SIDR
Internet-Draft
Intended status: Informational
Expires: July 16, 2017

S. Kent
BBN Technologies
D. Ma
ZDNS

January 12, 2017

Adverse Actions by a Certification Authority (CA) or Repository Manager in the Resource Public Key Infrastructure (RPKI) draft-ietf-sidr-adverse-actions-04

Abstract

This document analyzes actions by or against a CA or independent repository manager in the RPKI that can adversely affect the Internet Number Resources (INRs) associated with that CA or its subordinate CAs. The analysis is done from the perspective of an affected INR holder. The analysis is based on examination of the data items in the RPKI repository, as controlled by a CA (or independent repository manager) and fetched by Relying Parties (RPs). The analysis does not purport to be comprehensive; it does represent an orderly way to analyze a number of ways that errors by or attacks against a CA or repository manager can affect the RPKI and routing decisions based on RPKI data.

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 July 16, 2017.

Table of Contents

1. Introduction

In the context of this document, any change to the Resource Public Key Infrastructure (RPKI) [RFC6480] that diminishes the set of Internet Number Resources (INRs) associated with an INR holder, and that is contrary to the holder's wishes, is termed "adverse". This analysis is done from the perspective of an affected INR holder. An action that results in an adverse charge (as defined above), may be the result of an attack on a CA [RFC7132], an error by a CA, or an error by or an attack on a repository operator. Note that the CA that allocated the affected INRs may be acting in accordance with established policy, and thus the change may be contractually justified, even though viewed as adverse by the INR holder. This document examines the implications of adverse actions within the RPKI with respect to INRs irrespective of the cause of the actions.

Additionally, when a ROA or router certificate is created that "competes" with an existing ROA or router certificate (respectively), the creation of the new ROA or router certificate may be adverse. (A newer ROA competes with an older ROA if the newer ROA points to a different ASN, contains the same or a more specific prefix, and is issued by a different CA. A newer router certificate competes with an older router certificate if the newer one contains the same ASN a different public key, and is issued by a different CA.) Note that transferring resources, or changing of upstream providers may yield competing ROAs and/or router certificates, under some circumstances. Thus not all instances of competition are adverse actions.

As noted above, adverse changes to RPKI data may arise due to several types of causes. A CA may make a mistake in managing the RPKI objects it signs, or it may be subject to an attack. If an attack allows an adversary to use the private key of that CA to sign RPKI objects, then the effect is analogous to the CA making mistakes. There is also the possibility that a CA or repository operator may be subject to legal measures that compel them to make adverse changes to RPKI data. In many cases, such actions may be hard to distinguish from mistakes or attacks, other than with respect to the time required to remedy the adverse action. (Presumably the CA will take remedial action when a mistake or an attack is detected, so the effects are similar in these cases. If a CA has been legally compelled to effect an adverse change, remediation will likely not be swift.)

This document analyzes the various types of actions by a CA (or independent repository operator) that can adversely affect the INRs associated with that CA, as well as the INRs of subordinate CAs. The analysis is based on examination of the data items in the RPKI repository, as controlled by a CA (or independent repository operator) and fetched by Relying Parties (RPs).

2. Analysis of RPKI Repository Objects

This section enumerates the RPKI repository system objects and examines how changes to them affect Route Origination Authorizations (ROAs) and router certificate validation. Identifiers are assigned to errors for reference by later sections of this document. Note that not all adverse actions may be encompassed by this taxonomy.

The RPKI repository [RFC6481] contains a number of (digitally signed) objects that are fetched and processed by RPs. Until the deployment of BGPsec [I-D.ietf-sidr-bgpsec-protocol], the principal goal of the RPKI is to enable an RP to validate ROAs [RFC6482]. A ROA binds address space to an Autonomous System Number (ASN). A ROA can be used to verify BGP announcements with respect to route origin

[RFC6483]. The most important objects in the RPKI for origin validation are ROAs; all of the other RPKI objects exist to enable the validation of ROAs in a fashion consistent with the INR allocation system. Thus errors that result in changes to a ROA, or to RPKI objects needed to validate a ROA, can cause RPs to reach different (from what was intended) conclusions about the validity of the bindings expressed in a ROA.

When BGPsec is deployed, router certificates [I-D.ietf-sidr-bgpsec-pki-profiles] will be added to repository publication points. These are End-Entity (EE) certificates used to verify signatures applied to BGP update data, to enable path validation [I-D.ietf-sidr-bgpsec-protocol]. Router certificates are as important to path validation as ROAs are to origin validation.

The objects contained in the RPKI repository are of two types: conventional PKI objects (certificates and Certificate Revocation Lists (CRLs)) and RPKI-specific signed objects. The latter make use of a common encapsulation format [RFC6488] based on the Cryptographic Message Syntax (CMS) [RFC5652]. A syntax error in this common format will cause an RP to reject the object, e.g., a ROA or Manifest, as invalid.

Adverse actions take several forms:

* Deletion (D) is defined as removing an object from a publication point, without the permission of the INR holder.

* Suppression (S) is defined as not deleting an object, or not publishing an object, as intended by an INR holder. This action also includes retaining a prior version of an object in a publication point when a newer version is available for publication.

* Corruption (C) is defined as modification of a signed object in a fashion not requiring access to the private key used to sign the object. Thus a corrupted object will not carry a valid signature. Implicitly, the corrupted object replaces the legitimate version.

* Modification (M) is defined as publishing a syntactically valid, verifiable version of an object that differs from the (existing) version authorized by the INR holder. Implicitly, the legitimate version of the affected object is deleted and replaced by the modified object.

* Revocation (R) is defined as revoking a certificate (EE or CA) by placing its serial number on the appropriate CRL, without authorization of the INR holder.

* Injection (I) is defined as introducing an instance of a signed object into a publication point (without authorization of the INR holder). It assumes that the signature on the object will be viewed as valid by RPs.

The first three of these actions (deletion, suppression, and corruption) can be effected by any entity that manages the publication point of the affected INR holder. Also, an entity with the ability to act as a man-in-the-middle between an RP and a repository can effect these actions with respect to the RP in question.

The latter three actions (modification, revocation, and injection) nominally require access to the private key of the INR holder.

All six of these actions also can be effected by a parent CA. A parent CA could reissue the INR holder's CA certificate, but with a different public key, matching a private key to which the parent CA has access. The CA could generate new signed objects using the private key associated with the reissued certificate, and publish these objects at a location of its choosing.

Most of these actions may be performed independently or in combination with one another. For example, a ROA may be revoked and deleted or revoked and replaced with a modified ROA. Where appropriate, the analysis of adverse actions will distinguish between individual actions, or combinations thereof, that yield different outcomes for RPs. Recall that the focus of the analysis is the impact on ROAs and router certificates, with respect to RP processing.

The following sections examine how the actions enumerated above affect objects in the RPKI repository system. Each action is addressed in order (Deletion, Suppression, Corruption, Modification, Revocation, and Injection) for each object, making it easy to see how each action has been considered with regard to each object. (For the GhostBusters record we condensed the discussion of the actions because the impact is the same in each case.)

2.1. CA Certificates

Every INR holder is represented by one or more CA certificates. An INR holder has multiple CA certificates if it holds resources acquired from different sources. Also, every INR holder has more

than one CA certificate during key rollover [RFC6489] and algorithm rollover [RFC6916].

If a publication point is not a leaf in the RPKI hierarchy, then the publication point will contain one or more CA certificates, each representing a subordinate CA. Each subordinate CA certificate contains a pointer (SIA) to the publication point where the signed objects associated with that CA can be found [RFC6487].

A CA certificate is a complex data structure and thus errors in that structure may have different implications for RPs depending on the specific data that is in error.

Adverse actions against a CA certificate can cause the following errors:

A-1.1 Deletion

A-1.1.1 Deletion of a CA certificate would cause an RP to not be able to locate signed objects generated by that CA, except those that have been cached by the RP. Thus an RP would be unaware of changed or new (issued after the cached data) INR bindings asserted in subordinate ROAs, and the RP would be unable to validate new or changed router certificates. If the missed objects were intended to replace ROAs or router certificates prior to expiration, then when those objects expire, RPs may cease to view them as valid. As a result, valid routes may be viewed as NotFound or Invalid.

A-1.2 Suppression

A-1.2.1 If publication of a CA certificate is suppressed, the impact depends on what changes appeared in the suppressed certificate. If the SIA value changed, the effect would be the same as in A-1.1 or A-1.4.3. If the [RFC3779] extensions in the suppressed certificate changed, the impact would be the same as in A-1.4.1. If the AIA extension changed in the suppressed certificate, the impact would be the same as in A-1.4.4. Suppression of a renewed/re-issued certificate may cause an old certificate to expire and thus be rejected by RPs.

A-1.3 Corruption

A-1.3.1 Corruption of a CA certificate will cause it to be rejected by RPs. In turn, this may cause subordinate signed objects to become invalid. An RP that has cached the subtree under the affected CA certificate may continue to view it as valid, until objects expire. But changed or new objects might not be retrieved, depending on details of the design of the RP software. Thus this action may be equivalent to suppressing changes to the affected subtree.

A-1.4 Modification

A-1.4.1 If a CA certificate is modified, but still conforms to the RPKI certificate profile [RFC7935], it will be accepted by RPs. If an [RFC3779] extension in this certificate is changed to exclude INRs that were previously present, then subordinate signed objects will become invalid if they rely on the excised INRs. If these objects are CA certificates, their subordinate signed objects will be treated as invalid. If the objects are ROAs, the binding expressed by the affected ROAs will be ignored by RPs. If the objects are router certificates, BGPsec_Path attributes [I-D.ietf-sidr-bgpsec-protocol] verifiable under these certificates will be considered invalid.

A-1.4.2 If the SIA extension of a CA certificate is modified to refer to another publication point, this will cause an RP to look at another location for subordinate objects. This could cause RPs to not acquire the objects that the INR holder intended to be retrieved - manifests, ROAs, router certificates, Ghostbuster records, or any subordinate CA certificates associated with that CA. If the objects at this new location contain invalid signatures or appear to be corrupted, they may be rejected. In this case, cached versions of the objects may be viewed as valid by an RP, until they expire. If the objects at the new location have valid signatures and pass path validation checks, they will replace the cached objects, effectively replacing the INR holder’s objects.

A-1.4.3 If the AIA extension in a CA certificate is modified, it would point to a different CA

certificate, not the parent CA certificate. This extension is used only for path discovery, not path validation. Path discovery in the RPKI is usually performed on a top-down basis, starting with TAs and recursively descending the RPKI hierarchy. Thus there may be no impact on the ability of clients to acquire and validate certificates if the AIA is modified.

A-1.4.4 If the Subject Public Key Info (and Subject Key Identifier extension) in a CA certificate is modified to contain a public key corresponding to a private key held by the parent, the parent could sign objects as children of the affected CA certificate. With this capability, the parent could replace the INR holder, issuing new signed objects that would be accepted by RPs (as long as they do not violate the path validation criteria). This would enable the parent to effect modification, revocation, and injection actions against all of the objects under the affected CA certificate, including subordinate CA certificates. (Note that key rollover also yields a new CA certificate. However, the new certificate will co-exist with the old one for a while, which may help distinguish this legitimate activity from an adverse action.)

A-1.5 Revocation

A-1.5.1 If a CA certificate is revoked an RP will treat as invalid all subordinate signed objects, both immediate and transitively. The effects are essentially the same as described in A-3.4.2.

A-1.6 Injection

A-1.6.1 If a CA certificate is injected the impact will depend on the data contained in the injected certificate. Changes will generally be equivalent to modification actions as described in A-1.4.

2.2. Manifest

Each repository publication point contains a manifest [RFC6486]. The RPKI incorporates manifests to enable RPs to detect suppression and/or substitution of (more recent) publication point objects, as the result of a mistake or attack. A manifest enumerates (by filename)

all of the other signed objects at the publication point. The manifest also contains a hash of each enumerated file, to enable an RP to determine if the named file content matches what the INR holder identified in the manifest.

A manifest is an RPKI signed object, so it is validated as per [RFC6488]. If a manifest is modified in a way that causes any of these checks to fail, the manifest will be considered invalid. Suppression of a manifest itself (indicated by a stale manifest) also can cause an RP to not detect suppression of other signed objects at the publication point. (Note that if a Manifest's EE certificate expires at the time that the Manifest is scheduled to be replaced, a delay in publication will cause the Manifest to become invalid, not merely stale. This very serious outcome should be avoided, e.g., by making the Manifest EE certificate's notAfter value the same as that of the CA certificate under which it was issued). If a signed object at a publication point can be validated (using the rules applicable for that object type), then an RP may accept that object, even if there is no matching entry for it on the manifest. However, it appears that most RP software ignores publication point data that fails to match Manifest entries (at the time this document was written).

Corruption, suppression, modification, or deletion of a manifest might not affect RP processing of other publication point objects, as specified in [RFC6486]. However, as noted above, many RP implementations ignore objects that are present at a publication point but not listed in a valid Manifest. Thus the following actions against a manifest can impact RP processing:

A-2.1 Deletion

A-2.1.1 A Manifest may be deleted from the indicated publication point. In this circumstance an RP may elect to use the previous Manifest (if available), and may ignore any new/changed objects at the publication point. The implications of this action are equivalent to suppression of publication of the objects that are not recognized by RPs because the new objects are not present in the old Manifest. For example, a new ROA could be ignored (A-1.2). A newly issued CA certificate might be ignored (A-1.1). A subordinate CA certificate that was revoked might still be viewed as valid by RPs (A-4.1). A new or changed router

certificate might be ignored (A-6.2) as would a revised Ghostbusters record (A-4.1).

A-2.2 Suppression

A-2.2.1 Publication of a newer Manifest may be suppressed. Suppression of a newer Manifest probably will cause an RP to rely on a cached Manifest (if available). The older Manifest would not enumerate newly added objects, and thus those objects might be ignored by an RP, equivalent to deletion of those objects (A-1.1, A-3.1, A-4.1, A-5.1, A-6.1).

A-2.3 Corruption

A-2.3.1 A Manifest may be corrupted. A corrupted Manifest will be rejected by RPs. This may cause RPs to rely on a previous manifest, with the same impact as A-2.2. If an RP does not revert to using a cached Manifest, the impact of this action is very severe, i.e., all publication point objects probably will be viewed as invalid, including subordinate tree objects. This is equivalent to revoking or deleting an entire subtree (see A-4.4.2).

A-2.4 Modification

A-2.4.1 A Manifest may be modified to remove one or more objects. Because the modified Manifest is viewed as valid by RPs, any objects that were removed may be ignored by RPs. This is equivalent to deleting these objects from the repository. The impact of this action will vary, depending on which objects are (effectively) removed. However, the impact is equivalent to deletion of the object in question, (A-1.1, A-3.1, A-4.1, A-5.1, A-6.1).

A-2.4.2 A Manifest may be modified to add one or more objects. If an added object has a valid signature (and is non-expired), it will be accepted by RPs and processed accordingly. If the added object was previously deleted by the INR holder, this action is equivalent to suppressing deletion of that object. If the object is newly created, or modified, it is equivalent to a modification or injection action for the type of object in

question, and thus is discussed in the relevant section for those actions for the object type.

A-2.4.3 A Manifest may be modified to list an incorrect hash for one or more objects. An object with an incorrect hash may be ignored by an RP. Thus the effect may be equivalent to corrupting the object in question, although the error reported by RP software would differ from that reported for a corrupted object. (The Manifest specifications do not require an RP to ignore an object that has a valid signature and that is not revoked or expired, but for which the hash doesn't match the object. However, an RP may elect to do so.)

A-2.5 Revocation

A-2.5.1 A Manifest may be revoked (by including its EE certificate on the CRL for the publication point). A revoked Manifest will be ignored by an RP, which probably would revert to an older (cached) Manifest. The implications for RPs are equivalent to A-2.1, with regard to new/changed objects.

A-2.6 Injection

A-2.6.1 A Manifest representing different objects may be injected into a publication point. The effects are the same as for a modified Manifest (see above). The impact will depend on the type of the affected object(s), and thus is discussed in the relevant section(s) for each object type.

2.3. Certificate Revocation List

Each publication point contains a CRL that enumerates revoked (not yet expired) certificates issued by the CA associated with the publication point [RFC6481].

Adverse actions against a CRL can cause the following errors:

A-3.1 Deletion

A-3.1.1 If a CRL is deleted, RPs will continue to use an older, previously fetched Certificate Revocation List. As a result, they will not be informed of

any changes in revocation status of subordinate CA or router certificates or the EE certificates of signed objects, e.g., ROAs. This action is equivalent to corruption of a CRL, since a corrupted CRL will not be accepted by an RP.

