

Secure Inter-Domain Routing (sidr)
Internet Draft
Expires: March 2011
Intended Status: Best Current Practice

Kent, S.
Kong, D.
Seo, K.
Watro, R.
BBN Technologies
October 20, 2010

Certificate Policy (CP)
for the Resource PKI (RPKI)
draft-ietf-sidr-cp-15.txt

Status of this Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

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."

The list of current Internet-Drafts can be accessed at
<http://www.ietf.org/ietf/lid-abstracts.txt>

The list of Internet-Draft Shadow Directories can be accessed at
<http://www.ietf.org/shadow.html>

This Internet-Draft will expire on April 30, 2011.

Abstract

This document describes the certificate policy for a Public Key Infrastructure (PKI) used to support attestations about Internet resource holdings. Each organization that distributes IP addresses or Autonomous System (AS) numbers to an organization will, in parallel, issue a certificate reflecting this distribution. These certificates will enable verification that the resources indicated in the certificate have been distributed to the holder of the

associated private key and that this organization is the current, unique holder of these resources.

Table of Contents

1. Introduction.....	7
1.1. Overview.....	8
1.2. Document name and identification.....	8
1.3. PKI participants.....	9
1.3.1. Certification authorities.....	9
1.3.2. Registration authorities.....	9
1.3.3. Subscribers.....	9
1.3.4. Relying parties.....	9
1.3.5. Other participants.....	10
1.4. Certificate usage.....	10
1.4.1. Appropriate certificate uses.....	10
1.4.2. Prohibited certificate uses.....	10
1.5. Policy administration.....	10
1.5.1. Organization administering the document.....	10
1.5.2. Contact person.....	10
1.5.4. CP approval procedures.....	11
1.6. Definitions and acronyms.....	11
2. Publication And Repository Responsibilities.....	13
2.1. Repositories.....	13
2.2. Publication of certification information.....	13
2.3. Time or frequency of publication.....	13
2.4. Access controls on repositories.....	14
3. Identification and Authentication.....	15
3.1. Naming.....	15
3.1.1. Types of names.....	15
3.1.2. Need for names to be meaningful.....	15
3.1.3. Anonymity or pseudonymity of subscribers.....	15
3.1.4. Rules for interpreting various name forms.....	15
3.1.5. Uniqueness of names.....	15
3.2. Initial identity validation.....	16
3.2.1. Method to prove possession of private key.....	16
3.2.2. Authentication of organization identity.....	16
3.2.3. Authentication of individual identity.....	16
3.2.4. Non-verified subscriber information.....	16
3.2.5. Validation of authority.....	16
3.2.6. Criteria for interoperation.....	17

3.3. Identification and authentication for re-key requests....	17
3.3.1. Identification and authentication for routine re-key.....	17
3.3.2. Identification and authentication for re-key after revocation.....	17
3.4. Identification and authentication for revocation request.....	17
4. Certificate Life-Cycle Operational Requirements.....	19
4.1. Certificate Application.....	19
4.1.1. Who can submit a certificate application.....	19
4.1.2. Enrollment process and responsibilities.....	19
4.2. Certificate application processing.....	19
4.2.1. Performing identification and authentication functions.....	19
4.2.2. Approval or rejection of certificate applications.....	19
4.2.3. Time to process certificate applications.....	20
4.3. Certificate issuance.....	20
4.3.1. CA actions during certificate issuance.....	20
4.3.2. Notification to subscriber by the CA of issuance of certificate.....	20
4.4. Certificate acceptance.....	20
4.4.1. Conduct constituting certificate acceptance.....	20
4.4.2. Publication of the certificate by the CA.....	20
4.4.3. Notification of certificate issuance by the CA to other entities.....	20
4.5. Key pair and certificate usage.....	21
4.5.1. Subscriber private key and certificate usage.....	21
4.5.2. Relying party public key and certificate usage.....	21
4.6. Certificate renewal.....	21
4.6.1. Circumstance for certificate renewal.....	22
4.6.2. Who may request renewal.....	22
4.6.3. Processing certificate renewal requests.....	22
4.6.4. Notification of new certificate issuance to subscriber.....	22
4.6.5. Conduct constituting acceptance of a renewal certificate.....	22
4.6.6. Publication of the renewal certificate by the CA....	23
4.6.7. Notification of certificate issuance by the CA to other entities.....	23
4.7. Certificate re-key.....	23
4.7.1. Circumstance for certificate re-key.....	23
4.7.2. Who may request certification of a new public key.....	23
4.7.3. Processing certificate re-keying requests.....	24

4.7.4.	Notification of new certificate issuance to subscriber.....	24
4.7.5.	Conduct constituting acceptance of a re-keyed certificate.....	24
4.7.6.	Publication of the re-keyed certificate by the CA.....	24
4.7.7.	Notification of certificate issuance by the CA to other entities.....	24
4.8.	Certificate modification.....	24
4.8.1.	Circumstance for certificate modification.....	24
4.8.2.	Who may request certificate modification.....	24
4.8.3.	Processing certificate modification requests.....	25
4.8.4.	Notification of new certificate issuance to subscriber.....	25
4.8.5.	Conduct constituting acceptance of modified certificate.....	25
4.8.6.	Publication of the modified certificate by the CA.....	25
4.8.7.	Notification of certificate issuance by the CA to other entities.....	25
4.9.	Certificate revocation and suspension.....	25
4.9.1.	Circumstances for revocation.....	25
4.9.2.	Who can request revocation.....	25
4.9.3.	Procedure for revocation request.....	25
4.9.4.	Revocation request grace period.....	26
4.9.5.	Time within which CA must process the revocation request.....	26
4.9.6.	Revocation checking requirement for relying parties.....	26
4.9.7.	CRL issuance frequency.....	26
4.9.8.	Maximum latency for CRLs.....	26
4.10.	Certificate status services.....	26
5.	Facility, Management, And Operational Controls.....	28
5.1.	Physical controls.....	28
5.1.1.	Site location and construction.....	28
5.1.2.	Physical access.....	28
5.1.3.	Power and air conditioning.....	28
5.1.4.	Water exposures.....	28
5.1.5.	Fire prevention and protection.....	28
5.1.6.	Media storage.....	28
5.1.7.	Waste disposal.....	28
5.1.8.	Off-site backup.....	28
5.2.	Procedural controls.....	28
5.2.1.	Trusted roles.....	29
5.2.2.	Number of persons required per task.....	29

5.2.3. Identification and authentication for each role.....	29
5.2.4. Roles requiring separation of duties.....	29
5.3. Personnel controls.....	29
5.4. Audit logging procedures.....	29
5.4.1. Types of events recorded.....	29
5.4.2. Frequency of processing log.....	29
5.4.3. Retention period for audit log.....	30
5.4.4. Protection of audit log.....	30
5.4.5. Audit log backup procedures.....	30
5.4.8. Vulnerability assessments.....	30
5.6. Key changeover.....	30
5.8. CA or RA termination.....	30
6. Technical Security Controls.....	31
6.1. Key pair generation and installation.....	31
6.1.1. Key pair generation.....	31
6.1.2. Private key delivery to subscriber.....	31
6.1.3. Public key delivery to certificate issuer.....	31
6.1.4. CA public key delivery to relying parties.....	31
6.1.5. Key sizes.....	32
6.1.6. Public key parameters generation and quality checking.....	32
6.1.7. Key usage purposes (as per X.509 v3 key usage field).....	32
6.2. Private Key Protection and Cryptographic Module Engineering Controls.....	32
6.2.1. Cryptographic module standards and controls.....	32
6.2.2. Private key (n out of m) multi-person control.....	32
6.2.3. Private key escrow.....	32
6.2.4. Private key backup.....	32
6.2.5. Private key archival.....	33
6.2.6. Private key transfer into or from a cryptographic module.....	33
6.2.7. Private key storage on cryptographic module.....	33
6.2.8. Method of activating private key.....	33
6.2.9. Method of deactivating private key.....	33
6.2.10. Method of destroying private key.....	33
6.2.11. Cryptographic Module Rating.....	33
6.3. Other aspects of key pair management.....	33
6.3.1. Public key archival.....	33
6.3.2. Certificate operational periods and key pair usage periods.....	34
6.4. Activation data.....	34
6.5. Computer security controls.....	34
6.6. Life cycle technical controls.....	34

6.6.1. System development controls.....	34
6.6.2. Security management controls.....	34
6.6.3. Life cycle security controls.....	34
6.7. Network security controls.....	34
6.8. Time-stamping.....	35
7. Certificate and CRL Profiles.....	36
8. Compliance Audit And Other Assessments.....	37
9. Other Business And Legal Matters.....	38
9.12. Amendments.....	38
9.12.1. Procedure for amendment.....	38
9.12.2. Notification mechanism and period.....	38
9.12.3. Circumstances under which OID must be changed.....	38
10. Security Considerations.....	39
11. IANA Considerations.....	39
12. Acknowledgments.....	39
13. References.....	40
13.1. Normative References.....	40
13.2. Informative References.....	40
Authors' Addresses:.....	41
Pre-5378 Material Disclaimer.....	42
Copyright Statement.....	42

1. Introduction

This document describes the certificate policy for a Public Key Infrastructure (PKI) used to attest to Internet number resource holdings (INRs) (IP addresses or Autonomous System (AS) numbers). An organization that distributes INRs to another organization MAY, in parallel, issue a certificate reflecting this distribution. These certificates will enable verification that the resources indicated in the certificate have been distributed to the holder of the associated private key and that this organization is the current holder of these resources.

The most important and distinguishing aspect of the PKI for which this policy was created is that it does not purport to identify an INR holder via the subject name contained in the certificate issued to that entity. Rather, each certificate issued under this policy is intended to enable an entity to assert, in a verifiable fashion, that it is the current holder of an INR based on the current records of the entity responsible for the resources in question. Verification of the assertion is based on two criteria: the ability of the entity to digitally sign data that is verifiable using the public key contained in the corresponding certificate, and validation of that certificate in the context of this PKI.

This PKI is designed exclusively for use in support of validation of claims related to current INR holdings. This includes any certificates issued in support of operation of this infrastructure, e.g., for integrity or access control of the repository system described in section 2.4. Such transitive uses of certificates also are permitted under this policy. Use of the certificates and certificate revocation lists (CRLs) managed under this PKI for any other purpose is a violation of this CP, and relying parties (RPs) SHOULD reject certificates presented for such uses.

Note: This document is based on the template specified in the Internet Engineering Task Force (IETF) standards document RFC 3647 [RFC3647]. In the interest of keeping the document as short as reasonable, a number of sections contained in the template are omitted from this policy because they did not apply to this PKI. However, we have retained the section numbering scheme employed in the RFC to facilitate comparison with the outline in Section 6 of the RFC. Each of these omitted sections should be read as "No stipulation" in CP/CPS parlance.

Conventions used in this document

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

1.1. Overview

This PKI is designed to support validation of claims by current holders of INRs, in accordance with the records of the organizations that act as Certification Authorities (CAs) in this PKI. The ability to verify such claims is essential to ensuring the unambiguous distribution of these resources.

The structure of the RPKI is modeled on the existing organizational structure that is already responsible for IP address and AS number resource allocation. In this allocation hierarchy, IANA allocates resources to five Regional Internet Registries (RIRs), each of which manages address and AS number allocation within a defined geopolitical region. The RIRs in turn allocate resources to Internet Service Providers, to subscribers with so-called provider-independent ("portable") allocations, and in some regions, to National Internet Registries (NIRs). (The term LIR is used in some regions to refer to what other regions define as an ISP. Throughout the rest of this document we will use the term ISP to simplify references to these entities.)

This PKI encompasses several types of certificates (see IETF document draft-ietf-sidr-arch-xx [ARCH] for more details):

- . CA certificates for each organization distributing INRs, and for INR holder
- . End-entity (EE) certificates for organizations to validate digital signatures on RPKI-signed objects

1.2. Document name and identification

The name of this document is "Certificate Policy (CP) for the Resource PKI (RPKI)".

This policy has been assigned the following OID:

```
id-cp-ipAddr-asNumber OBJECT IDENTIFIER ::= { iso(1)
    identified-organization(3) dod(6) internet(1)
    security(5) mechanisms(5) pkix(7) cp(14) 2 }
```


1.3. PKI participants

Note: In a PKI, the term "subscriber" refers to an individual or organization that is a Subject of a certificate issued by a CA. The term is used in this fashion throughout this document, without qualification, and should not be confused with the networking use of the term to refer to an individual or organization that receives service from an ISP. In such cases the term "network subscriber" will be used. Also note that, for brevity, this document always refers to PKI participants as organizations or entities, even though some of them are individuals.

1.3.1. Certification authorities

The organizations that distribute IP addresses and AS numbers (IANA, RIRs, NIRs, ISPs) act as CAs in this PKI.

Organizations that do not distribute INRs, but hold such resources also act as CAs when they create EE certificates.

1.3.2. Registration authorities

This PKI does not require establishment or use of a separate registration authority (RA) in conjunction with the CA function. The RA function MUST be provided by the same entity operating as a CA, e.g., entities listed in Section 1.4.1. An entity acting as a CA in this PKI already has a formal relationship with each organization to which it distributes INRs. These organizations already perform the RA function implicitly since they already assume responsibility for distributing INRs.

1.3.3. Subscribers

These are the organizations receiving distributions of INRs - RIRs, NIRs, ISPs, and other organizations.

Note that any of these organizations may have received distributions from more than one source, over time. This is true even for RIRs, which participate in inter-registry exchanges of address space. This PKI accommodates such relationships.

1.3.4. Relying parties

Entities or individuals that act in reliance on certificates or RPKI-signed objects issued under this PKI are relying parties. Relying parties may or may not be subscribers within this PKI. (See section 1.7 for the definition of an RPKI-signed object.)

1.3.5. Other participants

Every organization that undertakes a role as a CA in this PKI is responsible for populating the RPKI distributed repository system with the certificates, CRLs, and RPKI-signed objects that it issues. The organization MAY operate its own publication point or it MAY outsource this function (See sections 2.1 and 2.2.)

1.4. Certificate usage

1.4.1. Appropriate certificate uses

The certificates issued under this hierarchy are for authorization in support of validation of claims of current holdings of INRs.

Additional uses of the certificates, consistent with the basic goal cited above, also are permitted under this policy. For example, certificates may be issued in support of integrity and access control for the repository system described in 2.4. Such transitive uses are permitted under this policy.

Some of the certificates that may be issued under this PKI could be used to support operation of this infrastructure, e.g., access control for the repository system as described in 2.4. Such uses also are permitted under this policy.

1.4.2. Prohibited certificate uses

Any uses other than those described in Section 1.4.1 are prohibited under this policy.

1.5. Policy administration

1.5.1. Organization administering the document

This CP is administered by

Internet Engineering Steering Group
c/o Internet Society
1775 Wiehle Avenue, Suite 201
Reston, VA 20190-5108
U.S.A.

1.5.2. Contact person

The contact information is

iesg@ietf.org

+1-703-439-2120 (Internet Society)

1.5.4. CP approval procedures

The IESG MUST approve a replacement BCP that either updates or obsoletes this BCP, following the procedures of the IETF Standards Process as defined in RFC 2026 [RFC2026].

1.6. Definitions and acronyms

CPS - Certification Practice Statement. A CPS is a document that specifies the practices that a Certification Authority employs in issuing certificates in this PKI.

Distribution of INRs - A process of distribution of the INRs along the respective number hierarchy. IANA distributes blocks of IP addresses and AS Numbers to the five Regional Internet Registries (RIRs). RIRs distribute smaller address blocks and AS Numbers to organizations within their service regions, who in turn distribute IP addresses to their customers.

IANA - Internet Assigned Numbers Authority. IANA is responsible for global coordination of the IP addressing systems and AS numbers used for routing internet traffic. IANA distributes INRs to Regional Internet Registries (RIRs).

INRs - Internet Number Resources. INRs are number values for three protocol parameter sets, namely:

- . IP Version 4 addresses,
- . IP version 6 addresses, and
- . Identifiers used in Internet inter-domain routing, currently Border Gateway Protocol-4 AS numbers.

ISP - Internet Service Provider. This is an organization managing and selling Internet services to other organizations.

LIR - In some regions, the term Local Internet Registry (LIR) is used to refer to what is called an ISP in other regions.

NIR - National Internet Registry. This is an organization that manages the distribution of INRs for a portion of the geopolitical area covered by a Regional Registry. NIRs form an optional second tier in the tree scheme used to manage INRs.

RIR - Regional Internet Registry. This is an organization that manages the distribution of INRs for a geopolitical area.

RPKI-signed object - An RPKI-signed object is a digitally signed data object (other than a certificate or CRL) declared to be such by a standards track RFC, and that can be validated using certificates issued under this PKI. The content and format of these data constructs depend on the context in which validation of claims of current holdings of INRs takes place. Examples of these objects are repository manifests and CRLs.

2. Publication And Repository Responsibilities

2.1. Repositories

Certificates, CRLs, and RPKI-signed objects (intended for public consumption) MUST be made available for downloading by all relying parties, to enable them to validate this data. This motivates use of a robust, distributed repository system. Each CA MUST maintain a publicly accessible online repository and publish all RPKI-signed objects (intended for public consumption) via this repository in a manner that conforms with RFCwww [RFCwww] (This function MAY be outsourced, as noted in Section 2.2 below.) The collection of repositories forms the RPKI distributed repository system.

2.2. Publication of certification information

Each CA MUST publish the certificates (intended for public consumption) that it issues via the repository system.

Each CA MUST publish the CRLs (intended for public consumption) that it issues via the repository system.

Each CA MUST publish its RPKI-signed objects (intended for public consumption) via the repository system.

Each CA that issues certificates to entities outside of its administrative domain SHOULD create and publish a CPS that meets the requirements set forth in this CP. Publication means that the entities to which the CA issues certificates MUST be able to acquire a copy of the CPS, and MUST be able to ascertain when the CPS changes. (An organization that does not allocate or assign INRs does not need to create or publish a CPS.)

An organization MAY choose to outsource publication of RPKI data - certificates, CRLs, and other RPKI-signed objects.

The CP will be published as an IETF RFC and will be available from the IETF RFC repository.

2.3. Time or frequency of publication

The CPS for each CA MUST specify the following information:

The period of time within which a certificate will be published after the CA issues the certificate.

The period of time within which a CA will publish a CRL with an entry for a revoked certificate after it revokes that certificate.

Expired and revoked certificates SHOULD be removed from the RPKI repository system, upon expiration or revocation, respectively. Also, please note that each CA MUST publish its CRL prior to the nextUpdate value in the scheduled CRL previously issued by the CA.

2.4. Access controls on repositories

Each CA or repository operator MUST implement access controls to prevent unauthorized persons from adding, modifying or deleting repository entries. A CA or repository operator MUST NOT intentionally use technical means of limiting read access to its CPS, certificates, CRLs or RPKI-signed objects. This data is supposed to be accessible to the public.

3. Identification and Authentication

3.1. Naming

3.1.1. Types of names

The distinguished name for every CA and end-entity consists of a single Common Name (CN) attribute with a value generated by the issuer of the certificate. Optionally, the serialNumber attribute MAY be included along with the common name (to form a terminal relative distinguished name set), to distinguish among successive instances of certificates associated with the same entity.

3.1.2. Need for names to be meaningful

The Subject name in each certificate SHOULD NOT be "meaningful", i.e., the name is NOT intended to convey the identity of the Subject to relying parties. The rationale here is that certificates issued under this PKI are used for authorization in support of applications that make use of attestations of INR holdings. They are not used to identify Subjects.

3.1.3. Anonymity or pseudonymity of subscribers

Although Subject (and Issuer) names need not be meaningful, and may appear "random," anonymity is not a function of this PKI, and thus no explicit support for this feature is provided.

3.1.4. Rules for interpreting various name forms

None

3.1.5. Uniqueness of names

There is no guarantee that subject names are globally unique in this PKI. Each CA certifies Subject names that MUST be unique among the certificates that it issues. Although it is desirable that these Subject names be unique throughout the PKI, name uniqueness within the RPKI cannot be guaranteed.

However, Subject names in certificates SHOULD be constructed in a way that minimizes the chances that two entities in the RPKI will be assigned the same name. The RPKI certificate profile [RFCwww] provides an example of how to generate (meaningless) subject names in a way that minimizes the likelihood of collisions.

3.2. Initial identity validation

3.2.1. Method to prove possession of private key

Each CA operating within the context of this PKI MUST require each Subject to demonstrate proof-of-possession (PoP) of the private key corresponding to the public key in the certificate, prior to issuing the certificate. The means by which PoP is achieved is determined by each CA and MUST be declared in the CPS of that CA.

3.2.2. Authentication of organization identity

Each CA operating within the context of this PKI MUST employ procedures to ensure that each certificate it issues accurately reflects its records with regard to the organization to which the CA has distributed the INRs identified in the certificate. The specific procedures employed for this purpose MUST be described by the CPS for each CA. Relying parties can expect each CA to employ procedures commensurate with those it already employs as a registry or ISP, in the management of the INRs. This authentication is solely for use by each CA in dealing with the organizations to which it distributes INRs, and thus should not be relied upon outside of this CA-subscriber relationship.

3.2.3. Authentication of individual identity

Each CA operating within the context of this PKI MUST employ procedures to identify at least one individual as a representative of each organization that is an INR holder. The specific means by which each CA authenticates individuals as representatives for an organization MUST be described by the CPS for each CA. Relying parties can expect each CA to employ procedures commensurate with those it already employs as a registry or ISP, in authenticating individuals as representatives for INR holders.

3.2.4. Non-verified subscriber information

A CA MUST NOT include any non-verified subscriber data in certificates issued under this certificate policy except for SIA extensions.

3.2.5. Validation of authority

Each CA operating within the context of this PKI MUST employ procedures to verify that an individual claiming to represent an organization to which a certificate is issued, is authorized to represent that organization in this context. The procedures MUST be described by the CPS for the CA. Relying parties can expect each CA

to employ procedures commensurate with those it already employs as a registry or ISP, in authenticating individuals as representatives for INR holders.

3.2.6. Criteria for interoperation

This PKI is neither intended nor designed to interoperate with any other PKI.

3.3. Identification and authentication for re-key requests

3.3.1. Identification and authentication for routine re-key

Each CA operating within the context of this PKI MUST employ procedures to ensure that an organization requesting a re-key is the legitimate holder of the certificate to be re-keyed and associated INRs and MUST require PoP of the private key corresponding to the new public key. The procedures employed for these purposes MUST be described in the CPS for the CA. With respect to authentication of the holder of the INRs, relying parties can expect each CA to employ procedures commensurate with those it already employs as a registry or ISP, in the management of INRs.

Note: An issuer MAY choose to require periodic re-keying consistent with contractual agreements with the recipient. If so, this MUST be described by the CPS for the CA.

3.3.2. Identification and authentication for re-key after revocation

Each CA operating within the context of this PKI MUST employ procedures to ensure that an organization requesting a re-key after revocation is the same entity to which the revoked certificate was issued and is the legitimate holder of the associated INR. The CA MUST require PoP of the private key corresponding to the new public key. The specific procedures employed for these purposes MUST be described by the CPS for the CA. With respect to authentication of the holder of the INRs, relying parties can expect each CA to employ procedures commensurate with those it already employs as a registry or ISP, in the management of INRs. Note that there MAY be different procedures for the case where the legitimate subject still possesses the original private key as opposed to the case when it no longer has access to that key.

3.4. Identification and authentication for revocation request

Each CA operating within the context of this PKI MUST employ procedures to ensure that:

- . an organization requesting revocation is the legitimate holder of the certificate to be revoked.
- . each certificate it revokes accurately reflects its records with regard to the organization to which the CA has distributed the INRs identified in the certificate.
- . an individual claiming to represent an organization for which a certificate is to be revoked, is authorized to represent that organization in this context.

The specific procedures employed for these purposes MUST be described by the CPS for the CA. Relying parties can expect each CA to employ procedures commensurate with those it already employs as a registry or ISP, in the management of INRs.

4. Certificate Life-Cycle Operational Requirements

4.1. Certificate Application

4.1.1. Who can submit a certificate application

Any entity that distributes INRs SHOULD acquire a certificate. This includes Internet Registries and ISPs. Additionally, entities that hold INRs from an Internet Registry, or that are multi-homed, MAY acquire a certificate under this PKI. The (CA) certificates issued to these entities MUST include one or both of the extensions defined by RFC 3779 [RFC3779], X.509 Extensions for IP Addresses and AS Identifiers, as appropriate.

The application procedure MUST be described in the CPS for each CA.

4.1.2. Enrollment process and responsibilities

The enrollment process and procedures MUST be described by the CPS for each CA. An entity that desires one or more certificates should contact the organization from which it receives its INRs.

4.2. Certificate application processing

CAs SHOULD make use of existing standards for certificate application processing. Section 6 of the resource certificate profile [RFCyyyy] defines the standard certificate request formats that MUST be supported

Each CA MUST define the certificate request/response standards that it employs, via its CPS.

4.2.1. Performing identification and authentication functions

Existing practices employed by registries and ISPs to identify and authenticate organizations that receive INRs form the basis for issuance of certificates to these subscribers. It is important to note that the Resource PKI SHOULD never be used to authenticate the identity of an organization, but rather to bind subscribers to the INRs they hold. Because identity is not being vouched for by this PKI, certificate application procedures need not verify legal organization names, etc.

4.2.2. Approval or rejection of certificate applications

Certificate applications MUST be approved based on the normal business practices of the entity operating the CA, based on the CA's records of INR holders. Each CA MUST follow the procedures specified

in 3.2.1 to verify that the requester holds the private key corresponding to the public key that will be bound to the certificate the CA issues to the requestor. The details of how certificate applications are approved MUST be described in the CPS for the CA in question.

4.2.3. Time to process certificate applications

No stipulation. Each CA MUST declare its expected time frame to process (approve, issue and publish) a certificate application as part of its CPS.

4.3. Certificate issuance

4.3.1. CA actions during certificate issuance

If a CA determines that the request is acceptable, it MUST issue the corresponding certificate and publish it in the RPKI distributed repository system via publication of the certificate at the CA's repository publication point.

4.3.2. Notification to subscriber by the CA of issuance of certificate

The CA MUST notify the subscriber when the certificate is published. The means by which a subscriber is notified is defined by each CA in its CPS.

4.4. Certificate acceptance

4.4.1. Conduct constituting certificate acceptance

Within the timeframe specified in its CPS, the CA MUST place the certificate in the repository and notify the subscriber. This MAY be done without subscriber review and acceptance. Each CA MUST state in its CPS the procedures it follows for publishing of the certificate and notification to the subscriber.

4.4.2. Publication of the certificate by the CA

Certificates MUST be published in the RPKI distributed repository system via publication of the certificate at the CA's repository publication point as per the conduct described in 4.4.1. The procedures for publication MUST be defined by each CA in its CPS.

4.4.3. Notification of certificate issuance by the CA to other entities

The CPS of each CA MUST indicate whether any other entities will be notified when a certificate is issued.

4.5. Key pair and certificate usage

A summary of the use model for the RPKI is provided below.

4.5.1. Subscriber private key and certificate usage

Each holder of an INR is eligible to request an X.509 CA certificate containing appropriate RFC 3779 extensions. Holders of CA resource certificates also MAY issue EE certificates to themselves to enable verification of RPKI-signed objects that they generate.

4.5.2. Relying party public key and certificate usage

Reliance on a certificate must be reasonable under the circumstances. If the circumstances indicate a need for additional assurances, the relying party must obtain such assurances in order for such reliance to be deemed reasonable.

Before any act of reliance, relying parties MUST independently (1) verify that the certificate will be used for an appropriate purpose that is not prohibited or otherwise restricted by this CP (see section 1.5), and (2) assess the status of the certificate and all the CAs in the chain (terminating at a TA accepted by the RP) that issued the certificates relevant to the certificate in question. If any of the certificates in the certificate chain have been revoked, the relying party is solely responsible to determine whether reliance on a digital signature to be verified by the certificate in question is acceptable. Any such reliance is made solely at the risk of the relying party.

If a relying party determines that use of the certificate is appropriate, the relying party must utilize appropriate software and/or hardware to perform digital signature verification as a condition of relying on the certificate. Moreover the relying party MUST validate the certificate in a manner consistent with the RPKI certificate profile [RFCyyyy], which specifies the extended validation algorithm for RPKI certificates.

4.6. Certificate renewal

This section describes the procedures for certificate renewal. Certificate renewal is the issuance of a new certificate to replace an old one prior to its expiration. Only the validity dates and the serial number are changed. The public key and all other information remain the same.

4.6.1. Circumstance for certificate renewal

A certificate MUST be processed for renewal based on its expiration date or a renewal request from the subscriber. Prior to the expiration of an existing subscriber's certificate, it is the responsibility of the subscriber to renew the certificate to maintain continuity of certificate usage. If the issuing CA initiates the renewal process based on the certificate expiration date, then that CA MUST notify the holder in advance of the renewal process. The validity interval of the new (renewed) certificate SHOULD overlap that of the previous certificate, to ensure continuity of certificate usage. It is RECOMMENDED that the renewed certificate be issued and published at least 1 week prior to the expiration of the certificate it replaces.

Certificate renewal SHOULD incorporate the same public key as the previous certificate, unless the private key has been reported as compromised. If a new key pair is being used, the stipulations of Section 4.7 apply.

4.6.2. Who may request renewal

Only the certificate holder or the issuing CA may initiate the renewal process. The certificate holder MAY request an early renewal, for example, if it wishes to change the public key, or if it expects to be unavailable to support the renewal process during the normal expiration period. An issuing CA MAY initiate the renewal process based on the certificate expiration date.

4.6.3. Processing certificate renewal requests

Renewal procedures MUST ensure that the person or organization seeking to renew a certificate is in fact the subscriber (or authorized by the subscriber) of the certificate and the legitimate holder of the INR associated with the renewed certificate. Renewal processing MUST verify that the certificate in question has not been revoked.

4.6.4. Notification of new certificate issuance to subscriber

No additional stipulations beyond those of section 4.3.2.

4.6.5. Conduct constituting acceptance of a renewal certificate

No additional stipulations beyond those of section 4.4.1.

4.6.6. Publication of the renewal certificate by the CA

No additional stipulations beyond those of section 4.4.2.

4.6.7. Notification of certificate issuance by the CA to other entities

No additional stipulations beyond those of section 4.3.3.

4.7. Certificate re-key

This section describes the procedures for certificate re-key. Certificate re-key is the issuance of a new certificate to replace an old one because the key needs to be replaced. Unlike with certificate renewal, the public key is changed.

4.7.1. Circumstance for certificate re-key

Re-key of a certificate SHOULD be performed only when required, based on:

1. knowledge or suspicion of compromise or loss of the associated private key, or
2. the expiration of the cryptographic lifetime of the associated key pair

A CA re-key operation has dramatic consequences, requiring the re-issuance of all certificates issued by the re-keyed entity. So it should be performed only when necessary and in a way that preserves the ability of relying parties to validate certificates whose validation path includes the re-keyed entity. CA key rollover MUST follow the procedures defined in RFCxxxx [RFCxxxx].

Note that if a certificate is revoked to replace the RFC 3779 extensions, the replacement certificate MUST incorporate the same public key rather than a new key. This applies to when one is adding INRs (revocation not required) and to when one is removing INRs (revocation required (see Section 4.8.1)).

If the re-key is based on a suspected compromise, then the previous certificate MUST be revoked.

4.7.2. Who may request certification of a new public key

The holder of the certificate may request a re-key. In addition, the CA that issued the certificate MAY chose to initiate a rekey based on a verified compromise report.

4.7.3. Processing certificate re-keying requests

The re-key process follows the general procedures of certificate generation as defined in section 4.3.

4.7.4. Notification of new certificate issuance to subscriber

No additional stipulations beyond those of section 4.3.2.

4.7.5. Conduct constituting acceptance of a re-keyed certificate

No additional stipulations beyond those of section 4.4.1.

4.7.6. Publication of the re-keyed certificate by the CA

No additional stipulations beyond those of section 4.4.2.

4.7.7. Notification of certificate issuance by the CA to other entities

No additional stipulations beyond those of section 4.3.3.

4.8. Certificate modification

4.8.1. Circumstance for certificate modification

Modification of a certificate occurs to implement changes to selected attribute values in a certificate. In the context of the RPKI, the only changes that are accommodated by certificate modification are changes to the INR holdings described by the RFC 3779 extension and changes to the SIA extension.

When a certificate modification is approved, a new certificate is issued. If no INR holdings are removed from the certificate, the new certificate **MUST** contain the same public key and the same expiration date as the original certificate, (but with the SIA extension and/or the INR set expanded). In this case, revocation of the previous certificate is not required.

When previously distributed INRs are removed from a certificate, then the old certificate **MUST** be revoked and a new certificate **MUST** be issued, reflecting the changed INR holdings. (The SIA extension in the new certificate will be unchanged, unless the affected INR holder supplies a new SIA value.)

4.8.2. Who may request certificate modification

Either the certificate holder or the issuer may initiate the certificate modification process.

4.8.3. Processing certificate modification requests

The CA MUST determine that the requested modification is appropriate and that the procedures for the issuance of a new certificate are followed (see Section 4.3).

4.8.4. Notification of new certificate issuance to subscriber

No additional stipulations beyond those of section 4.3.2.

4.8.5. Conduct constituting acceptance of modified certificate

No additional stipulations beyond those of section 4.4.1.

4.8.6. Publication of the modified certificate by the CA

No additional stipulations beyond those of section 4.4.2.

4.8.7. Notification of certificate issuance by the CA to other entities

No additional stipulations beyond those of section 4.3.3.

4.9. Certificate revocation and suspension

4.9.1. Circumstances for revocation

A certificate MUST be revoked (and published on a CRL) if there is reason to believe that there has been a compromise of a subscriber's private key. A certificate also MAY be revoked to invalidate a data object signed by the private key associated with that certificate. Other circumstances that justify revocation of a certificate MAY be specified in a CA's CPS.

Note: If new INRs are being added to an organization's existing distribution, the old certificate need not be revoked. Instead, a new certificate MAY be issued with both the old and the new resources and the old key. If INRs are being removed or if there has been a key compromise, then the old certificate MUST be revoked (and a re-key MUST be performed in the event of key compromise).

4.9.2. Who can request revocation

This MUST be defined in the CPS of the relevant organization.

4.9.3. Procedure for revocation request

A subscriber MAY submit a request to the certificate issuer for a revocation. This request MUST identify the certificate to be revoked

and MUST be authenticated. The procedures for making the request MUST be described in the CPS for each CA. The RPKI provisioning document [PROV] describes a protocol that MAY be used to make revocation requests.

A certificate issuer MUST notify the subscriber when revoking a certificate. The notification requirement is satisfied by CRL publication. The CPS for a CA MUST indicate the means by which the CA will inform a subscriber of certificate revocation.

4.9.4. Revocation request grace period

A subscriber SHOULD request revocation as soon as possible after the need for revocation has been identified. There is no specified grace period for the subscriber in this process.

4.9.5. Time within which CA must process the revocation request

No stipulation. Each CA SHOULD specify its expected revocation processing time in its CPS.

4.9.6. Revocation checking requirement for relying parties

A relying party MUST acquire and check the most recent, scheduled CRL from the issuer of the certificate, whenever the relying party validates a certificate.

4.9.7. CRL issuance frequency

The CRL issuance frequency MUST be determined by each CA and stated in its CPS. Each CRL carries a nextScheduledUpdate value and a new CRL MUST be published at or before that time. A CA MUST set the nextUpdate value when it issues a CRL, to signal when the next scheduled CRL will be issued.

4.9.8. Maximum latency for CRLs

The CPS for each CA MUST specify the maximum latency associated with posting its CRL to the repository system.

4.10. Certificate status services

This PKI does not make provision for use of OCSP or SCVP, because it is anticipated that the primary RPs (ISPs) will acquire and validate certificates for all participating resource holders. These protocols are not designed for such large-scale, bulk certificate status checking. RPs MUST check for new CRLs at least daily. It is RECOMMENDED that RPs perform this check several times per day, but

no more than 8-12 times per day (to avoid excessive repository accesses).

5. Facility, Management, And Operational Controls

5.1. Physical controls

Each CA MUST maintain physical security controls for its operation that are commensurate with those employed by the organization in the management of INR distribution. The physical controls employed for CA operation MUST be specified in its CPS. Possible topics to be covered in the CPS are shown below. (These sections are taken from [RFC3647].)

5.1.1. Site location and construction

5.1.2. Physical access

5.1.3. Power and air conditioning

5.1.4. Water exposures

5.1.5. Fire prevention and protection

5.1.6. Media storage

5.1.7. Waste disposal

5.1.8. Off-site backup

5.2. Procedural controls

Each CA MUST maintain procedural security controls that are commensurate with those employed by the organization in the management of INR distribution. The procedural security controls employed for CA operation MUST be specified in its CPS. Possible topics to be covered in the CPS are shown below. (These sections are taken from [RFC3647].)

5.2.1. Trusted roles

5.2.2. Number of persons required per task

5.2.3. Identification and authentication for each role

5.2.4. Roles requiring separation of duties

5.3. Personnel controls

Each CA MUST maintain personnel security controls that are commensurate with those employed by the organization in the management of INR distribution. The details for each CA MUST be specified in its CPS.

5.4. Audit logging procedures

Details of how a CA implements the audit logging described in this section (5.4.1 to 5.4.8) MUST be addressed in its CPS.

5.4.1. Types of events recorded

Audit records MUST be generated for the basic operations of the certification authority computing equipment. Audit records MUST include the date, time, responsible user or process, and summary content data relating to the event. Auditable events include:

- . Access to CA computing equipment (e.g., logon, logout)
- . Messages received requesting CA actions (e.g., certificate requests, certificate revocation requests, compromise notifications)
- . Certificate creation, modification, revocation, or renewal actions
- . Posting of any material to a repository
- . Any attempts to change or delete audit data
- . Key generation
- . Software and/or configuration updates to the CA
- . Clock adjustments

5.4.2. Frequency of processing log

Each CA MUST establish its own procedures for review of audit logs.

5.4.3. Retention period for audit log

Each CA MUST establish its own policies for retention of audit logs.

5.4.4. Protection of audit log

The audit log SHOULD be protected based on current industry standards.

5.4.5. Audit log backup procedures

The audit log SHOULD be backed up based on current industry standards.

5.4.8. Vulnerability assessments

The RPKI subsystems of a registry or ISP SHOULD participate in any vulnerability assessments that these organizations run as part of their normal business practice.

5.6. Key changeover

When a CA wishes to change keys, it MUST acquire a new certificate containing its new public key. See [RFCxxxx] for a description of how key changeover is effected in the RPKI.

5.8. CA or RA termination

In the RPKI, each subscriber acts as a CA authoritative for the specified INRs that were distributed to that entity. Procedures associated with the termination of a CA MUST be described in the CPS for that CA. These procedures MUST include a provision to notify each entity that issued a certificate to the organization that is operating the CA that is terminating.

Since the RA function MUST be provided by the same entity operating as the CA (see Section 1.4.2), there are no separate stipulations for RAs.

