ANIMA WG D. von Oheimb, Ed.
Internet-Draft S. Fries
Intended status: Standards Track H. Brockhaus
Expires: 6 January 2025 Siemens
5 July 2024

BRSKI-AE: Alternative Enrollment Protocols in BRSKI
draft-ietf-anima-brski-ae-12

Abstract

This document defines an enhancement of Bootstrapping Remote Secure Key Infrastructure (BRSKI, RFC 8995). It supports alternative certificate enrollment protocols, such as CMP, that use authenticated self-contained signed objects for certification messages.

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-brski-ae/.

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

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on 6 January 2025.

Copyright Notice

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

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

Table of Contents

1. Introduction

BRSKI [RFC8995] is typically used with Enrollment over Secure Transport (EST, [RFC7030]) as the enrollment protocol for operator-specific device certificates, employing HTTP over TLS for secure message transfer. BRSKI-AE is a variant using alternative enrollment protocols with authenticated self-contained objects for the device certificate enrollment.

This approach provides the following advantages. The origin of requests and responses can be authenticated independent of message transfer. This supports end-to-end authentication (proof of origin) also over multiple hops, as well as asynchronous operation of certificate enrollment. This in turn provides architectural flexibility where and when to ultimately authenticate and authorize certification requests while retaining full-strength integrity and authenticity of certification requests.

This specification carries over the main characteristics of BRSKI, namely:

* The pledge is assumed to have received its Initial Device IDentifier (IDevID, [IEEE_802.1AR-2018]) credentials during its production. It uses them to authenticate itself to the Manufacturer Authorized Signing Authority (MASA, [RFC8995]), and to the registrar, which is the access point of the target domain, and to possibly further components of the domain where it will be operated.

* The pledge first obtains via the voucher [RFC8366] exchange a trust anchor for authenticating entities in the domain such as the domain registrar.

* The pledge then obtains its Locally significant Device IDentifier (IDevID, [IEEE_802.1AR-2018]). To this end, the pledge generates a private key, called LDevID secret, and requests via the domain registrar from the PKI of its new domain a domain-specific device certificate, called LDevID certificate. On success, it receives the LDevID certificate along with its certificate chain.

The goals of BRSKI-AE are to provide an enhancement of BRSKI for LDevID certificate enrollment using, alternatively to EST, a protocol that

* supports end-to-end authentication over multiple hops

* enables secure message exchange over any kind of transfer, including asynchronous delivery.

Note that the BRSKI voucher exchange of the pledge with the registrar and MASA uses authenticated self-contained objects, so the voucher exchange already has these properties.

The well-known URI approach of BRSKI and EST messages is extended with an additional path element indicating the enrollment protocol being used.

Based on the definition of the overall approach and specific endpoints, this specification enables the registrar to offer multiple enrollment protocols, from which pledges and their developers can then pick the most suitable one.

It may be noted that BRSKI (RFC 8995) specifies how to use HTTP over TLS, but further variants are known, such as Constrained BRSKI [I-D.ietf-anima-constrained-voucher] using CoAP over DTLS. In the sequel, 'HTTP' and 'TLS' are just references to the most common case, where variants such as using CoAP and/or DTLS are meant to be subsumed - the differences are not relevant here.

This specification is sufficient together with its references to support BRSKI with the Certificate Management Protocol (CMP, [RFC9480]) profiled in the Lightweight CMP Profile (LCMPP, [RFC9483]). Combining BRSKI with a protocol or profile other than LCMPP will require additional IANA registrations based on the rules specified in this document. It may also require additional specifications for details of the protocol or profile (similar to [RFC9483]), which are outside the scope of this document.

1.1. Supported Scenarios

BRSKI-AE is intended to be used in situations like the following.

* Pledges and/or the target domain reuse an already established certificate enrollment protocol different from EST, such as CMP.

* The application scenario indirectly excludes the use of EST for certificate enrollment, for reasons like these:

  - The Registration Authority (RA) is not co-located with the registrar while it requires end-to-end authentication of requesters. Yet EST does not support end-to-end authentication over multiple hops.

  - The RA or certification authority (CA) operator requires auditable proof of origin for Certificate Signing Requests (CSRs). This is not possible with TLS because it provides only transient source authentication.

- Certificates are requested for types of keys, such as Key Encapsulation Mechanism (KEM) keys, that do not support signing (nor alternative forms of single-shot proof of possession like those described in [RFC6955]). This is not supported by EST because it uses CSRs in PKCS #10 [RFC2986] format, which can only use proof-of-possession methods that need just a single message, while proof of possession for KEM keys, for instance, requires receiving a fresh challenge value beforehand.

- The pledge implementation uses security libraries not providing EST support or uses a TLS library that does not support providing the so-called tls-unique value [RFC5929], which is needed by EST for strong binding of the source authentication.

* No full RA functionality is available on-site in the target domain, while connectivity to an off-site RA may be intermittent or entirely offline.

* Authoritative actions of a local RA at the registrar is not sufficient for fully and reliably authorizing pledge certification requests. This may be due to missing data access or due to an insufficient level of security, for instance regarding the local storage of private keys.

Bootstrapping can be handled in various ways, depending on the application domains. The informative Appendix A provides illustrative examples from various industrial control system environments and operational setups motivating the support of alternative enrollment protocols.

2. Terminology and abbreviations

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.

This document relies on the terminology defined in [RFC8995], [RFC5280], and [IEEE_802.1AR-2018]. The following terms are described partly in addition.

asynchronous communication: time-wise interrupted delivery of messages, here between a pledge and the registrar or an RA

authenticated self-contained object: a data structure that is cryptographically bound to the identity of its originator by an attached digital signature on the actual object, using a private key of the originator such as the IDevID secret.

backend: placement of a domain component separately from the domain registrar; may be on-site or off-site

BRSKI: Bootstrapping Remote Secure Key Infrastructure [RFC8995]

BRSKI-AE: BRSKI with *A*lternative *E*nrollment, a variation of BRSKI [RFC8995] in which BRSKI-EST, the enrollment protocol between pledge and the registrar, is replaced by enrollment protocols that support end-to-end authentication of the pledge to the RA, such as Lightweight CMP (see LCMPP).

CMP: Certificate Management Protocol [RFC9480]

CSR: Certificate Signing Request

EST: Enrollment over Secure Transport [RFC7030]

IDevID: Initial Device IDentifier of a pledge, provided by the manufacturer and comprising a private key and the related X.509 certificate with its chain

LDevID: Locally significant Device IDentifier of a pledge, provided by its target domain and comprising a private key and the related X.509 certificate with its chain

local RA (LRA): a subordinate RA that is close to entities being enrolled and separate from a subsequent RA. In BRSKI-AE it is needed if a backend RA is used, and in this case, the LRA is co-located with the registrar.

LCMPP: Lightweight CMP Profile [RFC9483]

MASA: Manufacturer Authorized Signing Authority

on-site: locality of a component or service or functionality at the site of the registrar

off-site: locality of component or service or functionality, such as RA or CA, not at the site of the registrar. This may be a central site or a cloud service, to which connection may be intermittent.

pledge: device that is to be bootstrapped into a target domain. It requests an LDevID using IDevID credentials installed by its manufacturer.

RA: Registration Authority, the PKI component to which a CA typically delegates certificate management functions such as authenticating pledges and performing authorization checks on certification requests

registrar: short for domain registrar

site: the locality where an entity, such as a pledge, registrar, or PKI component is deployed. The target domain may have multiple sites.

synchronous communication: time-wise uninterrupted delivery of messages, here between a pledge and a registrar or PKI component

target domain: the domain that a pledge is going to be bootstrapped into

This document utilizes generic terminology regarding PKI management operations to be independent of the terminology of a specific enrollment protocol.

certification request: message requesting a certificate with proof of identity

certification response: message providing the answer to a certification request

attribute request: message requesting content to be included in the certification request

attribute response: message providing the answer to the attribute request

CA certs request: message requesting CA certificates

CA certs response: message providing the answer to a CA certs request

certificate confirm: message stating to the backend PKI that the requester of a certificate received the new certificate and accepted it

PKI/registrar confirm: acknowledgment of the PKI on the certificate confirm

3. Basic Requirements and Mapping to Solutions

Based on the intended target scenarios described in Section 1.1 and the application examples described in Appendix A, the following requirements are derived to support authenticated self-contained objects as containers carrying certification requests.

The following properties are required for a certification request:

* Proof of possession: demonstrates access to the private key corresponding to the public key contained in a certification request. This is typically achieved by a self-signature using the corresponding private key but can also be achieved indirectly, see [RFC4210], Section 4.3.

* Proof of identity, also called proof of origin: provides data origin authentication of the certification request. Typically, this is achieved by a signature using the pledge IDevID secret over some data, which needs to include a sufficiently strong identifier of the pledge, such as the device serial number typically included in the subject of the IDevID certificate.

The remainder of this section gives a non-exhaustive list of solution examples, based on existing technology described in IETF documents.

3.1. Solution Options for Proof of Possession

Certificate signing request (CSR) objects: CSRs are data structures protecting only the integrity of the contained data and providing proof of possession for a (locally generated) private key. Important types of CSR data structures are:

* PKCS #10 [RFC2986]. This very common form of CSR is self-signed to protect its integrity and to prove possession of the private key that corresponds to the public key included in the request.

* Certificate Request Message Format (CRMF, [RFC4211]). This less common but more general CSR format supports several ways of integrity protection and proof of possession. Typically a self-signature is used, which is generated over (part of) the structure with the private key corresponding to the included public key. CRMF also supports further proof-of-possession methods for types of keys that do not have signing capability. For details see [RFC4211], Section 4.

It should be noted that the integrity protection of CSRs includes the public key because it is part of the data signed by the corresponding private key. Yet this signature does not provide data origin authentication, i.e., proof of identity of the requester because the key pair involved is new and therefore does not yet have a confirmed identity associated with it.

3.2. Solution Options for Proof of Identity

Binding a certificate signing request (CSR) to an existing authenticated credential (the BRSKI context, the IDevID certificate) enables proof of origin, which in turn supports an authorization decision on the CSR.

The binding of data origin authentication to the CSR is typically delegated to the protocol used for certificate management. This binding may be achieved through security options in an underlying transport protocol such as TLS if the authorization of the certification request is (sufficiently) done at the next communication hop. Depending on the key type, the binding can also be done in a stronger, transport-independent way by wrapping the CSR with a signature.

This requirement is addressed by existing enrollment protocols in various ways, such as:

* EST [RFC7030], also its variant EST-coaps [RFC9148], utilizes PKCS #10 to encode Certificate Signing Requests (CSRs). While such a CSR has not been designed to include proof of origin, there is a limited, indirect way of binding it to the source authentication of the underlying TLS session. This is achieved by including in the CSR the tls-unique value [RFC5929] resulting from the TLS handshake. As this is optionally supported by the EST "/simpleenroll" endpoint used in BRSKI and the TLS handshake employed in BRSKI includes certificate-based client authentication of the pledge with its IDevID credentials, the proof of pledge identity being an authenticated TLS client can be bound to the CSR.

  Yet this binding is only valid in the context of the TLS session established with the registrar acting as the EST server and typically also as an RA. So even such a cryptographic binding of the authenticated pledge identity to the CSR is not visible nor verifiable to authorization points outside the registrar, such as a (second) RA in the backend. What the registrar needs to do is to authenticate and pre-authorize the pledge and to indicate this to the (second) RA by signing the forwarded certification request with its private key and a related certificate that has the id-kp-cmcRA extended key usage attribute.

[RFC7030], Section 2.5 sketches wrapping PKCS #10-formatted CSRs with a Full PKI Request message sent to the "/fullcmc" endpoint. This would allow for source authentication at the message level, such that the registrar could forward it to external RAs in a meaningful way. This approach is so far not sufficiently described and likely has not been implemented.

* SCEP [RFC8894] supports using a shared secret (passphrase) or an existing certificate to protect CSRs based on SCEP Secure Message Objects using CMS wrapping ([RFC5652]). Note that the wrapping using an existing IDevID in SCEP is referred to as 'renewal'. This way SCEP does not rely on the security of the underlying message transfer.

* CMP [RFC4210] [RFC9480] supports using a shared secret (passphrase) or an existing certificate, which may be an IDevID credential, to authenticate certification requests via the PKIProtection structure in a PKIMessage. The certification request is typically encoded utilizing CRMF, while PKCS #10 is supported as an alternative. Thus, CMP does not rely on the security of the underlying message transfer.

* CMC [RFC5272] also supports utilizing a shared secret (passphrase) or an existing certificate to protect certification requests, which can be either in CRMF or PKCS #10 structure. The proof of identity can be provided as part of a FullCMCRequest, based on CMS [RFC5652] and signed with an existing IDevID secret. Thus, CMC does not rely on the security of the underlying message transfer.

To sum up, EST does not meet the requirements for authenticated self-contained objects, but SCEP, CMP, and CMC do. This document primarily focuses on CMP as it has more industry adoption than CMC and SCEP has issues not detailed here.

4. Adaptations to BRSKI

To enable using alternative certificate enrollment protocols supporting end-to-end authentication, asynchronous enrollment, and more general system architectures, BRSKI-AE provides some generalizations on BRSKI [RFC8995]. This way, authenticated self-contained objects such as those described in Section 3 above can be used for certificate enrollment, and RA functionality can be deployed freely in the target domain. Parts of the RA functionality can even be distributed over several nodes.

The enhancements are kept to a minimum to ensure the reuse of already defined architecture elements and interactions. In general, the communication follows the BRSKI model and utilizes the existing BRSKI architecture elements. In particular, the pledge initiates communication with the domain registrar and interacts with the MASA as usual for voucher request and response processing.

4.1. Architecture

The key element of BRSKI-AE is that the authorization of a certification request MUST be performed based on an authenticated self-contained object. The certification request is bound in a self-contained way to a proof of origin based on the IDevID credentials. Consequently, the certification request MAY be transferred using any mechanism or protocol. Authentication and authorization of the certification request can be done by the domain registrar and/or by backend domain components. As mentioned in Section 1.1, these components may be offline or off-site. The registrar and other on-site domain components may have no or only temporary (intermittent) connectivity to them.

This leads to generalizations in the placement and enhancements of the logical elements as shown in Figure 1.

```
                                         +------------------------+
   +--------------Drop-Ship--------------| Vendor Service         |
   |                                     +------------------------+
   |                                     | M anufacturer|         |
   |                                     | A uthorized  |Ownership|
   |                                     | S igning     |Tracker  |
   |                                     | A uthority   |         |
   |                                     +--------------+---------+
   |                                                      ^
   |                                                      |
   V                                                      |
+--------+     .........................................  |
|        |     .                                       .  | BRSKI-
|        |     .  +-------+          +--------------+  .  | MASA
| Pledge |     .  | Join  |          | Domain       |<----+
|        |<------>| Proxy |<-------->| Registrar w/ |  .
|        |     .  |.......|          | LRA or RA    |  .
| IDevID |     .  +-------+          +--------------+  .
|        |   BRSKI-AE over TLS                ^        .
+--------+   using, e.g., LCMPP               |        .
               .                              |        .
               ...............................|.........
            on-site (local) domain components |
                                              | e.g., LCMPP
                                              |
 .............................................|..................
 . Public-Key Infrastructure                  v                 .
 . +---------+     +------------------------------------------+ .
 . |         |<----+   Registration Authority                 | .
 . |    CA   +---->|   RA (unless part of Domain Registrar)   | .
 . +---------+     +------------------------------------------+ .
 ................................................................
         backend (central or off-site) domain components
```

Figure 1: Architecture Overview Using Backend PKI Components

The architecture overview in Figure 1 has the same logical elements as BRSKI, but with a more flexible placement of the authentication and authorization checks on certification requests. Depending on the application scenario, the registrar MAY still do all of these checks (as is the case in BRSKI), or part of them.

The following list describes the on-site components in the target domain of the pledge shown in Figure 1.

* Join Proxy: same requirements as in BRSKI, see [RFC8995], Section 4

* Domain Registrar including LRA or RA functionality: in BRSKI-AE, the domain registrar has mostly the same functionality as in BRSKI, namely to act as the gatekeeper of the domain for onboarding new devices and to facilitate the communication of pledges with their MASA and the domain PKI. Yet there are some generalizations and specific requirements:

  1. The registrar MUST support at least one certificate enrollment protocol with authenticated self-contained objects for certification requests. To this end, the URI scheme for addressing endpoints at the registrar is generalized (see Section 4.3).

  2. Rather than having full RA functionality, the registrar MAY act as a local registration authority (LRA) and delegate part of its involvement in certificate enrollment to a backend RA. In such scenarios, the registrar optionally checks certification requests it receives from pledges and forwards them to the backend RA, which performs the remaining parts of the enrollment request validation and authorization. Note that to this end the backend RA may need information regarding the authorization of pledges from the registrar or from other sources. On the way back, the registrar forwards responses by the PKI to the pledge on the same channel.

     To support end-to-end authentication of the pledge across the registrar to the backend RA, the certification request signed by the pledge needs to be upheld and forwarded by the registrar. Therefore, the registrar cannot use for its communication with the PKI an enrollment protocol that is different from the enrollment protocol used between the pledge and the registrar.

  3. The use of a certificate enrollment protocol with authenticated self-contained objects gives freedom how to transfer enrollment messages between the pledge and an RA. BRSKI demands that the RA accept certification requests for LDevIDs only with the consent of the registrar. BRSKI-AE guarantees this also in case that the RA is not part of the registrar, even if the message exchange with backend systems is unprotected and involves further transport hops. See Section 7 for details on how this can be achieved.

Despite the above generalizations to the enrollment phase, the final step of BRSKI, namely the enrollment status telemetry, is kept as it is.

The following list describes the components provided by the vendor or manufacturer outside the target domain.

* MASA: functionality as described in BRSKI [RFC8995]. The voucher exchange with the MASA via the domain registrar is performed as described in BRSKI.

  Note: From the definition of the interaction with the MASA in [RFC8995], Section 5 follows that it may be synchronous (using voucher request with nonces) or asynchronous (using nonceless voucher requests).

* Ownership tracker: as defined in BRSKI.

The following list describes backend target domain components, which may be located on-site or off-site in the target domain.

* RA: performs centralized certificate management functions as a public-key infrastructure for the domain operator. As far as not already done by the domain registrar, it performs the final validation and authorization of certification requests. Otherwise, the RA co-located with the domain registrar directly connects to the CA.

* CA, also called domain CA: generates domain-specific certificates according to certification requests that have been authenticated and authorized by the registrar and/or an extra RA.

Based on the diagram in BRSKI [RFC8995], Section 2.1 and the architectural changes, the original protocol flow is divided into several phases showing commonalities and differences to the original approach as follows.

* Discovery phase: mostly as in BRSKI step (1). For details see Section 4.2.1.

* Identification phase: same as in BRSKI step (2).

* Voucher exchange phase: same as in BRSKI steps (3) and (4).

* Voucher status telemetry: same as in BRSKI directly after step (4).

* Certificate enrollment phase: the use of EST in step (5) is changed to employing a certificate enrollment protocol that uses an authenticated self-contained object for requesting the LDevID certificate.

For transporting the certificate enrollment request and response messages, the (D)TLS channel established between pledge and registrar is MANDATORY to use. To this end, the enrollment protocol, the pledge, and the registrar need to support the use of this existing channel for certificate enrollment. Due to this architecture, the pledge does not need to establish additional connections for certificate enrollment and the registrar retains full control over the certificate enrollment traffic.

* Enrollment status telemetry: the final exchange of BRSKI step (5).

4.2. Message Exchange

The behavior of a pledge described in BRSKI [RFC8995], Section 2.1 is kept, with one major exception. After finishing the Imprint step (4), the Enroll step (5) MUST be performed with an enrollment protocol utilizing authenticated self-contained objects, as explained in Section 3. Section 5 discusses selected suitable enrollment protocols and options applicable.

An abstract overview of the BRSKI-AE protocol can be found at [BRSKI-AE-overview].

4.2.1. Pledge - Registrar Discovery

Discovery as specified in BRSKI [RFC8995], Section 4 does not support the discovery of registrars with enhanced feature sets. A pledge can not find out in this way whether discovered registrars support the certificate enrollment protocol it expects, such as CMP.

As a more general solution, the BRSKI discovery mechanism can be extended to provide up-front information on the capabilities of registrars. Future work such as [I-D.eckert-anima-brski-discovery] may provide this.

In the absence of such a generally applicable solution, BRSKI-AE deployments may use their particular way of doing discovery. Section 5.1 defines a minimalist approach that MAY be used for CMP.

4.2.2. Pledge - Registrar - MASA Voucher Exchange

The voucher exchange is performed as specified in [RFC8995].

4.2.3. Pledge - Registrar - MASA Voucher Status Telemetry

The voucher status telemetry is performed as specified in [RFC8995], Section 5.7.

4.2.4. Pledge - Registrar - RA/CA Certificate Enrollment

This replaces the EST integration for PKI bootstrapping described in [RFC8995], Section 5.9 (while [RFC8995], Section 5.9.4 remains as the final phase, see below).

The certificate enrollment phase may involve the transmission of several messages. Details can depend on the application scenario, the employed enrollment protocol, and other factors.

The only message exchange REQUIRED is for the actual certification request and response. Further message exchanges MAY be performed as needed.

Note: The message exchanges marked OPTIONAL in the below Figure 2 cover all those supported by the use of EST in BRSKI. The last OPTIONAL one, namely certificate confirmation, is not supported by EST, but by CMP and other enrollment protocols.

```
+--------+                              +------------+           +---------
---+
| Pledge |                              | Domain     |           | Operator
   |
|        |                              | Registrar  |           | RA/CA
   |
|        |                              |  (JRC)     |           | (PKI)
   |
+--------+                              +------------+           +---------
---+
     |                                         |                                |
     |  [OPTIONAL request of CA certificates]  |                                |
     |--------- CA Certs Request (1) --------->|                                |
     |                                         |  [OPTIONAL forwarding]         |
     |                                         |----- CA Certs Request ------->|
     |                                         |<---- CA Certs Response -------|
     |<-------- CA Certs Response (2) ---------|                                |
     |                                         |                                |
     |  [OPTIONAL request of attributes        |                                |
     |   to include in Certification Request]  |                                |
     |--------- Attribute Request (3) -------->|                                |
     |                                         |  [OPTIONAL forwarding]         |
     |                                         |----- Attribute Request ------>|
     |                                         |<---- Attribute Response ------|
     |<-------- Attribute Response (4) --------|                                |
     |                                         |                                |
     |  [REQUIRED certification request]       |                                |
     |--------- Certification Request (5) ---->|                                |
     |                                         |  [OPTIONAL forwarding]         |
     |                                         |---- Certification Request --->|
     |                                         |<--- Certification Response ---|
     |<-------- Certification Response (6) ----|                                |
     |                                         |                                |
     |  [OPTIONAL certificate confirmation]    |                                |
     |--------- Certificate Confirm (7) ------>|                                |
     |                                         |  [OPTIONAL forwarding]         |
     |                                         |----- Certificate Confirm ---->|
     |                                         |<---- PKI Confirm -------------|
     |<-------- PKI/Registrar Confirm (8) -----|                                |
```

Figure 2: Certificate Enrollment

It may be noted that connections between the registrar and the PKI components of the operator (RA, CA, etc.) may be intermittent or off-line. Messages should be sent as soon as sufficient transfer capacity is available.

The label [OPTIONAL forwarding] in Figure 2 means that on receiving from a pledge a request message of the given type, the registrar MAY answer the request directly. In this case, it MUST authenticate its responses with the same credentials as used for authenticating itself at the TLS level for the voucher exchange. Otherwise, the registrar MUST forward the request to the RA and forward any resulting response back to the pledge.

The decision of whether to forward a request or to answer it directly can depend on various static and dynamic factors. They include the application scenario, the capabilities of the registrar and of the local RA possibly co-located with the registrar, the enrollment protocol being used, and the specific contents of the request.

Note that there are several options for how the registrar could be able to directly answer requests for CA certificates or for certification request attributes. It could cache responses obtained from the domain PKI and later use their contents for responding to requests asking for the same data. The contents could also be explicitly provisioned at the registrar.

Further note that certification requests typically need to be handled by the backend PKI, but the registrar can answer them directly with an error response in case it determines that such a request should be rejected, for instance, because is not properly authenticated or not authorized.Also, certificate confirmation messages will usually be forwarded to the backend PKI, but if the registrar knows that they are not needed or wanted there it can acknowledge such messages directly.

The following list provides an abstract description of the flow depicted in Figure 2.

* CA Certs Request (1): The pledge optionally requests the latest relevant CA certificates. This ensures that the pledge has the complete set of current CA certificates beyond the pinned-domain-cert (which is contained in the voucher and which may be just the domain registrar certificate).

* CA Certs Response (2): This MUST contain any intermediate CA certificates that the pledge may need to validate certificates and MAY contain the LDevID trust anchor.

* Attribute Request (3): Typically, the automated bootstrapping occurs without local administrative configuration of the pledge. Nevertheless, there are cases in which the pledge may also include in the Certification Request (5) additional attributes that are specific to the target domain. To get these attributes in advance, the attribute request may be used.

* Attribute Response (4): This MUST contain the attributes requested in (3) to be included in the subsequent Certification Request (5).

  For example, [RFC8994], Section 6.11.7.2 specifies how the attribute request is used to signal to the pledge the acp-node-name field required for enrollment into an ACP domain.

* Certification Request (5): This MUST contain the authenticated self-contained object ensuring both the proof of possession of the corresponding private key and the proof of identity of the requester.

* Certification Response (6): This MUST contain on success the requested certificate and MAY include further information, like certificates of intermediate CAs and any additional trust anchors.

* Certificate Confirm (7): An optional confirmation sent after the requested certificate has been received and validated. If sent, it MUST contain a positive or negative confirmation by the pledge to the PKI whether the certificate was successfully enrolled and fits its needs.

* PKI/Registrar Confirm (8): An acknowledgment by the PKI that MUST be sent on reception of the Certificate Confirm.

The generic messages described above may be implemented using any certificate enrollment protocol that supports authenticated self-contained objects for the certification request as described in Section 3. Examples are available in Section 5.

Note that the optional certificate confirmation by the pledge to the PKI described above is independent of the mandatory enrollment status telemetry done between the pledge and the registrar in the final phase of BRSKI-AE, described next.

4.2.5. Pledge - Registrar Enrollment Status Telemetry

The enrollment status telemetry is performed as specified in [RFC8995], Section 5.9.4.

In BRSKI this is described as part of the certificate enrollment step, but due to the generalization on the enrollment protocol described in this document it is regarded as a separate phase here.

