Workgroup: Network Working Group
Internet-Draft: draft-dekater-scion-pki-02
Published: 28 February 2023
Intended Status: Informational
Expires: 1 September 2023
Authors: C. de Kater
SCION Association
N. Rustignoli
SCION Association

SCION Control-Plane PKI

Abstract

This document presents the trust concept and design of the SCION *control-plane Public Key Infrastructure (PKI)*, SCION's public key infrastructure model. SCION (Scalability, Control, and Isolation On Next-generation networks) is a path-aware, inter-domain network architecture. The control-plane PKI, or short CP-PKI, handles cryptographic material and lays the foundation for the authentication procedures in SCION. It is used by SCION's control plane to authenticate and verify path information, and builds the basis for SCION's special trust model based on so-called Isolation Domains.

This document first introduces the trust model behind the SCION's control-plane PKI, as well as clarifications to the concepts used in it. This is followed by specifications of the different types of certificates and the Trust Root Configuration. The document then specifies how to deploy the whole infrastructure.

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://scionassociation.github.io/scion-cppki_I-D/draft-dekater-scion-pki.html. Status information for this document may be found at https://datatracker.ietf.org/doc/draft-dekater-scion-pki/.

Source for this draft and an issue tracker can be found at https://github.com/scionassociation/scion-cppki_I-D.

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 September 2023.

Table of Contents

1. Introduction

The control-plane PKI (CP-PKI) lays the foundation for the authentication procedures in SCION. It handles all cryptographic material used in the public key infrastructure of SCION's control plane. This section first introduces the key concepts of the SCION CP-PKI, including the trust model, its core elements (certificates, keys, and roles), and their relationships. The sections after the Introduction provide detailed specifications of the building blocks of the CP-PKI.

Note: For extended information on the SCION next-generation inter-domain architecture, see [CHUAT22], especially Chapter 2, as well as the IETF Internet Drafts [I-D.scion-overview] and [I-D.scion-components].

1.1. Conventions and Definitions

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.

1.2. Trust Model

Given the diverse nature of the constituents in the current Internet, an important challenge is how to scale authentication of network elements (such as AS ownership, hop-by-hop routing information, name servers for DNS, and domains for TLS) to the

global environment. The roots of trust of currently prevalent public key infrastructure (PKI) models do not scale well to a global environment, because (1) mutually distrustful parties cannot agree on a single trust root (monopoly model), and because (2) the security of a plethora of roots of trust is only as strong as its weakest link (oligopoly model) - see also [BARRERA17].

The monopoly model suffers from two main drawbacks: First, all parties must agree on a single root of trust. Secondly, the single root of trust represents a single point of failure, the misuse of which enables the forging of certificates. Also, its revocation can result in a kill-switch for all the entities it certifies. The oligopoly model relies on several roots of trust, all equally and completely trusted. However, this is not automatically better: Whereas the monopoly model has a single point of failure, the oligopoly model has the drawback of exposing more than one point of failure.

Thus, there is a need for a trust architecture that supports meaningful trust roots in a global environment with inherently distrustful parties. This new trust architecture should provide the following properties:

*Trust agility (see further below);

*Resilience to single root of trust compromise;

*Multilateral governance; and

*Support for policy versioning and updates.

Ideally, the trust architecture allows parties that mutually trust each other to form their own trust "union" or "domain", and to freely decide whether to trust other trust unions (domains) outside their own trust bubble.

To fulfill the above requirements, which in fact apply well to inter-domain networking, SCION introduces the concept of **Isolation Domains**. An Isolation Domain (ISD) is a building block for achieving high availability, scalability, and support for heterogeneous trust. It consists of a logical grouping of ASes that share a uniform trust environment (i.e., a common jurisdiction). An ISD is administered by one or multiple ASes, called the **voting ASes**. Furthermore, each ISD has a set of ASes that form the ISD core; these are the **core ASes**. The set of core and voting ASes can, but not necessarily have to, overlap. It is governed by a policy called the **Trust Root Configuration** (TRC), which is negotiated by the ISD core. The TRC defines the locally scoped roots of trust used to validate bindings between names and public keys.

Authentication in SCION is based on digital certificates that bind identifiers to public keys and carry digital signatures that are verified by roots of trust. SCION allows each ISD to define its own

set of trust roots, along with the policy governing their use. Such scoping of trust roots within an ISD improves security, as compromise of a private key associated with a trust root cannot be used to forge a certificate outside the ISD. An ISD's trust roots and policy are encoded in the TRC, which has a version number, a list of public keys that serves as root of trust for various purposes, and policies governing the number of signatures required for performing different types of actions. The TRC serves as a way to bootstrap all authentication within SCION. Additionally, TRC versioning is used to efficiently revoke compromised roots of trust.

The TRC also provides *trust agility*, that is, it enables users to select the trust roots used to initiate certificate validation. This implies that users are free to choose an ISD they believe maintains a non-compromised set of trust roots. ISD members can also decide whether to trust other ISDs and thus transparently define trust relationships between parts of the network. The SCION trust model, therefore, differs from the one provided by other PKI architectures.

1.3. Trust Relations within an Isolation Domain

As already mentioned previously, the control-plane PKI, SCION's concept of trust, is organized on ISD-level. Each ISD can independently specify its own Trust Root Configuration (TRC) and build its own verification chain. Each TRC consists of a collection of signed root certificates, which are used to sign CA certificates, which are in turn used to sign AS certificates. The TRC also includes ISD-policies that specify, for example, the TRC's usage, validity, and future updates. A TRC is a fundamental component of an ISD's control-plane PKI. The so-called **base TRC** constitutes the ISD's trust anchor. It is signed during a signing ceremony by the voting ASes and then distributed throughout the ISD.

1.3.1. Updates and Trust Resets

There are two types of TRC updates: regular and sensitive. A **regular TRC update** is a periodic re-issuance of the TRC where the entities and policies listed in the TRC remain unchanged, whereas a **sensitive TRC update** is an update that modifies critical aspects of the TRC, such as the set of core ASes. In both cases, the base TRC remains unchanged. If the ISD's TRC has been compromised, it is necessary for an ISD to re-establish the trust root. This is possible with a process called **trust reset** (if allowed by the ISD's trust policy). In this case, a new base TRC is created.

1.4. Overview of Certificates, Keys, and Roles

The base TRC constitutes the root of trust within an ISD. Figure 1 provides a first impression of the trust chain within an ISD, based on its TRC. For detailed descriptions, please refer to Section 3 and Section 4.

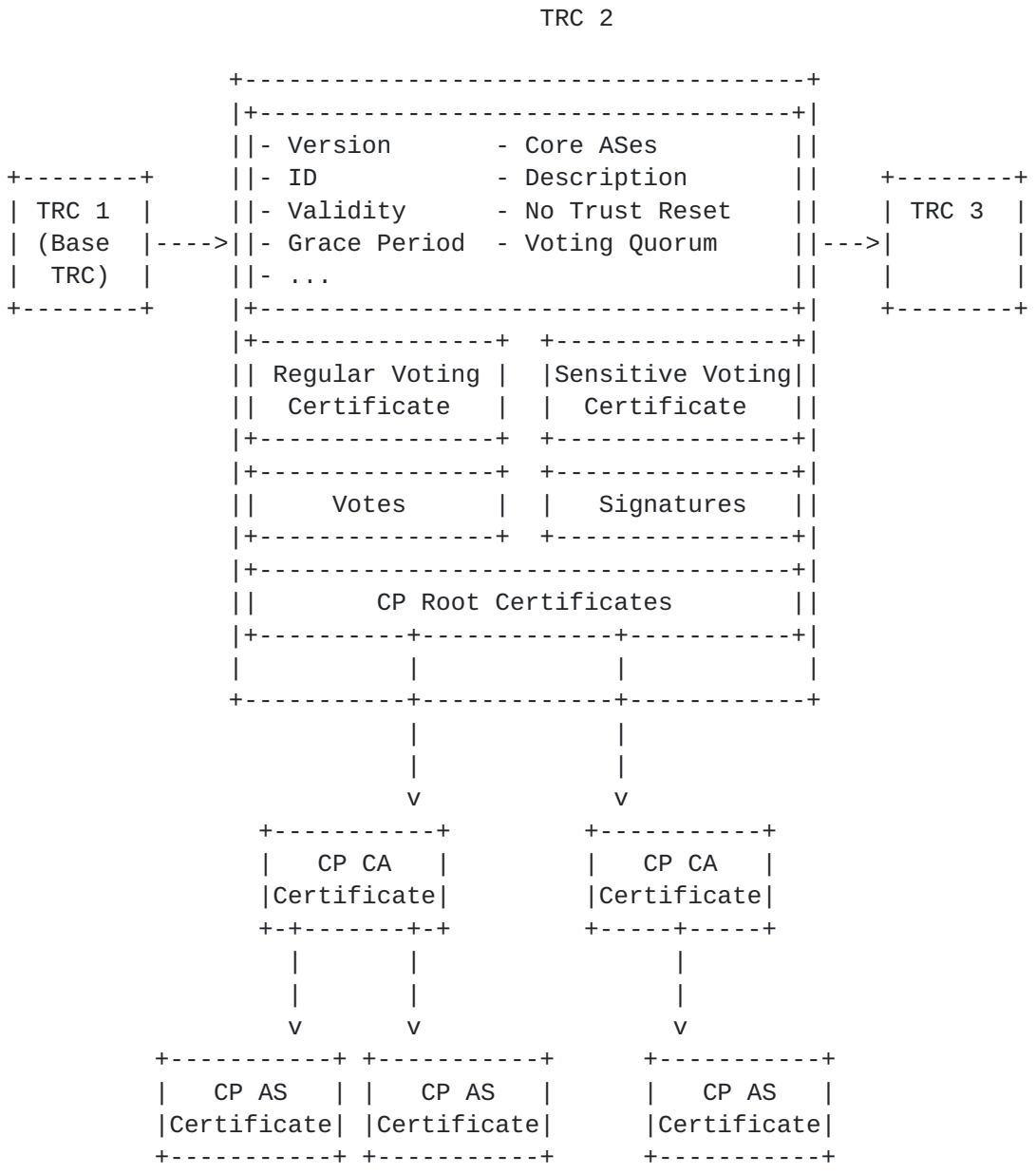


Figure 1: Chain of trust within an ISD

All certificates used in SCION's control-plane PKI are in X.509 v3 format [RFC5280]. Additionally, the TRC contains self-signed certificates instead of plain public keys. Self-signed certificates have the following advantages over plain public keys: (1) They make the binding between name and public key explicit; and (2) the binding is signed to prove possession of the corresponding private key.

All ASes in SCION have the task to sign and verify control-plane messages. However, certain ASes have additional roles:

* **Core ASes**: Core ASes are a distinct set of ASes in the SCION control plane. For each ISD, the core ASes are listed in the TRC.

Each core AS in an ISD has links to other core ASes (in the same or in different ISDs).

* **Certification authorities (CAs)**: CAs are responsible for issuing AS certificates to other ASes and/or themselves.

* **Voting ASes**: Only certain ASes within an ISD may sign TRC updates. The process of appending a signature to a new TRC is called "casting a vote"; the designated ASes that hold the private keys to sign a TRC update are "voting ASes".

* **Authoritative ASes**: Authoritative ASes are those ASes in an ISD that always have the latest TRCs of the ISD. They start the announcement of a TRC update.

All further details of the SCION control-plane PKI are specified in the following sections.

1.5. Trust as a Function

The SCION control-plane PKI can be seen as a function that transforms potential distrust among different parties into a mutually accepted trust contract including a trust update and reset policy as well as certificates used for authentication procedures in SCION's control plane.