6. Technical Security Controls

The organizations that distribute INRs to network subscribers are authoritative for these distributions. This PKI is designed to enable ISPs and network subscribers to demonstrate that they are the holders of the INRs that have been distributed to them. Accordingly, the security controls used by CAs and subscribers for this PKI need only to be as secure as those that apply to the procedures for administering the distribution of INR data by the extant organizations. Details of each CA's security controls MUST be described in the CPS issued by the CA.

6.1. Key pair generation and installation

6.1.1. Key pair generation

In most instances, public-key pairs will be generated by the subject, i.e., the organization receiving the distribution of INRs. However, some CAs MAY offer to generate key pairs on behalf of their subjects at the request of the subjects, e.g., to accommodate subscribers who do not have the ability to perform key generation in a secure fashion. (The CA has to check the quality of the keys only if it generates them (see 6.1.6)). Since the keys used in this PKI are not for non-repudiation purposes, generation of key pairs by CAs does not inherently undermine the security of the PKI. Each CA MUST describe its key pair generation procedures in its CPS.

6.1.2. Private key delivery to subscriber

If a CA provides key pair generation services for subscribers, its CPS MUST describe the means by which private keys are delivered to subscribers in a secure fashion.

6.1.3. Public key delivery to certificate issuer

When a public key is transferred to the issuing CA to be certified, it MUST be delivered through a mechanism ensuring that the public key has not been altered during transit and that the subscriber possesses the private key corresponding to the transferred public key.

6.1.4. CA public key delivery to relying parties

CA public keys for all entities (other than trust anchors) are contained in certificates issued by other CAs. These certificates MUST be published in the RPKI distributed repository system. Relying parties download these certificates from the repositories. Public key values and associated data for (putative) trust anchors are

distributed out of band and accepted by relying parties on the basis of locally-defined criteria.

6.1.5. Key sizes

The algorithms and key sizes used in the RPKI are specified in RFC ZZZZ [RFCzzzz].

6.1.6. Public key parameters generation and quality checking

The public key parameters used in the RPKI are specified in RFC ZZZZ [RFCzzzz]. Each subscriber is responsible for performing checks on the quality of its key pair. A CA is not responsible for performing such checks for subscribers except in the case where the CA generates the key pair on behalf of the subscriber.

6.1.7. Key usage purposes (as per X.509 v3 key usage field)

The Key usage extension bit values used in the RPKI are specified in RFC YYYY [RFCyyyy].

6.2. Private Key Protection and Cryptographic Module Engineering Controls

6.2.1. Cryptographic module standards and controls

The cryptographic module standards and controls employed by each CA MUST be described in the CPS issued by that CA.

6.2.2. Private key (n out of m) multi-person control

CAs MAY employ multi-person controls to constrain access to their private keys, but this is not a requirement for all CAs in the PKI. The CPS for each CA MUST describe which, if any, multi-person controls it employs.

6.2.3. Private key escrow

No private key escrow procedures are required for the RPKI.

6.2.4. Private key backup

Because of the adverse operational implications associated with the loss of use of a CA private key in the PKI, each CA MUST employ a secure means to backup its private keys. The details of the procedures for backing up a CA's private key MUST be described in the CPS issued by the CA.

6.2.5. Private key archival

The details of the process and procedures used to archive the CA's private key MUST be described in the CPS issued by the CA.

6.2.6. Private key transfer into or from a cryptographic module

The details of the process and procedures used to transfer the CA's private key into or from a cryptographic module MUST be described in the CPS issued by the CA.

6.2.7. Private key storage on cryptographic module

The details of the process and procedures used to store the CA's private key on a cryptographic module and protect it from unauthorized use MUST be described in the CPS issued by the CA.

6.2.8. Method of activating private key

The details of the process and procedures used to activate the CA's private key MUST be described in the CPS issued by the CA.

6.2.9. Method of deactivating private key

The details of the process and procedures used to deactivate the CA's private key MUST be described in the CPS issued by the CA.

6.2.10. Method of destroying private key

The details of the process and procedures used to destroy the CA's private key MUST be described in the CPS issued by the CA.

6.2.11. Cryptographic Module Rating

The security rating of the cryptographic module MUST be described in the CPS issued by the CA.

6.3. Other aspects of key pair management

6.3.1. Public key archival

Because this PKI does not support non-repudiation, there is no need to archive public keys.

6.3.2. Certificate operational periods and key pair usage periods

The INRs held by a CA may periodically change when it receives new distributions. To minimize disruption, the CA key pair **MUST NOT** change when INRs are added to its certificate.

If ISP and network subscriber certificates are tied to the duration of service agreements, these certificates should have validity periods commensurate with the duration of these agreements. In any case, the validity period for certificates **MUST** be chosen by the issuing CA and described in its CPS.

6.4. Activation data

Each CA **MUST** document in its CPS how it will generate, install and protect its activation data.

6.5. Computer security controls

Each CA **MUST** document the technical security requirements it employs for CA computer operation in its CPS.

6.6. Life cycle technical controls

6.6.1. System development controls

The CPS for each CA **MUST** document any system development controls required by that CA, if applicable.

6.6.2. Security management controls

The CPS for each CA **MUST** document the security controls applied to the software and equipment used for this PKI. These controls **MUST** be commensurate with those used for the systems used by the CAs for managing the INRs.

6.6.3. Life cycle security controls

The CPS for each CA **MUST** document how the equipment (hardware and software) used for this PKI will be procured, installed, maintained, and updated. This **MUST** be done in a fashion commensurate with the way in which equipment for the management and distribution of INRs is handled.

6.7. Network security controls

The CPS for each CA **MUST** document the network security controls employed for CA operation. These **MUST** be commensurate with the

protection it employs for the computers used for managing distribution of INRs.

6.8. Time-stamping

The RPKI does not make use of time stamping.

7. Certificate and CRL Profiles

Please refer to the RPKI Certificate and CRL Profile [RFCyyyy].

8. Compliance Audit And Other Assessments

The Certificate Policy for a typical PKI defines the criteria against which prospective CAs are evaluated and establishes requirements that they must meet. In this PKI, the CAs are already authoritative for the management of INRs, and the PKI simply supports verification of the distribution of these resources to network subscribers. Accordingly, whatever audit and other assessments are already used to ensure the security of the management of INRs is sufficient for this PKI. The CPS for each CA MUST describe what audits and other assessments are used.

9. Other Business And Legal Matters

As noted throughout this certificate policy, the organizations managing the distribution of INRs are authoritative in their roles as managers of this data. They MUST operate this PKI to allow the holders of INRs to generate digitally signed data that attest to these distributions. Therefore, the manner in which the organizations in question manage their business and legal matters for this PKI MUST be commensurate with the way in which they already manage business and legal matters in their existing roles. Since there is no single set of responses to this section that would apply to all organizations, the topics listed in sections 4.9.1 to 4.9.11 and 4.9.13 to 4.9.17 of RFC 3647 SHOULD be covered in the CPS issued by each CA, although not every CA may choose to address all of these topics.

9.12. Amendments

9.12.1. Procedure for amendment

The procedure for amending this CP is via written notice from the IESG in the form of a new (BCP) RFC that updates or obsoletes this document.

9.12.2. Notification mechanism and period

Successive versions of the CP will be published with the statement "This CP takes effect on MM/DD/YYYY." MM/DD/YYYY MUST be a minimum of 6 months from the date of publication.

9.12.3. Circumstances under which OID must be changed

If the IESG judges that changes to the CP do not materially reduce the acceptability of certificates issued for RPKI purposes, there will be no change to the CP OID. If the IESG judges that changes to the CP do materially change the acceptability of certificates for RPKI purposes, then there will be a new CP OID.

10. Security Considerations

According to X.509, a certificate policy (CP) is "a named set of rules that indicates the applicability of a certificate to a particular community and/or class of applications with common security requirements." A CP may be used by a relying party to help in deciding whether a certificate, and the binding therein, are sufficiently trustworthy and otherwise appropriate for a particular application. This document describes the CP for the Resource Public Key Infrastructure (RPKI). There are separate documents (Certification Practice Statements (CPS's)) that cover the factors that determine the degree to which a relying party can trust the binding embodied in a certificate. The degree to which such a binding can be trusted depends on several factors, e.g., the practices followed by the certification authority (CA) in authenticating the subject; the CA's operating policy, procedures, and technical security controls, including the scope of the subscriber's responsibilities (for example, in protecting the private key), and the stated responsibilities and liability terms and conditions of the CA (for example, warranties, disclaimers of warranties, and limitations of liability).

Since name uniqueness within the RPKI cannot be guaranteed, there is a risk that two or more CAs in the RPKI will issue certificates and CRLs under the same Issuer name. Path validation implementations that conform to the resource certification path validation algorithm [see RFCyyyy] verify that the same key was used to sign both the target (the resource certificate) and the corresponding CRL. So a name collision will not change the result. Use of the basic X.509 path validation algorithm, which assumes name uniqueness, could result in a revoked certificate being accepted as valid or a valid certificate being rejected as revoked. Relying parties must ensure that the software they use to validate certificates issued under this policy verifies that the same key was used to sign both the certificate and the corresponding CRL, as specified in [RFCyyyy].

11. IANA Considerations

None.

12. Acknowledgments

The authors would like to thank Geoff Huston, Randy Bush, Andrei Robachevsky and other members of the RPKI community for reviewing this document and Matt Lepinski for his help with the formatting.

13. References

13.1. Normative References

- [ARCH] Lepinski M., Kent S., "An Infrastructure to Support Secure Internet Routing," work in progress.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels," BCP 14, RFC 2119, March 1997.
- [RFC2026] Bradner, S., "The Internet Standards Process - Revision 3," BCP 9, RFC 2026, October 1996.
- [RFC3779] Lynn, C., Kent, S., Seo, K., "X.509 Extensions for IP Addresses and AS Identifiers," RFC 3779, June 2004.
- [RFCwww] Huston, G., Loomans, R., and Michaelson, G., "A Profile for Resource Certificate Repository Structure," work in progress.
- [RFCxxxx] Huston, G., Michaelson, G., Kent, S., "CA Key Rollover in the RPKI," work in progress.
- [RFCyyyy] Huston, G., Michaelson, G., Loomans, R., "A Profile for X.509 PKIX Resource Certificates," work in progress.
- [RFCzzzz] Huston, G., "A Profile for Algorithms and Key Sizes for use in the Resource Public Key Infrastructure," work in progress.

13.2. Informative References

- [PROV] Huston, G., Loomans, R., Ellacott, B., Austein, R., "A Protocol for Provisioning Resource Certificates," work in progress.
- [RFC3647] Chokhani, S., Ford, W., Sabett, R., Merrill, C., Wu, S., "Internet X.509 Public Key Infrastructure Certificate Policy and Certification Practices Framework," RFC 3647, November 2003.

Authors' Addresses:

Stephen Kent
BBN Technologies
10 Moulton Street
Cambridge MA 02138
USA

Phone: +1 (617) 873-3988
Email: skent@bbn.com

Derrick Kong
BBN Technologies
Moulton Street
Cambridge MA 02138
USA

Phone: +1 (617) 873-1951
Email: dkong@bbn.com

Karen Seo
BBN Technologies
10 Moulton Street
Cambridge MA 02138
USA

Phone: +1 (617) 873-3152
Email: kseo@bbn.com

Ronald Watro
BBN Technologies
10 Moulton Street
Cambridge MA 02138
USA

Phone: +1 (617) 873-2551
Email: rwatro@bbn.com

Pre-5378 Material Disclaimer

This document may contain material from IETF Documents or IETF Contributions published or made publicly available before November 10, 2008. The person(s) controlling the copyright in some of this material may not have granted the IETF Trust the right to allow modifications of such material outside the IETF Standards Process. Without obtaining an adequate license from the person(s) controlling the copyright in such materials, this document may not be modified outside the IETF Standards Process, and derivative works of it may not be created outside the IETF Standards Process, except to format it for publication as an RFC or to translate it into languages other than English.

Copyright Statement

Copyright (c) 2010 IETF Trust and the persons identified as the document authors. All rights reserved.

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

SIDR
Internet-Draft
Intended status: BCP
Expires: May 13, 2011

G. Huston
G. Michaelson
APNIC
S. Kent
BBN
November 9, 2010

CA Key Rollover in the RPKI
draft-ietf-sidr-keyroll-04.txt

Abstract

This document describes an algorithm to allow an entity who undertakes the role of a Certification Authority in the Resource Public Key Infrastructure to perform a rollover of its key pair. This document also notes the requirements placed on Relying Parties who maintain a local cache of the objects that have been published in the distributed Resource Public Key Infrastructure repository publication structure.

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 <http://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 May 13, 2011.

Copyright Notice

Copyright (c) 2010 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect

to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	3
1.1. Terminology and Concepts	3
2. CA Key Rollover Procedure	3
3. Relying Party Requirements	7
4. Re-issuing Certificates and RPKI Signed Objects	7
4.1. CA Certificates	7
4.2. RPKI Signed Objects	8
5. Security Considerations	8
6. IANA Considerations	9
7. Acknowledgements	9
8. References	9
8.1. Normative References	9
8.2. Informative References	9
Authors' Addresses	10

1. Introduction

This document describes an algorithm to be employed by a Certification Authority (CA) in the Resource Public Key Infrastructure (RPKI) [ID.ietf-sidr-arch] to perform a rollover of its key pair.

This document defines a conservative procedure for such entities to follow when performing a key rollover, so that Relying Parties are in a position to be able to validate all authentic objects in the RPKI using the validation procedure described in [ID.ietf-sidr-arch] at all times.

1.1. Terminology and Concepts

It is assumed that the reader is familiar with the terms and concepts described in "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile" [RFC5280], "X.509 Extensions for IP Addresses and AS Identifiers" [RFC3779], the profile for RPKI Certificates [ID.ietf-sidr-res-certs], and the RPKI repository structure [ID.ietf-sidr-repos-struct] .

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119.

2. CA Key Rollover Procedure

A Certification Authority (CA) in the Resource Public Key Infrastructure (RPKI) is an entity that issues CA and End Entity (EE) certificates and Certificate Revocation Lists (CRLs). A CA instance is associated with a single key pair ([ID.ietf-sidr-res-certs]), implying that if key rollover is a regularly scheduled event then, over time, there will be many instances of a CA. The implication in the context of key rollover is that, strictly speaking, a CA does not perform a key rollover per se. In order to perform the equivalent of a key rollover, the CA creates a "new" instance of itself, with a new key pair, and then effectively substitutes this "new" CA instance into the RPKI hierarchy in place of the old CA instance.

There are several considerations regarding this procedure that MUST be followed by a CA performing a key rollover operation. The critical consideration is that the RPKI has potential application in the area of control of routing integrity [ID.ietf-sidr-arch], and key rollover should not cause any transient hiatus where a Relying Party (RP) is led to incorrect conclusions regarding the authenticity of attestations made in the context of the RPKI. A CA cannot assume

that all RPs will perform path validation and path discovery in the same fashion, and therefore the key rollover procedure MUST preserve the integrity of the CRL Distribution Points (CRLDP), Subject Information Access (SIA) and Authority Information Access (AIA) pointers in RPKI certificates.

In the procedure described here, the CA creates a "new" CA instance, and has the associated new public key published in the form of a "new" CA certificate. While the "current" and "new" CA instances share a single repository publication point, each CA has its own CRL and its own manifest. Initially, the "new" CA publishes an empty CRL and a manifest that contains a single entry for the CRL. The "current" CA also maintains its published CRL and manifest at this Repository publication point.

The CA performing key rollover waits for a period of time to afford every RP an opportunity to discover and retrieve this "new" CA certificate, and store it in its local RPKI Repository cache instance. This period of time is termed the "staging period". During this period, the CA will have a "new" CA instance, with no subordinate products, and a "current" CA instance that has issued all subordinate products. At the expiration of the staging period the "new" CA instance MUST replace all (valid) subordinate products of the "current" CA instance, overwriting the "current" subordinate products in the CA's repository publication point. When this process is complete the "current" CA instance is retired, and the "new" CA instance becomes the "current" CA.

During the transition of the "current" and "new" CA instances it is necessary for the "new" CA instance to re-issue all subordinate products of the "current" CA. The procedure described here requires that, with the exception of manifests and CRLs, the re-issued subordinate products be published using the same repository publication point object names, effectively overwriting the old objects with these re-issued objects. The intent of this overwriting operation is to ensure that the AIA pointers of subordinate products at lower tiers in the RPKI hierarchy remain correct, and that CA key rollover does not require any associated actions by any subordinate CA.

There are three CA states described here:

CURRENT:

The CURRENT CA is the active CA instance used to accept and process certificate issuance and revocation requests. The starting point for this algorithm is that the key of the CURRENT CA is to be rolled over.

NEW:

The NEW CA is the CA instance that is being created. The NEW CA is not active, and thus does not accept nor process certificate issuance and revocation requests. The NEW CA SHOULD issue a CRL and an EE certificate in association with its manifest to provide a trivial, complete, consistent instance of a CA.

OLD:

The CA instance is in the process of being removed. An OLD CA instance is unable to process any certificate issuance and revocation requests. An OLD CA instance will continue to issue regularly scheduled CRLs and issue an EE certificate as part of the process of updating its manifest to reflect the updated CRL.

To perform a key rollover operation the CA MUST perform the following steps in the order given here. Unless specified otherwise each step SHOULD be performed without any intervening delay. The process MUST be run through to completion.

1. Generate a new key pair for use by the NEW CA. Because the goal of this algorithm is key rollover, the key pair generated in this step MUST be different from the pair in use by the CURRENT CA.
2. Generate a certificate request with this key pair and pass the request to the CA that issued the CURRENT CA certificate. This request MUST include the same SIA extension that is present in the CURRENT CA certificate. This request, when satisfied, will result in the publication of the NEW CA certificate. This (NEW) CA certificate will contain a Subject Name selected by the issuer, which MUST be distinct from the Subject Name used in the CURRENT CA certificate. The CPS for the issuer of the NEW CA certificate will indicate the time frame within which a certificate request is expected to be processed.
3. Publish the NEW CA's CRL and manifest.

The steps involved here are:

- Wait for the issuer of the NEW CA to publish the NEW CA certificate.
- As quickly as possible following the publication of the NEW CA certificate, use the key pair associated with the NEW CA to generate an initial, empty CRL, and publish this CRL in the NEW CA's repository publication point.

It is RECOMMENDED that the CRL for the NEW CA have a nextUpdate value that will cause the CRL to be replaced at the end of the Staging Period (see in Step 4 below).

- Generate a new key pair, and generate an associated EE certificate request with an AIA value of the NEW CA's repository publication point. Pass this EE certificate request to the NEW CA, and use the returned (single-use) EE certificate as the NEW CA's manifest EE certificate.
 - Generate a manifest containing the new CA's CRL as the only entry, and sign it with the private key associated with the manifest EE certificate. Publish the manifest at the NEW CA's repository publication point.
 - Destroy the private key associated with the manifest EE certificate.
4. The NEW CA enters a Staging Period. This duration of the Staging Period is determined by the CA, but it MUST be no less than 24 hours. The Staging Period is intended to afford an opportunity for all RPs to download the NEW CA certificate, prior to publication of certificates, CRLs, and RPKI signed objects under the NEW CA. During the Staging Period, the NEW CA SHOULD re-issue, but not publish, all of the products that were issued under the CURRENT CA. This includes all CA certificates, EE certificates, and RPKI signed objects. Section 4 describes how each re-issued product relates to the product that it replaces. During the Staging Period, the CURRENT CA SHOULD continue to accept and process certificate issuance requests and MUST continue to accept and process certificate revocation requests. If any certificates are issued by the CURRENT CA during the staging period, they MUST be re-issued under the NEW CA during the period. Any certificates that are revoked under the CURRENT CA MUST NOT be re-issued under the NEW CA.
5. Upon expiration of the Staging Period, the NEW CA MUST publish the signed products that have been re-issued under the NEW CA, replacing the corresponding products issued under the CURRENT CA at the NEW CA's repository publication point. This replacement is implied by the file naming requirements imposed by [ID.ietf-sidr-repos-struct] for these signed products. The trivial manifest for the NEW CA (which contained only one entry, for the NEW CA's CRL) is replaced by a manifest listing all of these re-issued, signed products. At this point the CURRENT CA becomes the OLD CA, and the NEW CA becomes the CURRENT CA. Use the OLD CA to issue a manifest that lists

only the OLD CA's CRL. It is anticipated that this step is very brief, perhaps a few minutes in duration, because the CA has re-issued all of the signed products during the Staging Period. Nonetheless, it is desirable that the activities performed in this step be viewed as atomic by RPs.

6. Generate a certificate revocation request for the OLD CA certificate and submit it to the issuer of that certificate. When the OLD CA certificate is revoked, the CRL for the OLD CA is removed from the repository, along with the manifest for the OLD CA. The private key for the OLD CA is destroyed.

3. Relying Party Requirements

This procedure defines a Staging Period for CAs performing a key rollover operation. This period is defined as a period no shorter than 24 hours.

RPs who maintain a local cache of the distributed RPKI repository MUST perform a local cache synchronisation operation against the distributed RPKI repository at regular intervals of no longer than 24 hours.

4. Re-issuing Certificates and RPKI Signed Objects

This section provides rules a CA MUST use when it re-issues subordinate certificates and RPKI signed objects [ID.ietf-sidr-signed-object] as part of the key rollover process. Note that CRLs and manifests are not re-issued, per se. They are generated for each CA instance. A manifest catalogues the contents of a publication point relative to a CA instance. A CRL lists revoked certificates, relative to a CA instance. Key rollover processing for CRLs and manifests is described above, in Section 3.

4.1. CA Certificates

When a CA, as part of the key rollover process, re-issues a CA certificate, it copies all of the field and extension values from the old certificate into the new certificate. The only exceptions to this rule are that the notBefore value MAY be set to the current date and time, and the certificate serial number MAY change. Because the re-issued CA certificate is issued by a different CA instance, it is not a requirement that the certificate serial number change in the re-issued certificate. Nonetheless, the CA MUST ensure that each certificate issued under a specific CA instance (a distinct name and

key) contains a unique serial number.

4.2. RPKI Signed Objects

An RPKI signed object is a CMS signed-data object, containing an EE certificate and a payload (content) [ID.ietf-sidr-signed-object]. When a key rollover occurs, the EE certificate for the RPKI signed object MUST be re-issued, under the key of the NEW CA. A CA MAY choose to treat this EE certificate the same way that it deals with CA certificates, i.e., to copy over all fields and extensions, and MAY change only the notBefore date and the serial number. If the CA adopts this approach, then the new EE certificate is inserted into the CMS wrapper, but the signed context remains the same. (If the signing time or binary signing time values in the CMS wrapper are non-null, they MAY be updated to reflect the current time.) Alternatively, the CA MAY elect to generate a new key pair for this EE certificate. If it does so, the object content MUST be resigned under the private key corresponding to the EE certificate. In this case the EE certificate MUST contain a new public key and a new notBefore value, it MAY contain a new notAfter value, but all other field and extension values remain constant. If the signing time or binary signing time values in the CMS wrapper are non-null, they MAY be updated to reflect the current time.

5. Security Considerations

No key should be used forever. The longer a key is in use, the greater the probability that it will have been compromised through carelessness, accident, espionage, or cryptanalysis. Infrequent key rollover increases the risk that the rollover procedures will not be followed to the appropriate level of precision, increasing the risk of operational failure of some form in the key rollover process. Regular scheduling of key rollover is generally considered to be a part of a prudent key management practice. However, key rollover does impose additional operational burdens on both the CA and upon the population of RPs.

These considerations imply that in choosing lifetimes for the keys it manages, a CA should balance security and operational impact (on RPs). A CA should perform key rollover at regularly scheduled intervals. These intervals should be frequent enough to minimize the risks associated with key compromise (noted above) and to maintain local operational proficiency with respect to the key rollover process. However, key lifetimes should be sufficiently long so that the (system-wide) load associated with key rollover events (across the entire RPKI) does not impose an excessive burden upon the population of RPs. RPs are encouraged to maintain an accurate local

cache of the current state of the RPKI, which implies frequent queries to the RPKI repository system to detect changes. When a CA rekeys, it changes many signed objects, thus impacting all RPs.

6. IANA Considerations

[Note to IANA, to be removed prior to publication: there are no IANA considerations stated in this document.]

7. Acknowledgements

The authors would like to acknowledge the review comments of Tim Bruijnzeels and Sean Turner in preparing this document.

8. References

8.1. Normative References

- [ID.ietf-sidr-repos-struct]
Huston, G., Loomans, R., and G. Michaelson, "A Profile for Resource Certificate Repository Structure", Internet Draft draft-ietf-sidr-repos-struct-04.txt, May 2010.
- [ID.ietf-sidr-res-certs]
Huston, G., Michaelson, G., and R. Loomans, "A Profile for X.509 PKIX Resource Certificates", Internet Draft draft-ietf-sidr-res-certs-18.txt, May 2010.
- [RFC3779] Lynn, C., Kent, S., and K. Seo, "X.509 Extensions for IP Addresses and AS Identifiers", RFC 3779, June 2004.
- [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, May 2008.

8.2. Informative References

- [ID.ietf-sidr-arch]
Lepinski, M. and S. Kent, "An Infrastructure to Support Secure Internet Routing", draft-ietf-sidr-arch-11 (work in progress), September 2010.
- [ID.ietf-sidr-signed-object]
Lepinski, M., Chi, A., and S. Kent, "Signed Object

Template for the Resource Public Key Infrastructure",
draft-ietf-sidr-signed-object-01.txt (work in progress),
October 2010.

Authors' Addresses

Geoff Huston
Asia Pacific Network Information Centre

Email: gih@apnic.net
URI: <http://www.apnic.net>

George Michaelson

Email: ggm@apnic.net
URI: <http://www.apnic.net>

Stephen Kent
BBN Technologies
10 Moulton St.
Cambridge, MA 02138
USA

Email: kent@bbn.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: April 21, 2011

S. Weiler
A. Sonalker
SPARTA, Inc.
October 18, 2010

A Publication Protocol for the Resource Public Key Infrastructure (RPKI)
draft-ietf-sidr-publication-00

Abstract

This document defines a protocol for publishing Resource Public Key Infrastructure (RPKI) objects. Even though the RPKI will have many participants issuing certificates and creating other objects, it is operationally useful to consolidate the publication of those objects. This document provides the protocol for that.

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 <http://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 April 21, 2011.

Copyright Notice

Copyright (c) 2010 IETF Trust and the persons identified as the document authors. All rights reserved.

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

Table of Contents

1. Introduction	3
1.1. Terminology	3
2. Context	3
3. Protocol Specification	4
3.1. Common Details	4
3.1.1. Common XML Message Format	4
3.2. Control Protocol	5
3.2.1. Config Object	5
3.2.2. Client Object	5
3.3. Publication Protocol	6
3.4. Error handling	6
3.5. XML Schema	7
4. Operational Considerations	10
5. IANA Considerations	10
6. Security Considerations	10
7. References	10
7.1. Normative References	10
7.2. Informative References	11
Appendix A. Acknowledgments	11
Authors' Addresses	11

1. Introduction

This document assumes a working knowledge of the Resource Public Key Infrastructure (RPKI), which is intended to support improved routing security on the Internet. [I-D.ietf-sidr-arch]

In order to make participation in the RPKI easier, it is helpful to have a few consolidated repositories for RPKI objects, thus saving every participant from the cost of maintaining a new service. Similarly, clients using the RPKI objects will find it faster and more reliable to retrieve the necessary set from a smaller number of repositories.

These consolidated RPKI object repositories will in many cases be outside the administrative scope of the organization issuing a given RPKI object. Hence the need for a protocol to publish RPKI objects.

This document defines the RPKI publication protocol, including a sub-protocol for configuring the publication engine.

1.1. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

"Publication engine" and "publication server" are used interchangeably to refer to the server providing the service described in this document.

"Business Public Key Infrastructure" ("Business PKI" or "BPKI") refers to a PKI, separate from the RPKI, used to authenticate clients to the publication engine.

2. Context

This protocol was designed specifically for the case where an internet registry, already issuing RPKI certificates to its children, also wishes to run a publication service for its children.

We use the term "Business PKI" here to suggest that an internet registry might already have a PKI, separate from the RPKI, for authenticating its clients and might wish to reuse that PKI for this protocol. Such reuse is not a requirement.

3. Protocol Specification

In summary, the publication protocol uses XML messages wrapped in CMS, carried over HTTP transport.

The publication protocol consists of two separate subprotocols. The first is a control protocol used to configure a publication engine. The second subprotocol, which we refer to by the overloaded term "publication protocol", is used to request publication of specific objects. The publication engine operates a single HTTP server on a single port. It distinguishes between the two protocols by using different URLs for them.

3.1. Common Details

This section discusses details that the two subprotocols have in common, including the transport and CMS wrappers. This portion of the protocol is largely inherited from the provisioning protocol ([I-D.ietf-sidr-rescerts-provisioning]).

Both protocols use a simple request/response interaction. The client passes a request to the server, and the server generates a corresponding response. A message exchange commences with the client initiating an HTTP POST with content type of "application/x-rpki", with the message object as the body. The server's response will similarly be the body of the response with a content type of "application/x-rpki".

The content of the POST, and the server's response, will be a well-formed Cryptographic Message Syntax (CMS) [RFC5652] object with OID = 1.2.840.113549.1.7.2 as described in Section 3.1 of [I-D.ietf-sidr-rescerts-provisioning].

3.1.1. Common XML Message Format

The publication protocol uses the same message passing design as the provisioning protocol. The XML schema for this protocol (including both subprotocols) is below in Section 3.5. Both subprotocols use the same basic XML message format, which looks like:

```
<?xml version='1.0' encoding='us-ascii'?>
<msg xmlns="http://www.hactrn.net/uris/rpki/publication-spec/"
      version="1"
      type="message type">
  [one or more PDUs]
</msg>
```

version:

The value of this attribute is the version of this protocol.
This document describes version 1.

type:

The possible values of this attribute are "reply" and "query".

3.2. Control Protocol

The control protocol is used to configure a publication server. It can set global variables (at the moment, limited to a BPKI CRL) and manage clients who are allowed to publish data on the server.

The control protocol has two objects: the <config/> object, and the <client/> object.

3.2.1. Config Object

The <config/> object allows configuration of data that apply to the entire publication server rather than a particular client. There is exactly one <config/> object in the publication server, and it only supports the "set" and "get" actions -- it cannot be created or destroyed.

The <config/> object only has one data element that can be set: the `bpki_crl`. This is used by the publication server when authenticating clients.

3.2.2. Client Object

Unlike the <config/> object the <client/> object represents one client authorized to use the publication server.

The <client/> object supports five actions: "create", "set", "get", "list", and "destroy". Each client has a "client_handle" attribute, which is used in responses and must be specified in "create", "set", "get", or "destroy" actions.

Payload data which can be configured in a <client/> object include:

- o `base_uri` (attribute): This attribute represents the base URI below which the client will be allowed to publish data. Additional constraints may be imposed by the Publication Server in certain cases, for e.g., a child publishing directly under its parent.
- o `bpki_cert` (element): This represents the X509 BPKI CA certificate for this client. This should be used as part of the certificate chain when validating incoming TLS and CMS messages. Two valid approaches exist. If the optional `bpki_glue` certificate is being used, then the `bpki_cert` certificate should be issued by the

- bpki_glue certificate; otherwise, the bpki_cert certificate should be issued by the publication engine's bpki_ta certificate.
- o bpki_glue (element): This is an additional (optional) type of X509 certificate for this client. It may be used in certain pathological cross-certification cases which require a two-certificate chain due to issuer name conflicts. When being used, issuing order is that the bpki_glue certificate should be the issuer of the bpki_cert certificate. Otherwise, it should be issued by the publication engine's bpki_ta certificate. Since this is an optional use certificate, it may be left unset if not needed.

3.3. Publication Protocol

The publication protocol is structured differently from the control protocol in that objects in the publication protocol represent objects to be published or objects to be withdrawn from publication.

Each kind of object supports two actions: "publish" and "withdraw". In each case the XML element representing the object to be published or withdrawn has a "uri" attribute which contains the publication URI. For "publish" actions, the XML element body contains the DER object to be published, encoded in Base64; for "withdraw" actions, the XML element body is empty.

The publication protocol uses four types of objects:

- o Certificate Object: The <certificate/> object represents an RPKI certificate to be published or withdrawn.
- o CRL Object: The <crl/> object represents an RPKI CRL to be published or withdrawn.
- o Manifest Object: The <manifest/> object represents an RPKI publication manifest to be published or withdrawn. See [I-D.ietf-sidr-rpki-manifests] for more information on manifests.
- o ROA Object: The <roa/> object represents a ROA to be published or withdrawn. See [I-D.ietf-sidr-roa-format] for more information on ROAs.

Note that every publication or withdrawal action requires a new manifest, thus every publication or withdrawal action will involve at least two objects.

3.4. Error handling

Errors are handled similarly in both subprotocols, and they're handled at two levels.

Since all messages in this protocol are conveyed over HTTP connections, basic errors are indicated via the HTTP response code.

4xx and 5xx responses indicate that something bad happened. Errors that make it impossible to decode a query or encode a response are handled in this way.

Where possible, errors will result in an XML <report_error/> message which takes the place of the expected protocol response message. <report_error/> messages are CMS-signed XML messages like the rest of this protocol, and thus can be archived to provide an audit trail.

<report_error/> messages only appear in replies, never in queries. The <report_error/> message can appear in both the control and publication subprotocols.

The <report_error/> message includes an optional "tag" attribute to assist in matching the error with a particular query when using batching.

The error itself is conveyed in the error_code (attribute). The value of this attribute is a token indicating the specific error that occurred. [TODO: define these tokens]

The body of the <report_error/> element itself is an optional text string; if present, this is debugging information. At present this capability is not used, debugging information goes to syslog.

3.5. XML Schema

The following is a RelaxNG compact form schema describing the Publication Protocol.

```
default namespace = "http://www.hactrn.net/uris/rpki/publication-spec/"

# Top level PDU

start = element msg { attribute version { xsd:positiveInteger {
    maxInclusive="1" } }, ((attribute type { "query" }, query_elt*) |
    (attribute type { "reply" }, reply_elt*)) }

# PDUs allowed in a query
query_elt = ( config_query | client_query | certificate_query |
    crl_query | manifest_query | roa_query )

# PDUs allowed in a reply
reply_elt = ( config_reply | client_reply | certificate_reply |
    crl_reply | manifest_reply | roa_reply | report_error_reply )

# Tag attributes for bulk operations
```

```
tag = attribute tag { xsd:token {maxLength="1024" } }

# Base64 encoded DER stuff
base64 = xsd:base64Binary { maxLength="512000" }

# Publication URLs
uri_t = xsd:anyURI { maxLength="4096" }
uri = attribute uri { uri_t }

# Handles on remote objects (replaces passing raw SQL IDs). NB:
# Unlike the up-down protocol, handles in this protocol allow "/" as a
# hierarchy delimiter.
object_handle = xsd:string { maxLength="255" pattern="[\-_A-Za-z0-9/]*" }

# <config/> element (use restricted to repository operator)
# config_handle attribute, create, list, and destroy commands omitted
# deliberately, see code for details

config_payload = (element bpki_crl { base64 }?)

config_query |= element config { attribute action { "set" }, tag?,
    config_payload }
config_reply |= element config { attribute action { "set" }, tag?,
    config_payload }
config_query |= element config { attribute action { "get" }, tag?,
    config_payload }
config_reply |= element config { attribute action { "get" }, tag?,
    config_payload }

# <client/> element (use restricted to repository operator)

client_handle = attribute client_handle { object_handle }

client_payload = (attribute base_uri { uri_t }?, element bpki_cert {
    base64 }?, element bpki_glue { base64 }?)

client_query |= element client { attribute action { "create" }, tag?,
    client_handle, client_payload }
client_reply |= element client { attribute action { "create" }, tag?,
    client_handle }
client_query |= element client { attribute action { "set" }, tag?,
    client_handle, client_payload }
client_reply |= element client { attribute action { "set" }, tag?,
    client_handle }
client_query |= element client { attribute action { "get" }, tag?,
    client_handle }
client_reply |= element client { attribute action { "get" }, tag?,
    client_handle, client_payload }
client_query |= element client { attribute action { "list" }, tag?,
    client_payload }
client_reply |= element client { attribute action { "list" }, tag?,
    client_payload }
```

```
    client_handle, client_payload }
client_query |= element client { attribute action { "destroy" }, tag?,
    client_handle }
client_reply |= element client { attribute action { "destroy" }, tag?,
    client_handle }

# <certificate/> element

certificate_query |= element certificate { attribute action {
    "publish" }, tag?, uri, base64 }

certificate_reply |= element certificate { attribute action {
    "publish" }, tag?, uri }

certificate_query |= element certificate { attribute action {
    "withdraw" }, tag?, uri }

certificate_reply |= element certificate { attribute action {
    "withdraw" }, tag?, uri }

# <crl/> element

crl_query |= element crl { attribute action { "publish" }, tag?, uri,
    base64 }
crl_reply |= element crl { attribute action { "publish" }, tag?, uri }
crl_query |= element crl { attribute action { "withdraw" }, tag?, uri }
crl_reply |= element crl { attribute action { "withdraw" }, tag?, uri }

# <manifest/> element

manifest_query |= element manifest { attribute action { "publish" },
    tag?, uri, base64 }
manifest_reply |= element manifest { attribute action { "publish" },
    tag?, uri }
manifest_query |= element manifest { attribute action { "withdraw" },
    tag?, uri }
manifest_reply |= element manifest { attribute action { "withdraw" },
    tag?, uri }

# <roa/> element

roa_query |= element roa { attribute action { "publish" }, tag?, uri,
    base64 }
roa_reply |= element roa { attribute action { "publish" }, tag?, uri }
roa_query |= element roa { attribute action { "withdraw" }, tag?, uri }
roa_reply |= element roa { attribute action { "withdraw" }, tag?, uri }

# <report_error/> element
```

```
error = xsd:token { maxLength="1024" }

report_error_reply = element report_error {
  tag?,
  attribute error_code { error },
  xsd:string { maxLength="512000" }?
}
```

4. Operational Considerations

Placeholder section to talk about nesting children under parents in the sameso repository, to allow for a single rsync to fetch both (observing that the rsync setup times tends to dominate over the sync time). And, more distressingly, talk about the access control impacts of that nesting.

5. IANA Considerations

This document specifies no IANA Actions.

6. Security Considerations

7. References

7.1. Normative References

- [I-D.ietf-sidr-res-certs]
Huston, G., Michaelson, G., and R. Loomans, "A Profile for X.509 PKIX Resource Certificates",
draft-ietf-sidr-res-certs-19 (work in progress),
October 2010.
- [I-D.ietf-sidr-rescerts-provisioning]
Huston, G., Loomans, R., Ellacott, B., and R. Austein, "A Protocol for Provisioning Resource Certificates",
draft-ietf-sidr-rescerts-provisioning-07 (work in progress), October 2010.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC2818] Rescorla, E., "HTTP Over TLS", RFC 2818, May 2000.
- [RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security

(TLS) Protocol Version 1.2", RFC 5246, August 2008.