A-3.1.2 Deletion of a CRL could cause an RP to continue to accept a ROA that no longer expresses the intent of an INR holder. As a result, an announcement for the affected prefixes would be viewed as Valid, instead of NotFound or Invalid. In this case, the effect is analogous to A-5.2.

A-3.1.3 If a router certificate were revoked, and the CRL were deleted, RPs would not be aware of the revocation. They might continue to accept the old, revoked, router certificate. If the certificate had been revoked due to a compromise of the router's private key, RPs would be vulnerable to accepting routes signed by an unauthorized entity.

A-3.1.4 If a subordinate CA certificate were revoked on the deleted CRL, the revocation would not take effect. This could interfere with a transfer of address space from the subordinate CA, adversely affecting routing to the new holder of the space.

A-3.2 Suppression

A-3.2.1 If publication of the most recent CRL is suppressed, an RP will not be informed of the most recent revocation status of subordinate CA or router certificates or the EE certificates of signed objects. If an EE certificate has been revoked and the associated signed object is still present in the publication point, an RP might mistakenly treat that object as valid. (This would happen if the object is still in the manifest or the RP is configured to process valid objects that are not on the manifest.) This type of action is of special concern if the affected object is a ROA, a router certificate, or a subordinate CA certificate. The effects here are equivalent to CRL deletion (A-3.1), but suppression of a new CRL may not even be reported as an error, i.e., if the suppressed CRL were

issued before the NextUpdate time (of the previous CRL).

A-3.3 Corruption

A-3.3.1 If a CRL is corrupted, an RP will reject it. If a prior CRL has not yet exceeded its NextUpdate time, an RP will continue to use the prior CRL. Even if the prior CRL has passed the NextUpdate time, an RP may choose to continue to rely on the prior CRL. The effects are essentially equivalent to suppression or deletion of a CRL (A-3.1, A-3.2).

A-3.4 Modification

A-3.4.1 If a CRL is modified to erroneously list a signed object's EE certificate as revoked, the corresponding object will be treated as invalid by RPs, even if it is present in a publication point. If this object is a ROA, the (legitimate) binding expressed by the ROA will be ignored by an RP (see A-5.5). If a CRL is modified to erroneously list a router certificate as revoked, a path signature associated with that certificate will be treated as Not Valid by RPs (see A-6.5).

A-3.4.2 If a CRL is modified to erroneously list a CA certificate as revoked, that CA and all subordinate signed objects will be treated as invalid by RPs. Depending on the location of the affected CA in the hierarchy, these effects could be very substantial, causing routes that should be Valid to be treated as NotFound.

A-3.4.3 If a CRL is modified to omit a revoked EE, router, or CA certificate, RPs likely will continue to accept the revoked, signed object as valid. This contravenes the intent of the INR holder. If an RP continues to accept a revoked ROA, it may make routing decisions on now-invalid data. This could cause valid routes to be de-preferenced and invalid routes to continue to be accepted.

A-3.5 Revocation

A-3.5.1 A CRL cannot be revoked, per se, but it will fail validation if the CA certificate under which it

was issued is revoked. See A-1.5 for a discussion of that action.

A-3.6 Injection

A-3.6.1 Insertion of a bogus CRL can have the same effects as listed above for a modified CRL, depending on how the inserted CRL differs from the correct CRL.

2.4. ROA

In addition to the generic RPKI object syntax checks, ROA validation requires that the signature on the ROA can be validated using the public key from the EE certificate embedded in the ROA [RFC6482]. It also requires that the EE certificate be validated consistently with the procedures described in [RFC6482] and [RFC6487]. Adverse actions against a ROA can cause the following errors:

A-4.1 Deletion

A-4.1.1 A ROA may be deleted from the indicated publication point. The result is to void the binding between the prefix(es) and the AS number in the ROA. An RP that previously viewed this binding as authentic will now not have any evidence about its validity. For origin validation, this means that a legitimate route will be treated as NotFound (if there are no other ROAs for the same prefix) or Invalid (if there is another ROA for the same prefix, but with a different AS number).

A-4.2 Suppression

A-4.2.1 Publication of a newer ROA may be suppressed. If the INR holder intended to change the binding between the prefix(es) and the AS number in the ROA, this change will not be effected. As a result, RPs may continue to believe an old prefix/ASN binding that is no longer what the INR holder intended.

A-4.2.2 If an INR holder intends to issue and publish two (or more) new ROAs for the same address space, one (or more) of the new ROAs may be suppressed while the other is published. In this case, RPs will

de-preference the suppressed prefix/ASN binding. Suppression of the new ROA might cause traffic to flow to an ASN other than the one(s) intended by the INR holder.

A-4.2.3 If an INR holder intends to delete all ROAs for the same address space, some of them may be retained while the others are deleted. Preventing the deletion of some ROAs can cause traffic to continue to be delivered to the ASNs that were advertised by these ROAs. Deletion of all ROAs is consistent with a transfer of address space to a different INR holder, in a phased fashion. Thus this sort of attack could interfere with the successful transfer of the affected address space (until such time as the prefixes are removed from the previous INR holder's CA certificate).

A-4.3 Corruption

A-4.3.1 A ROA may be corrupted. A corrupted ROA will be ignored by an RP, so the effect is essentially the same as for A-4.1 and A-4.5. A possible difference is that an RP may be alerted to the fact that the ROA was corrupted, which might attract attention to the attack.

A-4.4 Modification

A-4.4.1 A ROA may be modified so that the Autonomous System Number (ASN) or one or more of the address blocks in a ROA is different from the values the INR holder intended for this ROA. (This action assumes that the modified ROA's ASN and address ranges are authorized for use by the INR holder.) This attack will cause RPs to de-preference the legitimate prefix/ASN binding intended by the INR holder.

A-4.5 Revocation

A-4.5.1 A ROA may be revoked (by placing its EE certificate on the CRL for the publication point). This has the same effect as A-4.1.

A-4.6 Injection

A-4.6.1 A ROA expressing different bindings than those published by the INR holder may be injected into a publication point. This action could authorize an additional ASN to advertise the specified prefix, allowing that ASN to originate routes for the prefix, thus enabling route origin spoofing. In this case, the injected ROA is considered to be in competition with any existing authorized ROAs for the specified prefix.

A-4.6.2 An injected ROA might express a different prefix for an ASN already authorized to originate a route, e.g., a longer prefix, which could enable that ASN to override other advertisements using shorter prefixes. If there are other ROAs that authorize different ASNs to advertise routes to the injected ROA's prefix, then the injected ROA is in competition with these ROAs.

2.5. Ghostbusters Record

The Ghostbusters record [RFC6493] is a signed object that may be included at a publication point, at the discretion of the INR holder or publication point operator. The record is validated according to [RFC6488]. Additionally, the syntax of the record is verified based on the vCard profile from Section 5 of [RFC6493]. Errors in this record do not affect RP processing. However, if an RP encounters a problem with objects at a publication point, the RP may use information from the record to contact the publication point operator.

Adverse actions against a Ghostbusters record can cause the following error:

A-5.1 Deletion, suppression, corruption, or revocation of a Ghostbusters record could prevent an RP from contacting the appropriate entity when a problem is detected by the RP. Modification or injection of a Ghostbusters record could cause an RP to contact the wrong entity, thus delaying remediation of a detected anomaly. All of these actions are viewed as equivalent from an RP processing perspective; they do not alter RP validation of ROAs or router certificates. However, these actions can interfere with remediation of a problem when detected by an RP.

2.6. Router Certificates

Router certificates are used by RPs to verify signatures on BGPsec_Path attributes carried in Update messages.

Each AS is free to determine the granularity at which router certificates are managed [I-D.ietf-sidr-bgpsec-pki-profiles]. Each participating AS is represented by one or more router certificates. During key or algorithm rollover, multiple router certificates will be present in a publication point, even if the AS is normally represented by just one such certificate.

Adverse actions against router certificates can cause the following errors:

A-6.1 Deletion

A-6.1.1

Deletion of a router certificate would cause an RP to not be able to verify signatures applied to BGPsec_Path attributes on behalf of the AS in question. In turn, this would cause the route to be treated with lower preference than competing routes that have valid BGPsec_Path attribute signatures. (However, if another router certificate for the affected AS is valid and contains the same AS number and public key, and is in use by that AS, there would be no effect on routing. This scenario will arise if a router certificate is renewed, i.e., issued with a new validity interval.)

A-6.2 Suppression

A-6.2.1

Suppression of a router certificate could have the same impact as deletion of a certificate of this type, i.e., if no router certificate was available, BGPsec attributes that should be verified using the certificate would fail validation. If an older certificate existed, and had not expired, it would be used by RPs. If the older certificate contained a different ASN, the impact would be the same as in A-6.4.

A-6.3 Corruption

A-6.3.1 Corruption of a router certificate will result in the certificate being rejected by RPs. Absent a valid router certificate, BGPsec_Path attributes associated with that certificate will be unverifiable. In turn, this would cause the route to be treated with lower preference than competing routes that have valid BGPsec_Path attribute signatures.

A-6.4 Modification

A-6.4.1 If a router certificate is modified to represent a different ASN, but it still passes syntax checks, then this action could cause signatures on BGPsec_Path attributes to be associated with the wrong AS. This could cause signed routes to be inconsistent with the intent of the INR holder, e.g., traffic might be routed via a different AS than intended.

A-6.5 Revocation

A-6.5.1 If a router certificate were revoked, BGPsec_Path attributes verifiable using that certificate would not longer be considered valid. The impact would be the same as for a deleted certificate, as described in A-6.1.

A-6.6 Injection

A-6.6.1 Insertion of a router certificate could authorize additional routers to sign BGPsec traffic for the targeted ASN, and thus undermine fundamental BGPsec security guarantees. If there are existing, authorized router certificates for the same ASN, then the injected router certificate is in competition with these existing certificates.

3. Analysis of Actions Relative to Scenarios

This section examines the types of problems that can arise in four scenarios described below. We consider mistakes, (successful) attacks against a CA or a publication point, and situations in which a CA or publication point manager is compelled to take action by a law enforcement authority.

We explore the following four scenarios:

A. An INR holder operates its own CA and manages its own repository publication point.

B. An INR holder operates its own CA, but outsources management of its repository publication point to its parent or another entity.

C. An INR holder outsources management of its CA to its parent, but manages its own repository publication point.

D. An INR holder outsources management of its CA and its publication point to its parent.

Note that these scenarios focus on the affected INR holder as the party directly affected by an adverse action. The most serious cases arise when the INR holder appears as a high-tier CA in the RPKI hierarchy; in such situations subordinate INR holders may be affected as a result of an action. A mistake by or an attack against a "leaf" has more limited impact because all of the affected INRs belong to the INR holder itself.

In Scenario A, actions by the INR holder can adversely affect all of its resources and, transitively, resources of any subordinate CAs. (If the CA is a "leaf" in the RPKI, then it has no subordinate CAs and the damage is limited to its own INRs.)

In Scenario B, actions by the (outsourced) repository operator also can adversely affect the resources of the INR holder, and those of any subordinates CAs. (If the CA is a "leaf" in the RPKI, then it has no subordinate CAs and the damage is limited, as in Scenario A.) The range of adverse effects here includes those in Scenario A, and adds a new potential source of adverse actions, i.e., the outsourced repository operator.

In Scenario C, all signed objects associated with the INR holder are generated by the parent CA but are self-hosted. (We expect this scenario to be rare, because an INR holder that elects to outsource CA operation seems unlikely to manage its own repository publication point.) Because that CA has the private key used to sign them, it can generate alternative signed objects---ones not authorized by the INR holder. However, erroneous objects created by the parent CA will not be published by the INR holder IF the holder checks them first. Because the parent CA is acting on behalf of the INR holder, mistakes by or attacks against that entity are equivalent to ones effected by the INR holder in Scenario A.

The INR holder is most vulnerable in Scenario D. Actions by the parent CA, acting on behalf of the INR holder, can adversely affect

all signed objects associated with that INR holder, including any subordinate CA certificates. These actions will presumably translate directly into publication point changes, because the parent CA is managing the publication point for the INR holder. The range of adverse effects here includes those in Scenarios A, B, and C.

3.1. Scenario A

In this scenario, the INR holder acts as its own CA and it manages its own publication point. Actions by the INR holder can adversely affect all of its resources and, transitively, resources of any subordinate CAs. (If the CA is a "leaf" in the RPKI, then it has no subordinate CAs and the damage is limited to its own INRs.) Mistakes by the INR holder can cause any of the actions noted in Section 2. A successful attack against this CA can effect all of the modification, revocation, or injection actions noted in that section. (We assume that objects generated by the CA are automatically published). An attack against the publication point can effect all of the deletion, suppression, or corruption actions noted in that section.

3.2. Scenario B

In this scenario, the INR holder acts as its own CA and but it delegates management of it own publication point to a third party. Mistakes by the INR holder can cause any of the modification, revocation, or injection actions described in Section 2. Actions by the repository operator can adversely affect the resources of the INR holder, and those of any subordinate CAs. (If the CA is a "leaf" in the RPKI, then it has no subordinate CAs and the damage is limited, as in Scenario A.) The range of adverse effects here includes those in Scenario A, and adds a new potential source of adverse actions, i.e., the third party repository operator. A successful attack against the CA can effect all of the modification, revocation, or injection actions noted in that section (assuming that objects generated by the CA are automatically published). Here, actions by the publication point manager (or attacks against that entity) can effect all of the deletion, suppression, or corruption actions noted in Section 2.

3.3. Scenario C

In this scenario, the INR holder outsources management of its CA to its parent, but manages its own repository publication point. All signed objects associated with the INR holder are generated by the parent CA but are self-hosted. (We expect this scenario to be rare, because an INR holder that elects to outsource CA operation seems unlikely to manage its own repository publication point.) Because that CA has the private key used to sign them, it can generate

alternative signed objects -- ones not authorized by the INR holder. However, erroneous objects created by the parent CA will not be published by the INR holder IF the holder checks them first. Because the parent CA is acting on behalf of the INR holder, mistakes by or attacks against that entity are equivalent to ones effected by the INR holder in Scenario A. Mistakes by the INR holder, acted upon by the parent CA, can cause any of the actions noted in Section 2. Actions unilaterally undertaken by the parent CA also can have the same effect, unless the INR holder checks the signed objects before publishing them. A successful attack against the parent CA can effect all of the modification, revocation, or injection actions noted in Section 2, unless the INR holder checks the signed objects before publishing them. An attack against the INR holder (in its role as repository operator) can effect all of the deletion, suppression, or corruption actions noted in Section 2 (because the INR holder is managing its publication point), unless the INR holder checks the signed objects before publishing them. (An attack against the INR holder implies that the path it uses to direct the parent CA to issue and publish objects has been compromised.)

3.4. Scenario D

In this scenario an INR holder outsources management of both its CA and its publication point to its parent. The INR holder is most vulnerable in this scenario. Actions by the parent CA, acting on behalf of the INR holder, can adversely affect all signed objects associated with that INR holder, including any subordinate CA certificates. These actions will presumably translate directly into publication point changes, because the parent CA is managing the publication point for the INR holder. The range of adverse effects here includes those in Scenarios A, B, and C. Mistakes by the INR holder, acted upon by the parent CA, can cause any of the actions noted in Section 2. Actions unilaterally undertaken by the parent CA also can have the same effect. A successful attack against the parent CA can effect all of the modification, revocation, or injection actions noted in Section 2. An attack against the parent CA can also effect all of the deletion, suppression, or corruption actions noted in Section 2 (because the parent CA is managing the INR holder’s publication point).

4. Security Considerations

This informational document describes a threat model for the RPKI, focusing on mistakes by or attacks against CAs and independent repository managers. It is intended to provide a basis for the design of future RPKI security mechanisms that seek to address the concerns associated with such actions.

The analysis in this document identifies a number of circumstances in which attacks or errors can have significant impacts on routing. One ought not interpret this as a condemnation of the RPKI. It is only an attempt to document the implications of a wide range of attacks and errors, in the context of the RPKI. The primary alternative mechanism for disseminating routing information is Internet Routing Registry (IRR) technology ([RFC2650], [RFC2725]), which uses the Routing Policy Specification Language (RPSL) [RFC2622]. IRR technology exhibits its own set of security problems, which are discussed in [RFC7682].