For the function to work, it is not necessary that the ASes of the ISD all trust each other. However, all ASes **MUST** trust the ISD's core ASes, authoritative ASes, voting ASes, as well as its CA(s). These trusted parties negotiate the ISD trust contract in a "bootstrapping of trust" ceremony, where cryptographic material is exchanged, and the ISD's trust anchor (the initial Trust Root Configuration) is created and signed.

1.5.1. Input

Prior to the ceremony, the trusted parties must decide about the validity period of the TRC as well as the number of votes required to update a TRC. They must also bring the required keys and certificates, the so-called root and voting keys/certificates.

During the ceremony, the trusted parties decide about the number of the ISD. This must be an integer in the inclusive range between 64 and 4094. The next table shows the current allocation of ISD numbers in SCION:

ISD	Description
0	The wildcard ISD.
1 - 15	Reserved for documentation and sample code (analogous to [RFC5398].
16 - 63	Private use (analogous to [RFC6996]). Can be used for testing and private deployments.

ISD	Description
64 - 4094	Public ISDs. Should be allocated in ascending order, without gaps and "vanity" numbers.
4095 - 65535	Reserved for future use.

Table 1: ISD Number Allocations

1.5.2. Output

The output of the bootstrapping of trust ceremony, or the trust "function", are the ISD's initial Trust Root Configuration as well as mutually trusted and accepted CA and AS certificates--the latter are used to verify SCION's control-plane messages. Together with the ISD's control-plane root certificates, the CA and AS certificates build the ISD's trust and verification chain.

2. Terminology

Control-Plane PKI (CP-PKI): The control-plane PKI is the public-key infrastructure upon which SCION's control plane relies for the authentication of messages. It is a set of policies, roles, and procedures that are used to manage trust root configurations (TRCs) and certificates.

Autonomous System (AS): An autonomous system is a network under a common administrative control. For example, the network of an Internet service provider, company, or university can constitute an AS. If an organization operates multiple networks that are not directly connected together, then the different networks are considered different ASes.

Isolation Domain (ISD): In SCION, autonomous systems (ASes) are organized into logical groups called isolation domains or ISDs. Each ISD consists of ASes that span an area with a uniform trust environment (i.e., a common jurisdiction). A possible model is for ISDs to be formed along national boundaries or federations of nations.

Core AS: Each isolation domain (ISD) is administered by a set of distinguished autonomous systems (ASes) called core ASes, which are responsible for initiating the path-discovery and -construction process (in SCION called "beaconing").

Trust Root Configuration (TRC): A trust root configuration or TRC is a signed collection of certificates pertaining to an isolation domain (ISD). TRCs also contain ISD-specific policies.

Authoritative AS: Authoritative ASes are those ASes in an ISD that always have the latest TRCs of the ISD. As a consequence, authoritative ASes also start the announcement of a TRC update.

Base TRC: A base TRC is a trust root configuration (TRC) that other parties trust axiomatically. In other words, trust for a base TRC is assumed, not derived from another cryptographic object. Each ISD must create and sign a base TRC when the ISD is established. A base TRC is either the first TRC of the ISD or the result of a trust reset.

TRC Signing Ceremony: The ceremony during which the very first base TRC of an ISD, called the initial TRC, is signed. The initial TRC is a special case of the base TRC where the number of the ISD is assigned.

TRC Update: A *regular* TRC update is a periodic re-issuance of the TRC where the entities and policies listed in the TRC remain unchanged. A *sensitive* TRC update is an update that modifies critical aspects of the TRC, such as the set of core ASes. In both cases, the base TRC remains unchanged.

Voting ASes: Those ASes within an ISD that may sign TRC updates. The process of appending a signature to a new TRC is called "casting a vote".

Voting Quorum: The voting quorum is a trust root configuration (TRC) field that indicates the number of votes (signatures) needed on a successor TRC for it to be verifiable. A voting quorum greater than one will thus prevent a single entity from creating a malicious TRC update.

Grace Period: The grace period is an interval during which the previous version of a trust root configuration (TRC) is still considered active after a new version has been published.

Trust Reset: A trust reset is the action of announcing a new base TRC for an existing ISD. A trust reset should only be triggered after a catastrophic event involving the loss or compromise of several important private keys.

3. Certificate Specification

This section provides a detailed specification of all certificates used in SCION's control-plane PKI. It starts with an overview of the main keys and certificates.

3.1. SCION Control-Plane PKI Keys and Certificates - Overview

There are three types of control-plane (CP) certificates: root certificates, CA certificates, and AS certificates. Together, they build a chain of trust that is anchored in the trust root configuration (TRC) file of the respective Isolation Domain (ISD). Additionally, there are regular and sensitive voting certificates, which define the keys to cast votes in a regular and a sensitive TRC update, respectively.

All certificates in SCION's control-plane PKI are in X.509 v3 format [RFC5280].

The next section shows SCION's trust hierarchy. This is followed by sections that describe the main certificates and corresponding key pairs of SCION's control-plane PKI as well as the voting certificates and keys.

3.1.1. Trust Hierarchy

The trust is anchored in the Trust Root Configuration (TRC) for each ISD. The trust root is axiomatic: All trust derived from this anchor relies on all parties transitively trusting the TRC.

The trust hierarchy looks like this:

```
TRC
── Regular Voting Certificates
     └── TRC (next version, regular update)
── Sensitive Voting Certificates
     └── TRC (next version, sensitive update)
── CP Root Certificates
     └── CP CA Certificates
          └── CP AS Certificates
```

3.1.2. Control-Plane Root Certificate

The control-plane root private key is used to sign control-plane CA certificates. Consequently, the control-plane root certificate with the control-plane root public key is used to verify control-plane CA certificates, i.e., root certificates determine which ASes act as CA in an ISD.

In X.509 terms, CP root certificates are *self-signed* CA certificates. That is, issuer and subject of the certificate are the same entity, and the public key in the root certificate can be used to verify the root certificate's signature. The CP root public key and proof of ownership of the private key are embedded in the Trust Root Configuration (TRC) of an Isolation Domain (ISD), via the self-signed CP root certificate. This facilitates the bootstrapping of trust within an ISD, and marks the CP root certificates as the starting point of an ISD's certificate verification path.

The recommended **maximum validity period** of a CP root certificate is: 1 year.

Note: The TRC of each ISD contains a trusted set of control-plane root certificates. This set builds the root of each ISD's verification path. For more information on the selection of this trusted set of root certificates, see Section 4.

3.1.3. Control-Plane CA Certificate

The control-plane CA private key is used to sign control-plane AS certificates. Consequently, control-plane CA certificates holding the control-plane CA public key are used to verify control-plane AS certificates.

The public key needed to verify the CA certificate is in a CP root certificate. CA certificates do not bundle the root certificate needed to verify them. In order to verify a CA certificate, a pool of root certificates must first be extracted from one or more active TRCs (as described in Section 5.2).

The recommended **maximum validity period** of a CP CA certificate is: 11 days.

3.1.4. Control-Plane AS Certificate

SCION ASes sign control-plane messages, such as Path Construction Beacons, with their AS private key. Consequently, control-plane AS certificates holding the corresponding AS public key are required to verify control-plane messages.

In X.509 terms, control-plane AS certificates are end-entity certificates. That is, they cannot be used to verify other certificates.

The recommended **maximum validity period** of a CP AS certificate is: 3 days.

3.1.5. Voting Certificates

There are two types of voting certificates: the (1) regular voting certificates and the (2) sensitive voting certificates. They contain the public keys associated with the private keys that are allowed to cast votes in the TRC update process. Voting certificates are X.509-style certificates.

Regular and sensitive voting certificates are used to verify regular and sensitive TRC updates, respectively, and are embedded in the TRC.

3.1.5.1. Regular Voting Certificate

Regular voting certificates state which keys are allowed to cast votes in a regular update. In X.509 terms, regular voting certificates are self-signed end-entity certificates. This means that the issuer and subject of a regular voting certificate are the same entity, and the public key within the certificate can be used to verify the certificate's signature. However, a regular voting certificate cannot be used to verify other certificates.

The recommended **maximum validity period** of a regular voting certificate is: 1 year.

3.1.5.2. Sensitive Voting Certificate

Sensitive voting certificates specify which keys are allowed to cast votes in a sensitive update. In X.509 terms, sensitive voting certificates are self-signed end-entity certificates. This means that the issuer and subject of a sensitive voting certificate are the same entity, and the public key within the certificate can be used to verify the certificate's signature. However, a sensitive voting certificate cannot be used to verify other certificates.

The recommended **maximum validity period** of a sensitive voting certificate is: 5 years.

Note:

Both SCION's control-plane root certificates and control-plane CA certificates are in fact CA certificates. That is, they can both be used to verify other certificates.

One important difference between both certificate types lies in their validity period: A SCION control-plane root certificate has a recommended maximum validity period of one year, whereas the recommended maximum validity period of a SCION control-plane CA certificate is 11 days. This is because a root certificate is part of the Trust Root Configuration of an ISD, which itself also has a recommended maximum validity period of one year (see Table 2 below). This ensures that the TRC must not be updated all the time and is thus relatively stable.

The SCION root private key and public key/certificate are used to sign and verify the control-plane CA certificates, respectively. The control-plane CA certificates are explicitly NOT part of the TRC, for reasons of security. The control-plane CA certificates are used to verify the control-plane AS certificates, which in turn are used to verify control-plane messages. Routing is made more secure if both the SCION control-plane CA and AS certificates can be renewed on a very regular basis. Would the control-plane CA and AS certificates be part of the TRC, then the TRC would have to be updated constantly, which is undesirable.

3.1.6. Certificates - Formal Overview

Table 2 and Table 3 below provide a formal overview of the different types of key pairs and certificates in the control-plane PKI.

Name	Notation (1)	Used to verify/sign
Sensitive voting key	K_{sens}	TRC updates (sensitive)
Regular voting key	K_{reg}	TRC updates (regular)
CP root key	K_{root}	CP CA certificates
CP CA key	K_{CA}	CP AS certificates
CP AS key	K_{AS}	CP messages, path segments