- [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, May 2008.
- [RFC5652] Housley, R., "Cryptographic Message Syntax (CMS)", STD 70, RFC 5652, September 2009.
- [X.690] Postel, J., "ITU-T Recommendation X.690: ISO/IEC 8825-1:2002, Information technology - ASN.1 encoding rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER)", 2002.

7.2. Informative References

- [I-D.ietf-sidr-arch]
Lepinski, M. and S. Kent, "An Infrastructure to Support Secure Internet Routing", draft-ietf-sidr-arch-11 (work in progress), September 2010.
- [I-D.ietf-sidr-roa-format]
Lepinski, M., Kent, S., and D. Kong, "A Profile for Route Origin Authorizations (ROAs)", draft-ietf-sidr-roa-format-07 (work in progress), July 2010.
- [I-D.ietf-sidr-rpki-manifests]
Austein, R., Huston, G., Kent, S., and M. Lepinski, "Manifests for the Resource Public Key Infrastructure", draft-ietf-sidr-rpki-manifests-08 (work in progress), October 2010.

Appendix A. Acknowledgments

We acknowledge the editors of [I-D.ietf-sidr-rescerts-provisioning] (Geoff Huston, Robert Loomans, Byron Ellacott, and Rob Austein), from whom we took some of the text for this document.

We especially thank Rob Austein, who implemented the publication protocol and helped us to understand it.

Authors' Addresses

Samuel Weiler
SPARTA, Inc.
7110 Samuel Morse Drive
Columbia, Maryland 21046
US

Email: weiler@sparta.com

Anuja Sonalker
SPARTA, Inc.
7110 Samuel Morse Drive
Columbia, Maryland 21046
US

Email: Anuja.Sonalker@sparta.com

Secure Inter-Domain Routing
Internet-Draft
Intended status: BCP
Expires: May 12, 2011

G. Huston
R. Loomans
G. Michaelson
APNIC
November 8, 2010

A Profile for Resource Certificate Repository Structure
draft-ietf-sidr-repos-struct-06.txt

Abstract

This document defines a profile for the structure of the Resource PKI distributed repository. Each individual repository publication point is a directory that contains files that correspond to X.509 / PKIX Resource Certificates, Certificate Revocation Lists and signed objects. This profile defines the recommended object (file) naming scheme, the recommended contents of repository publication points (directories), and a suggested internal structure of a local repository cache that is intended to facilitate synchronisation across a distributed collection of repository publication points and facilitate certification path construction.

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 <http://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 May 12, 2011.

Copyright Notice

Copyright (c) 2010 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of

publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	3
1.1. Terminology	3
2. RPKI Repository Publication Point Content and Structure . . .	4
2.1. Manifests	6
2.2. CA Repository Publication Points	6
3. Resource Certificate Publication Repository Considerations . .	8
4. Certificate Re-issuance and Repositories	9
5. Synchronising Repositories with a Local Cache	10
6. Security Considerations	11
7. IANA Considerations	11
8. References	11
8.1. Normative References	11
8.2. Informative References	12
Authors' Addresses	12

1. Introduction

To validate attestations made in the context of the Resource Public Key Infrastructure (RPKI) [I-D.ietf-sidr-arch], relying parties (RPs) need access to all the X.509 / PKIX Resource Certificates, Certificate Revocation Lists (CRLs), and signed objects that collectively define the RPKI.

Each issuer of a certificate, CRL or a signed object makes it available for download to RPs through the publication of the object in an RPKI repository.

The repository system is collection of all signed objects that MUST be globally accessible to all RPs. When certificates, CRLs and signed objects are created, they are uploaded to a repository publication point, from whence they can be downloaded for use by RPs.

This profile defines the recommended object (file) naming scheme, the recommended contents of repository publication points (directories), and a suggested internal structure of a local repository cache that is intended to facilitate synchronisation across a distributed collection of repository publication points and facilitate certification path construction.

A Resource Certificate attests to a binding of an entity's public key to a set of IP address blocks and AS numbers. The Subject of a Resource Certificate can demonstrate that it is the holder of the resources enumerate in the certificate by using its private key to generate a digital signature (that can be verified using the public key from the certificate).

1.1. Terminology

It is assumed that the reader is familiar with the terms and concepts described in "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile" [RFC5280], and "X.509 Extensions for IP Addresses and AS Identifiers" [RFC3779].

In addition, the following terms are used in this document:

Repository Object (or Object):

This refers to a terminal object in a repository publication point. A terminal object is conventionally implemented as a file in a publicly accessible directory, where the file is not a directory itself, although other forms of objects that have an analogous public appearance to a file are encompassed by this term.

Repository Publication Point:

This refers to a collection of Repository Objects that are published at a common publication point. This is conventionally implemented as a directory in a publicly accessible filesystem that is identified by a URI [RFC3986], although other forms of local storage that have an analogous public appearance to a simple directory of files are also encompassed by this term.

Repository Instance:

This refers to a collection of one or more Repository Publication Points that share a common publication instance. This conventionally is implemented as a collection of filesystem directories that share a common URI prefix, where each directory is also identifiable by its own unique URI.

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119.

2. RPKI Repository Publication Point Content and Structure

The RPKI does not require that a single repository instance contain all published RPKI objects. Instead, the RPKI repository system is comprised of multiple repository instances. Each individual repository instance is composed of one or more repository publication points. Each repository publication point is used by one or more entities referenced in RPKI certificates, as defined in the certificate's Subject Information Authority (SIA) extension.

This section describes the collection of objects (RPKI certificates, CRLs, manifests and signed objects) held in repository publication points.

For every Certification Authority (CA) certificate in the RPKI there is a corresponding repository publication point that is the authoritative publication point for all current certificates and CRLs issued by this CA. The certificate's SIA extension contains a URI [RFC3986] that references this repository publication point and identifies the repository access mechanisms. Additionally, a certificate's Authority Information Access (AIA) extension contains a URI that references the authoritative location for the Certification Authority (CA) certificate under which the given certificate was issued.

For example, if the subject of certificate A has issued certificates B and C, then the AIA extensions of certificates B and C both point to the publication point for the certificate A object, and the SIA

extension of certificate A points to a repository publication point (directory) containing certificates B and C (see Figure 1).

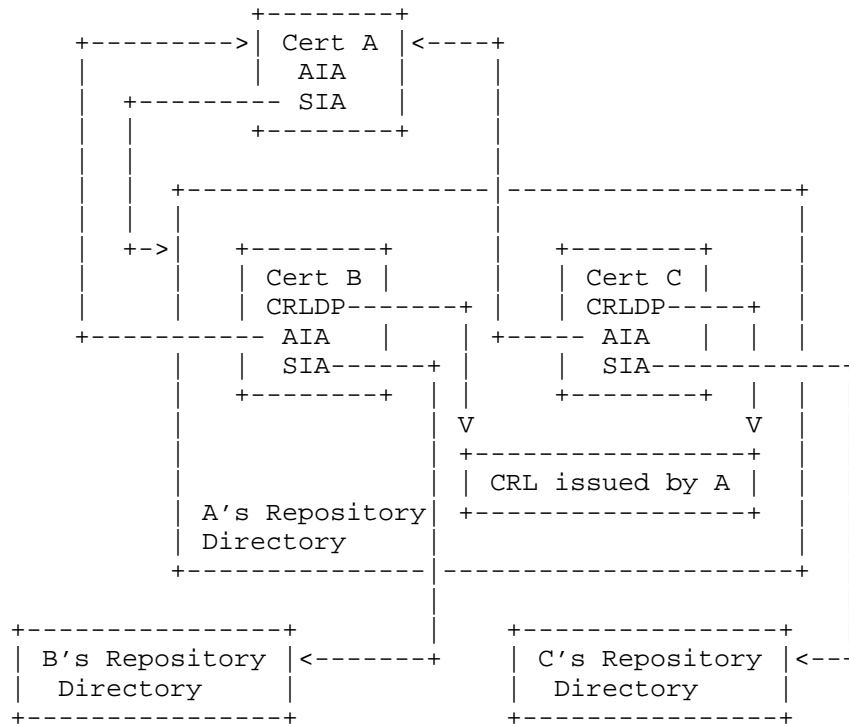


Figure 1. Use of AIA and SIA extensions in the RPKI.

In Figure 1, certificates B and C are issued by (CA) A. Therefore, the AIA extensions of certificates B and C point to (certificate) A, and the SIA extension of certificate A points to the repository publication point of CA A's subordinate products, which includes certificates B and C, as well as the CRL issued by A. The CRL Distribution Points (CRLDP) extension in certificates B and C both point to the Certificate Revocation List (CRL) issued by A.

In this distributed repository structure an instance of a CA's repository publication point contains all published certificates issued by that CA, and the CRL issued by that CA. This repository also contains all published digitally signed objects that are verified by an EE certificate issued by this CA.

2.1. Manifests

Every repository publication point MUST contain a manifest [I-D.ietf-sidr-rpki-manifests]. The manifest contains a list of the names of all objects, as well as the hash value of each object's contents, that are currently published by a CA, or by an EE.

An authority MAY perform a number of object operations on a publication repository within the scope of a repository change before issuing a single manifest that covers all the operations within the scope of this change. Repository operators SHOULD implement some form of directory management regime function on the repository to ensure that RPs who are performing retrieval operations on the repository are not exposed to intermediate states during changes to the repository and the associated manifest.

2.2. CA Repository Publication Points

A CA Certificate has two accessMethod elements specified in its SIA field. The id-ad-caRepository accessMethod element has an associated accessLocation element that points to the repository publication point of the certificates issued by this CA, as specified in [I-D.ietf-sidr-res-certs]. The id-ad-rpkiManifest accessMethod element has an associated accessLocation element that points to the manifest object, as an object URI (as distinct to a directory URI), that is associated with this CA.

A CA's publication repository contains the current (non-expired and non-revoked) certificates issued by this CA, the most recent CRL issued by this CA, the current manifest, and all other current signed objects that can be verified using an EE certificate [I-D.ietf-sidr-res-certs] issued by this CA.

The CA's manifest contains the names of this collection of objects, together with the hash value of each object's contents, with the single exception of the manifest itself.

The RPKI design requires that a CA be uniquely associated with a single key pair. Thus, the administrative entity that is a CA performs key rollover by generating a new CA certificate with a new Subject name, as well as a new key pair [I-D.ietf-sidr-keyroll]. (The reason for the new Subject name is that in the context of the RPKI the Subject names in all certificates issued by a CA are intended to be unique, and because the RPKI key rollover procedure creates a new instance of a CA with the new key, the name constraint implies the need for a new Subject name for the CA with the new key.) In such cases the entity SHOULD continue to use the same repository publication point for both CA instances during the key rollover,

ensuring that the value of the AIA extension in indirect subordinate objects that refer to the certificates issued by this CA remain valid across the key rollover, and that the re-issuance of subordinate certificates in a key rollover is limited to the collection of immediate subordinate products of this CA. In such cases the repository publication point will contain the CRL, manifest and subordinate certificates of both CA instances.

The following paragraphs provide guidelines for naming objects in a CA's repository publication point:

CRL:

When a CA issues a new CRL, it replaces the previous CRL (issued under the same CA key pair) in the repository publication point. CAs MUST NOT continue to publish previous CRLs in the repository publication point. Thus, it SHOULD replace (overwrite) previous CRLs signed by the same CA (instance). A non-normative guideline for naming such objects is that the file name chosen for the CRL in the repository be a value derived from the public key of the CA. One such method of generating a CRL publication name is described in section 2.1 of [RFC4387]; convert the 160-bit hash of a CA's public key value into a 27-character string using a modified form of Base64 encoding, with an additional modification as proposed in section 5, table 2, of [RFC4648]. The filename extension of ".crl" MUST be used, to denote the file as a CRL.

Manifest:

When a new instance of a manifest is published, it SHOULD replace the previous manifest, to avoid confusion. CAs MUST NOT continue to publish previous CA manifests in the repository publication point. A non-normative guideline for naming such objects is that the filename chosen for the manifest in the publication repository be a value derived from the public key part of the entity's key pair, using the algorithm described for CRLs above for generation of filenames. The filename extension of ".mft" MUST be used, to denote the object as a manifest.

Certificates:

Within the RPKI framework it is possible that a CA MAY issue a series of certificates to the same subject name, the same subject public key, and the same resource collection. However, a relying party requires access only to the most recently published certificate in such a series. Thus, the such a series of certificates SHOULD share the same filename. This ensures that each successive issued certificate in such a series effectively overwrites the previous instance of the certificate. A non-normative guideline for naming such objects is for the CA to adopt a (local) policy requiring a subject to use a unique key pair for

each unique instance of a certificate series issued to the same subject, thereby the CA to use a file name generation scheme based on the subject's public key, e.g., using the algorithm described above for CRLs above. Published certificates MUST use a filename extension of ".cer" to denote the object as a certificate.

Signed Objects:

Signed objects are published in the repository publication point referenced by the SIA of the CA certificate that issued the EE certificate used to validate the digital signature of the signed object (and are directly referenced by the SIA of that EE certificate). A non-normative guideline for naming such signed objects is for the filename of such objects to be derived from the associated EE certificate's public key, applying the algorithm described above. Published objects MUST NOT use the filename extensions ".crl", ".mft", or ".cer".

3. Resource Certificate Publication Repository Considerations

Each issuer MAY publish its issued certificates and CRL in any repository. However, there are a number of considerations that guide the choice of a suitable repository publication structure:

- * The publication repository SHOULD be hosted on a highly available service and high capacity publication platform.
- * The publication repository MUST be available using RSYNC [RFC5781]. Support of additional retrieval mechanisms is the choice of the repository operator. The supported retrieval mechanisms MUST be consistent with the accessMethod element value(s) specified in the SIA of the associated CA or EE.
- * Each CA repository publication point SHOULD contain the products of this CA, including those objects that can be verified by EE certificates that have been issued by this CA. The signed products of related CA's that are operated by the same entity MAY share this CA repository publication point. Aside from subdirectories, any other objects SHOULD NOT be placed in a repository publication point.

Any such subdirectory SHOULD be the repository publication point of a CA or EE certificate that is contained in the CA directory. These considerations also apply recursively to subdirectories of these directories.

- * Signed Objects are published in the location indicated by the SIA field of the EE certificate used to verify the signature of each object. Signed objects are published in the repository publication point of the CA certificate that issued the EE certificate. The SIA extension of the EE certificate references this object rather than the repository publication directory[I-D.ietf-sidr-res-certs].
- * It is recommended in Section 2.1 that repository operators SHOULD implement some form of directory management regime function on the repository to ensure that RPs who are performing retrieval operations on the repository are not exposed to intermediate states during changes to the repository and the associated manifest. Notwithstanding the following commentary, RPs SHOULD NOT assume that a consistent repository and manifest state is assured, and organise their retrieval operations accordingly (see Section 5).

The manner in which a repository operator can implement a directory update regime that mitigates the risk of the manifest and directory contents being inconsistent, to some extent, is dependent on the operational characteristics of the filesystem that hosts the repository, so the following comments are non-normative in terms of any implicit guidelines for repository operators.

A commonly used technique to avoid exposure to inconsistent retrieval states during updates to a large directory, is to batch a set of changes to be made, create a working copy of the directory's contents, and then perform the batch of changes to this local copy of the directory. On completion, rename the filesystem symbolic link of the repository directory name to point to this working copy of the directory. The old repository directory contents can be purged at a slightly later time. However, it is noted that the outcomes of this technique in terms of ensuring the integrity of client synchronization functions performed over the directory depend on the interaction between the supported access mechanisms and the local filesystem behaviour. It is probable that this technique will not remove all possibilities for RPs to see inconsistent states between the manifest and the repository.

4. Certificate Re-issuance and Repositories

If a CA certificate is re-issued, e.g., due to changes in the set of resources contained in the number resource extensions, it should not be necessary to re-issue all certificates issued under it. Because

these certificates contains AIA extensions that point to the publication point for the CA certificate, a CA SHOULD use a name for its repository publication point that persists across certificate re-issuance events. That is, re-issued CA certificates SHOULD use the same repository publication point as previously issued CA certificates having the same subject and subject public key, such that certificate re-issuance SHOULD intentionally overwrite the previously issued certificate within the repository publication point.

It is noted in section Section 2.2 that when a CA performs a key rollover the entity SHOULD use a name for its repository publication point that persists across key rollover. In such cases the repository publication point will contain the CRLs, and manifests of both CA instances as a transient state in the key rollover procedure. The RPKI key rollover procedure [I-D.ietf-sidr-keyroll] requires that the subordinate products of the old CA are overwritten in the common repository publication point by subordinate products issued by the new CA.

5. Synchronising Repositories with a Local Cache

It is possible to perform the validation-related task of certificate path construction using retrieval of individual certificates and certificate revocation lists using online retrieval of individual certificates, sets of candidate certificates and certificate revocation lists based on the AIA, SIA and CRLDP certificate fields. This is NOT recommended in circumstances where speed and efficiency are relevant considerations.

To enable efficient validation of RPKI certificates, CRLs, and signed objects, it is recommended that each relying party maintain a local repository containing a synchronized copy of all valid certificates, current certificate revocation lists, and all related signed objects.

The general approach to repository synchronisation is one of a "top-down" walk of the distributed repository structure. This commences with the collection of locally selected trust anchor material corresponding to the local choice of Trust Anchors, which can be used to load the initial set of self-signed resource certificate(s) that form the "seed" of this process [I-D.ietf-sidr-ta]. The process then populates the local repository cache with all valid certificates that have been issued by these issuers. This procedure can be recursively applied to each of these subordinate certificates. Such a repository traversal process SHOULD support a locally configured maximal chain length from the initial trust anchors to the current working validation point in order to ensure that the process does not follow

a loop or a non-terminating certificate chain.

RPs SHOULD ensure that this local synchronisation uses the retrieved manifests [I-D.ietf-sidr-rpki-manifests] to ensure that they are synchronising against a current consistent state of each repository publication point. It is noted in Section 3 that a repository operator cannot assure RPs that when the repository publication point contents are updated that the manifest contents and the repository contents will be precisely aligned at all times. RPs SHOULD use a retrieval algorithm that takes this potential for transient inconsistency into account. Possible algorithms for the RP to mitigate this situation include performing the synchronisation across the repository twice in succession, or performing a manifest retrieval both before and after the synchronisation of the directory contents, and repeating the synchronization function if the second copy of the manifest differs from the first.

6. Security Considerations

Repositories are not assumed to be integrity-protected databases, and repository retrieval operations MAY be vulnerable to various forms of "man-in-the-middle" attacks. Corruption of retrieved objects is detectable by a relying party through the validation of the signature associated with each retrieved object. Replacement of newer instances of an object with an older instance of the same object is detectable through the use of manifests. Insertion of revoked, deleted certificates is detected through the retrieval and processing of CRLs at scheduled intervals. However, even the use of manifests and CRLs will not allow a relying party to detect all forms of substitution attacks based on older (but not expired) valid objects.

7. IANA Considerations

[There are no IANA considerations in this document.]

8. References

8.1. Normative References

[I-D.ietf-sidr-arch]
Lepinski, M. and S. Kent, "An Infrastructure to Support Secure Internet Routing", draft-ietf-sidr-arch-11.txt (work in progress), September 2010.

[I-D.ietf-sidr-res-certs]

Huston, G., Michaelson, G., and R. Loomans, "A Profile for X.509 PKIX Resource Certificates", draft-ietf-sidr-res-certs-18.txt (work in progress), May 2010.

[I-D.ietf-sidr-rpki-manifests]

Austein, R., Huston, G., Kent, S., and M. Lepinski, "Manifests for the Resource Public Key Infrastructure", draft-ietf-sidr-rpki-manifests (work in progress), May 2010.

8.2. Informative References

[I-D.ietf-sidr-keyroll]

Huston, G., Michaelson, G., and S. Kent, "CA Key Rollover in the RPKI", draft-ietf-sidr-keyroll-02.txt (work in progress), October 2010.

[I-D.ietf-sidr-ta]

Michaelson, G., Kent, S., and G. Huston, "A Profile for Trust Anchor Material for the Resource Certificate PKI", draft-ietf-sidr-ta-04.txt (work in progress), May 2010.

[RFC3779] Lynn, C., Kent, S., and K. Seo, "X.509 Extensions for IP Addresses and AS Identifiers", RFC 3779, June 2004.

[RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, January 2005.

[RFC4387] Gutmann, P., "Internet X.509 Public Key Infrastructure Operational Protocols: Certificate Store Access via HTTP", RFC 4387, February 2006.

[RFC4648] Josefsson, S., "The Base16, Base32, and Base64 Data Encodings", RFC 4648, October 2006.

[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, May 2008.

[RFC5781] Weiler, S., Ward, D., and R. Housley, "The rsync URI Scheme", RFC 5781, February 2010.

Authors' Addresses

Geoff Huston
APNIC

Email: gih@apnic.net
URI: <http://www.apnic.net>

Robert Loomans
APNIC

Email: robertl@apnic.net
URI: <http://www.apnic.net>

George Michaelson
APNIC

Email: ggm@apnic.net
URI: <http://www.apnic.net>

SIDR
Internet-Draft
Intended status: Standards Track
Expires: May 12, 2011

G. Huston
G. Michaelson
R. Loomans
APNIC
November 8, 2010

A Profile for X.509 PKIX Resource Certificates
draft-ietf-sidr-res-certs-20

Abstract

This document defines a standard profile for X.509 certificates for the purposes of supporting validation of assertions of "right-of-use" of Resources (INRs). The certificates issued under this profile are used to convey the Issuer's authorisation of the Subject to be regarded as the current holder of a "right-of-use" of the INRs that are described in the certificate. This document contains the normative specification of Certificate and Certificate Revocation List (CRL) syntax in the Resource Public Key Infrastructure (RPKI). The document also specifies profiles for the format of certificate requests. The document also specifies the Relying Party RPKI certificate path validation procedure.

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 <http://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 May 12, 2011.

Copyright Notice

Copyright (c) 2010 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents

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

Table of Contents

1. Introduction	4
1.1. Terminology	5
2. Describing Resources in Certificates	5
3. End-Entity (EE) Certificates and Signing Functions in the RPKI	5
4. Resource Certificates	6
4.1. Version	6
4.2. Serial number	6
4.3. Signature Algorithm	6
4.4. Issuer	7
4.5. Subject	7
4.6. Valid From	7
4.7. Valid To	8
4.8. Subject Public Key Info	8
4.9. Resource Certificate Extensions	8
4.9.1. Basic Constraints	8
4.9.2. Subject Key Identifier	8
4.9.3. Authority Key Identifier	9
4.9.4. Key Usage	9
4.9.5. Extended Key Usage	9
4.9.6. CRL Distribution Points	9
4.9.7. Authority Information Access	10
4.9.8. Subject Information Access	11
4.9.9. Certificate Policies	12
4.9.10. IP Resources	12
4.9.11. AS Resources	12
5. Resource Certificate Revocation Lists	13
6. Resource Certificate Requests	13
6.1. PKCS#10 Profile	13
6.1.1. PKCS#10 Resource Certificate Request Template Fields	14
6.2. CRMF Profile	14
6.2.1. CRMF Resource Certificate Request Template Fields	15
6.2.2. Resource Certificate Request Control Fields	16
6.3. Certificate Extension Attributes in Certificate Requests	16
7. Resource Certificate Validation	17

7.1. Resource Extension Validation	17
7.2. Resource Certification Path Validation	18
8. Design Notes	19
9. Security Considerations	22
10. IANA Considerations	22
11. Acknowledgements	23
12. References	23
12.1. Normative References	23
12.2. Informative References	24
Appendix A. Example Resource Certificate	24
Appendix B. Example Certificate Revocation List	27
Authors' Addresses	28

1. Introduction

This document defines a standard profile for X.509 certificates [X.509] for use in the context of certification of Internet Number Resources (INRs), i.e., IP Addresses and Autonomous System (AS) Numbers. Such certificates are termed "Resource Certificates". A Resource Certificate is a certificate that conforms to the PKIX profile [RFC5280], and that conforms to the constraints specified in this profile. A Resource Certificate attests that the Issuer has granted the Subject a "right-of-use" for a listed set of IP addresses and/or Autonomous System numbers.

This document is referenced by Section 7 of the Certificate Policy (CP) for the Resource Public Key Infrastructure (RPKI) [ID.sidr-cp]. It is an integral part of that policy and the normative specification for certificate and Certificate Revocation List (CRL) syntax used in the RPKI. The document also specifies profiles for the format of certificate requests, and the Relying Party (RP) RPKI certificate path validation procedure.

Resource Certificates are to be used in a manner that is consistent with the RPKI Certificate Policy [ID.sidr-cp]. They are issued by entities that assign and/or allocate public INRs, and thus the RPKI is aligned with the public INR distribution function. When an INR is allocated or assigned by a number registry to an entity, this allocation can be described by an associated Resource Certificate. This certificate is issued by the number registry, and it binds the certificate subject's key to the INRs enumerated in the certificate. One or two critical extensions, the IP Address Delegation or AS Identifier Delegation Extensions [RFC3779], enumerate the INRs that were allocated or assigned by the Issuer to the Subject.

RP validation of a Resource Certificate is performed in the manner specified in Section 7.1. This validation procedure differs from that described in section 6 of [RFC5280], such that:

- o additional validation processing imposed by the INR extensions is required,
- o a conformation of a public key match between the CRL issuer and the Resource Certificate issuer is required, and
- o the Resource Certificate is required to conform to this profile.

This profile defines those fields that are used in a Resource Certificate that MUST be present for the certificate to be valid. Any extensions not explicitly mentioned MUST be absent. The same applies to the CRLs used in the RPKI, that are also profiled in this document. A CA conforming to the RPKI CP MUST issue certificates and CRLs consistent with this profile.

1.1. Terminology

It is assumed that the reader is familiar with the terms and concepts described in "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile" [RFC5280], and "X.509 Extensions for IP Addresses and AS Identifiers" [RFC3779].

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119.

2. Describing Resources in Certificates

The framework for describing an association between the Subject of a certificate and the INRs currently under the Subject's control is described in [RFC3779]. This profile further requires that:

- o Every Resource Certificate MUST contain either the IP Address Delegation or the Autonomous System Identifier Delegation extension, or both.
- o These extensions MUST be marked as CRITICAL.
- o The sorted canonical format describing INRs, with maximal spanning ranges and maximal spanning prefix masks, as defined in [RFC3779], MUST be used for the resource extension field, except where the "inherit" construct is used instead.

When validating a Resource Certificate, an RP MUST verify that the INRs described in the Issuer's Resource Certificate encompass the INRs of the Resource Certificate being validated. In this context "encompass" allows for the Issuer's INRs to be the same as, or a strict superset of the Subject's INRs.

3. End-Entity (EE) Certificates and Signing Functions in the RPKI

As noted in [ID.sidr-arch], the primary function of End-Entity (EE) certificates in the RPKI is the verification of signed objects that relate to the usage of the INRs described in the certificate, e.g., Route Origin Authorizations (ROAs) and manifests.

The private key associated with an EE certificate is used to sign a single RPKI signed object, i.e., the EE certificate is used to validate only one object. The EE certificate is embedded in the object as part of a Cryptographic Message Syntax (CMS) signed data structure [ID.sidr-signed-object]. Because of the one-to-one

relationship between the EE certificate and the signed object, revocation of the certificate effectively revokes the corresponding signed object.

A EE certificate may be used to validate a sequence of signed objects, where each signed object in the sequence overwrites the previous instance of the signed object in the repository publication point, such that only one instance of the signed object is published at any point in time (e.g., an EE certificate MAY be used to sign a sequence of manifests [ID.sidr-rpki-manifests]). Such EE certificates are termed "sequential use" EE certificates.

EE certificates used to validate only one instance of a signed object, and are not used thereafter, or in any other validation context, are termed "one-time-use" EE certificates.

4. Resource Certificates

A Resource Certificate is a valid X.509 public key certificate, consistent with the PKIX profile [RFC5280], containing the fields listed in this section. Only the differences from [RFC5280] are noted below.

Unless specifically noted as being OPTIONAL, all the fields listed here MUST be present, and any other field MUST NOT appear in a conforming Resource Certificate. Where a field value is specified here, this value MUST be used in conforming Resource Certificates.

4.1. Version

As Resource Certificates are X.509 Version 3 certificates, the version MUST be 3 (i.e. the value of this field is 2).

RPs need not process version 1 or version 2 certificates (in contrast to [RFC5280]).

4.2. Serial number

The serial number value is a positive integer that is unique for each certificate issued by a given CA.

4.3. Signature Algorithm

The algorithm used in this profile is specified in [ID.sidr-rpki-algs].

4.4. Issuer

The value of this field is a valid X.501 distinguished name.

An Issuer name **MUST** contain one instance of the Common Name attribute and **MAY** contain one instance of the Serial Number attribute. If both attributes are present, it is **RECOMMENDED** that they appear as a set. The Common Name attribute **MUST** be encoded as a printable string. Issuer names are not intended to be descriptive of the identity of Issuer.

The RPKI does not rely on Issuer names being globally unique, for reasons of security. However, it is **RECOMMENDED** that Issuer names be generated in a fashion that minimizes the likelihood of collisions. See Section 8 for (non-normative) suggested name generation mechanisms that fulfill this recommendation.

4.5. Subject

The value of this field is a valid X.501 distinguished name, and is subject to the same constraints as the Issuer name.

In the RPKI the Subject name is determined by the Issuer, not proposed by the subject [ID.sidr-repos-struct]. Each distinct subordinate CA and EE certified by the Issuer **MUST** be identified using a Subject name that is unique per Issuer. In this context "distinct" is defined as an entity and a given public key. An Issuer **SHOULD** use a different Subject name if the Subject's key pair has changed (i.e., when the CA issues a certificate as part of re-keying the Subject.) Subject names are not intended to be descriptive of the identity of Subject.

4.6. Valid From

The "Valid From" time **SHOULD** be no earlier than the time of certificate generation.

In the RPKI it is valid for a certificate to have a value for this field that pre-dates the same field value in any superior certificate. Relying Parties **SHOULD NOT** attempt to infer from this time information that a certificate was valid at a time in the past, or will be valid at a time in the future, as the scope of an RP's test of validity of a certificate refers specifically to validity at the current time.

4.7. Valid To

The Valid To time represents the anticipated lifetime of the current resource allocation or assignment arrangement between the Issuer and the Subject.

It is valid for a certificate to have a value for this field that post-dates the same field value in any superior certificate. The same caveats apply to RP's assumptions relating to the certificate's validity at any time other than the current time.

While a CA is typically advised against issuing a certificate with a validity interval that exceeds the validity interval of the CA's certificate that will be used to validate the issued certificate, in the context of this profile, a CA MAY have valid grounds to issue a certificate with a validity interval that exceeds the validity interval of its certificate.

4.8. Subject Public Key Info

The algorithm used in this profile is specified in [ID.sidr-rpki-algs].

4.9. Resource Certificate Extensions

The following X.509 V3 extensions MUST be present in a conforming Resource Certificate, except where explicitly noted otherwise. Each extension in a resource certificate is designated as either critical or non-critical. A certificate-using system MUST reject the certificate if it encounters a critical extension it does not recognise; however, a non-critical extension MAY be ignored if it is not recognised [RFC5280].

4.9.1. Basic Constraints

The Basic Constraints extension field is a critical extension in the Resource Certificate profile, and MUST be present when the Subject is a CA, and MUST NOT be present otherwise.

The Issuer determines whether the "cA" boolean is set.

The Path Length Constraint is not specified for RPKI certificates, and MUST NOT be present.

4.9.2. Subject Key Identifier

This extension MUST appear in all Resource Certificates. This extension is non-critical.

The Key Identifier used for resource certificates is the 160-bit SHA-1 hash of the value of the DER-encoded ASN.1 bit string of the Subject Public Key, as described in Section 4.2.1.2 of [RFC5280].

4.9.3. Authority Key Identifier

This extension MUST appear in all Resource Certificates, with the exception of a CA who issues a "self-signed" certificate. In a self-signed certificate, a CA MAY include this extension, and set it equal to the Subject Key Identifier. The authorityCertIssuer and authorityCertSerialNumber fields MUST NOT be present. This extension is non-critical.

The Key Identifier used for resource certificates is the 160-bit SHA-1 hash of the value of the DER-encoded ASN.1 bit string of the Issuer's public key, as described in Section 4.2.1.1 of [RFC5280].

4.9.4. Key Usage

This extension is a critical extension and MUST be present.

In certificates issued to Certification Authorities only the keyCertSign and CRLSign bits are set to TRUE, and these MUST be the only bits set to TRUE.

In EE certificates the digitalSignature bit MUST be set to TRUE and MUST be the only bit set to TRUE.

4.9.5. Extended Key Usage

The Extended Key Usage (EKU) extension MUST NOT appear in any CA certificate in the RPKI. This extension also MUST NOT appear in EE certificates used to verify RPKI objects (e.g., ROAs or manifests). The extension MUST NOT be marked critical.

The EKU extension MAY appear in EE certificates issued to routers or other devices. Permitted values for the EKU OIDs will be specified in Standards Track RFCs issued by other IETF working groups that adopt the RPKI profile and that identify application-specific requirements that motivate the use of such EKUs.

4.9.6. CRL Distribution Points

This extension MUST be present, except in "self-signed" certificates, and it is non-critical. In a self-signed certificate this extension MUST be omitted.

In this profile, the scope of the CRL is specified to be all

certificates issued by this CA Issuer.

The CRL Distribution Points (CRLDP) extension identifies the location(s) of the CRL(s) associated with certificates issued by this Issuer. The RPKI uses the URI form of object identification. The preferred URI access mechanism is a single RSYNC URI ("rsync://") [RFC5781] that references a single inclusive CRL for each Issuer.

In this profile the certificate Issuer is also the CRL Issuer, implying that the CRLIssuer field MUST be omitted, and the distributionPoint field MUST be present. The Reasons field MUST be omitted.

The distributionPoint MUST contain the fullName field, and MUST NOT contain a nameRelativeToCRLIssuer. The form of the generalName MUST be of type URI.

The sequence of distributionPoint values MUST contain only a single DistributionPoint. The DistributionPoint MAY contain more than one URI value. An RSYNC URI [RFC5781] MUST be present in the DistributionPoint, and reference the most recent instance of this Issuer's CRL. Other access form URIs MAY be used in addition to the RSYNC URI, representing alternate access mechanisms for this CRL.

4.9.7. Authority Information Access

In the context of the RPKI, this extension identifies the publication point of the certificate of the issuer of the certificate in which the extension appears. In this profile a single reference to the publication point of the immediate superior certificate MUST be present, except for a "self-signed" certificate, in which case the extension MUST be omitted. This extension is non-critical.

This profile uses a URI form of object identification. The preferred URI access mechanisms is "rsync", and an RSYNC URI [RFC5781] MUST be specified with an accessMethod value of id-ad-caIssuers. The URI MUST reference the point of publication of the certificate where this Issuer is the Subject (the Issuer's immediate superior certificate). Other accessMethod URIs referencing the same object MAY also be included in the value sequence of this extension.

A CA MUST use a persistent URL name scheme for CA certificates that it issues [ID.sidr-repos-struct]. This implies that a re-issued certificate overwrites a previously issued certificate (to the same Subject) in the publication repository. In this way certificates subordinate to the re-issued (CA) certificate can maintain a constant Authority Information Access (AIA) extension pointer and thus need not be re-issued when the parent certificate is re-issued.

4.9.8. Subject Information Access

In the context of the RPKI, this extension (SIA) identifies the publication point of products signed by the Subject of the certificate.

4.9.8.1. SIA for CA Certificates

This extension MUST be present, and is non-critical.

This extension MUST have an instance of an accessMethod of id-ad-caRepository, with an accessLocation form of a URI that MUST specify an RSYNC URI [RFC5781]. This URI points to the directory containing all published material issued by this CA. i.e., all valid CA certificates, published EE certificates, the current CRL, manifest and signed objects validated via EE certificates that have been issued by this CA [ID.sidr-repos-struct]. Other accessDescription elements with an accessMethod of id-ad-caRepository MAY be present. In such cases, the accessLocation values describe alternate supported URI access mechanisms for the same directory. The ordering of URIs in this accessDescription sequence reflect the CA's relative preferences for access methods to be used by RPs, with the first element of the sequence being the most preferred by the CA.

This extension MUST have an instance of an AccessDescription with an accessMethod of id-ad-rpkiManifest,

```
id-ad OBJECT IDENTIFIER ::= { id-pkix 48 }
```

```
id-ad-rpkiManifest OBJECT IDENTIFIER ::= { id-ad 10 }
```

with an RSYNC URI [RFC5781] form of accessLocation. The URI points to the CA's manifest of published objects [ID.sidr-rpki-manifests] as an object URL. Other accessDescription elements MAY exist for the id-ad-rpkiManifest accessMethod, where the accessLocation value indicates alternate access mechanisms for the same manifest object.

4.9.8.2. SIA for EE Certificates

This extension MUST be present, and is non-critical.

This extension MUST have an instance of an accessMethod of id-ad-signedObject,

```
id-ad-signedObject OBJECT IDENTIFIER ::= { id-ad 11 }
```

with an accessLocation form of a URI that MUST include an RSYNC URI [RFC5781]. This URI points to the signed object that is verified

using this EE certificate [ID.sidr-repos-struct]. Other accessDescription elements may exist for the id-ad-signedObject accessMethod, where the accessLocation value indicates alternate URI access mechanisms for the same object, ordered in terms of the EE's relative preference for supported access mechanisms.

Other AccessMethods MUST NOT be used for an EE certificates's SIA.

4.9.9. Certificate Policies

This extension MUST be present, and MUST be marked critical. It MUST include exactly one policy, as specified in the RPKI CP [ID.sidr-cp]

4.9.10. IP Resources

Either the IP Resources extension, or the AS Resources extension, or both, MUST be present in all RPKI certificates, and if present, MUST be marked critical.

This extension contains the list of IP address resources as per [RFC3779]. The value may specify the "inherit" element for a particular AFI value. In the context of resource certificates describing public number resources for use in the public Internet, the SAFI value MUST NOT be used.

This extension MUST either specify a non-empty set IP address records, or use the "inherit" setting to indicate that the IP address resource set of this certificate is inherited from that of the certificate's issuer.

4.9.11. AS Resources

Either the AS Resources extension, or the IP Resources extension, or both, MUST be present in all RPKI certificates, and if present, MUST be marked critical.

This extension contains the list of AS number resources as per [RFC3779], or may specify the "inherit" element. RDI values are NOT supported in this profile and MUST NOT be used.

This extension MUST either specify a non-empty set AS number records, or use the "inherit" setting to indicate that the AS number resource set of this certificate is inherited from that of the certificate's issuer.

5. Resource Certificate Revocation Lists

Each CA MUST issue a version 2 Certificate Revocation List (CRL), consistent with [RFC5280]. RPs are NOT required to process version 1 CRLs (in contrast to [RFC5280]). The CRL Issuer is the CA. CRLs conforming to this profile MUST NOT include Indirect or Delta CRLs. The scope of each CRL MUST be all certificates issued by this CA.