4.3. Enhancements to the Endpoint Addressing Scheme of BRSKI

BRSKI-AE provides generalizations to the addressing scheme defined in BRSKI [RFC8995], Section 5 to accommodate alternative enrollment protocols that use authenticated self-contained objects for certification requests. In existing RAs/CAs supporting such an enrollment protocol (see also Section 5), these generalizations can be employed without modifications.

The addressing scheme in BRSKI for certification requests and the related CA certificates and CSR attributes retrieval functions uses the definition from EST [RFC7030]. Here is the example of simple enrollment: "/.well-known/est/simpleenroll". This approach is generalized to the following notation: "/.well-known/<enrollment-protocol>/<request>" in which <enrollment-protocol> refers to a certificate enrollment protocol. Note that enrollment is considered here a message sequence that contains at least a certification request and a certification response. The following conventions are used to provide maximal compatibility with BRSKI:

* <enrollment-protocol>: MUST reference the protocol being used. Existing values include 'est' [RFC7030] as in BRSKI and 'cmp' as in [RFC9483] and Section 5.1 below. Values for other existing protocols such as CMC and SCEP, as well as for newly defined protocols are outside the scope of this document. For use of the <enrollment-protocol> and <request> URI components, they would need to be specified in a suitable RFC and placed into the Well-Known URIs registry, just as EST in [RFC7030].

* <request>: if present, this path component MUST describe, depending on the enrollment protocol being used, the operation requested. Enrollment protocols are expected to define their request endpoints, as done by existing protocols (see also Section 5).

Well-known URIs for various endpoints on the domain registrar are already defined as part of the base BRSKI specification or indirectly by EST. In addition, alternative enrollment endpoints MAY be supported by the registrar.

A pledge SHOULD use the endpoints defined for the enrollment protocol(s) that it can use. It will recognize whether the protocol it uses and the specific request it wants to perform are understood and supported by the domain registrar by sending the request to the respective endpoint according to the above addressing scheme and then evaluating the HTTP status code of the response. If the pledge uses endpoints that are not standardized, it risks that the registrar does not recognize a request and thus may reject it, even if the registrar supports the intended protocol and operation.

The following list of endpoints provides an illustrative example of a domain registrar supporting several options for EST as well as for CMP to be used in BRSKI-AE. The listing contains the supported endpoints to which the pledge may connect for bootstrapping. This includes the voucher handling as well as the enrollment endpoints. The CMP-related enrollment endpoints are defined as well-known URIs in CMP Updates [RFC9480] and the Lightweight CMP Profile [RFC9483].

```
/.well-known/brski/voucherrequest
/.well-known/brski/voucher_status
/.well-known/brski/enrollstatus
/.well-known/est/cacerts
/.well-known/est/csrattrs
/.well-known/est/fullcmc
/.well-known/cmp/getcacerts
/.well-known/cmp/getcertreqtemplate
/.well-known/cmp/initialization
/.well-known/cmp/pkcs10
```

5. Instantiation with Existing Enrollment Protocols

This section maps the generic requirements to support proof of possession and proof of identity to selected existing certificate enrollment protocols and specifies further aspects of using such enrollment protocols in BRSKI-AE.

5.1. BRSKI-CMP: BRSKI-AE instantiated with CMP

Instead of referring to CMP as specified in [RFC4210] and [RFC9480], this document refers to the Lightweight CMP Profile (LCMPP) [RFC9483] because the subset of CMP defined there is sufficient for the functionality needed here.

When using CMP, adherence to the LCMPP [RFC9483] is REQUIRED. In particular, the following specific requirements apply (cf. Figure 2).

* CA Certs Request (1) and Response (2):
  Requesting CA certificates is OPTIONAL.
  If supported, it SHALL be implemented as specified in [RFC9483], Section 4.3.1.

* Attribute Request (3) and Response (4):
  Requesting certification request attributes is OPTIONAL.
  If supported, it SHALL be implemented as specified in [RFC9483], Section 4.3.3.

  Alternatively, the registrar MAY modify the contents of the requested certificate contents as specified in [RFC9483], Section 5.2.3.2.

* Certification Request (5) and Response (6):
  Certificates SHALL be requested and provided as specified in the LCMPP [RFC9483], Section 4.1.1 (based on CRMF) or [RFC9483], Section 4.1.4 (based on PKCS #10).

  Proof of possession SHALL be provided in a way suitable for the key type. Proof of identity SHALL be provided by signature-based protection of the certification request message as outlined in [RFC9483], Section 3.2 using the IDevID secret.

  When the registrar forwards a certification request by the pledge to a backend RA/CA, the registrar is RECOMMENDED to wrap the original certification request in a nested message signed with its own credentials as described in [RFC9483], Section 5.2.2.1. This explicitly conveys the consent by the registrar to the RA while retaining the original certification request message with its proof of origin provided by the pledge signature.

  In case additional trust anchors (besides the pinned-domain-cert) need to be conveyed to the pledge, this SHOULD be done in the caPubs field of the certification response rather than in a CA Certs Response.

* Certificate Confirm (7) and PKI/Registrar Confirm (8):
  Explicit confirmation of new certificates to the RA/CA MAY be used as specified in [RFC9483], Section 4.1.1.

  Note that independently of the certificate confirmation within CMP, enrollment status telemetry with the registrar at BRSKI level will be performed as described in [RFC8995], Section 5.9.4.

* If delayed delivery of CMP messages is needed (for instance, to support enrollment over an asynchronous channel), it SHALL be performed as specified in Section 4.4 and Section 5.1.2 of [RFC9483].

How messages are exchanged between the registrar and backend PKI components (i.e., RA and/or CA) is out of scope of this document. Since CMP is independent of message transfer, the mechanism for this exchange can be freely chosen according to the needs of the application scenario. For the applicable security considerations, see Section 7. Further guidance can be found in [RFC9483], Section 6.

BRSKI-AE with CMP can also be combined with Constrained BRSKI [I-D.ietf-anima-constrained-voucher], using CoAP for enrollment message transport as described by CoAP Transport for CMP [RFC9482]. In this scenario, the EST-specific parts of [I-D.ietf-anima-constrained-voucher] do not apply.

For BRSKI-AE scenarios where a general solution (cf. Section 4.2.1) for discovering registrars with CMP support is not available, the following minimalist approach MAY be used. Perform discovery as defined in BRSKI [RFC8995], Appendix B but using the service name "brski-registrar-cmp" (defined in Section 6) instead of "brski-registrar" (defined in [RFC8995], Section 8.6). Note that this approach does not support join proxies.

5.2. Support of Other Enrollment Protocols

Further instantiations of BRSKI-AE can be done. They are left for future work.

In particular, CMC [RFC5272] (using its in-band source authentication options) and SCEP [RFC8894] (using its 'renewal' option) could be used.

The fullCMC variant of EST sketched in [RFC7030], Section 2.5 might also be used here. For EST-fullCMC further specification is necessary.

6. IANA Considerations

This document requires one IANA action: register in the Service Name and Transport Protocol Port Number Registry (https://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xhtml) the following service name.

Service Name: brski-registrar-cmp
Transport Protocol(s): tcp
Assignee: IESG iesg@ietf.org (mailto:iesg@ietf.org)
Contact: IESG iesg@ietf.org (mailto:iesg@ietf.org)
Description: Bootstrapping Remote Secure Key Infrastructure registrar with CMP capabilities according to the Lightweight CMP Profile (LCMPP, [RFC9483])
Reference: [THISRFC]

Note: We chose here the suffix "cmp" rather than some other abbreviation like "lcmpp" mainly because this document defines the normative CMP instantiation of BRSKI-AE, which implies adherence to LCMPP is necessary and sufficient.

7. Security Considerations

The security considerations laid out in BRSKI [RFC8995] apply to the discovery and voucher exchange as well as for the status exchange information.

In particular, even if the registrar delegates part or all of its RA role during certificate enrollment to a separate system, it still must be made sure that the registrar takes part in the decision on accepting or declining a request to join the domain, as required in [RFC8995], Section 5.3. As this pertains also to obtaining a valid domain-specific certificate, it must be made sure that a pledge can not circumvent the registrar in the decision of whether it is granted an LDevID certificate by the CA. There are various ways how to fulfill this, including:

* implicit consent

* the registrar signals its consent to the RA out-of-band before or during the enrollment phase, for instance by entering the pledge identity in a database.

* the registrar provides its consent using an extra message that is transferred on the same channel as the enrollment messages, possibly in a TLS tunnel.

* the registrar explicitly states its consent by signing, in addition to the pledge, the authenticated self-contained certificate enrollment request message.

Note: If EST was used, the registrar could give implicit consent on a certification request by forwarding the request to a PKI entity using a connection authenticated with a certificate containing an id-kp-cmcRA extension.

When CMP is used, the security considerations laid out in the LCMPP [RFC9483] apply.

Note that CMP messages are not encrypted. This may give eavesdroppers insight into which devices are bootstrapped into the domain, and this in turn might also be used to selectively block the enrollment of certain devices. To prevent this, the underlying message transport channel can be encrypted. This is already provided by TLS between the pledge and the registrar, and for the onward exchange with backend systems, encryption may need to be added.

8. Acknowledgments

We thank Eliot Lear for his contributions as a co-author at an earlier draft stage.

We thank Brian E. Carpenter, Michael Richardson, and Giorgio Romanenghi for their input and discussion on use cases and call flows.

Moreover, we thank Toerless Eckert (document shepherd), Barry Leiba (SECdir review), Mahesh Jethanandani (IETF area director), Meral Shirazipour (Gen-ART reviewer), Michael Richardson (ANIMA design team member), as well as Rajeev Ranjan, Rufus Buschart, Andreas Reiter, and Szofia Fazekas-Zisch (Siemens colleagues) for their reviews with suggestions for improvements.

9. References

9.1. Normative References

[IEEE_802.1AR-2018]
IEEE, "IEEE Standard for Local and Metropolitan Area Networks - Secure Device Identity", IEEE 802.1AR-2018, DOI 10.1109/IEEESTD.2018.8423794, August 2018, <https://ieeexplore.ieee.org/document/8423794>.

[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>.

[RFC5280] Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 5280, DOI 10.17487/RFC5280, May 2008, <https://www.rfc-editor.org/rfc/rfc5280>.

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

[RFC8995] 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>.

[RFC9483] Brockhaus, H., von Oheimb, D., and S. Fries, "Lightweight Certificate Management Protocol (CMP) Profile", RFC 9483, DOI 10.17487/RFC9483, November 2023, <https://www.rfc-editor.org/rfc/rfc9483>.

9.2. Informative References

[BRSKI-AE-overview]
S. Fries and D. von Oheimb, "BRSKI-AE Protocol Overview", March 2023, <https://datatracker.ietf.org/meeting/116/materials/slides-116-anima-update-on-brski-ae-alternative-enrollment-protocols-in-brski-00>. Graphics on slide 4 of the status update on the BRSKI-AE draft 04 at IETF 116.

[I-D.eckert-anima-brski-discovery]
Eckert, T. T., von Oheimb, D., and E. Dijk, "Discovery for BRSKI variations", Work in Progress, Internet-Draft, draft-eckert-anima-brski-discovery-01, 23 October 2023, <https://datatracker.ietf.org/doc/html/draft-eckert-anima-brski-discovery-01>.

[I-D.ietf-anima-constrained-voucher]
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-24, 3 March 2024, <https://datatracker.ietf.org/doc/html/draft-ietf-anima-constrained-voucher-24>.

[IEC-62351-9]
International Electrotechnical Commission, "IEC 62351 - Power systems management and associated information exchange - Data and communications security - Part 9: Cyber security key management for power system equipment", IEC 62351-9, May 2017.

[ISO-IEC-15118-2]
International Standardization Organization / International Electrotechnical Commission, "ISO/IEC 15118-2 Road vehicles - Vehicle-to-Grid Communication Interface - Part 2: Network and application protocol requirements", ISO/IEC 15118-2, April 2014.

[NERC-CIP-005-5]
North American Reliability Council, "Cyber Security - Electronic Security Perimeter", CIP 005-5, December 2013.

[OCPP] Open Charge Alliance, "Open Charge Point Protocol 2.0.1 (Draft)", December 2019.

[RFC2986] Nystrom, M. and B. Kaliski, "PKCS #10: Certification Request Syntax Specification Version 1.7", RFC 2986, DOI 10.17487/RFC2986, November 2000, <https://www.rfc-editor.org/rfc/rfc2986>.

[RFC4210] Adams, C., Farrell, S., Kause, T., and T. Mononen, "Internet X.509 Public Key Infrastructure Certificate Management Protocol (CMP)", RFC 4210, DOI 10.17487/RFC4210, September 2005, <https://www.rfc-editor.org/rfc/rfc4210>.

[RFC4211] Schaad, J., "Internet X.509 Public Key Infrastructure Certificate Request Message Format (CRMF)", RFC 4211, DOI 10.17487/RFC4211, September 2005, <https://www.rfc-editor.org/rfc/rfc4211>.

[RFC5272] Schaad, J. and M. Myers, "Certificate Management over CMS (CMC)", RFC 5272, DOI 10.17487/RFC5272, June 2008, <https://www.rfc-editor.org/rfc/rfc5272>.

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

[RFC5929] Altman, J., Williams, N., and L. Zhu, "Channel Bindings for TLS", RFC 5929, DOI 10.17487/RFC5929, July 2010, <https://www.rfc-editor.org/rfc/rfc5929>.

[RFC6955] Schaad, J. and H. Prafullchandra, "Diffie-Hellman Proof-of-Possession Algorithms", RFC 6955, DOI 10.17487/RFC6955, May 2013, <https://www.rfc-editor.org/rfc/rfc6955>.

[RFC7030] Pritikin, M., Ed., Yee, P., Ed., and D. Harkins, Ed., "Enrollment over Secure Transport", RFC 7030, DOI 10.17487/RFC7030, October 2013, <https://www.rfc-editor.org/rfc/rfc7030>.

[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>.

[RFC8894] Gutmann, P., "Simple Certificate Enrolment Protocol", RFC 8894, DOI 10.17487/RFC8894, September 2020, <https://www.rfc-editor.org/rfc/rfc8894>.

[RFC8994] Eckert, T., Ed., Behringer, M., Ed., and S. Bjarnason, "An Autonomic Control Plane (ACP)", RFC 8994, DOI 10.17487/RFC8994, May 2021, <https://www.rfc-editor.org/rfc/rfc8994>.

[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>.

[RFC9480] Brockhaus, H., von Oheimb, D., and J. Gray, "Certificate Management Protocol (CMP) Updates", RFC 9480, DOI 10.17487/RFC9480, November 2023, <https://www.rfc-editor.org/rfc/rfc9480>.

[RFC9482] Sahni, M., Ed. and S. Tripathi, Ed., "Constrained Application Protocol (CoAP) Transfer for the Certificate Management Protocol", RFC 9482, DOI 10.17487/RFC9482, November 2023, <https://www.rfc-editor.org/rfc/rfc9482>.

[UNISIG-Subset-137]
UNISIG, "Subset-137; ERTMS/ETCS On-line Key Management FFFIS; V1.0.0", December 2015, <https://www.era.europa.eu/sites/default/files/filesystem/ertms/ccs_tsi_annex_a_-_mandatory_specifications/set_of_specifications_3_etcs_b3_r2_gsm-r_b1/index083_-_subset-137_v100.pdf>. http://www.kmc-subset137.eu/index.php/download/

Appendix A. Application Examples

This informative annex provides some detail about application examples.

A.1. Rolling Stock

Rolling stock or railroad cars contain a variety of sensors, actuators, and controllers, which communicate within the railroad car but also exchange information between railroad cars forming a train, with track-side equipment, and/or possibly with backend systems. These devices are typically unaware of backend system connectivity. Enrolling certificates may be done during maintenance cycles of the railroad car, but can already be prepared during operation. Such asynchronous enrollment will include generating certification requests, which are collected and later forwarded for processing whenever the railroad car gets connectivity with the backend PKI of the operator. The authorization of the certification request is then done based on the operator’s asset/inventory information in the backend.

UNISIG has included a CMP profile for the enrollment of TLS client and server X.509 certificates of on-board and track-side components in the Subset-137 specifying the ETRAM/ETCS online key management for train control systems [UNISIG-Subset-137].

A.2. Building Automation

In building automation scenarios, a detached building or the basement of a building may be equipped with sensors, actuators, and controllers that are connected to each other in a local network but with only limited or no connectivity to a central building management system. This problem may occur during installation time but also during operation. In such a situation a service technician collects the necessary data and transfers it between the local network and the central building management system, e.g., using a laptop or a mobile phone. This data may comprise parameters and settings required in the operational phase of the sensors/actuators, like a component certificate issued by the operator to authenticate against other components and services.

The collected data may be provided by a domain registrar already existing in the local network. In this case connectivity to the backend PKI may be facilitated by the service technician’s laptop. Alternatively, the data can also be collected from the pledges directly and provided to a domain registrar deployed in a different network in preparation for the operational phase. In this case, connectivity to the domain registrar may also be facilitated by the service technician’s laptop.

A.3. Substation Automation

In electrical substation automation scenarios, a control center typically hosts PKI services to issue certificates for Intelligent Electronic Devices operated in a substation. Communication between the substation and control center is performed through a proxy/gateway/DMZ, which terminates protocol flows. Note that [NERC-CIP-005-5] requires inspection of protocols at the boundary of a security perimeter (the substation in this case). In addition, security management in substation automation assumes central support of several enrollment protocols to support the various capabilities of IEDs from different vendors. The IEC standard IEC62351-9

[IEC-62351-9] specifies for the infrastructure side mandatory support of two enrollment protocols: SCEP [RFC8894] and EST [RFC7030], while an Intelligent Electronic Device may support only one of them.

A.4. Electric Vehicle Charging Infrastructure

For electric vehicle charging infrastructure, protocols have been defined for the interaction between the electric vehicle and the charging point (e.g., ISO 15118-2 [ISO-IEC-15118-2]) as well as between the charging point and the charging point operator (e.g. OCPP [OCPP]). Depending on the authentication model, unilateral or mutual authentication is required. In both cases, the charging point uses an X.509 certificate to authenticate itself in TLS channels between the electric vehicle and the charging point. The management of this certificate depends, among others, on the selected backend connectivity protocol. In the case of OCPP, this protocol is meant to be the only communication protocol between the charging point and the backend, carrying all information to control the charging operations and maintain the charging point itself. This means that the certificate management needs to be handled in-band of OCPP. This requires the ability to encapsulate the certificate management messages in a transport-independent way. Authenticated self-containment will support this by allowing the transport without a separate enrollment protocol, binding the messages to the identity of the communicating endpoints.

A.5. Infrastructure Isolation Policy

This refers to any case in which network infrastructure is normally isolated from the Internet as a matter of policy, most likely for security reasons. In such a case, limited access to external PKI services will be allowed in carefully controlled short periods of time, for example when a batch of new devices is deployed, and forbidden or prevented at other times.

A.6. Sites with Insufficient Level of Operational Security

The RA performing (at least part of) the authorization of a certification request is a critical PKI component and therefore requires higher operational security than components utilizing the issued certificates for their security features. CAs may also demand higher security in the registration procedures from RAs, which domain registrars with co-located RAs may not be able to fulfill. Especially the CA/Browser forum currently increases the security requirements in the certificate issuance procedures for publicly trusted certificates, i.e., those placed in trust stores of browsers, which may be used to connect with devices in the domain. In case the on-site components of the target domain can not be operated securely enough for the needs of an RA, this service should be transferred to an off-site backend component that has a sufficient level of security.

Appendix B. History of Changes TBD RFC Editor: please delete

List of reviewers:

* Toerless Eckert (document shepherd)
* Barry Leiba (SECdir)
* Mahesh Jethanandani (IETF area director)
* Meral Shirazipour (Gen-ART reviewer)
* Michael Richardson (ANIMA design team)
* Rajeev Ranjan, Rufus Buschart, Szofia Fazekas-Zisch, etc. (Siemens)
* YANGDOCTORS Early review of 2021-08-15 (https://datatracker.ietf.org/doc/review-ietf-anima-brski-async-enroll-03-yangdoctors-early-rahman-2021-08-15/) referred to the PRM aspect of draft-ietf-anima-brski-async-enroll-03 (https://datatracker.ietf.org/doc/draft-ietf-anima-brski-async-enroll/03/). This has been carved out of the draft to a different one and thus is no more applicable here.

IETF draft ae-11 -> ae-12:

* Fix minor issues introduced during authors' response to the AD review,
  including nits spotted in the Gen-ART review by Meral Shirazipour

IETF draft ae-10 -> ae-11:

* In response to AD review by Mahesh Jethanandani,
  - replace most occurrences of 'Note:' by 'Note that' or the like
  - move 2nd paragraph of abstract to the introduction
  - remove section 1.2 and merge its first paragraph with the preceding section
  - reconsider normative language, replacing one 'may' by 'MAY' in section 4.1

  - fix several ambiguities and hard-to-read sentences by re-phrasing them

  - make wording more consistent, in particular: 'certification request'

  - fix a number of (mostly grammar) nits

* Improve item on limitations of PKCS#10 regarding keys that cannot sign

IETF draft ae-09 -> ae-10:

* Add reference to RFC 8633 at first occurrence of 'voucher' (fixes #37)

* Update reference of CoAP Transfer for CMP from I-D to RFC 9482

* Move RFC 4210 and RFC 9480 references from normative to informative

* Fix p10 vs. pkcs10 entry in list of example endpoints in Section 4.3

* Minor fix in Figure 1 and few text tweaks due to Siemens-internal review

* Extend the list of reviewers and acknowledgments by two Siemens colleagues

IETF draft ae-08 -> ae-09:

* In response to review by Toerless,

  - tweak abstract to make meaning of 'alternative enrollment' more clear

  - expand on first use not "well-known" abbreviations, such as 'EST',
    adding also a references on their first use

  - add summary and reason for choosing CMP at end of Section 3.2

  - remove paragraph on optimistic discovery in controlled environments

  - mention role of reviewers also in acknowledgments section

* A couple of grammar and spelling fixes

IETF draft ae-07 -> ae-08:

* Update references to service names in Section 5.1

IETF draft ae-06 -> ae-07:

* Update subsections on discovery according to discussion in the design team

* In Section 5.1, replace 'mandatory' by 'REQUIRED' regarding adherence to LCMPP,
  in response to SECDIR Last Call Review of ae-06 by Barry Leiba

IETF draft ae-05 -> ae-06:

* Extend section on discovery according to discussion in the design team

* Make explicit that MASA voucher status telemetry is as in BRSKI

* Add note that on delegation, RA may need info on pledge authorization

IETF draft ae-04 -> ae-05:

* Remove entries from the terminology section that should be clear from BRSKI

* Tweak use of the terms IDevID and LDevID and replace PKI RA/CA by RA/CA

* Add the abbreviation 'LCMPP' for Lightweight CMP Profile to the terminology section

* State clearly in Section 5.1 that LCMPP is mandatory when using CMP

* Change URL of BRSKI-AE-overview graphics to slide on IETF 116 meeting material

IETF draft ae-03 -> ae-04:

* In response to SECDIR Early Review of ae-03 by Barry Leiba,

  - replace 'end-to-end security' by the more clear 'end-to-end authentication'

- restrict the meaning of the abbreviation 'AE' to 'Alternative Enrollment'

- replace 'MAY' by 'may' in requirement on delegated registrar actions

- re-phrase requirement on certification request exchange, avoiding MANDATORY

- mention that further protocol names need be put in Well-Known URIs registry

- explain consequence of using non-standard endpoints, not following SHOULD

- remove requirement that 'caPubs' field in CMP responses SHOULD NOT be used

- add paragraph in security considerations on additional use of TLS for CMP

* In response to further internal reviews and suggestions for generalization,

  - significantly cut down the introduction because the original motivations and most explanations are no more needed and would just make it lengthy to read

  - sort out asynchronous vs. offline transfer, off-site vs. backend components

  - improve description of CSRs and proof of possession vs. proof of origin

  - clarify that the channel between pledge and registrar is not restricted to TLS, but in connection with constrained BRSKI may also be DTLS. Also move the references to Constrained BRSKI and CoAPS to better contexts.

  - clarify that the registrar must not be circumvented in the decision to grant and LDevID, and give hints and recommendations how to make sure this

  - clarify that the cert enrollment phase may involve additional messages and that BRSKI-AE replaces [RFC8995], Section 5.9 (except Section 5.9.4)

   - the certificate enrollment protocol needs to support transport over (D)TLS only as far as its messages are transported between pledge and registrar.

   - the certificate enrollment protocol chosen between pledge and registrar needs to be used also for the upstream enrollment exchange with the PKI only if end-to-end authentication shall be achieved across the registrar to the PKI.

   - add that with CMP, further trust anchors SHOULD be transported via caPubs

   - remove the former Appendix A: "Using EST for Certificate Enrollment", moving relevant points to the list of scenarios in Section 1.1: "Supported Scenarios",

   - streamline the item on EST in Section 3.2: "Solution Options for Proof of Identity",

   - various minor editorial improvements like making the wording more consistent

IETF draft ae-02 -> ae-03:

* In response to review by Toerless Eckert,

   - many editorial improvements and clarifications as suggested, such as the comparison to plain BRSKI, the description of offline vs. synchronous message transfer and enrollment, and better differentiation of RA flavors.

   - clarify that for transporting certificate enrollment messages between pledge and registrar, the TLS channel established between these two (via the join proxy) is used and the enrollment protocol MUST support this.

   - clarify that the enrollment protocol chosen between pledge and registrar MUST also be used for the upstream enrollment exchange with the PKI.

   - extend the description and requirements on how during the certificate enrollment phase the registrar MAY handle requests by the pledge itself and otherwise MUST forward them to the PKI and forward responses to the pledge.

* Change "The registrar MAY offer different enrollment protocols" to "The registrar MUST support at least one certificate enrollment protocol ..."

* In response to review by Michael Richardson,

  - slightly improve the structuring of the Message Exchange Section 4.2 and add some detail on the request/response exchanges for the enrollment phase

  - merge the 'Enhancements to the Addressing Scheme' Section 4.3 with the subsequent one: 'Domain Registrar Support of Alternative Enrollment Protocols'

  - add reference to SZTP (RFC 8572)

  - extend venue information

  - convert output of ASCII-art figures to SVG format

  - various small other text improvements as suggested/provided

* Remove the tentative informative application to EST-fullCMC

* Move Eliot Lear from co-author to contributor, add Eliot to the acknowledgments

* Add explanations for terms such as 'target domain' and 'caPubs'

* Fix minor editorial issues and update some external references

IETF draft ae-01 -> ae-02:

* Architecture: clarify registrar role including RA/LRA/enrollment proxy

* CMP: add reference to CoAP Transport for CMPV2 and Constrained BRSKI

* Include venue information

From IETF draft 05 -> IETF draft ae-01:

* Renamed the repo and files from 'anima-brski-async-enroll' to 'anima-brski-ae'

* Added graphics for abstract protocol overview as suggested by Toerless Eckert

* Balanced (sub-)sections and their headers

* Added details on CMP instance, now called BRSKI-CMP

From IETF draft 04 -> IETF draft 05:

* David von Oheimb became the editor.