Table 2: Key chain

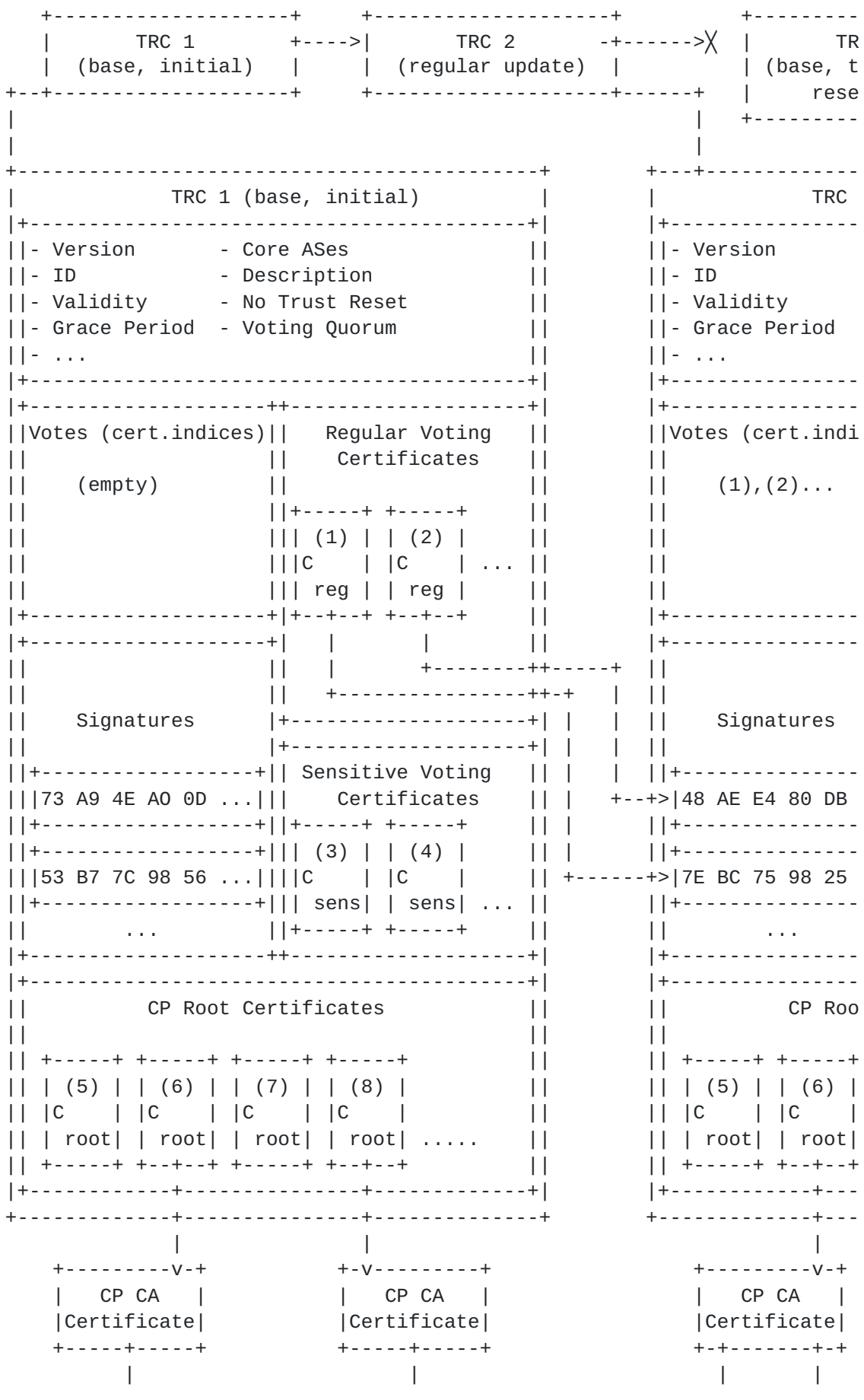
(1) $K_x = PK_x + SK_x$, where x = certificate type, PK_x = public key, and SK_x = private key

Name	**Notation**	**Signed with**	**Contains**	**Validity (2)**
TRC (trust root conf.)	TRC	SK_{sens}, SK_{reg} (1)	C_{root}, C_{sens}, C_{reg} (1)	1 year
Sensitive voting cert.	C_{sens}	SK_{sens}	PK_{sens}	5 years
Regular voting cert.	C_{reg}	SK_{reg}	PK_{reg}	1 year
CP root certificate	C_{root}	SK_{root}	PK_{root}	1 year
CP CA certificate	C_{CA}	SK_{root}	PK_{CA}	11 days (3)
CP AS certificate	C_{AS}	SK_{CA}	PK_{AS}	3 days

Table 3: Certificates

(1) Multiple signatures and certificates of each type may be included in a TRC.
(2) Recommended maximum validity period.
(3) A validity of 11 days with 4 days overlap between two CA certificates is recommended to enable best possible operational procedures when performing a CA certificate rollover.

Figure 2 illustrates, at a high level, the relationship between a TRC and the five types of certificates.


TRC 1
(base, initial)
TRC 2
(regular update)
TR
(base, t
rese
TRC 1 (base, initial)
- Version
- ID
- Validity
- Grace Period
- ...
- Core ASes
- Description
- No Trust Reset
- Voting Quorum
Votes (cert.indices)
(empty)
Regular Voting
Certificates
(1)
(2)
reg
reg
Signatures
Sensitive Voting
Certificates
(3)
(4)
sens
sens
...
CP Root Certificates
(5)
(6)
(7)
(8)
root
root
root
root
.....
CP CA
Certificate
TRC
Votes (cert.indi
(1),(2)...
Signatures
73 A9 4E AO 0D ...
53 B7 7C 98 56 ...
48 AE E4 80 DB
7E BC 75 98 25
CP Roo
(5)
(6)
root
root
CP CA
Certificate

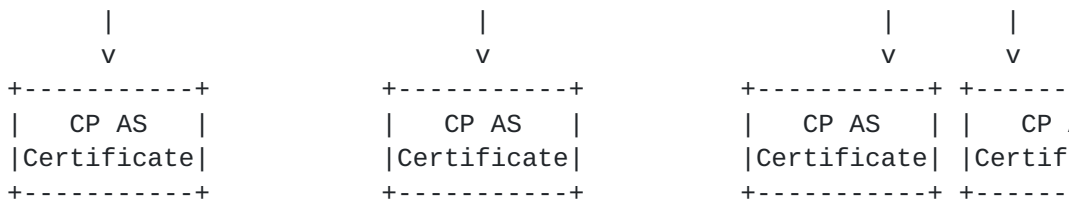

CP AS
Certificate
CP AS
Certificate
CP AS
Certificate
CP
Certif

Figure 2: TRC update chain and the different types of associated certificates. Arrows show how signatures are verified; in other words, they indicate that a public key contained in a certificate or TRC can be used to verify the authenticity of another item.

3.2. Certificate Specification

All certificates used in the SCION control-plane PKI are X.509 v3 certificates. However, the SCION specification is in some places more restrictive. This section defines these additional constraints and conditions compared to [RFC5280] for each type of SCION control-plane PKI certificate.

Note: The settings for the SCION-specific constraints and conditions are based on the SCION open-source implementation scionproto. Adjusting these settings to the requirements of a customer implementation may be possible and is allowed.

3.2.1. Basic Fields: SCION-Specific Constraints and Conditions

This section briefly describes the fields of the SCION control-plane PKI certificates based on X.509. These fields are relevant for each SCION certificate used in the control plane, regardless of the certificate type. For detailed descriptions of the full generic format of X.509 v3 certificates, see [RFC5280] and X509, clause 7.2. Additionally, the section lists the SCION-specific constraints and conditions compared to [RFC5280], per certificate field.

TBSCertificate sequence: Contains information associated with the subject of the certificate and the CA that issued it. It includes the following fields:

* version field: Describes the version of the encoded certificate.
  - **SCION constraints**: "v1" and "v2" are not allowed.
  - **Additional conditions and remarks**: **MUST** be set to "v3" (as extensions are used and mandatory in SCION).
* serialNumber field: A positive integer assigned by the CA to each certificate. It **MUST** be unique for each certificate issued by a given CA.
* signature field: Contains the identifier for the algorithm used by the CA to sign the certificate.
  - **SCION constraints**: Currently, SCION only supports the ECDSA signature algorithm. Find all details here: Section 3.2.1.1.
  - **Additional conditions and remarks**: As a consequence, the parameters field in the AlgorithmIdentifier sequence **MUST NOT** be used.

*issuer field: Contains the distinguished name (DN) of the entity that has issued and signed the certificate (usually a CA).

  - **SCION constraints**:

    oThis field **MUST** be non-empty.

    oSCION implementations **MUST** ONLY use the "UTF8String" value type for all attributes (including the SCION-specific attribute ISD-AS number).

  - **Additional conditions and remarks**: All SCION implementations **MUST** support the additional SCION-specific attribute ISD-AS number. For details, see Section 3.2.1.2 and Section 3.2.1.2.1.

*validity field: Defines the validity period of the certificate.

  - **SCION constraints**: All certificates used in SCION's control-plane PKI **MUST** have a well-defined expiration date. Certificates with a generalized time value are not valid and **MUST** be rejected.

  - **Additional conditions and remarks**: SCION recommends a specific maximum validity period for each type of control-plane PKI certificate. For details, see Section 3.1.6. SCION implementations should adopt these values.

*subject field: Defines the entity that owns the certificate.

  - **SCION constraints**:

    oThis field **MUST** be non-empty.

    oSCION implementations **MUST** ONLY use the "UTF8String" value type for all attributes (including the SCION-specific attribute ISD-AS number).

  - **Additional conditions and remarks**: The subject field is specified in the same way as the issuer field. For details, see Section 3.2.1.2 and Section 3.2.1.2.1.

*subjectPublicKeyInfo field: Carries the public key of the certificate's subject (the entity that owns the certificate, as defined in the subject field). The subjectPublicKeyInfo field also identifies which algorithm to use with the key.

  - **SCION constraints**: For constraints regarding the algorithm, see the signature field.

*issuerUniqueID field: If set, it enables reusing the issuer name over time.

  -**SCION constraints**: This field is disallowed in SCION and **MUST NOT** be used.

*subjectUniqueID field: If set, it enables reusing the subject name over time.

  -**SCION constraints**: This field is disallowed in SCION and **MUST NOT** be used.

*extensions sequence: Defines the extensions of the certificate. For a description of all extensions used in SCION, see Section 3.2.2.

3.2.1.1. signature Field - Additional Information

For security reasons, SCION uses a custom list of acceptable signature algorithms. This list of acceptable signature algorithms is specified in the signature field. The list currently only contains the ECDSA signature algorithm (defined in X962). However, the list might be extended in the future. The Object Identifiers (OIDs) for ECDSA are defined as ecdsa-with-SHA256, ecdsa-with-SHA384, and ecdsa-with-SHA512 in [RFC5758].

Important: The accepted cryptographic algorithms listed in this document are the only currently accepted cryptographic algorithms. SCION implementations **MUST** reject cryptographic algorithms not found in the list.

The only accepted curves for ECDSA are:

*NIST P-256 (NISTFIPS186-4, section D.1.2.3) (named secp256r1 in [RFC5480])

*NIST P-384 (NISTFIPS186-4, section D.1.2.4) (named secp384r1 in [RFC5480])

*NIST P-521 (NISTFIPS186-4, section D.1.2.5) (named secp521r1 in [RFC5480])

The OIDs for the above curves are specified in section 2.1.1.1 of [RFC5480].

The appropriate hash size to use when producing a signature with an ECDSA key is:

*ECDSA with SHA-256, for a P-256 signing key

*ECDSA with SHA-384, for a P-384 signing key

*ECDSA with SHA-512, for a P-521 signing key

Important: SCION implementations **MUST** include support for P-256, P-384, and P-521.

3.2.1.2. issuer Field - Additional Information

The issuer field contains the distinguished name (DN) of the CA that created the certificate. [RFC5280], section 4.1.2.4, describes the field's syntax and attributes. In addition to these attributes, SCION implementations **MUST** also support the SCION-specific attribute ISD-AS number. This attribute is specified below.

3.2.1.2.1. ISD-AS number Attribute

The ISD-AS number attribute identifies the SCION ISD and AS. In the SCION open source implementation, the attribute type is id-at-ia, defined as:
id-at-ia AttributeType ::= {id-scion id-cppki(1) id-at(2) 1}

where id-scion specifies the root SCION object identifier (OID).

Note: The root SCION object identifier (OID) for the SCION open-source implementation is the IANA Private Enterprise Number '55324':
id-scion ::= OBJECT IDENTIFIER {1 3 6 1 4 1 55324}

The following points apply when setting the attribute value of the ISD-AS number attribute:

- The string representation **MUST** follow the canonical formatting defined in ISD and AS numbering.
- The canonical string representation uses a dash separator between the ISD and AS numbers.
- The ISD numbers are formatted as decimal.
- The canonical string formatting of AS numbers in the BGP AS range $(0, 2^{32-1})$ is the decimal form. Larger AS numbers, i.e., from 2^{32} to 2^{48-1}, use a 16-bit, colon-separated, lower-case, hex encoding with leading zeros omitted: 1:0:0 to ffff:ffff:ffff.

Example: AS ff00:0:110 in ISD 1 is formatted as 1-ff00:0:110.

The ISD-AS number attribute **MUST** be present exactly once in the distinguished name of the certificate issuer or owner, specified in the issuer or subject field, respectively. Implementations **MUST NOT** create nor successfully verify certificates whose issuer and subject fields do not include the ISD-AS number at all, or include it more than once.