5. IANA Considerations

This document has no actions for IANA.

6. Acknowledgements

The authors thank Richard Hansen and David Mandelberg for their extensive review, feedback and editorial assistance. Thanks also go to Daiming Li for her editorial assistance.

7. References

7.1. Normative References

[I-D.ietf-sidr-bgpsec-pki-profiles]
Reynolds, M., Turner, S., and S. Kent, "A Profile for BGPsec Router Certificates, Certificate Revocation Lists, and Certification Requests", draft-ietf-sidr-bgpsec-pki-profiles-18 (work in progress), July 2016.

[I-D.ietf-sidr-bgpsec-protocol]
Lepinski, M. and K. Sriram, "BGPsec Protocol Specification", draft-ietf-sidr-bgpsec-protocol-21 (work in progress), December 2016.

[RFC3779] Lynn, C., Kent, S., and K. Seo, "X.509 Extensions for IP Addresses and AS Identifiers", RFC 3779, DOI 10.17487/RFC3779, June 2004, <http://www.rfc-editor.org/info/rfc3779>.

[RFC6480] Lepinski, M. and S. Kent, "An Infrastructure to Support Secure Internet Routing", RFC 6480, DOI 10.17487/RFC6480, February 2012, <http://www.rfc-editor.org/info/rfc6480>.

[RFC6481] Huston, G., Loomans, R., and G. Michaelson, "A Profile for Resource Certificate Repository Structure", RFC 6481, DOI 10.17487/RFC6481, February 2012, <http://www.rfc-editor.org/info/rfc6481>.

[RFC6482] Lepinski, M., Kent, S., and D. Kong, "A Profile for Route Origin Authorizations (ROAs)", RFC 6482, DOI 10.17487/RFC6482, February 2012, <http://www.rfc-editor.org/info/rfc6482>.

[RFC6483] Huston, G. and G. Michaelson, "Validation of Route Origination Using the Resource Certificate Public Key Infrastructure (PKI) and Route Origin Authorizations (ROAs)", RFC 6483, DOI 10.17487/RFC6483, February 2012, <http://www.rfc-editor.org/info/rfc6483>.

[RFC6486] Austein, R., Huston, G., Kent, S., and M. Lepinski, "Manifests for the Resource Public Key Infrastructure (RPKI)", RFC 6486, DOI 10.17487/RFC6486, February 2012, <http://www.rfc-editor.org/info/rfc6486>.

[RFC6487] Huston, G., Michaelson, G., and R. Loomans, "A Profile for X.509 PKIX Resource Certificates", RFC 6487, DOI 10.17487/RFC6487, February 2012, <http://www.rfc-editor.org/info/rfc6487>.

[RFC6488] Lepinski, M., Chi, A., and S. Kent, "Signed Object Template for the Resource Public Key Infrastructure (RPKI)", RFC 6488, DOI 10.17487/RFC6488, February 2012, <http://www.rfc-editor.org/info/rfc6488>.

[RFC6489] Huston, G., Michaelson, G., and S. Kent, "Certification Authority (CA) Key Rollover in the Resource Public Key Infrastructure (RPKI)", BCP 174, RFC 6489, DOI 10.17487/RFC6489, February 2012, <http://www.rfc-editor.org/info/rfc6489>.

[RFC6493] Bush, R., "The Resource Public Key Infrastructure (RPKI) Ghostbusters Record", RFC 6493, DOI 10.17487/RFC6493, February 2012, <http://www.rfc-editor.org/info/rfc6493>.

[RFC6916] Gagliano, R., Kent, S., and S. Turner, "Algorithm Agility Procedure for the Resource Public Key Infrastructure (RPKI)", BCP 182, RFC 6916, DOI 10.17487/RFC6916, April 2013, <http://www.rfc-editor.org/info/rfc6916>.

[RFC7935] Huston, G. and G. Michaelson, Ed., "The Profile for Algorithms and Key Sizes for Use in the Resource Public Key Infrastructure", RFC 7935, DOI 10.17487/RFC7935, August 2016, <http://www.rfc-editor.org/info/rfc7935>.

7.2. Informative References

[RFC2622] Alaettinoglu, C., Villamizar, C., Gerich, E., Kessens, D., Meyer, D., Bates, T., Karrenberg, D., and M. Terpstra, "Routing Policy Specification Language (RPSL)", RFC 2622, DOI 10.17487/RFC2622, June 1999, <http://www.rfc-editor.org/info/rfc2622>.

[RFC2650] Meyer, D., Schmitz, J., Orange, C., Prior, M., and C. Alaettinoglu, "Using RPSL in Practice", RFC 2650, DOI 10.17487/RFC2650, August 1999, <http://www.rfc-editor.org/info/rfc2650>.

[RFC2725] Villamizar, C., Alaettinoglu, C., Meyer, D., and S. Murphy, "Routing Policy System Security", RFC 2725, DOI 10.17487/RFC2725, December 1999, <http://www.rfc-editor.org/info/rfc2725>.

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

[RFC7132] Kent, S. and A. Chi, "Threat Model for BGP Path Security", RFC 7132, DOI 10.17487/RFC7132, February 2014, <http://www.rfc-editor.org/info/rfc7132>.

[RFC7682] McPherson, D., Amante, S., Osterweil, E., Blunk, L., and D. Mitchell, "Considerations for Internet Routing Registries (IRRs) and Routing Policy Configuration", RFC 7682, DOI 10.17487/RFC7682, December 2015, <http://www.rfc-editor.org/info/rfc7682>.

Authors' Addresses

Stephen Kent
BBN Technologies
10 Moulton St
Cambridge, MA 02138-1119
USA

Email: kent@alum.mit.edu

Di Ma
ZDNS
4 South 4th St. Zhongguancun
Haidian, Beijing 100190
China

Email: madi@zdns.cn

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: January 8, 2017

T. Bruijnzeels
O. Muravskiy
RIPE NCC
B. Weber
Cobenian
R. Austein
Dragon Research Labs

July 7, 2016

RPKI Repository Delta Protocol
draft-ietf-sidr-delta-protocol-03

Abstract

In the Resource Public Key Infrastructure (RPKI), certificate authorities publish certificates, including end entity certificates, Certificate Revocation Lists (CRL), and RPKI signed objects to repositories. Relying Parties (RP) retrieve the published information from those repositories. This document specifies a delta protocol which provides relying parties with a mechanism to query a repository for incremental updates, thus enabling the RP to keep its state in sync with the repository.

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 January 8, 2017.

Copyright Notice

Table of Contents

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

In the Resource Public Key Infrastructure (RPKI), Certificate Authorities (CAs) publish certificates [RFC6487], RPKI signed objects [RFC6488], manifests [RFC6486], and CRLs to repositories. CAs may have an embedded mechanism to publish to these repositories, or they may use a separate repository server and publication protocol. RPKI

repositories are currently accessible using the rsync protocol, allowing Relying Parties (RPs) to synchronise a local copy of the RPKI repository used for validation with the remote repositories [RFC6481].

This document specifies an alternative repository access protocol based on notification, snapshot and delta files that a RP can retrieve over the HTTPS protocol. This allows RPs to perform either a full (re-)synchronisation of their local copy of the repository using snapshot files, or use delta files to keep their local repository updated after initial synchronisation.

This protocol is designed to be consistent (in terms of data structures) with the publication protocol [I-D.ietf-sidr-publication] and treats publication events of one or more repository objects as discrete events that can be communicated to relying parties. This approach helps to minimize the amount of data that traverses the network and thus helps minimize the amount of time until repository convergence occurs. This protocol also provides a standards based way to obtain consistent, point in time views of a single repository, eliminating a number of consistency related issues. Finally, this approach allows these discrete events to be communicated as immutable files, so that caching infrastructure can be used to reduce the load on a repository server when a large number of relying parties are querying it.

3. RPKI Repository Delta Protocol Implementation

3.1. Informal Overview

Certification Authorities (CA) in the RPKI use a repository server to publish their RPKI products, such as manifests, CRLs, signed certificates and RPKI signed objects. This repository server may be remote, or embedded in the CA engine itself. Certificates in the RPKI that use a repository server that supports this delta protocol include a special Subject Information Access (SIA) pointer referring to a notification file.

The notification file includes a globally unique session_id in the form of a version 4 UUID, and serial number that can be used by the Relying Party (RP) to determine if it and the repository are synchronised. Furthermore it includes a link to the most recent complete snapshot of current objects that are published by the repository server, and a list of links to delta files, for each revision starting at a point determined by the repository server, up to the current revision of the repository.

A RP that learns about a notification file location for the first time can download it, and then proceed to download the latest snapshot file, and thus create a local copy of the repository that is in sync with the repository server. The RP should remember the location of this notification file, the session_id and current serial number.

RPs are encouraged to re-fetch this notification file at regular intervals, but not more often than once per minute. After re-fetching the notification file, the RP may find that there are one or more delta files available that allow it to synchronise its local repository with the current state of the repository server. If no contiguous chain of deltas from RP's serial to the latest repository serial is available, or if the session_id has changed, the RP should perform a full resynchronisation instead.

As soon as the RP fetches new content in this way it should start a validation process. An example of a reason why a RP may not do this immediately is because it has learned of more than one notification location and it prefers to complete all its updates before validating.

The repository server may use caching infrastructure to reduce its load. It should be noted that snapshots and deltas for any given session_id and serial number contain an immutable record of the state of the repository server at a certain point in time. For this reason these files can be cached indefinitely. Notification files are polled by RPs to discover if updates exist, and for this reason notification files may not be cached for longer than one minute.

3.2. Certificate Authority Use

Certificate Authorities that use this delta protocol MUST include an instance of an SIA AccessDescription extension in resource certificates they produce, in addition to the ones defined in [RFC6487],

```
AccessDescription ::= SEQUENCE {
  accessMethod OBJECT IDENTIFIER,
  accessLocation GeneralName }
```

This extension MUST use an accessMethod of id-ad-rpkiNotify, see: [IANA-AD-NUMBERS],

```
id-ad OBJECT IDENTIFIER ::= { id-pkix 48 }
id-ad-rpkiNotify OBJECT IDENTIFIER ::= { id-ad 13 }
```

The accessLocation MUST be an HTTPS URI as defined in [RFC2818], that will point to the update notification file for the repository server that publishes the products of this CA certificate.

Relying Parties that do not support this delta protocol MUST NOT reject a CA certificate merely because it has an SIA extension containing this new kind of AccessDescription.

3.3. Repository Server Use

3.3.1. Initialisation

When the repository server initialises it must perform the following actions:

The server MUST generate a new random version 4 UUID to be used as the session_id

The server MUST then generate a snapshot file for serial number ONE for this new session that includes all currently known published objects that the repository server is responsible for. Note that this snapshot file MAY contain zero publish elements at this point if no objects have been submitted for publication yet.

This snapshot file MUST be made available at a URL that is unique to this session_id and serial number, so that it can be cached indefinitely.

The format and caching concerns for snapshot files are explained in more detail in Section 3.5.2.

After the snapshot file has been published the repository server MUST publish a new notification file that contains the new session_id, has serial number ONE, has one reference to the snapshot file that was just published, and that contains no delta references.

The format and caching concerns for update notification files are explained in more detail in Section 3.5.1.

3.3.2. Publishing Updates

Whenever the repository server receives updates from a CA it SHOULD generate new snapshot and delta files. However, if a publication server services a large number of CAs it MAY choose to combine updates from multiple CAs. If a publication server combines updates in this way, it MUST NOT postpone publishing for longer than one minute.

Updates must be processed as follows:

- The new repository serial number MUST be one greater than the current repository serial number.
- A new delta file MUST be generated for this new serial. This delta file MUST include all new, replaced and withdrawn objects for multiple CAs if applicable, as a single change set.
- This delta file MUST be made available at a URL that is unique to the current session_id and serial number, so that it can be cached indefinitely.
- The format and caching concerns for delta files are explained in more detail in Section 3.5.3.
- The repository server MUST also generate a new snapshot file for this new serial. This file MUST contain all "publish" elements for all current objects.
- The snapshot file MUST be made available at a URL that is unique to this session and new serial, so that it can be cached indefinitely.
- The format and caching concerns for snapshot files are explained in more detail in Section 3.5.2.
- The update notification file SHOULD be kept small, and in order to do so the repository server needs to make a decision about which delta files to support. Any older delta files that, when combined with all more recent delta files, will result in total size of deltas exceeding the size of the snapshot, MUST be excluded.
- The server MAY also exclude more recent delta files if it finds that their usage by a small number of RPs that would be forced to perform a full synchronisation is outweighed by the performance penalty for all RPs in having a large update notification file. However the repository server SHOULD include all deltas for the last two hours.
- A new notification file MUST now be created by the repository server. This new notification file MUST include a reference to the new snapshot file, and all delta files selected in the previous steps.
- The format and caching concerns for update notification files are explained in more detail in Section 3.5.1.

If the repository server is not capable of performing the above for some reason, then it MUST perform a full re-initialisation, as explained above in Section 3.3.1.

3.4. Relying Party Use

3.4.1. Processing the Update Notification File

When a Relying Party (RP) performs RPKI validation and learns about a valid certificate with an SIA entry for the RRDP protocol, it SHOULD prefer to use this protocol as follows.

The RP SHOULD download the update notification file, unless an update notification file was already downloaded and processed from the same location in this validation run.

The RP MAY use a "User-Agent" header explained in section 5.5.3. of [RFC7231] to identify the name and version of the RP software used. This is not required, but would be useful to help track capabilities of Relying Parties in the event of changes to the RPKI standards.

When the RP downloads an update notification file it MUST verify the file format and validation steps described in section Section 3.5.1.3. If this verification fails, the file MUST be rejected.

The RP MUST verify whether the session_id in this update notification file matches the last known session_id for this update notification file location. If the session_id matches the last known session_id, then an RP MAY download and process missing delta files as described in section Section 3.4.3, provided that all delta files for serial numbers between the last processed serial number and the current serial number in the notification file can be processed this way.

If the session_id was not previously known, or if delta files could not be used, then the RP MUST update its last known session_id to this session_id and download and process snapshot file on the update notification file as described in section Section 3.4.2.

If neither update notification file and one snapshot file or delta files could be processed this way, the RP MUST issue an operator error, and SHOULD use an alternate repository retrieval mechanism if it is available.

3.4.2. Processing a Snapshot File

When the RP downloads a snapshot file it MUST verify the file format and validation steps described in Section 3.5.2.3. If this verification fails, the file MUST be rejected.

Furthermore the RP MUST verify that the hash of the contents of this file matches the hash on the update notification file that referenced it. In case of a mismatch of this hash, the file MUST be rejected.

If an RP retrieved a snapshot file that is valid according to the above criteria, it should perform the following actions:

The RP MUST verify that the session_id matches the session_id of the notification file. If the session_id values do not match the file MUST be rejected.

The RP MUST verify that the serial number of this snapshot file is greater than the last processed serial number for this session_id. If this fails the file MUST be rejected.

The RP SHOULD then add all publish elements to a local storage and update its last processed serial number to the serial number of this snapshot file.

3.4.3. Processing Delta Files

If an update notification file contains a contiguous chain of links to delta files from the last processed serial number to the current serial number, then RPs MUST attempt to download and process all delta files in order of serial number as follows.

When the RP downloads a delta file it MUST verify the file format and perform validation steps described in Section 3.5.3.3. If this verification fails, the file MUST be rejected.

Furthermore the RP MUST verify that the hash of the contents of this file matches the hash on the update notification file that referenced it. In case of a mismatch of this hash, the file MUST be rejected.

If an RP retrieved a delta file that is valid according to the above criteria, it should perform the following actions:

The RP MUST verify that the session_id matches the session_id of the notification file. If the session_id values do not match the file MUST be rejected.

The RP MUST verify that the serial number of this delta file is exactly one greater than the last processed serial number for this session_id, and if not this file MUST be rejected.

The RP SHOULD add all publish elements to a local storage and update its last processed serial number to the serial number of this snapshot file.

The RP SHOULD NOT remove objects from its local storage solely because it encounters a "withdraw" element, because this would enable a publication server to withdraw any object without the signing Certificate Authority consent. Instead it is RECOMMENDED that a RP uses additional strategies to determine if an object is still relevant for validation before removing it from its local storage.

3.4.4. Polling the Update Notification File

Once a Relying Party has learned about the location, session_id and last processed serial number of repository that uses the RRDP protocol, the RP MAY start polling the repository server for updates. However the RP MUST NOT poll for updates more often than once every 1 minute, and in order to reduce data usage RPs MUST use the "If-Modified-Since" header explained in section 3.3 of [RFC7232]in requests.