* Streamline wording, consolidate terminology, improve grammar, etc.

* Shift the emphasis towards supporting alternative enrollment protocols.

* Update the title accordingly - preliminary change to be approved.

* Move comments on EST and detailed application examples to informative annex.

* Move the remaining text of section 3 as two new sub-sections of section 1.

From IETF draft 03 -> IETF draft 04:

* Moved UC2-related parts defining the pledge in responder mode to a separate document. This required changes and adaptations in several sections. Main changes concerned the removal of the subsection for UC2 as well as the removal of the YANG model related text as it is not applicable in UC1.

* Updated references to the Lightweight CMP Profile (LCMPP).

* Added David von Oheimb as co-author.

From IETF draft 02 -> IETF draft 03:

* Housekeeping, deleted open issue regarding YANG voucher-request in UC2 as voucher-request was enhanced with additional leaf.

* Included open issues in YANG model in UC2 regarding assertion value agent-proximity and CSR encapsulation using SZTP sub module).

From IETF draft 01 -> IETF draft 02:

* Defined call flow and objects for interactions in UC2. Object format based on draft for JOSE signed voucher artifacts and aligned the remaining objects with this approach in UC2 .

* Terminology change: issue #2 pledge-agent -> registrar-agent to better underline agent relation.

* Terminology change: issue #3 PULL/PUSH -> pledge-initiator-mode and pledge-responder-mode to better address the pledge operation.

* Communication approach between pledge and registrar-agent changed by removing TLS-PSK (former section TLS establishment) and associated references to other drafts in favor of relying on higher layer exchange of signed data objects. These data objects are included also in the pledge-voucher-request and lead to an extension of the YANG module for the voucher-request (issue #12).

* Details on trust relationship between registrar-agent and registrar (issue #4, #5, #9) included in UC2.

* Recommendation regarding short-lived certificates for registrar-agent authentication towards registrar (issue #7) in the security considerations.

* Introduction of reference to agent signing certificate using SKID in agent signed data (issue #11).

* Enhanced objects in exchanges between pledge and registrar-agent to allow the registrar to verify agent-proximity to the pledge (issue #1) in UC2.

* Details on trust relationship between registrar-agent and pledge (issue #5) included in UC2.

* Split of use case 2 call flow into sub sections in UC2.

From IETF draft 00 -> IETF draft 01:

* Update of scope in Section 1.1 to include in which the pledge acts as a server. This is one main motivation for use case 2.

* Rework of use case 2 to consider the transport between the pledge and the pledge-agent. Addressed is the TLS channel establishment between the pledge-agent and the pledge as well as the endpoint definition on the pledge.

* First description of exchanged object types (needs more work)

* Clarification in discovery options for enrollment endpoints at the domain registrar based on well-known endpoints in Section 4.3 do not result in additional /.well-known URIs. Update of the illustrative example. Note that the change to /brski for the voucher-related endpoints has been taken over in the BRSKI main document.

* Updated references.

* Included Thomas Werner as additional author for the document.

From individual version 03 -> IETF draft 00:

* Inclusion of discovery options of enrollment endpoints at the domain registrar based on well-known endpoints in Section 4.3 as replacement of section 5.1.3 in the individual draft. This is intended to support both use cases in the document. An illustrative example is provided.

* Missing details provided for the description and call flow in pledge-agent use case UC2, e.g. to accommodate distribution of CA certificates.

* Updated CMP example in Section 5 to use Lightweight CMP instead of CMP, as the draft already provides the necessary /.well-known endpoints.

* Requirements discussion moved to separate section in Section 3. Shortened description of proof-of-identity binding and mapping to existing protocols.

* Removal of copied call flows for voucher exchange and registrar discovery flow from [RFC8995] in Section 4 to avoid doubling or text or inconsistencies.

* Reworked abstract and introduction to be more crisp regarding the targeted solution. Several structural changes in the document to have a better distinction between requirements, use case description, and solution description as separate sections. History moved to appendix.

From individual version 02 -> 03:

* Update of terminology from self-contained to authenticated self-contained object to be consistent in the wording and to underline the protection of the object with an existing credential. Note that the naming of this object may be discussed. An alternative name may be attestation object.

* Simplification of the architecture approach for the initial use case having an off-site PKI.

* Introduction of a new use case utilizing authenticated self-contain objects to onboard a pledge using a commissioning tool containing a pledge-agent. This requires additional changes in

the BRSKI call flow sequence and led to changes in the introduction, the application example,and also in the related BRSKI-AE call flow.

* Update of provided examples of the addressing approach used in BRSKI to allow for support of multiple enrollment protocols in Section 4.3.

From individual version 01 -> 02:

* Update of introduction text to clearly relate to the usage of IDevID and LDevID.

* Definition of the addressing approach used in BRSKI to allow for support of multiple enrollment protocols in Section 4.3. This section also contains a first discussion of an optional discovery mechanism to address situations in which the registrar supports more than one enrollment approach. Discovery should avoid that the pledge performs a trial and error of enrollment protocols.

* Update of description of architecture elements and changes to BRSKI in Section 4.1.

* Enhanced consideration of existing enrollment protocols in the context of mapping the requirements to existing solutions in Section 3 and in Section 5.

From individual version 00 -> 01:

* Update of examples, specifically for building automation as well as two new application use cases in Appendix A.

* Deletion of asynchronous interaction with MASA to not complicate the use case. Note that the voucher exchange can already be handled in an asynchronous manner and is therefore not considered further. This resulted in removal of the alternative path the MASA in Figure 1 and the associated description in Section 4.1.

* Enhancement of description of architecture elements and changes to BRSKI in Section 4.1.

* Consideration of existing enrollment protocols in the context of mapping the requirements to existing solutions in Section 3.

* New section starting Section 5 with the mapping to existing enrollment protocols by collecting boundary conditions.

Contributors

Eliot Lear
Cisco Systems
Richtistrasse 7
CH-8304 Wallisellen
Switzerland
Phone: +41 44 878 9200
Email: lear@cisco.com

Authors' Addresses

David von Oheimb (editor)
Siemens AG
Otto-Hahn-Ring 6
81739 Munich
Germany
Email: david.von.oheimb@siemens.com
URI: https://www.siemens.com/

Steffen Fries
Siemens AG
Otto-Hahn-Ring 6
81739 Munich
Germany
Email: steffen.fries@siemens.com
URI: https://www.siemens.com/

Hendrik Brockhaus
Siemens AG
Otto-Hahn-Ring 6
81739 Munich
Germany
Email: hendrik.brockhaus@siemens.com
URI: https://www.siemens.com/

ANIMA WG S. Fries
Internet-Draft H. Brockhaus
Intended status: Standards Track Siemens
Expires: December 26, 2021 E. Lear
Cisco Systems
T. Werner
Siemens
June 24, 2021

Support of asynchronous Enrollment in BRSKI (BRSKI-AE)
draft-ietf-anima-brski-async-enroll-03

Abstract

This document describes enhancements of bootstrapping a remote secure key infrastructure (BRSKI, [RFC8995]) to also operate in domains featuring no or only timely limited connectivity between involved components. Further enhancements are provided to perform the BRSKI approach in environments, in which the role of the pledge changes from a client to a server . This changes the interaction model from a pledge-initiator-mode to a pledge-responder-mode. To support both use cases, BRSKI-AE relies on the exchange of authenticated self-contained objects (signature-wrapped objects) also for requesting and distributing of domain specific device certificates. The defined approach is agnostic regarding the utilized enrollment protocol allowing the application of existing and potentially new certificate management protocols.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on December 26, 2021.

Copyright Notice

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

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

Table of Contents

1. Introduction

BRSKI as defined in [RFC8995] specifies a solution for secure zero-touch (automated) bootstrapping of devices (pledges) in a (customer) site domain. This includes the discovery of network elements in the target domain, time synchronization, and the exchange of security information necessary to establish trust between a pledge and the domain. Security information about the target domain, specifically the target domain certificate, is exchanged utilizing voucher objects as defined in [RFC8366]. These vouchers are authenticated self-contained (signed) objects, which may be provided online (synchronous) or offline (asynchronous) via the domain registrar to the pledge and originate from a Manufacturer's Authorized Signing Authority (MASA).

For the enrollment of devices BRSKI relies on EST [RFC7030] to request and distribute target domain specific device certificates. EST in turn relies on a binding of the certification request to an underlying TLS connection between the EST client and the EST server. According to BRSKI the domain registrar acts as EST server and is also acting as registration authority (RA) or local registration authority (LRA). The binding to TLS is used to protect the exchange of a certification request (for a LDevID EE certificate) and to provide data origin authentication (client identity information), to support the authorization decision for processing the certification request. The TLS connection is mutually authenticated and the client-side authentication utilizes the pledge's manufacturer issued device certificate (IDevID certificate). This approach requires an on-site availability of a local asset or inventory management system performing the authorization decision based on tuple of the certification request and the pledge authentication using the IDevID certificate, to issue a domain specific certificate to the pledge. The EST server (the domain registrar) terminates the security association with the pledge and thus the binding between the certification request and the authentication of the pledge via TLS.

This type of enrollment utilizing an online connection to the PKI is considered as synchronous enrollment.

For certain use cases on-site support of a RA/CA component and/or an asset management is not available and rather provided by an operator's backend and may be provided timely limited or completely through offline interactions. This may be due to higher security requirements for operating the certification authority or for optimization of operation for smaller deployments to avoid the always on-site operation. The authorization of a certification request based on an asset management in this case will not / can not be performed on-site at enrollment time. Enrollment, which cannot be performed in a (timely) consistent fashion is considered as asynchronous enrollment in this document. It requires the support of a store and forward functionality of certification request together with the requester authentication (and identity) information. This enables processing of the request at a later point in time. A similar situation may occur through network segmentation, which is utilized in industrial systems to separate domains with different security needs. Here, a similar requirement arises if the communication channel carrying the requester authentication is terminated before the RA/CA authorization handling of the certification request. If a second communication channel is opened to forward the certification request to the issuing RA/ CA, the requester authentication information needs to be retained and ideally bound to the certification request. This uses case is independent from timely limitations of the first use case. For both cases, it is assumed that the requester authentication information is utilized in the process of authorization of a certification request. There are different options to perform store and forward of certification requests including the requester authentication information:

- o Providing a trusted component (e.g., an LRA) in the target domain, which stores the certification request combined with the requester authentication information (based on the IDevID) and potentially the information about a successful proof of possession (of the corresponding private key) in a way prohibiting changes to the combined information. Note that the assumption is that the information elements may not be cryptographically bound together. Once connectivity to the backend is available, the trusted component forwards the certification request together with the requester information (authentication and proof of possession) to the off-site PKI for further processing. It is assumed that the off-site PKI in this case relies on the local pledge authentication result and thus performs the authorization and issues the requested certificate. In BRSKI the trusted component may be the EST server residing co-located with the registrar in the target domain.

o Utilization of authenticated self-contained objects for the enrollment, binding the certification request and the requester authentication in a cryptographic way. This approach reduces the necessary trust in a domain component to storage and delivery. Unauthorized modifications of the requester information (request and authentication) can be detected during the verification of the authenticated self-contained object.

Focus of this document the support of handling authenticated self-contained objects for bootstrapping. As it is intended to enhance BRSKI it is named BRSKI-AE, where AE stands for asynchronous enrollment. As BRSKI, BRSKI-AE results in the pledge storing an X.509 domain certificate and sufficient information for verifying the domain registrar / proxy identity (LDevID CA Certificate) as well as domain specific X.509 device certificates (LDevID EE certificate).

Based on the proposed approach, a second set of scenarios can be addressed, in which the pledge has either no direct communication path to the domain registrar, e.g., due to missing network connectivity or a different technology stack. In such scenarios the pledge is expected to act as a server rather than a client. The pledge will be triggered to generate request objects to be onboarded in the registrar's domain. For this, an additional component is introduced acting as an agent for the domain registrar (registrar-agent) towards the pledge. This could be a functionality of a commissioning tool or it may be even co-located with the registrar. In contrast to BRSKI the registrar-agent performs the object exchange with the pledge and provides/retrieves data objects to/from the domain registrar. For the interaction with the domain registrar the registrar agent will use existing BRSKI endpoints.

The goal is to enhance BRSKI to be applicable to the additional use cases. This is addressed by

o enhancing the well-known URI approach with an additional path for the utilized enrollment protocol.

o defining a certificate waiting indication and handling, if the certifying component is (temporarily) not available.

o allowing to utilize credentials different from the pledge's IDevID to establish a TLS connection to the domain registrar, which is necessary in case of using a registrar-agent.

o defining the interaction (dta exchange and data objects) between a pledge acting as server an a registrar-agent and the domain registrar.

Note that in contrast to BRSKI, BRSKI-AE assumes support of multiple enrollment protocols on the infrastructure side, allowing the pledge manufacturer to select the most appropriate. Thus, BRSKI-AE can be applied for both, asynchronous and synchronous enrollment.

2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

This document relies on the terminology defined in [RFC8995]. The following terms are defined additionally:

CA: Certification authority, issues certificates.

RA: Registration authority, an optional system component to which a CA delegates certificate management functions such as authorization checks.

LRA: Local registration authority, an optional RA system component with proximity to end entities.

IED: Intelligent Electronic Device (in essence a pledge).

on-site: Describes a component or service or functionality available in the target deployment domain.

off-site: Describes a component or service or functionality available in an operator domain different from the target deployment domain. This may be a central site or a cloud service, to which only a temporary connection is available, or which is in a different administrative domain.

asynchronous communication: Describes a timely interrupted communication between an end entity and a PKI component.

synchronous communication: Describes a timely uninterrupted communication between an end entity and a PKI component.

authenticated self-contained object: Describes an object, which is cryptographically bound to the EE certificate (IDevID certificate or LDEVID certificate) of a pledge. The binding is assumed to be provided through a digital signature of the actual object using the corresponding private key of the EE certificate.

3. Scope of solution

3.1. Supported environment

This solution is intended to be used in domains with limited support of on-site PKI services and comprises use cases in which:

- there is no registration authority available in the target domain. The connectivity to an off-site RA in an operator's network may only be available temporarily. A local store and forward device is used for the communication with the off-site services.

- authoritative actions of a LRA are limited and may not comprise authorization of certification requests of pledges. Final authorization is done at the RA residing in the operator domain.

- the target deployment domain already has an established certificate management approach that shall be reused to (e.g., in brownfield installations).

In addition, the solution is intended to be applicable in domains in which pledges have no direct connection to the domain registrar, but are expected to be managed by the registrar. This can be motivated by pledges featuring a different technology stack or by pledges without an existing connection to the domain registrar during bootstrapping. These pledges are likely to act in a server role. Therefore, the pledge has to offer endpoints on which it can be triggered for the generation of voucher-request objects and certification objects as well as to provide the response objects to the pledge.

3.2. Application Examples

The following examples are intended to motivate the support of different enrollment approaches in general and asynchronous enrollment specifically, by introducing industrial applications cases, which could leverage BRSKI as such but also require support of asynchronous operation as intended with BRSKI-AE.

3.2.1. Rolling stock

Rolling stock or railroad cars contain a variety of sensors, actuators, and controllers, which communicate within the railroad car but also exchange information between railroad cars building a train, or with a backend. These devices are typically unaware of backend connectivity. Managing certificates may be done during maintenance cycles of the railroad car, but can already be prepared during operation. The preparation may comprise the generation of certification requests by the components which are collected and forwarded for processing, once the railroad car is connected to the operator backend. The authorization of the certification request is then done based on the operator's asset/inventory information in the backend.

3.2.2. Building automation

In building automation, a use case can be described by a detached building or the basement of a building equipped with sensor, actuators, and controllers connected, but with only limited or no connection to the centralized building management system. This limited connectivity may be during the installation time but also during operation time. During the installation in the basement, a service technician collects the necessary information from the basement network and provides them to the central building management system, e.g., using a laptop or even a mobile phone to transport the information. This information may comprise parameters and settings required in the operational phase of the sensors/actuators, like a certificate issued by the operator to authenticate against other components and services.

The collected information may be provided by a domain registrar already existing in the installation network. In this case connectivity to the backend PKI may be facilitated by the service technician's laptop. Contrary, the information can also be collected from the pledges directly and provided to a domain registrar deployed in a different network. In this cases connectivity to the domain registrar may be facilitated by the service technician's laptop.

3.2.3. Substation automation

In electrical substation automation a control center typically hosts PKI services to issue certificates for Intelligent Electronic Devices (IED)s operated in a substation. Communication between the substation and control center is done through a proxy/gateway/DMZ, which terminates protocol flows. Note that [NERC-CIP-005-5] requires inspection of protocols at the boundary of a security perimeter (the substation in this case). In addition, security management in substation automation assumes central support of different enrollment protocols to facilitate the capabilities of IEDs from different vendors. The IEC standard IEC62351-9 [IEC-62351-9] specifies the mandatory support of two enrollment protocols, SCEP [RFC8894] and EST [RFC7030] for the infrastructure side, while the IED must only support one of the two.

3.2.4. Electric vehicle charging infrastructure

For the electric vehicle charging infrastructure protocols have been defined for the interaction between the electric vehicle (EV) and the charging point (e.g., ISO 15118-2 [ISO-IEC-15118-2]) as well as between the charging point and the charging point operator (e.g. OCPP [OCPP]). Depending on the authentication model, unilateral or mutual authentication is required. In both cases the charging point uses an X.509 certificate to authenticate itself in the context of a TLS connection between the EV and the charging point. The management of this certificate depends (beyond others) on the selected backend connectivity protocol. Specifically, in case of OCPP it is intended as single communication protocol between the charging point and the backend carrying all information to control the charging operations and maintain the charging point itself. This means that the certificate management is intended to be handled in-band of OCPP. This requires to be able to encapsulate the certificate management exchanges in a transport independent way. Authenticated self-containment will ease this by allowing the transport without a separate enrollment protocol. This provides a binding of the exchanges to the identity of the communicating endpoints.

3.2.5. Infrastructure isolation policy

This refers to any case in which network infrastructure is normally isolated from the Internet as a matter of policy, most likely for security reasons. In such a case, limited access to external PKI resources will be allowed in carefully controlled short periods of time, for example when a batch of new devices are deployed, but impossible at other times.

3.2.6. Less operational security in the target domain

The registration point performing the authorization of a certificate request is a critical PKI component and therefore implicates higher operational security than other components utilizing the issued certificates for their security features. CAs may also demand higher security in the registration procedures. Especially the CA/Browser forum currently increases the security requirements in the certificate issuance procedures for publicly trusted certificates. There may be the situation that the target domain does not offer enough security to operate a registration point and therefore wants to transfer this service to a backend that offers a higher level of operational security.

4. Requirement discussion and mapping to solution elements

For the requirements discussion it is assumed that the domain registrar receiving a certification request as authenticated self-contained object is not the authorization point for this certification request. If the domain registrar is the authorization point and the pledge has a direct connection to the registrar, BRSKI can be used directly. Note that BRSKI-AE could also be used in this case.

Based on the intended target environment described in Section 3.1 and the motivated application examples described in Section 3.2 the following base requirements are derived to support authenticated self-contained objects as container carrying the certification request and further information to support asynchronous operation.

At least the following properties are required:

- Proof of Possession: proves to possess and control the private key corresponding to the public key contained in the certification request, typically by adding a signature using the private key.

- Proof of Identity: provides data-origin authentication of a data object, e.g., a certificate request, utilizing an existing IDevID. Certificate updates may utilize the certificate that is to be updated.

Solution examples (not complete) based on existing technology are provided with the focus on existing IETF documents:

- Certification request objects: Certification requests are structures protecting only the integrity of the contained data providing a proof-of-private-key-possession for locally generated key pairs. Examples for certification requests are:

  * PKCS#10 [RFC2986]: Defines a structure for a certification request. The structure is signed to ensure integrity protection and proof of possession of the private key of the requester that corresponds to the contained public key.

  * CRMF [RFC4211]: Defines a structure for the certification request message. The structure supports integrity protection and proof of possession, through a signature generated over parts of the structure by using the private key corresponding to the contained public key. CRMF also supports further proof-of-possession methods for key pairs not capable to be used for signing.

Note that the integrity of the certification request is bound to the public key contained in the certification request by performing the signature operation with the corresponding private key. In the considered application examples, this is not sufficient to provide data origin authentication and needs to be bound to the existing credential of the pledge (IDevID) additionally. This binding supports the authorization decision for the certification request through the provisioning of a proof of identity. The binding of data origin authentication to the certification request may be delegated to the protocol used for certificate management.

- Proof of Identity options: The certification request should be bound to an existing credential (here IDevID) to enable a proof of identity and based on it an authorization of the certification request. The binding may be realized through security options in an underlying transport protocol if the authorization of the certification request is done at the next communication hop. Alternatively, this binding can be done by a wrapping signature employing an existing credential (initial: IDevID, renewal: LDevID). This requirement is addressed by existing enrollment protocols in different ways, for instance:

  * EST [RFC7030]: Utilizes PKCS#10 to encode the certification request. The Certificate Signing Request (CSR) may contain a binding to the underlying TLS by including the tls-unique value in the self-signed CSR structure. The tls-unique value is one result of the TLS handshake. As the TLS handshake is performed mutually authenticated and the pledge utilized its IDevID for it, the proof of identity can be provided by the binding to the TLS session. This is supported in EST using the simpleenroll endpoint. To avoid the binding to the underlying authentication in the transport layer, EST offers the support of a wrapping the CSR with an existing certificate by using Full PKI Request messages.

  * SCEP [RFC8894]: Provides the option to utilize either an existing secret (password) or an existing certificate to protect the CSR based on SCEP Secure Message Objects using CMS wrapping ([RFC5652]). Note that the wrapping using an existing IDevID credential in SCEP is referred to as renewal. SCEP therefore does not rely on the security of an underlying transport.

  * CMP [RFC4210] Provides the option to utilize either an existing secret (password) or an existing certificate to protect the PKIMessage containing the certification request. The certification request is encoded utilizing CRMF. PKCS#10 is

optionally supported. The proof of identity of the PKIMessage containing the certification request can be achieved by using IDevID credentials to a PKIProtection carrying the actual signature value. CMP therefore does not rely on the security of an underlying transport protocol.

* CMC [RFC5272] Provides the option to utilize either an existing secret (password) or an existing certificate to protect the certification request (either in CRMF or PKCS#10) based on CMS [RFC5652]). Here a FullCMCRequest can be used, which allows signing with an existing IDevID credential to provide a proof of identity. CMC therefore does not rely on the security of an underlying transport.

Note that besides the already existing enrollment protocols there is ongoing work in the ACE WG to define an encapsulation of EST messages in OSCORE to result in a TLS independent way of protecting EST. This approach [I-D.selander-ace-coap-est-oscore] may be considered as further variant.

5. Architectural Overview and Communication Exchanges

To support asynchronous enrollment, the base system architecture defined in BRSKI [RFC8995] is enhanced to facilitate the two target use cases.

o Use case 1 (Pledge-initiator-mode): the pledge requests certificates from a PKI operated off-site via the domain registrar. The communication model follows the BRSKI model in which the pledge initiates the communication.

o Use case 2 (Pledge-responder-mode): allows delegated bootstrapping using a registrar-agent instead a direct connection from the pledge to the domain registrar. The communication model between registrar-agent and pledge assumes that the pledge is acting as server and responds to requests.

Both use cases are described in the next subsections. They utilize the existing BRSKI architecture elements as much as possible. Necessary enhancements to support authenticated self-contained objects for certificate enrollment are kept on a minimum to ensure reuse of already defined architecture elements and interactions.

For the authenticated self-contained objects used for the certification request, BRSKI-AE relies on the defined message wrapping mechanisms of the enrollment protocols stated in Section 4 above.

5.1. Use Case 1 (pledge-initiator-mode): Support of off-site PKI service

One assumption of BRSKI-AE is that the authorization of a certification request is performed based on an authenticated self-contained object, binding the certification request to the authentication using the IDevID. This supports interaction with off-site or off-line PKI (RA/CA) components. In addition, the authorization of the certification request may not be done by the domain registrar but by a PKI residing in the backend of the domain operator (off-site) as described in Section 3.1. Also, the certification request may be piggybacked by another protocol. This leads to changes in the placement or enhancements of the logical elements as shown in Figure 1.

```
                                              +------------------------+
      +--------------Drop Ship--------------->| Vendor Service         |
      |                                       +------------------------+
      |                                       | M anufacturer|         |
      |                                       | A uthorized  |Ownership|
      |                                       | S igning     |Tracker  |
      |                                       | A uthority   |         |
      |                                       +--------------+---------+
      |                                                      ^
      |                                                      |
      V                                                      |
   +--------+     .........................................  |
   |        |     .                                       .  | BRSKI-
   |        |     .  +------------+       +------------+  .  | MASA
   | Pledge |     .  |   Join     |       | Domain     <-----+
   |        |     .  |   Proxy    |       | Registrar/ |  .
   |        |<-------->............<-------> Enrollment |  .
   |        |     .  |        BRSKI-AE    | Proxy      |  .
   | IDevID |     .  |            |       +------^-----+  .
   |        |     .  +------------+              |        .
   |        |     .                              |        .
   +--------+     ...............................|.........
                   "on-site domain" components   |
                                                 |e.g., RFC 7030,
                                                 |      RFC 4210, ...
    .............................................|.....................
    . +---------------------------+     +--------v------------------+ .
    . | Public Key Infrastructure |<----+ PKI RA                    | .
    . | PKI CA                    |---->+                           | .
    . +---------------------------+     +---------------------------+ .
    ...................................................................
            "off-site domain" components
```

Figure 1: Architecture overview using off-site PKI components

The architecture overview in Figure 1 utilizes the same logical elements as BRSKI but with a different placement in the deployment architecture for some of the elements. The main difference is the placement of the PKI RA/CA component, which is performing the authorization decision for the certification request message. It is placed in the off-site domain of the operator (not the deployment site directly), which may have no or only temporary connectivity to the deployment or on-site domain of the pledge. This is to underline the authorization decision for the certification request in the backend rather than on-site. The following list describes the components in the target domain:

o Join Proxy: same functionality as described in BRSKI.

o Domain Registrar / Enrollment Proxy: In general the domain registrar proxy has a similar functionality regarding the imprinting of the pledge in the deployment domain to facilitate the communication of the pledge with the MASA and the PKI. Different is the authorization of the certification request. BRSKI-AE allows to perform this in the operator's backend (off-site), and not directly at the domain registrar.