Note: Voting certificates are not required to include the ISD-AS number attribute in their distinguished name.

3.2.2. Extensions

[RFC5280], section 4.2.1, defines the syntax of the Extensions sequence in a X.509 certificate. Descriptions of each standard certificate extension can be found in [RFC5280], section 4.2.1. The corresponding clauses in X509 (10/2016) are clause 7.2 and clause 9, respectively.

Currently, the following extensions are relevant for SCION:

- *authorityKeyIdentifier
- *subjectKeyIdentifier
- *keyUsage
- *extKeyUsage
- *basicConstraints

The following sections describe the SCION-specifics in regard to these extensions.

3.2.2.1. authorityKeyIdentifier Extension

The authorityKeyIdentifier extension identifies the public key corresponding to the private key used to sign a certificate.

For the syntax and definition of the authorityKeyIdentifier extension, see [RFC5280], section 4.2.1.1, and X509, clause 9.2.2.1.

The authorityKeyIdentifier extension provides three attributes to specify the public key:

- *keyIdentifier
- *authorityCertIssuer
- *authorityCertSerialNumber

In SCION, using the keyIdentifier attribute is the preferred way to specify the authorityKeyIdentifier extension.

Important: SCION implementations may also support the use of the authorityCertIssuer and authorityCertSerialNumber attributes. However, if these attributes are set and support for them is missing, implementations should error out.

This extension **MUST** always be non-critical. However, SCION implementations **MUST** error out if the extension is not present AND the certificate is not self-signed.

3.2.2.2. subjectKeyIdentifier Extension

The subjectKeyIdentifier extension identifies certificates that contain a particular public key. It can be used, for example, by control-plane messages to identify which certificate to use for verification. The extension allows for overlapping control-plane CA keys, for example during updates.

For the syntax and definition of the subjectKeyIdentifier extension, see [RFC5280], section 4.2.1.2, and X509, clause 9.2.2.2.

This extension **MUST** always be non-critical. However, SCION implementations **MUST** error out if the extension is not present.

3.2.2.3. keyUsage Extension

The keyUsage extension identifies the intended usage of the public key in the corresponding certificate. For the syntax and definition of the keyUsage extension, see [RFC5280], section 4.2.1.3, and X509, clause 9.2.2.3.

The attributes of the keyUsage extension define possible ways of using the public key. The attributes have the following meaning in SCION:

- *digitalSignature: The public key can be used to verify the digital signature of a control-plane payload.
- *contentCommitment: Not used.
- *keyEncipherment: Not used.
- *dataEncipherment: Not used.
- *keyAgreement: Not used.
- *keyCertSign: The public key can be used to verify the CA signature on a control-plane certificate.
- *cRLSign: Not used.
- *encipherOnly: Not used.
- *decipherOnly: Not used.

Important: If a certificate’s public key is used to verify the signature of a control-plane payload (digitalSignature attribute), it must be possible to trace back the private key used to sign the certificate. This is done by referencing the ISD-AS and the subject key identifier (via the subjectKeyIdentifier extension). For more information about the subjectKeyIdentifier extension, see Section 3.2.2.2.

If present, the keyUsage extension should be marked as "critical". That is, the critical Boolean attribute of this extension must be set to TRUE (the default is FALSE).

Note: If a certificate extension is marked "critical", the public key in the certificate should only be used for the purpose set in the critical extension.

Each control-plane PKI certificate type uses the public key differently, and consequently also specifies the attributes of the keyUsage extension differently. The next table shows the specifications per certificate type.

Certificate Type	**Root**	**CA**	**AS**	**Voting (regular and sensitive)**
Attribute:				
keyUsage extension itself	**MUST** be present	**MUST** be present	**MUST** be present	**MAY** be present (but is not required)
digitalSignature	**MUST NOT** be set (1)	**MUST NOT** be set (2)	**MUST** be set	If the extension is present, the digitalSignature attribute **MUST NOT** be set
keyCertSign	**MUST** be set	**MUST** be set	**MUST NOT** be set	If the extension is present, the keyCertSign attribute **MUST NOT** be set

Table 4: keyUsage extension - Specifications per certificate type

(1) The root certificate should not be used to verify control-plane messages.
(2) The CA certificate should not be used to verify control-plane messages.

3.2.2.4. extKeyUsage Extension

The extKeyUsage extension specifies additional usages of the public key in the certificate. For the syntax and definition of the extKeyUsage extension, see X509, clause 9.2.2.4.

SCION uses the following attributes of the Extended Key Usage extension, as defined in Section 4.2.1.12 of [RFC5280]:

- *id-kp-serverAuth: If set, the public key can be used for SCION control-plane server authentication.

- *id-kp-clientAuth: If set, the public key can be used for SCION control-plane client authentication.

- *id-kp-timeStamping: If set, the public key can be used for the verification of timestamps.

Additionally, the Extended Key Usage extension sequence may include the SCION-specific attributes id-kp-root, id-kp-regular, and id-kp-sensitive. These attributes are used in the Trust Root Configuration setup, to distinguish root certificates, regular voting certificates, and sensitive voting certificates from each other. For more information, see Section 4.1.2.2.11.

The specifications of the extKeyUsage extension differ per SCION control-plane PKI certificate type. The next table provides an overview of the specifications per certificate type.

Certificate Type	**Root**	**CA**	**AS**	**Voting (regular and sensitive)**
Attribute:				
extKeyUsage extension itself	**MUST** be present	**MAY** be present (not required)	**MUST** be present	**MUST** be present
id-kp-serverAuth	**MUST NOT** be included	**MUST NOT** be included	**MUST** be included, if the certificate is used on the server-side of a control-plane TLS session.	**MUST NOT** be included
id-kp-clientAuth	**MUST NOT** be included	**MUST NOT** be included	**MUST** be included, if the certificate is used on the client-side of a control-plane TLS session.	**MUST NOT** be included
id-kp-timeStamping	**MUST** be included		**MUST** be included	**MUST** be included
SCION-specific	id-kp-root **MUST** be included. For details, see Section 3.2.2.4.1			Regular voting cert: id-kp-regular **MUST** be included. For details, see Section 3.2.2.4.1 Sensitive voting cert: id-kp-sensitive **MUST** be included. For details, see Section 3.2.2.4.1

Table 5: extKeyUsage extension - Specifications per certificate type

3.2.2.4.1. SCION-Specific Attributes

The id-kp-root, id-kp-regular, and id-kp-sensitive attributes must be specified as follows:

- *Root certificate:
  id-kp-root AttributeType ::= {id-scion id-cppki(1) id-kp(3) 3}

- *Regular voting certificate:
  id-kp-regular AttributeType ::= {id-scion id-cppki(1) id-kp(3) 2}

- *Sensitive voting certificate:
  id-kp-sensitive AttributeType ::= {id-scion id-cppki(1) id-kp(3) 1}

where id-scion specifies the root SCION object identifier (OID).

Note: The root SCION object identifier (OID) for the SCION open-source implementation is the IANA Private Enterprise Number '55324': id-scion ::= OBJECT IDENTIFIER {1 3 6 1 4 1 55324}

3.2.2.5. basicConstraints Extension

The basicConstraints extension specifies whether the certificate subject may act as a CA. For the syntax and definition of the basicConstraints extension, see X509, clause 9.4.2.1.

The basicConstraints extension includes the following attributes relevant for SCION:

- *cA attribute: Specifies whether the certificate subject may act as a CA. If yes, this attribute **MUST** be set to TRUE.

- *pathLenConstraint attribute: This attribute is only relevant if the cA attribute is set to TRUE. It specifies the maximum number of CA certificates that may follow this CA certificate in the certification chain. Value "0" means that this CA may only issue end-entity certificates, but no CA certificates. If the attribute is not set, there is no limit to the allowed length of the certification path.

The settings of the basicConstraints extension differ for each SCION control-plane PKI certificate type. The next table shows the specifications per certificate type.

Certificate Type	**Root**	**CA**	**AS**	**Voting (regular and sensitive)**
Attribute:				
basicConstraints extension itself	**MUST** be present	**MUST** be present	**SHOULD NOT** be present	**SHOULD NOT** be present

Certificate Type	Root	CA	AS	Voting (regular and sensitive)
cA	**MUST** be set to TRUE	**MUST** be set to TRUE	If the extension is present, this attribute **MUST** be set to FALSE	If the extension is present, this attribute **MUST** be set to FALSE
pathLenConstraint	**SHOULD** be set to "1", **MUST** be marked as "critical"	**SHOULD** be set to "0" (1), **MUST** be marked as "critical"	If the extension is present, this attribute **MUST** be absent.	If the extension is present, this attribute **MUST** be absent.

Table 6: basicConstraints extension - Specifications per certificate type

(1) Control-plane CAs can only issue end-entity certificates.

4. Trust Root Configuration Specification

This section provides an in-depth specification of the trust root configuration (TRC) file (see Section 4.1). The TRC contains policy information about an ISD and acts as a distribution mechanism for the trust anchors of that ISD. It enables securing the control-plane interactions, and is thus an integral part of the SCION infrastructure.

The initial TRC of an ISD is signed during a signing ceremony and then distributed throughout the ISD. This signing ceremony follows specific rules; Section 4.2 describes these rules.

4.1. TRC Specification

The trust root configuration (TRC) is a signed collection of X.509 v3 certificates. Additionally, the TRC contains ISD-specific policies encoded in a Cryptographic Message Syntax (CMS) [RFC5652] envelope.

The TRC's certificates collection consists of a set of control-plane root certificates, which build the root of the certification chain for the AS certificates in an ISD. The other certificates in the TRC are solely used for signing the next TRC, a process called "voting". The verification of a new TRC thus depends on the policies and voting certificates defined in the previous TRC.

Note: See Section 3 for the general specifications of SCION's control-plane PKI certificates, as well as Section 3.1.2 and Section 3.1.5, for the specifications of the control-plane root certificates and voting certificates, respectively.

This section provides a detailed specification of the TRC. It presents the TRC format definitions and describes the TRC payload fields. The section uses the ITU-T X.680 syntax.

4.1.1. TRC Types and States

The following types of TRCs exist:

- Initial: The very first TRC of an ISD is the initial TRC of that ISD. It is a special case of the base TRC, where the number of the ISD is specified.

- Base: A base TRC is either the initial TRC, or the first TRC after a trust reset (see Section 1.3.1). Trust for a base TRC cannot be inferred by verifying a TRC update; base TRCs are trusted axiomatically, similarly to how root CA certificates are trusted by clients in the Web PKI.

- Update: All non-base TRCs are updated TRCs. They are the product of either a regular or a sensitive update.

A TRC can have the following states:

- Valid: The validity period of a TRC is defined in the TRC itself, in the validity field (see Section 4.1.2.2.3). A TRC is considered valid if the current time falls within its validity period.

- Active: An active TRC is a valid TRC that can be used for verifying certificate signatures. This is either the latest TRC or the predecessor TRC, if it is still in its grace period (as defined in the gracePeriod field of the new TRC, see Section 4.1.2.2.4). No more than two TRCs can be active at the same time for any ISD.

Figure 3 shows the content of both a base/initial TRC and the first regularly-updated TRC based on the base TRC. All elements of the shown TRCs are specified in detail in the following subsections.