If an RP finds that updates are available it SHOULD download and process the file as described in Section 3.4.1, and initiate a new validation process. A detailed description of the validation process itself is out of scope of this document.

3.5. File Definitions

3.5.1. Update Notification File

3.5.1.1. Purpose

The update notification file is used by RPs to discover whether any changes exist between the state of the repository and the RP’s cache. It describes the location of the files containing the snapshot and incremental deltas which can be used by the RP to synchronise with the repository.

3.5.1.2. Cache Concerns

A repository server MAY use caching infrastructure to cache the notification file and reduce the load of HTTPS requests. However, since this file is used by RPs to determine whether any updates are

available the repository server MUST ensure that this file is not cached for longer than 1 minute. An exception to this rule is that it is better to serve a stale notification file, then no notification file.

How this is achieved exactly depends on the caching infrastructure used. In general a repository server may find certain HTTP headers to be useful, such as: Cache-Control: max-age=60. Another approach can be to have the repository server push out new versions of the notification file to the caching infrastructure when appropriate.

Relying Parties SHOULD NOT cache the notification file for longer than 1 minute, regardless of the headers set by the repository server or CDN.

3.5.1.3. File Format and Validation

Example notification file:

```
<notification xmlns="http://www.ripe.net/rpki/rrdp"
      version="1"
      session_id="9df4b597-af9e-4dca-bdda-719cce2c4e28"
      serial="3">
  <snapshot uri="https://host/9d-8/3/snapshot.xml" hash="AB"/>
  <delta serial="3" uri="https://host/9d-8/3/delta.xml" hash="CD"/>
  <delta serial="2" uri="https://host/9d-8/2/delta.xml" hash="EF"/>
</notification>
```

Note: URIs and hash values in this example are shortened because of formatting.

The following validation rules must be observed when creating or parsing notification files:

- A RP MUST reject any update notification file that is not well-formed, or which does not conform to the RELAX NG schema outlined in Section 3.5.4 of this document.
- The XML namespace MUST be http://www.ripe.net/rpki/rrdp
- The encoding MUST be US-ASCII
- The version attribute in the notification root element MUST be 1
- The session_id attribute MUST be a random version 4 UUID unique to this session

- The serial attribute must be an unbounded, unsigned positive integer in decimal format indicating the current version of the repository.
- The notification file MUST contain exactly one ’snapshot’ element for the current repository version.
- If delta elements are included they MUST form a contiguous sequence of serial numbers starting at a revision determined by the repository server, up to the serial number mentioned in the notification element.
- The hash attribute in snapshot and delta elements must be the hexadecimal encoding of the SHA-256 hash of the referenced file. The RP MUST verify this hash when the file is retrieved and reject the file if the hash does not match.

3.5.2. Snapshot File

3.5.2.1. Purpose

A snapshot is intended to reflect the complete and current contents of the repository for a specific session and version. Therefore it MUST contain all objects from the repository current as of the time of the publication.

3.5.2.2. Cache Concerns

A snapshot reflects the content of the repository at a specific point in time, and for that reason can be considered immutable data. Snapshot files MUST be published at a URL that is unique to the specific session and serial.

Because these files never change, they MAY be cached indefinitely. However, in order to prevent that these files use a lot of space in caching infrastructure it is RECOMMENDED that a limited interval is used in the order of hours or days.

To avoid race conditions where an RP downloads a notification file moments before it’s updated, Repository Servers SHOULD retain old snapshot files for at least 5 minutes after a new notification file is published.

3.5.2.3. File Format and Validation

Example snapshot file:

```
<snapshot xmlns="http://www.ripe.net/rpki/rrdp"
       version="1"
       session_id="9df4b597-af9e-4dca-bdda-719cce2c4e28"
       serial="2">
  <publish uri="rsync://rpki.ripe.net/Alice/Bob.cer">
    ZXhhbXBsZTE=
  </publish>
  <publish uri="rsync://rpki.ripe.net/Alice/Alice.mft">
    ZXhhbXBsZTI=
  </publish>
  <publish uri="rsync://rpki.ripe.net/Alice/Alice.crl">
    ZXhhbXBsZTM=
  </publish>
</snapshot>
```

The following rules must be observed when creating or parsing snapshot files:

- A RP MUST reject any snapshot file that is not well-formed, or which does not conform to the RELAX NG schema outlined in Section 3.5.4 of this document.
- The XML namespace MUST be http://www.ripe.net/rpki/rrdp.
- The encoding MUST be US-ASCII.
- The version attribute in the notification root element MUST be 1
- The session_id attribute MUST match the expected session_id in the reference in the notification file.
- The serial attribute MUST match the expected serial in the reference in the notification file.
- Note that the publish element is defined in the publication protocol [I-D.ietf-sidr-publication]

3.5.3. Delta File

3.5.3.1. Purpose

An incremental delta file contains all changes for exactly one serial increment of the repository server. In other words a single delta will typically include all the new objects, updated objects and withdrawn objects that a Certification Authority sent to the repository server. In its simplest form the update could concern only a single object, but it is recommended that CAs send all changes

for one of their key pairs: i.e. updated objects as well as a new manifest and CRL as one atomic update message.

3.5.3.2. Cache Concerns

Deltas reflect the difference between two consecutive versions of a repository for a given session. For that reason deltas can be considered immutable data. Delta files MUST be published at a URL that is unique to the specific session and serial.

Because these files never change, they MAY be cached indefinitely. However, in order to prevent these files from using a lot of space in caching infrastructure it is RECOMMENDED that a limited interval is used in the order of hours or days.

To avoid race conditions where an RP downloads a notification file moments before it’s updated, Repository Servers SHOULD retain old delta files for at least 5 minutes after they are no no longer included in the latest notification file.

3.5.3.3. File Format and Validation

Example delta file:

```
  <delta xmlns="http://www.ripe.net/rpki/rrdp"
         version="1"
         session_id="9df4b597-af9e-4dca-bdda-719cce2c4e28"
         serial="3">
    <publish uri="rsync://rpki.ripe.net/repo/Alice/Alice.mft"
             hash="50d8...545c">
      ZXhhbXBsZTQ=
    </publish>
    <publish uri="rsync://rpki.ripe.net/repo/Alice/Alice.crl"
             hash="5fb1...6a56">
      ZXhhbXBsZTU=
    </publish>
    <withdraw uri="rsync://rpki.ripe.net/repo/Alice/Bob.cer"
              hash="caeb...15c1"/>
  </delta>
```

Note that a formal RELAX NG specification of this file format is included later in this document. A RP MUST NOT process any delta file that is incomplete or not well-formed.

The following validation rules must be observed when creating or parsing delta files:

- A RP MUST reject any delta file that is not well-formed, or which does not conform to the RELAX NG schema outlined in Section 3.5.4 of this document.
- The XML namespace MUST be http://www.ripe.net/rpki/rrdp.
- The encoding MUST be US-ASCII.
- The version attribute in the delta root element MUST be 1
- The session_id attribute MUST be a random version 4 UUID unique to this session
- The session_id attribute MUST match the expected session_id in the reference in the notification file.
- The serial attribute MUST match the expected serial in the reference in the notification file.
- Note that the publish and withdraw elements are defined in the publication protocol [I-D.ietf-sidr-publication]

3.5.4. XML Schema

The following is a RELAX NG compact form schema describing version 1 of this protocol.

```
#
# RelaxNG schema for RPKI Repository Delta Protocol (RRDP).
#

default namespace = "http://www.ripe.net/rpki/rrdp"

version = xsd:positiveInteger   { maxInclusive="1" }
serial  = xsd:nonNegativeInteger
uri     = xsd:anyURI
uuid    = xsd:string            { pattern = "[\-0-9a-fA-F]+" }
hash    = xsd:string            { pattern = "[0-9a-fA-F]+" }
base64  = xsd:base64Binary

# Notification file: lists current snapshots and deltas

start |= element notification {
  attribute version    { version },
  attribute session_id { uuid },
  attribute serial     { serial },
  element snapshot {
    attribute uri  { uri },
```

```
      attribute hash { hash }
    },
    element delta {
      attribute serial { serial },
      attribute uri    { uri },
      attribute hash   { hash }
    }*
  }

  # Snapshot segment: think DNS AXFR.

  start |= element snapshot {
    attribute version    { version },
    attribute session_id { uuid },
    attribute serial     { serial },
    element publish      {
      attribute uri { uri },
      base64
    }*
  }

  # Delta segment: think DNS IXFR.

  start |= element delta {
    attribute version    { version },
    attribute session_id { uuid },
    attribute serial     { serial },
    delta_element+
  }

  delta_element |= element publish  {
    attribute uri  { uri },
    attribute hash { hash }?,
    base64
  }

  delta_element |= element withdraw {
    attribute uri  { uri },
    attribute hash { hash }
  }

  # Local Variables:
  # indent-tabs-mode: nil
  # comment-start: "# "
  # comment-start-skip: "#[ \t]*"
  # End:
```

4. HTTPS considerations

It is RECOMMENDED that Relying Parties and Publication Servers follow the Best Current Practices outlined in [RFC7525] on the use of HTTP over TLS (https).

Note that a Man-in-the-Middle (MITM) cannot produce validly signed RPKI data, but they can perform withhold or replay attacks targeting an RP, and keep the RP from learning about changes in the RPKI. Because of this RPs SHOULD do TLS certificate and host name validation when they fetch from an RRDP Publication Server

However, such validation issues are often due to configuration errors, or a lack of a common TLS trust anchor. In these cases it would be better that the RP retrieves the signed RPKI data regardless, and performs validation on it.

Therefore RPs SHOULD log any TLS certificate or host name validation issues they find, so that an operator can investigate the cause. But the RP SHOULD continue to retrieve the data. The RP MAY choose to log this issue only when fetching the notification update file, but not when it subsequently fetches snapshot or delta files from the same host. Furthermore the RP MAY provide a way for operators to accept untrusted connections for a given host, after the cause has been identified.

5. Security Considerations

TBD

6. IANA Considerations

This document has no actions for IANA.

7. Acknowledgements

The authors would like to thank David Mandelberg for reviewing this document.

8. Normative References

[I-D.ietf-sidr-publication]
Weiler, S., Sonalker, A., and R. Austein, "A Publication Protocol for the Resource Public Key Infrastructure (RPKI)", draft-ietf-sidr-publication-08 (work in progress), March 2016.

[IANA-AD-NUMBERS]
"SMI Security for PKIX Access Descriptor",
<http://www.iana.org/assignments/smi-numbers/
smi-numbers.xhtml#smi-numbers-1.3.6.1.5.5.7.48>.

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

[RFC2818] Rescorla, E., "HTTP Over TLS", RFC 2818, DOI 10.17487/RFC2818, May 2000, <http://www.rfc-editor.org/info/rfc2818>.

[RFC6481] Huston, G., Loomans, R., and G. Michaelson, "A Profile for Resource Certificate Repository Structure", RFC 6481, DOI 10.17487/RFC6481, February 2012, <http://www.rfc-editor.org/info/rfc6481>.

[RFC6486] Austein, R., Huston, G., Kent, S., and M. Lepinski, "Manifests for the Resource Public Key Infrastructure (RPKI)", RFC 6486, DOI 10.17487/RFC6486, February 2012, <http://www.rfc-editor.org/info/rfc6486>.

[RFC6487] Huston, G., Michaelson, G., and R. Loomans, "A Profile for X.509 PKIX Resource Certificates", RFC 6487, DOI 10.17487/RFC6487, February 2012, <http://www.rfc-editor.org/info/rfc6487>.

[RFC6488] Lepinski, M., Chi, A., and S. Kent, "Signed Object Template for the Resource Public Key Infrastructure (RPKI)", RFC 6488, DOI 10.17487/RFC6488, February 2012, <http://www.rfc-editor.org/info/rfc6488>.

[RFC7231] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content", RFC 7231, DOI 10.17487/RFC7231, June 2014, <http://www.rfc-editor.org/info/rfc7231>.

[RFC7232] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Conditional Requests", RFC 7232, DOI 10.17487/RFC7232, June 2014, <http://www.rfc-editor.org/info/rfc7232>.

[RFC7525] Sheffer, Y., Holz, R., and P. Saint-Andre, "Recommendations for Secure Use of Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS)", BCP 195, RFC 7525, DOI 10.17487/RFC7525, May 2015, <http://www.rfc-editor.org/info/rfc7525>.

Authors' Addresses

Tim Bruijnzeels
RIPE NCC

Email: tim@ripe.net

Oleg Muravskiy
RIPE NCC

Email: oleg@ripe.net

Bryan Weber
Cobenian

Email: bryan@cobenian.com

Rob Austein
Dragon Research Labs

Email: sra@hactrn.net

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: September 14, 2017

T. Bruijnzeels
O. Muravskiy
RIPE NCC
B. Weber
Cobenian
R. Austein
Dragon Research Labs

March 13, 2017

RPKI Repository Delta Protocol (RRDP)
draft-ietf-sidr-delta-protocol-08

Abstract

In the Resource Public Key Infrastructure (RPKI), Certificate Authorities publish certificates, including end entity certificates, Certificate Revocation Lists (CRL), and RPKI signed objects to repositories. Relying Parties retrieve the published information from those repositories. This document specifies a new RPKI Repository Delta Protocol (RRDP) for this purpose. RRDP was specifically designed for scaling. It relies on a notification file which lists the current snapshot and delta files that can be retrieved using HTTP over TLS (HTTPS), and enables to use of CDNs or other caching infrastructure for the retrieval of these files.

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 September 14, 2017.

Table of Contents

1. Requirements notation

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

2. Introduction

In the Resource Public Key Infrastructure (RPKI), Certificate Authorities publish certificates [RFC6487], RPKI signed objects [RFC6488], manifests [RFC6486], and CRLs to repositories. CAs may have an embedded mechanism to publish to these repositories, or they may use a separate Repository Server and publication protocol. RPKI repositories are currently accessible using the [rsync] protocol, allowing Relying Parties to synchronise a local copy of the RPKI repository used for validation with the remote repositories [RFC6481].

[rsync] has proven valuable in the early deployment of RPKI, because it allowed operators to gain experience without the need to invent a custom protocol. However, operational experience has brought concerns to light that we wish to address here:

- [rsync] is designed to limit the amount of data that needs to be transferred between client and server. However the server needs to spend significant resources in terms of CPU and memory for every connection. This is a problem in an envisioned RPKI deployment where thousands of Relying Parties query a small number of central repositories, and it makes these repositories weak to denial of service attacks.

- A secondary concern is the lack of supported rsync server and client libraries. In practice all implementations have to make system calls to an rsync binary. This is inefficient, introduces fragility with regards to updates of this binary, makes it difficult to catch and report problems to operators, and it complicates software development and testing.

This document specifies an alternative repository access protocol based on notification, snapshot and delta files that a Relying Party can retrieve over the HTTPS protocol. This allows Relying Parties to perform either a full (re-)synchronisation of their local copy of the repository using snapshot files, or use delta files to keep their local repository updated after initial synchronisation. We call this the RPKI Repository Delta Protocol, or RRDP in short.

RRDP was designed to support scaling in RPKI’s asymmetric deployment. It is consistent (in terms of data structures) with the publication protocol [I-D.ietf-sidr-publication] and treats publication events of one or more repository objects as discrete events that can be communicated to Relying Parties. This approach helps to minimize the amount of data that traverses the network and thus helps minimize the amount of time until repository convergence occurs. RRDP also provides a standards based way to obtain consistent, point in time

views of a single repository, eliminating a number of consistency related issues. Finally, this approach allows these discrete events to be communicated as immutable files. This enables Repository Servers to pre-calculate these files only once for all clients - thus limiting the CPU and memory investments required, and enables the use of caching infrastructure to reduce the load on a repository server when a large number of Relying Parties are querying it.

This document allows the use of RRDP as an additional repository distribution mechanism for RPKI. In time RRDP may replace [rsync] as the only mandatory to implement repository distribution mechanism. However this transition is outside of the scope of this document.

3. RPKI Repository Delta Protocol Implementation

3.1. Informal Overview

Certification Authorities in the RPKI use a repository server to publish their RPKI products, such as manifests, CRLs, signed certificates and RPKI signed objects. This repository server may be remote, or embedded in the Certificate Authority engine itself. Certificates in the RPKI that use a repository server that supports RRDP include a special Subject Information Access (SIA) pointer referring to a notification file.