  * Voucher exchange: The voucher exchange with the MASA via the domain registrar is performed as described in BRSKI [RFC8995].

  * Certificate enrollment: For the pledge enrollment the domain registrar in the deployment domain supports the adoption of the pledge in the domain based on the voucher request. Nevertheless, it may not have sufficient information for authorizing the certification request. If the authorization of the certification request is done in the off-site domain, the domain registrar forwards the certification request to the RA to perform the authorization. Note that this requires, that the certification request object is enhanced with a proof-of-identity to allow the authorization based on the bound identity information of the pledge. As stated above, this can be done by an additional signature using the IDevID. The domain registrar here acts as an enrollment proxy or local registration authority. It is also able to handle the case having no connection temporarily to an off-site PKI, by storing the authenticated certification request and forwarding it to the RA upon reestablished connectivity. As authenticated self-contained objects are used, it requires an enhancement of the domain registrar. This is done by supporting alternative enrollment approaches (protocol options, protocols, encoding) by enhancing the addressing scheme to communicate with the domain registrar (see Section 5.1.5).

The following list describes the vendor related components/service outside the deployment domain:

o MASA: general functionality as described in [RFC8995]. Assumption is that the interaction with the MASA may be synchronous (voucher request with nonce) or asynchronous (voucher request without nonce).

o Ownership tracker: as defined in [RFC8995].

The following list describes the operator related components/service operated in the backend:

- o PKI RA: Performs certificate management functions (validation of certification requests, interaction with inventory/asset management for authorization of certification requests, etc.) for issuing, updating, and revoking certificates for a domain as a centralized infrastructure for the domain operator. The inventory (asset) management may be a separate component or integrated into the RA directly.

- o PKI CA: Performs certificate generation by signing the certificate structure provided in the certification request.

Based on BRSKI and the architectural changes the original protocol flow is divided into three phases showing commonalities and differences to the original approach as depicted in the following.

- o Discovery phase (same as BRSKI)

- o Voucher exchange with deployment domain registrar (same as BRSKI).

- o Enrollment phase (changed to support the application of authenticated self-contained objects).

5.1.1. Behavior of a pledge

The behavior of a pledge as described in [RFC8995] is kept with one exception. After finishing the imprinting phase (4) the enrollment phase (5) is performed with a method supporting authenticated self-contained objects. Using EST with simple-enroll cannot be applied here, as it binds the pledge authentication with the existing IDevID to the transport channel (TLS) rather than to the certification request object directly. This authentication in the transport layer is not visible / verifiable at the authorization point in the off-site domain. Section 7 discusses potential enrollment protocols and options applicable.

5.1.2. Pledge - Registrar discovery and voucher exchange

The discovery phase is applied as specified in [RFC8995].

5.1.3. Registrar - MASA voucher exchange

The voucher exchange is performed as specified in [RFC8995].

5.1.4. Pledge - Registrar - RA/CA certificate enrollment

As stated in Section 4 the enrollment shall be performed using an authenticated self-contained object providing proof of possession and proof of identity.

```
   +--------+         +----------+    +------------+     +------------+
   | Pledge |         | Circuit  |    | Domain     |     | Operator   |
   |        |         | Join     |    | Registrar  |     | RA/CA      |
   |        |         | Proxy    |    |  (JRC)     |     | (OPKI)     |
   +--------+         +----------+    +------------+     +------------+
     /-->                                      |                     |
   [Request of CA Certificates]                |                     |
     |---------- CA Certs Request ------------>|                     |
     |              [if connection to operator domain is available]  |
     |                                         |-Request CA Certs -> |
     |                                         |<- CA Certs Response |
     |<-------- CA Certs Response--------------|                     |
     /-->                                      |                     |
   [Request of Certificate Attributes to be included]                |
     |---------- Attribute Request ----------->|                     |
     |              [if connection to operator domain is available]  |
     |                                         |Attribute Request -> |
     |                                         |<-Attribute Response |
     |<--------- Attribute Response -----------|                     |
     /-->                                      |                     |
   [Certification request]                     |                     |
     |-------------- Cert Request ------------>|                     |
     |              [if connection to operator domain is available]  |
     |                                         |--- Cert Request --> |
     |                                         |                     |
   [Optional Certificate waiting indication]   |                     |
     /-->                                      |                     |
     |<----- Cert Response (with Waiting) -----|                     |
     |-- Cert Polling (with orig request ID) ->|                     |
     |                                         |                     |
     /-->                                      |                     |
     |                                         |<-- Cert Response -- |
     |                                         |                     |
     |<-- Cert Response (with Certificate) ----|                     |
     /-->                                      |                     |
   [Certificate confirmation]                  |                     |
     |-------------- Cert Confirm ------------>|                     |
     |                                        /-->                   |
     |                                         |[optional]           |
     |                                         |--- Cert Confirm --> |
     |                                         |<-- PKI Confirm ---- |
     |<------------- PKI/Registrar Confirm ----|                     |
```

Figure 2: Certificate enrollment

The following list provides an abstract description of the flow depicted in Figure 2.

- o CA Cert Request: The pledge SHOULD request the full distribution of CA Certificates. This ensures that the pledge has the complete set of current CA certificates beyond the pinned-domain-cert (which may be the domain registrar certificate contained in the voucher).

- o CA Cert Response: Contains at least one CA certificate of the issuing CA.

- o Attribute Request: Typically, the automated bootstrapping occurs without local administrative configuration of the pledge. Nevertheless, there are cases, in which the pledge may also include additional attributes specific to the deployment domain into the certification request. To get these attributes in advance, the attribute request SHOULD be used.

- o Attribute Response: Contains the attributes to be included in the certification request message.

- o Cert Request: Depending on the utilized enrollment protocol, this certification request contains the authenticated self-contained object ensuring both, proof-of-possession of the corresponding private key and proof-of-identity of the requester.

- o Cert Response: certification response message containing the requested certificate and potentially further information like certificates of intermediary CAs on the certification path.

- o Cert Waiting: waiting indication for the pledge to retry after a given time. For this a request identifier is necessary. This request identifier may be either part of the enrollment protocol or build based on the certification request.

- o Cert Polling: querying the registrar, if the certificate request was already processed; can be answered either with another Cert Waiting, or a Cert Response.

- o Cert Confirm: confirmation message from pledge after receiving and verifying the certificate.

- o PKI/Registrar Confirm: confirmation message from PKI/registrar about reception of the pledge’s certificate confirmation.

The generic messages described above can implemented using various protocols implementing authenticated self-contained objects, as described in Section 4. Examples are available in Section 7.

5.1.5. Addressing Scheme Enhancements

BRSKI-AE provides enhancements to the addressing scheme defined in [RFC8995] to accommodate the additional handling of authenticated self-contained objects for the certification request. As this is supported by different enrollment protocols, they can be directly employed (see also Section 7).

The addressing scheme in BRSKI for client certificate request and CA certificate distribution function during the enrollment uses the definition from EST [RFC7030], here on the example on simple enroll: "/.well-known/est/simpleenroll" This approach is generalized to the following notation: "/.well-known/enrollment-protocol/request" in which enrollment-protocol may be an already existing protocol or a newly defined approach. Note that enrollment is considered here as a sequence of at least a certification request and a certification response. In case of existing enrollment protocols the following notation is used proving compatibility to BRSKI:

- enrollment-protocol: references either EST [RFC7030] as in BRSKI or CMP, CMC, SCEP, or newly defined approaches as alternatives. Note: additional endpoints (well-known URI) at the registrar may need to be defined by the utilized enrollment protocol.

- request: depending on the utilized enrollment protocol, the request describes the required operation at the registrar side. Enrollment protocols are expected to define the request endpoints as done by existing protocols (see also Section 7).

5.2. Use Case 2 (pledge-responder-mode): Registrar-agent communication with Pledges

To support mutual trust establishment of pledges, not directly connected to the domain registrar. It relies on the exchange of authenticated self-contained objects (the voucher request/response objects as known from BRSKI and the enrollment request/response objects as introduced by BRSKI-AE). This approach has also been applied also for the use case 1. This allows independence of a potential protection provided by the used transport protocol.

In contrast to BRSKI, the object exchanges performed with the help of a registrar-agent component, supporting the interaction of the pledge with the domain registrar. It may be an integrated functionality of a commissioning tool. This leads to enhancements of the logical elements in the BRSKI architecture as shown in Figure 3. The registrar-agent interacts with the pledge to acquire and to supply the required data objects for bootstrapping, which are also exchanged between the registrar-agent and the domain registrar. Moreover, the addition of the registrar-agent also influences the sequences for the data exchange between the pledge and the domain registrar described in [RFC8995]. The general goal for the registrar-agent application is the reuse of already defined endpoints of the domain registrar side. The functionality of the already existing registrar endpoints may need small enhancements.

```
                                             +------------------------+
      +--------------Drop Ship---------------| Vendor Service         |
      |                                      +------------------------+
      |                                      | M anufacturer|         |
      |                                      | A uthorized  |Ownership|
      |                                      | S igning     |Tracker  |
      |                                      | A uthority   |         |
      |                                      +--------------+---------+
      |                                                     ^
      |                                                     |   BRSKI-
      V                                                     |    MASA
   +-------+     +---------+   .............................|.........
   |       |     |         |   .                            |        .
   |       |     |         |   .  +-----------+       +-----v-----+  .
   |       |     |Registrar|   .  |           |       |           |  .
   |Pledge |     |Agent    |   .  |    Join   |       | Domain    |  .
   |       |     |         |   .  |    Proxy  |       | Registrar |  .
   |       <----->.........<------>...........<-------> (PKI RA)  |  .
   |       |     |         |   .  |       BRSKI-AE    |           |  .
   |       |     |         |   .  |           |       +-----+-----+  .
   |IDevID |     | LDevID  |   .  +-----------+             |        .
   |       |     |         |   .       +-----------------+-----+  .
   +-------+     +---------+   .       | Key Infrastructure    |  .
                               .       | (e.g., PKI Certificate |  .
                               .       |       Authority)       |  .
                               .       +------------------------+  .
                               .......................................
                                       "Domain" components
```

Figure 3: Architecture overview using registrar-agent

The architecture overview in Figure 3 utilizes the same logical components as BRSKI with the registrar-agent component in addition.

For authentication towards the domain registrar, the registrar-agent uses its LDevID. The provisioning of the registrar-agent LDevID may be done by a separate BRSKI run or other means in advance. It is recommended to use short lived registrar-agent LDevIDs in the range of days or weeks.

If a registrar detects a request originates from a registrar-agent it is able to switch the operational mode from BRSKI to BRSKI-AE.

In addition, the domain registrar may authenticate the user operating the registrar-agent to perform additional authorization of a pledge enrollment action. Examples for such user level authentication are the application of HTTP authentication or the usage of authorization tokens or other. This is out of scope of this document.

The following list describes the components in a (customer) site domain:

o Pledge: The pledge is expected to respond with the necessary data objects for bootstrapping to the registrar-agent. The transport protocol used between the pledge and the registrar-agent is assumed to be HTTP in the context of this document. Other transport protocols may be used but are out of scope of this document. As the pledge is acting as a server during bootstrapping it leads to some differences to BRSKI:

  * Discovery of the domain registrar by the pledge is not needed as the pledge will be triggered by the registrar-agent.

  * Discovery of the pledge by the registrar-agent must be possible.

  * As the registrar-agent must be able to request data objects for bootstrapping of the pledge, the pledge must offer corresponding endpoints.

  * The registrar-agent may provide additional data to the pledge, in the context of the triggering request.

  * Order of exchanges in the call flow may be different as the registrar-agent collects both objects, pledge-voucher-request objects and pledge-enrollment-request objects, at once and provides them to the registrar. This approach may also be used to perform a bulk bootstrapping of several devices.

  * The data objects utilized for the data exchange between the pledge and the registrar are self-contained authenticated objects (signature-wrapped objects) as in use case 1 Section 5.1.

o Registrar-agent: provides a communication path to exchange data objects between the pledge and the domain registrar. The registrar-agent facilitates situations, in which the domain registrar is not directly reachable by the pledge, either due to a different technology stack or due to missing connectivity. The registrar-agent triggers the pledge to create bootstrapping information such as voucher request objects and enrollment request objects from one or multiple pledges at once and performs a bulk bootstrapping based on the collected data. The registrar-agent is expected to possess information of the domain registrar, either by configuration or by using the discovery mechanism defined in [RFC8995]. There is no trust assumption between the pledge and the registrar-agent as only authenticated self-contained objects are applied, which are transported via the registrar-agent and provided either by the pledge or the registrar. The trust assumption between the registrar-agent and the registrar bases on an own LDevID of the registrar-agent, acting as registrar component. This allows the registrar-agent to authenticate towards the registrar. The registrar can utilize this authentication to distinguish communication with a pledge from a registrar-agent based on the exchanged objects.

- Join Proxy: same functionality as described in [RFC8995]. Note that it may be used by the registrar-agent instead of the pledge to find the registrar, if not configured.

- Domain Registrar: In general the domain registrar fulfills the same functionality regarding the bootstrapping of the pledge in a (customer) site domain by facilitating the communication of the pledge with the MASA service and the domain PKI service. In contrast to [RFC8995], the domain registrar does not interact with a pledge directly but through the registrar-agent. The registrar detects if the bootstrapping is performed by the pledge directly or by the registrar-agent. The manufacturer provided components/ services (MASA and Ownership tracker) are used as defined in [RFC8995]. For issuing a voucher, the MASA may perform additional checks on voucher-request objects, to issue a voucher indicating agent-proximity instead of registrar-proximity.

[RFC Editor: please delete] /*

Open Issues: The voucher defined in [RFC8366] defines the leaf assertion as enum, which cannot be extended. To define an additional assertion RFC 8366 may be revised. */

"Agent-proximity" is a weaker assertion then "proximity". In case of "agent-proximity" it is a statement, that the proximity-registrar-certificate was provided via the registrar-agent and not directly. This can be verified by the registrar and also by the MASA through voucher-request processing. Note that at the time of creating the voucher-request, the pledge cannot verify the LDevID(Reg) EE certificate and has no proof-of-possession of the corresponding private key for the certificate. Trust handover to the domain is established via the "pinned-domain-certificate" in the voucher.

In contrast, "proximity" provides a statement, that the pledge was in direct contact with the registrar and was able to verify proof-of-possession of the private key in the context of the TLS handshake. The provisionally accepted LDevID(Reg) EE certificate can be verified after the voucher has been processed by the pledge.

5.2.1. Behavior of a pledge in pledge-responder-mode

In contrast to use case 1 Section 5.1 the pledge acts as a server component if data is triggered by the registrar-agent for the generation of pledge-voucher-request and pledge-enrollment-request objects as well as for the processing of the response objects and the generation of status information. Due to the use of the registrar-agent, the interaction with the domain registrar is changed as shown in Figure 5. To enable interaction with the registrar-agent, the pledge provides endpoints using the BRSKI interface based on the "/.well-known/brski" URI tree. The following endpoints are defined for the pledge in this document:

- /.well-known/brski/pledge-voucher-request: trigger pledge to create voucher request. It returns the pledge-voucher-request.

- /.well-known/brski/pledge-enrollment-request: trigger pledge to create enrollment request. it returns the pledge-enrollment-request.

- /.well-known/brski/pledge-voucher: supply MASA provided voucher to pledge. It returns the pledge-voucher-status.

- /.well-known/brski/pledge-enrollment: supply enroll response (certificate) to pledge. It returns the pledge-enrollment-status.

- /.well-known/brski/pledge-CACerts: supply CACerts to pledge (optional).

5.2.2. Behavior of a registrar-agent

The registrar-agent is a new component in the BRSKI context. It provides connectivity between the pledge and the domain registrar and reuses the endpoints of the domain registrar side already specified in [RFC8995]. It facilitates the exchange of data objects between the pledge and the domain registrar, which are the voucher request/response objects, the enrollment request/response objects, as well as related status objects. For the communication the registrar-agent utilizes communication endpoints provided by the pledge. The transport in this specification is based on HTTP but may also be done using other transport mechanisms. This new component changes the general interaction between the pledge and the domain registrar as shown in Figure 9.

The registrar-agent is expected to already possess an LDevID(RegAgt) to authenticate towards the domain registrar. The registrar-agent will use this LDevID(RegAgt) when establishing the TLS session with the domain registrar in the context of for TLS client-side authentication. The LDevID(RegAgt) certificate MUST include a SubjectKeyIdentifier (SKID), which is used as reference in the context of an agent-signed-data object. Note that this is an additional requirement for issuing the certificate, as [IEEE-802.1AR] only requires the SKID to be included for intermediate CA certificates. In the specific application of BRSKI-AE, it is used in favor of a certificate fingerprint to avoid additional computations.

Using an LDevID for TLS client-side authentication is a deviation from [RFC8995], in which the pledge's IDevID credential is used to perform TLS client authentication. The use of the LDevID(RegAgt) allows the domain registrar to distinguish, if bootstrapping is initiated from a pledge or from a registrar-agent and adopt the internal handling accordingly. As BRSKI-AE uses authenticated self-contained data objects between the pledge and the domain registrar, the binding of the pledge identity to the request object is provided by the data object signature employing the pledge's IDevID. The objects exchanged between the pledge and the domain registrar used in the context of this specifications are JOSE objects

In addition to the LDevID(RegAgt), the registrar-agent is provided with the product-serial-numbers of the pledges to be bootstrapped. This is necessary to allow the discovery of pledges by the registrar-agent using mDNS. The list may be provided by administrative means or the registrar agent may get the information via an interaction with the pledge, like scanning of product-serial-number information using a QR code or similar.

According to [RFC8995] section 5.3, the domain registrar performs the pledge authorization for bootstrapping within his domain based on the pledge voucher-request object.

The following information is therefore available at the registrar-agent:

o LDevID(RegAgt): own operational key pair.

o LDevID(reg) certificate: certificate of the domain registrar.

o Serial-number(s): product-serial-number(s) of pledge(s) to be bootstrapped.

5.2.2.1. Registrar discovery by the registrar-agent

The discovery of the domain registrar may be done as specified in [RFC8995] with the deviation that it is done between the registrar-agent and the domain registrar. Alternatively, the registrar-agent may be configured with the address of the domain registrar and the certificate of the domain registrar.

5.2.2.2. Pledge discovery by the registrar-agent

The discovery of the pledge by registrar-agent should be done by using DNS-based Service Discovery [RFC6763] over Multicast DNS [RFC6762] to discover the pledge at "product-serial-number.brski-pledge._tcp.local." The pledge constructs a local host name based on device local information (product-serial-number), which results in "product-serial-number.brski-pledge._tcp.local.". It can then be discovered by the registrar-agent via mDNS. Note that other mechanisms for discovery may be used.

The registrar-agent is able to build the same information based on the provided list of product-serial-number.

5.2.3. Bootstrapping objects and corresponding exchanges

The interaction of the pledge with the registrar-agent may be accomplished using different transport means (protocols and or network technologies). For this document the usage of HTTP is targeted as in BRSKI. Alternatives may be CoAP, Bluetooth Low Energy (BLE), or Nearfield Communication (NFC). This requires independence of the exchanged data objects between the pledge and the registrar from transport security. Therefore, authenticated self-contained objects (here: signature-wrapped objects) are applied in the data exchange between the pledge and the registrar.

The registrar-agent provides the domain-registrar certificate (LDevID(Reg) EE certificate) to the pledge to be included into the "agent-provided-proximity-registrar-certificate" leaf in the pledge-voucher-request object. This enables the registrar to verify, that it is the target registrar for handling the request. The registrar certificate may be configured at the registrar-agent or may be fetched by the registrar-agent based on a prior TLS connection establishment with the domain registrar. In addition, the registrar-agent provides agent-signed-data containing the product-serial-number in the body, signed with the LDevID(RegAgt). This enables the registrar to verify and log, which registrar-agent was in contact with the pledge. Optionally the registrar-agent may provide its LDevID(RegAgt) certificate to the pledge for inclusion into the pledge-voucher-request as "agent-sign-cert" leaf. Note that this may be omitted in constraint environments to safe bandwidth between the registrar-agent and the pledge. If not contained, the registrar-agent MUST fetch the LDevID(RegAgt) certificate based on the SubjectKeyIdentifier (SKID) in the header of the agent-signed-data. The registrar may include the LDevID(RegAgt) certificate information into the registrar-voucher-request.

The MASA in turn verifies the LDevID(Reg) certificate is included in the pledge-voucher-request (prior-signed-voucher-request) in the "agent-provided-proximity-registrar-certificate" leaf and may assert in the voucher "verified" or "logged" instead of "proximity", as there is no direct connection between the pledge and the registrar. If the LDevID(RegAgt) certificate is included contained in the "agent-sign-cert" leave of the registrar-voucher-request, the MASA can verify the LDevID(RegAgt) certificate and the signature of the registrar-agent in the agent-signed-data provided in the prior-signed-voucher-request. If both can be verified successfully, the MASA can assert "agent-proximity" in the voucher. Otherwise, it may assert "verified" or "logged". The voucher can then be supplied via the registrar to the registrar-agent.

Figure 4 provides an overview of the exchanges detailed in the following sub sections.

```
+--------+  +-----------+    +-----------+   +---------+  +---------+
| Pledge |  | Registrar |    | Domain    |   | Domain  |  | Vendor  |
|        |  | Agent     |    | Registrar |   | CA      |  | Service |
|        |  | (RegAgt)  |    |  (JRC)    |   |         |  | (MASA)  |
+--------+  +-----------+    +-----------+   +---------+  +---------+
     |              |                  |              |   Internet |
[discovery of pledge]
     | mDNS query   |                  |              |            |
     |<-------------|                  |              |            |
     |------------->|                  |              |            |
     |              |                  |              |            |
[trigger pledge-voucher-request and
 pledge-enrollment-request generation]
     |<- vTrigger --|                  |              |            |
     |-Voucher-Req->|                  |              |            |
     |              |                  |              |            |
     |<- eTrigger --|                  |              |            |
     |- Enroll-Req->|                  |              |            |
     ~              ~                  ~              ~            ~
[provide pledge-voucher-request to infrastructure]
     |              |<------ TLS ----->|              |            |
```

```
        |              |-- Voucher-Req -->|              |            |
        |              |          [accept device?]       |            |
        |              |          [contact vendor]       |            |
        |              |                  |------- Voucher-Req ------>|
        |              |                  |           [extract DomainID]
        |              |                  |           [update audit log]
        |              |                  |<-------- Voucher ---------|
        |              |<---- Voucher ----|              |            |
        |              |                  |              |            |
   [provide pledge enrollment request to infrastructure]
        |              |-- Enroll-Req --->|              |            |
        |              |                  |- Cert-Req -->|            |
        |              |                  |<-Certificate-|            |
        |              |<-- Enroll-Resp --|              |            |
        ~              ~                  ~              ~            ~

   [provide voucher and certificate
    to pledge and collect status info]
        |<-- Voucher --|                  |              |            |
        |-- vStatus -->|                  |              |            |
        |<-Enroll-Resp-|                  |              |            |
        |-- eStatus -->|                  |              |            |
        ~              ~                  ~              ~            ~

   [provide voucher-status and enrollment status to registrar]
        |              |<------ TLS ----->|              |            |
        |              |----  vStatus --->|              |            |
        |              |                  |-- req. device audit log ->|
        |              |                  |<---- device audit log ----|
        |              |           [verify audit log]
        |              |                  |              |            |
        |              |----  eStatus --->|              |            |
        |              |                  |              |            |
```

Figure 4: Overview pledge-responder-mode exchanges

The following sub sections split the interactions between the different components into:

o Request objects acquisition targets exchanges and objects between the registrar-agent and the pledge.

o Request handling targets exchanges and objects between the registrar-agent and the registrar and also the interaction of the registrar with the MASA and the domain CA.

o Response object supply targets the exchanges and objects between the registrar-agent and the pledge including the status objects.

- o Status handling addresses the exchanges between the registrar-agent and the registrar.

5.2.3.1. Request objects acquisition (registrar-agent - pledge)

The following description assumes that the registrar-agent already discovered the pledge. This may be done as described in Section 5.2.2.2 based on mDNS.

The focus is on the exchange of signature-wrapped objects using endpoints defined for the pledge in Section 5.2.1.

Preconditions:

- o Pledge: possesses IDevID
- o Registrar-agent: possesses IDevID CA certificate and an own LDevID(RegAgt) EE credential for the registrar domain. In addition, the registrar-agent can be configured with the product-serial-number(s) of the pledge(s) to be bootstrapped. Note that the product-serial-number may have been used during the pledge discovery already.
- o Registrar: possesses IDevID CA certificate and an own LDevID/Reg) credential.
- o MASA: possesses own credentials (voucher signing key, TLS server certificate) as well as IDevID CA certificate of pledge vendor / manufacturer and site-specific LDevID CA certificate.

```
+--------+                             +-----------+
| Pledge |                             | Registrar |
|        |                             | Agent     |
|        |                             | (RegAgt)  |
+--------+                             +-----------+
    |                                        |-create
    |                                        | agent-signed-data
    |<--- trigger pledge-voucher-request ----|
    |-agent-provided-proximity-registrar-cert|
    |-agent-signed-data                      |
    |-agent-sign-cert (optional)             |
    |                                        |
    |----- pledge-voucher-request ---------->|-store
    |                                        | pledge-voucher-request
    |<----- trigger enrollment request ------|
    |       (empty)                          |
    |                                        |
    |------ pledge-enrollment-request ------>|-store
    |                                        | pledge-enrollment-req.
```

Figure 5: Request collection (registrar-agent - pledge)

Triggering the pledge to create the pledge-voucher-request is done using HTTPS POST on the defined pledge endpoint "/.well-known/brski/pledge-voucher-request".

The registrar-agent pledge-voucher-request Content-Type header is:

application/json: defines a JSON document to provide three parameter:

o agent-provided-proximity-registrar-cert: base64-encoded LDevID(Reg) TLS EE certificate.

o agent-sign-cert: base64-encoded LDevID(RegAgt) signing certificate (optional).

o agent-signed-data: base64-encoded JWS-object.

Note that optionally including the agent-sign-cert enables the pledge to verify at least the signature of the agent-signed-data. It may not verify the agent-sign-cert itself due to missing issuing CA information.

The agent-signed-data is a JOSE object and contains the following information:

The header of the agent-signed-data contains:

o alg: algorithm used for creating the object signature.

o kid: contains the base64-encoded SubjectKeyIdentifier of the LDevID(RegAgt) certificate.