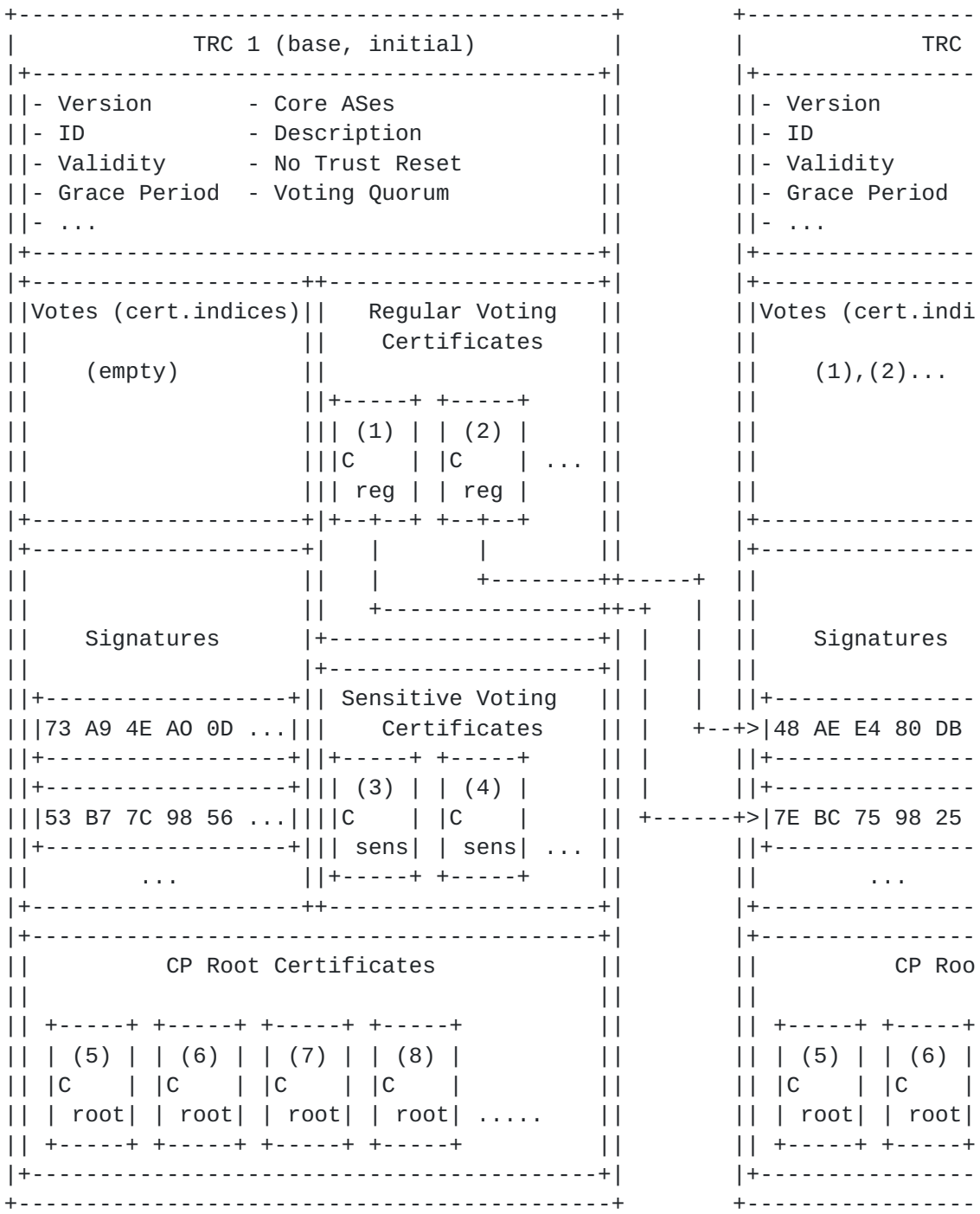


Figure 3: The TRC on the left-hand side is the initial base TRC. The TRC on the right is the product of the first regular update of the base TRC.

4.1.2. TRC Format

The trust root configuration (TRC) of an ISD defines the roots of trust of the ISD, and builds the base of the ISD's control-plane PKI. It holds the root and voting certificates of the ISD and defines the ISD's trust policy.

4.1.2.1. TRC Schema

The following code block shows the format of a TRC specification file (the payload schema):

```
TRCPayload  ::=  SEQUENCE {
    version   TRCFormatVersion,
    iD        TRCID,
    validity  Validity,

    gracePeriod   INTEGER,
    noTrustReset  BOOLEAN DEFAULT FALSE,
    votes         SEQUENCE OF INTEGER (SIZE (1..255)),

    votingQuorum  INTEGER (1..255),

    coreASes           SEQUENCE OF ASN,
    authoritativeASes  SEQUENCE OF ASN,
    description        UTF8String (SIZE (0..1024)),

    certificates       SEQUENCE OF Certificate }

TRCFormatVersion  ::=  INTEGER { v1(0) }

TRCID  ::=  SEQUENCE {
    iSD           ISD,
    serialNumber  INTEGER (1..MAX),
    baseNumber    INTEGER (1..MAX) }

ISD  ::=  INTEGER (1..65535)

Validity  ::=  SEQUENCE {
    notBefore  Time,
    notAfter   Time }

ASN  ::=  INTEGER (1..281474976710655)
```

The TRCPayload sequence contains the identifying information of a TRC as well as policy information for TRC updates. Furthermore, it defines the list of certificates that build the trust anchor of the ISD.

For signature calculation, the data that is to be signed is encoded using ASN.1 distinguished encoding rules (DER) X.690. For more details, see Section 4.1.3.

4.1.2.2. TRC Fields

This section describes the syntax and semantics of all TRC payload fields.

4.1.2.2.1. version Field

The version field describes the version of the TRC format specification.

Currently, the version **MUST** always be "v1".

4.1.2.2.2. iD Field

The iD field specifies the unique identifier of the TRC.

The identifier is a unique sequence of

- *ISD number (iSD attribute),
- *base number (baseNumber attribute), and
- *TRC serial number (serialNumber attribute).

All numbers **MUST** be positive integers.

- *The **ISD number** **MUST** be an integer in the inclusive range from 64 to 4094 (i.e., the numbering range for public ISDs, see Section 1.5.1).
- *The **base number** indicates the starting point of the current TRC update chain. This starting point is either the ISD's initial TRC or the currently valid base TRC, if the valid base TRC differs from the initial TRC. The latter **MUST** be the case after a trust reset.
- *The **serial number** represents the current update cycle, counting from the initial TRC of a specific ISD.

A TRC where the base number is equal to the serial number is a base TRC. The initial TRC is a special case of a base TRC. An ISD's initial TRC **MUST** have a serial number of 1 and a base number of 1. With every TRC update, the serial number **MUST** be incremented by one. This facilitates uniquely identifying the predecessor and successor TRC in a TRC update chain.

If a trust reset is necessary, a new base TRC is announced, in order to start a new and clean TRC update chain. The base number of this new TRC update chain **SHOULD** be the number following the serial number of the latest TRC that was produced by a non-compromised TRC update for this ISD.

Example
The following simple example illustrates how to specify the ID of the TRCs in an TRC update chain for *ISD 74*. The IDs are given in a human-readable notation, where Bxx is the base number, and Sxx the serial number.

Update	TRC ID	Remarks
Initial	ISD74-B01-S01	
Regular	ISD74-B01-S02	Only the serial number is incremented.
Regular	ISD74-B01-S03	Only the serial number is incremented.
Sensitive	ISD74-B01-S04	Only the serial number is incremented.
Trust reset	ISD74-**B05**-S05	A trust reset includes the creation of a new base TRC. The new base number follows the serial number "04" of the latest TRC resulting from a non-compromised TRC update for this ISD.
Regular	ISD74-B05-S06	Only the serial number is incremented.
Regular	ISD74-B05-S07	Only the serial number is incremented.
And so on		

Table 7: ID of TRCs in TRC update chain

4.1.2.2.3. validity Field

The validity field defines the validity period of the TRC. This is the period of time during which the TRC is in the "valid" state. The notBefore and notAfter attributes of the validity field specify the lower and upper bound of the time interval during which a TRC can be active.

Note: An active TRC is a valid TRC that can be used for verifying certificate signatures. The time period during which a TRC is active can be shorter than the time period during which the TRC is valid. For more information, see Section 4.1.1.

The validity field consists of a sequence of two dates, as defined in section 7.2. of X.509.

In addition to this standard definition, the following constraint applies to the validity field of the TRC used in SCION:

- All TRCs **MUST** have a well-defined expiration date. SCION implementations **MUST NOT** create TRCs that use the "99991231235959Z" generalized time value, and verifiers **MUST** error out when encountering such a TRC.

4.1.2.2.4. gracePeriod Field

The gracePeriod field of a TRC specifies the period of time during which the predecessor TRC can still be considered active (the "grace period"). The grace period starts at the beginning of the validity period of the new TRC.

The validity period of the predecessor TRC ends when

- the grace period has passed,
- the predecessor's expiration time is reached, or
- the successor TRC of the new TRC has been announced.

Note: The event that happens first marks the end of the predecessor's validity period.

The gracePeriod field defines the grace period as a number of seconds (positive integer).

The value of the gracePeriod field in a base TRC **MUST** be zero. The value of the gracePeriod field in a non-base TRC **SHOULD** be non-zero. It should be long enough to provide sufficient overlap between the TRCs in order to facilitate interruption-free operations in the ISD. If the grace period is too short, some control-plane AS certificates might expire before the corresponding AS can fetch an updated version from its CA.

4.1.2.2.5. noTrustReset Boolean

The noTrustReset Boolean specifies whether a trust reset is forbidden by the ISD. Within a TRC update chain, this value CANNOT be changed by a regular or sensitive update. However, it is possible to change the noTrustReset value in the event of a trust reset, where a new base TRC is created.

The noTrustReset field is optional and defaults to FALSE.

Important: Note that once the noTrustReset Boolean is set to TRUE and a trust reset is disallowed, this cannot be reversed. Therefore, ISDs **SHOULD** always set this value to FALSE, unless they have sufficiently assessed the risks and implications of making a trust reset impossible.

Note: A trust reset represents a special use case where a new base TRC is created. It therefore differs from a TRC update (regular or sensitive), as the signatures in the new base TRC cannot be verified with the certificates contained in the predecessor TRC. Instead, a trust reset base TRC must be axiomatically trusted, similarly to how the initial TRC is trusted.

4.1.2.2.6. votes Field

The votes field contains a sequence of indices that refer to the voting certificates in the predecessor TRC. If index i is part of the votes field, then the voting certificate at position i in the certificates sequence of the predecessor TRC casted a vote on the successor TRC. For more information on the certificates sequence, see Section 4.1.2.2.11.

Note: In a base TRC, the votes sequence is empty.

Every entry in the votes sequence **MUST** be unique.
Further restrictions on votes are discussed in Section 4.1.5.

Note: The votes sequence of indices is mandatory in order to prevent stripping voting signatures from the TRC. Absence of the votes sequence makes it possible to transform a TRC with more voting signatures than the voting quorum into multiple verifiable TRCs with the same payload, but different voting signature sets. This would violate the requirement of uniqueness of a TRC.

4.1.2.2.7. votingQuorum Field

The votingQuorum field defines the number of necessary votes on a successor TRC to make it verifiable.

A voting quorum greater than one will prevent a single entity from creating a malicious TRC update.

4.1.2.2.8. coreASes Field

The coreASes field contains the AS numbers of the core ASes in this ISD.

Each core AS number **MUST** be unique in the sequence of core AS numbers. That is, each AS number must appear only once in the coreASes field.

4.1.2.2.8.1. Revoking or Assigning Core Status

- To revoke the core status of a given AS, remove the respective AS number from the sequence of AS numbers in the coreASes field.

- To assign the core status to a given AS, add the respective AS number to the sequence of AS numbers in the coreASes field.

Important: Revoking or assigning the core status of/to an AS always requires a (sensitive) TRC update.

4.1.2.2.9. authoritativeASes Field

The authoritativeASes field contains the AS numbers of the authoritative ASes in this ISD.

Authoritative ASes are those ASes in an ISD that always have the latest TRCs of the ISD. As a consequence, authoritative ASes also start the announcement of a TRC update.