The Issuer name is as in Section 4.4 above.

Where two or more CRLs issued by the same CA, the CRL with the highest value of the "CRL Number" field supersedes all other CRLs issued by this CA.

The algorithm used in CRLs issued under this profile is specified in [ID.sidr-rpki-algs].

The contents of the CRL are a list of all non-expired certificates that have been revoked by the CA.

An RPKI CA MUST include the two extensions Authority Key Identifier and CRL Number in every CRL that it issues. RPs MUST be prepared to process CRLs with these extensions. No other CRL extensions are allowed.

For each revoked resource certificate only the two fields Serial Number and Revocation Date MUST be present, and all other fields MUST NOT be present. No CRL entry extensions are supported in this profile, and CRL entry extensions MUST NOT be present in a CRL.

6. Resource Certificate Requests

A resource certificate request MAY use either of PKCS#10 or Certificate Request Message Format (CRMF). A CA MUST support certificate issuance in PKCS#10 and a CA MAY support CRMF requests.

Note that there is no certificate response defined in this profile. For CA certificate requests, the CA places the Resource Certificate in the repository, as per [ID.sidr-cp]. No response is defined for EE Certificate requests.

6.1. PKCS#10 Profile

This profile refines the specification in [RFC2986], as it relates to Resource Certificates. A Certificate Request Message object, formatted according to PKCS#10, is passed to a CA as the initial step in issuing a certificate.

With the exception of the SubjectPublicKeyInfo and the SIA extension request, the CA is permitted to alter any field in the request when issuing a certificate.

6.1.1. PKCS#10 Resource Certificate Request Template Fields

This profile applies the following additional requirements to fields that MAY appear in a CertificationRequestInfo:

Version

This field is mandatory and MUST have the value 0.

Subject

This field MAY be omitted. If present, the value of this field SHOULD be empty (i.e., NULL), in which case the CA MUST generate a Subject name that is unique in the context of certificates issued by this CA. This field is allowed to be non-empty only for a rekey/reissuance request, and only if the CA has adopted a policy (in its Certificate Practice Statement (CPS)) that permits name reuse in these circumstances.

SubjectPublicKeyInfo

This field specifies the Subject's public key and the algorithm with which the key is used. The algorithm used in this profile is specified in [ID.sidr-rpki-algs].

Attributes

[RFC2986] defines the attributes field as key-value pairs where the key is an OID and the value's structure depends on the key.

The only attribute used in this profile is the ExtensionRequest attribute as defined in [RFC2985]. This attribute contains certificate Extensions. The profile for extensions in certificate requests is specified in Section 6.3.

This profile applies the following additional constraints to fields that MAY appear in a CertificationRequest Object:

signatureAlgorithm

The signatureAlgorithm value is specified in [ID.sidr-rpki-algs].

6.2. CRMF Profile

This profile refines the Certificate Request Message Format (CRMF) specification in [RFC4211], as it relates to Resource Certificates.

A Certificate Request Message object, formatted according to the CRMF, is passed to a CA as the initial step in certificate issuance.

With the exception of the SubjectPublicKeyInfo and the SIA extension request, the CA is permitted to alter any requested field when issuing the certificate.

6.2.1. CRMF Resource Certificate Request Template Fields

This profile applies the following additional requirements to fields that may appear in a Certificate Request Template:

version

This field SHOULD be omitted. If present, it MUST specify a request for a Version 3 Certificate. It

serialNumber

This field MUST be omitted.

signingAlgorithm

This field MUST be omitted.

issuer

This MUST be omitted in this profile.

Validity

This field MAY be omitted. If omitted, the CA will issue a Certificate with Validity dates as determined by the CA. If specified, then the CA MAY override the requested values with dates as determined by the CA.

Subject

This field MAY be omitted. If present, the value of this field SHOULD be empty (i.e., NULL), in which case the CA MUST generate a Subject name that is unique in the context of certificates issued by this CA. This field is allowed to be non-empty only for a rekey/reissuance request, and only if the CA has adopted a policy (in its CPS) that permits name reuse in these circumstances.

PublicKey

This field MUST be present.

extensions

The profile for extensions in certificate requests is specified in Section 6.3.

6.2.2. Resource Certificate Request Control Fields

The following control fields are supported in this profile:

Authenticator Control

'The intended model of authentication of the Subject is a "long term" model, and the guidance offered in [RFC4211] is that the Authenticator Control field be used.

6.3. Certificate Extension Attributes in Certificate Requests

The following extensions MAY appear in a PKCS#10 or CRMF Certificate Request. Any other extensions MUST NOT appear in a Certificate Request. This profile places the following additional constraints on these extensions:

BasicConstraints

If this is omitted then the CA will issue an EE certificate (hence no BasicConstraints extension will be included).

The pathLengthConstraint is not supported in this profile, and this field MUST be omitted.

The CA MAY honour the cA boolean if set to true (CA certificate request). If this bit is set, then it indicates that the Subject is requesting a CA certificate.

The CA MUST honour the cA bit if set to false (EE certificate request), in which case the corresponding EE certificate will not contain a Basic Constraints extension.

KeyUsage

The CA MAY honour KeyUsage extensions of keyCertSign and cRLSign if present, as long as this is consistent with the BasicConstraints SubjectType sub field, when specified.

ExtendedKeyUsage

The CA MAY honour ExtendedKeyUsage extensions of keyCertSign and cRLSign if present, as long as this is consistent with the BasicConstraints SubjectType sub field, when specified.

SubjectInformationAccess

This field MUST be present, and the field value SHOULD be honoured by the CA if it conforms to the requirements set forth in Section 4.9.8. If the CA is unable to honour the requested value for this field, then the CA MUST reject the Certificate Request.

7. Resource Certificate Validation

This section describes the Resource Certificate validation procedure. This refines the generic procedure described in section 6 of [RFC5280].

7.1. Resource Extension Validation

The IP Resources and AS Resources extensions definitions [RFC3779] define critical extensions for INRs. These are ASN.1 encoded representations of the IPv4 and IPv6 address range and an AS number set.

Valid Resource Certificates MUST have a valid IP address and/or AS number resource extension. In order to validate a Resource Certificate the resource extension MUST also be validated. This validation process relies on definitions of comparison of resource sets:

more specific

Given two IP address or AS number contiguous ranges, A and B, A is "more specific" than B if range B includes all IP addresses or AS numbers described by range A, and if range B is larger than range A.

equal

Given two IP address or AS number contiguous ranges, A and B, A is "equal" to B if range A describes precisely the same collection of IP addresses or AS numbers as described by range B. The definition of "inheritance" in [RFC3779] is equivalent to this "equality" comparison.

encompass

Given two IP address and AS number sets X and Y, X "encompasses" Y if, for every contiguous range of IP addresses or AS numbers elements in set Y, the range element is either "more specific" than or "equal" to a contiguous range element within the set X.

Validation of a certificate's resource extension in the context of a Certification Path (see Section 7.2 entails that for every adjacent pair of certificates in the certification path (certificates 'x' and 'x + 1'), the number resources described in certificate 'x' "encompass" the number resources described in certificate 'x + 1', and the resources described in the trust anchor information "encompass" the resources described in the first certificate in the certification path.

7.2. Resource Certification Path Validation

Validation of signed resource data using a target resource certificate consists of verifying that the digital signature of the signed resource data is valid, using the public key of the target resource certificate, and also validating the resource certificate in the context of the RPKI, using the path validation process. This path validation process verifies, among other things, that a prospective certification path (a sequence of n certificates) satisfies the following conditions:

1. for all 'x' in {1, ..., n-1}, the Subject of certificate 'x' is the Issuer of certificate ('x' + 1);
2. certificate '1' is issued by a trust anchor;
3. certificate 'n' is the certificate to be validated; and
4. for all 'x' in {1, ..., n}, certificate 'x' is valid.

Certificate validation entails verifying that all of the following conditions hold, in addition to the Certification Path Validation criteria specified in Section 6 of [RFC5280]:

1. The certificate can be verified using the Issuer's public key and the signature algorithm
2. The current time lies within the certificate's Validity From and To values.
3. The certificate contains all fields that MUST be present, as defined by this specification, and contains values for selected fields that are defined as allowable values by this specification.

4. No field, or field value, that this specification defines as MUST NOT be present is used in the certificate.
5. The Issuer has not revoked the certificate. A revoked certificate is identified by the certificate's serial number being listed on the Issuer's current CRL, as identified by the CRLDP of the certificate, the CRL is itself valid, and the public key used to verify the signature on the CRL is the same public key used to verify the certificate itself.
6. That the resource extension data is "encompassed" by the resource extension data contained in a valid certificate where this Issuer is the Subject (the previous certificate in the context of the ordered sequence defined by the Certification Path).
7. The Certification Path originates with a certificate issued by a trust anchor, and there exists a signing chain across the Certification Path where the Subject of Certificate 'x' in the Certification Path matches the Issuer in Certificate 'x + 1' in the Certification Path, and the public key in Certificate 'x' can verify the signature value in Certificate 'x+1'.

A certificate validation algorithm MAY perform these tests in any chosen order.

Certificates and CRLs used in this process MAY be found in a locally maintained cache, maintained by a regular synchronisation across the distributed publication repository structure [ID.sidr-repos-struct].

There exists the possibility of encountering certificate paths that are arbitrarily long, or attempting to generate paths with loops as means of creating a potential DOS attack on an RP. A RP executing this procedure MAY apply further heuristics to guide halting the certification path validation process in order to avoid some of the issues associated with attempts to validate such malformed certification path structures. Implementations of Resource Certificate validation MAY halt with a validation failure if the certification path length exceeds a locally defined configuration parameter.

8. Design Notes

The following notes provide some additional commentary on the considerations that lie behind some of the design choices that were made in the design of this certificate profile. These notes are non-normative, i.e. this section of the document does not constitute a

formal part of the profile specification, and the interpretation of key words as defined in RFC2119 are not applicable in this section of the document.

Certificate Extensions:

This profile does not permit the use of any other critical or non-critical extensions. The rationale for this restriction is that the resource certificate profile is intended for a specific defined use. In this context it is not seen as being appropriate to be in the position of having certificates with additional non-critical extensions that RPs may see as valid certificates without understanding the extensions, but were the RP in a position to understand the extensions, would contradict or qualify in some way this original judgment of validity. This profile takes the position of minimalism over extensibility. The specific goal for the associated RPKI is to precisely match the INR allocation structure through an aligned certificate structure that describes the allocation and its context within the INR distribution hierarchy. The profile defines a resource certificate that is structured to meet these requirements.

Certification Authorities and Key Values:

This profile uses a definition of an instance of a CA as a combination of a named entity and a key pair. Within this definition a CA instance cannot rollover a key pair. However, the entity can generate a new instance of a CA with a new key pair and roll over all the signed subordinate products to the new CA [ID.sidr-keyroll].

This has a number of implications in terms of Subject name management, CRL Scope and repository publication point management.

CRL Scope and Key Values:

For CRL Scope this profile specifies that a CA issues a single CRL at a time, and the scope of the CRL is all certificates issued by this CA. Because the CA instance is bound to a single key pair this implies that the CA's public key, the key used to validate the CA's CRL, and the key used to validate the certificates revoked by that CRL are all the same key value.

Repository Publication Point:

The definition of a CA affects the design of the repository publication system. In order to minimize the amount of forced re-certification on key rollover events, a repository publication regime that uses the same repository publication point for all CA instances that refers to the same entity, but

with different key values will minimize the extent of re-generation of certificates to only immediate subordinate certificates. This is described in [ID.sidr-keyroll].

Subject Name:

This profile specifies that Subject names must be unique per Issuer, and does not specify that Subject names must be globally unique (in terms of assured uniqueness). This is due to the nature of the RPKI as a distributed PKI, implying that there is no ready ability for Certification authorities to coordinate a simple RPKI-wide unique name space without resorting to additional critical external dependencies. CAs are advised to use Subject name generation procedures that minimize the potential for name clashes.

One way to achieve this is for a CA to use a Subject name practice that uses the CommonName component of the Distinguished Name as a constant value for any given entity that is the Subject of CA-issued certificates, and set the serialNumber component of the Distinguished Name to a value that is derived from the hash of the subject public key value.

If the CA elects not to use the serialNumber component of the DistinguishedName, then it is considered beneficial that a CA generates CommonNames that have themselves a random component that includes significantly more than 40 bits of entropy in the name. Some non-normative recommendations to achieve this include:

- 1) Hash of the subject public key (encoded as ASCII HEX).
example: cn="999d99d564de366a29cd8468c45ede1848e2cc14"
- 2) A Universally Unique Identifier (UUID) [RFC4122]
example: cn="6437d442-6fb5-49ba-bbdb-19c260652098"
- 3) A randomly generated ASCII HEX encoded string of length 20 or greater:
example: cn="0f8fcc28e3be4869bc5f8fa114db05e1">
(A string of 20 ASCII HEX digits would have 80-bits of entropy)
- 4) An internal database key or subscriber ID combined with one of the above
example: cn="<DBkey1> (6437d442-6fb5-49ba-bbdb-19c2606520980)"
(The issuing CA may wish to be able to extract the database key or subscriber ID from the commonName. Since only the issuing CA would need to be able to parse the

commonName, the database key and the source of entropy (e.g., a UUID) could be separated in any way that the CA wanted, as long as it conformed to the rules for PrintableString. The separator could be a space character, parenthesis, hyphen, slash, question mark, etc.

9. Security Considerations

The Security Considerations of [RFC5280] and [RFC3779] apply to Resource Certificates. The Security Considerations of [RFC2986] and [RFC4211] apply to Resource Certificate certification requests.

A Resource Certificate PKI cannot in and of itself resolve any forms of ambiguity relating to uniqueness of assertions of rights of use in the event that two or more valid certificates encompass the same resource. If the issuance of resource certificates is aligned to the status of resource allocations and assignments then the information conveyed in a certificate is no better than the information in the allocation and assignment databases.

This profile requires that the key used to sign an issued certificate is the same key used to sign the CRL that can revoke the certificate, implying that the certification path used to validate the signature on a certificate is the same as that used to validate the signature of the CRL that can revoke the certificate. It is noted that this is a higher constraint than required in X.509 PKIs, and there may be a risk in using a path validation implementation that is capable of using separate validation paths for a certificate and the corresponding CRL. If there are subject name collisions in the RPKI as a result of CAs not following the guidelines provided here relating to ensuring sufficient entropy in constructing subject names, and this is combined with the situation that an RP uses an implementation of validation path construction that is not in conformance with this RPKI profile, then it is possible that the subject name collisions can cause an RP to conclude that an otherwise valid certificate has been revoked.

10. IANA Considerations

[Note to IANA, to be removed prior to publication: there are no IANA considerations stated in this document.]

11. Acknowledgements

The authors would like to particularly acknowledge the valued contribution from Stephen Kent in reviewing this document and proposing numerous sections of text that have been incorporated into the text. The authors also acknowledge the contributions of Sandy Murphy, Robert Kisteleki, Randy Bush, Russ Housley, Ricardo Patara and Rob Austein in the preparation and subsequent review of this document. The document also reflects review comments received from Roque Gagliano, Sean Turner and David Cooper.

12. References

12.1. Normative References

- [ID.sidr-cp] Kent, S., Kong, D., Seo, K., and R. Watro, "Certificate Policy (CP) for the Resource PKI (RPKI)", Work in progress: Internet Drafts draft-ietf-sidr-c-13.txt, September 2010.
- [ID.sidr-rpki-algs] Huston, G., "A Profile for Algorithms and Key Sizes for use in the Resource Public Key Infrastructure", Work in progress: Internet Drafts draft-ietf-sidr-rpki-algs-00.txt, August 2009.
- [RFC2986] Nystrom, M. and B. Kaliski, "PKCS #10: Certification Request Syntax Specification Version 1.7", RFC 2986, November 2000.
- [RFC3779] Lynn, C., Kent, S., and K. Seo, "X.509 Extensions for IP Addresses and AS Identifiers", RFC 3779, June 2004.
- [RFC4211] Schaad, J., "Internet X.509 Public Key Infrastructure Certificate Request Message Format (CRMF)", RFC 4211, September 2005.
- [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, May 2008.
- [X.509] ITU-T, "Recommendation X.509: The Directory - Authentication Framework", 2000.

12.2. Informative References

[ID.sidr-arch]

Lepinski, M. and S. Kent, "An Infrastructure to Support Secure Internet Routing", Work in progress: Internet Drafts draft-ietf-sidr-arch-04.txt, November 2008.

[ID.sidr-keyroll]

Huston, G., Michaelson, G., and S. Kent, "CA Key Rollover in the RPKI", draft-ietf-sidr-keyroll-02.txt (work in progress), October 2010.

[ID.sidr-repos-struct]

Huston, G., Loomans, R., and G. Michaelson, "A Profile for Resource Certificate Repository Structure", draft-ietf-sidr-repos-struct-04.txt (work in progress), May 2010.

[ID.sidr-rpki-manifests]

Austein, R., Huston, G., Kent, S., and M. Lepinski, "Manifests for the Resource Public Key Infrastructure", Work in progress: Internet Drafts draft-ietf-sidr-rpki-manifests-04.txt, October 2008.

[ID.sidr-signed-object]

Lepinski, M., Chi, A., and S. Kent, "Signed Object Template for the Resource Public Key Infrastructure", draft-ietf-sidr-signed-object-01.txt (work in progress), October 2010.

[RFC2985] Nystrom, M. and B. Kaliski, "PKCS #9: Selected Object Classes and Attribute Types Version 2.0", RFC 2985, November 2000.

[RFC4122] Leach, P., Mealling, M., and R. Salz, "A Universally Unique Identifier (UUID) URN Namespace", RFC 4122, July 2005.

[RFC5781] Weiler, S., Ward, D., and R. Housley, "The rsync URI Scheme", RFC 5781, February 2010.

Appendix A. Example Resource Certificate

The following is an example Resource Certificate.

Certificate Name: 9JfgAEcq7Q-47IwMC5CJIJr6EJs.cer

Data:

Version: 3 (0x2)
Serial: 1500 (0x5dc)
Signature Algorithm: SHA256WithRSAEncryption
Issuer: CN=APNIC Production-CVPQSgUkLy7pOXdNeVWGvnFX_0s
Validity
Not Before: Oct 25 12:50:00 2008 GMT
Not After : Jan 31 00:00:00 2010 GMT
Subject: CN=A91872ED
Subject Public Key Info:
Public Key Algorithm: rsaEncryption
RSA Public Key: (2048 bit)
Modulus (2048 bit):
00:bb:fb:4a:af:a4:b9:dc:d0:fa:6f:67:cc:27:39:
34:d1:80:40:37:de:88:d1:64:a2:f1:b3:fa:c6:7f:
bb:51:df:e1:c7:13:92:c3:c8:a2:aa:8c:d1:11:b3:
aa:99:c0:ac:54:d3:65:83:c6:13:bf:0d:9f:33:2d:
39:9f:ab:5f:cd:a3:e9:a1:fb:80:7d:1d:d0:2b:48:
a5:55:e6:24:1f:06:41:35:1d:00:da:1f:99:85:13:
26:39:24:c5:9a:81:15:98:fb:5f:f9:84:38:e5:d6:
70:ce:5a:02:ca:dd:61:85:b3:43:2d:0b:35:d5:91:
98:9d:da:1e:0f:c2:f6:97:b7:97:3e:e6:fc:c1:c4:
3f:30:c4:81:03:25:99:09:4c:e2:4a:85:e7:46:4b:
60:63:02:43:46:51:4d:ed:fd:a1:06:84:f1:4e:98:
32:da:27:ee:80:82:d4:6b:cf:31:ea:21:af:6f:bd:
70:34:e9:3f:d7:e4:24:cd:b8:e0:0f:8e:80:eb:11:
1f:bc:c5:7e:05:8e:5c:7b:96:26:f8:2c:17:30:7d:
08:9e:a4:72:66:f5:ca:23:2b:f2:ce:54:ec:4d:d9:
d9:81:72:80:19:95:57:da:91:00:d9:b1:e8:8c:33:
4a:9d:3c:4a:94:bf:74:4c:30:72:9b:1e:f5:8b:00:
4d:e3
Exponent: 65537 (0x10001)
X509v3 extensions:
X509v3 Subject Key Identifier:
F4:97:E0:00:47:2A:ED:0F:B8:EC:8C:0C:0B:90:89:
20:9A:FA:10:9B

X509v3 Authority Key Identifier:
keyid:09:53:D0:4A:05:24:2F:2E:E9:39:77:4D:79:
55:86:BE:71:57:FF:4B

X509v3 Key Usage: critical
Certificate Sign, CRL Sign

X509v3 Basic Constraints: critical
CA:TRUE

X509v3 CRL Distribution Points:

URI:rsync://rpki.apnic.net/repository/A3C38A24
D60311DCAB08F31979BDBE39/CVPQSgUkLy7pOXdNe
VWGvnFX_0s.crl

Authority Information Access:

CA Issuers - URI:rsync://rpki.apnic.net/repos
itory/8BDFC7DED5FD11DCB14CF4B1A703F9B7/CVP
QSgUkLy7pOXdNeVWGvnFX_0s.cer

X509v3 Certificate Policies: critical

Policy: 1.3.6.1.5.5.7.14.2

Subject Information Access:

CA Repository - URI:rsync://rpki.apnic.net/mem
ber_repository/A91872ED/06A83982887911DD81
3F432B2086D636/

Manifest - URI:rsync://rpki.apnic.net/member_r
epository/A91872ED/06A83982887911DD813F432
B2086D636/9JfgAEcq7Q-47IwMC5CJIJr6EJs.mft

sbgp-autonomousSysNum: critical

Autonomous System Numbers:

24021

38610

131072

131074

sbgp-ipAddrBlock: critical

IPv4:

203.133.248.0/22

203.147.108.0/23

Signature Algorithm: sha256WithRSAEncryption

51:4c:77:e4:21:64:80:e9:35:30:20:9f:d8:4b:88:60:b8:1f:
73:24:9d:b5:17:60:65:6a:28:cc:43:4b:68:97:ca:76:07:eb:
dc:bd:a2:08:3c:8c:56:38:c6:0a:1e:a8:af:f5:b9:42:02:6b:
77:e0:b1:1c:4a:88:e6:6f:b6:17:d3:59:41:d7:a0:62:86:59:
29:79:26:76:34:d1:16:2d:75:05:cb:b2:99:bf:ca:c6:68:1b:
b6:a9:b0:f4:43:2e:df:e3:7f:3c:b3:72:1a:99:fa:5d:94:a1:
eb:57:9c:9a:2c:87:d6:40:32:c9:ff:a6:54:b8:91:87:fd:90:
55:ef:12:3e:1e:2e:cf:c5:ea:c3:4c:09:62:4f:88:00:a0:7f:
cd:67:83:bc:27:e1:74:2c:18:4e:3f:12:1d:ef:29:0f:e3:27:
00:ce:14:eb:f0:01:f0:36:25:a2:33:a8:c6:2f:31:18:22:30:
cf:ca:97:43:ed:84:75:53:ab:b7:6c:75:f7:2f:55:5c:2e:82:
0a:be:91:59:bf:c9:06:ef:bb:b4:a2:71:9e:03:b1:25:8e:29:
7a:30:88:66:b4:f2:16:6e:df:ad:78:ff:d3:b2:9c:29:48:e3:
be:87:5c:fc:20:2b:df:da:ca:30:58:c3:04:c9:63:72:48:8c:
0a:5f:97:71

Appendix B. Example Certificate Revocation List

The following is an example Certificate Revocation List.

CRL Name: q66IrWSGuBE7jqx8PAUHALHCqRw.crl

Data:

Version: 2

Signature Algorithm:

Hash: SHA256, Encryption: RSA

Issuer: CN=Demo Production APNIC CA - Not for real use,
E=ca@apnic.net

This Update: Thu Jul 27 06:30:34 2006 GMT

Next Update: Fri Jul 28 06:30:34 2006 GMT

Authority Key Identifier: Key Identifier:

ab:ae:88:ad:64:86:b8:11:3b:8e:ac:7c:3c:05:
07:02:51:c2:a9:1c

Authority Key Identifier: Key Identifier g(AKI):

q66IrWSGuBE7jqx8PAUHALHCqRw

CRLNumber: 4

Revoked Certificates: 1

Serial Number: 1

Revocation Date: Mon Jul 17 05:10:19 2006 GMT

Serial Number: 2

Revocation Date: Mon Jul 17 05:12:25 2006 GMT

Serial Number: 4

Revocation Date: Mon Jul 17 05:40:39 2006 GMT

Signature:

b2:5a:e8:7c:bd:a8:00:0f:03:1a:17:fd:40:2c:46:
0e:d5:64:87:e7:e7:bc:10:7d:b6:3e:39:21:a9:12:
f4:5a:d8:b8:d4:bd:57:1a:7d:2f:7c:0d:c6:4f:27:
17:c8:0e:ae:8c:89:ff:00:f7:81:97:c3:a1:6a:0a:
f7:d2:46:06:9a:d1:d5:4d:78:e1:b7:b0:58:4d:09:
d6:7c:1e:a0:40:af:86:5d:8c:c9:48:f6:e6:20:2e:
b9:b6:81:03:0b:51:ac:23:db:9f:c1:8e:d6:94:54:
66:a5:68:52:ee:dd:0f:10:5d:21:b8:b8:19:ff:29:
6f:51:2e:c8:74:5c:2a:d2:c5:fa:99:eb:c5:c2:a2:
d0:96:fc:54:b3:ba:80:4b:92:7f:85:54:76:c9:12:
cb:32:ea:1d:12:7b:f8:f9:a2:5c:a1:b1:06:8e:d8:
c5:42:61:00:8c:f6:33:11:29:df:6e:b2:cc:c3:7c:
d3:f3:0c:8d:5c:49:a5:fb:49:fd:e7:c4:73:68:0a:
09:0e:6d:68:a9:06:52:3a:36:4f:19:47:83:59:da:
02:5b:2a:d0:8a:7a:33:0a:d5:ce:be:b5:a2:7d:8d:
59:a1:9d:ee:60:ce:77:3d:e1:86:9a:84:93:90:9f:
34:a7:02:40:59:3a:a5:d1:18:fb:6f:fc:af:d4:02:
d9

Authors' Addresses

Geoff Huston
APNIC

Email: gih@apnic.net
URI: <http://www.apnic.net>

George Michaelson
APNIC

Email: ggm@apnic.net
URI: <http://www.apnic.net>

Robert Loomans
APNIC

Email: robertl@apnic.net
URI: <http://www.apnic.net>

Secure Inter-Domain Routing
Internet-Draft
Intended status: Standards Track
Expires: April 18, 2011

G. Huston
R. Loomans
B. Ellacott
APNIC
R. Austein
ISC
October 15, 2010

A Protocol for Provisioning Resource Certificates
draft-ietf-sidr-rescerts-provisioning-07.txt

Abstract

This document defines a framework for certificate management interactions between a resource issuer ("Issuer") and a resource recipient ("Subject") through the specification of a protocol for interaction between the two parties. The protocol supports the transmission of requests from the Subject, and corresponding responses from the Issuer encompassing the actions of certificate issuance, certificate revocation and certificate status information reports. This protocol is intended to be limited to the application of resource certificate management and is not intended to be used as part of a more general certificate management framework.

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 <http://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 April 18, 2011.

Copyright Notice

Copyright (c) 2010 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal

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

Table of Contents

1. Introduction	3
1.1. Terminology	3
2. Scope	3
3. Protocol Specification	4
3.1. CMS Profile	5
3.1.1. SignedData Content Type	5
3.1.2. CMS Object Validation	10
3.2. Common Message format	11
3.3. Control - Resource Class Query	13
3.3.1. Resource Class List Query	13
3.3.2. Resource Class List Response	14
3.4. CA - Certificate Issuance	18
3.4.1. Certificate Issuance Request	18
3.4.2. Certificate Issuance Response	20
3.5. Certificate Revocation	21
3.5.1. Certificate Revocation Request	21
3.5.2. Certificate Revocation Response	22
3.6. Request-Not-Performed Response	23
4. Security Considerations	24
5. IANA Considerations	25
6. Acknowledgements	25
7. References	25
7.1. Normative References	25
7.2. Informative References	26
Appendix A. CMS Signed Object	26
Appendix B. XML Schema	28
Authors' Addresses	30

1. Introduction

This document defines a framework for certificate management interactions between a resource issuer ("Issuer") and a resource recipient ("Subject") through the specification of a protocol for interaction between the two parties. The protocol supports the transmission of requests from the Subject, and corresponding responses from the Issuer encompassing the actions of certificate issuance, certificate revocation and certificate status information reports. This protocol is intended to be limited to the application of resource certificate management and is not intended to be used as part of a more general certificate management framework.

1.1. Terminology

Terms used in this document are:

"Issuer" used in the context of this document as an entity undertaking the role of resource issuer. An "Issuer" is a Certificate Authority, and can issue Resource Certificates.

"Subject" used in the context of this document as an entity undertaking the role of resource recipient who is the subject of a Resource Certificate. A "Subject" may be issued with a CA-enabled certificate, allowing the entity to also assume the role of an "Issuer".

"resource class" a resource class refers to a collection of resources that can be certified in a single resource certificate by an issuer.

"server" in the context of this client/server protocol specification, the Issuer assumes the role of the "server."

"client" in the context of this client/server protocol specification, the Subject assumes the role of the "client."

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119.

2. Scope

This Resource Public Key Infrastructure (RPKI) certificate provisioning protocol defines a basic set of interactions that allow a Subject to request certificate issuance, revocation and status

information from the Issuer, and for a Issuer to maintain an issued certificate set that is aligned to the allocation records relating to each Subject. The Issuer's resource allocation database is the authoritative source of what resource allocations the Issuer may certify for a Subject.

A resource recipient (Subject) may also undertake the role of a resource issuer (Issuer).

This protocol specification does not encompass:

- o signing of objects with keys that are certified by resource certificates, nor the issuance of end-entity certificates.
- o the specification of interaction with the Issuer's resource allocation database, nor the specification of a protocol to manage the publication repository.
- o the interactions between client and server that establish identities, exchange the keys used in the protection of the communications channel between client and server, and the exchange of the certificates and validation Public Key Infrastructure (PKI) contexts used in the Cryptographic Message Syntax message exchange.

3. Protocol Specification

This RPKI certificate provisioning protocol is expressed as a simple request/response interaction, where the client passes a request to the server, and the server generates a corresponding response.

The protocol is implemented as an exchange of messages.

Messages are passed over an HTTP [RFC2616] end-to-end connection. A message exchange commences with the client initiating an HTTP POST with content type of "application/x-rpki", with the message object as the body. The server's response will similarly be the body of the response with a content type of "application/x-rpki".

The content of the POST, and the server's response, will be a "well-formed" Cryptographic Message Syntax (CMS) [RFC5652] object, encoded using the Distinguished Encoding Rules for ASN.1 (DER) [X.509-88], formatted in accordance with the CMS profile specified in the following section. CMS is used as the signing format to sign the message object. The public part of the signing key and the associated certificate chain that is used to validate the CMS digital signature is assumed to have been communicated between the two

entities, through mechanisms not defined in this specification.

The protocol's request / response interaction is assumed to be reliable, in that all requests will generate a matching response. The protocol requires sequential operation for each distinct client, where the server **MUST NOT** accept a client's request unless it has generated and sent a response to the client's previous request. Attempts by the client to initiate multiple requests in parallel **MUST** be detected by the server and rejected with an error response.

3.1. CMS Profile

The format of the CMS object is:

```
ContentInfo ::= SEQUENCE {  
    contentType ContentType,  
    content [0] EXPLICIT ANY DEFINED BY contentType }  
  
ContentType ::= OBJECT IDENTIFIER
```

The ContentType is the signed-data type of id-data, namely id-signedData, OID = 1.2.840.113549.1.7.2. [RFC5652]

3.1.1. SignedData Content Type

According to the CMS standard [RFC5652], signed-data content types is the ASN.1 type SignedData:

```
SignedData ::= SEQUENCE {  
    version CMSVersion,  
    digestAlgorithms DigestAlgorithmIdentifiers,  
    encapContentInfo EncapsulatedContentInfo,  
    certificates [0] IMPLICIT CertificateSet OPTIONAL,  
    crls [1] IMPLICIT RevocationInfoChoices OPTIONAL,  
    signerInfos SignerInfos }  
  
DigestAlgorithmIdentifiers ::= SET OF DigestAlgorithmIdentifier  
  
SignerInfos ::= SET OF SignerInfo
```

Additionally, the SignerInfos set **MUST** contain only a single SignerInfo object.

3.1.1.1. version

The version is the syntax version number. It **MUST** be 3, corresponding to the signerInfo structure having version number 3.

3.1.1.2. digestAlgorithms

The digestAlgorithms set contains the OIDs of the digest algorithm(s) used in signing the encapsulated content. This set MUST contain exactly one digest algorithm OID, and the OID MUST be selected from those specified in [ID.sidr-rpki-als].

3.1.1.3. encapContentInfo

encapContentInfo is the signed content, consisting of a content type identifier and the content itself. The encapContentInfo represents the payload of the RPKI certificate provisioning protocol.

```
EncapsulatedContentInfo ::= SEQUENCE {  
    eContentType ContentType,  
    eContent [0] EXPLICIT OCTET STRING OPTIONAL }
```

```
ContentType ::= OBJECT IDENTIFIER
```

3.1.1.3.1. eContentType

The eContentType for the RPKI Protocol Message object is defined as id-ct-xml, and has the numerical value of 1.2.840.113549.1.9.16.1.28.

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

```
id-ct OBJECT IDENTIFIER ::= { id-smime 1 }
```

```
id-ct-xml OBJECT IDENTIFIER ::= { id-ct 28 }
```

3.1.1.3.2. eContent

The content of a RPKI XML Protocol Object consists of a single protocol message, structured according to a defined XML schema, as defined in subsequent sections of this document. The eContent field of the CMS object is formally defined using ASN.1 as:

```
id-ct-xml ::= OCTET STRING -- XML encoded message
```

3.1.1.4. certificates

This field MUST be present, and MUST contain the EE certificate of the key pair whose private key value was used to sign the CMS. This MUST NOT be an RPKI certificate, and SHOULD be a certificate that is recognised to attest to the identity of the party that created the CMS object.

This field MAY contain CA certificates that a relying party may use to validate the EE certificate.

3.1.1.5. crls

This field MUST be present. The contents of the field are specified in [RFC5652].

3.1.1.6. signerInfo

SignerInfo is defined in CMS as:

```
SignerInfo ::= SEQUENCE {  
    version CMSVersion,  
    sid SignerIdentifier,  
    digestAlgorithm DigestAlgorithmIdentifier,  
    signedAttrs [0] IMPLICIT SignedAttributes OPTIONAL,  
    signatureAlgorithm SignatureAlgorithmIdentifier,  
    signature SignatureValue,  
    unsignedAttrs [1] IMPLICIT UnsignedAttributes OPTIONAL }
```

3.1.1.6.1. version

The version number MUST be 3, corresponding with the choice of SubjectKeyIdentifier for the sid.

3.1.1.6.2. sid

The sid is defined as:

```
SignerIdentifier ::= CHOICE {  
    issuerAndSerialNumber IssuerAndSerialNumber,  
    subjectKeyIdentifier [0] SubjectKeyIdentifier }
```

In this profile, the sid MUST be the SubjectKeyIdentifier that appears in the EE certificate carried in the CMS certificates field.

3.1.1.6.3. digestAlgorithm

The digestAlgorithm MUST consist of the OID of a digest algorithm that conforms to the RPKI Algorithms and Key Size Profile specification [ID.sidr-rpki-algs].

3.1.1.6.4. signedAttrs

The signedAttrs field is defined as:

SignedAttributes ::= SET SIZE (1..MAX) OF Attribute

UnsignedAttributes ::= SET SIZE (1..MAX) OF Attribute

Attribute ::= SEQUENCE {
 attrType OBJECT IDENTIFIER,
 attrValues SET OF AttributeValue }

AttributeValue ::= ANY

The signedAttr element MUST be present and MUST include the content-type, and message-digest [RFC5652] attributes. If the either the signing-time [RFC5652] attribute or the the binary-signing-time attribute [RFC6019] attribute, or both attributes, are present they MUST also be included as the SignedAttributes. Other signed attributes MUST NOT be included.

The signedAttr MUST include only a single instance of any particular attribute. Additionally, even though the syntax allows for a SET OF AttributeValue, in this profile the attrValues MUST consist of only a single AttributeValue.

3.1.1.6.4.1. Content-Type Attribute

The ContentType attribute MUST be present. The attrType OID for the ContentType attribute is 1.2.840.113549.1.9.3.

id-contentType OBJECT IDENTIFIER ::= { iso(1) member-body(2)
 us(840) rsadsi(113549) pkcs(1) pkcs9(9) 3 }

ContentType ::= OBJECT IDENTIFIER

The attrValues for the ContentType attribute MUST match the eContentType in the EncapsulatedContentInfo. This OID value is defined as id-ct-xml, and has the numerical value of 1.2.840.113549.1.9.16.1.28.

3.1.1.6.4.2. Message-Digest Attribute

The MessageDigest Attribute MUST be present. The attrType OID for the MessageDigest Attribute is 1.2.840.113549.1.9.4.

id-messageDigest OBJECT IDENTIFIER ::= { iso(1) member-body(2)
 us(840) rsadsi(113549) pkcs(1) pkcs9(9) 4 }

MessageDigest ::= OCTET STRING

The attrValues for the MessageDigest attribute contains the output of

the digest algorithm applied to the content being signed, as specified in Section 5.4 of [RFC5652].

3.1.1.6.4.3. SigningTime Attribute

The SigningTime attribute MAY be present. The attrType OID for the SigningTime attribute is 1.2.840.113549.1.9.5.

```
id-signingTime OBJECT IDENTIFIER ::= { iso(1) member-body(2)
    us(840) rsadsi(113549) pkcs(1) pkcs9(9) 5 }
```

```
SigningTime ::= Time
```

```
Time ::= CHOICE {
    utcTime UTCTime,
    generalizedTime GeneralizedTime }
```

The SigningTime attribute specifies the time, based on the local system clock, when the digital signature was applied to the content.

Guidelines regarding the use of UTCTime and GeneralizedTime in the Signing Time attribute can be found in Section 11.3 of [RFC5652].

Either one of the SigningTime attribute or the BinarySigningTime attribute, or both attributes, MUST be present. If both the SigningTime and BinarySigningTime attributes are present they MUST both represent the same underlying time value.

3.1.1.6.4.4. BinarySigningTime Attribute

The BinarySigningTime attribute MAY be present. The attrType OID for the Binary-SigningTime attribute is 1.2.840.113549.1.9.16.2.46.

```
id-aa-binarySigningTime OBJECT IDENTIFIER ::= { iso(1)
    member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs9(9)
    smime(16) aa(2) 46 }
```

```
BinarySigningTime ::= BinaryTime
```

```
BinaryTime ::= INTEGER (0..MAX)
```

The BinarySigningTime attribute specifies the time, based on the local system clock, when the digital signature was applied to the content. The precise definition of the BinarySigningTime attribute can be found at [RFC6019].