The notification file includes a globally unique session_id in the form of a version 4 UUID ([RFC4122]), and serial number that can be used by the Relying Party to determine if it and the repository are synchronised. Furthermore it includes a link to the most recent complete snapshot of current objects that are published by the repository server, and a list of links to delta files, for each revision starting at a point determined by the repository server, up to the current revision of the repository.

A Relying Party that learns about a notification file location for the first time can download it, and then proceed to download the latest snapshot file, and thus create a local copy of the repository that is in sync with the repository server. The Relying Party records the location of this notification file, the session_id and current serial number.

Relying Parties are encouraged to re-fetch this notification file at regular intervals, but not more often than once per minute. After re-fetching the notification file, the Relying Party may find that there are one or more delta files available that allow it to synchronise its local repository with the current state of the repository server. If no contiguous chain of deltas from the Relying Party’s serial to the latest repository serial is available, or if

the session_id has changed, the Relying Party performs a full resynchronisation instead.

As soon as the Relying Party fetches new content in this way it could start a validation process. An example of a reason why a Relying Party may not choose to do this immediately is because it has learned of more than one notification location and it prefers to complete all its updates before validating.

The repository server could use caching infrastructure to reduce its load, particularly because snapshots and deltas for any given session_id and serial number contain an immutable record of the state of the repository server at a certain point in time. For this reason these files can be cached indefinitely. Notification files are polled by Relying Parties to discover if updates exist, and for this reason notification files may not be cached for longer than one minute.

3.2. Certificate Authority Use

Certificate Authorities that use RRDP MUST include an instance of an SIA AccessDescription extension in resource certificates they produce, in addition to the ones defined in [RFC6487],

```
AccessDescription ::= SEQUENCE {
  accessMethod OBJECT IDENTIFIER,
  accessLocation GeneralName }
```

This extension MUST use an accessMethod of id-ad-rpkiNotify, see Section 6:

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

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

id-ad-rpkiNotify OBJECT IDENTIFIER ::= { id-ad 13 }
```

The accessLocation MUST be an HTTPS URI as defined in [RFC7230], that will point to the update notification file for the repository server that publishes the products of this Certificate Authority certificate.

3.3. Repository Server Use

3.3.1. Initialisation

When the repository server initialises it performs the following actions:

- The server MUST generate a new random version 4 UUID (see section 4.1.3 of [RFC4122]) to be used as the session_id
- The server MUST then generate a snapshot file for serial number ONE for this new session that includes all currently known published objects that the repository server is responsible for. Note that this snapshot file may contain zero publish elements at this point if no objects have been submitted for publication yet.
- This snapshot file MUST be made available at a URL that is unique to this session_id and serial number, so that it can be cached indefinitely. The format and caching concerns for snapshot files are explained in more detail in Section 3.5.2.
- After the snapshot file has been published the repository server MUST publish a new notification file that contains the new session_id, has serial number ONE, has one reference to the snapshot file that was just published, and that contains no delta references. The format and caching concerns for update notification files are explained in more detail in Section 3.5.1.

3.3.2. Publishing Updates

Whenever the repository server receives updates from a Certificate Authority it MUST generate new snapshot and delta files within one minute. If a Repository Server services a large number of Certificate Authorities it MAY choose to combine updates from multiple CAs. If a Repository Server combines updates in this way, it MUST ensure that publication never postponed for longer than one minute for any of the CAs involved.

Updates are processed as follows:

- The new repository serial number MUST be one greater than the current repository serial number.
- A new delta file MUST be generated for this new serial. This delta file MUST include all new, replaced and withdrawn objects for multiple CAs if applicable, as a single change set.
- This delta file MUST be made available at a URL that is unique to the current session_id and serial number, so that it can be cached indefinitely.

- The format and caching concerns for delta files are explained in more detail in Section 3.5.3.

- The repository server MUST also generate a new snapshot file for this new serial. This file MUST contain all "publish" elements for all current objects.

- The snapshot file MUST be made available at a URL that is unique to this session and new serial, so that it can be cached indefinitely.

- The format and caching concerns for snapshot files are explained in more detail in Section 3.5.2.

- Any older delta files that, when combined with all more recent delta files, will result in total size of deltas exceeding the size of the snapshot, MUST be excluded to avoid that Relying Parties download more data than necessary.

- A new notification file MUST now be created by the repository server. This new notification file MUST include a reference to the new snapshot file, and all delta files selected in the previous steps.

- The format and caching concerns for update notification files are explained in more detail in Section 3.5.1.

If the repository server is not capable of performing the above for some reason, then it MUST perform a full re-initialisation, as explained above in Section 3.3.1.

3.4. Relying Party Use

3.4.1. Processing the Update Notification File

When a Relying Party performs RPKI validation and learns about a valid certificate with an SIA entry for the RRDP protocol, it SHOULD use this protocol as follows.

The Relying Party MUST download the update notification file, unless an update notification file was already downloaded and processed from the same location in this validation run, or because a polling strategy was used (see Section 3.4.4).

It is RECOMMENDED that Relying Party uses a "User-Agent" header explained in section 5.5.3. of [RFC7231] to identify the name and version of the Relying Party software used. It is useful to track

capabilities of Relying Parties in the event of changes to the RPKI standards.

When the Relying Party downloads an update notification file it MUST verify the file format and validation steps described in section Section 3.5.1.3. If this verification fails, the file MUST be rejected and RRDP cannot be used. See Section 3.4.5 for considerations.

The Relying Party MUST verify whether the session_id matches the last known session_id for this update notification file location. Note that even though the session_id is a random UUID value, it alone MUST NOT be used by a Relying Party as a unique identifier of a session, but always together with the location of the notification file. The reason for this is that a malicious server can use an existing session_id from another Repository Server.

If the session_id matches the last known session_id, then a Relying Party MAY download and process missing delta files as described in Section 3.4.2, provided that all delta files for serial numbers between the last processed serial number and the current serial number in the notification file can be processed this way.

If the session_id matches the last known session_id, but delta files were not used, then the Relying Party MUST download and process the snapshot file on the update notification file as described in Section 3.4.3.

If the session_id does not match the last known session_id, the Relying Party MUST update its last known session_id to the value specified in the downloaded notification file. The Relying Party MUST then download and process the snapshot file specified in the downloaded update notification file as described in Section 3.4.3.

3.4.2. Processing Delta Files

If an update notification file contains a contiguous chain of links to delta files from the last processed serial number to the current serial number, then Relying Parties MUST attempt to download and process all delta files in order of serial number as follows.

When the Relying Party downloads a delta file it MUST verify the file format and perform validation steps described in Section 3.5.3.3. If this verification fails, the file MUST be rejected.

Furthermore the Relying Party MUST verify that the hash of the contents of this file matches the hash on the update notification

file that referenced it. In case of a mismatch of this hash, the file MUST be rejected.

If a Relying Party retrieved a delta file that is valid according to the above criteria, it performs the following actions:

- The Relying Party MUST verify that the session_id matches the session_id of the notification file. If the session_id values do not match the file MUST be rejected.

- The Relying Party MUST verify that the serial number of this delta file is exactly one greater than the last processed serial number for this session_id, and if not this file MUST be rejected.

- The Relying Party SHOULD add all publish elements to a local storage and update its last processed serial number to the serial number of this delta file.

- When a Relying Party encounters a "withdraw" element, or a "publish" element where an object is replaced, in a delta that it retrieves from a Repository Server, it MUST verify that the object to be withdrawn or replaced was retrieved from this same Repository Server, before applying the appropriate action. Failing to do so will leave the Relying Party vulnerable to malicious Repository Servers instructing it to delete or change arbitrary objects.

If any delta file is rejected Relying Parties MUST process the current Snapshot File instead, as described in Section 3.4.3.

3.4.3. Processing a Snapshot File

Snapshot Files MUST only be used if Delta Files are unavailable, or were rejected. As is ensured, if the process described in Section 3.4.1 is followed.

When the Relying Party downloads a snapshot file it MUST verify the file format and validation steps described in Section 3.5.2.3. If this verification fails, the file MUST be rejected.

Furthermore the Relying Party MUST verify that the hash of the contents of this file matches the hash on the update notification file that referenced it. In case of a mismatch of this hash, the file MUST be rejected.

If a Relying Party retrieved a snapshot file that is valid according to the above criteria, it performs the following actions:

- The Relying Party MUST verify that the session_id matches the session_id of the notification file. If the session_id values do not match the file MUST be rejected.

- The Relying Party MUST verify that the serial number of this snapshot file is greater than the last processed serial number for this session_id. If this fails the file MUST be rejected.

- The Relying Party SHOULD then add all publish elements to a local storage and update its last processed serial number to the serial number of this snapshot file.

If a Snapshot File is rejected that means that RRDP cannot be used. See Section 3.4.5 for considerations.

3.4.4. Polling the Update Notification File

Once a Relying Party has learned about the location, session_id and last processed serial number of repository that uses the RRDP protocol, the Relying Party MAY start polling the repository server for updates. However the Relying Party MUST NOT poll for updates more often than once every 1 minute, and in order to reduce data usage Relying Parties MUST use the "If-Modified-Since" header explained in section 3.3 of [RFC7232] in requests.

If a Relying Party finds that updates are available it SHOULD download and process the file as described in Section 3.4.1, and initiate a new RPKI object validation process. However, a detailed description of the RPKI object validation process itself is out of scope of this document.

3.4.5. Considerations Regarding Operational Failures in RRDP

If a Relying Party experiences any issues with retrieving or processing any of the files used in this protocol, it will be unable to retrieve new RPKI data from the affected Repository Server.

Relying Parties could attempt to use alternative repository access mechanisms, if they are available, according to the accessMethod element value(s) specified in the SIA of the associated certificate (see Section 4.8.8 of [RFC6487]).

Furthermore Relying Parties may wish to employ re-try strategies while fetching RRDP files. Relying Parties are also advised to keep old objects in their local cache so that validation can be done using old objects.

It is also recommendable that re-validation and retrieval is performed pro-actively before manifests or CRLs go stale, or certificates expire, to ensure that problems on the side of the Relying Party can be identified and resolved before they cause major concerns.

3.5. File Definitions

3.5.1. Update Notification File

3.5.1.1. Purpose

The update notification file is used by Relying Parties to discover whether any changes exist between the state of the repository and the Relying Party's cache. It describes the location of the files containing the snapshot and incremental deltas which can be used by the Relying Party to synchronise with the repository.

3.5.1.2. Cache Concerns

A repository server MAY use caching infrastructure to cache the notification file and reduce the load of HTTPS requests. However, since this file is used by Relying Parties to determine whether any updates are available the repository server SHOULD ensure that this file is not cached for longer than 1 minute. An exception to this rule is that it is better to serve a stale notification file, than no notification file.

How this is achieved exactly depends on the caching infrastructure used. In general a repository server may find certain HTTP headers to be useful, such as: "Cache-Control: max-age=60" (see Section 5.2 of [RFC7234]). Another approach can be to have the repository server push out new versions of the notification file to the caching infrastructure when appropriate.

In case of a high load on a repository server or its distribution network, the Cache-Control HTTP header, or a similar mechanism, MAY be used to suggest an optimal (for the repository server) poll interval for Relying Parties. However, setting it to an interval longer than 1 hour is NOT RECOMMENDED. Relying parties SHOULD align the suggested interval with their operational practices and the expected update frequency of RPKI repository data, and MAY discard suggested value.

3.5.1.3. File Format and Validation

Example notification file:

```
<notification xmlns="http://www.ripe.net/rpki/rrdp"
      version="1"
      session_id="9df4b597-af9e-4dca-bdda-719cce2c4e28"
      serial="3">
  <snapshot uri="https://host/9d-8/3/snapshot.xml" hash="AB"/>
  <delta serial="3" uri="https://host/9d-8/3/delta.xml" hash="CD"/>
  <delta serial="2" uri="https://host/9d-8/2/delta.xml" hash="EF"/>
</notification>
```

Note: URIs and hash values in this example are shortened because of formatting.

The following validation rules MUST be observed when creating or parsing notification files:

- A Relying Party MUST reject any update notification file that is not well-formed, or which does not conform to the RELAX NG schema outlined in Section 3.5.4 of this document.
- The XML namespace MUST be http://www.ripe.net/rpki/rrdp
- The encoding MUST be US-ASCII
- The version attribute in the notification root element MUST be 1
- The session_id attribute MUST be a random version 4 UUID ([RFC4122]), unique to this session
- The serial attribute MUST be an unbounded, unsigned positive integer in decimal format indicating the current version of the repository.
- The notification file MUST contain exactly one ’snapshot’ element for the current repository version.
- If delta elements are included they MUST form a contiguous sequence of serial numbers starting at a revision determined by the repository server, up to the serial number mentioned in the notification element. Note that the elements may not be ordered.
- The hash attribute in snapshot and delta elements MUST be the hexadecimal encoding of the SHA-256 [SHS] hash of the referenced file. The Relying Party MUST verify this hash when the file is retrieved and reject the file if the hash does not match.

3.5.2. Snapshot File

3.5.2.1. Purpose

A snapshot is intended to reflect the complete and current contents of the repository for a specific session and version. Therefore it MUST contain all objects from the repository current as of the time of the publication.

3.5.2.2. Cache Concerns

A snapshot reflects the content of the repository at a specific point in time, and for that reason can be considered immutable data. Snapshot files MUST be published at a URL that is unique to the specific session and serial.

Because these files never change, they MAY be cached indefinitely. However, in order to prevent that these files use a lot of space in caching infrastructure it is RECOMMENDED that a limited interval is used in the order of hours or days.

To avoid race conditions where a Relying Party downloads a notification file moments before it’s updated, Repository Servers SHOULD retain old snapshot files for at least 5 minutes after a new notification file is published.

3.5.2.3. File Format and Validation

Example snapshot file:

```
<snapshot xmlns="http://www.ripe.net/rpki/rrdp"
       version="1"
       session_id="9df4b597-af9e-4dca-bdda-719cce2c4e28"
       serial="2">
  <publish uri="rsync://rpki.ripe.net/Alice/Bob.cer">
    ZXhhbXBsZTE=
  </publish>
  <publish uri="rsync://rpki.ripe.net/Alice/Alice.mft">
    ZXhhbXBsZTI=
  </publish>
  <publish uri="rsync://rpki.ripe.net/Alice/Alice.crl">
    ZXhhbXBsZTM=
  </publish>
</snapshot>
```

The following rules MUST be observed when creating or parsing snapshot files:

- A Relying Party MUST reject any snapshot file that is not well-formed, or which does not conform to the RELAX NG schema outlined in Section 3.5.4 of this document.
- The XML namespace MUST be http://www.ripe.net/rpki/rrdp.
- The encoding MUST be US-ASCII.
- The version attribute in the notification root element MUST be 1
- The session_id attribute MUST match the expected session_id in the reference in the notification file.
- The serial attribute MUST match the expected serial in the reference in the notification file.
- Note that the publish element is similar to the publish element defined in the publication protocol [I-D.ietf-sidr-publication]. However, the "tag" attribute is not used here because it is not relevant to Relying Parties. The "hash" attribute is not used here because this file represents a complete current state of the repository, and therefore it is not relevant to know which existing RPKI object (if any) is updated.

3.5.3. Delta File

3.5.3.1. Purpose

An incremental delta file contains all changes for exactly one serial increment of the repository server. In other words a single delta will typically include all the new objects, updated objects and withdrawn objects that a Certification Authority sent to the repository server. In its simplest form the update could concern only a single object, but it is RECOMMENDED that CAs send all changes for one of their key pairs (updated objects as well as a new manifest and CRL) as one atomic update message.

3.5.3.2. Cache Concerns

Deltas reflect the difference between two consecutive versions of a repository for a given session. For that reason deltas can be considered immutable data. Delta files MUST be published at a URL that is unique to the specific session and serial.

Because these files never change, they MAY be cached indefinitely. However, in order to prevent these files from using a lot of space in caching infrastructure it is RECOMMENDED that a limited interval is used in the order of hours or days.

To avoid race conditions where a Relying Party downloads a notification file moments before it's updated, Repository Servers SHOULD retain old delta files for at least 5 minutes after they are no longer included in the latest notification file.

3.5.3.3. File Format and Validation

Example delta file:

```
<delta xmlns="http://www.ripe.net/rpki/rrdp"
       version="1"
       session_id="9df4b597-af9e-4dca-bdda-719cce2c4e28"
       serial="3">
  <publish uri="rsync://rpki.ripe.net/repo/Alice/Alice.mft"
           hash="50d8...545c">
    ZXhhbXBsZTQ=
  </publish>
  <publish uri="rsync://rpki.ripe.net/repo/Alice/Alice.crl"
           hash="5fb1...6a56">
    ZXhhbXBsZTU=
  </publish>
  <withdraw uri="rsync://rpki.ripe.net/repo/Alice/Bob.cer"
            hash="caeb...15c1"/>
</delta>
```

Note that a formal RELAX NG specification of this file format is included later in this document. A Relying Party MUST NOT process any delta file that is incomplete or not well-formed.

The following validation rules MUST be observed when creating or parsing delta files:

- A Relying Party MUST reject any delta file that is not well-formed, or which does not conform to the RELAX NG schema outlined in Section 3.5.4 of this document.
- The XML namespace MUST be http://www.ripe.net/rpki/rrdp.
- The encoding MUST be US-ASCII.
- The version attribute in the delta root element MUST be 1
- The session_id attribute MUST be a random version 4 UUID unique to this session
- The session_id attribute MUST match the expected session_id in the reference in the notification file.

- The serial attribute MUST match the expected serial in the reference in the notification file.

- Note that the publish element is similar to the publish element defined in the publication protocol [I-D.ietf-sidr-publication]. However, the "tag" attribute is not used here because it is not relevant to Relying Parties.

3.5.4. XML Schema

The following is a RELAX NG compact form schema describing version 1 of this protocol.

