Internet Engineering Task Force (IETF)
Request for Comments: 6481
Category: Standards Track
ISSN: 2070-1721

G. Huston
R. Loomans
G. Michaelson
APNIC

February 2012

A Profile for Resource Certificate Repository Structure

Abstract

This document defines a profile for the structure of the Resource Public Key Infrastructure (RPKI) 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 object (file) naming scheme, the contents of repository publication points (directories), and a suggested internal structure of a local repository cache that is intended to facilitate synchronization across a distributed collection of repository publication points and to facilitate certification path construction.

Status of This Memo

This is an Internet Standards Track document.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Further information on Internet Standards is available in Section 2 of RFC 5741.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at http://www.rfc-editor.org/info/rfc6481.

Table of Contents

1. Introduction

To validate attestations made in the context of the Resource Public Key Infrastructure (RPKI) [RFC6480], 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 a 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 synchronization 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 enumerated 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 another form of object that has an analogous public appearance to a file is 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 another form of local storage that has an analogous public appearance to a simple directory of files is 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 [RFC2119].

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 Access (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 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).

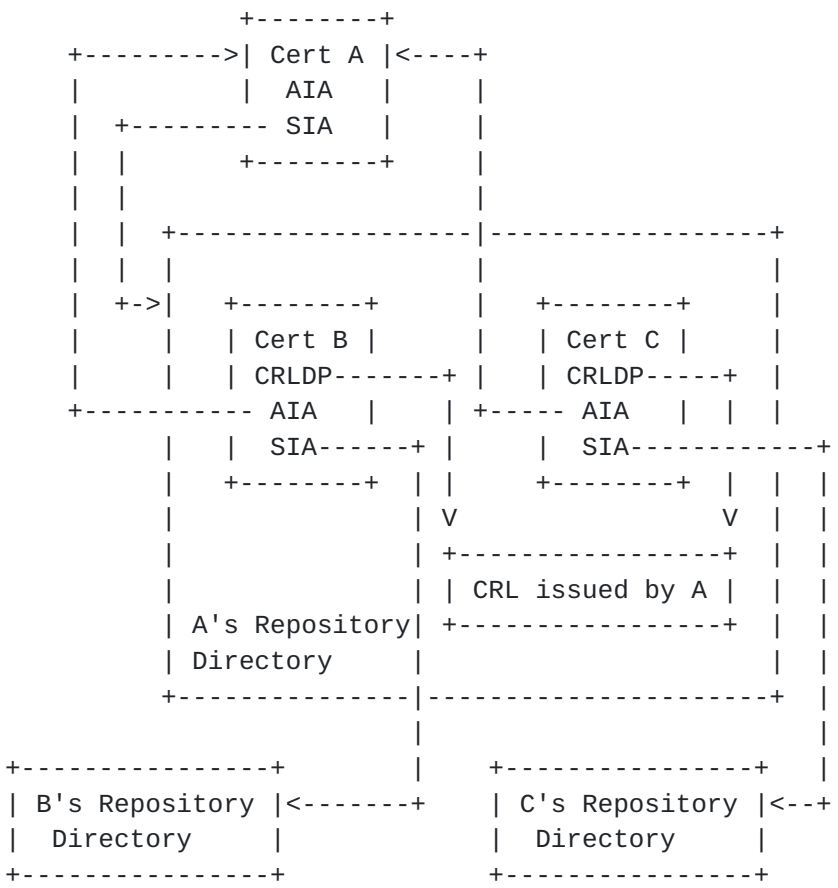

Figure 1. Use of AIA and SIA Extensions in the RPKI

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 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 end-entity (EE) certificate issued by this CA.

2.1. Manifests

Every repository publication point MUST contain a manifest [RFC6486]. 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 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. (It is noted that if no such access regime is in place, then RPs MAY be exposed to intermediate repository states where the manifest and the repository contents may not be precisely aligned. Specific cases and actions in such a situation of misalignment of the manifest and the repository contents are considered in [RFC6486].)

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 [RFC6487]. 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 [RFC6487] 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 [RFC6489]. (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 reissuance of subordinate certificates in a key rollover is limited to the collection of immediate subordinate products of this CA [RFC6489]. In such cases, the repository publication point will contain the CRL, manifest and subordinate certificates of both CA instances. (It is feasible for the entity to use distinct repository publication points for the old and new CA keys, but, in such a case, very careful coordination would be required with subordinate CAs and EEs to ensure that the AIA pointers in the indirect subordinate levels of the RPKI hierarchy are correctly aligned to the subordinate products of the new CA.)

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 MUST 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. Each ".crl" file contains exactly one CRL encoded in DER format.

Manifest:
: When a new instance of a manifest is published, it MUST 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, 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. It is feasible to use different filenames, but this imposes a burden on the validating user. 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 allowing 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. Each ".cer" file contains exactly one certificate encoded in DER format.

Signed Objects:

RPKI signed objects [RFC6488] 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 general non-normative guideline for naming such RPKI 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 RPKI signed objects MUST NOT use the filename extensions ".crl", ".mft", or ".cer".

One form of signed object defined at the time of publication of this document is a Route Origination Authorization (ROA) [RFC6482]. Published ROAs MUST use a filename extension of ".roa" to denote the object as a ROA.

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] [RSYNC]. 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 certificate.

* 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. Detection of content that is not a CA product has the potential to cause confusion to RPs, and in such a case RPs should exercise caution not to invalidate the valid CA products found at the CA's repository publication point.

* 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 [RFC6487].