Either one of the SigningTime or the BinarySigningTime attributes, or both attributes, MUST be present. If both the SigningTime and

BinarySigningTime attributes are present they MUST both represent the same underlying time value.

3.1.1.6.5. signatureAlgorithm

The signatureAlgorithm MUST conform to the RPKI Algorithms and Key Size Profile specification [ID.sidr-rpki-algs].

3.1.1.6.6. signature

The signature value is defined as:

SignatureValue ::= OCTET STRING

The signature characteristics are defined by the digest and signature algorithms.

3.1.1.6.7. UnsignedAttributes

unsignedAttrs MUST be omitted.

3.1.2. CMS Object Validation

Before a recipient of a CMS signed object can use the content of the object, the recipient MUST validate the signed object by verifying that all of the following conditions hold. A recipient may perform these checks in any order.

1. The CMS object is well formed, such that the signed object syntax complies with this specification. In particular, that each of the following is true:
 - a. The contentType of the CMS object is SignedData (OID 1.2.840.113549.1.7.2)
 - b. The version of the SignedData object is 3.
 - c. The certificates field in the SignedData object is present and contains one EE certificate, the SubjectKeyIdentifier field of which matches the sid field of the SignerInfo object.
 - d. The crls field in the SignedData object is present.

- e. The version of the `SignerInfo` is 3.
 - f. The `signedAttrs` field in the `SignerInfo` object is present and contains one each of the `ContentType` attribute (OID 1.2.840.113549.1.9.3), the `MessageDigest` attribute (OID 1.2.840.113549.1.9.4), and either or both of a single instance of the `SigningTime` attribute (OID 1.2.840.113549.1.9.5) and the `BinarySigningTime` attribute (OID 1.2.840.113549.1.9.16.2.46), and no other attributes.
 - g. The `eContentType` in the `EncapsulatedContentInfo` is an OID that matches the `attrValues` in the `ContentType` attribute.
 - h. The `unsignedAttrs` field in the `SignerInfo` object is omitted.
 - i. If both the `SigningTime` attribute and the `BinarySigningTime` attribute are present, then their values represent the same time.
 - j. The `digestAlgorithm` in the `SignedData` and `SignerInfo` objects conforms to the RPKI Algorithms and Key Size Profile specification [ID.sidr-rpki-algs].
 - k. The `signatureAlgorithm` in the `SignerInfo` object conforms to the RPKI Algorithms and Key Size Profile specification [ID.sidr-rpki-algs].
 - l. The signed object is DER encoded.
- 2. The public key of the EE certificate (contained within the CMS signed-data object) can be used to successfully verify the signature on the signed object.
 - 3. The EE certificate (contained within the CMS signed-data object) is a valid EE certificate. In particular, there exists a valid certification path from a trust anchor selected by the recipient to this EE certificate.
 - 4. The time represented by the `SigningTime` attribute or the `BinarySigningTime` attribute is greater than or equal to the time value passed in previously valid CMS objects that were passed from the same originator to this recipient.

3.2. Common Message format

The XML template for all messages is as follows:

```
-----  
<?xml version="1.0" encoding="UTF-8"?>  
<message xmlns="http://www.apnic.net/specs/rescerts/up-down/"  
  version="1"  
  sender="sender name"  
  recipient = "recipient name"  
  type="message type">  
  
  [payload]  
  
</message>  
-----
```

version:

the value of this attribute is the version of this protocol. This document describes version 1.

sender:

the value of this attribute is the agreed name of the message sender, as determined between the client and the server by prior arrangement.

recipient:

the value of this attribute is the agreed name of the message recipient, as determined between the client and the server by prior arrangement.

type:

the possible values of this attribute are "list", "list_response", "issue", "issue_response", "revoke", "revoke_response", and "error_response".

Conforming parsers **MUST** reject any document with a version number they do not understand, or with any elements or attributes they do not understand. Servers must generate an error response when receiving such a request. Clients should generate an operator alert error when receiving such a response.

The encapsulated content of the CMS wrapping is an XML document. The remainder of this protocol specification omits this CMS wrapper and only discusses the XML document.

Messages are checked using the following tests:

1. Check that the CMS is well-formed (see test 1 of Section 3.1.2).
2. Check that the XML is well-formed.
3. Check that the XML sender and recipient attributes reference a known client and this server's system respectively.
4. Verify the digital signature using the public key provided in the certificate carried in the CMS wrapper (see test 2 of Section 3.1.2).
5. Validate the CMS-provided certificate using the PKI that has been determined by prior arrangement between client and server (see test 3 of Section 3.1.2).
6. Check that the CMS-signing time is equal to or greater than the signing time provided in the most recent previous message that this recipient has received from this sender (see test 4 of Section 3.1.2).
7. Check that the value of the version number of the message is 1.

These checks SHOULD be applied in the order specified here.

Any errors encountered while checking items 1 through 7 MUST cause a server to generate an "HTTP 400 Bad Data" response to the HTTP POST operation. An error in step 8 MUST cause the server to generate a "Request-Not-Performed" error response. Any errors encountered in these tests by a client SHOULD cause the client to generate an operator alert.

A server MAY perform flow control on the rate of processed requests. Requests not processed due to such a flow control constraint MAY cause the server to generate a "HTTP 503 Service Unavailable" response.

3.3. Control - Resource Class Query

3.3.1. Resource Class List Query

The value of the message "type" message attribute for this query is:

type="list"

Payload:

[No message payload is defined for this query]

3.3.2. Resource Class List Response

The value of the message "type" element for this response is:

type="list_response"

Payload:

```
<class class_name="class name"
  cert_url="url"
  resource_set_as="as resource set"
  resource_set_ipv4="ipv4 resource set"
  resource_set_ipv6="ipv6 resource set"
  resource_set_notafter="datetime"
  suggested_sia_head="[directory uri]" >
  <certificate cert_url="url"
    req_resource_set_as="as resource set"
    req_resource_set_ipv4="ipv4 resource set"
    req_resource_set_ipv6="ipv6 resource set" >
    [certificate]
  </certificate>

  ...

  (repeated for each current certificate where the client
   is the certificate's subject)

  <issuer>[issuer's certificate]</issuer>
</class>

...

(repeated for each of the issuer's resource class where the
client has been allocated resources)
```

Where the client has been allocated resources from multiple resource classes, then the response will contain multiple class elements, corresponding to the complete set of the issuer's resource classes where the client holds allocated resources. Those issuer's resource classes where the client holds no allocated resources will not be included in the response.

Where the issuer has issued multiple certificates in a resource class signed with different keys (as may occur during a staged issuer-key rollover), only the most recent certificate issued with the currently "active" issuer's key will be listed in the response.

Each "class" element describes a set of resources that are certified within the scope of a single certificate, referring to a single resource class with a common validation path.

class_name:

the value of this attribute is the issuer-assigned name of the issuer's Resource Class.

cert_url:

in the context of a class element, the value of this attribute is a pointer to the issuer's CA certificate (i.e. a reference to the immediate superior certificate, being the CA-enabled certificate where the issuer is the certificate's subject). Its value is a comma-separated list of URIs, of which at least one MUST be an RSYNC URI [RFC5781]. Any comma values within a URI MUST be escaped ("%2C"). The ordering of the list may be interpreted by the client as a relative preference for access methods as expressed by the publisher of this certificate.

resource_set_as:

in the context of a class element, the value of this attribute is the set of AS numbers and AS number ranges that the issuer has allocated to the client within the scope of this resource class, presented in ASCII as a comma-separated list. The list elements are decimal integer values and ranges of decimal integers specified by the low and high value of the range with a hyphen delimiter, using the canonical order as described in [RFC3779], without leading zeros, and with no white space or punctuation other than the comma and the hyphen range designator (e.g.: resource_set_as="123,456-789,123456"). If there are no AS numbers in this Resource Class the empty set will be represented by a null string value ("") for this attribute.

resource_set_ipv4:

in the context of a class element, the value of this attribute is the set of IPv4 addresses that the issuer has allocated to the client within the scope of this resource class. The value is presented in ASCII as a comma-separated list of elements. Each element is either an address prefix using the notation of <dotted quad>/mask length, or a range specified as low and high range values in dotted quad notation with a hyphen delimiter. The list is presented in canonical order, as described in [RFC3779]. The dotted quad notation is without leading zeros, and the list contains no white space or punctuation other than the period, forward slash, hyphen and comma. (e.g. resource_set_ipv4="192.0.2.0/26,192.0.2.66-192.0.2.76") If there are no IPv4 addresses in this resource class the empty set will be represented by a null string value ("") for this attribute.

resource_set_ipv6:

in the context of a class element, the value of this attribute is the set of IPv6 addresses that the issuer has allocated to the client within the scope of this resource class. The value is presented in ASCII as a comma-separated list of elements. Each element is either an address prefix using the notation of <hex nibble sequence>/mask length, or a range specified as low and high range values in hex nibble notation with a hyphen delimiter. Trailing zero nibbles are truncated and represented by '::'. The list is presented in canonical order, as described in [RFC3779]. The hex nibble sequence notation is without leading zeros, and the list contains no white space or punctuation other than the colon, forward slash, hyphen and comma (e.g. resource_set_ipv6="2001:0DB8::/48,2001:0DB8:002::-2001:0DB8:005::"). The XML Schema data type is "http://www.w3.org/TR/xmlschema-2/#hexBinary" and value is case insensitive, with the canonical form being upper case. If there are no IPv6 addresses in this resource class the empty set will be represented by a null string value ("") for this attribute.

resource_set_notafter:

The value of this attribute specified the date/time that would be set in the Validity notAfter field in any new certificate issued for this particular client within the scope of this resource class, should the client request a new certificate. The time format used for the value of this attribute is specified as ISO 8601 [ISO.8601:2004], and MUST use UTC time (i.e. YYYY-MM-DDThh:mm:ssZ, e.g. 2007-11-29T04:40:00Z). If the client's certificate has a validity notAfter time that is different to this time then the client SHOULD request a new certificate to be issued for this resource class.

suggested_sia_head: (OPTIONAL)

If this field is present then it's value is a directory URI that indicates a repository publication point that the server has made available to the client to use for the client's collection of published products. This specification does not encompass the protocols that the client may use with the operator of the repository publication point in order to publish objects at this publication point.

[issuer's certificate]

value is the Base64 encoding of the DER-encoded issuer's CA certificate (the CA-enabled certificate where the issuer is the certificate's subject).

Each certificate element describes the most recently issued current certificate where the certificate's subject refers to the client for each active client key pair. A "current" certificate is a non-expired, non-revoked certificate. If no current certificate has been issued, then no certificate element will be included in the response.

cert_url:

in the context of a certificate element, this is a pointer to the location where the certificate issuer has published this certificate. This field is the issuer's suggestion for the AIA field for the subject to use in subordinate certificates that are issued by the subject. According to the Resource Certificate Profile [ID.sidr-res-certs] the AIA field is a non-empty (contains a minimum of 1 element) list of URI's, one of which MUST be an RSYNC URI [RFC5781]. The order of URI's in the AIA field may be interpreted as the publisher's relative preference for access methods for this certificate. The cert_url conforms to this AIA specification. Its value is a comma-separated list of URIs, one of which MUST be an RSYNC URI. Any comma values within a URI MUST be escaped ("%2C").

req_resource_set_as:

the set of AS numbers that were specified in the corresponding original certificate request that defined the maximal requested span of the certified AS number set, following the syntax described above. If this attribute was present in the certificate request, then the attribute MUST be present in this response, otherwise it MUST NOT be present.

req_resource_set_ipv4:

the set of IPv4 addresses that were specified in the corresponding original certificate request that defined the maximal requested span of the certified IPv4 address set, following the syntax described above. If this attribute was present in the certificate

request, then the attribute MUST be present in this response, otherwise it MUST NOT be present.

req_resource_set_ipv6:

the set of IPv6 addresses that were specified in the corresponding original certificate request that defined the maximal requested span of the certified IPv6 address set, following the syntax described above. If this attribute was present in the certificate request, then the attribute MUST be present in this response, otherwise it MUST NOT be present.

[certificate]

value is the Base64 encoding of the DER-encoded certificate.

3.4. CA - Certificate Issuance

3.4.1. Certificate Issuance Request

The value of the message "type" element for this request is:

type="issue"

Payload:

<request

```
class_name="class name"
req_resource_set_as="as resource set"
req_resource_set_ipv4="ipv4 resource set"
req_resource_set_ipv6="ipv6 resource set">
[Certificate request]
</request>
```

The client MUST use different key pairs for each distinct resource class.

If any of the req_resource_set attributes are specified in the request, then any missing req_resource_set attributes are to be interpreted as specifying the complete set of the corresponding resource type that match the client's current resource allocation. If the value of any req_resource_set attributes is the null value (""), then this indicates that no resources of that resource type are to be certified with this request.

The requested resource set values are held as a local record by the issuer against the resource class and the client's public key. Any subsequent Certificate Issuance Requests that specify the same Resource Class and the same client's public key will (re)set the issuer's local record of the requested resource sets to the most recently specified values.

`class_name:`

value is the server's identifier of a Resource Class.

`req_resource_set_as:` (OPTIONAL)

the set of AS numbers that define the maximal requested span of the certified AS number set, formatted as per the `resource_set_as` attribute of the Resource Class List Response.

`req_resource_set_ipv4:` (OPTIONAL)

the set of IPv4 addresses that define the maximal requested span of the certified IPv4 address set, formatted as per the `resource_set_ipv4` attribute of the Resource Class List Response.

`req_resource_set_ipv6:` (OPTIONAL)

the set of IPv6 addresses that define the maximal requested span of the certified IPv6 address set, formatted as per the `resource_set_ipv6` attribute of the Resource Class List Response.

[Certificate request]

value is the certificate request. This is a Base-64 encoded DER version of a request formatted using PKCS#10 [RFC2986]. The certificate request is signed using the private key part of the key pair whose public part is the subject key value in the certification request. The signing algorithm is specified in [ID.sidr-rpki-algs]. (This signature component is intended to demonstrate proof of possession of the private key.)

The response to this request is a Certificate Issuance Response if the request can be processed online. If the request cannot be undertaken immediately then the server MUST response with a Request-Not-Performed message, using the appropriate error code.:

- o If the resource class is not defined by the server, then the server SHOULD return error code 1201.
- o If the client holds no resources in a defined resource class then, the server MUST return error code 1202 and not proceed with the request.

- o If the certificate request payload is badly formed, then the server MUST return error code 1203.
- o If the public key used in the certificate request implies that client is attempting to use identical key pairs for multiple resource classes, then the server MUST respond with a 1204 error code.
- o If the certificate issuer uses an off-line process to undertake certificate issuance, and the server cannot directly respond to the certificate issuance request with an issued certificate, then the certificate issuer MUST respond to the first instance of this request with an error code 1104 to indicate that the request is being processed asynchronously. Subsequent repetitions of this request while the off-line actions are being undertaken SHOULD cause a response with error code 1101. In this context, where off-line processes are invoked for certificate issuance, if the certificate issuer determines in processing the request that the issued certificate would be identical in all respects to the most recently issued certificate for this client, other than the certificate's serial number, were the certificate to be issued, the issuer may choose to respond with the most recently issued certificate and not initiate an off-line certificate issuance request.

It is noted that a client, when receiving a 1104 response to a certificate issuance request MAY periodically resubmit the request, in which case the client will receive error code 1101 response while the request is being processed, and a Certificate Issuance Response when the certificate issuance process has completed. In such circumstances a client SHOULD limit the frequency of such repeated requests to no more than 1 request in each 24 hour interval.

3.4.2. Certificate Issuance Response

The value of the message "type" element for this response is:

```
type="issue_response"
```

Payload:

```
<class class_name="class name"
  cert_url="url"
  resource_set_as="as resource set"
  resource_set_ipv4="ipv4 resource set"
  resource_set_ipv6="ipv6 resource set" >
  <certificate cert_url="url"
    req_resource_set_as="as resource set"
    req_resource_set_ipv4="ipv4 resource set"
    req_resource_set_ipv6="ipv6 resource set" >
    [certificate]
  </certificate>
  <issuer>[issuer's certificate]</issuer>
</class>
```

If the certificate issuer determines that the issued certificate would be identical in all respects to the most recently issued certificate for this client, other than the certificate's serial number, were the certificate to be issued, the issuer may choose to respond with the most recently issued certificate and not issue a new certificate for this request.

The definition of the attributes and syntax of the values is the same as the resource class list response, but the response only references the (single) named resource class, and the (single) certificate issued against the client's public key as provided in the corresponding certificate request.

3.5. Certificate Revocation

3.5.1. Certificate Revocation Request

The value of the message "type" element for this request is:

```
type="revoke"
```

Payload:

```
<key class_name="class name"
    ski="[encoded hash of the subject public key]" />
```

This command 'retires' a client's key pair by requesting the issuer to revoke all certificates for this client that contain the matching public key, within the scope of a named Resource Class. Individual issued certificates cannot be revoked within the scope of this protocol.

This command directs the issuer to immediately mark all issued valid certificates issued by this issuer within the named Resource Class with this client's SKI value to be marked as revoked, causing the issued certificates to be withdrawn from the publication repository and to be listed in the server's subsequent CRLs within this Resource Class.

class_name:
value is the issuer-assigned name of the issuer's Resource Class.

ski:
value is the encoded hash of the client's public key that is to be revoked. The algorithm for the encoding is to generate the 160-bit SHA-1 hash of the client's public key, as defined in method (1) of section 4.2.1.2 of [RFC5280], and encode this value using the Base 64 encoding with URL and Filename Safe Alphabet, as defined in section 5 of [RFC4648].

3.5.2. Certificate Revocation Response

The value of the message "type" element for this response is:

```
type="revoke_response"
```

Payload:

```
<key class_name="class name"
    ski="[encoded hash of the subject public key]" />
```

class_name:
value is the issuer-assigned name of the server's Resource Class.

ski:
value is the encoded hash of the client's public key that is to be revoked. The algorithm for the encoding is to generate the 160-bit SHA-1 hash of the client's public key, as defined in method (1) of section 4.2.1.2 of [RFC5280], and encode this value using the Base 64 encoding with URL and Filename Safe Alphabet, as defined in section 5 of [RFC4648].

3.6. Request-Not-Performed Response

The value of the message "type" element for this response is:

```
type="error_response"
```

Payload:

```
<status>[Code]</status>
<description xml:lang="en-US">[Readable text]</description>
```

All states where an error response is to be generated, either due to detected errors or inconsistencies in the content of the request or server-side states that prevent the request being performed, generate a Request-Not-Performed response.

description:

value is a text field. This element MAY be present. It's value has no defined meaning within the scope of this protocol, and implementations may assume that some form of human-readable text may be used here. If the HTTP request that triggered this error response includes an Accept-Language header as defined in section 14.4 of the HTTP/1.1 specification [RFC2616] then the server will make a best effort to include a second description element using the highest ranked preferred language of the client. The en-US description will always be included if the element is present.

The error code set is:

Code Value	Description
1101	already processing request
1102	version number error
1103	unrecognised request type
1104	request scheduled for processing
1201	request - no such resource class
1202	request - no resources allocated in resource class
1203	request - badly formed certificate request
1204	request - already used key in request
1301	revoke - no such resource class
1302	revoke - no such key
2001	Internal Server Error - Request not performed

4. Security Considerations

The intent of this protocol is to define a protocol to support the maintenance of Resource Certificates that the Issuer issues for a Subject in certifying resources that have been allocated or assigned by the Issuer to the Subject [ID.sidr-arch]. This protocol assumes that the Issuer and Subject are known to each other and have exchanged credentials so as to support the mutual recognition of the digital signatures used to sign the CMS messages. The mechanisms used to perform the associated credential exchange are not described in this specification.

The protocol is a minimal query / response protocol, that imposes strict serialization on each query / response transaction, reducing the potential for the Subject and the Issuer to lose synchronization over the issued certificate state.

The inner protocol elements explicitly reference the intended sender and receiver to present an Issuer or an Subject attempting to masquerade as another party within the secure channel.

5. IANA Considerations

[Note to IANA, to be removed prior to publication: there are no IANA considerations stated in this version of the document.]

6. Acknowledgements

The authors would like to acknowledge the valued contributions from Russ Housley, Steve Kent, Randy Bush, George Michaelson, Robert Kisteleki and Tim Bruijnzeels in the preparation of the protocol described in this document.

7. References

7.1. Normative References

- [ID.sidr-rpki-algs]
Huston, G., "A Profile for Algorithms and Key Sizes for use in the Resource Public Key Infrastructure", draft-huston-sidr-rpki-algs-00.txt (work in progress), July 2009.
- [ISO.8601:2004]
ISO, "ISO 8601:2004 Representation of dates and Times", 2004.
- [RFC2616] Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., and T. Berners-Lee, "Hypertext Transfer Protocol -- HTTP/1.1", RFC 2616, June 1999.
- [RFC2986] Nystrom, M. and B. Kaliski, "PKCS #10: Certification Request Syntax Specification Version 1.7", RFC 2986, November 2000.
- [RFC3779] Lynn, C., Kent, S., and K. Seo, "X.509 Extensions for IP Addresses and AS Identifiers", RFC 3779, June 2004.
- [RFC4648] Josefsson, S., "The Base16, Base32, and Base64 Data Encodings", RFC 4648, October 2006.
- [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, May 2008.
- [RFC5652] Housley, R., "Cryptographic Message Syntax (CMS)",

RFC 5652, September 2009.

[RFC5781] Weiler, S., Ward, D., and R. Housley, "The rsync URI Scheme", RFC 5781, February 2010.

[RFC6019] Housley, R., "BinaryTime: An Alternate Format for Representing Date and Time in ASN.1", RFC 6019, September 2010.

[X.509-88] CCITT, "Recommendation X.509: The Directory - Authentication Framework", 1988.

7.2. Informative References

[ID.sidr-arch] Lepinski, M. and S. Kent, "An Infrastructure to Support Secure Internet Routing", draft-ietf-sidr-arch (work in progress), July 2009.

[ID.sidr-res-certs] Huston, G., Michaelson, G., and R. Loomans, "A Profile for X.509 PKIX Resource Certificates", Work in progress: Internet Drafts draft-ietf-sidr-res-certs-16.txt, February 2009.

Appendix A. CMS Signed Object

The following is the ASN.1 specification of the CMS signed object used by the RPKI provisioning protocol.

```
ContentInfo ::= SEQUENCE {
    contentType ContentType,
    content [0] EXPLICIT ANY DEFINED BY contentType }

ContentType ::= OBJECT IDENTIFIER

id-smime OBJECT IDENTIFIER ::= { iso(1) member-body(2) us(840)
    rsadsi(113549) pkcs(1) pkcs9(9) 16 }

id-ct OBJECT IDENTIFIER ::= { id-smime 1 }

id-ct-xml OBJECT IDENTIFIER ::= { id-ct 28 }

id-ct-xml ::= OCTET STRING -- XML encoded message

id-signedData OBJECT IDENTIFIER ::= { iso(1) member-body(2)
```


us(840) rsadsi(113549) pkcs(1) pkcs7(7) 2 }

SignedData ::= SEQUENCE {
 version CMSVersion,
 digestAlgorithms DigestAlgorithmIdentifiers,
 encapContentInfo EncapsulatedContentInfo,
 certificates [0] IMPLICIT CertificateSet OPTIONAL,
 crls [1] IMPLICIT RevocationInfoChoices OPTIONAL,
 signerInfos SignerInfos }

DigestAlgorithmIdentifiers ::= SET OF DigestAlgorithmIdentifier

SignerInfos ::= SET OF SignerInfo

SignerInfo ::= SEQUENCE {
 version CMSVersion,
 sid SignerIdentifier,
 digestAlgorithm DigestAlgorithmIdentifier,
 signedAttrs [0] IMPLICIT SignedAttributes OPTIONAL,
 signatureAlgorithm SignatureAlgorithmIdentifier,
 signature SignatureValue,
 unsignedAttrs [1] IMPLICIT UnsignedAttributes OPTIONAL }

SignerIdentifier ::= CHOICE {
 issuerAndSerialNumber IssuerAndSerialNumber,
 subjectKeyIdentifier [0] SubjectKeyIdentifier }

SignedAttributes ::= SET SIZE (1..MAX) OF Attribute

Attribute ::= SEQUENCE {
 attrType OBJECT IDENTIFIER,
 attrValues SET OF AttributeValue }

AttributeValue ::= ANY

SignatureValue ::= OCTET STRING

id-contentType OBJECT IDENTIFIER ::= { iso(1) member-body(2)
 us(840) rsadsi(113549) pkcs(1) pkcs9(9) 3 }

ContentType ::= OBJECT IDENTIFIER

id-messageDigest OBJECT IDENTIFIER ::= { iso(1) member-body(2)
 us(840) rsadsi(113549) pkcs(1) pkcs9(9) 4 }

MessageDigest ::= OCTET STRING

id-signingTime OBJECT IDENTIFIER ::= { iso(1) member-body(2)

```

        us(840) rsadsi(113549) pkcs(1) pkcs9(9) 5 }

SigningTime ::= Time

Time ::= CHOICE {
    utcTime UTCTime,
    generalizedTime GeneralizedTime }

id-aa-binarySigningTime OBJECT IDENTIFIER ::= { iso(1)
    member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs9(9)
    smime(16) aa(2) 46 }

BinarySigningTime ::= BinaryTime

BinaryTime ::= INTEGER (0..MAX)

```

Appendix B. XML Schema

The following is a RelaxNG compact form schema describing the Issuer-Subject Protocol, version 1.

Note: "The namespace name, to serve its intended purpose, SHOULD have the characteristics of uniqueness and persistence. It is not a goal that it be directly usable for retrieval of a schema (if any exists)." ["Namespaces in XML 1.0 (Third Edition)", W3C Recommendation 8 December 2009, <http://www.w3.org/TR/2009/REC-xml-names-20091208/>]

default namespace = "http://www.apnic.net/specs/rescerts/up-down/"

```

grammar {
  start = element message {
    attribute version { xsd:positiveInteger { maxInclusive="1" } },
    attribute sender { xsd:token { maxLength="1024" } },
    attribute recipient { xsd:token { maxLength="1024" } },
    payload
  }

  payload |= attribute type { "list" }, list_request
  payload |= attribute type { "list_response" }, list_response
  payload |= attribute type { "issue" }, issue_request
  payload |= attribute type { "issue_response" }, issue_response
  payload |= attribute type { "revoke" }, revoke_request
  payload |= attribute type { "revoke_response" }, revoke_response
  payload |= attribute type { "error_response" }, error_response

```

```

list_request = empty
list_response = class*

class = element class {
  attribute class_name { xsd:token { maxLength="1024" } },
  attribute cert_url { xsd:string { maxLength="4096" } },
  attribute resource_set_as { xsd:string { maxLength="512000"
    pattern="[\-,0-9]*" } },
  attribute resource_set_ipv4 { xsd:string { maxLength="512000"
    pattern="[\-,/.0-9]*" } },
  attribute resource_set_ipv6 { xsd:string { maxLength="512000"
    pattern="[\-,/:0-9a-fA-F]*" } },
  attribute resource_set_notafter { xsd:dateTime },
  attribute suggested_sia_head { xsd:anyURI { maxLength="1024"
    pattern="rsync://.+" } }?,
  element certificate {
    attribute cert_url { xsd:string { maxLength="4096" } },
    attribute req_resource_set_as { xsd:string {
      maxLength="512000" pattern="[\-,0-9]*" } }?,
    attribute req_resource_set_ipv4 { xsd:string {
      maxLength="512000" pattern="[\-,/.0-9]*" } }?,
    attribute req_resource_set_ipv6 { xsd:string {
      maxLength="512000" pattern="[\-,/:0-9a-fA-F]*" } }?,
    xsd:base64Binary { maxLength="512000" }
  }*,
  element issuer { xsd:base64Binary { maxLength="512000" } }
}

issue_request = element request {
  attribute class_name { xsd:token { maxLength="1024" } },
  attribute req_resource_set_as { xsd:string {
    maxLength="512000" pattern="[\-,0-9]*" } }?,
  attribute req_resource_set_ipv4 { xsd:string {
    maxLength="512000" pattern="[\-,/.0-9]*" } }?,
  attribute req_resource_set_ipv6 { xsd:string {
    maxLength="512000" pattern="[\-,/:0-9a-fA-F]*" } }?,
  xsd:base64Binary { maxLength="512000"
  }
}
issue_response = class

revoke_request = revocation

revoke_response =
  revocation

revocation = element key { attribute class_name { xsd:token {
  maxLength="1024" } }, attribute ski {

```

```
        xsd:token { maxLength="1024" } }
    }

    error_response =
      element status { xsd:positiveInteger {
        maxInclusive="9999999999999999" }
      },
      element description { attribute xml:lang { xsd:language },
        xsd:string { maxLength="1024" }
      }?
    }
```

Authors' Addresses

Geoff Huston
APNIC

Email: gih@apnic.net
URI: <http://www.apnic.net>

Robert Loomans
APNIC

Email: robertl@apnic.net
URI: <http://www.apnic.net>

Byron Ellacott
APNIC

Email: bje@apnic.net
URI: <http://www.apnic.net>

Rob Austein
Internet Systems Consortium

Email: sra@isc.org

SIDR
Internet-Draft
Intended status: Standards Track
Expires: May 13, 2011

G. Huston
APNIC
November 9, 2010

A Profile for Algorithms and Key Sizes for use in the Resource Public
Key Infrastructure
draft-ietf-sidr-rpki-algs-04.txt

Abstract

This document specifies the algorithms, algorithms' parameters, asymmetric key formats, asymmetric key size and signature format for the Resource Public Key Infrastructure subscribers that generate digital signatures on certificates, Certificate Revocation Lists, and signed objects as well as for the Relying Parties (RPs) that verify these digital signatures.

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 <http://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 May 13, 2011.

Copyright Notice

Copyright (c) 2010 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of

the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

1. Introduction

This document specifies:

- * the digital signature algorithm and parameters;
- * the hash algorithm and parameters;
- * the public and private key formats; and,
- * the signature format

used by Resource Public Key Infrastructure (RPKI) subscribers when they apply digital signatures to certificates, Certificate Revocation Lists (CRLs), and signed objects (e.g., Route Origin Authorizations (ROAs) and manifests). Relying Parties (RPs) also use this document when verify RPKI subscribers' digital signatures [ID.ietf-sidr-arch].

This document is referenced by other RPKI profiles and specifications, including the RPKI Certificate Policy (CP) [ID.ietf-sidr-cp], the RPKI Certificate Profile [ID.ietf-sidr-res-certs], the SIDR architecture [ID.ietf-sidr-arch], and the signed object template for the RPKI [ID.ietf-sidr-signed-object]. Familiarity with these documents is assumed.

1.1. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC2119.

2. Algorithms

Two cryptographic algorithms are used in the RPKI:

- * The signature algorithm used in certificates, CRLs, and signed objects is RSA Public-Key Cryptography Standards (PKCS) #1 Version 1.5 (sometimes referred to as "RSASSA-PKCS1-v1_5") from Section 5 of [RFC4055].
- * The hashing algorithm used in certificates, CRLs, and signed objects is SHA-256 [SHS]. Hash algorithms are not identified by themselves in certificates and CRLs instead they are combined with the digital signature algorithm (see below). When used in the Cryptographic Message Syntax (CMS) SignedData, the hash algorithm (in this case, the hash algorithm is

sometimes called a message digest algorithm) is identified by itself. For CMS SignedData, the object identifier and parameters for SHA-256 in [RFC5754] MUST be used when populating the digestAlgorithms and digestAlgorithm fields.

NOTE: The exception to the above hashing algorithm is the use of SHA-1 [SHS] when CAs generate authority and subject key identifiers [ID.ietf-sidr-res-certs].

When used to generate and verify digital signatures the hash and digital signature algorithms are referred together, i.e., "RSA PKCS#1 v1.5 with SHA-256" or more simply "RSA with SHA-256". The Object Identifier (OID) sha256withRSAEncryption from [RFC4055] MUST be used.

Locations for this OID are as follows:

In the certificate, the OID appears in the signature and signatureAlgorithm fields [RFC4055];- In the CRL, the OID appears in the signatureAlgorithm field [RFC4055]; and,- In CMS SignedData, the OID appears in each SignerInfo signatureAlgorithm field [RFC3370] using the OID from above.

3. Asymmetric Key Pair Formats

The RSA key pairs used to compute the signatures MUST have a 2048-bit modulus and a public exponent (e) of 65,537.

3.1. Public Key Format

The Subject's public key is included in subjectPublicKeyInfo [RFC5280]. It has two sub-fields: algorithm and subjectPublicKey. The values for the structures and their sub-structures follow:

algorithm (which is an AlgorithmIdentifier type):

The object identifier for RSA PKCS#1 v1.5 with SHA-256 MUST be used in the algorithm field, as specified in Section 5 of [RFC4055]. The value for the associated parameters from that clause MUST also be used for the parameters field.

subjectPublicKey:

RSAPublicKey MUST be used to encode the certificate's subjectPublicKey field, as specified in [RFC4055].

3.2. Private Key Format

Local Policy determines private key format.

4. Signature Format

The structure for the certificate's signature field is as specified in Section 1.2 of [RFC4055]. The structure for the Cryptographic Message Syntax (CMS) SignedData's signature field is as specified in [RFC3370].

5. Additional Requirements

It is anticipated that the RPKI will require the adoption of updated key sizes and a different set of signature and hash algorithms over time, in order to maintain an acceptable level of cryptographic security to protect the integrity of signed products in the RPKI. This profile should be updated to specify such future requirements, as and when appropriate.

CAs and RPs SHOULD be capable of supporting a transition to allow for the phased introduction of additional encryption algorithms and key specifications, and also accommodate the orderly deprecation of previously specified algorithms and keys. Accordingly, CAs and RPs SHOULD be capable of supporting multiple RPKI algorithm and key profiles simultaneously within the scope of such anticipated transitions. The recommended procedures to implement such a transition of key sizes and algorithms is not specified in this document.

6. Security Considerations

The Security Considerations of [RFC4055], [RFC5280], and [ID.ietf-sidr-res-certs] apply to certificate and CRLs. The Security Considerations of [RFC5754] apply to signed objects. No new security are introduced as a result of this specification.

7. IANA Considerations

[There are no IANA considerations in this document.]

8. Acknowledgments

The author acknowledges the re-use in this draft of material originally contained in working drafts the RPKI Certificate Policy and Resource Certificate profile documents. The co-authors of these two documents, namely Stephen Kent, Derrick Kong, Karen Seo, Ronald Watro, George Michaelson and Robert Loomans, are acknowledged, with thanks. The constraint on key size noted in this profile is the outcome of comments from Stephen Kent and review comments from David Cooper. Sean Turner has provided additional review input to this document.

9. Normative References

[ID.ietf-sidr-arch]

Lepinski, M. and S. Kent, "An Infrastructure to Support Secure Internet Routing", draft-ietf-sidr-arch (work in progress), September 2010.

[ID.ietf-sidr-cp]

Kent, S., Kong, D., Seo, K., and R. Watro, "Certificate Policy (CP) for the Resource PKI (RPKI)", draft-ietf-sidr-cp (work in progress), September 2010.

[ID.ietf-sidr-res-certs]

Husotn, G., Michaelson, G., and R. Loomans, "A Profile for X.509 PKIX Resource Certificates", draft-ietf-sidr-res-certs (work in progress), May 2008.

[ID.ietf-sidr-signed-object]

Lepinski, M., Chi, A., and S. Kent, "Signed Object Template for the Resource Public Key Infrastructure", draft-ietf-sidr-signed-object-01.txt (work in progress), October 2010.

[RFC3370] Housley, R., "Cryptographic Message Syntax (CMS) Algorithms", RFC 3370, August 2002.

[RFC4055] Schaad, J., Kaliski, B., and R. Housley, "Additional Algorithms and Identifiers for RSA Cryptography for use in the Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 4055, June 2005.

[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, May 2008.

[RFC5754] Turner, S., "Using SHA2 Algorithms with Cryptographic Message Syntax", RFC 5754, January 2010.

[SHS] National Institute of Standards and Technology (NIST), "FIPS Publication 180-3: Secure Hash Standard", FIPS Publication 180-3, October 2008.

Author's Address

Geoff Huston
APNIC

Email: gih@apnic.net

Secure Inter-Domain Routing
Internet-Draft
Intended status: Standards Track
Expires: May 13, 2011

R. Austein
ISC
G. Huston
APNIC
S. Kent
M. Lepinski
BBN
November 9, 2010

Manifests for the Resource Public Key Infrastructure
draft-ietf-sidr-rpki-manifests-09.txt

Abstract

This document defines a "manifest" for use in the Resource Public Key Infrastructure (RPKI). A manifest is a signed object (file) that contains a listing of all the signed objects (files) in the repository publication point (directory) associated with an authority responsible for publishing in the repository. For each certificate, Certificate Revocation List (CRL), or other type of signed objects issued by the authority, that are published at this repository publication point, the manifest contains both the name of the file containing the object, and a hash of the file content. Manifests are intended to enable a relying party (RP) to detect certain forms of attacks against a repository. Specifically, if an RP checks a manifest's contents against the signed objects retrieved from a repository publication point, then the RP can detect "stale" (valid) data and deletion of signed objects.

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 <http://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 May 13, 2011.

Copyright Notice

Copyright (c) 2010 IETF Trust and the persons identified as the document authors. All rights reserved.

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

Table of Contents

1. Introduction	3
1.1. Terminology	3
2. Manifest Scope	4
3. Manifest Signing	4
4. Manifest Definition	5
4.1. eContentType	5
4.2. eContent	5
4.2.1. Manifest	5
4.3. ContentType Attribute	7
4.4. Manifest Validation	7
5. Manifest Generation	7
5.1. Manifest Generation Procedure	7
5.2. Considerations for Manifest Generation	9
6. Relying Party Use of Manifests	9
6.1. Tests for Determining Manifest State	10
6.2. Missing Manifests	11
6.3. Invalid Manifests	12
6.4. Stale Manifests	12
6.5. Mismatch between Manifest and Publication Point	13
6.6. Hash Values Not Matching Manifests	14
7. Publication Repositories	15
8. Security Considerations	15
9. IANA Considerations	16
10. Acknowledgements	16
11. References	16
11.1. Normative References	16
11.2. Informative References	17
Appendix A. ASN.1 Module	18
Authors' Addresses	18

1. Introduction

The Resource Public Key Infrastructure (RPKI) [ID.ietf-sidr-arch] makes use of a distributed repository system [ID.ietf-sidr-repos-struct] to make available a variety of objects needed by relying parties (RPs). Because all of the objects stored in the repository system are digitally signed by the entities that created them, attacks that modify these published objects are detectable by RPs. However, digital signatures provide no protection against attacks that substitute "stale" versions of signed objects (i.e., objects that were valid and have not expired, but have since been superseded) or attacks that remove an object that should be present in the repository. To assist in the detection of such attacks, the RPKI repository system can make use of a signed object called a "manifest".