The body of the agent-signed-data contains an ietf-voucher-request:agent-signed-data element (defined in Section 6):

o created-on: MUST contain the creation date and time in yang:date-and-time format.

o serial-number: MUST contain the product-serial-number as type string as defined in [RFC8995], section 2.3.1. The serial-number corresponds with the product-serial-number contained in the X520SerialNumber field of the IDevID certificate of the pledge.


```
{
    "alg": "ES256",
    "kid": "base64encodedvalue=="
}
{
  "ietf-voucher-request-trigger:agent-signed-data": {
    "created-on": "2021-04-16T00:00:01.000Z",
    "serial-number": "callee4711"
  }
}
{
    SIGNATURE
}
```


Figure 6: Example of agent-signed-data

Upon receiving the voucher-request trigger, the pledge SHOULD construct the body of the pledge-voucher-request object as defined in [RFC8995]. This object becomes a JSON-in-JWS object as defined in [I-D.richardson-anima-jose-voucher].

The header of the pledge-voucher-request SHALL contain the following parameter as defined in [RFC7515]:

o alg: algorithm used for creating the object signature.

o x5c: contains the base64-encoded pledge IDevID certificate.

The body of the pledge-voucher-request object MUST contain the following parameter as part of the ietf-voucher-request:voucher as defined in [RFC8995]:

- o created-on: contains the current date and time in yang:date-and-time format.

- o nonce: contains a cryptographically strong random or pseudo-random number.

- o serial-number: contains the base64-encoded pledge product-serial-number.

- o assertion: contains the requested voucher assertion.

The ietf-voucher-request:voucher is enhanced with additional parameters:

- o agent-provided-proximity-registrar-cert: MUST be included and contains the base64-encoded LDevID(Reg) EE certificate (provided as trigger parameter by the registrar-agent).

- o agent-signed-data: MUST contain the base64-encoded agent-signed-data (as defined in Figure 6) and provided as trigger parameter.

- o agent-sign-cert: May contain the base64-encoded LDevID(RegAgt) EE certificate if provided as trigger parameter.

The enhancements of the YANG module for the ietf-voucher-request with these new leafs are defined in Section 6.

The object is signed using the pledges IDevID credential contained as x5c parameter of the JOSE header.

```
{
   "alg": "ES256",
   "x5c": ["MIIB2jCC...dA=="]
}
{
  "ietf-voucher-request:voucher": {
   "created-on": "2021-04-16T00:00:02.000Z",
   "nonce": "eDs++/FuDHGUnRxN3E14CQ==",
   "serial-number": "callee4711",
   "assertion": "agent-proximity",
   "agent-provided-proximity-registrar-cert": "base64encodedvalue==",
   "agent-signed-data": "base64encodedvalue==",
   "agent-sign-cert": "base64encodedvalue=="
  }
}
{
    SIGNATURE
}
```


Figure 7: Example of pledge-voucher-request

The pledge-voucher-request Content-Type is defined in [I-D.richardson-anima-jose-voucher] as:

application/voucher-jose+json

The pledge SHOULD include an "Accept" header field indicating the acceptable media type for the voucher response. The media type "application/voucher-jose+json" is defined in [I-D.richardson-anima-jose-voucher].

Once the registrar-agent has received the pledge-voucher-request it can trigger the pledge to generate an enrollment-request object. As in BRSKI the enrollment request object is a PKCS#10, additionally signed by the IDevID. Note, as the initial enrollment aims to request a general certificate, no certificate attributes are provided to the pledge.

Triggering the pledge to create the enrollment-request is done using HTTPS GET on the defined pledge endpoint "/.well-known/brski/pledge-enrollment-request".

The registrar-agent pledge-enrollment-request Content-Type header is:

application/json:

with an empty body.

Upon receiving the enrollment-trigger, the pledge SHALL construct the pledge-enrollment-request as authenticated self-contained object. The CSR already assures proof of possession of the private key corresponding to the contained public key. In addition, based on the additional signature using the IDevID, proof of identity is provided. Here, a JOSE object is being created in which the body utilizes the YANG module for the CSR as defined in [I-D.ietf-netconf-sztp-csr].

[RFC Editor: please delete] /* Open Issues: Reuse of the sub-tree ietf-sztp-csr:csr may not be possible as it is not the complete module. */

Depending on the capability of the pledge, it MAY construct the enrollment request as plain PKCS#10. Note that the focus here is placed on PKCS#10 as PKCS#10 can be transmitted in different enrollment protocols like EST, CMP, CMS, and SCEP. If the pledge is already implementing an enrollment protocol, it may leverage that functionality for the creation of the enrollment request object. Note also that [I-D.ietf-netconf-sztp-csr] also allows for inclusion of certificate request objects from CMP or CMC.

The pledge SHOULD construct the pledge-enrollment-request as PKCS#10 object and sign it additionally with its IDevID credential. The pledge-enrollment-request should be encoded as JOSE object.

[RFC Editor: please delete] /* Open Issues: Depending on target environment, it may be useful to assume that the pledge may already "know" its functional scope and therefore the number of certificates needed during operation. As a result, multiple CSRs may be generated to provide achieve multiple certificates as a result of the enrollment. This would need further description and potential enhancements also in the enrollment-request object to transport different CSRs. */

[I-D.ietf-netconf-sztp-csr] considers PKCS#10 but also CMP and CMC as certificate request format. Note that the wrapping signature is only necessary for plain PKCS#10 as other request formats like CMP and CMS support the signature wrapping as part of their own certificate request format.

The registrar-agent enrollment-request Content-Type header for a wrapped PKCS#10 is:

application/jose:

The header of the pledge enrollment-request SHALL contain the following parameter as defined in [RFC7515]:

- alg: algorithm used for creating the object signature.

- x5c: contains the base64-encoded pledge IDevID certificate.

The body of the pledge enrollment-request object SHOULD contain a P10 parameter (for PKCS#10) as defined for ietf-sztp-csr:csr in [I-D.ietf-netconf-sztp-csr]:

- P10: contains the base64-encoded PKCS#10 of the pledge.

The JOSE object is signed using the pledge's IDevID credential, which corresponds to the certificate signaled in the JOSE header.

```
{
    "alg": "ES256",
    "x5c": ["MIIB2jCC...dA=="]
}
{
  "ietf-sztp-csr:csr": {
    "p10": "base64encodedvalue=="
  }
}
{
    SIGNATURE
}
```

Figure 8: Example of pledge-enrollment-request

With the collected pledge-voucher-request object and the pledge-enrollment-request object, the registrar-agent starts the interaction with the domain registrar.

[RFC Editor: please delete] /*

Open Issues: further description necessary at least for */

- Values to be taken from the IDevID into the PKCS#10 (like product-serial-number or subjectName, or certificate template)

Once the registrar-agent has collected the pledge-voucher-request and pledge-enrollment-request objects, it connects to the registrar and sends the request objects. As the registrar-agent is intended to work between the pledge and the domain registrar, a collection of requests from more than one pledges is possible, allowing a bulk bootstrapping of multiple pledges using the same connection between the registrar-agent and the domain registrar.

5.2.3.2. Request handling (registrar-agent - infrastructure)

The bootstrapping exchange between the registrar-agent and the domain registrar resembles the exchanges between the pledge and the domain registrar from BRSKI in the pledge-initiator-mode with some deviations.

Preconditions:

- Registrar-agent: possesses IDevID CA certificate and own LDevID(RegAgt) EE credential of registrar domain. It knows the address of the domain registrar through configuration or discovery by, e.g., mDNS/DNSSD. The registrar-agent has acquired pledge-voucher-request and pledge-enrollment-request objects(s).

- Registrar: possesses IDevID CA certificate of pledge vendors / manufacturers and an own LDevID(Reg) EE credential.

- MASA: possesses own credentials (voucher signing key, TLS server certificate) as well as IDevID CA certificate of pledge vendor / manufacturer and site-specific LDevID CA certificate.

```
+-----------+    +-----------+   +--------+   +---------+
| Registrar |    | Domain    |   | Domain |   | Vendor  |
| Agent     |    | Registrar |   | CA     |   | Service |
| (RegAgt)  |    |  (JRC)    |   |        |   | (MASA)  |
+-----------+    +-----------+   +--------+   +---------+
    |                   |              |   Internet |
[exchange between pledge and ]
[registrar-agent done. ]
    |                   |              |            |
    |<------ TLS ----->|              |            |
    |                   |              |            |
    |-- Voucher-Req -->|              |            |
    |           [accept device?]       |            |
    |           [contact vendor]       |            |
    |                   |------------ TLS --------->|
    |                   |-- Voucher-Req ---------->|
    |                   |              [extract DomainID]
    |                   |              [update audit log]
    |<---- Voucher ----|<-------- Voucher ---------|
    |                   |              |            |
[certification request handling registrar-agent]
[and site infrastructure]
    |--- Enroll-Req -->|              |            |
    |                   |---- TLS ---->|            |
    |                   |- Enroll-Req->|            |
    |                   |<-Enroll-Resp-|            |
    |<-- Enroll-Resp---|              |            |
    |                   |              |            |
```

Figure 9: Request processing between registrar-agent and infrastructure bootstrapping services

The registrar-agent establishes a TLS connection with the registrar. As already stated in [RFC8995], the use of TLS 1.3 (or newer) is encouraged. TLS 1.2 or newer is REQUIRED on the registrar-agent side. TLS 1.3 (or newer) SHOULD be available on the registrar, but TLS 1.2 MAY be used. TLS 1.3 (or newer) SHOULD be available on the MASA, but TLS 1.2 MAY be used.

In contrast to [RFC8995] client authentication is achieved by using the LDevID(RegAgt) of the registrar-agent instead of the IDevID of the pledge. This allows the registrar to distinguish between pledge-initiator-mode and pledge-responder-mode. In pledge-responder-mode the registrar has no direct connection to the pledge but via the registrar-agent. The registrar can receive request objects in different forms as defined in [RFC8995]. Specifically, the registrar will receive JOSE objects from the pledge for voucher-request and enrollment-request (instead of the objects for voucher-request (CMS-signed JSON) and enrollment-request (PKCS#10).

The registrar-agent sends the pledge-voucher-request to the registrar with an HTTPS POST to the endpoint "/.well-known/brski/requestvoucher".

The pledge-voucher-request Content-Type used in the pledge-responder-mode is defined in [I-D.richardson-anima-jose-voucher] as:

application/voucher-jose+json (see Figure 7 for the content definition).

The registrar-agent SHOULD include the "Accept" header field received during the communication with the pledge, indicating the pledge acceptable Content-Type for the voucher-response. The voucher-response Content-Type "application/voucher-jose+json" is defined in [I-D.richardson-anima-jose-voucher].

Upon reception of the pledge-voucher-request, the registrar SHALL perform the verification of the voucher-request parameter as defined in section 5.3 of [RFC8995]. In addition, the registrar shall verify the following parameters from the pledge-voucher-request:

- o agent-provided-proximity-registrar-cert: MUST contain the own LDevID(Reg) EE certificate to ensure the registrar in proximity is the target registrar for the request.

- o agent-signed-data: The registrar MUST verify that the data has been signed with the LDevID(RegAgt) credential indicated in the "kid" JOSE header parameter. If the certificate is not contained in the agent-sign-cert component of the pledge-voucher-request, it must fetch the certificate from a repository.

- o agent-sign-cert: May contain the base64-encoded LDevID(RegAgt) certificate. If contained the registrar MUST verify that the connected credential used to sign the data was valid at signature creation time and that the corresponding registrar-agent was authorized to be involved in the bootstrapping.

If validation fails the registrar SHOULD respond with the HTTP 404 error code to the registrar-agent. If the pledge-voucher-request is in an unknown format, then an HTTP 406 error code is more appropriate.

If validation succeeds, the registrar will accept the pledge request to join the domain as defined in section 5.3 of [RFC8995]. The registrar then establishes a TLS connection with the MASA as described in section 5.4 of [RFC8995] to obtain a voucher for the pledge.

The registrar SHALL construct the body of the registrar-voucher-request object as defined in [RFC8995]. The encoding SHALL be done as JOSE object as defined in [I-D.richardson-anima-jose-voucher].

The header of the registrar-voucher-request SHALL contain the following parameter as defined in [RFC7515]:

o alg: algorithm used for creating the object signature.

o x5c: contains the base64-encoded registrar LDevID certificate.

The body of the registrar-voucher-request object MUST contain the following parameter as part of the ietf-voucher-request:voucher as defined in [RFC8995]:

o created-on: contains the current date and time in yang:date-and-time format for the registrar-voucher-request creation time.

o nonce: copied form the pledge-voucher-request

o serial-number: contains the base64-encoded product-serial-number. The registrar MUST verify that the product-serial-number contained in the IDevID certificate of the pledge matches the serial-number field in the pledge-voucher-request. In addition, it MUST be equal to the serial-number field contained in the agent-signed data of pledge-voucher-request.

o assertion: contains the voucher assertion requested the pledge (agent-proximity). The registrar provides this information to assure successful verification of agent proximity based on the agent-signed-data.

The ietf-voucher-request:voucher can be optionally enhanced with the following additional parameter:

o agent-sign-cert: Contain the base64-encoded LDevID(RegAgt) EE certificate if MASA verification of agent-proximity is required to provide the assertion "agent-proximity".

The object is signed using the registrar LDevID(Reg) credential, which corresponds to the certificate signaled in the JOSE header.

```
{
   "alg": "ES256",
   "x5c": ["MIIB2jCC...dA=="]
}
{
  "ietf-voucher-request:voucher": {
   "created-on": "2021-04-16T02:37:39.235Z",
   "nonce": "eDs++/FuDHGUnRxN3E14CQ==",
   "serial-number": "callee4711",
   "assertion": "agent-proximity",
   "prior-signed-voucher-request": "base64encodedvalue==",
   "agent-sign-cert": "base64encodedvalue=="
  }
}
{
    SIGNATURE
}
```


Figure 10: Example of registrar-voucher-request

The registrar sends the registrar-voucher-request to the MASA with an HTTPS POST at the endpoint "/.well-known/brski/requestvoucher".

The registrar-voucher-request Content-Type is defined in [I-D.richardson-anima-jose-voucher] as:

application/voucher-jose+json

The registrar SHOULD include an "Accept" header field indicating the acceptable media type for the voucher-response. The media type "application/voucher-jose+json" is defined in [I-D.richardson-anima-jose-voucher].

Once the MASA receives the registrar-voucher-request it SHALL perform the verification of the contained components as described in section 5.5 in [RFC8995]. In addition, the following additional processing SHALL be done for components contained in the prior-signed-voucher-request:

- o agent-provided-proximity-registrar-cert: The MASA MAY verify that this field contains the LDevID(Reg) certificate. If so, it MUST be consistent with the certificate used to sign the registrar-voucher-request.

- o agent-signed-data: The MASA MAY verify this field to be able to provide an assertion "agent-proximity". If so, the agent-signed-data MUST contain the product-serial-number of the pledge contained in the serial-number component of the prior-signed-voucher and also in serial-number component of the registrar-voucher-request. The LDevID(RegAgt) used to generate provide the signature is identified by the "kid" parameter of the JOSE header (agent-signed-data). If the assertion "agent-proximity" is requested, the registrar-voucher-request MUST contain the corresponding LDevID(RegAgt) EE certificate in the agent-sign-cert, which can be verified by the MASA as issued by the same domain CA as the LDevID(Reg) EE certificate. If the agent-sign-cert is not provided, the MASA MAY provide a lower level assertion "logged" or "verified"

If validation fails, the MASA SHOULD respond with an HTTP error code to the registrar. The error codes are kept as defined in section 5.6 of [RFC8995]. and comprise the response codes 403, 404, 406, and 415.

The voucher response format is as indicated in the submitted Accept header fields or based on the MASA's prior understanding of proper format for this pledge. Specifically for the pledge-responder-mode the "application/voucher-jose+json" as defined in [I-D.richardson-anima-jose-voucher] is applied. The syntactic details of vouchers are described in detail in [RFC8366]. Figure 11 shows an example of the contents of a voucher.


```
{
    "alg": "ES256",
    "x5c": ["MIIBkzCCAT...dA=="]
}
{
  "ietf-voucher:voucher": {
    "assertion": "agent-proximity",
    "serial-number": "callee4711",
    "nonce": "eDs++/FuDHGUnRxN3E14CQ==",
    "created-on": "2021-04-17T00:00:02.000Z",
    "pinned-domain-cert": "MIIBpDCCA...w=="
  }
}
{
    SIGNATURE
}
```


Figure 11: Example of MASA issued voucher

The MASA sends the voucher in the indicated form to the registrar. After receiving the voucher the registrar may evaluate the voucher for transparency and logging purposes as outlined in section 5.6 of

[RFC8995]. The registrar forwards the voucher without changes to the registrar-agent.

After receiving the voucher, the registrar-agent sends the pledge's enrollment-request to the registrar. Deviating from BRSKI the enrollment-request is not a raw PKCS#10 request. As the registrar-agent is involved in the exchange, the PKCS#10 is contained in the JOSE object. The signature is created using the pledge's IDevID to provide proof-of-identity as outlined in Figure 8.

When using EST, the registrar-agent sends the enrollment request to the registrar with an HTTPS POST at the endpoint "/.well-known/est/simpleenroll".

The enrollment-request Content-Type is:

application/jose

If validation of the wrapping signature fails, the registrar SHOULD respond with the HTTP 404 error code. If the voucher-request is in an unknown format, then an HTTP 406 error code is more appropriate. A situation that could be resolved with administrative action (such as adding a vendor/manufacturer IDevID CA as trusted party) MAY be responded with an 403 HTTP error code.

This results in a deviation from the content types used in [RFC7030] and results in additional processing at the domain registrar as EST server as following. Note that the registrar is already aware that the bootstrapping is performed in a pledge-responder-mode due to the use of the LDevID(RegAgt) certificate in the TLS establishment and the provided pledge-voucher-request in JOSE object.

- o If registrar receives the enrollment-request with the Content Type application/jose, it MUST verify the signature using the certificate indicated in the JOSE header.

- o The domain registrar verifies that the serial-number contained in the pledge's IDevID certificate contained in the JOSE header as being accepted to join the domain, based on the verification of the pledge-voucher-request.

- o If both succeed, the registrar utilizes the PKCS#10 request contained in the JOSE body as "P10" parameter of "ietf-sztp-csr:csr" for further processing of the enrollment request with the domain CA.

[RFC Editor: please delete] /*

Open Issues:

- The domain registrar may either enhance the PKCS#10 request or generate a structure containing the attributes to be included by the CA and sends both (the original PKCS#10 request and the enhancements) to the domain CA. As enhancing the PKCS#10 request destroys the initial proof of possession of the corresponding private key, the CA would need to accept RA-verified requests.

A successful interaction with the domain CA will result in the pledge LDevID EE certificate, which is then forwarded by the registrar to the registrar-agent using the content type "application/pkcs7-mime".

The registrar-agent has now finished the exchanges with the domain registrar. Now the registrar-agent can supply the voucher-response (from MASA via Registrar) and the enrollment-response (LDevID EE certificate) to the pledge. It can close the TLS connection to the domain registrar and provide the objects to the pledge(s). The content of the response objects is defined through the voucher [RFC8366] and the certificate [RFC5280].

5.2.3.3. Response object supply (registrar-agent - pledge)

The following description assumes that the registrar-agent has obtained the response objects from the domain registrar. It will re-start the interaction with the pledge. To contact the pledge, it may either discover the pledge as described in Section 5.2.2.2 or use stored information from the first contact with the pledge.

Preconditions in addition to Section 5.2.3.2:

- Registrar-agent: possesses voucher and LDevID certificate.

```
+--------+                         +-----------+
| Pledge |                         | Registrar |
|        |                         | Agent     |
|        |                         | (RegAgt)  |
+--------+                         +-----------+
    |                                    |
    |<------- supply voucher -----------|
    |                                    |
    |--------- voucher-status --------->| - store
    |                                    |   pledge voucher-status
    |<--- supply enrollment response ---|
    |                                    |
    |--------- enroll-status ---------->| - store
    |                                    |   pledge enroll-status
```

Figure 12: Response and status handling between pledge and registrar-agent

The registrar-agent provides the information via two distinct endpoints to the pledge as following.

The voucher response is provided with a HTTP POST using the operation path value of "/.well-known/brski/pledge-voucher".

The registrar-agent voucher-response Content-Type header is "application/voucher-jose+json and contains the voucher as provided by the MASA. An example if given in Figure 11.

The pledge verifies the voucher as described in section 5.6.1 in [RFC8995].

After successful verification the pledge MUST reply with a status telemetry message as defined in section 5.7 of [RFC8995]. As for the other objects, the defined object is provided with an additional signature using JOSE. The pledge generates the voucher-status-object and provides it in the response message to the registrar-agent.

The response has the Content-Type "application/jose", signed using the IDevID of the pledge as shown in Figure 13. As the reason field is optional (see [RFC8995]), it MAY be omitted in case of success.

```
{
    "alg": "ES256",
    "x5c": ["MIIB2jCC...dA=="]
{
    "version": 1,
    "status":true,
    "reason":"Informative human readable message",
    "reason-context": { "additional" : "JSON" }
}
{
    SIGNATURE
}
```


Figure 13: Example of pledge voucher-status telemetry

The enrollment response is provided with a HTTP POST using the operation path value of "/.well-known/brski/pledge-enrollment".

The registrar-agent enroll-response Content-Type header when using EST [RFC7030] as enrollment protocol, from the registrar-agent to the infrastructure is:

application/pkcs7-mime: note that it only contains the LDevID certificate for the pledge, not the certificate chain.

[RFC Editor: please delete] /*

Open Issue: the enrollment response object may also be an application/jose object with a signature of the domain registrar. This may be used either to transport additional data which is bound to the LDevID or it may be considered for enrollment status to ensure that in an error case the registrar providing the certificate can be identified. */

After successful verification the pledge MUST reply with a status telemetry message as defined in section 5.9.4 of [RFC8995]. As for the other objects, the defined object is provided with an additional signature using the JOSE. The pledge generates the enrollment status and provides it in the response message to the registrar-agent.

The response has the Content-Type "application/jose", signed using the LDevID of the pledge as shown in Figure 14. As the reason field is optional, it MAY be omitted in case of success.

```
{
  "alg": "ES256",
  "x5c": ["MIIB56uz...dA=="]
{
  "version": 1,
  "status":true,
  "reason":"Informative human readable message",
  "reason-context": { "additional" : "JSON" }
}
{
  SIGNATURE
}
```


Figure 14: Example of pledge enroll-status telemetry

Once the registrar-agent has collected the information, it can connect to the registrar agent to provide the status responses to the registrar.

5.2.3.4. Telemetry status handling (registrar-agent - domain registrar)

The following description assumes that the registrar-agent has collected the status objects from the pledge. It will provide the status objects to the registrar for further processing and audit log information of voucher-status for MASA.

Preconditions in addition to Section 5.2.3.2:

- Registrar-agent: possesses voucher-status and enroll-status objects from pledge.

```
   +-----------+    +-----------+   +--------+   +---------+
   | Registrar |    | Domain    |   | Domain |   | Vendor  |
   | Agent     |    | Registrar |   | CA     |   | Service |
   | RegAgt)   |    |  (JRC)    |   |        |   | (MASA)  |
   +-----------+    +-----------+   +--------+   +---------+
        |                   |               |   Internet |
        |                   |               |            |
        |<------ TLS ----->|               |            |
        |                   |               |            |
        |--Voucher-Status->|               |            |
        |                   |<---- device audit log ----|
        |           [verify audit log ]                  |
        |                   |               |            |
        |--Enroll-Status-->|               |            |
        |                   |               |            |
        |                   |               |            |
```

Figure 15: Bootstrapping status handling

The registrar-agent MUST provide the collected pledge voucher-status to the registrar. This status indicates the pledge could process the voucher successfully or not.

If the TLS connection to the registrar was closed, the registrar-agent establishes a TLS connection with the registrar as stated in Section 5.2.3.2.

The registrar-agent sends the pledge voucher-status object without modification to the registrar with an HTTPS POST using the operation path value of "/.well-known/brski/voucher_status". The Content-Type header is kept as "application/jose" as described in Figure 12 and depicted in the example in Figure 13.

The registrar SHALL verify the signature of the pledge voucher-status and validate that it belongs to an accepted device in his domain based on the contained "serial-number" in the IDevID certificate referenced in the header of the voucher-status object.

According to [RFC8995] section 5.7, the registrar SHOULD respond with an HTTP 200 but MAY simply fail with an HTTP 404 error. The registrar-agent may use the response to signal success / failure to the service technician operating the registrar agent. Within the server logs the server SHOULD capture this telemetry information.

The registrar SHOULD proceed with the collecting and logging the status information by requesting the MASA audit-log from the MASA service as described in section 5.8 of [RFC8995].

The registrar-agent MUST provide the enroll-status object to the registrar. The status indicates the pledge could process the enroll-response object and holds the corresponding private key.

The registrar-agent sends the pledge enroll-status object without modification to the registrar with an HTTPS POST using the operation path value of "/.well-known/brski/enrollstatus". The Content-Type header is kept as "application/jose" as described in Figure 12 and depicted in the example in Figure 14.

The registrar SHALL verify the signature of the pledge enroll-status object and validate that it belongs to an accepted device in his domain based on the contained product-serial-number in the LDevID EE certificate referenced in the header of the enroll-status object. Note that the verification of a signature of the object is a deviation form the described handling in section 5.9.4 of [RFC8995].

According to [RFC8995] section 5.9.4, the registrar SHOULD respond with an HTTP 200 but MAY simply fail with an HTTP 404 error. The registrar-agent may use the response to signal success / failure to the service technician operating the registrar agent. Within the server log the registrar SHOULD capture this telemetry information.

5.3. Domain registrar support of different enrollment options

Well-known URIs for different endpoints on the domain registrar are already defined as part of the base BRSKI specification. In addition, alternative enrollment endpoints may be supported at the domain registrar. The pledge / registrar-agent will recognize if its supported enrollment option is supported by the domain registrar by sending a request to its preferred enrollment endpoint.

