3RqQUkwQ0QxZkpm
SlIvaEl5eURtSFd5WWlORmJSQ0g5ZnlhcmZremdYNHAwelRpenFqUGpBOE1D
b0dBMVVkSlFFQi93UWdNQjRHQ0NzR0FRVUZCd01jQmdnckJnRUZCUWNEQWdZ
SUt3WUJCUVVIQXdFd0RnWURWUjBQQVFIL0JBUURBZ2VBTUFvR0NDcUdTTTQ5
QkFNQ0EyZ0FNR1VDTVFDZFNaUko4M01OUkN6YTMrdk9CYTAxaDRxWnYybEto
ZCtEZmhCNFlEaHZHcGtXb2xaZUhId05iN0F0QkNNdGJVd0NNSG9eG9payt4
VzdBdDFoWEVocDMvTWNYaUFkem5aYnBBWcSt4SkVaaWhYVTM2SUJqdllnV0RG
OWl2cXhhKcERieXc9PSJ9faCCAXQwggFwMIH2oAMCAQICBAuHCjEwCgYIKoZI
zj0EAwIwJjEkMCIGA1UEAwwbaGlnaHdheS10ZXN0LmV4YW1wbGUuY29tIENB
MB4XDTIxMDQxMzIxNDAxNloXDTIzMDQxMzIxNDAxNlowKDEmMCQGA1UEAwwd
aGlnaHdheS10ZXN0LmV4YW1wbGUuY29tIE1BU0EwWTATBgcqhkjOPQIBBggq
hkjOPQMBBwNCAASqBBWjRLniRPjJ+RsHG6Z0c5weumyps6kwqaIyWfegHUcB
bbkwlX6CqLi0wV9InSITC3ySzN9ZcrisuAlLaaeloxAwDjAMBgNVHRMBAf8E
AjAAMAoGCCqGSM49BAMCA2kAMGYCMQCuy2Et1FyNboaqCwYdxtNgujJzNiXT
I4VJhxzOlCN5Gp5BVSQdFSKhSLsKwKs8E3MCMQCGPGezlaLi5fmt+R2cwTQy
ePXP6tVHA58Av9BZy1HCmASBJIpRE1CxdbIvnai09LkxggEEMIIBAAIBATAu
MCYxJDAiBgNVBAMMG2hpZ2h3YXktdGVzdC5leGFtcGxlLmNvbSBDQQIEC4cK
MTALBglghkgBZQMEAgGgaTAYBgkqhkiG9w0BCQMxCwYJKoZIhvcNAQcBMBwG
CSqGSIb3DQEJBTEPFw0yMjA3MTAyMTA4MThaMC8GCSqGSIb3DQEJBDEiBCBA
77EhoAybh5R6kK89jDefpxRy8Q6rDo1cnlwgvCzXbzAKBggqhkjOPQQDAgRH
MEUCIQD4RnuXwKvYVvwamwVq3VYv7dXcM7bzLg7FXTkhvYqPzwIgXTJxVV5a
cLMAroeHgThS5JU5QA2PJMLGF82UcSNTsEY=
```


Acknowledgements

The authors would like to thank for following for lively discussions on list and in the halls (ordered by last name): William Atwood, Michael H. Behringer, Esko Dijk, Steffen Fries, Sheng Jiang, Thomas Werner.

Authors' Addresses

Kent Watsen
Watsen Networks

Email: kent+ietf@watsen.net

Michael C. Richardson
Sandelman Software
Email: mcr+ietf@sandelman.ca
URI: http://www.sandelman.ca/

Max Pritikin
Cisco Systems
Email: pritikin@cisco.com

Toerless Eckert
Futurewei Technologies Inc.
2330 Central Expy
Santa Clara, 95050
United States of America
Email: tte@cs.fau.de

Qiufang Ma
Huawei
101 Software Avenue, Yuhua District
Nanjing
210012
China
Email: maqiufang1@huawei.com