* Section 2.1 states 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 are assured, and they SHOULD organize 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 the 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 behavior. It is probable that this technique will not remove all possibilities for RPs to see inconsistent states between the manifest and the repository. Because a repository has the potential to be in an partially updated state, it cannot be guaranteed to be internally self consistent all the time.

4. Certificate Reissuance and Repositories

If a CA certificate is reissued, e.g., due to changes in the set of resources contained in the number resource extensions, it should not be necessary to reissue all certificates issued under it. Because these certificates contain 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 reissuance events. That is, reissued 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 reissuance SHOULD intentionally overwrite the previously issued certificate within the repository publication point.

It is noted in 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 [RFC6489] requires that the subordinate products of the old CA be overwritten in the common repository publication point by subordinate products issued by the new CA.

5. Synchronizing Repositories with a Local Cache

It is possible to perform the validation-related task of certificate path construction using the 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 synchronization 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 [RFC6490]. 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. If this is not done, then there might be a SIA pointer loop, or other degenerate forms of the logical RPKI hierarchy, that would cause an RP to malfunction when performing a repository synchronization operation with the RP's local RPKI cache.

RPs SHOULD ensure that this local synchronization uses the retrieved manifests [RFC6486] to ensure that they are synchronizing against a current, consistent state of each repository publication point. It is noted in Section 3 that when the repository publication point contents are updated, a repository operator cannot assure RPs 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. For the RP to mitigate this situation, possible algorithms include performing the synchronization across the repository twice in succession, or performing a manifest retrieval both before and after the synchronization 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 might 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.

Confidentiality is not provided by the repository or by the signed objects published in the repository. Data that is subject to controlled access should not be included in signed objects in the repository unless there is some specified mechanism used to ensure the confidentiality of the data contained in the signed object.

7. IANA Considerations

7.1. Media Types

IANA has registered the following two media types:

```
application/rpki-manifest
application/rpki-roa
```

This document also uses the .cer and .crl file extensions from the application/pkix-cert and application/pkix-crl media registries defined in [RFC2585].

7.1.1. application/rpki-manifest

```
MIME media type name: application
MIME subtype name: rpki-manifest
Required parameters: None
Optional parameters: None
Encoding considerations: binary
Security considerations: Carries an RPKI Manifest [RFC6486]
Interoperability considerations: None
Published specification: This document
Applications that use this media type: Any MIME-complaint transport
Additional information:
   Magic number(s): None
   File extension(s): .mft
   Macintosh File Type Code(s):
Person & email address to contact for further information:
   Geoff Huston <gih@apnic.net>
Intended usage: COMMON
Author/Change controller: Geoff Huston <gih@apnic.net>
```

7.1.2. application/rpki-roa

MIME media type name: application
MIME subtype name: rpki-roa
Required parameters: None
Optional parameters: None
Encoding considerations: binary
Security considerations: Carries an RPKI ROA [RFC6482]
Interoperability considerations: None
Published specification: This document
Applications that use this media type: Any MIME-complaint transport
Additional information:
   Magic number(s): None
   File extension(s): .roa
   Macintosh File Type Code(s):
Person & email address to contact for further information:
   Geoff Huston <gih@apnic.net>
Intended usage: COMMON
Author/Change controller: Geoff Huston <gih@apnic.net>

7.2. RPKI Repository Name Scheme Registry

IANA has created the "RPKI Repository Name Scheme" registry. The registry contains three-letter filename extensions for RPKI repository objects. The registry's contents are managed by IETF Review [RFC5226]. The initial contents of this registry are the following:

Filename extension	RPKI Object	Reference
.cer	Certificate	[RFC6481]
.crl	Certificate Revocation List	[RFC6481]
.mft	Manifest	[RFC6481]
.roa	Route Origination Authorization	[RFC6481]

8. Acknowledgements

This document has benefitted from helpful review comments and input from Stephen Kent, Matt Lepenski, Michael Elkins, Russ Housley, and Sean Turner.

9. References

9.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC6482] Lepinski, M., Kent, S., and D. Kong, "A Profile for Route Origin Authorizations (ROAs)", RFC 6482, February 2012.

[RFC6486] Austein, R., Huston, G., Kent, S., and M. Lepinski, "Manifests for the Resource Public Key Infrastructure (RPKI)", RFC 6486, February 2012.

[RFC6487] Huston, G., Michaelson, G., and R. Loomans, "A Profile for X.509 PKIX Resource Certificates", RFC 6487, February 2012.

[RFC6488] Lepinski, M., Chi, A., and S. Kent, "Signed Object Template for the Resource Public Key Infrastructure (RPKI)", RFC 6488, February 2012.

[RSYNC] rsync web pages, <http://rsync.samba.org/>.

9.2. Informative References

[RFC2585] Housley, R. and P. Hoffman, "Internet X.509 Public Key Infrastructure Operational Protocols: FTP and HTTP", RFC 2585, May 1999.

[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., Ed., "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.

[RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, May 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.

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

[RFC6480] Lepinski, M. and S. Kent, "An Infrastructure to Support Secure Internet Routing", RFC 6480, February 2012.

[RFC6489] Huston, G., Michaelson, G., and S. Kent, "Certification Authority (CA) Key Rollover in the Resource Public Key Infrastructure (RPKI)", BCP 174, RFC 6489, February 2012.

[RFC6490] Huston, G., Weiler, S., Michaelson, G., and S. Kent, "Resource Public Key Infrastructure (RPKI) Trust Anchor Locator", RFC 6490, February 2012.

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