The following provides an illustrative example for a domain registrar supporting different options for EST as well as CMP to be used in BRSKI-AE. The listing contains the supported endpoints for the bootstrapping, to which the pledge may connect. This includes the voucher handling as well as the enrollment endpoints. The CMP related enrollment endpoints are defined as well-known URI in CMP Updates [I-D.ietf-lamps-cmp-updates].

```
</brski/voucherrequest>,ct=voucher-cms+json
</brski/voucher_status>,ct=json
</brski/enrollstatus>,ct=json
</est/cacerts>;ct=pkcs7-mime
</est/simpleenroll>;ct=pkcs7-mime
</est/simplereenroll>;ct=pkcs7-mime
</est/fullcmc>;ct=pkcs7-mime
</est/serverkeygen>;ct= pkcs7-mime
</est/csrattrs>;ct=pkcs7-mime
</cmp/initialization>;ct=pkixcmp
</cmp/certification>;ct=pkixcmp
</cmp/keyupdate>;ct=pkixcmp
</cmp/p10>;ct=pkixcmp
</cmp/getCAcert>;ct=pkixcmp
</cmp/getCSRparam>;ct=pkixcmp
```

[RFC Editor: please delete] /*

Open Issues:

- In addition to the current content types, we may specify that the response provide information about different content types as multiple values. This would allow to further adopt the encoding of the objects exchanges (ASN.1, JSON, CBOR, ...). -> dependent on the utilized protocol. */

6. YANG Extensions to Voucher Request

The following modules extends the [RFC8995] Voucher Request to include a signed artifact from the registrar-agent as well as the registrar-proximity-certificate and the agent-signing certificate.

```
module ietf-async-voucher-request {
  yang-version 1.1;

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

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

```
     import ietf-voucher-request {
       prefix ivr;
       description
         "This module defines the format for a voucher request,
             which is produced by a pledge as part of the RFC8995
             onboarding process.";
       reference
         "RFC 8995: Bootstrapping Remote Secure Key Infrastructure";
     }

     organization
      "IETF ANIMA Working Group";

     contact
      "WG Web:   <http://tools.ietf.org/wg/anima/>
       WG List:  <mailto:anima@ietf.org>
       Author:   Steffen Fries
                 <mailto:steffen.fries@siemens.com>
       Author:   Hendrik Brockhaus
                 <mailto: hendrik.brockhaus@siemens.com>
       Author:   Eliot Lear
                 <mailto: lear@cisco.com>";
       Author:   Thomas Werner
                 <mailto: thomas-werner@siemens.com>";
     description
      "This module defines an extension of the RFC8995 voucher
       request to permit a registrar-agent to convey the adjacency
       relationship from the registrar-agent to the registrar.

       The key words 'MUST', 'MUST NOT', 'REQUIRED', 'SHALL',
       'SHALL NOT', 'SHOULD', 'SHOULD NOT', 'RECOMMENDED', 'MAY',
       and 'OPTIONAL' in the module text are to be interpreted as
       described in RFC 2119.";
     revision "YYYY-MM-DD" {
       description
        "Initial version";
       reference
        "RFC XXXX: Voucher Request for Asynchronous Enrollment";
     }
     rc:yang-data voucher-request-async-artifact {
       // YANG data template for a voucher.
       uses voucher-request-async-grouping;
     }
     // Grouping defined for future usage
     grouping voucher-request-async-grouping {
       description
         "Grouping to allow reuse/extensions in future work.";
       uses ivr:voucher-request-grouping {
```

```
         augment "voucher-request" {
           description "Base the constrained voucher-request upon the
             regular one";
           leaf agent-signed-data {
             type binary;
             description
               "The agent-signed-data field contains a JOSE [RFC7515]
                object provided by the Registrar-Agent to the Pledge.

                This artifact is signed by the Registrar-Agent
                and contains a copy of the pledge's serial-number.";
           }

           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";
           }
         }
       }
     }
   }
```

7. Example for signature-wrapping using existing enrollment protocols

This section map the requirements to support proof of possession and proof of identity to selected existing enrollment protocols. Note that that the work in the ACE WG described in [I-D.selander-ace-coap-est-oscore] may be considered here as well, as it also addresses the encapsulation of EST in a way to make it independent from the underlying TLS using OSCORE resulting in an authenticated self-contained object.

7.1. EST Handling

When using EST [RFC7030], the following constraints should be considered:

- o Proof of possession is provided by using the specified PKCS#10 structure in the request.
- o Proof of identity is achieved by signing the certification request object, which is only supported when Full PKI Request (the /fullcmc endpoint) is used. This contains sufficient information for the RA to make an authorization decision on the received certification request. Note: EST references CMC [RFC5272] for the definition of the Full PKI Request. For proof of identity, the signature of the SignedData of the Full PKI Request would be calculated using the IDevID credential of the pledge.
- o [RFC Editor: please delete] /* TBD: in this case the binding to the underlying TLS connection is not be necessary. */

- When the RA is not available, as per [RFC7030] Section 4.2.3, a 202 return code should be returned by the Registrar. The pledge in this case would retry a simpleenroll with a PKCS#10 request. Note that if the TLS connection is teared down for the waiting time, the PKCS#10 request would need to be rebuilt if it contains the unique identifier (tls_unique) from the underlying TLS connection for the binding.

- [RFC Editor: please delete] /* TBD: clarification of retry for fullcmc is necessary as not specified in the context of EST */

7.2. CMP Handling

Instead of using CMP [RFC4210], this specification refers to the lightweight CMP profile [I-D.ietf-lamps-lightweight-cmp-profile], as it restricts the full featured CMP to the functionality needed here. For this, the following constrains should be observed:

- For proof of possession, the defined approach in Lightweight CMP Profile section 4.1.1 (based on CRMF) and 4.1.5 (based on PCKS#10) should be supported.

- Proof of identity can be provided by using the signatures to protect the certificate request message as outlined in section 3.2. of [I-D.ietf-lamps-lightweight-cmp-profile].

- When the RA/CA is not available, a waiting indication should be returned in the PKIStatus by the Registrar. The pledge in this case would retry using the PollReqContent with a request identifier certReqId provided in the initial CertRequest message as specified in section 5.2.4 of [I-D.ietf-lamps-lightweight-cmp-profile] with delayed enrollment.

8. IANA Considerations

This document requires the following IANA actions:

IANA is requested to enhance the Registry entitled: "BRSKI well-known URIs" with the following:

URI	document	description
pledge-voucher-request	[THISRFC]	create pledge-voucher-request
pledge-enrollment-request	[THISRFC]	create pledge-enrollment-request
pledge-voucher	[THISRFC]	supply voucher response
pledge-enrollment	[THISRFC]	supply enrollment response
pledge-CACerts	[THISRFC]	supply CA certs to pledge

9. Privacy Considerations

The credential used by the registrar-agent to sign the data for the pledge in case of the pledge-initiator-mode should not contain personal information. Therefore, it is recommended to use an LDevID certificate associated with the device instead of a potential service technician operating the device, to avoid revealing this information to the MASA.

10. Security Considerations

10.1. Exhaustion attack on pledge

Exhaustion attack on pledge based on DoS attack (connection establishment, etc.)

10.2. Misuse of acquired voucher and enrollment responses

Registrar-agent that uses acquired voucher and enrollment response for domain 1 in domain 2: can be detected in Voucher Request processing on domain registrar side. Requires domain registrar to verify the proximity-registrar-cert leaf in the pledge-voucher-request against his own as well as the association of the pledge to his domain based on the product-serial-number contained in the voucher.

Misbinding of pledge by a faked domain registrar is countered as described in BRSKI security considerations (section 11.4).

Misuse of registrar-agent LDevID may be addressed by utilizing short-lived certificates to be used for authenticating the registrar-agent against the registrar. The LDevID certificate for the registrar-agent may be provided by a prior BRSKI execution based on an existing IDevID. Alternatively, the LDevID may be acquired by a service technician after authentication against the issuing CA.

11. Acknowledgments

We would like to thank the various reviewers for their input, in particular Brian E. Carpenter, Michael Richardson, Giorgio Romanenghi, Oskar Camenzind, for their input and discussion on use cases and call flows.

12. References

12.1. Normative References

[I-D.ietf-netconf-sztp-csr]
Watsen, K., Housley, R., and S. Turner, "Conveying a Certificate Signing Request (CSR) in a Secure Zero Touch Provisioning (SZTP) Bootstrapping Request", draft-ietf-netconf-sztp-csr-03 (work in progress), June 2021.

[I-D.richardson-anima-jose-voucher]
Richardson, M. and T. Werner, "JOSE signed Voucher Artifacts for Bootstrapping Protocols", draft-richardson-anima-jose-voucher-01 (work in progress), June 2021.

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

[RFC6762] Cheshire, S. and M. Krochmal, "Multicast DNS", RFC 6762, DOI 10.17487/RFC6762, February 2013, <https://www.rfc-editor.org/info/rfc6762>.

[RFC6763] Cheshire, S. and M. Krochmal, "DNS-Based Service Discovery", RFC 6763, DOI 10.17487/RFC6763, February 2013, <https://www.rfc-editor.org/info/rfc6763>.

[RFC7030] Pritikin, M., Ed., Yee, P., Ed., and D. Harkins, Ed., "Enrollment over Secure Transport", RFC 7030, DOI 10.17487/RFC7030, October 2013, <https://www.rfc-editor.org/info/rfc7030>.

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

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

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

[RFC8995] 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/info/rfc8995>.

12.2. Informative References

[I-D.ietf-lamps-cmp-updates]
Brockhaus, H. and D. V. Oheimb, "Certificate Management Protocol (CMP) Updates", draft-ietf-lamps-cmp-updates-10 (work in progress), May 2021.

[I-D.ietf-lamps-lightweight-cmp-profile]
Brockhaus, H., Fries, S., and D. V. Oheimb, "Lightweight Certificate Management Protocol (CMP) Profile", draft-ietf-lamps-lightweight-cmp-profile-05 (work in progress), February 2021.

[I-D.selander-ace-coap-est-oscore]
Selander, G., Raza, S., Furuhed, M., Vucinic, M., and T. Claeys, "Protecting EST Payloads with OSCORE", draft-selander-ace-coap-est-oscore-05 (work in progress), May 2021.

[IEC-62351-9]
International Electrotechnical Commission, "IEC 62351 - Power systems management and associated information exchange - Data and communications security - Part 9: Cyber security key management for power system equipment", IEC 62351-9 , May 2017.

[IEEE-802.1AR]
Institute of Electrical and Electronics Engineers, "IEEE 802.1AR Secure Device Identifier", IEEE 802.1AR , June 2018.

[ISO-IEC-15118-2]
International Standardization Organization / International Electrotechnical Commission, "ISO/IEC 15118-2 Road vehicles - Vehicle-to-Grid Communication Interface - Part 2: Network and application protocol requirements", ISO/IEC 15118-2 , April 2014.

[NERC-CIP-005-5]
North American Reliability Council, "Cyber Security - Electronic Security Perimeter", CIP 005-5, December 2013.

[OCPP] Open Charge Alliance, "Open Charge Point Protocol 2.0.1 (Draft)", December 2019.

[RFC2986] Nystrom, M. and B. Kaliski, "PKCS #10: Certification Request Syntax Specification Version 1.7", RFC 2986, DOI 10.17487/RFC2986, November 2000, <https://www.rfc-editor.org/info/rfc2986>.

[RFC4210] Adams, C., Farrell, S., Kause, T., and T. Mononen, "Internet X.509 Public Key Infrastructure Certificate Management Protocol (CMP)", RFC 4210, DOI 10.17487/RFC4210, September 2005, <https://www.rfc-editor.org/info/rfc4210>.

[RFC4211] Schaad, J., "Internet X.509 Public Key Infrastructure Certificate Request Message Format (CRMF)", RFC 4211, DOI 10.17487/RFC4211, September 2005, <https://www.rfc-editor.org/info/rfc4211>.

[RFC5272] Schaad, J. and M. Myers, "Certificate Management over CMS (CMC)", RFC 5272, DOI 10.17487/RFC5272, June 2008, <https://www.rfc-editor.org/info/rfc5272>.

[RFC5280] Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 5280, DOI 10.17487/RFC5280, May 2008, <https://www.rfc-editor.org/info/rfc5280>.

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

[RFC8894] Gutmann, P., "Simple Certificate Enrolment Protocol", RFC 8894, DOI 10.17487/RFC8894, September 2020, <https://www.rfc-editor.org/info/rfc8894>.

Appendix A. History of changes [RFC Editor: please delete]

From IETF draft 02 -> IETF draft 03:

- Housekeeping, deleted open issue regarding YANG voucher-request in Section 5.2.3.1 as voucher-request was enhanced with additional leaf.

- Included open issues in YANG model in Section 5.2 regarding assertion value agent-proximity and csr encapsulation using SZTP sub module).

From IETF draft 01 -> IETF draft 02:

o Defined call flow and objects for interactions in UC2. Object format based on draft for JOSE signed voucher artifacts and aligned the remaining objects with this approach in Section 5.2.3 .

o Terminology change: issue #2 pledge-agent -> registrar-agent to better underline agent relation.

o Terminology change: issue #3 PULL/PUSH -> pledge-initiator-mode and pledge-responder-mode to better address the pledge operation.

o Communication approach between pledge and registrar-agent changed by removing TLS-PSK (former section TLS establishment) and associated references to other drafts in favor of relying on higher layer exchange of signed data objects. These data objects are included also in the pledge-voucher-request and lead to an extension of the YANG module for the voucher-request (issue #12).

o Details on trust relationship between registrar-agent and registrar (issue #4, #5, #9) included in Section 5.2.

o Recommendation regarding short-lived certificates for registrar-agent authentication towards registrar (issue #7) in the security considerations.

o Introduction of reference to agent signing certificate using SKID in agent signed data (issue #11).

o Enhanced objects in exchanges between pledge and registrar-agent to allow the registrar to verify agent-proximity to the pledge (issue #1) in Section 5.2.3.

o Details on trust relationship between registrar-agent and pledge (issue #5) included in Section 5.2.

o Split of use case 2 call flow into sub sections in Section 5.2.3.

From IETF draft 00 -> IETF draft 01:

o Update of scope in Section 3.1 to include in which the pledge acts as a server. This is one main motivation for use case 2.

o Rework of use case 2 in Section 5.2 to consider the transport between the pledge and the pledge-agent. Addressed is the TLS channel establishment between the pledge-agent and the pledge as well as the endpoint definition on the pledge.

o First description of exchanged object types (needs more work)

- o Clarification in discovery options for enrollment endpoints at the domain registrar based on well-known endpoints in Section 5.3 do not result in additional /.well-known URIs. Update of the illustrative example. Note that the change to /brski for the voucher related endpoints has been taken over in the BRSKI main document.

- o Updated references.

- o Included Thomas Werner as additional author for the document.

From individual version 03 -> IETF draft 00:

- o Inclusion of discovery options of enrollment endpoints at the domain registrar based on well-known endpoints in Section 5.3 as replacement of section 5.1.3 in the individual draft. This is intended to support both use cases in the document. An illustrative example is provided.

- o Missing details provided for the description and call flow in pledge-agent use case Section 5.2, e.g. to accommodate distribution of CA certificates.

- o Updated CMP example in Section 7 to use lightweight CMP instead of CMP, as the draft already provides the necessary /.well-known endpoints.

- o Requirements discussion moved to separate section in Section 4. Shortened description of proof of identity binding and mapping to existing protocols.

- o Removal of copied call flows for voucher exchange and registrar discovery flow from [RFC8995] in Section 5.1 to avoid doubling or text or inconsistencies.

- o Reworked abstract and introduction to be more crisp regarding the targeted solution. Several structural changes in the document to have a better distinction between requirements, use case description, and solution description as separate sections. History moved to appendix.

From individual version 02 -> 03:

- o Update of terminology from self-contained to authenticated self-contained object to be consistent in the wording and to underline the protection of the object with an existing credential. Note that the naming of this object may be discussed. An alternative name may be attestation object.

o Simplification of the architecture approach for the initial use case having an offsite PKI.

o Introduction of a new use case utilizing authenticated self-contain objects to onboard a pledge using a commissioning tool containing a pledge-agent. This requires additional changes in the BRSKI call flow sequence and led to changes in the introduction, the application example,and also in the related BRSKI-AE call flow.

o Update of provided examples of the addressing approach used in BRSKI to allow for support of multiple enrollment protocols in Section 5.1.5.

From individual version 01 -> 02:

o Update of introduction text to clearly relate to the usage of IDevID and LDevID.

o Definition of the addressing approach used in BRSKI to allow for support of multiple enrollment protocols in Section 5.1.5. This section also contains a first discussion of an optional discovery mechanism to address situations in which the registrar supports more than one enrollment approach. Discovery should avoid that the pledge performs a trial and error of enrollment protocols.

o Update of description of architecture elements and changes to BRSKI in Section 5.

o Enhanced consideration of existing enrollment protocols in the context of mapping the requirements to existing solutions in Section 4 and in Section 7.

From individual version 00 -> 01:

o Update of examples, specifically for building automation as well as two new application use cases in Section 3.2.

o Deletion of asynchronous interaction with MASA to not complicate the use case. Note that the voucher exchange can already be handled in an asynchronous manner and is therefore not considered further. This resulted in removal of the alternative path the MASA in Figure 1 and the associated description in Section 5.

o Enhancement of description of architecture elements and changes to BRSKI in Section 5.

- o Consideration of existing enrollment protocols in the context of mapping the requirements to existing solutions in Section 4.

- o New section starting Section 7 with the mapping to existing enrollment protocols by collecting boundary conditions.

Authors' Addresses

Steffen Fries
Siemens AG
Otto-Hahn-Ring 6
Munich, Bavaria 81739
Germany

Email: steffen.fries@siemens.com
URI: https://www.siemens.com/

Hendrik Brockhaus
Siemens AG
Otto-Hahn-Ring 6
Munich, Bavaria 81739
Germany

Email: hendrik.brockhaus@siemens.com
URI: https://www.siemens.com/

Eliot Lear
Cisco Systems
Richtistrasse 7
Wallisellen CH-8304
Switzerland

Phone: +41 44 878 9200
Email: lear@cisco.com

Thomas Werner
Siemens AG
Otto-Hahn-Ring 6
Munich, Bavaria 81739
Germany

Email: thomas-werner@siemens.com
URI: https://www.siemens.com/

LAMPS Working Group R. Housley
Internet-Draft Vigil Security
Obsoletes: 8708 (if approved) 13 August 2024
Intended status: Standards Track
Expires: 14 February 2025

Use of the HSS/LMS Hash-Based Signature Algorithm in the Cryptographic Message Syntax (CMS)

draft-ietf-lamps-rfc8708bis-02

Abstract

This document specifies the conventions for using the Hierarchical Signature System (HSS) / Leighton-Micali Signature (LMS) hash-based signature algorithm with the Cryptographic Message Syntax (CMS). In addition, the algorithm identifier and public key syntax are provided. The HSS/LMS algorithm is one form of hash-based digital signature; it is described in RFC 8554. This document obsoletes RFC 8708.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on 14 February 2025.

Copyright Notice

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

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

Table of Contents

1. Introduction

This document specifies the conventions for using the Hierarchical Signature System (HSS) / Leighton-Micali Signature (LMS) hash-based signature algorithm with the Cryptographic Message Syntax (CMS) [CMS] signed-data content type. The LMS system provides a one-time digital signature that is a variant of Merkle Tree Signatures (MTS). The HSS is built on top of the LMS system to efficiently scale for a larger numbers of signatures. The HSS/LMS algorithm is one form of hash-based digital signature, and it is described in [HASHSIG]. The HSS/LMS signature algorithm can only be used for a fixed number of signing operations with a given private key, and the number of signing operations depends upon the size of the tree. The HSS/LMS signature algorithm uses small public keys, and it has low computational cost; however, the signatures are quite large. The HSS/LMS private key can be very small when the signer is willing to perform additional computation at signing time; alternatively, the private key can consume additional memory and provide a faster signing time. The HSS/LMS signatures [HASHSIG] are currently defined to use SHA-256 [SHS] and SHAKE256 [SHA3].

1.1. ASN.1

CMS values are generated using ASN.1 [ASN1-B], using the Basic Encoding Rules (BER) and the Distinguished Encoding Rules (DER) [ASN1-E].

1.2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

1.3. Motivation

Recent advances in cryptanalysis [BH2013] and progress in the development of quantum computers [NAS2019] pose a threat to widely deployed digital signature algorithms. As a result, there is a need to prepare for a day when cryptosystems such as RSA and DSA that depend on discrete logarithms and factoring cannot be depended upon.

If cryptographically relevant quantum computers (CRQCs) are ever built, these computers will be able to break many of the public key cryptosystems currently in use. A post-quantum cryptosystem [PQC] is a system that is secure against quantum computers that have more than a trivial number of quantum bits (qubits). It is open to conjecture when it will be feasible to build such computers; however, RSA, DSA, Elliptic Curve Digital Signature Algorithm (ECDSA), and Edwards-curve Digital Signature Algorithm (EdDSA) are all vulnerable if CRQCs are ever developed.

Since the HSS/LMS signature algorithm does not depend on the difficulty of discrete logarithms or factoring, but on a second-preimage-resistant cryptographic hash function, the HSS/LMS signature algorithm is considered to be post-quantum secure. One use of post-quantum-secure signatures is the protection of software updates, perhaps using the format described in [FWPROT], to enable deployment of software that implements new cryptosystems.

1.4. Changes Since RFC 8708

At the time RFC 8708 was published, there were no plans to put an HSS/LMS public key in a certificate. The expectation was that the HSS/LMS public key would be distributed by some other means. Today, there are plans to put an HSS/LMS public key in a certificate. The KEY field of the pk-HSS-LMS-HashSig definition in the ASN.1 module does not come into play when using HSS/LMS signatures in the CMS;

however, it needs to be consistent with the yet-to-be-published document that will describe the conventions for carrying an HSS/LMS public key in a certificate. The pk-HSS-LMS-HashSig definition is updated to reflect no ASN.1 wrapping for the public key.

Additional HSS/LMS tree sizes have been defined. The list in Section 2.2 was expanded to include the recently defined ones.

Additional LM-OTS Signatures have been defined. The list in Section 2.3 was expanded to include the recently defined ones.

Provide more detail in Section 4 regarding allowed values in the X.509 certificate key usage extension for an HSS/LMS public key.

2. HSS/LMS Hash-Based Signature Algorithm Overview

Merkle Tree Signatures (MTS) are a method for signing a large but fixed number of messages. An MTS system depends on a one-time signature method and a collision-resistant hash function.

This specification makes use of the hash-based algorithm specified in [HASHSIG], which is the Leighton and Micali adaptation [LM] of the original Lamport-Diffie-Winternitz-Merkle one-time signature system [M1979] [M1987] [M1989a] [M1989b].

As implied by the name, the hash-based signature algorithm depends on a collision-resistant hash function. The hash-based signature algorithm specified in [HASHSIG] uses only the SHA-256 one-way hash function [SHS], but it establishes an IANA registry [IANA-LMS] to permit the registration of additional one-way hash functions in the future.

2.1. Hierarchical Signature System (HSS)

The MTS system specified in [HASHSIG] uses a hierarchy of trees. The N-time Hierarchical Signature System (HSS) allows subordinate trees to be generated when needed by the signer. Otherwise, generation of the entire tree might take weeks or longer.

An HSS signature as specified in [HASHSIG] carries the number of signed public keys (Nspk), followed by that number of signed public keys, followed by the LMS signature as described in Section 2.2. The public key for the topmost LMS tree is the public key of the HSS system. The LMS private key in the parent tree signs the LMS public key in the child tree, and the LMS private key in the bottom-most tree signs the actual message. The signature over the public key and the signature over the actual message are LMS signatures as described in Section 2.2.

The elements of the HSS signature value for a standalone tree (a top tree with no children) can be summarized as:

```
u32str(0) ||
lms_signature  /* signature of message */
```

where, u32str() and || are used as defined in [HASHSIG].

The elements of the HSS signature value for a tree with Nspk signed public keys can be summarized as:

```
u32str(Nspk) ||
signed_public_key[0] ||
signed_public_key[1] ||
   ...
signed_public_key[Nspk-2] ||
signed_public_key[Nspk-1] ||
lms_signature  /* signature of message */
```

where, as defined in Section 3.3 of [HASHSIG], the signed_public_key structure contains the lms_signature over the public key, followed by the public key itself. Note that Nspk is the number of levels in the hierarchy of trees minus 1.

2.2. Leighton-Micali Signature (LMS)

Each tree in the system specified in [HASHSIG] uses the Leighton-Micali Signature (LMS) system. LMS systems have two parameters. The first parameter is the height of the tree, h, which is the number of levels in the tree minus one. The [HASHSIG] specification supports five values for this parameter: h=5, h=10, h=15, h=20, and h=25. There are 2^h leaves in the tree. The second parameter, m, is the number of bytes output by the hash function, and it is the amount of data associated with each node in the tree. The [HASHSIG] specification supports the SHA-256 hash function [SHS], with m=32. Recently, support for SHA-256 with m=24, SHAKE256 [SHA3] with m=32, and SHAKE256 with m=24 have been specified. As a result, the HSS/LMS supports the following tree sizes:

* LMS_SHA256_M32_H5

* LMS_SHA256_M32_H10

* LMS_SHA256_M32_H15

* LMS_SHA256_M32_H20

* LMS_SHA256_M32_H25

* LMS_SHA256_M24_H5
* LMS_SHA256_M24_H10
* LMS_SHA256_M24_H15
* LMS_SHA256_M24_H20
* LMS_SHA256_M24_H25
* LMS_SHAKE_M32_H5
* LMS_SHAKE_M32_H10
* LMS_SHAKE_M32_H15
* LMS_SHAKE_M32_H20
* LMS_SHAKE_M32_H25
* LMS_SHAKE_M24_H5
* LMS_SHAKE_M24_H10
* LMS_SHAKE_M24_H15
* LMS_SHAKE_M24_H20
* LMS_SHAKE_M24_H25

All of these appear in the IANA registry [IANA-LMS]. Additional hash functions and tree sizes may be registered in the future.

As specified in [HASHSIG], the LMS public key consists of four elements: the lms_algorithm_type from the list above, the otstype to identify the Leighton-Micali One-Time Signature (LM-OTS) type as discussed in Section 2.3, the private key identifier (I) as described in Section 5.3 of [HASHSIG], and the m-byte string associated with the root node of the tree (T[1]).

The LMS public key can be summarized as:

```
u32str(lms_algorithm_type) || u32str(otstype) || I || T[1]
```

As specified in [HASHSIG], an LMS signature consists of four elements: the number of the leaf (q) associated with the LM-OTS signature value, an LM-OTS signature value as described in Section 2.3, a typecode indicating the particular LMS algorithm, and an array of values that is associated with the path through the tree from the leaf associated with the LM-OTS signature value to the root. The array of values contains the siblings of the nodes on the path from the leaf to the root but does not contain the nodes on the path itself. The array for a tree with height h will have h values. The first value is the sibling of the leaf, the next value is the sibling of the parent of the leaf, and so on up the path to the root.

The four elements of the LMS signature value can be summarized as:

```
u32str(q) ||
ots_signature ||
u32str(type) ||
path[0] || path[1] || ... || path[h-1]
```

2.3. Leighton-Micali One-Time Signature (LM-OTS) Algorithm

Merkle Tree Signatures (MTS) depend on a one-time signature method, and [HASHSIG] specifies the use of the LM-OTS, which has five parameters:

n: The length in bytes of the hash function output.

H: A preimage-resistant hash function that accepts byte strings of any length and returns an n-byte string.

w: The width in bits of the Winternitz coefficients. [HASHSIG] supports four values for this parameter: w=1, w=2, w=4, and w=8.

p: The number of n-byte string elements that make up the LM-OTS signature value.

ls: The number of bits that are left-shifted in the final step of the checksum function, which is defined in Section 4.4 of [HASHSIG].

The values of p and ls are dependent on the choices of the parameters n and w, as described in Appendix B of [HASHSIG].

The [HASHSIG] specifies four LM-OTS variants. Recently, support for SHA-256 with n=24, SHAKE256 [SHA3] with n=32, and SHAKE256 with n=24 have been specified. As a result, the LM-OTS Signatures are:

* LMOTS_SHA256_N32_W1

* LMOTS_SHA256_N32_W2

* LMOTS_SHA256_N32_W4

* LMOTS_SHA256_N32_W8
* LMOTS_SHA256_N24_W1
* LMOTS_SHA256_N24_W2
* LMOTS_SHA256_N24_W4
* LMOTS_SHA256_N24_W8
* LMOTS_SHAKE_N32_W1
* LMOTS_SHAKE_N32_W2
* LMOTS_SHAKE_N32_W4
* LMOTS_SHAKE_N32_W8
* LMOTS_SHAKE_N24_W1
* LMOTS_SHAKE_N24_W2
* LMOTS_SHAKE_N24_W4
* LMOTS_SHAKE_N24_W8

All of these appear in the IANA registry [IANA-LMS]. Additional LM-OTS signatures may be registered in the future.

Signing involves the generation of C, an n-byte random value.

The LM-OTS signature value can be summarized as the identifier of the LM-OTS variant, the random value, and a sequence of hash values (y[0] through y[p-1]) that correspond to the elements of the public key, as described in Section 4.5 of [HASHSIG]:

```
u32str(otstype) || C || y[0] || ... || y[p-1]
```

3. Algorithm Identifiers and Parameters

The algorithm identifier for an HSS/LMS hash-based signature is:

```
id-alg-hss-lms-hashsig OBJECT IDENTIFIER ::= { iso(1)
    member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs9(9)
    smime(16) alg(3) 17 }
```

When this object identifier is used for an HSS/LMS signature, the AlgorithmIdentifier parameters field MUST be absent (that is, the parameters are not present, and the parameters are not set to NULL).

In the CMS, the HSS/LMS signature value is a large OCTET STRING. The HSS/LMS signature generation is described in Section 2 of this document. The signature format is designed for easy parsing. The HSS, LMS, and LM-OTS components of the signature value each include a counter and a typecode that indirectly provide all of the information that is needed to parse the value during signature validation.

The signature value identifies the hash function used in the HSS/LMS tree.

4. HSS/LMS Public Key Identifier

The AlgorithmIdentifier for an HSS/LMS public key uses the id-alg-hss-lms-hashsig object identifier, and the parameters field MUST be absent.

When this AlgorithmIdentifier appears in the SubjectPublicKeyInfo field of a certification authority (CA) X.509 certificate [RFC5280], the certificate key usage extension MUST contain at least one of the following values: digitalSignature, nonRepudiation, keyCertSign, and cRLSign. However, it MUST NOT contain other values.

When this AlgorithmIdentifier appears in the SubjectPublicKeyInfo field of an end entity X.509 certificate [RFC5280], the certificate key usage extension MUST contain at least one of the following: digitalSignature or nonRepudiation. However, it MUST NOT contain other values.

