ANMA Working Group
Internet-Draft
Intended status: Standards Track
Expires: June 10, 2017

K. Watsen
Juniper Networks
M. Richardson
SSW
M. Pritikin
T. Eckert
Cisco Systems

December 7, 2016

Voucher and Voucher Revocation Profiles for Bootstrapping Protocols
draft-kwatsen-anima-voucher-00

Abstract

This memo defines the two artifacts "voucher" and "voucher-revocation", which are YANG-defined structures that have been signed by a TBD algorithm.

The voucher artifact is generated by the device's manufacture or delegate. The voucher's purpose is to securely assign one or more devices to an owner. The voucher informs each device which entity it should consider to be its owner.

The voucher revocation artifact is used by the manufacturer or delegate (i.e. the issuer of the voucher) to revoke vouchers, if ever necessary. The voucher revocation format defined herein supports both issuer-wide and voucher-specific constructs, enabling usage flexibility.

For both artifacts, this memo only defines the artifact, leaving it to future work to describe specialized protocols for accessing them.

Status of This Memo

This Internet-Draft will expire on June 10, 2017.

Copyright Notice

Table of Contents

1. Introduction

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

A voucher may be useful in several contexts, but the driving motivation herein is to support secure bootstrapping mechanisms, such as are defined in [draft-ietf-netconf-zerotouch] and [draft-ietf-anima-bootstrapping-keyinfra]. Assigning ownership is important to bootstrapping mechanisms so that the booting device can authenticate the network that's trying to take control of it.

The lifetimes of vouchers may vary. In some bootstrapping protocols the vouchers may be ephemeral, whereas in others the vouchers may be potentially long-lived. In order to support the second category of vouchers, this document also defines a voucher revocation artifact, enabling the manufacturer or delegate to communicate the validity of its vouchers.

For both artifacts, this memo only defines the artifact, leaving it to future work to describe specialized protocols for accessing them.

This document uses YANG [RFC7950] to define the voucher and voucher revocation formats. YANG is a data modeling language with established mappings to XML and JSON, with mappings to other encodings in progress. Which encodings a particular solution uses is outside the scope of this document.

2. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in the sections below are to be interpreted as described in RFC 2119 [RFC2119].

3. Tree Diagram Notation

The meaning of the symbols in the above diagram is as follows:

- Brackets "[" and "]" enclose list keys.

- Braces "{" and "}" enclose feature names, and indicate that the named feature must be present for the subtree to be present.

- Abbreviations before data node names: "rw" (read-write) represents configuration data and "ro" (read-only) represents state data.

- Symbols after data node names: "?" means an optional node, "!" means a presence container, and "*" denotes a list and leaf-list.

- Parentheses enclose choice and case nodes, and case nodes are also marked with a colon (":").

- o Ellipsis ("...") stands for contents of subtrees that are not shown.

4. Voucher

The voucher is generated by the device's manufacture or delegate. The voucher's purpose is to securely assign one or more devices to an owner. The voucher informs each device which entity it should consider to be its owner.

The voucher is signed by the device's manufacturer or delegate. NOTE: AT THIS TIME, THE SIGNING STRATEGY HAS NOT BEEN SELECTED.

4.1. Tree Diagram

Following is the tree diagram for the YANG module specified in Section 4.3. Details regarding each node in the tree diagram are provided in the YANG module. Please see Section 3 for information on tree diagram notation.

```
module: ietf-voucher
    +--ro voucher
       +--ro assertion                 enumeration
       +--ro trusted-ca-certificate?   binary
       +--ro certificate-id
       |  +--ro cn-id?    string
       |  +--ro dns-id?   string
       +--ro unique-id*                string
       +--ro nonce?                    string
       +--ro created-on?               yang:date-and-time
       +--ro expires-on?               yang:date-and-time
       +--ro revocation-location?      inet:uri
       +--ro additional-data?
```