```
#
# RelaxNG schema for RPKI Repository Delta Protocol (RRDP).
#

default namespace = "http://www.ripe.net/rpki/rrdp"

version = xsd:positiveInteger   { maxInclusive="1" }
serial  = xsd:positiveInteger
uri     = xsd:anyURI
uuid    = xsd:string            { pattern = "[\-0-9a-fA-F]+" }
hash    = xsd:string            { pattern = "[0-9a-fA-F]+" }
base64  = xsd:base64Binary

# Notification file: lists current snapshots and deltas

start |= element notification {
  attribute version    { version },
  attribute session_id { uuid },
  attribute serial     { serial },
  element snapshot {
    attribute uri  { uri },
    attribute hash { hash }
  },
  element delta {
    attribute serial { serial },
    attribute uri    { uri },
    attribute hash   { hash }
  }*
}

# Snapshot segment: think DNS AXFR.

start |= element snapshot {
  attribute version    { version },
  attribute session_id { uuid },
```

```
    attribute serial     { serial },
    element publish      {
      attribute uri { uri },
      base64
    }*
  }

  # Delta segment: think DNS IXFR.

  start |= element delta {
    attribute version    { version },
    attribute session_id { uuid },
    attribute serial     { serial },
    delta_element+
  }

  delta_element |= element publish  {
    attribute uri  { uri },
    attribute hash { hash }?,
    base64
  }

  delta_element |= element withdraw {
    attribute uri  { uri },
    attribute hash { hash }
  }

  # Local Variables:
  # indent-tabs-mode: nil
  # comment-start: "# "
  # comment-start-skip: "#[ \t]*"
  # End:
```

4. Operational Considerations

4.1. Compatibility with previous standards

This protocol has been designed to replace rsync as a distribution mechanism of an RPKI repository. However, it is also designed to co-exist with existing implementations based on rsync, to enable smooth transition from one distribution mechanism to another.

For every repository object listed in the snapshot and delta files both the hash of the object's content and the rsync URI [RFC5781] of its location in the repository are listed. This makes it possible to distribute the same RPKI repository, represented by a set of files on a filesystem, using both rsync and RRDP. It also enables Relying

Parties tools to query, combine, and consequently validate objects from repositories of different types.

4.2. Distribution considerations

One of the design goals of RRDP was to minimise load on a repository server while serving clients. To achieve this, neither the content, nor the URLs of the snapshot and delta files are modified after they have been published in the notification file. This allows their effective distribution, either by a single HTTP server, or using a Content Distribution Network (CDN).

The RECOMMENDED way for Relying Parties to keep up with the repository updates is to poll the Update Notification File for changes. The content of that file is updated with every new serial version of a repository (while its URL remains stable). To effectively implement distribution of the notification file, an "If-Modified-Since" HTTP request header is required to be present in all requests for notification file (see Section 3.4.4.) Therefore it is RECOMMENDED that Relying Party tools implement a mechanism to keep track of a previous successful fetch of a notification file.

Implementations of RRDP should also take care of not producing new versions of the repository (and subsequently, new Notification, Snapshot and Delta files) too often. Usually the maintenance of the RPKI repository includes regular updates of manifest and CRL objects, performed on a schedule. This often results in bursts of repository updates during a short period of time. Since the Relying Parties are required to poll for the Update Notification File not more often than once per minute (Section 3.4.4), it is not practical to generate new serial versions of the repository much more often than 1 per minute. It is allowed to combine multiple updates, possibly from different CAs, into a new serial repository version (Section 3.3.2). This will significantly shorten the size of the Update Notification File and total amount of data distributed to all Relying Parties.

4.3. HTTPS considerations

Note that a Man-in-the-Middle (MITM) cannot produce validly signed RPKI data, but can perform withhold or replay attacks targeting a Relying Party, and keep the Relying Party from learning about changes in the RPKI. Because of this Relying Parties SHOULD do TLS certificate and host name validation when they fetch from an RRDP Repository Server.

Relying Party tools SHOULD log any TLS certificate or host name validation issues found, so that an operator can investigate the cause. However, such validation issues are often due to

configuration errors, or a lack of a common TLS trust anchor. In these cases it is better if the Relying Party retrieves the signed RPKI data regardless, and performs validation on it. Therefore Relying Party MUST continue to retrieve the data in case of errors. The Relying Party MAY choose to log encountered issues only when fetching the notification update file, but not when it subsequently fetches snapshot or delta files from the same host. Furthermore the Relying Party MAY provide a way for operators to accept untrusted connections for a given host, after the cause has been identified.

It is RECOMMENDED that Relying Parties and Repository Servers follow the Best Current Practices outlined in [RFC7525] on the use of HTTP over TLS (HTTPS) [RFC7230]. Relying Parties SHOULD do TLS certificate and host name validation using subjectAltName dNSName identities as described in [RFC6125]. The rules and guidelines defined in [RFC6125] apply here, with the following considerations:

- Relying Parties and Repository Servers SHOULD support the DNS-ID identifier type. The DNS-ID identifier type SHOULD be present in Repository Server certificates.

- DNS names in Repository Server certificates SHOULD NOT contain the wildcard character "*".

- A CN field may be present in Repository Server certificates's subject name, but SHOULD NOT be used for authentication within the rules described in [RFC6125].

- This protocol does not require the use of SRV-IDs.

- This protocol does not require the use of URI-IDs.

Note however that this validation is done on a best effort basis, and serves to highlight potential issues, but RPKI object security does not depend on this. Therefore Relying Parties MAY deviate from the validation steps listed above.

5. Security Considerations

RRDP deals exclusively with transfer of RPKI objects from a repository server to a Relying Party. The trust relation between a Certificate Authority and its repository server is out of scope for this document. However, it should be noted that from a Relying Party point of view all RPKI objects (certificates, CRLs, and CMS-wrapped objects) are already covered by object security mechanisms including signed manifests. This allows validation of these objects even though the repository server itself is not trusted. This document makes no change to RPKI validation procedures per se.

The original RPKI transport protocol is rsync, which offers no channel security mechanism. RRDP replaces the use of rsync by HTTPS; while the channel security mechanism underlying RRDP (HTTPS) is not a cure-all, it does make some forms of denial of service attack more difficult for the attacker. HTTPS issues are discussed in more detail in Section 4.3.

Supporting both RRDP and rsync necessarily increases the number of opportunities for a malicious RPKI Certificate Authority to perform denial of service attacks on Relying Parties, by expanding the number of URIs which the Relying Party may need to contact in order to complete a validation run. However, other than the relative cost of HTTPS versus rsync, adding RRDP to the mix does not change this picture significantly: with either RRDP or rsync a malicious Certificate Authority can supply an effectively infinite series of URIs for the Relying Party to follow. The only real solution to this is for the Relying Party to apply some kind of bound to the amount of work it is willing to do. Note also that the attacker in this scenario must be an RPKI Certificate Authority, since otherwise the normal RPKI object security checks would reject the malicious URIs.

Processing costs for objects retrieved using RRDP may be somewhat different from the same objects retrieved using rsync: because RRDP treats an entire set of changes as a unit (one "delta"), it may not be practical to start processing any of the objects in the delta until the entire delta has been received. With rsync, by contrast, incremental processing may be easy, but the overall cost of transfer may be higher, as may be the number of corner cases in which the Relying Party retrieves some but not all of the updated objects. Overall, RRDP's behavior is closer to a proper transactional system, which (probably) leads to an overall reliability increase.

RRDP is designed to scale much better than rsync. In particular, RRDP is designed to allow use of HTTPS caching infrastructure to reduce load on primary Repository Servers and increase resilience against denial of service attacks on the RPKI publication service.

6. IANA Considerations

IANA is requested to update the reference for id-ad-rpkiNotify to this document in the PKIX Access Descriptor registry [IANA-AD-NUMBERS].

7. Acknowledgements

The authors would like to thank David Mandelberg for reviewing this document.

8. References

8.1. Normative References

[I-D.ietf-sidr-publication]
Weiler, S., Sonalker, A., and R. Austein, "A Publication Protocol for the Resource Public Key Infrastructure (RPKI)", draft-ietf-sidr-publication-12 (work in progress), March 2017.

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

[RFC4122] Leach, P., Mealling, M., and R. Salz, "A Universally Unique IDentifier (UUID) URN Namespace", RFC 4122, DOI 10.17487/RFC4122, July 2005, <http://www.rfc-editor.org/info/rfc4122>.

[RFC5781] Weiler, S., Ward, D., and R. Housley, "The rsync URI Scheme", RFC 5781, DOI 10.17487/RFC5781, February 2010, <http://www.rfc-editor.org/info/rfc5781>.

[RFC6125] Saint-Andre, P. and J. Hodges, "Representation and Verification of Domain-Based Application Service Identity within Internet Public Key Infrastructure Using X.509 (PKIX) Certificates in the Context of Transport Layer Security (TLS)", RFC 6125, DOI 10.17487/RFC6125, March 2011, <http://www.rfc-editor.org/info/rfc6125>.

[RFC6481] Huston, G., Loomans, R., and G. Michaelson, "A Profile for Resource Certificate Repository Structure", RFC 6481, DOI 10.17487/RFC6481, February 2012, <http://www.rfc-editor.org/info/rfc6481>.

[RFC6487] Huston, G., Michaelson, G., and R. Loomans, "A Profile for X.509 PKIX Resource Certificates", RFC 6487, DOI 10.17487/RFC6487, February 2012, <http://www.rfc-editor.org/info/rfc6487>.

[RFC7230] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing", RFC 7230, DOI 10.17487/RFC7230, June 2014, <http://www.rfc-editor.org/info/rfc7230>.

[RFC7231] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content", RFC 7231, DOI 10.17487/RFC7231, June 2014, <http://www.rfc-editor.org/info/rfc7231>.

[RFC7232] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Conditional Requests", RFC 7232, DOI 10.17487/RFC7232, June 2014, <http://www.rfc-editor.org/info/rfc7232>.

[RFC7234] Fielding, R., Ed., Nottingham, M., Ed., and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Caching", RFC 7234, DOI 10.17487/RFC7234, June 2014, <http://www.rfc-editor.org/info/rfc7234>.

[RFC7525] Sheffer, Y., Holz, R., and P. Saint-Andre, "Recommendations for Secure Use of Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS)", BCP 195, RFC 7525, DOI 10.17487/RFC7525, May 2015, <http://www.rfc-editor.org/info/rfc7525>.

[SHS] National Institute of Standards and Technology, "Secure Hash Standard", March 2012, <http://csrc.nist.gov/publications/fips/fips180-4/fips-180-4.pdf>.

8.2. Informative References

[IANA-AD-NUMBERS] "SMI Security for PKIX Access Descriptor", <http://www.iana.org/assignments/smi-numbers/smi-numbers.xhtml#smi-numbers-1.3.6.1.5.5.7.48>.

[RFC6486] Austein, R., Huston, G., Kent, S., and M. Lepinski, "Manifests for the Resource Public Key Infrastructure (RPKI)", RFC 6486, DOI 10.17487/RFC6486, February 2012, <http://www.rfc-editor.org/info/rfc6486>.

[RFC6488] Lepinski, M., Chi, A., and S. Kent, "Signed Object Template for the Resource Public Key Infrastructure (RPKI)", RFC 6488, DOI 10.17487/RFC6488, February 2012, <http://www.rfc-editor.org/info/rfc6488>.

[rsync] "Rsync home page", <https://rsync.samba.org>.

Authors' Addresses

Tim Bruijnzeels
RIPE NCC

Email: tim@ripe.net

Oleg Muravskiy
RIPE NCC

Email: oleg@ripe.net

Bryan Weber
Cobenian

Email: bryan@cobenian.com

Rob Austein
Dragon Research Labs

Email: sra@hactrn.net

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: January 9, 2017

G. Huston
G. Michaelson
APNIC
C. Martinez
LACNIC
T. Bruijnzeels
RIPE NCC
A. Newton
ARIN
D. Shaw
AFRINIC

July 8, 2016

RPKI Validation Reconsidered
draft-ietf-sidr-rpki-validation-reconsidered-06

Abstract

This document proposes an update to the certificate validation procedure specified in RFC 6487 that reduces aspects of operational fragility in the management of certificates in the RPKI, while retaining essential security features.

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 January 9, 2017.

Copyright Notice

Table of Contents

1. Introduction

This document proposes an update to the certificate validation procedure specified in [RFC6487] that reduces aspects of operational fragility in the management of certificates in the RPKI, while retaining essential security features.

2. Certificate Validation in the RPKI

As currently defined in section 7.2 of [RFC6487], validation of PKIX certificates that conform to the RPKI profile relies on the use of a path validation process where each certificate in the validation path is required to meet the certificate validation criteria.

These criteria require, in particular, that the Internet Number Resources (INRs) of each certificate in the validation path are "encompassed" by INRs on the issuing certificate. The first certificate in the path is required to be a trust anchor, and its resources are considered valid by definition.

For example, in the following sequence:

```
  Certificate 1 (trust anchor):
    Issuer TA,
    Subject TA,
    Resources 192.0.2.0/24, 198.51.100.0/24,
              2001:db8::/32, AS64496-AS64500

  Certificate 2:
   Issuer TA,
   Subject CA1,
   Resources 192.0.2.0/24, 198.51.100.0/24, 2001:db8::/32

  Certificate 3:
   Issuer CA1,
   Subject CA2,
   Resources 192.0.2.0/24, 2001:db8::/32

  ROA 1:
   Embedded Certificate 4 (EE certificate):
    Issuer CA2,
    Subject R1,
    Resources 192.0.2.0/24

    Prefix 192.0.2.0/24, Max Length 24, ASN 64496
```

All certificates in this scenario are considered valid since the INRs of each certificate are encompassed by those of the issuing certificate. ROA1 is valid because the specified prefix is encompassed by the embedded EE certificate, as required by [RFC6482].

3. Operational Considerations

The allocations recorded in the RPKI change as a result of resource transfers. For example, the CAs involved in transfer might choose to modify CA certificates in an order that causes some of these certificates to "over-claim" temporarily. A certificate is said to "over-claim" if it includes INRs not contained in the INRs of the CA that issued the certificate in question.

It may also happen that a child CA does not voluntarily request a shrunk resource certificate when resources are being transferred or reclaimed by the parent. Furthermore operational errors that may occur during management of RPKI databases also may create CA certificates that, temporarily, no longer encompass all of the INRs of subordinate certificates.

Consider the following sequence:

```
Certificate 1 (trust anchor):
 Issuer TA,
 Subject TA,
 Resources 192.0.2.0/24, 198.51.100.0/24,
           2001:db8::/32, AS64496-AS64500

Certificate 2:
 Issuer TA,
 Subject CA1,
 Resources 192.0.2.0/24, 2001:db8::/32

Certificate 3 (invalid):
 Issuer CA1,
 Subject CA2,
 Resources 192.0.2.0/24, 198.51.100.0/24, 2001:db8::/32

ROA 1 (invalid):
 Embedded Certificate 4 (EE certificate):
  Issuer CA2,
  Subject R1,
  Resources 192.0.2.0/24