- Every authoritative AS **MUST** be a core AS and be listed in the coreASes field.

- Each authoritative AS number **MUST** be unique in the sequence of authoritative AS numbers. That is, each AS number must appear only once in the authoritativeASes field.

4.1.2.2.9.1. Revoking or Assigning Authoritative Status

- To revoke the authoritative status of a given AS, remove the respective AS number from the sequence of AS numbers in the authoritativeASes field.

- To assign the authoritative status to a given AS, add the respective AS number to the sequence of AS numbers in the authoritativeASes field.

Important: Revoking or assigning the authoritative status of/to an AS always requires a (sensitive) TRC update.

4.1.2.2.10. description Field

The description field contains a UTF-8 encoded string that describes the ISD.

- The description field **SHOULD NOT** be empty.

- The description of the ISD **MUST** be in English. Additionally, the description field **MAY** contain information in other languages.

4.1.2.2.11. certificates Field

The voting ASes and the certification authorities (CAs) of an ISD are not specified explicitly in the ISD's TRC. Instead, this information is defined by the list of voting and root certificates in the certificates field of the TRC payload.

The certificates field is a sequence of self-signed X.509 certificates. Each certificate in the certificate sequence must be one of the following types:

- a sensitive voting certificate,

- a regular voting certificate, or

- a CP root certificate.

A certificate that is no control-plane root or voting certificate **MUST NOT** be included in the sequence of certificates in the certificates field.

Note: A certificate's type (voting or root) is specified in the extKeyUsage extension of the certificate, by means of the SCION-specific attributes id-kp-regular, id-kp-sensitive, and id-kp-root, respectively. For more information, see Section 3.2.2.4.

The following constraints **MUST** hold for each certificate in the certificates field of the TRC payload:

- Each certificate **MUST** be unique in the sequence of certificates. That is, each certificate must appear only once in the certificates field.

- The issuer / serialNumber pair for each certificate **MUST** be unique.

- If an ISD-AS number is present in the distinguished name of the certificate, this ISD number **MUST** be equal to the ISD number of the TRC (which is defined in the iD field (see Section 4.1.2.2.2).

- Every certificate **MUST** have a validity period that fully contains the validity period of this TRC. That is, the notBefore date of this TRC's validity period **MUST** be equal to or later than the certificate's notBefore date, and the notAfter date of this TRC's validity period **MUST** be before or equal to the certificate's notAfter date.

- Per certificate type, every certificate distinguished name **MUST** be unique.

The following must hold for the entire sequence of certificates in the certificates field:

- votingQuorum <= count (sensitive voting certificates)
  That is, the quorum defined in the TRC's votingQuorum field (Section 4.1.2.2.7) must be smaller than or equal to the number of *sensitive* voting certificates specified in the TRC's certificates field.

- votingQuorum <= count (regular voting certificates)
  That is, the quorum defined in the TRC's votingQuorum field (Section 4.1.2.2.7) must be smaller than or equal to the number of *regular* voting certificates specified in the TRC's certificates field.

4.1.3. TRC Signature Syntax

A TRC contains policy information about an ISD and acts as a distribution mechanism for the trust anchors of that ISD. Each TRC (payload) is digitally signed. The syntax used to sign and encapsulate the TRC payload is the Cryptographic Message Syntax (CMS), as defined in [RFC5652]. The signed TRC payload is of the CMS signed-data content type, as defined in Section 5 of [RFC5652], and encapsulated in a CMS ContentInfo element, as defined in Section 3 of [RFC5652]. For detailed information on the general syntax definitions of the Cryptographic Message Syntax, see sections 3 and 5 of [RFC5652].

4.1.3.1. SCION-specific rules

SCION implementations have to fulfil the following additional rules, on top of the general syntax rules from [RFC5652]:

*EncapsulatedContentInfo sequence:

-The eContentType field must be set to "id-data".

-The content of the eContent field must be the DER-encoded TRC payload. This has the benefit that the format is backwards compatible with PKCS #7, as described in Section 5.2.1 of [RFC5652].

*SignedData sequence:

-The certificates field **MUST** be left empty. The certificate pool used to verify a TRC update is already specified in the certificates field of the predecessor TRC's payload (see also Section 4.1.2.2.11).

-The version field **MUST** be set to "1". This is because SCION uses the "id-data" content type to encapsulate content info, and does not specify any certificate in the SignedData sequence (see also Section 5.1 of [RFC5652]).

*SignerIdentifier choice:

-The type of signer identifier chosen here **MUST** be IssuerAndSerialNumber.

*SignerInfo sequence:

-The version field **MUST** be set to "1". This is because SCION uses the IssuerAndSerialNumber type of signer identifier (see also Section 5.3 of [RFC5652]).

-The algorithm specified in the signatureAlgorithm field **MUST** be one of the algorithms supported by SCION (for details, see signature Field - Additional Information (Section 3.2.1.1)).

-The digestAlgorithm is determined by the algorithm specified in the signatureAlgorithm field.

4.1.3.2. TRC Equality

The signer infos in the signed TRC are part of an unordered set, per [RFC5652]. This implies that the signer infos can be reordered without affecting verification. Certain operations, however, require TRCs to be equal, according to the following equality definition:

Two TRCs are equal, if and only if their payloads are byte equal.

Two TRCs with byte equal payloads can be considered as equal, because the TRC payload exactly defines which signatures must be attached in the signed TRC:

* The required signatures from voting certificates are explicitly mentioned in the votes field of the payload: If index "i" is part of the votes field, then the voting certificate at position i in the certificates sequence of the predecessor TRC casted a vote on the successor TRC. See also Section 4.1.2.2.6.

* The required signatures for new certificates are implied by the currently valid TRC payload, and, in case of a TRC update, the predecessor payload.

4.1.4. Control-Plane Certification Path

The certification path of a control-plane AS certificate starts in a control-plane root certificate. The control-plane root certificates for a given ISD are distributed via the TRC.

To be able to validate the certification path, the relying party must build a trust anchor pool, which consists of a set of control-plane root certificates from the available TRCs. Based on this pool, the relying party can select candidate certification paths and verify them.

4.1.4.1. Trust Anchor Pool - TRC Selection

The selection of the right set of TRCs to build the trust anchor pool depends on the time of verification. The trust anchor pool is usually used to verify control-plane messages. In this case, the time of verification is the current time. However, if the trust anchor pool will be used for auditing, the time of verification is the point in time for which you want to check whether a given signature was verifiable.

The selection algorithm for building the trust anchor pool is described in pseudo-python code below.

```
def select_trust_anchors(trcs: Dict[(int,int), TRC], verification_ti
    """
    Args:
        trcs: The dictionary mapping (serial number, base number) to
        verification_time: The time of verification.

    Returns:
        The set of CP Root certificates that act as trust anchors.
    """
    # Find highest base number that has a TRC with a validity period
    # starting before verification time.
    base_nr = 1
    for trc in trcs.values():
        if trc.id.base_nr > base_nr and trc.validity.not_before <= v
            base_nr = trc.id.base_nr

    # Find TRC with highest serial number with the given base number
    # validity period starting before verification time.
    serial_nr = 1
    for trc in trcs[isd].values():
        if trc.id.base_nr != base_nr:
            continue
        if trc.id.serial_nr > serial_nr and trc.validity.not_before
            serial_nr = trc.id.serial_nr

    candidate = trcs[(serial_nr, base_nr)]

    # If the verification time is not inside the validity period,
    # there is no valid set of trust anchors.
    if not candidate.validity.contains(verification_time):
        return set()

    # If the grace period has passed, only the certificates in that
    # may be used as trust anchors.
    if candidate.validity.not_before + candidate.grace_period < veri
        return collect_trust_anchors(candidate)

    predecessor = trcs.get((serial_nr-1, base_nr))
    if not predecessor or predecessor.validity.not_after < verificat
        return collect_trust_anchors(candidate)

    return collect_trust_anchors(candidate) | collect_trust_anchors(


def collect_trust_anchors(trc: TRC) -> Set[RootCert]:
    """
    Args:
        trc: A TRC from which the CP Root Certificates shall be extr

    Returns:
        The set of CP Root certificates that act as trust anchors.
    """
    roots = set()
```

```
for cert in trc.certificates:
    if not cert.basic_constraints.ca:
        continue
    roots.add(cert)
return roots
```

4.1.5. TRC Updates

All non-base TRCs of an ISD are updates of the ISD's base TRC(s). The TRC update chain consists of regular and sensitive TRC updates. Based on the type of update, a different set of voters is necessary to create a verifiable TRC update. The type of update also determines the (payload) information that changes in the updated TRC. This section describes the rules that apply to updating a TRC in regard to the payload information contained in the TRC. Some rules are valid for both update types, some only apply to a regular or a sensitive TRC update, respectively.

4.1.5.1. Changed or New Certificates

In the context of a TRC update,

- A certificate is *changing*, if the certificate is part of the certificates sequence in the predecessor TRC, but no longer part of the certificates sequence in the updated TRC. Instead, the certificates sequence of the updated TRC holds another certificate of the *same type* and with the *same distinguished name*.

- A certificate is *new*, if there is **no** certificate of the same type and distinguished name at all in the certificates sequence of the predecessor TRC.

Note: Every new sensitive or regular voting certificate in a TRC attaches a signature to the TRC. This is done to ensure that the freshly included voting entity agrees with the contents of the TRC it is now part of.

4.1.5.2. Update Rules - Overview

The following table gives an overview of the types of TRC update as well as the rules that must apply in regard to the updated TRC's payload information.
The sections that follow provide more detailed descriptions of each rule.

Type of Update	**Payload Updated TRC - Unchanged Elements**	**Payload Updated TRC - Required Changes**	**Payload Updated TRC: Other Rules to Hold**
Both Regular AND Sensitive Updates	- iD field: iSD and baseNumber - noTrustReset field	iD field: serialNumber **MUST** be incremented by 1	votes field: Number of votes (indices) => number set in the votingQuorum field of the predecessor TRC

Type of Update	Payload Updated TRC - Unchanged Elements	Payload Updated TRC - Required Changes	Payload Updated TRC: Other Rules to Hold
Regular TRC Update	- Quorum in the votingQuorum field - Core ASes in the coreASes field - ASes in the authoritativeASes field - Nr. and distinguished names of root & voting certificates in the certificates field - Set of sensitive voting certificates in the certificates field		votes field: - All votes must only refer to *regular* voting certificates in the predecessor TRC - Must include votes of each changed regular voting certificate from the predecessor TRC signatures field: - Must include signatures of each changed root certificate from the predecessor TRC
Sensitive TRC Update	If the update does not qualify as a regular update, it is a sensitive update		votes field: - All votes must only refer to *sensitive* voting certificates in the predecessor TRC

Table 8: Overview of the update types and corresponding rules

4.1.5.3. General Update Rules

The following rules **MUST** hold for each updated TRC, independent of the update type:

*The iSD and baseNumber in the iD field **MUST NOT** change (see also Section 4.1.2.2.2).

*The serialNumber in the iD field **MUST** be incremented by one.

*The noTrustReset field **MUST NOT** change (see also Section 4.1.2.2.5).

*The votes sequence of the updated TRC **MUST** only contain indices that refer to sensitive or regular voting certificates in the

predecessor TRC. This guarantees that the updated TRC only contains valid votes authenticated by sensitive or regular voting certificates in the predecessor TRC. For more information, see Section 4.1.2.2.6 and Section 4.1.2.2.11.

* The number of votes in the updated TRC **MUST** be greater than or equal to the number set in the votingQuorum field of the predecessor TRC (see Section 4.1.2.2.7). The number of votes corresponds to the number of indices in the votes field of the updated TRC.

4.1.5.4. Regular TRC Update

A regular TRC update is a periodic re-issuance of the TRC where the entities and policies listed in the TRC remain unchanged.