A manifest is a signed object that enumerates all the signed objects (files) in the repository publication point (directory) that are associated with an authority responsible for publishing at that publication point. Each manifest contains both the name of the file containing the object, and a hash of the file content, for every signed object issued by an authority that is published at the authority's repository publication point. A manifest is intended to allow an RP to detect unauthorized object removal, or the substitution of "stale" versions of objects at a publication point. A manifest also intended to allow an RP to detect similar outcomes that may result from a man-in-the middle attack on the retrieval of objects from the repository. Manifests are intended to be used in Certification Authority (CA) publication points in repositories (directories containing files that are subordinate certificates and Certificate Revocation Lists (CRLs) issued by this CA and other signed objects that are verified by EE certificates issued by this CA).

Manifests are modeled on CRLs, as the issues involved in detecting stale manifests, and detection of potential attacks using manifest replays, etc are similar to those for CRLs. The syntax of the manifest payload differs from CRLs, since RPKI repositories contain objects not covered by CRLs, e.g., digitally signed objects, such as Route Origination Authorizations (ROAs).

1.1. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119.

2. Manifest Scope

A manifest associated with a CA's repository publication point contains a list of:

- * the set of (non-expired, non-revoked) certificates issued and published by this CA,
- * the most recent CRL issued by this CA, and
- * all published signed objects that are verifiable using EE certificates [I-D.sidr-res-certs], issued by this CA.

Where an EE certificate is placed in the Cryptographic Message Syntax (CMS) wrapper of a published RPKI signed object [ID.sidr-signed-object] there is no requirement to separately publish the EE certificate in the CA's repository publication point.

Where multiple CA instances share a common publication point, as can occur when an entity performs a key-rollover operation [ID.sidr-keyroll], the repository publication point will contain multiple manifests. In this case, each manifest describes only the collection of published products of its associated CA instance.

3. Manifest Signing

A CA's manifest is verified using an EE certificate. The SIA field of this EE certificate contains the access method OID of id-ad-signedObject.

The CA MAY choose to sign only one manifest with each generated private key, and generate a new key pair for each new version of the manifest. This form of use of the associated EE certificate is termed a "one-time-use" EE certificate.

Alternatively, the CA MAY elect to use the same private key to sign a sequence of manifests. Because only a single manifest (issued under a single CA instance) is current at any point in time, the associated EE certificate is used to verify only a single object at a time. As long as the sequence of objects verified by this EE certificate are published using the same file name, then this sequential, multiple use of this EE certificate is also valid. This form of use of a EE certificate is termed a "sequential-use" EE certificate.

4. Manifest Definition

A manifest is an RPKI signed object, as specified in [ID.sidr-signed-object]. The RPKI signed object template requires specification of the following data elements in the context of the manifest structure.

4.1. eContentType

The eContentType for a Manifest is defined as id-ct-rpkiManifest, and has the numerical value of 1.2.840.113549.1.9.16.1.26.

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

```
id-ct OBJECT IDENTIFIER ::= { id-smime 1 }
```

```
id-ct-rpkiManifest OBJECT IDENTIFIER ::= { id-ct 26 }
```

4.2. eContent

The content of a Manifest is defined as follows:

```
Manifest ::= SEQUENCE {
    version          [0] INTEGER DEFAULT 0,
    manifestNumber   INTEGER (0..MAX),
    thisUpdate       GeneralizedTime,
    nextUpdate       GeneralizedTime,
    fileHashAlg      OBJECT IDENTIFIER,
    fileList         SEQUENCE SIZE (0..MAX) OF FileAndHash
}
```

```
FileAndHash ::= SEQUENCE {
    file            IA5String,
    hash            BIT STRING
}
```

4.2.1. Manifest

The manifestNumber, thisUpdate, and nextUpdate fields are modelled after the corresponding fields in X.509 CRLs (see [RFC5280]). Analogous to CRLs, a manifest is nominally current until the time specified in nextUpdate or until a manifest is issued with a greater manifest number, whichever comes first.

If a "one-time-use" EE certificate is employed to verify a manifest, the EE certificate MUST have an validity period that coincides with the interval from thisUpdate to nextUpdate, to prevent needless

growth of the CA's CRL.

If a "sequential-use" EE certificate is employed to verify a manifest, the EE certificate's validity period needs to be no shorter than the nextUpdate time of the current manifest. The extended validity time raises the possibility of a substitution attack using a stale manifest, as described in Section 6.4.

The data elements of the Manifest structure are defined as follows:

version:

The version number of this version of the manifest specification MUST be 0.

manifestNumber:

This field is an integer that is incremented each time a new manifest is issued for a given publication point. This field allows an RP to detect gaps in a sequence of published manifest.

As the manifest is modelled on the CRL specification, the ManifestNumber is analogous to the CRLNumber, and the guidance in [RFC5280] for CRLNumber values is appropriate as to the range of number values that can be used for the manifestNumber. Manifest numbers can be expected to contain long integers. Manifest verifiers MUST be able to handle number values up to 20 octets. Conforming Manifest issuers MUST NOT use number values longer than 20 octets

thisUpdate:

This field contains the time when the manifest was created. This field has the same format constraints as specified in [RFC5280] for the CRL field of the same name.

nextUpdate:

This field contains the time at which the next scheduled manifest will be issued. The value of nextUpdate MUST be later than the value of thisUpdate. The specification of the GeneralizedTime value is the same as required for the thisUpdate field.

If the authority alters any of the items that it has published in the repository publication point, then the authority MUST issue a new manifest before the nextUpdate time. If a manifest encompasses a CRL, the nextUpdate field of the manifest MUST match that of the CRL's nextUpdate field, as the manifest will be re-issued when a new CRL is published. If a "one-time-use" EE certificate is used to verify the manifest, then when a new manifest is issued before the time specified in nextUpdate of the current manifest, the CA MUST also issue a new CRL that includes

the EE certificate corresponding to the old manifest.

fileHashAlg:

This field contains the OID of the hash algorithm used to hash the files that the authority has placed into the repository. The hash algorithm used MUST conform to the RPKI Algorithms and Key Size Profile specification [ID.ietf-sidr-rpki-algs].

fileList:

This field is a sequence of FileAndHash objects. There is one FileAndHash entry for each currently valid signed object that has been published by the authority (at this publication point). Each FileAndHash is an ordered pair consisting of the name of the file in the repository publication point (directory) that contains the object in question, and a hash of the file's contents.

4.3. ContentType Attribute

The mandatory Content-Type Attribute MUST have its attrValues field set to the same OID as eContentType. This OID is id-ct-rpkiManifest, and has the numerical value of 1.2.840.113549.1.9.16.1.26.

4.4. Manifest Validation

To determine whether a manifest is valid, the RP MUST perform the following checks in addition to those specified in [ID.sidr-signed-object]:

1. The eContentType in the EncapsulatedContentInfo is id-ad-rpkiManifest (OID 1.2.840.113549.1.9.16.1.26).
2. The version of the rpkiManifest is 0.
3. In the rpkiManifest, thisUpdate precedes nextUpdate.

If the above procedure indicates that the manifest is invalid, then the manifest MUST be discarded and treated as though no manifest were present.

5. Manifest Generation

5.1. Manifest Generation Procedure

For a CA publication point in the RPKI repository system, a CA MUST perform the following steps to generate a manifest:

1. If no key pair exists, or if using a "one-time-use" EE certificate with a new key pair, generate a key pair.
2. If using a "one-time-use" EE certificate, or if a key pair was generated in step 1, issue a EE certificate for this key pair.

This EE certificate MUST have an SIA extension access description field with an accessMethod OID value of id-ad-signedobject where the associated accessLocation references the publication point of the manifest as an object URL.

This EE certificate MUST describe its Internet Number Resources (INRs) using the "inherit" attribute, rather than explicit description of a resource set (see [RFC3779]).

In the case of a "one-time-use" EE certificate, the validity times of the EE certificate MUST exactly match the thisUpdate and nextUpdate times of the manifest.

In the case of a "sequential-use" EE certificate the validity times of the EE certificate MUST encompass the time interval from thisUpdate to nextUpdate.

3. The EE certificate MUST NOT be published in the authority's repository publication point.
4. Construct the manifest content.

The manifest content is described in Section 4.2.1. The manifest's fileList includes the file name and hash pair for each object issued by this CA that has been published at this repository publication point (directory). The collection of objects to be included in the manifest includes all certificates issued by this CA that are published at the CA's repository publication point, the most recent CRL issued by the CA, and all objects verified by EE certificates that were issued by this CA that are published at this repository publication point.

Note that the manifest does not include a self reference (i.e., its own file name and hash), since it would be impossible to compute the hash of the manifest itself prior to it being signed.

5. Encapsulate the manifest content using the CMS SignedData content type (as specified Section 4), sign the manifest using the private key corresponding to the subject key contained in the EE certificate, and publish the manifest in repository system publication point that is described by the manifest.

6. In the case of a key pair that is to be used only once, in conjunction with a "one-time-use" EE certificate, the private key associated with this key pair SHOULD now be destroyed.

5.2. Considerations for Manifest Generation

A new manifest MUST be issued on or before the nextUpdate time.

An authority MUST issue a new manifest in conjunction with the finalization of changes made to objects in the publication point. An authority MAY perform a number of object operations on a publication repository within the scope of a repository change before issuing a single manifest that covers all the operations within the scope of this change. Repository operators SHOULD implement some form of repository update procedure that mitigates, to the extent possible, the risk that RPs who are performing retrieval operations on the repository are exposed to inconsistent transient intermediate states during updates to the repository publication point (directory) and the associated manifest.

Since the manifest object URL is included in the SIA of issued Certificates, a new manifest MUST NOT invalidate the manifest object URL of previously issued certificates. This implies that the manifest's publication name in the repository, in the form of an object URL, is unchanged across manifest generation cycles.

When a CA entity is performing a key rollover, the entity MAY choose to have two CAs instances simultaneously publishing into the same repository publication point. In this case there will be one manifest associated with each active CA instance that is publishing into the common repository publication point (directory).

6. Relying Party Use of Manifests

The goal of an RP is to determine which signed objects to use for validating assertions about INRs and their use (e.g., which ROAs to use in the construction of route filters). Ultimately, this selection is a matter of local policy. However, in the following sections, we describe a sequence of tests that the RP SHOULD perform to determine the manifest state of the given publication point. We then discuss the risks associated with using signed objects in the publication point, given the manifest state; we also provide suitable warning text that SHOULD be placed in a user-accessible log file. It is the responsibility of the RP to weigh these risks against the risk of routing failure that could occur if valid data is rejected, and to implement a suitable local policy. Note that if a certificate is

deemed unfit for use due to local policy, then any signed object that is validated using this certificate also SHOULD be deemed unfit for use (regardless of the status of the manifest at its own publication point).

6.1. Tests for Determining Manifest State

For a given publication point, the RP SHOULD perform the following tests to determine the manifest state of the publication point:

1. For each CA using this publication point, select the CA's current manifest (The "current" manifest is the manifest issued by this CA having highest manifestNumber among all valid manifests, and where manifest validity is defined in Section 4.4).

If the publication point does not contain a valid manifest, see Section 6.2. Lacking a valid manifest, the following tests cannot be performed.

2. To verify completeness, an RP MAY check that every file at each publication point appears in one and only one current manifest, and that every file listed in a current manifest that is published at the same publication point as the manifest.

If there exist files at the publication point that do not appear on any manifest, or files listed in a manifest that do not appear at the publication point then see Section 6.5, but still continue with the following test.

3. Check that the current time (translated to UTC) is between thisUpdate and nextUpdate.

If the current time does not lie within this interval then see Section 6.4, but still continue with the following tests.

4. Verify that listed hash value of every file listed in each manifest matches the value obtained by hashing the file at the publication point.

If the computed hash value of a file listed on the manifest does not match the hash value contained in the manifest, then see Section 6.6.

5. An RP MAY check that the contents of each current manifest conforms to the manifest's scope constraints, as specified in Section 2.

If a current manifest contains entries for objects that are not

within the scope of the manifest, then the out-of-scope entries SHOULD be disregarded in the context of this manifest. If there is no other current manifest that describes these objects within that other manifest's scope, then see Section 6.2.

For each signed object, if all of the following conditions hold:

- * the manifest for its publication, and the associated publication point, pass all of the above checks;
- * the signed object is valid; and
- * the manifests for every certificate on the certification path used to validate the signed object, and the associated publication points, pass all of the above checks;

then the RP can conclude that no attack against the repository system has compromised the given signed object, and the signed object MUST be treated as valid.

6.2. Missing Manifests

The absence of a current manifest at a publication point could occur due to an error by the publisher or due to (malicious or accidental) deletion or corruption of all valid manifests.

When no valid manifest is available, there is no protection against attacks that delete signed objects or replay old versions of signed objects. All signed objects at the publication point, and all descendant objects that are validated using a certificate at this publication point SHOULD be viewed as suspect, but MAY be used by the RP, as per local policy.

The primary risk in using signed objects at this publication point is that a superseded (but not stale) CRL would cause an RP to improperly accept a revoked certificate as valid (and thus rely upon signed objects that are validated using that certificate). This risk is somewhat mitigated if the CRL for this publication point has a short time between thisUpdate and nextUpdate (and the current time is within this interval). The risk in discarding signed objects at this publication point is that an RP may incorrectly discard a large number of valid objects. This gives significant power to an adversary that is able to delete a manifest at the publication point.

Regardless of whether signed objects from this publication are deemed fit for use by an RP, this situation SHOULD result in a warning to the effect that: "No manifest is available for <pub point name>, and

thus there may have been undetected deletions or replay substitutions from the publication point."

In the case where an RP has access to a local cache of previously issued manifests that are valid, the RP MAY use the most recently previously issued valid manifests for this RPKI repository publication collection in this case for each entity that publishes at his publication point.

6.3. Invalid Manifests

The presence of an invalid manifest at a publication point could occur due to an error by the publisher or due to (malicious or accidental) corruption of a valid manifest. An invalid manifest MUST never be used even if the manifestNumber is greater than that of other valid manifests.

There are no risks associated with using signed objects at a publication point containing an invalid manifest, provided that valid manifests that collectively cover all the signed objects are also present.

If an invalid manifest is present at a publication point that also contains one or more valid manifests, this situation SHOULD result in a warning to the effect that: "An invalid manifest was found at <pub point name>, this indicates an attack against the publication point or an error by the publisher. Processing for this publication point will continue using the most recent valid manifest(s)."

In the case where the RP has access to a local cache of previously issued (valid) manifests, an RP MAY make use of that locally cached data. Specifically, the RP MAY use the locally cached, most recent, previously issued, valid manifest issued by the entity that (appears to have) issued the invalid manifest.

6.4. Stale Manifests

A manifest is considered stale if the current time is after the nextUpdate time for the manifest. This could be due to publisher failure to promptly publish a new manifest, or due to (malicious or accidental) corruption or suppression of a more recent manifest.

All signed objects at the publication point issued by the entity that has published the stale manifest, and all descendant signed objects that are validated using a certificate issued by the entity that has published the stale manifest at this publication point SHOULD be viewed as somewhat suspect, but MAY be used by the RP as per local policy.

The primary risk in using such signed objects is that a newer manifest exists that, if present, would indicate that certain objects are have been removed or replaced. (For example, the new manifest might show the existence of a newer CRL and the removal of one or more revoked certificates). Thus, the use of objects from a stale manifest may cause an RP to incorrectly treat invalid objects as valid. The risk is that the CRL covered by the stale manifest has been superseded, and thus an RP will to improperly treat improperly treat a revoked certificate as valid. This risk is somewhat mitigated if the time between the nextUpdate field of the manifest and the current time is short. The risk in discarding signed objects at this publication point is that the RP may incorrectly discard a large number of valid objects. This gives significant power to an adversary that is able to prevent the publication of a new manifest at a given publication point.

Regardless of whether signed objects from this publication are deemed fit for use by an RP, this situation SHOULD result in a warning to the effect that: "A manifest found at <pub point name> is no longer current. It is possible that undetected deletions have occurred at this publication point."

Note that there is also the potential for the current time to be before the thisUpdate time for the manifest. This case could be due to publisher error, or a local clock error, and in such a case this situation SHOULD result in a warning to the effect that: "A manifest found at <pub point name> has an incorrect thisUpdate field. This could be due to publisher error, or a local clock error, and processing for this publication point will continue using this otherwise valid manifest."

6.5. Mismatch between Manifest and Publication Point

If there exist valid signed objects that do not appear in any manifest, then, provided the manifest is not stale (see Section 6.4) it is likely that their omission is an error by the publisher. It is also possible that this state could be the result of a (malicious or accidental) replacement of a current manifest with an older, but still valid manifest. However, regarding the appropriate interpretation such objects, it remains the case that if the objects were intended to be invalid, then they should have been revoked using whatever revocation mechanism is appropriate for the signed object in question.) Therefore, there is little risk in using such signed objects. If the publication point contains a stale manifest, then there is a greater risk that the objects in question were revoked, along with a missing Certificate Revocation List (CRL), the absence of which is undetectable since the manifest is stale. In any case, the use of signed objects not present on a manifest, or descendant

objects that are validated using such signed objects, is a matter of local policy.

Regardless of whether objects not appearing on a manifest are deemed fit for use by the RP, this situation SHOULD result in a warning to the effect that: "The following files are present in the repository at <pub point name>, but are not listed on any manifest <file list> for <pub point name>."

If there exists files listed on the manifest that do not appear in the repository, then these objects are likely to have been improperly (via malice or accident) deleted from the repository. A primary purpose of manifests is to detect such deletions. Therefore, in such a case this situation SHOULD result in a warning to the effect that: "The following files that should have been present in the repository at <pub point name>, are missing <file list>. This indicates an attack against this publication point, or the repository, or an error by the publisher."

6.6. Hash Values Not Matching Manifests

A file appearing on a manifest with an incorrect hash value could occur because of publisher error, but it also may indicate that an attack has occurred.

If an object appeared on a previous valid manifest with a correct hash value, and it now appears with an invalid hash value, then it is likely that the object has been superseded by a new (unavailable) version of the object. If the object is used, there is a risk that the RP will be treating a stale object as valid. This risk is more significant if the object in question is a CRL. If the object can be validated using the RPKI, the use of these objects is a matter of local policy.

If an object appears on a manifest with an invalid hash and has never previously appeared on a manifest, then it is unclear whether the available version of the object is more or less recent than the version indicated by the manifest. If the manifest is stale (see Section 6.4), then it becomes more likely that the available version is more recent than the version indicated on the manifest, but this is never certain. Whether to use such objects is a matter of local policy. However, in general, it is better to use a possibly outdated version of the object than to discard the object completely.

While it is a matter of local policy, in the case of CRLs, an RP SHOULD endeavour to use the most recently issued valid CRL, even where the hash value in the manifest matches an older CRL, or does not match any available CRL for a CA instance. The `thisUpdate` field

of the CRL can be used to establish the most recent CRL in the case where an RP has more than one valid CRL for a CA instance.

Regardless of whether objects with incorrect hashes are deemed fit for use by the RP, this situation SHOULD result in a warning to the effect that: "The following files at the repository <pub point name> appear on a manifest with incorrect hash values <file list>. It is possible that these objects have been superseded by a more recent version. It is very likely that this problem is due to an attack on the publication point, although it also could be due to a publisher error."

7. Publication Repositories

The RPKI publication system model requires that every publication point be associated with one or more CAs, and be non-empty. Upon creation of the publication point associated with a CA, the CA MUST create and publish a manifest as well as a CRL. A CA's manifest will always contain at least one entry, namely the CRL issued by the CA upon repository creation. [ID.ietf-sidr-repos-struct].

Every published signed object in the RPKI [ID.sidr-signed-object] is published in the repository publication point of the CA that issued the EE certificate, and is listed in the manifest associated with that CA certificate.

8. Security Considerations

Manifests provide an additional level of protection for RPKI RPs. Manifests can assist an RP to determine if a repository object has been deleted, occluded or otherwise removed from view, or if a publication of a newer version of an object has been suppressed (and an older version of the object has been substituted).

Manifests cannot repair the effects of such forms of corruption of repository retrieval operations. However, a manifest enables an RP to determine if a locally maintained copy of a repository is a complete and up to date copy, even when the repository retrieval operation is conducted over an insecure channel. In cases where the manifest and the retrieved repository contents differ, the manifest can assist in determining which repository objects form the difference set in terms of missing, extraneous or superseded objects.

The signing structure of a manifest and the use of the nextUpdate value allows an RP to determine if the manifest itself is the subject of attempted alteration. The requirement for every repository

publication point to contain at least one manifest allows an RP to determine if the manifest itself has been occluded from view. Such attacks against the manifest are detectable within the time frame of the regular schedule of manifest updates. Forms of replay attack within finer-grained time frames are not necessarily detectable by the manifest structure.

9. IANA Considerations

[Note to IANA, to be removed prior to publication: there are no IANA considerations stated in this version of the document.]

10. Acknowledgements

The authors would like to acknowledge the contributions from George Michelson and Randy Bush in the preparation of the manifest specification. Additionally, the authors would like to thank Mark Reynolds and Christopher Small for assistance in clarifying manifest validation and RP behaviour. The authors also wish to thank Sean Turner for his helpful review of this document.

11. References

11.1. Normative References

[I-D.sidr-res-certs]

Huston, G., Michaelson, G., and R. Loomans, "A Profile for X.509 PKIX Resource Certificates", draft-ietf-sidr-res-certs-16.txt (work in progress), February 2009.

[ID.ietf-sidr-repos-struct]

Huston, G., Loomans, R., and G. Michaelson, "A Profile for Resource Certificate Repository Structure", draft-ietf-sidr-repos-struct-04.txt (work in progress), May 2010.

[ID.ietf-sidr-rpki-algs]

Huston, G., "A Profile for Algorithms and Key Sizes for use in the Resource Public Key Infrastructure", draft-huston-sidr-rpki-algs-00.txt (work in progress), July 2009.

[ID.sidr-signed-object]

Lepinski, M., Chi, A., and S. Kent, "Signed Object

Template for the Resource Public Key Infrastructure",
draft-ietf-sidr-signed-object-01.txt (work in progress),
October 2010.

- [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, May 2008.

11.2. Informative References

- [ID.ietf-sidr-arch]
Lepinski, M. and S. Kent, "An Infrastructure to Support
Secure Internet Routing", draft-ietf-sidr-arch-11.txt
(work in progress), September 2010.
- [ID.sidr-keyroll]
Huston, G., Michaelson, G., and S. Kent, "CA Key Rollover
in the RPKI", draft-ietf-sidr-keyroll-02.txt (work in
progress), October 2010.
- [RFC3779] Lynn, C., Kent, S., and K. Seo, "X.509 Extensions for IP
Addresses and AS Identifiers", RFC 3779, June 2004.

Appendix A. ASN.1 Module

```
RPKIManifest { iso(1) identified-organization(3)
  dod(6) internet(1) security(5) mechanisms(5) smime(7)
  mod(0) TBD }

DEFINITIONS EXPLICIT TAGS ::=

BEGIN

-- EXPORTS ALL --

-- IMPORTS NOTHING --

-- Manifest Content Type: OID

id-smime OBJECT IDENTIFIER ::= { iso(1) member-body(2)
  us(840) rsadsi(113549) pkcs(1) pkcs9(9) 16 }

id-ct OBJECT IDENTIFIER ::= { id-smime 1 }

id-ct-rpkiManifest OBJECT IDENTIFIER ::= { id-ct 26 }

-- Manifest Content Type: eContent

Manifest ::= SEQUENCE {
  version          [0] INTEGER DEFAULT 0,
  manifestNumber    INTEGER (0..MAX),
  thisUpdate        GeneralizedTime,
  nextUpdate        GeneralizedTime,
  fileHashAlg       OBJECT IDENTIFIER,
  fileList          SEQUENCE SIZE (0..MAX) OF FileAndHash
}

FileAndHash ::= SEQUENCE {
  file  IA5String,
  hash  BIT STRING
}

END
```

Authors' Addresses

Rob Austein
Internet Systems Consortium
950 Charter St.
Redwood City, CA 94063
USA

Email: sra@isc.org

Geoff Huston
Asia Pacific Network Information Centre
33 Park Rd.
Milton, QLD 4064
Australia

Email: gih@apnic.net
URI: <http://www.apnic.net>

Stephen Kent
BBN Technologies
10 Moulton St.
Cambridge, MA 02138
USA

Email: kent@bbn.com

Matt Lepinski
BBN Technologies
10 Moulton St.
Cambridge, MA 02138
USA

Email: mlepinski@bbn.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: March 2, 2011

R. Bush
IIJ
R. Austein
ISC
August 29, 2010

The RPKI/Router Protocol
draft-ietf-sidr-rpki-rtr-02

Abstract

In order to formally validate the origin ASes of BGP announcements, routers need a simple but reliable mechanism to receive RPKI [I-D.ietf-sidr-arch] or analogous prefix origin data from a trusted cache. This document describes a protocol to deliver validated prefix origin data to routers over ssh.

Requirements Language

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

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 <http://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 March 2, 2011.

Copyright Notice

Copyright (c) 2010 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents

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

Table of Contents

1. Introduction	3
2. Deployment Structure	3
3. Operational Overview	3
4. Protocol Data Units (PDUs)	4
4.1. Serial Notify	5
4.2. Serial Query	5
4.3. Reset Query	6
4.4. Cache Response	6
4.5. IPv4 Prefix	7
4.6. IPv6 Prefix	8
4.7. End of Data	8
4.8. Cache Reset	9
4.9. Error Report	9
4.10. Fields of a PDU	10
5. Protocol Sequences	11
5.1. Start or Restart	11
5.2. Typical Exchange	12
5.3. No Incremental Update Available	13
5.4. Cache has No Data Available	13
6. SSH Transport	14
7. Router-Cache Set-Up	14
8. Deployment Scenarios	16
9. Error Codes	16
10. Security Considerations	17
11. Glossary	18
12. IANA Considerations	18
13. Acknowledgments	19
14. References	19
14.1. Normative References	19
14.2. Informative References	19
Authors' Addresses	20

1. Introduction

In order to formally validate the origin ASes of BGP announcements, routers need a simple but reliable mechanism to receive RPKI [I-D.ietf-sidr-arch] or analogous formally validated prefix origin data from a trusted cache. This document describes a protocol to deliver validated prefix origin data to routers over ssh.

Section 2 describes the deployment structure and Section 3 then presents an operational overview. The binary payloads of the protocol are formally described in Section 4, and the expected PDU sequences are described in Section 5. The transport protocol is described in Section 6. Section 7 details how routers and caches are configured to connect and authenticate. Section 8 describes likely deployment scenarios. The traditional security and IANA considerations end the document.

The protocol is extensible to support new PDUs with new semantics when and as needed, as indicated by deployment experience. PDUs are versioned should deployment experience call for change.

2. Deployment Structure

Deployment of the RPKI to reach routers has a three level structure as follows:

Global RPKI: The authoritative data of the RPKI are published in a distributed set of servers, RPKI publication repositories, e.g. the IANA, RIRs, NIRs, and ISPs, see [I-D.ietf-sidr-repos-struct].

Local Caches: A local set of one or more collected and verified non-authoritative caches. A relying party, e.g. router or other client, MUST have a formally authenticated trust relationship with, and a secure transport channel to, any non-authoritative cache(s) it uses.

Routers: A router fetches data from a local cache using the protocol described in this document. It is said to be a client of the cache. There are mechanisms for the router to assure itself of the authenticity of the cache and to authenticate itself to the cache.

3. Operational Overview

A router establishes and keeps open an authenticated connection to a cache with which it has a client/server relationship. It is

configured with a semi-ordered list of caches, and establishes a connection to the most preferred cache, or set of caches, which accepts one.

Periodically, the router sends to the cache the serial number of the highest numbered data record it has received from that cache, i.e. the router's current serial number. When a router establishes a new connection to a cache, or wishes to reset a current relationship, it sends a Reset Query.

The Cache responds with all data records which have serial numbers greater than that in the router's query. This may be the null set, in which case the End of Data PDU is still sent. Note that 'greater' must take wrap-around into account, see [RFC1982].

When the router has received all data records from the cache, it sets its current serial number to that of the serial number in the End of Data PDU.

When the cache updates its database, it sends a Notify message to every currently connected router. This is a hint that now would be a good time for the router to poll for an update, but is only a hint. The protocol requires the router to poll for updates periodically in any case.

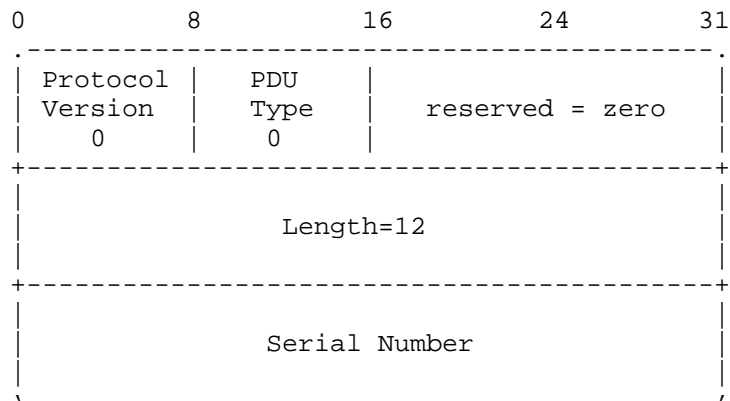
Strictly speaking, a router could track a cache simply by asking for a complete data set every time it updates, but this would be very inefficient. The serial number based incremental update mechanism allows an efficient transfer of just the data records which have changed since last update. As with any update protocol based on incremental transfers, the router must be prepared to fall back to a full transfer if for any reason the cache is unable to provide the necessary incremental data. Unlike some incremental transfer protocols, this protocol requires the router to make an explicit request to start the fallback process; this is deliberate, as the cache has no way of knowing whether the router has also established sessions with other caches that may be able to provide better service.

4. Protocol Data Units (PDUs)

The exchanges between the cache and the router are sequences of exchanges of the following PDUs according to the rules described in Section 5.

4.1. Serial Notify

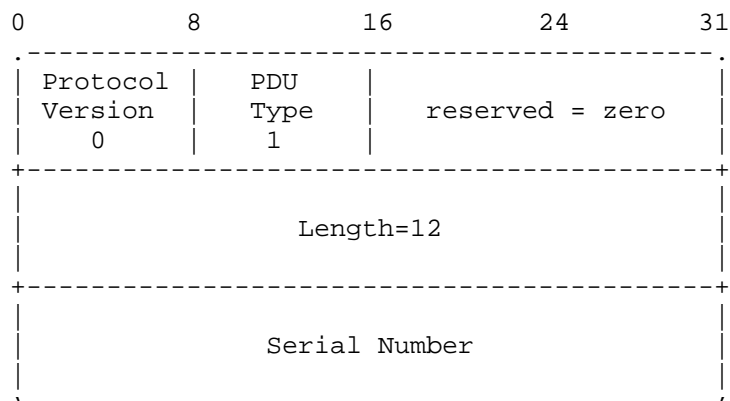
The cache notifies the router that the cache has new data.



4.2. Serial Query

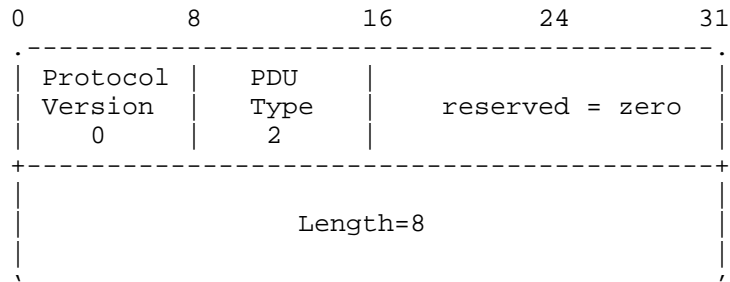
Serial Query: The router sends Serial Query to ask the cache for all payload PDUs which have serial numbers higher than the serial number in the Serial Query.

The cache replies to this query with a Cache Response PDU (Section 4.4) if the cache has a record of the changes since the serial number specified by the router. If there have been no changes since the router last queried, the cache responds with an End Of Data PDU. If the cache does not have the data needed to update the router, perhaps because its records do not go back to the Serial Number in the Serial Query, then it responds with a Cache Reset PDU (Section 4.8).



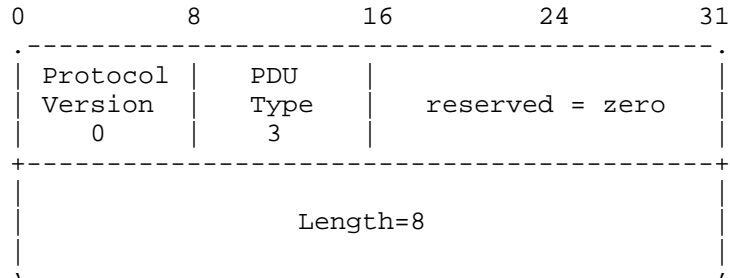
4.3. Reset Query

Reset Query: The router tells the cache that it wants to receive the total active, current, non-withdrawn, database. The cache responds with a Cache Response PDU (Section 4.4).

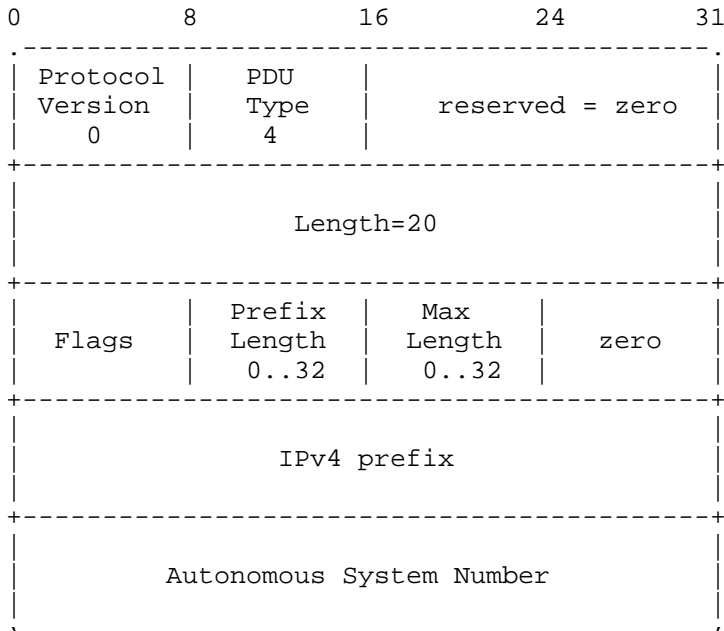


4.4. Cache Response

Cache Response: The cache responds with zero or more payload PDUs. When replying to a Serial Query request (Section 4.2), the cache sends the set of all data records it has with serial numbers greater than that sent by the client router. When replying to a Reset Query, the cache sends the set of all data records it has; in this case the withdraw/announce field in the payload PDUs MUST have the value 1 (announce).



4.5. IPv4 Prefix



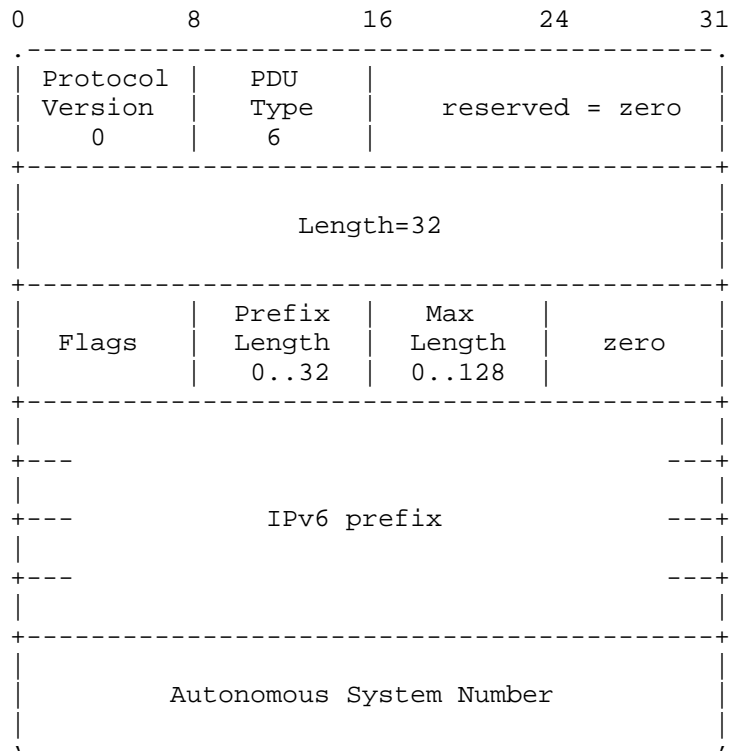
Due to the nature of the RPKI and the IRR, there can be multiple identical IPvX PDUs. A router MUST be prepared to receive multiple identical record announcements and MUST NOT consider a record to have been deleted until it has received a corresponding number of withdrawals or a reset is performed. Hence the router will likely keep an internal reference count on each IPvX PDU.

In the RPKI, nothing prevents a signing certificate from issuing two identical ROAs, and nothing prohibits the existence of two identical route: or route6: objects in the IRR. In this case there would be no semantic difference between the objects, merely a process redundancy.

In the RPKI, there is also an actual need for what will appear to the router as identical IPvX PDUs. This occurs when an upstream certificate is being reissued or a site is changing providers, often a 'make and break' situation. The ROA is identical in the router sense, i.e. has the same {prefix, len, max-len, asn}, but has a different validation path in the RPKI. This is important to the RPKI, but not to the router.

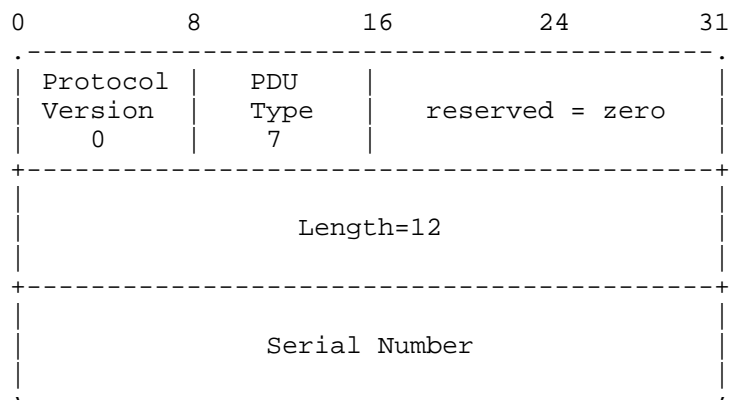
The lowest order bit of the Flags field is 1 for an announcement and 0 for a withdrawal.

4.6. IPv6 Prefix



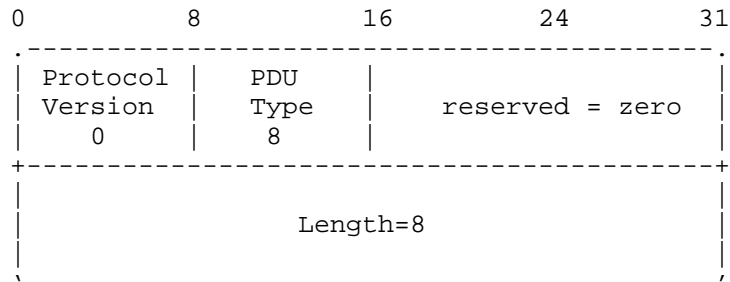
4.7. End of Data

End of Data: Cache tells router it has no more data for the request.