```
pk-HSS-LMS-HashSig PUBLIC-KEY ::= {
    IDENTIFIER id-alg-hss-lms-hashsig
    -- KEY no ASN.1 wrapping --
    PARAMS ARE absent
    CERT-KEY-USAGE
      { digitalSignature, nonRepudiation, keyCertSign, cRLSign } }

HSS-LMS-HashSig-PublicKey ::= OCTET STRING
```

The id-alg-hss-lms-hashsig algorithm identifier is also referred to as id-alg-mts-hashsig. This synonym is based on the terminology used in an early draft version of the document that became [HASHSIG].

When the public key appears outside a certificate, it is an OCTET STRING. Like the signature format, it is designed for easy parsing. The value is the number of levels in the public key, L, followed by the LMS public key.

The HSS/LMS public key value can be described as:

```
      u32str(L) || lms_public_key
```

The public key for the topmost LMS tree is the public key of the HSS system. When L=1, the HSS system is a single tree.

5. Signed-Data Conventions

As specified in [CMS], the digital signature is produced from the message digest and the signer's private key. The signature is computed over different values depending on whether signed attributes are absent or present.

When signed attributes are absent, the HSS/LMS signature is computed over the content. When signed attributes are present, a hash is computed over the content using the same hash function that is used in the HSS/LMS tree, then a message-digest attribute is constructed with the hash of the content, and then the HSS/LMS signature is computed over the DER-encoded set of signed attributes (which MUST include a content-type attribute and a message-digest attribute). In summary:

```
   IF (signed attributes are absent)
   THEN HSS_LMS_Sign(content)
   ELSE message-digest attribute = Hash(content);
        HSS_LMS_Sign(DER(SignedAttributes))
```

When using [HASHSIG], the fields in the SignerInfo are used as follows:

* digestAlgorithm MUST contain the one-way hash function used in the HSS/LMS tree. For convenience, the AlgorithmIdentifier for SHA-256 from [PKIXASN1] and the AlgorithmIdentifier for SHAKE256 from [RFC8692] are repeated here:

```
mda-sha256 DIGEST-ALGORITHM ::= {
    IDENTIFIER id-sha256
    PARAMS TYPE NULL ARE preferredAbsent }

id-sha256 OBJECT IDENTIFIER ::= { joint-iso-itu-t(2)
    country(16) us(840) organization(1) gov(101) csor(3)
    nistAlgorithms(4) hashalgs(2) 1 }

mda-shake256 DIGEST-ALGORITHM ::= {
    IDENTIFIER id-shake256 }

id-shake256 OBJECT IDENTIFIER ::= { joint-iso-itu-t(2)
    country(16) us(840) organization(1) gov(101) csor(3)
    nistAlgorithm(4) hashAlgs(2) 12 }
```

* signatureAlgorithm MUST contain id-alg-hss-lms-hashsig, and the algorithm parameters field MUST be absent.

* signature contains the single HSS/LMS signature value resulting from the signing operation as specified in [HASHSIG].

6. Security Considerations

Implementations MUST protect the private keys. Compromise of the private keys will result in the ability to forge signatures. Along with the private key, the implementation MUST keep track of which leaf nodes in the tree have been used. Loss of integrity of this tracking data can cause a one-time key to be used more than once. As a result, when a private key and the tracking data are stored on non-volatile media or in a virtual machine environment, failed writes, virtual machine snapshotting or cloning, and other operational concerns must be considered to ensure confidentiality and integrity.

When generating an LMS key pair, an implementation MUST generate each key pair independently of all other key pairs in the HSS tree.

An implementation MUST ensure that an LM-OTS private key is used to generate a signature only one time and ensure that it cannot be used for any other purpose.

The generation of private keys relies on random numbers. The use of inadequate pseudorandom number generators (PRNGs) to generate these values can result in little or no security. An attacker may find it much easier to reproduce the PRNG environment that produced the keys, searching the resulting small set of possibilities, rather than brute-force searching the whole key space. The generation of quality random numbers is difficult, and [RFC4086] offers important guidance in this area.

The generation of hash-based signatures also depends on random numbers. While the consequences of an inadequate pseudorandom number generator (PRNG) to generate these values is much less severe than in the generation of private keys, the guidance in [RFC4086] remains important.

When computing signatures, the same hash function SHOULD be used to compute the message digest of the content and the signed attributes, if they are present.

7. IANA Considerations

In the "SMI Security for S/MIME Module Identifier (1.2.840.113549.1.9.16.0)" registry, IANA is requested to change the reference for value 64 to point to this document.

In the "SMI Security for S/MIME Algorithms (1.2.840.113549.1.9.16.3)" registry, IANA is requested to change the reference for value 17 to point to this document.

8. References

8.1. Normative References

[ASN1-B] ITU-T, "Information technology -- Abstract Syntax Notation One (ASN.1): Specification of basic notation", ITU-T Recommendation X.680, August 2015.

[ASN1-E] ITU-T, "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, August 2015.

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

[HASHSIG] McGrew, D., Curcio, M., and S. Fluhrer, "Leighton-Micali Hash-Based Signatures", RFC 8554, DOI 10.17487/RFC8554, April 2019, <https://www.rfc-editor.org/info/rfc8554>.

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

[RFC5280] Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 5280, DOI 10.17487/RFC5280, May 2008, <https://www.rfc-editor.org/info/rfc5280>.

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

[RFC8692] Kampanakis, P. and Q. Dang, "Internet X.509 Public Key Infrastructure: Additional Algorithm Identifiers for RSASSA-PSS and ECDSA Using SHAKEs", RFC 8692, DOI 10.17487/RFC8692, December 2019, <https://www.rfc-editor.org/info/rfc8692>.

[SHA3] National Institute of Standards and Technology, "SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions", DOI 10.6028/NIST.FIPS.202, FIPS PUB 202, August 2015, <https://doi.org/10.6028/NIST.FIPS.202>.

[SHS] National Institute of Standards and Technology (NIST), "Secure Hash Standard (SHS)", FIPS PUB 180-4, DOI 10.6028/NIST.FIPS.180-4, August 2015, <https://doi.org/10.6028/NIST.FIPS.180-4>.

8.2. Informative References

[BH2013] Ptacek, T., Ritter, T., Samuel, J., and A. Stamos, "The Factoring Dead: Preparing for the Cryptopocalypse", August 2013, <https://media.blackhat.com/us-13/us-13-Stamos-The-Factoring-Dead.pdf>.

[CMSASN1] Hoffman, P. and J. Schaad, "New ASN.1 Modules for Cryptographic Message Syntax (CMS) and S/MIME", RFC 5911, DOI 10.17487/RFC5911, June 2010, <https://www.rfc-editor.org/info/rfc5911>.

[CMSASN1U] Schaad, J. and S. Turner, "Additional New ASN.1 Modules for the Cryptographic Message Syntax (CMS) and the Public Key Infrastructure Using X.509 (PKIX)", RFC 6268, DOI 10.17487/RFC6268, July 2011, <https://www.rfc-editor.org/info/rfc6268>.

[FWPROT] Housley, R., "Using Cryptographic Message Syntax (CMS) to Protect Firmware Packages", RFC 4108, DOI 10.17487/RFC4108, August 2005, <https://www.rfc-editor.org/info/rfc4108>.

[IANA-LMS] IANA, "Leighton-Micali Signatures (LMS)", <https://www.iana.org/assignments/leighton-micali-signatures/>.

[LM] Leighton, T. and S. Micali, "Large provably fast and secure digital signature schemes based on secure hash functions", U.S. Patent 5,432,852, July 1995.

[M1979] Merkle, R., "Secrecy, Authentication, and Public Key Systems", Technical Report No. 1979-1, Information Systems Laboratory, Stanford University, 1979.

[M1987] Merkle, R., "A Digital Signature Based on a Conventional Encryption Function", Advances in Cryptology -- CRYPTO '87 Proceedings, Lecture Notes in Computer Science Vol. 293, DOI 10.1007/3-540-48184-2_32, 1988, <https://doi.org/10.1007/3-540-48184-2_32>.

[M1989a] Merkle, R., "A Certified Digital Signature", Advances in Cryptology -- CRYPTO '89 Proceedings, Lecture Notes in Computer Science Vol. 435, DOI 10.1007/0-387-34805-0_21, 1990, <https://doi.org/10.1007/0-387-34805-0_21>.

[M1989b] Merkle, R., "One Way Hash Functions and DES", Advances in Cryptology -- CRYPTO '89 Proceedings, Lecture Notes in Computer Science Vol. 435, DOI 10.1007/0-387-34805-0_40, 1990, <https://doi.org/10.1007/0-387-34805-0_40>.

[NAS2019] National Academies of Sciences, Engineering, and Medicine, "Quantum Computing: Progress and Prospects", The National Academies Press, DOI 10.17226/25196, 2019, <https://doi.org/10.17226/25196>.

[PKIXASN1] Hoffman, P. and J. Schaad, "New ASN.1 Modules for the Public Key Infrastructure Using X.509 (PKIX)", RFC 5912, DOI 10.17487/RFC5912, June 2010, <https://www.rfc-editor.org/info/rfc5912>.

[PQC] Bernstein, D., "Introduction to post-quantum cryptography", DOI 10.1007/978-3-540-88702-7_1, 2009, <http://www.springer.com/cda/content/document/cda_downloaddocument/9783540887010-c1.pdf>.

[RFC4086] Eastlake 3rd, D., Schiller, J., and S. Crocker, "Randomness Requirements for Security", BCP 106, RFC 4086, DOI 10.17487/RFC4086, June 2005, <https://www.rfc-editor.org/info/rfc4086>.

Appendix A. ASN.1 Module

```
   <CODE BEGINS>
   MTS-HashSig-2013
     { iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs9(9)
       id-smime(16) id-mod(0) id-mod-mts-hashsig-2013(64) }

   DEFINITIONS IMPLICIT TAGS ::= BEGIN

   EXPORTS ALL;

   IMPORTS
     PUBLIC-KEY, SIGNATURE-ALGORITHM, SMIME-CAPS
       FROM AlgorithmInformation-2009  -- RFC 5911 [CMSASN1]
         { iso(1) identified-organization(3) dod(6) internet(1)
           security(5) mechanisms(5) pkix(7) id-mod(0)
           id-mod-algorithmInformation-02(58) } ;

   --
   -- Object Identifiers
   --

   id-alg-hss-lms-hashsig OBJECT IDENTIFIER ::= { iso(1)
       member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs9(9)
       smime(16) alg(3) 17 }

   id-alg-mts-hashsig OBJECT IDENTIFIER ::= id-alg-hss-lms-hashsig

   --
   -- Signature Algorithm and Public Key
   --

   sa-HSS-LMS-HashSig SIGNATURE-ALGORITHM ::= {
       IDENTIFIER id-alg-hss-lms-hashsig
       PARAMS ARE absent
       PUBLIC-KEYS { pk-HSS-LMS-HashSig }
       SMIME-CAPS { IDENTIFIED BY id-alg-hss-lms-hashsig } }

   pk-HSS-LMS-HashSig PUBLIC-KEY ::= {
       IDENTIFIER id-alg-hss-lms-hashsig
       -- KEY no ASN.1 wrapping --
       PARAMS ARE absent
       CERT-KEY-USAGE
           { digitalSignature, nonRepudiation, keyCertSign, cRLSign } }

   HSS-LMS-HashSig-PublicKey ::= OCTET STRING

   --
```

```
   -- Expand the signature algorithm set used by CMS [CMSASN1U]
   --

   SignatureAlgorithmSet SIGNATURE-ALGORITHM ::=
       { sa-HSS-LMS-HashSig, ... }

   --
   -- Expand the S/MIME capabilities set used by CMS [CMSASN1]
   --

   SMimeCaps SMIME-CAPS ::=
       { sa-HSS-LMS-HashSig.&smimeCaps, ... }

   END
   <CODE ENDS>
```

Acknowledgements

Many thanks to the people that provided comments on the draft documents that resulted in RFC 8708. Thanks to Joe Clarke, Roman Danyliw, Scott Fluhrer, Jonathan Hammell, Ben Kaduk, Panos Kampanakis, Barry Leiba, John Mattsson, Jim Schaad, Sean Turner, Daniel Van Geest, and Dale Worley for their careful review and comments.

Many thanks to Daniel Van Geest for motivating the creation of this document.

Author’s Address

Russ Housley
Vigil Security, LLC
516 Dranesville Road
Herndon, VA 20170
United States of America
Email: housley@vigilsec.com

W. Cheng, Ed.
China Mobile
X. Min, Ed.
ZTE Corp.
T. Zhou
Huawei
J. Dai
FiberHome
Y. Peleg
Broadcom

28 August 2024

Encapsulation For MPLS Performance Measurement with Alternate-Marking Method

draft-ietf-mpls-inband-pm-encapsulation-15

Abstract

This document defines the encapsulation for MPLS performance measurement with the Alternate-Marking method, which performs flow-based packet loss, delay, and jitter measurements on the MPLS traffic.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on 1 March 2025.

Copyright Notice

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

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

Table of Contents

1. Introduction

[RFC9341] describes a performance measurement method, which can be used to measure packet loss, delay, and jitter on data traffic. Since this method is based on marking consecutive batches of packets, it is referred to as the Alternate-Marking Method. [RFC8372] discusses aspects to consider when developing a solution for MPLS flow identification for performance monitoring of MPLS flows.

This document defines the encapsulation for MPLS performance measurement with the Alternate-Marking method, which performs flow-based packet loss, delay, and jitter measurements on the MPLS traffic. The encapsulation defined in this document supports performance monitoring at the intermediate nodes and MPLS flow identification at both transport and service layers.

Note that in parallel to the work of this document, there is ongoing work on MPLS Network Actions (MNA) [I-D.ietf-mpls-mna-fwk]. Considering the MPLS performance measurement with the Alternate-Marking method can also be achieved by MNA encapsulation, it is agreed that this document will be made Historic once the MNA solution of performance measurement with the Alternate-Marking method is published as an RFC.

2. Conventions Used in This Document

2.1. Abbreviations

ACL: Access Control List

BoS: Bottom of Stack

cSPL: Composite Special Purpose Label

ECMP: Equal-Cost Multipath

DSCP: Differentiated Services Code Point

ELC: Entropy Label Capability

ERLD: Entropy Readable Label Depth

eSPL: Extended Special Purpose Label

FL: Flow-ID Label

FLC: Flow-ID Label Capability

FLI: Flow-ID Label Indicator

FRLD: Flow-ID Readable Label Depth

IPFIX: IP Flow Information Export

LSP: Label Switched Path

MNA: MPLS Network Actions

MPLS: Multi-Protocol Label Switching

NMS: Network Management System

PHP: Penultimate Hop Popping

PM: Performance Measurement

PW: PseudoWire

SFL: Synonymous Flow Label

SID: Segment ID

SR: Segment Routing

TC: Traffic Class

TTL: Time to Live

VC: Virtual Channel

VPN: Virtual Private Network

XL: Extension Label

2.2. 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.

3. Flow-based PM Encapsulation in MPLS

This document defines the Flow-based MPLS performance measurement encapsulation with alternate marking method, as shown in figure below.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          Extension Label (15)         | TC  |S|      TTL      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Flow-ID Label Indicator (TBA1)    | TC  |S|      TTL      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|              Flow-ID Label            |L|D|T|S|      TTL      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 1: Flow-based PM Encapsulation in MPLS

The Flow-ID Label Indicator (FLI) is an Extended Special Purpose Label (eSPL), which is combined with the Extension Label (XL, value 15) to form a Composite Special Purpose Label (cSPL), as defined in [RFC9017]. The FLI is defined in this document as value TBA1.

The Traffic Class (TC) and Time To Live (TTL) fields of the XL and FLI MUST use the same values of the label immediately preceding the XL. In this case the TC and TTL for the XL and FLI MAY be of different values. The Bottom of the Stack (BoS) bit [RFC3032] for the XL and FLI MUST be zero.

The Flow-ID Label (FL) is used as an MPLS flow identification [RFC8372]. Its value MUST be unique within the administrative domain. The Flow-ID Label values MAY be allocated by an external NMS or controller based on the measurement object instances (such as LSP or PW). There is a one-to-one mapping between a Flow-ID and a flow. The specific method on how to allocate the Flow-ID Label values is described in Section 5.

The FL, preceded by a cSPL, can be placed either at the bottom or in the middle, but not at the top, of the MPLS label stack, and it MAY appear multiple times within a label stack. Section 3.1 of this document provides several examples to illustrate the application of FL in a label stack. The TTL for the FL MUST be zero to ensure that it is not used inadvertently for forwarding. The BoS bit for the FL depends on whether the FL is placed at the bottom of the MPLS label stack, i.e., the BoS bit for the FL is set only when the FL is placed at the bottom of the MPLS label stack.

Besides the flow identification, a color-marking field is also necessary for the Alternate-Marking method. To achieve the purpose of coloring the MPLS traffic, and to distinguish between hop-by-hop measurement and edge-to-edge measurement, the TC for the FL is defined as follows:

* L(oss) bit is used for coloring the MPLS packets for loss measurement. Setting the bit means color 1 and unsetting the bit means color 0.

* D(elay) bit is used for coloring the MPLS packets for delay/jitter measurement. Setting the bit means color for delay measurement.

* T(ype) bit is used to indicate the measurement type. When the T bit is set to 1, that means edge-to-edge performance measurement. When the T bit is set to 0, that means hop-by-hop performance measurement.

Considering the FL is not used as a forwarding label, the repurposing of the TC for the FL is feasible and viable.

3.1. Examples for Applying Flow-ID Label in a label stack

Three examples of different layouts of the Flow-ID label (4 octets) are illustrated as follows. Note that more examples may exist.

(1) Layout of the Flow-ID label when applied to MPLS transport.

```
             +-----------------------+
             |          LSP          |
             |         Label         |
             +-----------------------+ <--+
             |       Extension       |    |
             |         Label         |    |
             +-----------------------+    |--- cSPL
             |     Flow-ID Label     |    |
             |       Indicator       |    |
             +-----------------------+ <--+
             |        Flow-ID        |
             |         Label         |
             +-----------------------+
             |      Application      |
             |         Label         |
             +-----------------------+ <= Bottom of stack
             |                       |
             |        Payload        |
             |                       |
             +-----------------------+
```

Figure 2: Applying Flow-ID to MPLS transport

Note that here if the penultimate hop popping (PHP) is in use, the PHP LSR that recognizes the cSPL MUST NOT pop the cSPL and the following Flow-ID label, otherwise the egress LSR would be excluded from the performance measurement.

Also note that in other examples of applying Flow-ID to MPLS transport, one LSP label can be substituted by multiple SID labels in the case of using SR Policy, and the combination of cSPL and Flow-ID label can be placed between SID labels, as specified in Section 6.

(2) Layout of the Flow-ID label when applied to MPLS service.

```
             +-----------------------+
             |          LSP          |
             |         Label         |
             +-----------------------+
             |      Application      |
             |         Label         |
             +-----------------------+ <--+
             |       Extension       |    |
             |         Label         |    |
             +-----------------------+    |--- cSPL
             |     Flow-ID Label     |    |
             |       Indicator       |    |
             +-----------------------+ <--+
             |        Flow-ID        |
             |         Label         |
             +-----------------------+ <= Bottom of stack
             |                       |
             |        Payload        |
             |                       |
             +-----------------------+
```

Figure 3: Applying Flow-ID to MPLS service

Note that in this case, the application label can be an MPLS PW label, MPLS Ethernet VPN label or MPLS IP VPN label, and it is also called a VC label as defined in [RFC4026].

(3) Layout of the Flow-ID label when applied to both MPLS transport and MPLS service.