A TRC update qualifies as a regular update, if the following rules apply in regard to the TRC's payload information.

* The settings of the following fields in the updated TRC **MUST** remain the same compared to the predecessor TRC:
  - The voting quorum set in the votingQuorum field.
  - The core ASes specified in the coreASes field.
  - The authoritative ASes specified in the authoritativeASes field.
  - The number of sensitive and regular voting certificates as well as CP root certificates included in the certificates field, and their distinguished names.
  - The set of sensitive voting certificates specified in the certificates field.
* For every regular voting certificate that changes, the regular voting certificate in the predecessor TRC is part of the voters on the updated TRC. That is, for each changed regular voting certificate, an index in the votes field of the updated TRC **MUST** refer to the changed regular voting certificate in the predecessor TRC.
* For every CP root certificate that changes, the updated TRC **MUST** include a signature created with the private key belonging to the changed CP root certificate (which is part of the predecessor TRC).
* In order for a regular TRC update to be verifiable, all votes **MUST** be cast by *regular* voting certificates. That is, each index in the votes field of the regularly updated TRC **MUST** refer to a *regular* voting certificate in the certificates field of the predecessor TRC.

4.1.5.5. Sensitive TRC Update

If a TRC update does not qualify as a regular update, it is considered a sensitive update.

* In order for a sensitive update to be verifiable, all votes **MUST** be cast by *sensitive* voting certificates. That is, each index in the votes field of the sensitively updated TRC **MUST** refer to a *sensitive* voting certificate in the certificates field of the predecessor TRC.

4.1.5.6. TRC Update Verification

To verify a TRC update, the relying party must perform the following checks:

* Check that the specified update rules as described above are respected.

* Check whether the update is regular or sensitive.

  - In case of a regular update,

    - check that the signatures for the changing certificates are present and verifiable, and

    - check that all votes are cast by a regular voting certificate.

  - In case of a sensitive update, check that all votes are cast by a sensitive voting certificate.

* In both cases, check that all signatures are verifiable, and no superfluous signatures are attached.

If one or more of the above checks gives a negative result, the updated TRC should be rejected.

4.2. TRC Signing Ceremony

The very first base TRC of an ISD, called the initial TRC, is a special case of the base TRC where the number of the ISD is chosen. The initial TRC must be signed during a signing ceremony--all voting representatives of the initial TRC need to take part in this signing ceremony to sign the agreed-upon TRC. As part of the ceremony, the public keys of all voters are exchanged. The TRC is then distributed throughout the ISD. All entities within an ISD can initially obtain an authentic TRC, by means of a secure off- or online mechanism.

Appendix "Appendix A. Signing Ceremony Base TRC" describes a possible procedure for the signing ceremony of an ISD's initial TRC. It is in principle up to the initial members of an ISD how to shape the signing ceremony. However, it is recommended having a process in line with the ceremony described in the Appendix.

4.2.1. Non-Base TRC Updates

A non-base TRC is the result of a TRC update, either regular or sensitive. Only a predefined quorum of voters needs to partake in a non-base TRC signing ceremony. This is defined in the votingQuorum field of the predecessor TRC (see Section 4.1.2.2.7). Depending on the kind of update, these voters represent regular or sensitive voting certificates, respectively. Furthermore, if one or more *new* certificates are added to the updated TRC, the corresponding voting representatives must also join the signing ceremony. For the distinction between changed and new certificates in a TRC update, see Section 4.1.5.1.

During the signing ceremony of an updated TRC, it may be necessary to cast votes with both old and new keys: Voters representing regular or sensitive voting certificates already present in the predecessor TRC must cast their votes on the payload file of the updated TRC; the purpose of signing a TRC with keys contained in the previous TRC is to certify the update. Furthermore, if previously non-included voting certificates are added to the TRC, the corresponding voting representatives must show that they have access to the private keys listed in these fresh certificates. This is called "showing proof-of-possession", and done by signing the TRC with the respective private key.

The ISD members decide themselves about the updating procedure. Some ISDs will make a distinction between regular and sensitive updates. These ISDs divide the regular and sensitive signing keys in different security classes and act accordingly. For example, they keep the regular key in an online vault while the sensitive key would be stored offline in cold storage. This way, the regular TRC update would lend itself to being automated (since the keys are accessible online) whereas the sensitive one would require manual actions to access the offline key. Other ISDs, however, keep both regular and sensitive keys online and perform both updates automatically.

5. Deploying the CP PKI - Specifications

This section provides several specifications regarding the deployment of the control-plane PKI.

5.1. Deploying a TRC

5.1.1. Base TRC

Base TRCs are trust anchors and thus axiomatically trusted. All ASes within an ISD must be pre-loaded with the currently valid base-version TRC of their own ISD. For all specifications regarding the creation and distribution of initial/base TRCs, see Section 4.2.

5.1.2. TRC Update

All non-base TRCs of an ISD are updates of the ISD's base TRC(s). The TRC update chain consists of regular and sensitive TRC updates. The specifications and rules that apply to updating a TRC are described in Section 4.1.5.

5.1.2.1. TRC Update Discovery

Relying parties **MUST** have at least one valid TRC available. Relying parties **MUST** discover TRC updates within the grace period defined in the updated TRC. They **SHOULD** discover TRC updates in a matter of minutes to hours. Additionally, the following requirement must be satisfied:

Requirement
Any entity sending information that is secured by the CP-PKI **MUST** be able to provide all the necessary trust material to verify said information.

SCION provides the following mechanisms for discovering TRC updates and fulfilling the above requirement:

* *Beaconing Process*
  The TRC version is announced in the beaconing process. Each AS must announce what it considers to be the latest TRC. Furthermore, each AS must include the hash value of the TRC contents to facilitate the discovery of discrepancies. Therefore, relying parties that are part of the beaconing process discover TRC updates passively. That is, the beacon service in a core AS notices TRC updates for remote ISDs that are on the beaconing path. The beacon service in a non-core AS only notices TRC updates for the local ISD through the beaconing process. The creation of a new TRC should trigger the generation of new control-plane messages, as the propagation of control-plane messages will help other ASes rapidly discover the new TRC.

* *Path Lookup*
  In every path segment, all ASes must reference the latest TRC of their ISD. Therefore, when resolving paths, every relying party will notice TRC updates, even remote ones.
  Note: The above mechanism only works when there is an active communication between the relying party and the ISD in question.

5.2. Signing and Verifying Control-Plane Messages

SCION requires that control-plane messages are signed. The main purpose of the SCION control-plane PKI is providing a mechanism to distribute and authenticate public keys that are used to verify control-plane messages and information. For example, each hop information in a path segment is signed by the respective AS. Consequently, all relying parties must be able to verify signatures with the help of the CP-PKI.

The following sections specify the requirements that apply to the signing and verification of control-plane messages.

5.2.1. Signing a Control-Plane Message

An AS signs control-plane messages with the private key that corresponds to the (valid) AS' certificate.

The AS **MUST** attach the following information as signature metadata. It is the minimum information a relying party requires to identify which certificate to use to verify the signed message.

- ISD-AS number: The ISD-AS number of the signing entity. For specification details, see Section 3.2.1.2.1.

- Subject key identifier: The identifier of the public key that must be used to verify the message. For specification details, see Section 3.2.2.2.

Additionally, the signer **SHOULD** include the following information:

- Serial and base number of the latest TRC: Including this information allows relying parties to discover TRC updates and trust resets. For specification details, see Section 4.1.2.2.2.

- Timestamp: For many messages, the time at which it was signed is useful information to ensure freshness.

5.2.2. Verifying a Control-Plane Message

When the relying party receives a control-plane message they want to verify, the relying party first needs to identify the certificate needed to validate the corresponding signature on the message.

AS certificates are bundled together with the corresponding signing CA certificate into certificate chains. For efficiency, SCION distributes these certificate chains separately from the signed messages. A certificate chain is verified against the CP root certificate. However, the root certificate is **not** bundled in the chain, but with the TRC. This makes it possible to extend the validity period of the root certificate, and to update the corresponding TRC, without having to modify the certificate chain.

To verify a control-plane message, the relying party must perform the following steps:

1. Build a collection of root certificates from the latest TRC of the relevant ISD (that is, the ISD referenced in the signature metadata of the message). If the grace period (see Section 4.1.2.2.4) introduced by the latest TRC is still on-going, the root certificates in the second-to-latest TRC must also be included. For a description on how to build the correct collection of certificates, see Section 4.1.4.1.

2. If the signature metadata of the message contains the serial and base number of the latest TRC, the relying party must check that they have this latest TRC. If not, the relying party must request the latest TRC.

3. After constructing the pool of root certificates, the relying party must select the certificate chain used to verify the message. The AS certificate included in this certificate chain **MUST** have the following properties:

   * The ISD-AS number in the subject of the AS certificate **MUST** match the ISD-AS number in the signature metadata. See also Section 3.2.1.2.1.

   * The subject key identifier of the AS certificate **MUST** match the subject key identifier in the signature metadata. See also Section 3.2.2.2.

   * The AS certificate **MUST** be valid at verification time. Normally, this will be the current time. In special cases, e.g., auditing, the time can be set to the past to check if the message was verifiable at the given time.

4. After selecting a certificate chain to verify the control-plane messages, the relying party must verify the certificate chain, by:

   * Executing the regular X.509 verification procedure. For details, see X.509.

   * Checking that

     - all subjects of the certificates in the chain carry the same ISD number (see also Section 3.2.1.2.1,

     - each certificate is of the correct type (see also Section 3.1), and

     - the CA certificate validity period covers the AS certificate validity period.

5. If the verification of the certificate chain was successful, the relying party can now verify the control-plane messages, with the root certificates from the certificate chain.

If any cryptographic material is missing in the process, the relying party queries the originator of the message for the missing material. If it cannot be resolved, the verification process fails.

Important: An implication of the above procedure is that path segments should be verifiable at time of use. It is not enough to rely on path segments being verified on insert, since TRC updates that change the root key can invalidate a certificate chain.

5.3. Creating a New Control-Plane AS Certificate

The steps required to create a new AS certificate are the following:

1. The AS creates a new key pair and a certificate signing request (CSR) using that key pair.

2. The AS sends the certificate signing request to the relevant CA within the ISD.

3. The CA uses its CA key and the CSR to create the new AS certificate.

4. The CA sends the AS certificate back to the AS.

6. Security Considerations

The entire document is about security considerations. More details will follow in future versions of this draft.

7. IANA Considerations

Future iterations of this draft will comprise more detailed IANA considerations.

8. References

8.1. 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/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>.

[RFC5398] Huston, G., "Autonomous System (AS) Number Reservation for Documentation Use", RFC 5398, DOI 10.17487/RFC5398, December 2008, <https://www.rfc-editor.org/rfc/rfc5398>.

[RFC5480] Turner, S., Brown, D., Yiu, K., Housley, R., and T. Polk, "Elliptic Curve Cryptography Subject Public Key

Information", RFC 5480, DOI 10.17487/RFC5480, March 2009, <https://www.rfc-editor.org/rfc/rfc5480>.

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

[RFC5758] Dang, Q., Santesson, S., Moriarty, K., Brown, D., and T. Polk, "Internet X.509 Public Key Infrastructure: Additional Algorithms and Identifiers for DSA and ECDSA", RFC 5758, DOI 10.17487/RFC5758, January 2010, <https://www.rfc-editor.org/rfc/rfc5758>.

[RFC6996] Mitchell, J., "Autonomous System (AS) Reservation for Private Use", BCP 6, RFC 6996, DOI 10.17487/RFC6996, July 2013, <https://www.rfc-editor.org/rfc/rfc6996>.

[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>.

8.2. Informative References

[BARRERA17] Barrera, D., Chuat, L., Perrig, A., Reischuk, R., and P. Szalachowski, "The SCION internet architecture", Communications of the ACM vol. 60, no. 6, pp. 56-65, DOI 10.1145/3085591, May 2017, <https://doi.org/10.1145/3085591>.