4.2. Examples

The following illustrates an ephemeral voucher encoded in JSON:

```
{
  "ietf-voucher:voucher": {
    "assertion": "logged",
    "trusted-ca-certificate": "base64-encoded X.509 DER",
    "owner-id": "Registrar3245",
    "unique-id": "JADA123456789",
    "created-on": "2016-10-07T19:31:42Z",
    "nonce": "987987623489567"
  }
}
```

The following illustrates a long-lived voucher encoded in XML:

```
<voucher
   xmlns="urn:ietf:params:xml:ns:yang:ietf-voucher">
  <assertion>verified</assertion>
  <trusted-ca-certificate>
    base64-encoded X.509 DER
  </trusted-ca-certificate>
  <certificate-id>
    <cn-id>Example Inc.</cn-id>  <!-- maybe this should be a DN? -->
    <dns-id>example.com</dns-id>
  </certificate-id>
  <unique-id>AAA123456789</unique-id>
  <unique-id>BBB123456789</unique-id>
  <unique-id>CCC123456789</unique-id>
  <created-on>2016-10-07T19:31:42Z</created-on>
</voucher>
```

4.3. YANG Module

```
<CODE BEGINS> file "ietf-voucher@2016-12-07.yang"

module ietf-voucher {
  yang-version 1.1;

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

  import ietf-yang-types { prefix yang; }
  import ietf-inet-types { prefix inet; }

  organization
   "IETF ANIMA Working Group";

  contact
   "WG Web:   <http://tools.ietf.org/wg/anima/>
    WG List:  <mailto:anima@ietf.org>
    Author:   Kent Watsen
              <mailto:kwatsen@juniper.net>
    Author:   Max Pritikin
              <mailto:pritikin@cisco.com>
    Author:   Michael Richardson
              <mailto:mcr+ietf@sandelman.ca>";

  description
   "This module defines the format for a voucher, which is
    produced by a device's manufacturer or delegate to securely
```

```
    assign one or more devices to an 'owner', so that the
    devices may establish a secure connection to the owner's
    network infrastructure.";

  revision "2016-12-07" {
    description
     "Initial version";
    reference
     "RFC XXXX: Voucher and Voucher Revocation Profiles
      for Bootstrapping Protocols";
  }

  // top-level container
  container voucher {
    config false;
    description
      "A voucher that can be used to assign one or more devices to
       an owner.";

    leaf assertion {
      type enumeration {
        enum verified {
          description
            "Indicates that the ownership has been positively
             verified by the device's manufacturer or delegate
             (e.g., through sales channel integration).";
        }
        enum logged {
          description
            "Indicates that this ownership assignment has been
             logged into a database maintained by the device's
             manufacturer or delegate (voucher transparency).";
        }
      }
      mandatory true;
      description
        "The assertion is a statement from the manufacturer or
         delegate regarding the nature of this voucher.  This
         allows the device to know what assurance the manufacturer
         provides, which supports more detailed policy checks
         such as 'I only want to allow verified devices, not
         just logged devices'.";
    }

    leaf trusted-ca-certificate {
      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.

         This certificate is used by a bootstrapping device to
         trust another public key infrastructure, in order to
         verify another certificate supplied to the device
         separately by the bootstrapping protocol, the other
         certificate must have this certificate somewhere in
         its chain of certificates.";

      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).";
    }

    container certificate-id {
      description
        "When provided, the device MUST also perform RFC 6125
         style validation of another certificate supplied to
         the device separately by the bootstrapping protocol
         against all the provided ids.";
      leaf cn-id {
        type string;
        description
          "The common name field in the cetificate must match
           this value.";
      }
      leaf dns-id {
        type string;
        description
          "A subjectAltName entry of type dNSName in the
           certificate must match this value.";
      }
    }

    leaf-list unique-id {
      type string;
      min-elements 1;
      description
        "A regular expression identifying one more more device
         unique identifiers (e.g., serial numbers).  For instance,
         the expression could match just a single serial number,
```

```
         or it might match a range of serial numbers.  Devices
         use this value to determine if the voucher applies to
         them.";

         // Ed. both the zerotouch and brwski solutions are devid
         // oriented, and so renaming this field to 'serial-number'
         // wouldn't be crazy.  But devid/serial-number (typically)
         // assumes physical chassis, is it worth using this
         // term which might extend to e.g. virtual appliances?
    }

    leaf nonce {
      type string;  // unit64?
      description
        "what can be said about this that's ANIMA-neutral?";
    }

    leaf created-on {
      type yang:date-and-time;
      description
        "The date this voucher was created";
    }

    leaf expires-on {
      type yang:date-and-time;
      description
        "An optional date value for when this voucher expires.";
    }

    leaf revocation-location {
      type inet:uri;
      description
        "A URI indicating where revocation information may be
         obtained.";
    }

    anydata additional-data {
      description
        "Additional data signed by the manufacturer.  The manufacturer
         might put additional data into its vouchers, for human or
         device consumption.";

         // Ed. is the additional data normative? - if so, should we
         // remove this free-form field, and assume it will be formally
         // extended later?  Note: the zerotouch draft doesn't need this
         // field...
    }
  }
```

```
}

<CODE ENDS>
```