  Prefix 192.0.2.0/24, Max Length 24, ASN 64496
```

Here Certificate 2 from the previous example was re-issued by TA to CA1 and the prefix 198.51.100.0/24 was removed. However, CA1 failed to re-issue a new Certificate 3 to CA2. As a result Certificate 3 is now over-claiming and considered invalid; by recursion the embedded Certificate 4 used for ROA1 is also invalid. And ROA1 is invalid because the specified prefix contained in the ROA is no longer encompassed by a valid embedded EE certificate, as required by [RFC6482]

However, it should be noted that ROA1 does not make use of any of the address resources that were removed from CA1’s certificate, and thus it would be desirable if ROA1 could still be viewed as valid. Technically CA1 should re-issue a Certificate 3 to CA2 without 198.51.100.0/24, and then ROA1 would be considered valid according to [RFC6482]. But as long as CA1 does not take this action, ROA1 remains invalid. It would be preferable if ROA1 could be considered valid, since the assertion it makes was not affected by the reduced scope of CA1’s certificate.

4. An Amended RPKI Certification Validation Process

4.1. Verified Resource Sets

The problem described above can be considered as a low probability problem today. However the potential impact on routing security would be high if an over-claiming occurred near the apex of the RPKI hierarchy, as this would invalidate the entirety of the sub-tree located below this point.

The changes proposed here to the validation procedure in [RFC6487] do not change the probability of this problem, but they do limit the impact to just the over-claimed resources. This revised validation algorithm is intended to avoid causing CA certificates to be treated as completely invalid as a result of over-claims. However, these changes are designed to not degrade the security offered by the RPKI. Specifically, ROAs and router certificates will be treated as valid only if all of the resources contained in them are encompassed by all superior certificates along a path to a trust anchor.

The way this is achieved conceptually is by maintaining Verified Resource Set (VRS) for each certificate that is separate from the INRs found in the [RFC3779] resource extension in the certificate.

4.2. Changes to existing standards

4.2.1. Resource Certificate Path Validation

The following is an amended specification to be used in place of section 7.2 of [RFC6487].

The following algorithm is employed to validate CA and EE resources certificates. It is modeled on the path validation algorithm from [RFC5280], but modified to make use of the IP Address Delegation and AS Identifier Delegation Extensions from [RFC3779].

There are two inputs to the validation algorithm:

1. a trust anchor

2. a certificate to be validated

The algorithm is initialized with two new variables for use in the RPKI: Validated Resource Set-IP (VRS-IP) and Validated Resource Set-AS (VRS-AS). These sets are used to track the set of INRs (IP address space and AS Numbers) that are considered valid for each CA certificate. The VRS-IP and VRS-AS sets are initially set to the IP Address Delegation and AS Identifier Delegation values, respectively, from the trust anchor used to perform validation.

This path validation algorithm verifies, among other things, that a prospective certification path (a sequence of n certificates) satisfies the following conditions:

a. for all 'x' in {1, ..., n-1}, the subject of certificate 'x' is the issuer of certificate ('x' + 1);

b. certificate '1' is issued by a trust anchor;

c. certificate 'n' is the certificate to be validated; and

d. for all 'x' in {1, ..., n}, certificate 'x' is valid.

Certificate validation requires verifying that all of the following conditions hold, in addition to the certification path validation criteria specified in Section 6 of [RFC5280].

1. The signature of certificate x (x>1) is verified using the public key of the issuer's certificate (x-1), using the signature algorithm specified for that public key (in certificate x-1).

2. The current time lies within the interval defined by the NotBefore and NotAfter values in the Validity field of certificate x.

3. The Version, Issuer, and Subject fields of certificate x satisfy the constraints established in Section 4.1-4.7 of this specification.

4. Certificate x contains all the extensions that MUST be present, as defined in Section 4.8 of this specification. The value(s) for each of these extensions MUST be satisfy the constraints established for each extension in the respective sections. Any extension not identified in Section 4.8 MUST NOT appear in certificate x.

5. Certificate x MUST NOT have been revoked, i.e., it MUST NOT appear on a CRL issued by the CA represented by certificate x-1

6. Compute the VRS-IP and VRS-AS set values as indicated below:

   * If the IP Address Delegation extension is present in certificate x, compute the intersection of the resources between this extension and the value of the VRS-IP computed for certificate x-1.

   * If the IP Address Delegation extension is absent in certificate x, set the VRS-IP to NULL.

* If the AS Identifier Delegation extension is present in certificate x, compute the intersection of the resources between this extension and the value of the VRS-AS computed for certificate x-1

* If the AS Identifier Delegation extension is absent in certificate x, set the VRS-AS to NULL.

* If x = n (i.e., this is the certificate being validated), then:

  1. If IP Address Delegation extension is present, it is replaced with the intersection of the values from that extension and the current value of the VRS-IP.

  2. If an AS Identifier Delegation extension is present, it is replaced with the intersection of the values from that extension and the current value of the VRS-IP.

* If an RP is caching the results of validation, these values MAY be stored along with the certificate, to facilitate incremental validation based on cached results.

These rules allow a CA certificate to contain resources that are not present in (all of) the certificates along the path from the trust anchor to the CA certificate. If none of the resources in the CA certificate are present in all certificates along the path, no subordinate certificates could be valid. However, the certificate is not immediately rejected as this may be a transient condition. Not immediately rejecting the certificate does not result in a security problem because the associated VRS sets accurately reflect the resources validly associated with the certificate in question.

4.2.2. ROA Validation

Section 4 of [RFC6482] currently has the following text on the validation of resources on a ROA:

o The IP address delegation extension [RFC3779] is present in the end-entity (EE) certificate (contained within the ROA), and each IP address prefix(es) in the ROA is contained within the set of IP addresses specified by the EE certificate’s IP address delegation extension.

The following is an amended specification to be used in place of this text.

- The IP address delegation extension [RFC3779] is present in the end-entity (EE) certificate (contained within the ROA), and each IP address prefix(es) in the ROA is contained within the VRS-IP set that is specified as an outcome of EE certificate validation.

Note that this ensures that ROAs can be valid only, if all IP address prefixes in the ROA are encompassed by the VRS-IP of all certificates along the path to the trust anchor used to verify it.

Operators MAY issue separate ROAs for each IP address prefix, so that the loss of on IP address prefix from the VRS-IP of any certificate along the path to the trust anchor would not invalidate authorizations for other IP address prefixes.

4.2.3. BGPsec Router Certificate Validation

BGPsec Router Certificate Validation is defined in section 3.3 of [I-D.ietf-sidr-bgpsec-pki-profiles]. Path validation defined section 7 of [RFC6487] is used as the first step in validation, and a number of additional constraints are applied.

We request that the authors add the following constraint:

- The VRS-AS of BGPsec Router Certificates MUST encompass all ASNs in the AS Resource Identifier Delegation extension.

Furthermore we request that the authors include text instructing operators that they MAY issue separate BGPsec Router Certificates for different ASNs, so that the loss of on ASN from the VRS-AS of any certificate along the path to the trust anchor would not invalidate router keys for other ASNs.

4.3. An example

Consider the following example under the amended approach:

```
  Certificate 1 (trust anchor):
   Issuer TA,
   Subject TA,
   Resources 192.0.2.0/24, 198.51.100.0/24,
             2001:db8::/32, AS64496-AS64500

    Verified Resource Set: 192.0.2.0/24, 198.51.100.0/24,
                           2001:db8::/32, AS64496-AS64500
    Warnings: none

  Certificate 2:
   Issuer TA,
```

```
      Subject CA1,
      Resources 192.0.2.0/24, 2001:db8::/32, AS64496

       Verified Resource Set: 192.0.2.0/24,
                              2001:db8::/32, AS64496
       Warnings: none

     Certificate 3:
      Issuer CA1,
      Subject CA2,
      Resources 192.0.2.0/24, 198.51.100.0/24, AS64496

       Verified Resource Set: 192.0.2.0/24, AS64496
       Warnings: over-claim for 198.51.100.0/24

     ROA 1 (valid):
      Embedded Certificate 4 (EE certificate):
       Issuer CA2,
       Subject R1,
       Resources 192.0.2.0/24

        Verified resources: 192.0.2.0/24
        Warnings: none

        Prefix 192.0.2.0/24, Max Length 24, ASN 64496

       ROA1 is considered valid because the prefix matches the Verified
       Resource Set on the embedded EE certificate, as required by
       RFC 6482.

     ROA 2 (invalid):
      Embedded Certificate 5 (EE certificate invalid):
       Issuer CA2,
       Subject R2,
       Resources 198.51.100.0/24

        EE certificate is invalid due to over-claim for 198.51.100.0/24

        Prefix 198.51.100.0/24, Max Length 24, ASN 64496

       ROA2 is considered invalid because he embedded EE certificate is
       considered invalid.

     BGPSec Certificate 1 (valid):
      Issuer CA2
      Subject ROUTER-64496
      Resources AS64496
```

```
       Verified resources: AS64496
       Warnings: none

     BGPSec Certificate 2 (invalid):
      Issuer CA2
      Subject ALL-ROUTERS
      Resources AS64496-AS64497

       EE certificate is invalid due to over-claim for AS64497

       This problem can be mitigated by issuing separate certificates
       for each AS number.
```

5. Security Considerations

The authors believe that the revised validation algorithm introduces no new security vulnerabilities into the RPKI.

6. IANA Considerations

No updates to the registries are suggested by this document.

7. Acknowledgements

The authors would like to thank Stephen Kent for reviewing and contributing to this document.

8. References

8.1. Normative References

[I-D.ietf-sidr-bgpsec-pki-profiles]
Reynolds, M., Turner, S., and S. Kent, "A Profile for BGPsec Router Certificates, Certificate Revocation Lists, and Certification Requests", draft-ietf-sidr-bgpsec-pki-profiles-17 (work in progress), June 2016.

[RFC3779] Lynn, C., Kent, S., and K. Seo, "X.509 Extensions for IP Addresses and AS Identifiers", RFC 3779, DOI 10.17487/RFC3779, June 2004, <http://www.rfc-editor.org/info/rfc3779>.

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

[RFC6482] Lepinski, M., Kent, S., and D. Kong, "A Profile for Route Origin Authorizations (ROAs)", RFC 6482, DOI 10.17487/RFC6482, February 2012, <http://www.rfc-editor.org/info/rfc6482>.

[RFC6487] Huston, G., Michaelson, G., and R. Loomans, "A Profile for X.509 PKIX Resource Certificates", RFC 6487, DOI 10.17487/RFC6487, February 2012, <http://www.rfc-editor.org/info/rfc6487>.

8.2. Informative References

[RFC3849] Huston, G., Lord, A., and P. Smith, "IPv6 Address Prefix Reserved for Documentation", RFC 3849, DOI 10.17487/RFC3849, July 2004, <http://www.rfc-editor.org/info/rfc3849>.

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

[RFC5737] Arkko, J., Cotton, M., and L. Vegoda, "IPv4 Address Blocks Reserved for Documentation", RFC 5737, DOI 10.17487/RFC5737, January 2010, <http://www.rfc-editor.org/info/rfc5737>.

Authors' Addresses

Geoff Huston
Asia Pacific Network Information Centre
6 Cordelia St
South Brisbane, QLD 4101
Australia

Phone: +61 7 3858 3100
Email: gih@apnic.net

George Michaelson
Asia Pacific Network Information Centre
6 Cordelia St
South Brisbane, QLD 4101
Australia

Phone: +61 7 3858 3100
Email: ggm@apnic.net

Carlos M. Martinez
Latin American and Caribbean IP Address Regional Registry
Rambla Mexico 6125
Montevideo 11400
Uruguay

Phone: +598 2604 2222
Email: carlos@lacnic.net

Tim Bruijnzeels
RIPE Network Coordination Centre
Singel 258
Amsterdam 1016 AB
The Netherlands

Email: tim@ripe.net

Andrew Lee Newton
American Registry for Internet Numbers
3635 Concorde Parkway
Chantilly, VA 20151
USA

Email: andy@arin.net

Daniel Shaw
African Network Information Centre (AFRINIC)
11th Floor, Standard Chartered Tower
Cybercity, Ebene
Mauritius

Phone: +230 403 51 00
Email: daniel@afrinic.net

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: June 24, 2018

G. Huston
G. Michaelson
APNIC
C. Martinez
LACNIC
T. Bruijnzeels
RIPE NCC
A. Newton
ARIN
D. Shaw
AFRINIC

December 21, 2017

RPKI Validation Reconsidered
draft-ietf-sidr-rpki-validation-reconsidered-10

Abstract

This document specifies an alternative to the certificate validation procedure specified in RFC 6487 that reduces aspects of operational fragility in the management of certificates in the RPKI, while retaining essential security features.

Where the procedure specified in RFC 6487 requires that Resource Certificates are rejecting entirely if they are found to over-claim any resources not contained on the issuing certificate, the validation process defined here allows an issuing Certificate Authority to chose to communicate that such Resource Certificates should be accepted for the intersection of their resources and the issuing certificate.

It should be noted that the validation process defined here considers validation under a single Trust Anchor only. In particular, concerns regarding over-claims where multiple configured Trust Anchors claim overlapping resources are considered out of scope for this document.

This choice is signalled by form of a set of alternative Object Identifiers (OIDs) of RFC 3779 X.509 Extensions for IP Addresses and AS Identifiers, and certificate policy for the Resource Public Key Infrastructure (RFC 6484). It should be noted that in case these OIDs are not used for any certificate under a Trust Anchor, the validation procedure defined here has the same outcome as the procedure defined in RFC 6487

Furthermore this document provides an alternative to ROA (RFC 6482), and BGPSec Router Certificate (BGPSec PKI Profiles - publication requested) validation.

Status of This Memo

Copyright Notice

Table of Contents

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. Certificate Validation in the RPKI

As currently defined in section 7.2 of [RFC6487], validation of PKIX certificates that conform to the RPKI profile relies on the use of a path validation process where each certificate in the validation path is required to meet the certificate validation criteria.

These criteria require, in particular, that the Internet Number Resources (INRs) of each certificate in the validation path are "encompassed" by INRs on the issuing certificate. The first certificate in the path is required to be a trust anchor, and its resources are considered valid by definition.

For example, in the following sequence:

Certificate 1 (trust anchor):
Issuer TA,
Subject TA,
Resources 192.0.2.0/24, 198.51.100.0/24,
2001:db8::/32, AS64496-AS64500

Certificate 2:
Issuer TA,
Subject CA1,
Resources 192.0.2.0/24, 198.51.100.0/24, 2001:db8::/32

Certificate 3:
Issuer CA1,
Subject CA2,
Resources 192.0.2.0/24, 198.51.100.0/24, 2001:db8::/32

ROA 1:
Embedded Certificate 4 (EE certificate):
Issuer CA2,
Subject R1,
Resources 192.0.2.0/24

Prefix 192.0.2.0/24, Max Length 24, ASN 64496

All certificates in this scenario are considered valid since the INRs of each certificate are encompassed by those of the issuing certificate. ROA1 is valid because the specified prefix is encompassed by the embedded EE certificate, as required by [RFC6482].

3. Operational Considerations

The allocations recorded in the RPKI change as a result of resource transfers. For example, the CAs involved in transfer might choose to modify CA certificates in an order that causes some of these certificates to "over-claim" temporarily. A certificate is said to "over-claim" if it includes INRs not contained in the INRs of the CA that issued the certificate in question.

It may also happen that a child CA does not voluntarily request a shrunk resource certificate when resources are being transferred or reclaimed by the parent. Furthermore operational errors that may occur during management of RPKI databases also may create CA certificates that, temporarily, no longer encompass all of the INRs of subordinate certificates.

Consider the following sequence:

Certificate 1 (trust anchor):
Issuer TA,
Subject TA,
Resources 192.0.2.0/24, 198.51.100.0/24,
2001:db8::/32, AS64496-AS64500

Certificate 2:
Issuer TA,
Subject CA1,
Resources 192.0.2.0/24, 2001:db8::/32

Certificate 3 (invalid):
Issuer CA1,
Subject CA2,
Resources 192.0.2.0/24, 198.51.100.0/24, 2001:db8::/32

ROA 1 (invalid):
Embedded Certificate 4 (EE certificate, invalid):
Issuer CA2,
Subject R1,
Resources 192.0.2.0/24

Prefix 192.0.2.0/24, Max Length 24, ASN 64496

Here Certificate 2 from the previous example was re-issued by TA to CA1 and the prefix 198.51.100.0/24 was removed. However, CA1 failed to re-issue a new Certificate 3 to CA2. As a result Certificate 3 is now over-claiming and considered invalid; by recursion the embedded Certificate 4 used for ROA1 is also invalid. And ROA1 is invalid because the specified prefix contained in the ROA is no longer encompassed by a valid embedded EE certificate, as required by [RFC6482]

However, it should be noted that ROA1 does not make use of any of the address resources that were removed from CA1’s certificate, and thus it would be desirable if ROA1 could still be viewed as valid. Technically CA1 should re-issue a Certificate 3 to CA2 without 198.51.100.0/24, and then ROA1 would be considered valid according to [RFC6482]. But as long as CA1 does not take this action, ROA1 remains invalid. It would be preferable if ROA1 could be considered valid, since the assertion it makes was not affected by the reduced scope of CA1’s certificate.

4. An Amended RPKI Certification Validation Process

4.1. Verified Resource Sets

The problem described above can be considered as a low probability problem today. However the potential impact on routing security would be high if an over-claiming occurred near the apex of the RPKI hierarchy, as this would invalidate the entirety of the sub-tree located below this point.

The changes specified here to the validation procedure in [RFC6487] do not change the probability of this problem, but they do limit the impact to just the over-claimed resources. This revised validation algorithm is intended to avoid causing CA certificates to be treated as completely invalid as a result of over-claims. However, these changes are designed to not degrade the security offered by the RPKI. Specifically, ROAs and router certificates will be treated as valid only if all of the resources contained in them are encompassed by all superior certificates along a path to a trust anchor.

The way this is achieved conceptually is by maintaining a Verified Resource Set (VRS) for each certificate that is separate from the INRs found in the [RFC3779] resource extension in the certificate.

4.2. Differences with existing standards

4.2.1. Certificate Policy (CP) for use with validation reconsidered in the Resource PKI (RPKI)

Note that section 1.2 of [RFC6484] defines the "Certificate Policy (CP) for the Resource PKI (RPKI)" with 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 }
```

This document requests an assignment of a new OID for an alternative "Certificate Policy (CP) for use with validation reconsidered in the Resource PKI (RPKI)" as follows:

```
id-cp-ipAddr-asNumber-v2 OBJECT IDENTIFIER ::= { iso(1)
        identified-organization(3) dod(6) internet(1)
        security(5) mechanisms(5) pkix(7) cp(14) TBD1 }
```

This alternative Certificate Policy is the same as the Certificate Policy described in [RFC6484], except that it is used to drive the decision in step 8 of the validation procedure described in Section 4.2.4.4.