[CHUAT22] Chuat, L., Legner, M., Basin, D., Hausheer, D., Hitz, S., Mueller, P., and A. Perrig, "The Complete Guide to SCION", ISBN 978-3-031-05287-3, 2022, <https://doi.org/10.1007/978-3-031-05288-0>.

[I-D.scion-components] Rustignoli, N. and C. de Kater, "SCION Components Analysis", 2022, <https://datatracker.ietf.org/doc/draft-rustignoli-panrg-scion-components/>.

[I-D.scion-overview] de Kater, C., Rustignoli, N., and A. Perrig, "SCION Overview", 2022, <https://datatracker.ietf.org/doc/draft-dekater-panrg-scion-overview/>.

Acknowledgments

Many thanks go to Samuel Hitz (Anapaya), Fritz Steinmann (SIX Group AG), Juan A. Garcia Prado (ETH Zurich) and Russ Housley (IETF) for reviewing this document. We are also very grateful to Adrian Perrig (ETH Zurich), for providing guidance and feedback about each aspect of SCION. Finally, we are indebted to the SCION development teams of Anapaya and ETH Zurich, for their practical knowledge and for the documentation about the CP PKI, as well as to the authors of [CHUAT22] - the book is an important source of input and inspiration for this draft.

Appendix A. Signing Ceremony Base TRC

The following sections describe a possible signing ceremony for the first (initial) base TRC of an ISD. Although each ISD is free to decide how to shape this signing ceremony, it is recommended establishing a procedure similar to the one below.

Ceremony Participants

A signing ceremony includes participants from member organizations of the respective Isolation Domain. The participants of the signing ceremony fulfill different roles:

- The **ceremony administrator** is in charge of moderating the signing process. He/she guides all participants through the steps they need to take. The ceremony administrator may also act as an intermediary between participants when they share information with each other.

- A **voting AS representative** is capable of creating voting signatures on the TRC. This means the voting representative is in possession of a device with the private keys of the respective certificates in the TRC.

- A **witness** is any person that participates in the ceremony as a passive entity. The witness has no active role in any of the steps of the ceremony, but can stop the process and inquire for more information if they feel the integrity of the process might have been compromised.

Note: It is assumed that the member organizations of the ISD have decided in advance, before the signing ceremony, on the roles of the ceremony participants. That is, they have reached agreement about the Certificate Authority (CA) ASes (that will also issue the root certificates), the voting ASes, the representatives of the voting ASes, the ceremony administrator and the witnesses.

Note: For the signing ceremony, it is assumed that all parties are trustworthy. Issues encountered during the ceremony are assumed to be caused by honest mistakes, and not by malicious intent. Hash comparison checks are included to counter mistakes, such that every participant is sure that they operate on the same data. Furthermore, the private keys of each participant never leave their machine. The ceremony administrator does not have to be entrusted with private keys.

Ceremony Preparations

Prior to the ceremony, participants decide on the physical location of the ceremony, the devices that will be used during the ceremony and the policy of the ISD. Specifically, the voting entities agree on the following parameters:

- validity of the TRC,

*voting quorum,

*core ASes/authoritative ASes,

*description, and

*list of CP root certificates.

When these values are agreed upon, a number of voters, equal to or larger than the specified voting quorum, needs to execute the signing ceremony. For the base TRC, all voting entities need to be present with both their sensitive and regular voting keys. The ceremony process is structured in multiple rounds of data sharing. The ceremony administrator leads the interaction and gives instructions to each participant.

Location

The location must provide electricity and enough power sockets for each participant. Furthermore, it should provide a monitor (or projector) that allows the ceremony administrator to screen cast.

Devices

Each party brings their own device that is provisioned with the required material, as described below.

*Device to exchange data. This device can either be provided by the ceremony administrator, or, if preferable, by any of the voting representatives.

*Ceremony administrator's device: The ceremony administrator should bring a machine that is capable of creating and verifying a TRC. Furthermore, it needs to be able to compute the SHA-512 digest (hash value) of files.

*Voting representative's device: The voting representative should bring a machine that is capable of signing and verifying TRCs. Thus, the machine needs to have access to all the voting private keys. Furthermore, it needs to be able to compute the SHA-512 digest (hash value) of the files. The exact binaries that are required are described in a separate document.

Important: It is very important that all devices, especially the data exchange device, are not compromised. Therefore, the ceremony should ideally include a procedure to verify that the devices are secure.

Preparation Steps

Each party involved in a TRC signing ceremony must go through a few steps in preparation for the ceremony. This section outlines these steps.

Preparatory Tasks of the Ceremony Administrator

In the preparation phase of the TRC Signing Ceremony, the ceremony administrator has the following tasks:

1. Send out the high-level TRC Signing Ceremony description and the document describing the TRC Signing Ceremony Phases to the participants, all in digital form.

2. Remind all representatives of the voting ASes that they need to agree on a common TRC policy before scheduling the TRC ceremony.

3. Bring all digitally distributed documents as a printout for all parties that take part.

Preparatory Tasks of the Voting AS Representatives

The preparatory task of the representatives of the voting ASes (short: the voters) is to generate the necessary certificates.

Important: Before generating the certificates, all voters need to agree on a preliminary TRC policy, in particular on the **validity period of the TRC**. This is necessary because all the certificates that are generated in advance must **cover the full TRC validity period**. The other policy values could be amended during the ceremony itself.

Each representative of a voting AS must create the following keys and certificates:

*A sensitive voting private key, and a certificate holding the corresponding public key.

*A regular voting private key, and a certificate holding the corresponding public key.

Preparatory Tasks of the Certificate Authority ASes

Each AS that will be a Certificate Authority (a so-called CA AS) must ensure that the following key and certificate is available:

*A control-plane root private key, and a certificate holding the corresponding public key.

This implies that there will be one control-plane root certificate per CA AS.

Note: Representatives of CA ASes must not be present at the signing ceremony themselves, as they do not have to put a signature on the TRC. However, if a CA AS does not attend the signing ceremony in person, it must ensure that the corresponding root certificate is available at the ceremony to be shared.

Ceremony Process

The ceremony process for the initial base TRC is structured in multiple rounds of data sharing. The ceremony administrator leads the interaction and instructs each participant with what to do.

The ceremony process contains the following phases:

- Phase 1: Certificate Exchange (Appendix "Phase 1: Certificate Exchange"). In the first phase of the ceremony, all voting parties share the certificates that must be part of the TRC with the ceremony administrator.

- Phase 2: Generation of the TRC Payload (Appendix "Phase 2: Generation of the TRC Payload"). In the second phase, the ceremony administrator generates the TRC payload based on the bundled certificates and the agreed-upon ISD policy.

- Phase 3: TRC Signing (Appendix "Phase 3: TRC Signing"). In the third phase, each voting representative attaches the required signatures to the TRC.

- Phase 4: TRC Validation (Appendix "Phase 4: TRC Validation"). In the final phase of the ceremony, all voting representatives share the signed TRC with the ceremony administrator, who aggregates it in a single signed TRC document.

A detailed description of each phase follows below.

Phase 1: Certificate Exchange

In Phase 1 of the signing ceremony, all parties share the certificates that must be part of the TRC with the ceremony administrator. For the representatives of the voting ASes, these are the sensitive and the regular voting certificates. For the representatives of the CA ASes, these are the CP root certificates. If a CA AS does not attend the signing ceremony in person, it must ensure that the corresponding root certificate is available at the ceremony to be shared.

The actual sharing happens over the data exchange device, which goes from one voting representative to the next. Each representative copies the requested certificates from their own machine onto the data exchange device, before forwarding the device to the next voter. The last representative returns the device to the ceremony administrator.

Important: Note that only the **certificates** must be shared during this step, **not** the private keys. Copying a private key by mistake invalidates the security of the ceremony.

For each provided certificate, the ceremony administrator checks that its validity period covers the previously agreed-upon TRC

validity, that the signature algorithms are correct, and that the certificate is of the valid type (root, sensitive voting or regular voting certificate). If the results of these checks are as expected, the ceremony administrator computes the SHA256 sum for each certificate.

The ceremony administrator then aggregates and bundles the provided certificates, and calculates the hash value (SHA-512 digest) over the entire bundle. Additionally, the ceremony administrator displays all hash values on the monitor.

The ceremony administrator now shares the bundle with the representatives of the voting and CA ASes. This could happen again via the data exchange device, which goes from one representative to the next. Each representative verifies that the certificates they contributed have the same hash value as the displayed value on the monitor. Furthermore, all representatives must confirm that the hash value of the bundled certificates on their machine is equal to the value on the monitor.

Phase 1 is concluded when every representative has confirmed that the SHA256 sums are correct.

Note: If there is a mismatch in any of the SHA256 sums, Phase 1 needs to be repeated.

Phase 2: Generation of the TRC Payload

In Phase 2 of the ceremony, the ceremony administrator generates the TRC payload based on the bundled certificates and the agreed-upon ISD policy. The result is displayed on the monitor along with a hash value (SHA-512 digest).

To be able to generate the payload, the ceremony administrator must ask the voting representatives for

- The ISD number of the ISD. The number (identifier, ID) of an ISD must be chosen and agreed upon by the participants during the signing ceremony of the ISD's initial TRC. The ceremony administrator needs the ISD number to specify the identifier (ID) of the initial TRC. This iD is part of the TRC payload. For more information, see Section 4.1.2.2.2.

- The description of the TRC. For more information, see Section 4.1.2.2.10.

- The AS numbers of the core ASes of the ISD. For more information, see Section 4.1.2.2.8.

- The AS numbers of the authoritative ASes of the ISD. For more information, see Section 4.1.2.2.9.

- The voting quorum for the next TRC update. For more information, see Section 4.1.2.2.7.

*The validity period of the new TRC. For more information, see Section 4.1.2.2.3.

Note: It is assumed that the voting ASes have agreed on the answers to the above questions in advance, before the signing ceremony.

The ceremony administrator can now specify the TRC payload variables in the payload template file, and show the filled-in template on the monitor. When the voters have verified the data, the ceremony administrator can compute the DER encoding of the TRC data as well as the SHA256 sum of the TRC payload file. The ceremony administrator then distributes the TRC payload (via the data exchange device) to all voting representatives, who verify the payload's hash value. The voters do this by computing the hash value of the TRC payload on their machine and checking whether their value matches the one on the monitor.

Phase 2 successfully concludes once every voting representative confirms that the contents of the TRC payload are correct.

Phase 3: TRC Signing

In Phase 3, each voting representative attaches a signature created with each one of their private voting keys to the TRC (payload file). They do this on their own machine. The purpose of signing a TRC that contains newly introduced public keys with the corresponding private keys is to prove the possession of the private keys.

Phase 3 concludes after all voting representatives have cast their votes.

Phase 4: TRC Validation

In Phase 4, all voting representatives share the signed TRC with the ceremony administrator. This happens again over the data exchange device, which goes from one voter to the next. Each voting representative copies the TRC payload signed with the voter's private keys from their own machine onto the data exchange device. The last voter returns the device to the ceremony administrator, who assembles the final TRC by aggregating the payload data with the votes (signatures) cast by the voting representatives.

The signed TRC is validated by inspecting its contents on the monitor and verifying the signatures based on the exchanged certificates in Phase 1. The ceremony administrator then shares the signed TRC with all participants. Each of them must then inspect it once more, and verify it based on the certificates exchanged in Phase 1. At this point, the ceremony is completed. All participants have the signed TRC, and can use it to distribute the trust anchors for their ISD.

Authors' Addresses

Corine de Kater
SCION Association

Email: cdk@scion.org

Nicola Rustignoli
SCION Association

Email: nic@scion.org