4.8. Cache Reset

The cache may respond to a Serial Query informing the router that the cache cannot provide an incremental update starting from the serial number specified by the router. The router must decide whether to issue a Reset Query or switch to a different cache.



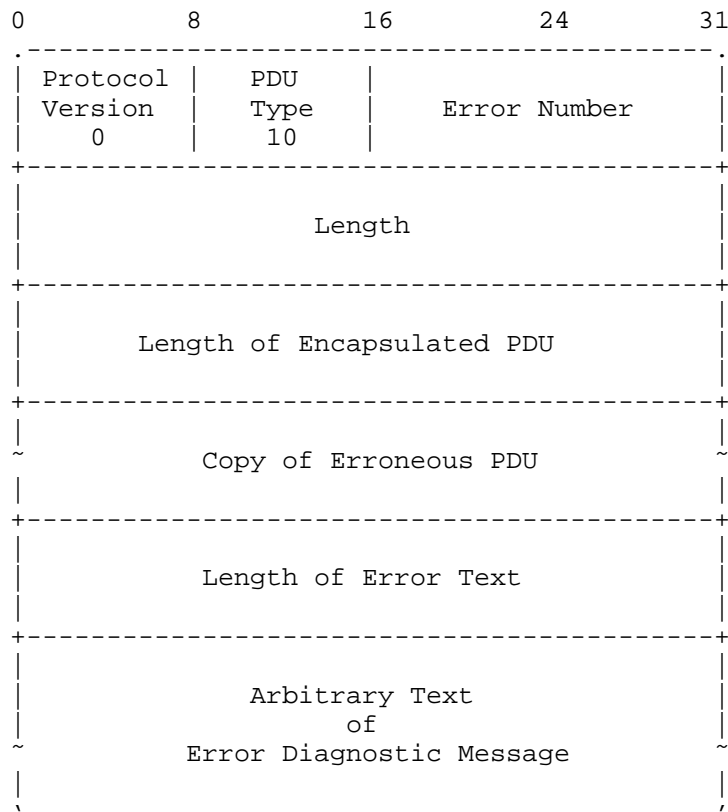
4.9. Error Report

This PDU is used by either party to report an error to the other.

The Error Number is described in Section 9.

If the error is not associated with any particular PDU, the Erroneous PDU field should be empty and the Length of Encapsulated PDU field should be zero.

The diagnostic text is optional, if not present the Length of Error Text field should be zero. If error text is present, it SHOULD be a string in US-ASCII, for maximum portability; if non-US-ASCII characters are absolutely required, the error text MUST use UTF-8 encoding.



4.10. Fields of a PDU

PDU's contain the following data elements:

Protocol Version: An ordinal, currently 0, denoting the version of this protocol.

Serial Number: The serial number of the RPKI Cache when this ROA was received from the cache's up-stream cache server or gathered from the global RPKI. A cache increments its serial number when completing an rcynic from a parent cache. See [RFC1982] on DNS Serial Number Arithmetic for too much detail on serial number arithmetic.

Length: A 32 bit ordinal which has as its value the count of the bytes in the entire PDU, including the eight bytes of header which end with the length field.

Flags: The lowest order bit of the Flags field is 1 for an announcement and 0 for a withdrawal, whether this PDU announces a new right to announce the prefix or withdraws a previously announced right. A withdraw effectively deletes one previously announced IPvX Prefix PDU with the exact same Prefix, Length, Max-Len, and ASN.

Prefix Length: An ordinal denoting the shortest prefix allowed for the prefix.

Max Length: An ordinal denoting the longest prefix allowed by the prefix. This MUST NOT be less than the Prefix Length element.

Prefix: The IPv4 or IPv6 prefix of the ROA.

Autonomous System Number: ASN allowed to announce this prefix, a 32 bit ordinal.

Zero: Fields shown as zero or reserved MUST be zero. The value of such a field MUST be ignored on receipt.

5. Protocol Sequences

The sequences of PDU transmissions fall into three conversations as follows:

5.1. Start or Restart

Cache ~	Router ~
<----- Reset Query ----->	R requests data
----- Cache Response ----->	C confirms request
----- IPvX Prefix ----->	C sends zero or more
----- IPvX Prefix ----->	IPv4 and IPv6 Prefix
----- IPvX Prefix ----->	Payload PDUs
----- End of Data ----->	C sends End of Data
	and sends new serial

When a transport session is first established, the router sends a Reset Query and the cache responds with a data sequence of all data it contains.

This Reset Query sequence is also used when the router receives a Cache Reset, chooses a new cache, or fears that it has otherwise lost its way.

To limit the length of time a cache must keep the data necessary to generate incremental updates, a router **MUST** send either a Serial Query or a Reset Query no less frequently than once an hour. This also acts as a keep alive at the application layer.

5.2. Typical Exchange

Cache	Router
~	~
----- Notify ----->	(optional)
<----- Serial Query -----	R requests data
----- Cache Response ----->	C confirms request
----- IPvX Prefix ----->	C sends zero or more
----- IPvX Prefix ----->	IPv4 and IPv6 Prefix
----- IPvX Prefix ----->	Payload PDUs
----- End of Data ----->	C sends End of Data
	and sends new serial
~	~

The cache server **SHOULD** send a notify PDU with its current serial number when the cache's serial changes, with the expectation that the router **MAY** then issue a serial query earlier than it otherwise might. This is analogous to DNS NOTIFY in [RFC1996]. The cache **SHOULD** rate limit Serial Notices to no more frequently than one per minute.

When the transport layer is up and either a timer has gone off in the router, or the cache has sent a Notify, the router queries for new data by sending a Serial Query, and the cache sends all data newer than the serial in the Serial Query.

To limit the length of time a cache must keep old withdraws, a router **MUST** send either a Serial Query or a Reset Query no less frequently than once an hour.

5.3. No Incremental Update Available

Cache	Router
~	~
<----- Serial Query ----->	R requests data
----- Cache Reset ----->	C cannot supply update
	from specified serial
<----- Reset Query ----->	R requests new data
----- Cache Response ----->	C confirms request
----- IPvX Prefix ----->	C sends zero or more
----- IPvX Prefix ----->	IPv4 and IPv6 Prefix
----- IPvX Prefix ----->	Payload PDUs
----- End of Data ----->	C sends End of Data
	and sends new serial
~	~

The cache may respond to a Serial Query with a Cache Reset, informing the router that the cache cannot supply an incremental update from the serial number specified by the router. This might be because the cache has lost state, or because the router has waited too long between polls and the cache has cleaned up old data that it no longer believes it needs, or because the cache has run out of storage space and had to expire some old data early. Regardless of how this state arose, the cache replies with a Cache Reset to tell the router that it cannot honor the request. When a router receives this, the router SHOULD attempt to connect to any more preferred caches in its cache list. If there are no more preferred caches it MUST issue a Reset Query and get an entire new load from the cache.

5.4. Cache has No Data Available

Cache	Router
~	~
<----- Serial Query ----->	R requests data
----- Error Report PDU ----->	C cannot supply update
~	~

Cache	Router
~	~
<----- Reset Query ----->	R requests data
----- Error Report PDU ----->	C cannot supply update
~	~

The cache may respond to either a Serial Query or a Reset Query informing the router that the cache cannot supply any update at all. The most likely cause is that the cache has lost state, perhaps due to a restart, and has not yet recovered. While it is possible that a cache might go into such a state without dropping any of its active

sessions, a router is more likely to see this behavior when it initially connects and issues a Reset Query while the cache is still rebuilding its database.

When a router receives this kind of error, the router SHOULD attempt to connect to any other caches in its cache list, in preference order. If no other caches are available, the router MUST issue periodic Reset Queries until it gets a new usable load from the cache.

6. SSH Transport

The transport layer session between a router and a cache carries the binary Protocol Data Units (PDUs) in a persistent SSH session.

To run over SSH, the client router first establishes an SSH transport connection using the SSH transport protocol, and the client and server exchange keys for message integrity and encryption. The client then invokes the "ssh-userauth" service to authenticate the application, as described in the SSH authentication protocol RFC 4252 [RFC4252]. Once the application has been successfully authenticated, the client invokes the "ssh-connection" service, also known as the SSH connection protocol.

After the ssh-connection service is established, the client opens a channel of type "session", which results in an SSH session.

Once the SSH session has been established, the application invokes the application transport as an SSH subsystem called "rpki-rtr". Subsystem support is a feature of SSH version 2 (SSHv2) and is not included in SSHv1. Running this protocol as an SSH subsystem avoids the need for the application to recognize shell prompts or skip over extraneous information, such as a system message that is sent at shell start-up.

It is assumed that the router and cache have exchanged keys out of band by some reasonably secured means.

7. Router-Cache Set-Up

A cache has the public authentication data for each router it is configured to support.

A router may be configured to peer with a selection of caches, and a cache may be configured to support a selection of routers. Each must have the name of, and authentication data for, each peer. In

addition, in a router, this list has a non-unique preference value for each server in order of preference. This preference merely denotes proximity, not trust, preferred belief, etc. The client router attempts to establish a session with each potential serving cache in preference order, and then starts to load data from the most preferred cache to which it can connect and authenticate. The router's list of caches has the following elements:

Preference: An ordinal denoting the router's preference to connect to that cache, the lower the value the more preferred.

Name: The IP Address or fully qualified domain name of the cache.

Key: The public ssh key of the cache.

MyKey: The private ssh key of this client.

Due to the distributed nature of the RPKI, caches simply can not be rigorously synchronous. A client may hold data from multiple caches, but MUST keep the data marked as to source, as later updates MUST affect the correct data.

Just as there may be more than one covering ROA from a single cache, there may be multiple covering ROAs from multiple caches. The results are as described in [I-D.ietf-sidr-roa-validation].

If data from multiple caches are held, implementations MUST NOT distinguish between data sources when performing validation.

When a more preferred cache becomes available, if resources allow, it would be prudent for the client to start fetching from that cache.

The client SHOULD attempt to maintain at least one set of data, regardless of whether it has chosen a different cache or established a new connection to the previous cache.

A client MAY drop the data from a particular cache when it is fully in synch with one or more other caches.

A client SHOULD delete the data from a cache when it has been unable to refresh from that cache for a configurable timer value. The default for that value is twice the polling period for that cache.

If a client loses connectivity to a cache it is using, or otherwise decides to switch to a new cache, it SHOULD retain the data from the previous cache until it has a full set of data from one or more other caches. Note that this may already be true at the point of connection loss if the client has connections to more than one cache.

8. Deployment Scenarios

For illustration, we present three likely deployment scenarios.

Small End Site: The small multi-homed end site may wish to outsource the RPKI cache to one or more of their upstream ISPs. They would exchange authentication material with the ISP using some out of band mechanism, and their router(s) would connect to one or more up-streams' caches. The ISPs would likely deploy caches intended for customer use separately from the caches with which their own BGP speakers peer.

Large End Site: A larger multi-homed end site might run one or more caches, arranging them in a hierarchy of client caches, each fetching from a serving cache which is closer to the global RPKI. They might configure fall-back peerings to up-stream ISP caches.

ISP Backbone: A large ISP would likely have one or more redundant caches in each major PoP, and these caches would fetch from each other in an ISP-dependent topology so as not to place undue load on the global RPKI publication infrastructure.

Experience with large DNS cache deployments has shown that complex topologies are ill-advised as it is easy to make errors in the graph, e.g. not maintaining a loop-free condition.

Of course, these are illustrations and there are other possible deployment strategies. It is expected that minimizing load on the global RPKI servers will be a major consideration.

To keep load on global RPKI services from unnecessary peaks, it is recommended that primary caches which load from the distributed global RPKI not do so all at the same times, e.g. on the hour. Choose a random time, perhaps the ISP's AS number modulo 60 and jitter the inter-fetch timing.

9. Error Codes

This section contains a preliminary list of error codes. The authors expect additions to this section during development of the initial implementations. Eventually, these error codes will probably need to reside in an IANA registry.

0: Reserved.

- 1: Internal Error: The party reporting the error experienced some kind of internal error unrelated to protocol operation (ran out of memory, a coding assertion failed, et cetera).
- 2: No Data Available: The cache believes itself to be in good working order, but is unable to answer either a Serial Query or a Reset Query because it has no useful data available at this time. This is likely to be a temporary error, and most likely indicates that the cache has not yet completed pulling down an initial current data set from the global RPKI system after some kind of event that invalidated whatever data it might have previously held (reboot, network partition, etcetera).

10. Security Considerations

As this document describes a security protocol, many aspects of security interest are described in the relevant sections. This section points out issues which may not be obvious in other sections.

Cache Validation: In order for a collection of caches as described in Section 8 to guarantee a consistent view, they need to be given consistent trust anchors to use in their internal validation process. Distribution of a consistent trust anchor is assumed to be out of band.

Cache Peer Identification: The router initiates an ssh transport session to a cache, which it identifies by either IP address or fully qualified domain name. Be aware that a DNS or address spoofing attack could make the correct cache unreachable. No session would be established, as the authorization keys would not match.

Transport Security: The RPKI relies on object, not server or transport, trust. I.e. the IANA root trust anchor is distributed to all caches through some out of band means, and can then be used by each cache to validate certificates and ROAs all the way down the tree. The inter-cache relationships are based on this object security model, hence the inter-cache transport can be lightly protected.

But this protocol document assumes that the routers can not do the validation cryptography. Hence the last link, from cache to router, is secured by server authentication and transport level security. This is dangerous, as server authentication and transport have very different threat models than object security.

So the strength of the trust relationship and the transport between the router(s) and the cache(s) are critical. You're betting your routing on this.

While we can not say the cache must be on the same LAN, if only due to the issue of an enterprise wanting to off-load the cache task to their upstream ISP(s), locality, trust, and control are very critical issues here. The cache(s) really SHOULD be as close, in the sense of controlled and protected (against DDoS, MITM) transport, to the router(s) as possible. It also SHOULD be topologically close so that a minimum of validated routing data are needed to bootstrap a router's access to a cache.

11. Glossary

The following terms are used with special meaning:

Global RPKI: The authoritative data of the RPKI are published in a distributed set of servers at the IANA, RIRs, NIRs, and ISPs, see [I-D.ietf-sidr-repos-struct].

Non-authoritative Cache: A coalesced copy of the RPKI which is periodically fetched/refreshed directly or indirectly from the global RPKI using the [RFC5781] protocol/tools

Cache: The rcynic system is used to gather the distributed data of the RPKI into a validated cache. Trusting this cache further is a matter between the provider of the cache and a relying party.

Serial Number: A 32-bit monotonically increasing ordinal which wraps from $2^{32}-1$ to 0. It denotes the logical version of a cache. A cache increments the value by one when it successfully updates its data from a parent cache or from primary RPKI data. As a cache is rcynicing, new incoming data, and implicit deletes, are marked with the new serial but MUST not be sent until the fetch is complete. A serial number is not commensurate between caches, nor need it be maintained across resets of the cache server. See [RFC1982] on DNS Serial Number Arithmetic for too much detail on serial number arithmetic.

12. IANA Considerations

This document requests the IANA to create a registry for PDU types.

This document requests the IANA to create a registry for Error Codes.

In addition, a registry for Version Numbers would be needed if new Version Number is defined in a new RFC.

Note to RFC Editor: this section may be replaced on publication as an RFC.

13. Acknowledgments

The authors wish to thank Steve Bellovin, Rex Fernando, Russ Housley, Pradosh Mohapatra, Keyur Patel, Sandy Murphy, Robert Raszuk, John Scudder, Ruediger Volk, and David Ward. Particular thanks go to Hannes Gredler for showing us the dangers of unnecessary fields.

14. References

14.1. Normative References

- [I-D.ietf-sidr-roa-validation]
Huston, G. and G. Michaelson, "Validation of Route Origination using the Resource Certificate PKI and ROAs", draft-ietf-sidr-roa-validation-06 (work in progress), May 2010.
- [RFC1982] Elz, R. and R. Bush, "Serial Number Arithmetic", RFC 1982, August 1996.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC4252] Ylonen, T. and C. Lonvick, "The Secure Shell (SSH) Authentication Protocol", RFC 4252, January 2006.
- [RFC5781] Weiler, S., Ward, D., and R. Housley, "The rsync URI Scheme", RFC 5781, February 2010.

14.2. Informative References

- [I-D.ietf-sidr-arch]
Lepinski, M. and S. Kent, "An Infrastructure to Support Secure Internet Routing", draft-ietf-sidr-arch-09 (work in progress), October 2009.
- [I-D.ietf-sidr-repos-struct]
Huston, G., Loomans, R., and G. Michaelson, "A Profile for Resource Certificate Repository Structure", draft-ietf-sidr-repos-struct-04 (work in progress),

May 2010.

[RFC1996] Vixie, P., "A Mechanism for Prompt Notification of Zone Changes (DNS NOTIFY)", RFC 1996, August 1996.

Authors' Addresses

Randy Bush
Internet Initiative Japan, Inc.
5147 Crystal Springs
Bainbridge Island, Washington 98110
US

Phone: +1 206 780 0431 x1
Email: randy@psg.com

Rob Austein
Internet Systems Consortium
950 Charter Street
Redwood City, CA 94063
USA

Email: sra@isc.org

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: April 21, 2011

R. Gagliano
Cisco Systems
S. Kent
BBN Technologies
S. Turner
IECA, Inc.
October 18, 2010

Algorithm Agility Procedure for RPKI.
draft-rgaglian-sidr-algorithm-agility-00

Abstract

This document specifies the process that Certificate Authorities (CAs) and Relying Parties (RP) participating in the Resource Public Key Infrastructure (RPKI) will need to follow to transition to a new (and probably cryptographically stronger) algorithm set. The process is expected to be completed in a time scale of months or years. Consequently, no emergency transition is specified.

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 <http://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 April 21, 2011.

Copyright Notice

Copyright (c) 2010 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect

to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Requirements notation	3
2. Introduction	4
3. Terminology	6
4. Key Rollover steps for algorithm migration	8
4.1. Milestones definition	8
4.2. Process overview	8
4.3. Phase 0	9
4.4. Phase 1	10
4.5. Phase 2	12
4.6. Phase 3	13
4.7. Phase 4	14
4.8. Phase 5	14
4.9. Phase 0	14
5. Synchronization issues during the algorithm transition	15
5.1. Signed Objects Information	15
5.2. Revocations	15
5.3. Key rollover	15
5.4. Repository structure	15
6. IANA Considerations	16
7. Security Considerations	17
8. Acknowledgements	18
9. References	19
9.1. Normative References	19
9.2. Informative References	20
Appendix A. Change Log	21
Authors' Addresses	22

1. Requirements notation

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

2. Introduction

The RPKI must accommodate transitions between the public keys by used by CAs. Transitions of this sort are usually termed "key rollover". Planned key rollover will occur at regular intervals throughout the life of the RPKI, as each CA changes its public keys, in a non-coordinated fashion. (By non-coordinated we mean that the time at which each CA elects to change its keys is locally determined, not coordinated across the RPKI.) Moreover, because a key change might be necessitated by suspected private key compromise, one can never assume coordination of these events among all of the CAs in the RPKI. In an emergency key rollover, the old certificate is revoked and a new certificate with a new key is issued. The mechanisms to perform a key rollover in RPKI (either planned or in an emergency), while maintaining the same algorithm suite, are covered in [I-D.ietf-sidr-keyroll].

This document describes the mechanism to perform a key rollover in RPKI due to the migration to a new signature algorithm suite. A signature algorithm suite encompasses both a signature algorithm (with a specified key size range) and a one-way hash algorithm. It is anticipated that the RPKI will require the adoption of updated key sizes and/or different algorithm suites over time. (One might adopt a new hash algorithm while retaining the current signature algorithm. In such circumstances this document treats the change as equivalent to a key rollover, and requires the CA to change its key as well). Such transitions will be required in order to maintain an acceptable level of cryptographic security, to protect the integrity of certificates, CRLs and signed objects in the RPKI. All of the data structures in the RPKI explicitly identify the signature and hash algorithms being used. However, experience has demonstrated that the ability to represent algorithm IDs is not sufficient to enable migration to new algorithm suites (algorithm agility). One also must ensure that protocols, infrastructure elements, and operational procedures also accommodate migration from one algorithm suite to another. Algorithm migration is expected to be very infrequent, but it also will require support of a "current" and "next" suite for a prolonged interval, probably several years.

This document defines how entities in the RPKI execute (planned) CA key rollover when the algorithm suite changes. The description covers actions by CAs, repository operators, and RPs. It describes the behavior required of both CAs and RPs to make such key changes work in the RPKI context, including how the RPKI repository system is used to support key rollover.

A failure to comply with this process during an algorithm transition MUST be considered as non-compliance with the RPKI Certificate Policy

(CP).

3. Terminology

This document assumes that the reader is familiar with the terms and concepts described in "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile" [RFC5280], "X.509 Extensions for IP Addresses and AS Identifiers" [RFC3779], and "A Profile for Resource Certificate Repository Structure" [I-D.ietf-sidr-repos-struct]. Additional terms and conventions used in examples are provided below.

Algorithm migration A planned transition from one signature and hash algorithm to a new signature and hash algorithm.

Algorithm Suite A The "current" algorithm suite used for hashing and signing, in examples in this document

Algorithm Suite B The "next" algorithm suite used for hashing and signing, used in examples in this document

Algorithm Suite C The "old" algorithm suite used for hashing and signing, used in examples in this document

CA X The CA that issued CA Y's certificate (i.e., CA Y's parent), used in examples in this document.

CA Y The CA that is changing keys and/or algorithm suites, used in examples in this document

CA Z A CA that is a "child" of CA Y, used in examples in this document

Certificate reissuance (unilateral) Certificate reissuance (unilateral) - a CA MAY reissue a certificate to a subordinate Subject without the involvement of the Subject. The public key, resource extensions, and most other fields (see Section X.X) are copied from the current Subject certificate into the next Subject certificate. The Issuer name MAY change, if necessary to reflect the Subject name in the CA certificate under which the reissued certificate will be validated. The validity interval also MAY be changed. This action is defined as a unilateral certificate reissuance.

Key rollover (planned) - reissuance of CA's certificate with a new key, at a pre-determined time.

Key rollover (emergency) - reissuance of CA's certificate with a new key, e.g., as a result of a suspected compromise or loss of access to a private key. An emergency key rollover is not a planned event (from the perspective of the CA).

Non-Leaf CA - a CA that issues certificates to entities not under its administrative control.

PoP (proof of possession) - execution of a protocol that demonstrates to an issuer that a subject requesting a certificate possesses the private key corresponding to the public key in the certificate submitted by the subject.

Signed Product Set (or Set) - a collection of certificates, signed objects, a CRL and a manifest that are associated by virtue of being verifiable under the same parent CA certificate

4. Key Rollover steps for algorithm migration

The "current" RPKI algorithm suite (Suite A) definition is defined in the RPKI's CP document , by reference to [I-D.ietf-sidr-rpki-algs]. If a migration of the RPKI algorithm suite is needed, the first step MUST be an update of the [I-D.ietf-sidr-rpki-algs] document that will include all the information described in section Section 4.3.

4.1. Milestones definition

CA Ready Algorithm B Date - After this date, all (non-leaf) CAs MUST be ready to process a request from a child CA to issue a certificate containing an Algorithm B key for signature, even if signing the certificate using the Algorithm A suite.

CA Set Algorithm B Date - After this date, all CAs MUST be able to issue a certificate signed under the Algorithm B suite to a child CA that requests such. At this stage there is no requirement that any CA issue such certificates, but rather that all (non-eaf) CAs be prepared to do so upon request.

CA Go Algorithm B Date - After this date, all (non-leaf) CAs MUST have re-issued all of its signed product set under the Algorithm B suite.

RP Ready Algorithm B Date - After this date, all RPs MUST be prepared to process signed material issued under the Algorithm B suite.

Twilight Algorithm B - After this date, a CA MAY cease issuing signed products under the old algorithm suite. Also, after this date, a RP MAY cease to validate signed materials issued under the Algorithm A suite.

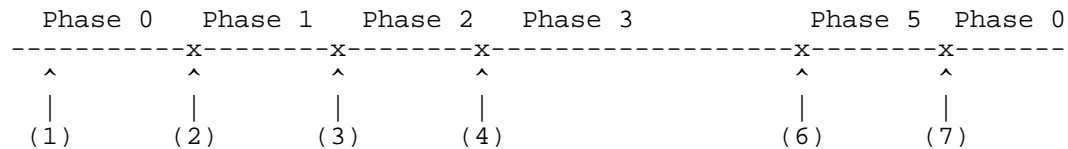
End Of Life (EOL) Algorithm A - After this date every CA MUST NOT generate certificates, CRLs, or other RPKI signed objects under the Algorithm A suite. Also, after this date, no RP should validate any certificate, CRL or signed object using the Algorithm A suite.

4.2. Process overview

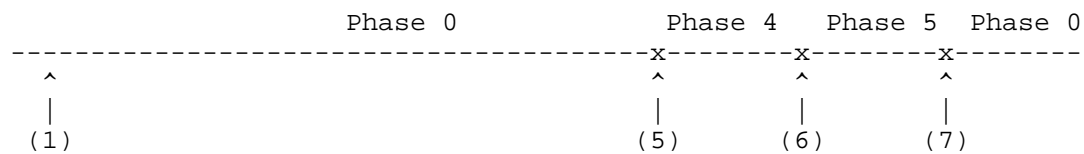
The migration process described in this document involves a series of steps that MUST be executed in chronological order by CAs and RPs. The only milestone that affects both CAs and RPs, at the same moment is the EOL date. Due to the decentralized nature of the RPKI

infrastructure, it is expected that the process will take several months or even years. It also is expected that different CAs and RPs will achieve the various milestones at different moments without central synchronization. The following figure gives an overview of the process:

Process for RPKI CAs:



Process for RPKI RPs:



- (1) RPKI's algorithm document updated.
- (2) CA Ready Algorithm B Date
- (3) CA Set Algorithm B Date
- (4) CA Go Algorithm B Date
- (5) RP Ready Algorithm B Date
- (6) Twilight Date
- (7) End Of Live (EOL) Date

4.3. Phase 0

Phase 0 is the initial phase of the process, during which the Algorithm suite A is the only required algorithm suite in RPKI.

The first milestone here (1), is updating the [alg] document with the following definitions:

- o Algorithm Suite A
- o Algortihm Suite B
- o CA Ready Algorithm B Date
- o CA Set Algorithm B Date
- o CA Go Algorithm B Date

- o RP Ready Algorithm B Date
- o Twilight Date
- o EOL Date

All Dates MUST be represented using the local UTC date-time format specified in [RFC 3339].

As an example, during Phase 0, CAs X, Y and Z are required to generate signed product sets using only the Algorithm suite A. Also, RPs are required to validate signed product sets issued using only Algorithm suite A.

```
CA X-Certificate-Algorithm-Suite-A (Cert-XAA)
    |-> CA-Y-Certificate-Algorithm-Suite-A (Cert-YAA)
        |-> CA-Z-Certificate-Algorithm-Suite-A (Cert-ZAA)
            |-> CA-Z-Signed-Objects-Algorithm-Suite-A
                |-> CA-Y-CRL-Algorithm-Suite-A (CRL-YA)
                    |-> CA-Y-Other-Signed-Objects-Algorithm-Suite-A
```

Note: Cert-XAA represent the certificate for CA X, that includes an algorithm suite A key in its Subject Public Key Info field (first A) and is signed using the algorithm suite A (second A).

4.4. Phase 1

Phase 1 starts at the CA Ready Algorithm B Date. During the Phase 1, ALL (non-leaf) CAs MUST be ready to process a request from a child CA to issue a certificate containing an Algorithm B key for signature. In order to perform this task, the issuing CA MUST be able to demonstrate the subject's PoP by verifying the certificate request's signature, which MAY be calculated using the signature algorithm B.

During this phase, the issuing CA is NOT required to sign the certificates it issues using the Algorithm suite B. Consequently, it is possible to issue certificates signed using the Algorithm suite A, but that have a "Subject Public Key Info" field that correspond to the Algorithm suite B. Also, the issuing CA MAY receive requests with the same Subject field but different Subject Public Key Info fields (for either the Algorithm A or B suite).

During the complete transition process, all CA MUST NOT sign a CSR that includes an Algorithm Suite A "Subject Public Key Info" using the Algorithm Suite B.

(Note 1: Now we may have an issue with the decision from the WG of not using the "top-down" model. If I am a subordinate CA and my

parent has the possibility of issuing certificates with both, the A and B algorithm suite, how does a request indicate the algorithm suite that should be used? We may need to modify the provisioning protocol to indicate which algorithm suite to use. In the following sections, I assume that the subordinate CA has the ability to chose which Algorithm Suite it will use).

(Note 2: Also in the provisioning protocol, it looks that it would be great to have the possibility to question the server on what algorithm suite it supports in order to not depend in any off-band communication).

A RP is not required to modify its behavior during the Phase 1. However, a RP MAY validate the signed objects signed using the Algorithm suite B. A RP that choses to do so MUST consider as equally valid any validated signed object signed with Algorithm suite A or B.

The objective of this phase is to allow a subordinate CA to start its transition to the Algorithm Suite B, although its parent CA is not ready to issue certificates using the new suite. In our example if CA Y would like to start the transition, it should send a certificate request that includes an Algorithm Suite B key in its "Subject Public Key Info". CA X will be able to verify the signature of the certificate request and issue the certificate that, in this case, is signed using Algorithm Suite A. CA Y then will be able to issue a certificate signed by Algorithm Suite B to CA Z, and to generate any signed object using the new algorithm suite. CA Y will have two certificates with the same resources, both signed with the same key from CA X. CA Z may have three certificates, depending on which validation paths CA Z would like to enable. The typical certification hierarchy is shown in this figure:

```
CA X-Certificate-Algorithm-Suite-A (Cert-XAA)
|
|-> CA-Y-Certificate-Algorithm-Suite-A (Cert-YAA)
|   |
|   |-> CA-Z-Certificate-Algorithm-Suite-A (Cert-ZAA)
|   |   |
|   |   |-> CA-Z-Signed-Objects-Algorithm-Suite-A
|   |   |-> CA-Z-Certificate-Algorithm-Suite-B (Cert-ZBA)
|   |   |   |
|   |   |   |-> CA-Z-Signed-Objects-Algorithm-Suite-B
|   |   |   |-> CA-Y-CRL-Algorithm-Suite-A (CRL-ZA)
|   |   |   |-> CA-Y-Other-Signed-Objects-Algorithm-Suite-A
|   |-> CA-Y-Certificate-Algorithm-Suite-B (Cert-YBA)
|       |
|       |-> CA-Z-Certificate-Algorithm-Suite-B (Cert-ZBB)
|       |   |
|       |   |-> CA-Z-Signed-Objects-Algorithm-Suite-B
|       |   |-> CA-Y-CRL-Algorithm-Suite-B (CRL-YB)
|       |-> CA-Y-Other-Signed-Objects-Algorithm-Suite-B
```

In the example of the figure, an RP that can validate signed product

sets using either algorithm suites, and that is configure with Certificate Cert-XAA as its Trust Anchor (TA) would be able to validate the CA Y signed product sets signed using Algorithm Suite B, by using the following validation path: Cert-XAA --> Cert-YBA.

The RP also will be able to validate CA Z signed product sets signed using Algorithm Suite B, by using either of the following validation paths:

Cert-XAA --> Cert-YAA --> Cert-ZBA-> CA-Z-Signed-Objects-Algorithm-Suite-B

or

Cert-XAA --> Cert-YBA --> Cert-ZBB-> CA-Z-Signed-Objects-Algorithm-Suite-B

If any CA in the example were in the process of a key rollover (using the same algorithm suite), more validation paths would be possible.

4.5. Phase 2

Phase 2 starts at the CA Set Algorithm B Date. During this phase, ALL CAs MUST be able to issue a certificate signed under the Algorithm B suite to a child CA that requests such. Also, every CA MUST issue its CRLs using both Algorithm suites (A and B).

At this phase a subordinate CA has the ability to choose which Algorithm suite should be used by the issuing CA to generate the requested certificate. (Note: here we should add a reference to the mechanism that should be part of the provisioning protocol).

Every CA MUST issue a CRL for each CA Certificate associated with the CA. Each CRL should be signed with the correspondent algorithm suite.

The same comment for RP behaviour during Phase 1 is valid during Phase 2.

In our three CA example, CA A will be able to issue certificates using Algorithm Suite B:

```

CA X-Certificate-Algorithm-Suite-A (Cert-XAA)
|
|-> CA-Y-Certificate-Algorithm-Suite-A (Cert-YAA)
|   |-> CA-Z-Certificate-Algorithm-Suite-A (Cert-ZAA)
|   |   |-> CA-Z-Signed-Objects-Algorithm-Suite-A
|   |   |-> CA-Z-Certificate-Algorithm-Suite-B (Cert-ZBA)
|   |   |   |-> CA-Z-Signed-Objects-Algorithm-Suite-B
|   |   |-> CA-Y-CRL-Algorithm-Suite-A (CRL-ZA)
|   |   |-> CA-Y-Other-Signed-Objects-Algorithm-Suite-A
|-> CA-Y-Certificate-Algorithm-Suite-B (Cert-YBA)
|   |-> CA-Z-Certificate-Algorithm-Suite-B (Cert-ZBB)
|   |   |-> CA-Z-Signed-Objects-Algorithm-Suite-B
|   |   |-> CA-Y-CRL-Algorithm-Suite-B (CRL-YB)
|   |   |-> CA-Y-Other-Signed-Objects-Algorithm-Suite-B

CA X-Certificate-Algorithm-Suite-B (Cert-XBB)
|-> CA-Y-Certificate-Algorithm-Suite-B (Cert-YBB)
|   |-> CA-Z-Certificate-Algorithm-Suite-B (Cert-ZBB)
|   |   |-> CA-Z-Signed-Objects-Algorithm-Suite-B
|   |   |-> CA-Y-CRL-Algorithm-Suite-B (CRL-YB)
|   |   |-> CA-Y-Other-Signed-Objects-Algorithm-Suite-B

```

In this example, the certificate requests for Cert-YBB and Cert-YBA are the same, the only difference is that, in one case, the certificate is issued signed using Algorithm-Suite A (Cert-YBA) and in the other case using Algorithm-Suite B (Cert-YBB). This architecture simplifies the operation of the CA-Y and the structure of the repository system.

A RP that can validate signed product sets using Algorithm Suite B, could use Cert-XBB as trust anchor and validate all CA Y and CA Z signed product sets using only the new suite. At this stage not all CAs are required to issue signed product sets using Algorithm Suite B and it is not expected that an RP uses only Algorithm Suite B trust anchors.

4.6. Phase 3

Phase 3 starts at the CA Go Algorithm B Date. During this phase all signed product sets are available either using Algorithm Suite A or Algorithm Suite B. At this phase, RPs are not yet required to validate the sets issued using Algorithm Suite B.

At this phase, RPs are still required to be able to validate signed product sets using only the Algorithm Suite A.

An RP that validates all signed product sets using exclusively either Algorithm Suite A or Algorithm Suite B, should expect the same

results.

4.7. Phase 4

Phase 4 starts at the RP Ready Algorithm B Date. During this phase, all signed product sets are available using both algorithm suites and ALL RPs MUST be able to validate them using either suite.

4.8. Phase 5

Phase 5 starts at the Algorithm B Twilight Date. At that date, the Algorithm A is labeled as "old" and the Algorithm B is labeled as "current":

A->C
B->A

During this phase, all signed product sets MUST use using the current Algorithm Suite A and MAY be used using the old Algorithm Suite C. Also, every RP MUST validate signed product sets using the current Algorithm Suite A, but also MAY validate signed product sets using the old Algorithm Suite C.

4.9. Phase 0

Phase 0 starts at the EOL Algorithm Date. At this phase, ALL signed product sets under using Algorithm Suite C MUST be considered invalid.

This phase closes the loop as Algorithm suite A is the only required algorithm suite in RPKI.

5. Synchronization issues during the algorithm transition

TBC

5.1. Signed Objects Information

TBC

5.2. Revocations

TBC

5.3. Key rollover

TBC

5.4. Repository structure

TBC

6. IANA Considerations

No IANA requirements

7. Security Considerations

TBC

8. Acknowledgements

9. References

9.1. Normative References

- [I-D.ietf-sidr-cp]
Kent, S., Kong, D., Seo, K., and R. Watro, "Certificate Policy (CP) for the Resource PKI (RPKI)", draft-ietf-sidr-cp-14 (work in progress), October 2010.
- [I-D.ietf-sidr-keyroll]
Huston, G., Michaelson, G., and S. Kent, "CA Key Rollover in the RPKI", draft-ietf-sidr-keyroll-01 (work in progress), September 2010.
- [I-D.ietf-sidr-repos-struct]
Huston, G., Loomans, R., and G. Michaelson, "A Profile for Resource Certificate Repository Structure", draft-ietf-sidr-repos-struct-05 (work in progress), October 2010.
- [I-D.ietf-sidr-res-certs]
Huston, G., Michaelson, G., and R. Loomans, "A Profile for X.509 PKIX Resource Certificates", draft-ietf-sidr-res-certs-19 (work in progress), October 2010.
- [I-D.ietf-sidr-rescerts-provisioning]
Huston, G., Loomans, R., Ellacott, B., and R. Austein, "A Protocol for Provisioning Resource Certificates", draft-ietf-sidr-rescerts-provisioning-07 (work in progress), October 2010.
- [I-D.ietf-sidr-rpki-algs]
Huston, G., "A Profile for Algorithms and Key Sizes for use in the Resource Public Key Infrastructure", draft-ietf-sidr-rpki-algs-03 (work in progress), October 2010.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC2560] Myers, M., Ankney, R., Malpani, A., Galperin, S., and C. Adams, "X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP", RFC 2560, June 1999.
- [RFC3779] Lynn, C., Kent, S., and K. Seo, "X.509 Extensions for IP Addresses and AS Identifiers", RFC 3779, June 2004.

- [RFC4193] Hinden, R. and B. Haberman, "Unique Local IPv6 Unicast Addresses", RFC 4193, October 2005.
- [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, May 2008.

9.2. Informative References

- [RFC5781] Weiler, S., Ward, D., and R. Housley, "The rsync URI Scheme", RFC 5781, February 2010.

Appendix A. Change Log

Authors' Addresses

Roque Gagliano
Cisco Systems
Avenue des Uttins 5
Rolle, 1180
Switzerland

Email: rogaglia@cisco.com

Stephen Kent
BBN Technologies
10 Moulton St.
Cambridge, MA 02138
USA

Email: kent@bbn.com

Sean Turner
IECA, Inc.
3057 Nutley Street, Suite 106
Fairfax, VA 22031
USA

Email: turners@ieca.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: May 12, 2011

R. Bush
IIJ
November 8, 2010

The RPKI Ghostbusters Record
draft-ymbk-ghostbusters-01

Abstract

In the Resource Public Key Infrastructure (RPKI), resource certificates completely obscure names or any other information which might be useful for contacting responsible parties to deal with issues of certificate expiration, maintenance, roll-overs, compromises, etc. This draft describes the RPKI Ghostbusters Record containing human contact information to be signed (indirectly) by a resource-owning certificate. The data in the record are those of a severely profiled vCARD.