5. Voucher Revocation

The vouchers revocation artifact is used to verify the revocation status of vouchers. Voucher revocations are signed by the manufacturer or delegate (i.e. the issuer of the voucher). Vouchers revocation statements MAY be verified by devices during the bootstrapping process, or at any time before or after by any entity (e.g., registrar or equivalent) as needed. Registrars or equivalent SHOULD verify voucher revocation statements and make policy decisions in case devices are not doing so themselves.

Revocations are generally needed when it is critical for devices to know that assurances implied at the time the voucher was signed are still valid at the time the voucher is being processed.

As mentioned in Section 1, the lifetimes of vouchers may vary. In some bootstrapping protocols the vouchers may be ephemeral, whereas in others the vouchers may be potentially long-lived. For bootstrapping protocols that support ephemeral vouchers, there is no need to support revocations. For bootstrapping protocols that support long-lived vouchers, the need to support revoking vouchers is a decision for each manufacturer.

If revocations are not supported then voucher assignments are essentially forever, which may be acceptable for various kinds of devices. If revocations are supported, then it becomes possible to support various scenarios such as handling a key compromise or change in ownership.

The voucher revocation format defined herein supports both issuer-wide (similar to a CRL) or voucher-specific (similar to an OCSP response) constructs, enabling usage flexibility.

NOTE: AT THIS TIME, THE SIGNING STRATEGY HAS NOT BEEN SELECTED.

5.1. Tree Diagram

Following is the tree diagram for the YANG module specified in Section 5.3. Details regarding each node in the tree diagram are provided in the YANG module. Please see Section 3 for information on tree diagram notation.

```
   module: ietf-voucher-revocation
       +--ro voucher-revocation
          +--ro revocation-type     enumeration
          +--ro created-on          yang:date-and-time
          +--ro expires-on?         yang:date-and-time
          +--ro (voucher-revocation-type)?
          |  +--:(issuer-wide)
          |  |  +--ro issuer-wide
          |  |     +--ro (list-type)?
          |  |        +--:(whitelist)
          |  |        |  +--ro whitelist
          |  |        |     +--ro voucher-identifier*   string
          |  |        +--:(blacklist)
          |  |           +--ro blacklist
          |  |              +--ro voucher-identifier*   string
          |  +--:(voucher-specific)
          |     +--ro voucher-specific
          |        +--ro voucher-identifier        string
          |        +--ro voucher-status            enumeration
          |        +--ro revocation-information
          |           +--ro revoked-on           yang:date-and-time
          |           +--ro revocation-reason    enumeration
          +--ro additional-data?
```

5.2. Examples

The following illustrates an issuer-wide voucher revocation in XML:

```
   <voucher-revocation
      xmlns="urn:ietf:params:xml:ns:yang:ietf-voucher-revocation">
     <revocation-type>issuer-wide</revocation-type>
     <created-on>2016-10-31T23:59:59Z</created-on>
     <expires-on>2016-12-31T23:59:59Z</expires-on>
     <issuer-wide>
       <blacklist>
         <voucher-identifier>some fingerprint</voucher-identifier>
         <voucher-identifier>some fingerprint</voucher-identifier>
         <voucher-identifier>some fingerprint</voucher-identifier>
       </blacklist>
     </issuer-wide>
   </voucher>
```