```
                +----------------------+
                |         LSP          |
                |        Label         |
                +----------------------+ <--+
                |      Extension       |    |
                |        Label         |    |
                +----------------------+    |--- cSPL
                |    Flow-ID Label     |    |
                |      Indicator       |    |
                +----------------------+ <--+
                |       Flow-ID        |
                |        Label         |
                +----------------------+
                |     Application      |
                |        Label         |
                +----------------------+ <--+
                |      Extension       |    |
                |        Label         |    |
                +----------------------+    |--- cSPL
                |    Flow-ID Label     |    |
                |      Indicator       |    |
                +----------------------+ <--+
                |       Flow-ID        |
                |        Label         |
                +----------------------+ <= Bottom of stack
                |                      |
                |       Payload        |
                |                      |
                +----------------------+
```

Figure 4: Applying Flow-ID to both MPLS transport and MPLS service

Note that for this example, the two Flow-ID Label values appearing in a label stack MUST be different. In other words, the Flow-ID label applied to the MPLS transport and the Flow-ID label applied to the MPLS service MUST be different. Also, note that the two Flow-ID label values are independent of each other. For example, two packets can belong to the same VPN flow but different LSP flows, or two packets can belong to different VPN flows but the same LSP flow.

4. Procedures of Encapsulation, Look-up and Decapsulation

The procedures for Flow-ID label encapsulation, look-up and decapsulation are summarized as follows:

* The MPLS ingress node [RFC3031] inserts the XL, FLI and FL into the MPLS label stack. At the same time, the ingress node sets the Flow-ID Label value, the two color-marking bits and the T bit, as defined in Section 3.

* If the edge-to-edge measurement is applied, i.e., the T bit is set to 1, then only the MPLS ingress/egress node [RFC3031] is the processing node, otherwise all the MPLS nodes along the LSP are the processing nodes. The processing node looks up the FL with the help of the XL and FLI, and exports the collected data, such as the Flow-ID, block counters and timestamps, to an external NMS/controller, referring to the Alternate-Marking method. Section 6 of [I-D.ietf-ippm-alt-mark-deployment] describes protocols for collected data export, and the details on how to export the collected data are outside the scope of this document. Note that while looking up the Flow-ID label, the transit node needs to perform some deep packet inspection beyond the label (at the top of the label stack) used to make forwarding decisions.

* The processing node MUST pop the XL, FLI and FL from the MPLS label stack when it needs to pop the preceding forwarding label. The egress node MUST pop the whole MPLS label stack, and this document doesn’t introduce any new process to the decapsulated packet.

5. Procedures of Flow-ID allocation

There are at least two ways of allocating Flow-ID. One way is to allocate Flow-ID by a manual trigger from the network operator, and the other way is to allocate Flow-ID by an automatic trigger from the ingress node. Details are as follows:

* In the case of a manual trigger, the network operator would manually input the characteristics (e.g. IP five tuples and IP DSCP) of the measured flow, then the NMS/controller would generate one or two Flow-IDs based on the input from the network operator, and provision the ingress node with the characteristics of the measured flow and the corresponding allocated Flow-ID(s).

* In the case of an automatic trigger, the ingress node would identify the flow entering the measured path, export the characteristics of the identified flow to the NMS/controller by IPFIX [RFC7011], then the NMS/controller would generate one or two Flow-IDs based on the characteristics exported from the ingress node, and provision the ingress node with the characteristics of the identified flow and the corresponding allocated Flow-ID(s).

The policy pre-configured at the NMS/controller decides whether one Flow-ID or two Flow-IDs would be generated. If the performance measurement on the MPLS service is enabled, then one Flow-ID applied to the MPLS service would be generated. If the performance measurement on the MPLS transport is enabled, then one Flow-ID applied to the MPLS transport would be generated. If both of them are enabled, then two Flow-IDs are respectively applied to the MPLS service and the MPLS transport would be generated. In this case, the transit node needs to look up both of the two Flow-IDs by default. However, this behaviour can be changed through configuration, such as by setting it to look up only the Flow-ID applied to the MPLS transport.

Whether using the two methods mentioned above or other methods to allocate Flow-ID, the NMS/controller MUST ensure that every generated Flow-ID is unique within the administrative domain and MUST NOT have any value in the reserved label space (0-15) [RFC3032].

6. FLC and FRLD Considerations

Analogous to the Entropy Label Capability (ELC) defined in Section 5 of [RFC6790] and the Entropy Readable Label Depth (ERLD) defined in Section 4 of [RFC8662], the Flow-ID Label Capability (FLC) and the Flow-ID Readable Label Depth (FRLD) are defined in this document. Both FLC and FRLD have similar semantics with the ELC and ERLD to a router, except that the Flow-ID is used in its flow identification function while the Entropy is used in its load-balancing function.

The ingress node MUST insert each FL at an appropriate depth, which ensures the node to which the FL is exposed has the FLC. The ingress node SHOULD insert each FL within an appropriate FRLD, which is the minimum FRLD of all the on-path nodes that need to read and use the FL in question. How the ingress node knows the FLC and FRLD of all the on-path nodes is outside the scope of this document. However, [I-D.xzc-lsr-mpls-flc-frld] provides a method to achieve this.

When the SR paths are used for transport, the label stack grows as the number of on-path segments increases. If the number of on-path segments is high, that may become a challenge for the FL to be placed within an appropriate FRLD. To overcome this potential challenge, an implementation MAY allow the ingress node to place FL between SID labels. This means that multiple identical FLs at different depths MAY be interleaved with SID labels. When this occurs, sophisticated network planning may be needed, which is beyond the scope of this document.

7. Equal-Cost Multipath Considerations

Analogous to what's described in Section 5 of [RFC8957], under conditions of Equal-Cost Multipath (ECMP), the introduction of the FL may lead to the same problem as caused by the Synonymous Flow Label (SFL). The two solutions proposed for SFL would also apply here. Specifically, adding FL to an existing flow may cause that flow to take a different path. If the operator expects to resolve this problem, they can choose to apply entropy labels [RFC6790] or add FL to all flows.

8. Security Considerations

As specified in Section 3, the value of a Flow-ID label MUST be unique within the administrative domain. In other words, the administrative domain is the scope of a Flow-ID label. The method for achieving multi-domain performance measurement with the same Flow-ID label is outside the scope of this document. The Flow-ID label MUST NOT be signaled and distributed outside the administrative domain. Improper configuration that allows the Flow-ID label to be passed from one administrative domain to another would result in Flow-ID conflicts.

To prevent packets carrying Flow-ID labels from leaking from one domain to another, domain boundary nodes SHOULD deploy policies (e.g., ACL) to filter out these packets. Specifically, at the sending edge, the domain boundary node SHOULD filter out the packets that carry the Flow-ID Label Indicator and are sent to other domains. At the receiving edge, the domain boundary node SHOULD drop the packets that carry the Flow-ID Label Indicator and are from other domains.

9. Implementation Status

[Note to the RFC Editor - remove this section before publication, as well as remove the reference to [RFC7942].

This section records the status of known implementations of the protocol defined by this specification at the time of posting of this Internet-Draft, and is based on a proposal described in [RFC7942]. The description of implementations in this section is intended to assist the IETF in its decision processes in progressing drafts to RFCs. Please note that the listing of any individual implementation here does not imply endorsement by the IETF. Furthermore, no effort has been spent to verify the information presented here that was supplied by IETF contributors. This is not intended as, and must not be construed to be, a catalog of available implementations or their features. Readers are advised to note that other implementations may exist.

According to [RFC7942], "this will allow reviewers and working groups to assign due consideration to documents that have the benefit of running code, which may serve as evidence of valuable experimentation and feedback that have made the implemented protocols more mature. It is up to the individual working groups to use this information as they see fit".

9.1. Fiberhome

* Organization: Fiberhome Corporation.

* Implementation: Fiberhome R82*, R800*, S680*, S780* series routers are running the common-building block 'Flow-based PM Encapsulation in MPLS'.

* Maturity Level: Product

* Coverage: Partial, section 3 and example (2) of section 3.1.

* Version: Draft-08

* Licensing: N/A

* Implementation experience: Nothing specific.

* Contact: djy@fiberhome.com

* Last updated: December 25, 2023

9.2. Huawei Technologies

* Organization: Huawei Technologies.

* Implementation: Huawei ATN8XX, ATN910C, ATN980B, CX600-M2, NE40E, ME60-X1X2, ME60-X3X8X16 Routers running VRPV800R021C00 or above. Huawei NCE-IP Controller running V1R21C00 or above.

* Maturity Level: Product

* Coverage: Partial, section 3 and example (2) of section 3.1.

* Version: Draft-08

* Licensing: N/A

* Implementation experience: Nothing specific.

* Contact: zhoutianran@huawei.com

* Last updated: January 10, 2024

9.3. ZTE Corp

* Organization: ZTE Corporation.

* Implementation: ZTE ZXCTN 6500-32 routers running V5.00.20 or above. ZTE ZXCTN 6170H routers running V5.00.30.20 or above. ZTE ElasticNet UME Controller running V16.22.20 or above.

* Maturity Level: Product

* Coverage: Partial, section 3 and example (2) of section 3.1.

* Version: Draft-08

* Licensing: N/A

* Implementation experience: Nothing specific.

* Contact: xiao.min2@zte.com.cn

* Last updated: January 22, 2024

9.4. China Mobile

China Mobile reported that they have conducted interconnection tests with multiple vendors according to this draft. The tests result have proven that the solutions from multiple vendors are mature and ready for large-scale deployment. This report was last updated on January 10, 2024.

10. IANA Considerations

From the "Extended Special-Purpose MPLS Label Values" registry in the "Special-Purpose Multiprotocol Label Switching (MPLS) Label Values" namespace, a new value for the Flow-ID Label Indicator is requested from IANA as follows:

Value	Description	Reference
TBA1 (value 18 is recommended)	Flow-ID Label Indicator (FLI)	This Document

Table 1: New Extended Special-Purpose MPLS Label Value for Flow-ID Label Indicator

11. Acknowledgements

The authors would like to acknowledge Loa Andersson, Tarek Saad, Stewart Bryant, Rakesh Gandhi, Greg Mirsky, Aihua Liu, Shuangping Zhan, Ming Ke, Wei He, Ximing Dong, Darren Dukes, Tony Li, James Guichard, and Daniele Ceccarelli for their careful review and very helpful comments.

They also wish to acknowledge Italo Busi and Chandrasekar Ramachandran for their insightful MPLS-RT review and constructive comments.

Additionally, the authors would like to thank Dhruv Dhody for the English grammar review.

12. Contributors

Minxue Wang
China Mobile
Email: wangminxue@chinamobile.com

Wen Ye
China Mobile
Email: yewen@chinamobile.com

13. References

13.1. Normative References

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

[RFC3031] Rosen, E., Viswanathan, A., and R. Callon, "Multiprotocol Label Switching Architecture", RFC 3031, DOI 10.17487/RFC3031, January 2001, <https://www.rfc-editor.org/info/rfc3031>.

[RFC3032] Rosen, E., Tappan, D., Fedorkow, G., Rekhter, Y., Farinacci, D., Li, T., and A. Conta, "MPLS Label Stack Encoding", RFC 3032, DOI 10.17487/RFC3032, January 2001, <https://www.rfc-editor.org/info/rfc3032>.

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

[RFC9017] Andersson, L., Kompella, K., and A. Farrel, "Special-Purpose Label Terminology", RFC 9017, DOI 10.17487/RFC9017, April 2021, <https://www.rfc-editor.org/info/rfc9017>.

13.2. Informative References

[I-D.ietf-ippm-alt-mark-deployment]
Fioccola, G., Keyi, Z., Graf, T., Nilo, M., and L. Zhang, "Alternate Marking Deployment Framework", Work in Progress, Internet-Draft, draft-ietf-ippm-alt-mark-deployment-01, 3 July 2024, <https://datatracker.ietf.org/doc/html/draft-ietf-ippm-alt-mark-deployment-01>.

[I-D.ietf-mpls-mna-fwk]
Andersson, L., Bryant, S., Bocci, M., and T. Li, "MPLS Network Actions (MNA) Framework", Work in Progress, Internet-Draft, draft-ietf-mpls-mna-fwk-10, 6 August 2024, <https://datatracker.ietf.org/doc/html/draft-ietf-mpls-mna-fwk-10>.

[I-D.xzc-lsr-mpls-flc-frld]
Min, X., Zhang, Z., and W. Cheng, "Signaling Flow-ID Label Capability and Flow-ID Readable Label Depth", Work in Progress, Internet-Draft, draft-xzc-lsr-mpls-flc-frld-04, 28 January 2024, <https://datatracker.ietf.org/doc/html/draft-xzc-lsr-mpls-flc-frld-04>.

[RFC4026] Andersson, L. and T. Madsen, "Provider Provisioned Virtual Private Network (VPN) Terminology", RFC 4026, DOI 10.17487/RFC4026, March 2005, <https://www.rfc-editor.org/info/rfc4026>.

[RFC6790] Kompella, K., Drake, J., Amante, S., Henderickx, W., and L. Yong, "The Use of Entropy Labels in MPLS Forwarding", RFC 6790, DOI 10.17487/RFC6790, November 2012, <https://www.rfc-editor.org/info/rfc6790>.

[RFC7011] Claise, B., Ed., Trammell, B., Ed., and P. Aitken, "Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of Flow Information", STD 77, RFC 7011, DOI 10.17487/RFC7011, September 2013, <https://www.rfc-editor.org/info/rfc7011>.

[RFC7942] Sheffer, Y. and A. Farrel, "Improving Awareness of Running Code: The Implementation Status Section", BCP 205, RFC 7942, DOI 10.17487/RFC7942, July 2016, <https://www.rfc-editor.org/info/rfc7942>.

[RFC8372] Bryant, S., Pignataro, C., Chen, M., Li, Z., and G. Mirsky, "MPLS Flow Identification Considerations", RFC 8372, DOI 10.17487/RFC8372, May 2018, <https://www.rfc-editor.org/info/rfc8372>.

[RFC8662] Kini, S., Kompella, K., Sivabalan, S., Litkowski, S., Shakir, R., and J. Tantsura, "Entropy Label for Source Packet Routing in Networking (SPRING) Tunnels", RFC 8662, DOI 10.17487/RFC8662, December 2019, <https://www.rfc-editor.org/info/rfc8662>.

[RFC8957] Bryant, S., Chen, M., Swallow, G., Sivabalan, S., and G. Mirsky, "Synonymous Flow Label Framework", RFC 8957, DOI 10.17487/RFC8957, January 2021, <https://www.rfc-editor.org/info/rfc8957>.

[RFC9341] Fioccola, G., Ed., Cociglio, M., Mirsky, G., Mizrahi, T., and T. Zhou, "Alternate-Marking Method", RFC 9341, DOI 10.17487/RFC9341, December 2022, <https://www.rfc-editor.org/info/rfc9341>.

Authors' Addresses

Weiqiang Cheng (editor)
China Mobile
Beijing
China

Email: chengweiqiang@chinamobile.com

Xiao Min (editor)
ZTE Corp.
Nanjing
China
Email: xiao.min2@zte.com.cn

Tianran Zhou
Huawei
Beijing
China
Email: zhoutianran@huawei.com

Jinyou Dai
FiberHome
Wuhan
China
Email: djy@fiberhome.com

Yoav Peleg
Broadcom
United States of America
Email: yoav.peleg@broadcom.com

SIDROPS J. Snijders
Internet-Draft Fastly
Intended status: Informational T. de Kock
Expires: 14 February 2025 RIPE NCC
13 August 2024

Detecting RRDP Session Desynchronization
draft-ietf-sidrops-rrdp-desynchronization-02

Abstract

This document describes an approach for Resource Public Key Infrastructure (RPKI) Relying Parties to detect a particular form of RPKI Repository Delta Protocol (RRDP) session desynchronization and how to recover.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on 14 February 2025.

Copyright Notice

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

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

Table of Contents

1. Introduction

The Resource Public Key Infrastructure (RPKI) Repository Delta Protocol (RRDP) [RFC8182] is a one-way synchronization protocol for distributing RPKI data in the form of _differences_ (deltas) between sequential repository states. Relying Parties apply a contiguous chain of deltas to synchronize their local copy of the repository with the current state of the remote Repository Server. Delta files for any given session_id and serial number are expected to contain an immutable record of the state of the Repository Server at that given point in time, but this is not always the case.

This document describes an approach for Relying Parties (RPs) to detect a particular form of RRDP session desynchronization and how to recover.

1.1. 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.

2. Immutability of RRDP files

Section 3.1 of [RFC8182] describes how discrete publication events such as the addition, modification, or deletion of one or more repository objects _can_ be communicated as immutable files, highlighting advantages for publishers, such as the ability to pre-calculate files and make use of caching infrastructure.

While the global RPKI is understood to present a loosely consistent view, depending on timing, updating, fetching (see Section 6 of [RFC7115]), different caches having different data for the same RRDP session at the same serial violates the principle of least astonishment.

If an RRDP server over time serves differing data for a given session_id and serial number, distinct RP instances (depending on the moment they connected to the RRDP server) would end up with divergent local repositories. Comparing only the server-provided session_id and latest serial number across distinct RP instances would not bring such divergence to light.

The [RFC8182] specification does allude to immutability being a property of RRDP files, but doesn't make it clear that immutability is an absolute requirement for the RRDP protocol to work well. A future update to [RFC8182] should set a hard rule to establish that the immutability of RRDP files must not be violated after publication, and that RPs should check for unexpected mutations.

3. Detection of Desynchronization

Relying Parties can implement a mechanism to keep a record of the serial and hash attribute values in delta elements of the previous successful fetch of an Update Notification File. Then, after fetching a new Update Notification File, the Relying Party should compare if the serial and hash values of previously seen serials match those in the newly fetched file. If any differences are detected, this means that the Delta files were unexpectedly mutated, and the RP should proceed to Section 4.

RP implementations decide on the number of Delta Files to process before switching to downloading the latest Snapshot File. The same upper bound can be used as a limit to the number of delta element serial and hash values to track.

3.1. Example

This section contains two versions of an Update Notification File to demonstrate an unexpected mutation. The initial Update Notification File is as follows:

```
<notification xmlns="http://www.ripe.net/rpki/rrdp" version="1"
              session_id="fe528335-db5f-48b2-be7e-bf0992d0b5ec"
              serial="1774">
  <snapshot uri="https://rrdp.example.net/1774/snapshot.xml"
    hash="4b5f27b099737b8bf288a33796bfe825fb2014a69fd6aa99080380299952f2e2"/>
  <delta serial="1774"
    hash="effac94afd30bbf1cd6e180e7f445a4d4653cb4c91068fa9e7b669d49b5aaa00"
    uri="https://rrdp.example.net/1774/delta.xml" />
  <delta serial="1773"
    hash="731169254dd5de0ede94ba6999bda63b0fae9880873a3710e87a71bafb64761a"
    uri="https://rrdp.example.net/1773/delta.xml" />
  <delta serial="1772"
    hash="d4087585323fd6b7fd899ebf662ef213c469d39f53839fa6241847f4f6ceb939"
    uri="https://rrdp.example.net/1772/delta.xml />
</notification>
```


Based on the above Update Notification File, an RP implementation could record the following state:

```
fe528335-db5f-48b2-be7e-bf0992d0b5ec
1774 effac94afd30bbf1cd6e180e7f445a4d4653cb4c91068fa9e7b669d49b5aaa00
1773 731169254dd5de0ede94ba6999bda63b0fae9880873a3710e87a71bafb64761a
1772 d4087585323fd6b7fd899ebf662ef213c469d39f53839fa6241847f4f6ceb939
```

Figure 1

A new version of the Update Notification File is published, as following:


```
<notification xmlns="http://www.ripe.net/rpki/rrdp" version="1"
              session_id="fe528335-db5f-48b2-be7e-bf0992d0b5ec"
              serial="1775">
  <snapshot uri="https://rrdp.example.net/1775/snapshot.xml"
    hash="cd430c386deacb04bda55301c2aa49f192b529989b739f412aea01c9a77e5389"/>
  <delta serial="1775"
    hash="d199376e98a9095dbcf14ccd49208b4223a28a1327669f89566475d94b2b08cc"
    uri="https://rrdp.example.net/1775/delta.xml />
  <delta serial="1774"
    hash="10ca28480a584105a059f95df5ca8369142fd7c8069380f84ebe613b8b89f0d3"
    uri="https://rrdp.example.net/1774/delta.xml" />
  <delta serial="1773"
    hash="731169254dd5de0ede94ba6999bda63b0fae9880873a3710e87a71bafb64761a"
    uri="https://rrdp.example.net/1773/delta.xml" />
</notification>
```

Using its previously recorded state (Figure 1), the RP can compare the hash values for serials 1773 and 1774. For serial 1774, compared to the earlier version of the Update Notification File, a different hash value is now listed, meaning an unexpected delta mutation occurred.

4. Recovery after Desynchronization

Following the detection of RRDP session desynchronization, the RP implementation SHOULD issue a warning and SHOULD download the latest Snapshot File and process it as described in Section 3.4.3 of [RFC8182].

5. Security Considerations

Due to the lifetime of RRDP sessions (often measured in months), desynchronization can persist for an extended period if undetected.

Caches in a desynchronized state pose a risk by emitting a different set of Validated Payloads than they would otherwise emit with a consistent repository copy. Through the interaction of the desynchronization and the _failed fetch_ mechanism described in Section 6.6 of [RFC9286], Relying Parties could spuriously omit Validated Payloads or emit Validated Payloads that the Certification Authority intended to withdraw. In a desynchronized state, all bets are off.

Missing Validated Payloads negatively impact the ability to validate BGP announcements using mechanisms such as those described in [RFC6811] and [I-D.ietf-sidrops-aspa-verification].

Section 6.6 of [RFC9286] advises RP implementations to continue to use cached versions of objects, but only until such time as they become stale. By detecting whether the remote Repository Server is in an inconsistent state and then immediately switching to using the latest Snapshot File, RPs increase the probability to successfully replace objects before they become stale.

6. IANA Considerations

No IANA actions required.

7. References

7.1. Normative References

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

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

[RFC8182] Bruijnzeels, T., Muravskiy, O., Weber, B., and R. Austein, "The RPKI Repository Delta Protocol (RRDP)", RFC 8182, DOI 10.17487/RFC8182, July 2017, <https://www.rfc-editor.org/info/rfc8182>.

7.2. Informative References

[FORT-validator]
Leiva, A., "FORT validator 1.7.0", March 2024, <https://github.com/NICMx/FORT-validator/compare/main...draft-spaghetti-sidrops-rrdp-desynchronization>.

[I-D.ietf-sidrops-aspa-verification]
Azimov, A., Bogomazov, E., Bush, R., Patel, K., Snijders, J., and K. Sriram, "BGP AS_PATH Verification Based on Autonomous System Provider Authorization (ASPA) Objects", Work in Progress, Internet-Draft, draft-ietf-sidrops-aspa-verification-18, 8 July 2024, <https://datatracker.ietf.org/doc/html/draft-ietf-sidrops-aspa-verification-18>.

[RFC6811] Mohapatra, P., Scudder, J., Ward, D., Bush, R., and R. Austein, "BGP Prefix Origin Validation", RFC 6811, DOI 10.17487/RFC6811, January 2013, <https://www.rfc-editor.org/info/rfc6811>.

[RFC7115] Bush, R., "Origin Validation Operation Based on the Resource Public Key Infrastructure (RPKI)", BCP 185, RFC 7115, DOI 10.17487/RFC7115, January 2014, <https://www.rfc-editor.org/info/rfc7115>.

[RFC9286] Austein, R., Huston, G., Kent, S., and M. Lepinski, "Manifests for the Resource Public Key Infrastructure (RPKI)", RFC 9286, DOI 10.17487/RFC9286, June 2022, <https://www.rfc-editor.org/info/rfc9286>.

[Routinator]
NLNet Labs, "Routinator", <https://github.com/NLnetLabs/routinator/pull/951>.

[rpki-client]
Jeker, C., Snijders, J., Dzonsons, K., and T. Buehler, "rpki-client 8.5", July 2023, <https://www.rpki-client.org/>.

[rpki-prover]
Puzanov, M., "rpki-prover 0.9.0", February 2024, <https://github.com/lolepezy/rpki-prover>.

Appendix A. Acknowledgements

During the hallway track at RIPE 86, Ties de Kock shared the idea for detecting this particular form of RRDP desynchronization, after which Claudio Jeker, Job Snijders, and Theo Buehler produced an implementation based on rpki-client. Equipped with tooling to detect this particular error condition, in subsequent months it became apparent that unexpected delta mutations in the global RPKI repositories do happen from time to time.

The authors wish to thank Theo Buehler, Mikhail Puzanov, Alberto Leiva, Tom Harrison, Warren Kumari, and Behcet Sarikaya for their careful review and feedback on this document.

Appendix B. Implementation status

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

This section records the status of known implementations of the protocol defined by this specification at the time of posting of this Internet-Draft, and is based on a proposal described in RFC 7942. The description of implementations in this section is intended to assist the IETF in its decision processes in progressing drafts to RFCs. Please note that the listing of any individual implementation here does not imply endorsement by the IETF. Furthermore, no effort has been spent to verify the information presented here that was supplied by IETF contributors. This is not intended as, and must not be construed to be, a catalog of available implementations or their features. Readers are advised to note that other implementations may exist.

According to RFC 7942, "this will allow reviewers and working groups to assign due consideration to documents that have the benefit of running code, which may serve as evidence of valuable experimentation and feedback that have made the implemented protocols more mature. It is up to the individual working groups to use this information as they see fit".

* OpenBSD [rpki-client] 8.5 and higher

* Mikhail Puzanov's [rpki-prover] 0.9.0 and higher

* FORT project's [FORT-validator] 1.7.0 and higher

* NLnet Labs' [Routinator] main branch

Authors' Addresses

Job Snijders
Fastly
Amsterdam
Netherlands
Email: job@fastly.com

Ties de Kock
RIPE NCC
Amsterdam
Netherlands
Email: tdekock@ripe.net

Network Working Group W. Kumari
Internet-Draft Google, LLC
Updates: 8110 (if approved) D. Harkins
Intended status: Informational Hewlett-Packard Enterprise
Expires: 8 February 2025 7 August 2024

Transferring Opportunistic Wireless Encryption to the IEEE 802.11 Working Group
draft-wkumari-rfc8110-to-ieee-02

Abstract

RFC8110 describes Opportunistic Wireless Encryption (OWE), a mode that allows unauthenticated clients to connect to a network using encrypted traffic. This document transfers the ongoing maintenance and further development of the protocol to the IEEE 802.11 Working Group.

This document updates RFC8110 by noting that future work on the protocol described in RFC8110 will occur in the IEEE 802.11 Working Group.

About This Document

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

The latest revision of this draft can be found at https://wkumari.github.io/draft-wkumari-rfc8110-to-ieee/draft-wkumari-rfc8110-to-ieee.html. Status information for this document may be found at https://datatracker.ietf.org/doc/draft-wkumari-rfc8110-to-ieee/.

Source for this draft and an issue tracker can be found at https://github.com/wkumari/draft-wkumari-rfc8110-to-ieee.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on 8 February 2025.

Copyright Notice

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

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

Table of Contents

1. Introduction

[RFC8110] describes Opportunistic Wireless Encryption (OWE), a mode of opportunistic security [RFC7435] for IEEE Std 802.11 that provides encryption of the wireless medium without authentication.

Since publication, [RFC8110] (also known as "[Wi-Fi_Enhanced_Open]") has been widely implemented and deployed.

[IEEE_802.11] has requested [IEEE_LS] that in order to allow for ongoing maintenance and further development of the protocol, and to ensure that the protocol remains in sync with the IEEE protocols, future work on the protocol described in RFC8110 will now occur in [IEEE_802.11]. This document is a concurrence.

2. Transfer of Maintenance

At the request of [IEEE_802.11], in order to allow for ongoing maintenance and further development of the protocol, and to ensure that the protocol remains in sync with the IEEE protocols, this document specifies that future work on the protocol described in RFC8110 will now occur in [IEEE_802.11].

The protocol defined in RFC8110 will be duplicated in [IEEE_802.11] such that that document alone will be enough to implement it and any further maintenance or modification of the protocol will be performed in IEEE under its policies and procedures.

3. Security Considerations

This document simply notes that future work on the protocol described in RFC8110 will now occur in the IEEE. As such, it does not introduce any new security considerations.

4. IANA Considerations

This document has no IANA actions.

5. References

5.1. Normative References

[RFC8110] Harkins, D., Ed. and W. Kumari, Ed., "Opportunistic Wireless Encryption", RFC 8110, DOI 10.17487/RFC8110, March 2017, <https://www.rfc-editor.org/rfc/rfc8110>.

5.2. Informative References

[IEEE_802.11] "IEEE 802.11 Working Group", n.d., <https://www.ieee802.org/11/>.

[IEEE_LS] "Liaison Statement from IEEE 802.11 to the IETF - OWE (RFC8110) now in 802.11", n.d., <https://datatracker.ietf.org/liaison/1929/>.

[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>.

[Wi-Fi_Enhanced_Open]
"Wi-Fi CERTIFIED Enhanced Openâ\204¢: Transparent Wi-FiÂ® protections without complexity", n.d., <https://www.wi-fi.org/beacon/dan-harkins/wi-fi-certified-enhanced-open-transparent-wi-fi-protections-without-complexity>.

Acknowledgments

The authors would like to thank the IEEE 802.11 working group for their work, and for taking on the responsibility for future work on the protocol described in RFC8110.

In addition, we would like to thank Stephen Farrell, who AD sponsored the original work, as well as Clemens Schimpe, Dorothy Stanley, Paul Wouters, Eric Vyncke, Mike Montemurro, and Peter Yee.

Apologies to anyone we forgot to acknowledge; RFC8110 was written 7+ years ago and we have had many conversations with many people since then...

Change Log

* From -00 to -01:

  - Fixed a nit ("This documents updates" -> "This document updates")

  - We have the liaison from the IEEE 802.11 WG; update to point at the liaison statement.

  - For some reason, pushing the -01 version to GitHub didn’t trigger the build. Trying to post manually.

Authors’ Addresses

Warren Kumari
Google, LLC
Email: warren@kumari.net

Dan Harkins
Hewlett-Packard Enterprise
Email: daniel.harkins@hpe.com