Requirements Language

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

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79. This document may not be modified, and derivative works of it may not be created, and it may not be published except as an Internet-Draft.

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 <http://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 May 12, 2011.

Copyright Notice

Copyright (c) 2010 IETF Trust and the persons identified as the

document authors. All rights reserved.

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

Table of Contents

1. Introduction	3
2. Suggested Reading	3
3. RPKI Ghostbuster Record Payload Example	3
4. vCARD Profile	4
5. CMS Packaging	5
6. Validation	5
7. Security Considerations	5
8. IANA Considerations	6
9. Acknowledgments	6
10. References	6
10.1. Normative References	6
10.2. Informative References	6
Author's Address	7

1. Introduction

In the operational use of the RPKI it can become necessary to contact, human to human, the party responsible for a resource-owning certificate. The primary example of this need is when the owner of a Route Origin Authorizations (ROA) sees that a upstream certificate in the chain needed to validate the ROA is soon to expire or a CRL associated with the certificate is stale, thus placing the quality of the routing of the address space described by the ROA in jeopardy.

As the names in RPKI certificates are intentionally obscured hashes, see [I-D.ietf-sidr-cp], there is no way to use the certificate itself to lead to the certificate's maintainer. So, "Who do you call?"

This document specifies the RPKI Ghostbusters Record, signed indirectly by the certificate whose maintainer needs to be contacted, which contains human usable contact information for that maintainer.

This document and the Ghostbusters Record conform to the syntax defined in [I-D.achi-rpki-signed-object].

Note that this record is not an identity certificate, but an attestation to the contact data made by the holder of the signing certificate, and its parent if that is an EE certificate.

This record is not meant to supplant or be used as resource registry whois data. It gives information about a certificate maintainer not a resource holder.

This specification has three main sections. The first, Section 4, is the format of the contact payload information, a severely profiled vCARD. The second, Section 5, profiles the packaging of the payload as a profile of the RPKI Signed Object Template specification [I-D.achi-rpki-signed-object]. The third, Section 6, describes the proper validation of the signed Ghostbusters Record.

2. Suggested Reading

It is assumed that the reader understands the RPKI, [I-D.ietf-sidr-arch], the RPKI Repository Structure, [I-D.ietf-sidr-repos-struct], Signed RPKI Objects, [I-D.achi-rpki-signed-object], and vCARDS [RFC2426].

3. RPKI Ghostbuster Record Payload Example

An example of an RPKI Ghostbusters Record payload with all fields

used is as follows:

```
BEGIN:vCard
VERSION:3.0
FN:Human's Name
ORG:Organizational Entity
ADR;TYPE=WORK;;;42 Twisty Passage;Deep Cavern; WA; 98666;U.S.A.
TEL;TYPE=VOICE,MSG,WORK:+1-666-555-1212
TEL;TYPE=FAX,WORK:+1-666-555-1213
EMAIL;TYPE=INTERNET:human@example.com
END:vCard
```

4. vCARD Profile

The goal in profiling the vCARD is not to include as much information as possible, but rather to include as few fields as possible while providing the minimal necessary data to enable one to contact the maintainer of the RPKI data which threatens the ROA[s] of concern.

The Ghostbusters vCARD payload is a minimalist subset of the vCARD as described in [RFC2426].

BEGIN - pro forma packaging which MUST be the first line in the vCARD and MUST have the value "BEGIN:vCARD" as described in [RFC2426].

VERSION - pro forma packaging which MUST be the second line in the vCARD and MUST have the value "VERSION:3.0" as described in 3.6.9 of [RFC2426].

FN - the name, as described in 3.1.1 of [RFC2426], of a contactable person who responsible for the certificate.

ORG - an organization as described in 3.5.5 of [RFC2426].

ADR - a postal address as described in 3.2.1 of [RFC2426].

TEL - a voice and/or fax phone as described in 3.3.1 of [RFC2426].

EMAIL - an Email address as described in 3.3.2 of [RFC2426]

END - pro forma packaging which MUST be the last line in the vCARD and MUST have the value "END:vCARD" as described in [RFC2426].

The BEGIN, VERSION, and END lines MUST be included in a record. To be useful, FN and one or more of ADR, TEL, and EMAIL SHOULD be included.

5. CMS Packaging

The Ghostbusters Record is a CMS signed-data object conforming to the RPKI Signed Data Object Template, [I-D.achi-rpki-signed-object].

The ASN.1 for the ContentType is as follows:

```
rpkiGhostbusters ::= SEQUENCE {
    Version      [0] INTEGER DEFAULT 0,
    vCard        VCARD}

VCARD           ::= PrintableString
```

The ContentType of a Ghostbusters Record is defined as rpkiGhostbusters, and has the numerical value of TO BE ASSIGNED. This OID MUST appear both within the eContentType in the encapContentInfo object as well as the ContentType signed attribute in the signerInfo object. See [I-D.achi-rpki-signed-object].

eContent: The content of a Ghostbusters Record is described above in Section 4 above.

Similarly to a ROA, the Ghostbusters Record is usually signed by an end-entity certificate which is, in turn, signed by the resource-holding certificate whose maintainer is described in the vCARD.

6. Validation

The validation procedure defined in Section 3 of [I-D.achi-rpki-signed-object] is applied to a Ghostbusters Record. After this procedure has been performed, the Version number field within the payload is checked, and the OCTET STRING containing the vCARD data is extracted. These data are checked against the profile defined in Section 4 of this document. Only if all of these checks pass is the Ghostbusters payload deemed valid and made available to the application that requested the payload.

7. Security Considerations

Though there is no on the wire protocol in this specification, there are attacks which could abuse the data described. As the data, to be useful, need to be public, little can be done to avoid this exposure.

Phone Numbers: The vCARDS may contain real world telephone numbers which could be abused for telemarketing, abusive calls, etc.

Email Addresses: The vCARDS may contain Email addresses which could be abused for purposes of spam.

Relying parties are warned that the data in a Ghostbusters Record are self-asserted. These data have not been verified by the CA that issued a (CA) certificate to the entity that issued the EE certificate used to validate the Ghostbusters Record.

8. IANA Considerations

This document has no IANA Considerations.

9. Acknowledgments

The author wishes to thank Russ Housley for suggesting profiling the vCARD specification, the authors of [I-D.achi-rpki-signed-object], and particularly Steven Kent.

10. References

10.1. Normative References

- [I-D.achi-rpki-signed-object]
Lepinski, M., Chi, A., and S. Kent, "Signed Object Template for the Resource Public Key Infrastructure", draft-achi-rpki-signed-object-00 (work in progress), August 2010.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC2426] Dawson, F. and T. Howes, "vCard MIME Directory Profile", RFC 2426, September 1998.

10.2. Informative References

- [I-D.ietf-sidr-arch]
Lepinski, M. and S. Kent, "An Infrastructure to Support Secure Internet Routing", draft-ietf-sidr-arch-11 (work in progress), September 2010.
- [I-D.ietf-sidr-cp]

Kent, S., Kong, D., Seo, K., and R. Watro, "Certificate Policy (CP) for the Resource PKI (RPKI)", draft-ietf-sidr-cp-15 (work in progress), October 2010.

[I-D.ietf-sidr-repos-struct]

Huston, G., Loomans, R., and G. Michaelson, "A Profile for Resource Certificate Repository Structure", draft-ietf-sidr-repos-struct-05 (work in progress), October 2010.

Author's Address

Randy Bush
Internet Initiative Japan, Inc.
5147 Crystal Springs
Bainbridge Island, Washington 98110
US

Phone: +1 206 780 0431 x1
Email: randy@psg.com

Network Working Group
Internet-Draft
Intended status: BCP
Expires: May 12, 2011

R. Bush
IIJ
November 8, 2010

RPKI-Based Origin Validation Operations
draft-ymbk-rpki-origin-ops-00

Abstract

Deployment of the RPKI-based BGP origin validation has many operational considerations. This document attempts to collect and present them. It is expected to evolve as RPKI-based origin validation is deployed and the dynamics are better understood.

Requirements Language

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

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79. This document may not be modified, and derivative works of it may not be created, and it may not be published except as an Internet-Draft.

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 <http://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 May 12, 2011.

Copyright Notice

Copyright (c) 2010 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents

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

Table of Contents

1. Introduction	3
2. Suggested Reading	3
3. RPKI Distribution and Maintenance	3
4. Within a Network	4
5. Routing Policy	4
6. Notes	5
7. Security Considerations	5
8. IANA Considerations	6
9. Acknowledgments	6
10. References	6
10.1. Normative References	6
10.2. Informative References	7
Author's Address	7

1. Introduction

RPKI-based origin validation relies on widespread propagation of the Resource Public Key Infrastructure (RPKI) [I-D.ietf-sidr-arch]. How the RPKI is distributed and maintained globally is a serious concern from many aspects.

The global RPKI has yet to be deployed, only a testbed exists, and some beta testing is being done by the IANA and some RIRs. It is expected to be deployed incrementally over a number of years. It is thought that origin validation based on the RPKI will deploy over the next year to five years.

Origin validation only need be done by an AS's border routers and is designed so that it can be used to protect announcements which are originated by large providers, upstreams and downstreams, and by small stub/enterprise/edge routers.

Origin validation has been designed to be deployed on current routers without hardware upgrade. It should be used by everyone from large backbones to small stub/enterprise/edge routers.

RPKI-based origin validation has been designed so that, with prudent local routing policies, there is no liability that normal Internet routing is threatened by unprudent deployment of the global RPKI, see Section 5.

2. Suggested Reading

It is assumed that the reader understands BGP, [RFC4271], the RPKI, see [I-D.ietf-sidr-arch], the RPKI Repository Structure, see [I-D.ietf-sidr-repos-struct], ROAs, see [I-D.ietf-sidr-roa-format], the RPKI to Router Protocol, see [I-D.ymbk-rpki-rtr-protocol], and RPKI-based Prefix Validation, see [I-D.pmhapat-sidr-pfx-validate].

3. RPKI Distribution and Maintenance

The RPKI is a distributed database containing certificates, CRLs, manifests, ROAs, and Ghostbuster Records as described in [I-D.ietf-sidr-repos-struct]. Policies and considerations for RPKI object generation and maintenance are discussed elsewhere.

A local valid cache containing all RPKI data may be gathered from the global distributed database using the rsync protocol and a validation tool such as rcynic.

Validated caches may also be created and maintained from other validated caches. An operator should take maximum advantage of this feature to minimize load on the global distributed RPKI database.

As RPKI-based origin validation relies on the availability of RPKI data, operators will likely want border routers to have one or more nearby caches.

For redundancy, a router may peer with more than one cache at the same time. Peering with two or more, one local and others remote, is recommended.

If an operator or site trusts upstreams to carry their traffic, they might as well trust the RPKI data those upstreams cache and feed off of those caches. Note that this places an obligation on those upstreams to maintain fresh and reliable caches.

A transit provider or a network with peers will want to validate origins in announcements made by downstreams and peers. They still may choose to trust the caches provided by their upstreams.

4. Within a Network

Origin validation need only be done by edge routers in a network, those which border other networks/ASs.

A validating router will use the result of origin validation to influence local policy within its network, see Section 5. In deployment this policy should fit into the AS's existing policy, preferences, etc. This allows a network to incrementally deploy validation capable border routers.

eBGP speakers which face more critical peers or up/downstreams would be candidates for the earliest deployment. Validating more critical received announcements should be considered in partial deployment.

5. Routing Policy

Origin validation based on the RPKI merely marks a received announcement as having an origin which is Validated, Unknown, or Invalid. How this is used in routing is up to the router operator's local policy. See [I-D.pmohapat-sidr-pfx-validate].

Reasonable application of local policy should be designed eliminate the threat of unroutability of prefixes due to ill-advised or incorrect certification policies.

As origin validation will be rolled out over years coverage will be spotty for a long time. Hence a normal operator's policy should not be overly strict, perhaps preferring valid announcements and giving very low preference, but still using, invalid announcements.

Some may choose to use the large Local-Preference hammer. Others might choose to let AS-Path rule and set their internal metric, which comes after AS-Path in the BGP decision process.

Certainly, routing on unknown validity state will be prevalent for a long time.

Until the community feels comfortable relying on RPKI data, routing on invalid origin validity, though at a low preference, may be prevalent for a long time.

Announcements with valid origins SHOULD be preferred over those with unknown or invalid origins.

Announcements with unvalidatable origins SHOULD be preferred over those with invalid origins.

Announcements with invalid origins MAY be used, but SHOULD be less preferred than those with valid or unknown.

6. Notes

Like the DNS, the global RPKI presents only a loosely consistent view, depending on timing, updating, fetching, etc. Thus, one cache or router may have different data about a particular prefix than another cache or router. There is no 'fix' for this, it is the nature of distributed data with distributed caches.

There is some uncertainty about the origin AS of aggregates and what, if any, ROA can be used. The long range solution to this is the deprecation of AS-SETs, see [I-D.wkumari-deprecate-as-sets].

7. Security Considerations

As the BGP origin is not signed, origin validation is open to malicious spoofing. It is only designed to deal with inadvertent mis-advertisement.

Origin validation does nothing about AS-Path validation and therefore is open to monkey in the middle path attacks.

The data plane may not follow the control plane.

8. IANA Considerations

This document has no IANA Considerations.

9. Acknowledgments

The author wishes to thank Rob Austein, Steve Bellovin, Pradosh Mohapatra, Chris Morrow, Keyur Patel, Heather and Jason Schiller, John Scudder, and Dave Ward.

10. References

10.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [I-D.ietf-sidr-arch]
Lepinski, M. and S. Kent, "An Infrastructure to Support Secure Internet Routing", draft-ietf-sidr-arch-11 (work in progress), September 2010.
- [I-D.ietf-sidr-repos-struct]
Huston, G., Loomans, R., and G. Michaelson, "A Profile for Resource Certificate Repository Structure", draft-ietf-sidr-repos-struct-05 (work in progress), October 2010.
- [I-D.ietf-sidr-roa-format]
Lepinski, M., Kent, S., and D. Kong, "A Profile for Route Origin Authorizations (ROAs)", draft-ietf-sidr-roa-format-07 (work in progress), July 2010.
- [I-D.ymbk-rpki-rtr-protocol]
Bush, R. and R. Austein, "The RPKI/Router Protocol", draft-ymbk-rpki-rtr-protocol-06 (work in progress), July 2010.
- [I-D.pmohapat-sidr-pfx-validate]
Mohapatra, P., Scudder, J., Ward, D., Bush, R., and R. Austein, "BGP Prefix Origin Validation", draft-pmohapat-sidr-pfx-validate-07 (work in progress),

April 2010.

[I-D.wkumari-deprecate-as-sets]

Kumari, W., "Deprecation of BGP AS_SET, AS_CONFED_SET.",
draft-wkumari-deprecate-as-sets-01 (work in progress),
September 2010.

10.2. Informative References

[RFC4271] Rekhter, Y., Li, T., and S. Hares, "A Border Gateway
Protocol 4 (BGP-4)", RFC 4271, January 2006.

Author's Address

Randy Bush
Internet Initiative Japan, Inc.
5147 Crystal Springs
Bainbridge Island, Washington 98110
US

Phone: +1 206 780 0431 x1
Email: randy@psg.com

INTERNET-DRAFT
Intended Status: Proposed Standard
Expires: April 28, 2011

Mingui Zhang
Huawei
Bin Liu
Tsinghua University
Dacheng Zhang
Huawei
Beichuan Zhang
The University of Arizona
October 25, 2010

Secure Extension of BGP by Decoupling Path Propagation and Adoption
draft-zhang-idr-decoupling-01

Status of this Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

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."

The list of current Internet-Drafts can be accessed at
<http://www.ietf.org/lid-abstracts.html>

The list of Internet-Draft Shadow Directories can be accessed at
<http://www.ietf.org/shadow.html>

Copyright Notice

Copyright (c) 2010 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must

include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Abstract

This draft proposes a novel mitigation scheme to protect the inter-domain data delivery during false routing announcements. A new path attribute is defined to Decouple propagation of a path and adoption of a path for data forwarding in BGP (DBGP). DBGP does not use suspicious paths for data forwarding, but still propagates them in the routing system to facilitate attack detection. It can extensively protect data delivery from routing announcements of false sub-prefixes, false origins, false nodes and false links and works well with ongoing attack detection and prevention systems.

Conventions used in this document

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

Table of Contents

1	Introduction	5
2	Terminology	5
3	False Routing Announcements	5
3.1	Problem	5
3.2	Countermeasures	6
3.2.1	Prevention	6
3.2.2	Detection	6
3.2.3	Traditional Mitigation	8
3.3	Paradox of Blocking Suspicious Routing Updates and Attack Detection	8
4.	DBGP's Mitigation: Decoupling Path Propagation and Adoption	9
4.1	Quarantine Time	9
4.2	Working with Detection and Prevention	10
5	Protocol Descriptions	10
5.1	DAS_PATH Attribute Definition	11
5.2	Identification of Suspicious Paths	12
5.3	Propagating Updates	14
5.4	Choosing Alternative Paths	14
5.5	Releasing Quarantined Paths	15
6	Security Considerations	15
7	IANA Considerations	15
8	References	15
8.1	Normative References	15
8.2	Informative References	16
	Appendix A: Empirical Evaluation	16
	Author's Addresses	17

1 Introduction

False routing announcements cause serious security issues to the inter-domain routing system, which can lead to widespread service disruptions. A special case is prefix hijacking, in which a network announces an IP prefix that belongs to another network. Existing works such as Pretty Good BGP (PGBGP)[PGBGP] block suspicious routing updates to protect data delivery. However, such an approach has the side effect of blocking the view of detection systems at the control plane and data plane. As a result, an attack will not be detected and operators will not be alerted to take actions to stop the attack. This draft proposes an extension of BGP, to solve this paradox by decoupling path propagation and adoption in BGP.

In current BGP, the path a router adopts for data forwarding is the same path being propagated to neighbors. That is why upon receiving a suspicious path, a router has to either accept it (no mitigation but good detection) or reject it (good mitigation but no detection). Our idea is for a router to use trusted paths for data forwarding, but still inform its neighbors about the suspicious path. The suspicious paths will be carried in an optional transitive attribute in BGP updates, while the routers still use trusted paths for data forwarding. This way the data traffic is protected while false routing announcements are being propagated to detection systems.

2 Terminology

DBGP: Decoupling path propagation and adoption in BGP

3 False Routing Announcements

3.1 Problem

False routing announcements can be caused by either inadvertent misconfigurations or malicious attacks. For the ease of exposition, we use "attacker" to refer to the network (or Autonomous System, AS) that makes the false routing announcements regardless of the intention. Based on which part of the routing path is false, such announcements can be classified into five types, each of which has different severity:

- o Prefix origin: The attacker originates someone else's prefix. Depending on where a network is in the Internet topology, some will choose paths leading to the true origin and some will choose paths leading to the false origin.
- o Intermediate node: The attacker does not forge the prefix or its origin, but inserts itself into the path as an intermediate

AS. Similar to the case of false origin, some networks will choose the correct paths and some the false paths. But usually fewer networks will choose the false path since it is longer than that of false origin attack.

- o Intermediate link: The attacker forges a new link to bypass some of the ASes to get a shorter path. The shortened path is expected to attract more networks' traffic.
- o Sub-prefix: The attacker originates a sub-prefix of someone else's prefix. Due to the longest match in routing lookup, a false sub-prefix will win over the original prefix. Thus, all traffic destined to the sub-prefix range will be forwarded to the attacker.
- o Super-prefix: Theoretically the attacker can also announce a false super-prefix, but that will not attract any traffic unless part of the prefix range is unused by the real owner, in which case it is equivalent to announcing a false origin of the unused prefix range.

3.2 Countermeasures

The current strategies proposed for the problem of false announcements fall in three categories: prevention, detection and mitigation.

3.2.1 Prevention

Prevention schemes (e.g., [SBGP], [SoBGP], and [SPV]) use cryptographic mechanisms to protect the routing updates and let routers reject any forged announcement. Unfortunately no prevention scheme has seen much deployment on the Internet due to the lack of incentives for those first movers. Since crypto-based schemes add significant computational load to routers and require upgrade on software or hardware, individual ISPs need to see immediate benefits to justify the deployment. On the current Internet, however, the first mover's routing announcements will be accepted by other networks without authentication, and adding authentication does not bring any immediate benefit.

3.2.2 Detection

The representative detection systems proposed in recent years include [Cyclops], [PHAS], [MyASN], [IAR], [iSPY], [NWatch], [OList], and [LWeight]. Such systems are designed to detect false routing by examining routing updates, probing data paths, cross-checking with registry databases, or a combination of these techniques. Once a

false routing case is detected, the owner of the prefix will be notified, and it is expected that the owner will take actions to resolve the problem, which, in today's Internet, usually involves contacting the offending network or its upstream provider to stop the false routing announcements. This process of detection, notification and resolution takes time, ranging from an hour to a day in some past incidents and varying from network to network [NWatch][RIPE]. In the meantime, the damage to data traffic has already been made and malicious attackers may have already achieved their objectives.

3.2.3. Traditional Mitigation

A mitigation schemes attempts to protect the data traffic while an attack is going on. The common approach is to somehow identify abnormal routing updates and block them. As proposed in [PGBGP] and [PGBGP++], a router can examine the content of incoming updates. If an AS path contains unexpected prefix origin or links, it will be suppressed from propagating for a period of time to wait for the network operator's validation. The length of the time is configurable by the network operators. Instead, an alternative path (via trusted prefix origin and links) will be employed for data delivery in the suppression period. After this period, if the path is not proved to be illegal, the router will adopt and announce the new path. PGBGP gives operators certain reaction time to resolve potential false announcements while protecting data delivery in the meantime. When a real attack is detected and resolved, the corresponding false announcement will be withdrawn from the routing system, whereas legitimate announcement will stay in the routing system and eventually be accepted.

But blocking false routing announcements can get in the way of detection systems. For instance, on September 22, 2008, a Russian ISP AS8997 hijacked a large number of prefixes as it leaked an entire table [ASN8997]. However, since the upstream ISP of AS8997 blocked the routing updates, detection systems such as MyASN and IAR did not pick up this incident. The attack mainly affected ISPs and users within Russia but largely went unnoticed by prefix owners.

Each existing solution has its drawbacks, and none is sufficiently effective by itself. The future of the Internet probably will have several different solutions at the same time, complementary to each other, forming a multi-line defense to protect routing and data delivery before, during, and after attacks.

3.3 Paradox of Blocking Suspicious Routing Updates and Attack Detection

It has been realized that a mitigation mechanism and a detection system that complement each other well can be integrated into an effective routing defense solution. For instance, the mitigation mechanism can help the detection system to confine the damage caused by an attack, as the affected data traffic may be vulnerable for hours before the attack can be actually detected by the detection mechanism and be eventually stopped. Also, the mitigation mechanism can also obtain benefit from the detection system because a mitigation mechanism normally cannot identify false routing information accurately with its limited knowledge, resource and time. The output from the detection system can be used to correct the many false positives generated by the mitigation mechanism and also inform

the prefix owner to resolve the attack.

However, there is a dilemma: the mitigation mechanism tries to render the attack ineffective while the detection system needs the attack to be effective in order to detect it.

4. DBGP's Mitigation: Decoupling Path Propagation and Adoption

The suspicious paths are serially blocked hop by hop for validation in traditional mitigation schemes, which gets in the way of detection systems. In order to solve this dilemma, this document proposes a solution which decouples path propagation and path adoption. The basic idea of this solution is to extend BGP's update message with a new optional transitive path attribute so that a router can inform its neighbor routers about the suspicious path and meanwhile the router uses another trusted path for data forwarding. In order to achieve this, a new BGP attribute, DAS_PATH, is defined in Section 5.

4.1 Quarantine Time

Based on operating experiences, a false routing announcement can be detected and corrected in a certain period (e.g., one day) after it is launched, and so an announcement will be trusted if it is not withdrawn in a pre-defined period. Therefore, in mitigation schemes, when a new path is identified as a suspicious one, it will be quarantined (or blocked) from being used for a period of time, which is called the quarantine time and noted as T_q . If a new path has stayed in a router's Adj-RIBs-In for more than T_q , it will be trusted by this router. If this path is the most preferred from the Adj-RIBs-In, the router will use this path for data delivery and announce this path to its neighbors.

A router MAY determine the quarantine time itself. Assume there are two routers, R1 and R2. R1 is the downstream of R2, and the quarantine time of R1 and R2 is T1 and T2 respectively. The suspicious path is PATH at R1. There are two possibilities.

- o T1 is shorter than T2. When T1 expires, PATH becomes trusted by R1 and R1 begins to use it for data delivery. R1 will announce R1-PATH as a legitimate AS path to R2. However, at this time, the path R1-PATH is still being quarantined by R2.
- o T1 is longer than T2. When T2 expires, R1-PATH becomes trusted by R2. However, R2 can not use this path for data delivery as R1 has not announced R1-PATH as an AS path. It will be cached in the Adj-RIBs-In until the downstream router R1 has announced it as an AS path when T1 expires.

Compared to the traditional mitigation schemes, the propagation of suspicious paths through DAS_PATHs in DBGP enables the parallel validation, which accelerates the adoption process of suspected legitimate paths (the false positive).

4.2 Working with Detection and Prevention

The traditional mitigation mechanisms block the propagation of suspicious paths, which get in the way of detection systems. DBGP is proposed to address this shortage and to recover the transfer passage of suspicious paths as that in BGP. In DBGP, the data traffic is protected and at the same time, the false routing announcements are propagated which can be monitored by the detection systems. The capacity of propagating the suspicious path in DBGP should be equivalent to that in BGP. However, the issue of enhancing this capacity is out of the scope of this document.

Mitigation schemes only validate the AS_PATHs according to the limited information stored in the Adj-RIBs-In. This would cause some legitimate paths to be identified as suspicious and blocked from being used for data delivery (false positive). If a down stream router would like their paths be adopted quickly, it can add authentication to its AS_PATHs. Then, the up stream routers can validate and adopt this authenticated AS_PATH immediately after receiving them. Therefore, the deployment of DBGP actually creates the incentive for deploying prevention systems.

5 Protocol Descriptions

Take Figure 5.1 for example. A, B, C, and O are DBGP routers residing in different ASes, , X is the attacker and p is the prefix of interest. Before the attack, the preferred path for traffic is ABCO. Here, we use the notation "R1R2...Rn-p" to denote the AS path which is destined to prefix "p" via routers R1, R2, ..., Rn. When X makes a false announcement of X-p to B, B will regard this new path as suspicious because it would divert traffic to an AS(X) that previously was not on the data path (BCO). B will store the suspicious path in its Adj-RIBs-In (The routing tables of an AS router is comprised of three sub-tables: Adj-RIBs-In, Loc-RIB and Adj-RIBs-Out [BGP4]), but keep using the existing path in its Loc-RIB for data forwarding. At the same time, B re-announces its path (BCO) in an update message to A, and encapsulates the new, suspicious path (BX) as an optional transitive attribute which is defined in Section 5.1. After receiving the update message, router A learns this suspicious path, stores it, and propagates it further to its neighbors using the optional attribute the same way. Therefore, the suspicious path is propagated to the Internet while not adopted for data forwarding. This approach enable the detection systems to

intercept the suspicious path and notify the prefix owner to take actions. Once the attack is stopped, the false announcements will be withdrawn from the routing system, i.e., deleted or replaced in the Adj-RIBs-In of the involved routers. However, a router realizes that the quarantine time has been expired and the suspicious path is still in the Adj-RIBs-In, the router regards the path as a legitimate one. Hence the DBGP router will install the path in its Loc-RIB for data forwarding and re-announce the path using the regular AS_PATH attribute in the update message. For example, if BX-p has been validated as legitimate, B will announce it to A as its AS_PATH. The rest of this section will discuss the design details in new BGP attribute definition, identifying suspicious paths, choosing alternative paths for data forwarding, propagating the paths, and releasing quarantined paths.

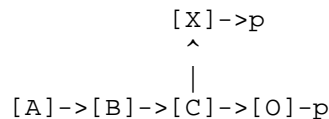


Figure 5.1: Attack Example 1

The rest of this section will discuss the design details in the definition of the new DAS_PATH Attribute, identifying suspicious paths, choosing alternative paths for data forwarding, propagating the paths, and releasing quarantined paths.

5.1 DAS_PATH Attribute Definition

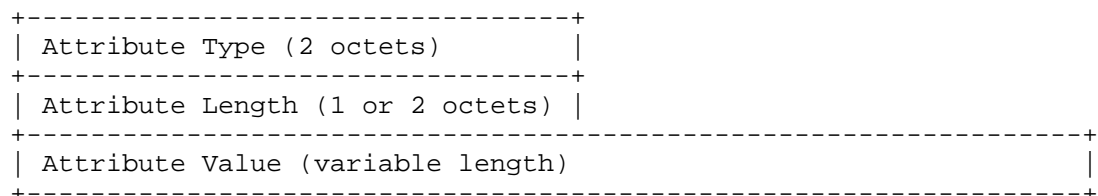


Figure 5.2: The DAS_PATH Attribute

DAS_PATH (Decoupled AS_PATH) is defined as a new optional transitive Path Attribute (Figure 5.2) to be included in BGP's UPDATE message.

The first bit of the Attribute Type is set (1), therefore the attribute is optional. The second bit of Attribute Type is set (1), therefore the attribute is transitive and SHOULD be passed on to other BGP peers. The third and fourth bits are Partial bit and Extended Length bit respectively, which has already been defined in [BGP4]. The Attribute Type code is assigned by the IANA (Internet

Assigned Numbers Authority). The definition of Attribute Length is the same as that in [BGP4].

The Attribute Value is composed of a sequence of DAS path segments. Each DAS path segment is encoded as a triple <path segment type, path segment length, path segment value>.

The path segment type is a 1-octet field with the following two allowable values:

Value	Segment	Type
1	DAS_SET	unordered set of ASs a route in the UPDATE message has traversed
2	DAS_SEQUENCE	ordered set of ASs a route in the UPDATE message has traversed

The path segment length is a 1-octet long field containing the number of ASs in the path segment value field.

The path segment value field contains one or more AS numbers, each encoded as a 2-octets long field.

When a DAS_PATH is propagated across the network, the operations on DAS_PATH follows the well-known AS_PATH attribute only that DAS_PATH is non-mandatory. If a router which does not deploy DBGp receives the update messages containing DAS_PATH attribute, i.e., does not understand this attribute, it will just pass it on to the next router. If its downstream router is a DBGp router, it will be able to pick up the information from this attribute and continue DBGp operations. Therefore DBGp can be incrementally deployed over the Internet.

5.2 Identification of Suspicious Paths

For a given prefix p , a path is trusted if it has been staying in the Adj-RIBs-In continuously for the required quarantine time, T_q . All the nodes, links, and origins that appear in trusted paths are trusted components, and the set of them is denoted by $\text{trusted}(p)$. This set of trusted components is derived from current contents of all Adj-RIBs-In without using a database to store historical information like PGBGP does. Nodes, links, and origins that do not belong to $\text{trusted}(p)$ are said to be suspicious components for this particular prefix p . A new path is suspicious if it contains any suspicious component for its prefix. However, not all suspicious paths need to be explicitly quarantined. DBGp quarantines paths that satisfy the following condition:

- o A new path is quarantined if and only if it is suspicious, more

5.3 Propagating Updates

DBGP uses the new BGP attribute, `DAS_PATH`, to carry a suspicious path in a update. In certain cases DBGp router may need to select a `DAS_PATH` from multiple ones to announce. For example, in Figure 5.4, assume C does not deploy DBGp and accepts the false announcement of X-p. When B receives BCX-p, B quarantines this new path and switches to its backup BDO-p. The update from B to A will have BDO as the `AS_PATH`, and BCX as the `DAS_PATH` attribute. However, since BDO is suspicious to A as well, A will quarantine BDO and switch to AEFO. Thus the update from A can contain either ABCX or ABDO. A router's neighbors may simultaneously send updates containing the `DAS_PATH` to it, which also makes the router have multiple `DAS_PATH`s to choose. DBGp always let a router select the best `DAS_PATH` from the multiple ones to announce, which is similar to the traditional BGP's `AS_PATH` selection process. This rule works well due to the following two facts. First, during attacks, the best `DAS_PATH` is most likely the bogus path aiming to attract data traffic. Second, during false positives, the best `DAS_PATH` will most likely become the best `AS_PATH` when the quarantine time ends.

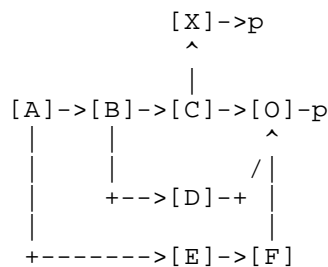


Figure 5.4: Attack Example 3

5.4 Choosing Alternative Paths

When a new path is the most preferred but suspicious, DBGp routers will use an alternative path for data delivery. The question is which alternative path to be chosen. First, if the existing path that is being used for data forwarding is still the best, then the router can stick to that path without any changes. Second, if the existing path in use will have been replaced by the suspicious path, then the router needs to pick an alternative. For example, in Figure 5.4, suppose C does not deploy DBGp and blindly accepts the false announcement X-p. B's existing path BCO-p will be replaced by a suspicious path BCX-p, therefore B needs to temporarily switch to a backup path BDO-p from its Adj-RIBs-In. Third, if there is no alternative path or all alternative paths are labeled as suspicious, then the router err on data delivery by adopting a suspicious path to

forward packets.

5.5 Releasing Quarantined Paths

If the quarantined paths are false announcements, it is likely that within T_q , the attack will be stopped and these paths being withdrawn from the routing system. In this case, there is no explicit release of the quarantined path. Just the upstream router will send an update with empty `DAS_PATH` attribute. If T_q has passed and the quarantined path is still in the `Adj-RIBs-In`, then it is more likely that this is a legitimate path. The router will treat the path as a regular path and make it go through the path selection process. If the path turns out to be the most preferred one, it will be used for data forwarding and trigger routing updates to neighbor routers.

6 Security Considerations

The entire document is about security consideration.

7 IANA Considerations

The attribute type code of `DAS_PATH` should be assigned by the IANA, which identifies the attribute uniquely from all others.

8 References

8.1 Normative References

- [PGBGP] J. Karlin, S. Forrest, and J. Rexford, "Pretty Good BGP: Improving BGP by Cautiously Adopting Routes," in Proceedings of IEEE ICNP, 2006.
- [SBGP] S. Kent, C. Lynn, and K. Seo, "Secure Border Gateway Protocol (SBGP)," IEEE Journal on Selected Areas in Communications, vol. 18, no. 4, pp. 582-592, 2000.
- [Cyclops] Y.-J. Chi, R. Olivera, and L. Zhang, "Cyclops: the as-level connectivity observatory," SIGCOMM Comput. Commun. Rev., vol. 38, no. 5, pp. 5-16, 2008.
- [PHAS] M. Lad, D. Massey, D. Pei, B. Zhang, and L. Zhang, "PHAS: A Prefix Hijack Alert System," in 15th USENIX Security Symposium, 2006, pp.153-166.
- [MyASN] "RIPE myASN System," <http://www.ris.ripe.net/myasn.html>.
- [IAR] [Online]. Available: <http://iar.cs.unm.edu/>
- [iSPY] Z. Zhang, Y. Zhang, Y. C. Hu, Z. M. Mao, and R. Bush, "iSPY: Detecting IP Prefix Hijacking on My Own," in Proceedings of ACM SIGCOMM, 2008.
- [NWatch] G. Siganos and M. Faloutsos, "Neighborhood Watch for Internet Routing: Can We Improve the Robustness of Internet Routing Today?" in Proceedings of IEEE INFOCOM,

- 2007.
- [Olist] X. Zhao, D. Pei, L. Wang, D. Massey, A. Mankin, S. Wu, and L. Zhang, "Detection of Invalid Routing Announcement in the Internet," in Proceedings of the IEEE DSN, June 2002.
- [LWeight] C. Zheng, L. Ji, D. Pei, J. Wang, and P. Francis, "A Light-weight Distributed Scheme for Detecting IP Prefix Hijacks in Real-time," in Proceedings of ACM SIGCOMM, 2007.
- [SoBGP] J. Ng, "Extensions to BGP to Support Secure Origin BGP," April 2004, <ftp://ftp-eng.cisco.com/sobgp/drafts/draft-ng-sobgp-bgp-extensions-02.txt>.
- [SPV] Y.-C. Hu, A. Perrig, and M. Sirbu, "SPV: Secure Path Vector Routing for Securing BGP," in Proceedings of ACM SIGCOMM, 2004.
- [RIPE] [Online]. Available: <http://www.ripe.net/news/study-youtubehijacking.html>
- [ASN8997] "Prefix hijack by ASN 8997." [Online]. Available: <http://www.merit.edu/mail.archives/nanog/2008-09/msg00704.html>
- [BGPpop] J. Rexford, J. Wang, Z. Xiao, and Y. Zhang, "BGP Routing Stability of Popular Destinations," in Proceedings of ACM IMC 2002, pp. 197-202.
- [Stable] K. Butler, P. McDaniel, and W. Aiello, "Optimizing BGP Security by Exploiting Path Stability," in Proceedings of ACM CCS, Alexandria, VA, United States, 2006, pp. 298-310.
- [BGPHA] R. V. Oliveira, R. Izhak-Ratzin, B. Zhang, and L. Zhang, "Measurement of Highly Active Prefixes in BGP," in Proceedings of IEEE Globecom, 2005.
- [BGP4] J. W. Stewart, BGP4: Inter-Domain Routing in the Internet. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1998.

8.2 Informative References

Appendix A: Empirical Evaluation

The following aspects of DBGP are tested on the SSFNet-2.0 simulation platform which has implemented BGP4.

- o The ability to counter different types of attacks
- o The ability to rectify the false positives
- o The memory and message overhead

The evaluation proves that DBGP can be used to mitigate all types of attacks. Compared with previous mitigation approaches [PGBGP], it reduces the delay of legitimate announcements significantly, only incurs a small amount of messages and memory overhead.

Author's Addresses

Mingui Zhang
Huawei Technologies Co.,Ltd
HuaWei Building, No.3 Xinxu Rd., Shang-Di
Information Industry Base, Hai-Dian District,
Beijing, 100085 P.R. China

Email: mingui@huawei.com

Bin Liu
Tsinghua University
East Main Building RM9-416
Tsinghua University, Hai-Dian District,
Beijing, 100084 P.R. China

Email: lmyujie@gmail.com

Dacheng Zhang
Huawei Technologies Co.,Ltd
HuaWei Building, No.3 Xinxu Rd., Shang-Di
Information Industry Base, Hai-Dian District,
Beijing, 100085 P.R. China

Email: zhangdacheng@huawei.com

Beichuan Zhang
University of Arizona
Computer Science Department,
The University of Arizona
Tucson, AZ 85721 U.S.A.

Email: bzhang@arizona.edu