The following illustrates a voucher-specific revocation in JSON:

```
   {
     "ietf-voucher-revocation:voucher-revocation": {
       "revocation-type": "voucher-specific",
       "created-on": "2016-10-31T23:59:59Z"
       "expires-on": "2016-12-31T23:59:59Z"
       "voucher-specific": [
         "voucher-identifier": "some fingerprint",
         "voucher-status": "revoked",
         "revocation-information": [
           "revoked-on": "2016-11-31T23:59:59Z",
           "revocation-reason": "key-compromise"
         ]
       ]
     }
   }
```

5.3. YANG Module

<CODE BEGINS> file "ietf-voucher-revocation@2016-12-07.yang"

```
module ietf-voucher-revocation {
  yang-version 1.1;

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

  import ietf-yang-types { prefix yang; }

  organization
   "IETF ANIMA Working Group";

  contact
   "WG Web:   <http://tools.ietf.org/wg/anima/>
    WG List:  <mailto:anima@ietf.org>
    Author:   Kent Watsen
              <mailto:kwatsen@juniper.net>
    Author:   Max Pritikin
              <mailto:pritikin@cisco.com>
    Author:   Michael Richardson
              <mailto:mcr+ietf@sandelman.ca>";

  description
   "This module defines the format for a voucher revocation,
    which is produced by a manufacturer or delegate to indicate
    the revocation status of vouchers.";

  revision "2016-12-07" {
```

```
    description
     "Initial version";
    reference
     "RFC XXXX: Voucher and Voucher Revocation Profiles
      for Bootstrapping Protocols";
  }

  // top-level container
  container voucher-revocation {
    config false;
    description
      "A voucher revocation that can provide revocation status
       information for one or more devices.";

    leaf revocation-type {
      type enumeration {
        enum issuer-wide {
          description
            "Indicates that this revocation spans all
             the vouchers the issuer has issued to date";
        }
        enum voucher-specific {
          description
            "Indicated that this revocation only regards
             a single voucher.";
        }
      }
      mandatory true;
      description
        "The revocation-type indicates if the revocation
         is issuer-wide or voucher-specific.  Both variations
         exist to enable implementations to choose between the
         number of revocation artifacts generated versus
         individual artifact size.";
    }

    leaf created-on {
      type yang:date-and-time;
      mandatory true;
      description
        "The date this voucher was created";
    }

    leaf expires-on {
      type yang:date-and-time;
      description
        "An optional date value for when this voucher expires.";
    }
```

```
    choice voucher-revocation-type {
      description
        "Identifies the revocation type as being either issuer-wide
         or voucher-specific.";

      container issuer-wide {
        description
          "This revocation provides issuer-wide revocation status
           (similar to a CRL).";

        choice list-type {
          description
            "Indentifies if this issuer-wide revocation is provided
             in the form of a whitelist or a blacklist";

          container whitelist {
            leaf-list voucher-identifier {
              type string;
              description
                "A fingerprint over the voucher artifact.";
            }
            description
              "Indicates that the listed of vouchers are known
               to be good.  If a voucher is not listed, then
               it is considered revoked.";
          }

          container blacklist {
            leaf-list voucher-identifier {
              type string;
              description
                "A fingerprint over the voucher artifact.
                 Missing if list is empty.";
            }
            description
              "Indicates that the list of vouchers have been
               revoked.  If a voucher is not listed, then it
               is considered good.";
          }

        } // end list-type

      } // end issuer-wide


      container voucher-specific {
        description
          "This revocation provides voucher-specific revocation
```

```
           status (similar to an OCSP response).";

        leaf voucher-identifier {
          type string;
          mandatory true;
          description
            "A fingerprint over the voucher artifact.";
        }

        leaf voucher-status {
          type enumeration {
            enum good {
              description
                "Indicates that this voucher is valid";
            }
            enum revoked {
              description
                "Indicates that this voucher is invalid";
            }
            enum unknown {
              description
                "Indicates that the voucher's status is unknown";
            }
          }
          mandatory true;
          description
            "Indicates if the revocation status for the specified
             voucher.";
        }

        container revocation-information {
          must "../voucher-status = 'revoked'";

          leaf revoked-on {
            type yang:date-and-time;
            mandatory true;
            description
              "The date this voucher was revoked";
          }

          leaf revocation-reason {
            type enumeration {
              enum unspecified {
                description
                  "Indicates that the reason the voucher
                   was revoked is unspecified.";
              }
              enum key-compromise {
```

```
                description
                  "Indicates that the reason the voucher
                   was revoked is because its key was
                   compromised.";
              }
              enum issuer-compromise {
                description
                  "Indicates that the reason the voucher
                   was revoked is because its issuer was
                   compromised.";
              }
              enum affiliation-changed {
                description
                  "Indicates that the reason the voucher
                   was revoked is because its affiliation
                   changed (e.g., device assigned to a
                   new owner.";
              }
              enum superseded {
                description
                  "Indicates that the reason the voucher
                   was revoked is because it has been
                   superseded (e.g., the previous voucher
                   expired.";
              }
              enum cessation-of-operation {
                description
                  "Indicates that the reason the voucher
                   was revoked is because its issuer has
                   ceased operations.";
              }
            }  // end enumeration

            mandatory true;
            description
              "modeled after 'CRLReason' in RFC 5280.";
          } // end revocation reason

          description
            "Provides details regarding why a voucher's revocation.
             Modeled after 'ResponseData' in RFC6960.";

        } // end revocation-information

      } // end voucher-specific
    }

    anydata additional-data {
```

```
      description
        "Additional data signed by the manufacturer.  The manufacturer
         might put additional data into its voucher revocations, for
         human or device consumption.";

         // Ed. is the additional data normative? - if so, should we
         // remove this free-form field, and assume it will be formally
         // extended later?  Note: the zerotouch draft doesn't need this
         // field...
    }

  }
}

<CODE ENDS>
```

6. Security Considerations

6.1. Clock Sensitivity

This document defines artifacts containing time values for voucher expirations and revocations, which require an accurate clock in order to be processed correctly. Implementations MUST ensure 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, it is RECOMMENDED that implementations disable the aspects of the solution having clock sensitivity. In particular, such implementations should assume that vouchers neither ever expire or are revokable.

It is important to note that implementations SHOULD NOT rely on NTP for time, as it is not a secure protocol.

7. IANA Considerations

7.1. The IETF XML Registry

This document registers two URIs in the IETF XML registry [RFC3688]. Following the format in [RFC3688], the following registrations are requested:

```
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.

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

7.2. 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 registrations are requested:

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

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

8. Acknowledgements

The authors would like to thank for following for lively discussions on list and in the halls (ordered by last name):

9. References

9.1. Normative References

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

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

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

9.2. Informative References

[draft-ietf-anima-bootstrapping-keyinfra]
Pritikin, M., Richardson, M., Behringer, M., and S. Bjarnason, "Bootstrapping Key Infrastructures", draft-ietf-anima-bootstrapping-keyinfra (work in progress), 2016, <https://tools.ietf.org/html/draft-ietf-anima-bootstrapping-keyinfra>.

[draft-ietf-netconf-zerotouch]
Watsen, K. and M. Abrahamsson, "Zero Touch Provisioning for NETCONF or RESTCONF based Management", draft-ietf-netconf-zerotouch (work in progress), 2016, <https://tools.ietf.org/html/draft-ietf-netconf-zerotouch>.

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

Appendix A. Change Log

Authors' Addresses

Kent Watsen
Juniper Networks

EMail: kwatsen@juniper.net

Michael C. Richardson
Sandelman Software Works

EMail: mcr+ietf@sandelman.ca
URI: http://www.sandelman.ca/

Max Pritikin
Cisco Systems

EMail: pritikin@cisco.com

Toerless Eckert
Cisco Systems

EMail: tte+anima@cs.fau.de