4.2.2. An alternative to RFC3779 X.509 Extensions for IP Addresses and AS Identifiers

This document defines an alternative to [RFC3779]. All specifications and procedures described in [RFC3779] apply, with the following notable exceptions.

4.2.2.1. OID for id-pe-ipAddrBlocks-v2

This document request an OID for the extension id-pe-ipAddrBlocks-v2 (id-pe TBD2). This OID MUST only be used in conjunction with the alternative Certificate Policy OID defined in Section 4.2.1.

The following is an amended specification to be used as an alternative to the specification in section 2.2.1 of [RFC3779].

The OID for this extension is id-pe-ipAddrBlocks-v2.

```
id-pe-ipAddrBlocks-v2  OBJECT IDENTIFIER ::= { id-pe TBD2 }
```

where [RFC5280] defines:

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

id-pe    OBJECT IDENTIFIER ::= { id-pkix 1 }
```

4.2.2.2. Syntax for id-pe-ipAddrBlocks-v2

```
id-pe-ipAddrBlocks-v2      OBJECT IDENTIFIER ::= { id-pe TBD2 }

IPAddrBlocks        ::= SEQUENCE OF IPAddressFamily

IPAddressFamily     ::= SEQUENCE {    -- AFI & optional SAFI --
addressFamily        OCTET STRING (SIZE (2..3)),
ipAddressChoice      IPAddressChoice }

IPAddressChoice     ::= CHOICE {
inherit              NULL, -- inherit from issuer --
addressesOrRanges    SEQUENCE OF IPAddressOrRange }

IPAddressOrRange    ::= CHOICE {
addressPrefix        IPAddress,
addressRange         IPAddressRange }

IPAddressRange      ::= SEQUENCE {
min                  IPAddress,
max                  IPAddress }

IPAddress           ::= BIT STRING
```

Note that the descriptions of objects referenced in the syntax above are defined in sections 2.2.3.1 through 2.2.3.9 of [RFC3779].

4.2.2.3. OID for id-pe-autonomousSysIds-v2

This document request an OID for the extension id-pe-autonomousSysIds-v2 (id-pe TBD3). This OID MUST only be used in conjunction with the alternative Certificate Policy OID defined in Section 4.2.1.

The following is an amended specification to be used as an alternative to the specification in section 3.2.1 of [RFC3779].

The OID for this extension is id-pe-autonomousSysIds-v2.

```
id-pe-autonomousSysIds-v2  OBJECT IDENTIFIER ::= { id-pe TBD3 }
```

where [RFC5280] defines:

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

id-pe    OBJECT IDENTIFIER ::= { id-pkix 1 }
```

4.2.2.4. Syntax for id-pe-autonomousSysIds-v2

```
id-pe-autonomousSysIds-v2  OBJECT IDENTIFIER ::= { id-pe TBD3 }

ASIdentifiers       ::= SEQUENCE {
asnum               [0] EXPLICIT ASIdentifierChoice OPTIONAL,
rdi                 [1] EXPLICIT ASIdentifierChoice OPTIONAL}

ASIdentifierChoice  ::= CHOICE {
inherit              NULL, -- inherit from issuer --
asIdsOrRanges        SEQUENCE OF ASIdOrRange }

ASIdOrRange         ::= CHOICE {
id                  ASId,
range               ASRange }

ASRange             ::= SEQUENCE {
min                 ASId,
max                 ASId }

ASId                ::= INTEGER
```

4.2.2.5. Amended IP Address Delegation Extension Certification Path Validation

Certificate path validation is performed as specified in Section 4.2.4.4.

4.2.2.6. Amended Autonomous System Identifier Delegation Extension Certification Path Validation

Certificate path validation is performed as specified in Section 4.2.4.4.

4.2.2.7. Amended ASN.1 module

This document requests an OID for id-mod-ip-addr-and-as-ident-v2, as follows:

```
IPAddrAndASCertExtn-v2 { iso(1) identified-organization(3) dod(6)
   internet(1) security(5) mechanisms(5) pkix(7) mod(0)
   id-mod-ip-addr-and-as-ident-v2(TBD4) }
```

The following is an amended specification to be used as an alternative to the specification in section appendix A of [RFC3779].

This normative appendix describes the IP address and AS identifiers extensions used by conforming PKI components in ASN.1 syntax.

```
IPAddrAndASCertExtn-v2 { iso(1) identified-organization(3) dod(6)
   internet(1) security(5) mechanisms(5) pkix(7) mod(0)
   id-mod-ip-addr-and-as-ident-v2(TBD4) }

DEFINITIONS EXPLICIT TAGS ::=

BEGIN

-- EXPORTS ALL --

IMPORTS

-- PKIX specific OIDs and arcs --

id-pe FROM PKIX1Explicit88 { iso(1) identified-organization(3)
     dod(6) internet(1) security(5) mechanisms(5) pkix(7)
     id-mod(0) id-pkix1-explicit(18) }

-- IP Address Block and AS Identifiers Syntax --

IPAddrBlocks, ASIdentifiers FROM  IPAddrAndASCertExtn { iso(1)
   identified-organization(3) dod(6) internet(1) security(5)
   mechanisms(5) pkix(7) mod(0) id-mod-ip-addr-and-as-ident(30) }
;

-- Validation Reconsidered IP Address Delegation Extension OID --

id-pe-ipAddrBlocks-v2  OBJECT IDENTIFIER ::= { id-pe TBD2 }

-- Validation Reconsidered IP Address Delegation Extension Syntax --
-- Syntax is imported from [RFC3779] --

-- Validation Reconsidered Autonomous System Identifier --
--     Delegation Extension OID                          --

id-pe-autonomousSysIds-v2  OBJECT IDENTIFIER ::= { id-pe TBD3 }

-- Validation Reconsidered Autonomous System Identifier --
--     Delegation Extension Syntax                       --

-- Syntax is imported from [RFC3779] --

END
```

4.2.3. Addendum to RFC6268

This document requests an OID for id-mod-ip-addr-and-as-ident-2v2 as follows:

```
IPAddrAndASCertExtn-2010v2 { iso(1) identified-organization(3) dod(6)
         internet(1) security(5) mechanisms(5) pkix(7) mod(0)
         id-mod-ip-addr-and-as-ident-2v2(TBD5) }
```

[RFC6268] is an informational RFC that updates some auxiliary ASN.1 modules to conform to the 2008 version of ASN.1; the 1988 ASN.1 modules in Section 4.2.2.7 remain the normative version.

The following is an additional module confirming to the 2008 version of ASN.1 to be used with the extensions defined in Section 4.2.2.1 and Section 4.2.2.3.

```
IPAddrAndASCertExtn-2010v2 { iso(1) identified-organization(3) dod(6)
         internet(1) security(5) mechanisms(5) pkix(7) mod(0)
         id-mod-ip-addr-and-as-ident-2v2(TBD5) }

DEFINITIONS EXPLICIT TAGS ::=

BEGIN

   EXPORTS ALL;
   IMPORTS

   -- PKIX specific OIDs and arcs --

   id-pe
   FROM PKIX1Explicit-2009
     { iso(1) identified-organization(3) dod(6) internet(1)
       security(5) mechanisms(5) pkix(7) id-mod(0)
       id-mod-pkix1-explicit-02(51)}

   EXTENSION
   FROM PKIX-CommonTypes-2009
     { iso(1) identified-organization(3) dod(6) internet(1)
       security(5) mechanisms(5) pkix(7) id-mod(0)
       id-mod-pkixCommon-02(57)}

-- IP Address Block and AS Identifiers Syntax --

   IPAddrBlocks, ASIdentifiers
   FROM IPAddrAndASCertExtn-2010
      { iso(1) identified-organization(3) dod(6)
        internet(1) security(5) mechanisms(5) pkix(7) mod(0)
```

```
          id-mod-ip-addr-and-as-ident-2(72) }
     ;

     --
     -- Extensions contains the set of extensions defined in this
     -- module
     --
     -- These are intended to be placed in public key certificates
     -- and thus should be added to the CertExtensions extension
     -- set in PKIXImplicit-2009 defined for [RFC5280]
     --

     Extensions EXTENSION ::= {
        ext-pe-ipAddrBlocks-v2 | ext-pe-autonomousSysIds-v2
     }

     -- Validation Reconsidered IP Address Delegation Extension OID --

     ext-pe-ipAddrBlocks-v2 EXTENSION ::= {
       SYNTAX IPAddrBlocks
       IDENTIFIED BY id-pe-ipAddrBlocks-v2
     }

     id-pe-ipAddrBlocks-v2  OBJECT IDENTIFIER ::= { id-pe TBD2 }

     -- Validation Reconsidered IP Address Delegation --
     --      Extension Syntax                         --

     -- Syntax is imported from [RFC6268] --

     -- Validation Reconsidered Autonomous System Identifier --
     --      Delegation Extension OID                        --

     ext-pe-autonomousSysIds-v2 EXTENSION ::= {
       SYNTAX ASIdentifiers
       IDENTIFIED BY id-pe-autonomousSysIds-v2
     }

     id-pe-autonomousSysIds OBJECT IDENTIFIER ::= { id-pe TBD3 }

  -- Validation Reconsidered Autonomous System Identifier --
  --      Delegation Extension Syntax                     --

  -- Syntax is imported from [RFC6268] --

  END
```

4.2.4. An alternative to RFC6487 Profile for X.509 PKIX Resource Certificates

This document defines an alternative Profile for X.509 PKIX Resource Certificates. This profile follows all definitions and procedures described in [RFC6487] with the following notable exceptions.

4.2.4.1. Amended Certificate Policies

The following is an amended specification to be used in this profile, in place of section 4.8.9 of [RFC6487].

This extension MUST be present and MUST be marked critical. It MUST include exactly one policy of type id-cp-ipAddr-asNumber-v2, as specified in the updated RPKI CP in Section 4.2.1.

4.2.4.2. Amended IP Resources

The following is an amended specification to be used in this profile, in place of section 4.8.10 of [RFC6487].

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 Section 4.2.2.1. The value may specify the "inherit" element for a particular Address Family Identifier (AFI) value. In the context of resource certificates describing public number resources for use in the public Internet, the Subsequent AFI (SAFI) value MUST NOT be used.

This extension MUST either specify a non-empty set of 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.

4.2.4.3. Amended AS Resources

The following is an amended specification to be used in this profile, in place of section 4.8.11 of [RFC6487].

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 Section 4.2.2.3, or it may specify the "inherit" element. Routing

Domain Identifier (RDI) values are NOT supported in this profile and MUST NOT be used.

This extension MUST either specify a non-empty set of 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.

4.2.4.4. Amended Resource Certificate Path Validation

The following is an amended specification for path validation to be used in place of section 7.2 of [RFC6487] allowing for the validation of both certificates following the profile defined in [RFC6487], as well as certificates following the profile described above.

The following algorithm is employed to validate CA and EE resources certificates. It is modelled on the path validation algorithm from [RFC5280], but modified to make use of the IP Address Delegation and AS Identifier Delegation Extensions from [RFC3779].

There are two inputs to the validation algorithm:

1. a trust anchor

2. a certificate to be validated

The algorithm is initialized with two new variables for use in the RPKI: Validated Resource Set-IP (VRS-IP) and Validated Resource Set-AS (VRS-AS). These sets are used to track the set of INRs (IP address space and AS Numbers) that are considered valid for each CA certificate. The VRS-IP and VRS-AS sets are initially set to the IP Address Delegation and AS Identifier Delegation values, respectively, from the trust anchor used to perform validation.

This path validation algorithm verifies, among other things, that a prospective certification path (a sequence of n certificates) satisfies the following conditions:

a. for all 'x' in {1, ..., n-1}, the subject of certificate 'x' is the issuer of certificate ('x' + 1);

b. certificate '1' is issued by a trust anchor;

c. certificate 'n' is the certificate to be validated; and

d. for all 'x' in {1, ..., n}, certificate 'x' is valid.

Certificate validation requires verifying that all of the following conditions hold, in addition to the certification path validation criteria specified in Section 6 of [RFC5280].

1. The signature of certificate x (x>1) is verified using the public key of the issuer's certificate (x-1), using the signature algorithm specified for that public key (in certificate x-1).

2. The current time lies within the interval defined by the NotBefore and NotAfter values in the Validity field of certificate x.

3. The Version, Issuer, and Subject fields of certificate x satisfy the constraints established in Section 4.1-4.7 of this specification.

4. If certificate x uses the Certificate Policy defined in section 4.8.9 of [RFC6487], then the certificate MUST contain all extensions defined in section 4.8 of [RFC6487] that must be present. The value(s) for each of these extensions MUST satisfy the constraints established for each extension in the respective sections. Any extension not thus identified MUST NOT appear in certificate x.

5. If certificate x uses the Certificate Policy defined in Section 4.2.4.1, then all extensions defined in section 4.8 of [RFC6487], except sections 4.8.9, 4.8.10 and 4.8.10 MUST be present. The certificate MUST contain an extension as defined in Section 4.2.4.2 or Section 4.2.4.3, or both. The value(s) for each of these extensions MUST satisfy the constraints established for each extension in the respective sections. Any extension not thus identified MUST NOT appear in certificate x.

6. Certificate x MUST NOT have been revoked, i.e., it MUST NOT appear on a CRL issued by the CA represented by certificate x-1

7. Compute the VRS-IP and VRS-AS set values as indicated below:

   * If the IP Address Delegation extension is present in certificate x and x=1, set the VRS-IP to the resources found in this extension.

   * If the IP Address Delegation extension is present in certificate x and x>1, set the VRS-IP to the intersection of the resources between this extension and the value of the VRS-IP computed for certificate x-1.

* If the IP Address Delegation extension is absent in certificate x, set the VRS-IP to NULL.

* If the IP Address Delegation extension is present in certificate x and x=1, set the VRS-IP to the resources found in this extension.

* If the AS Identifier Delegation extension is present in certificate x and x>1, set the VRS-AS to the intersection of the resources between this extension and the value of the VRS-AS computed for certificate x-1

* If the AS Identifier Delegation extension is absent in certificate x, set the VRS-AS to NULL.

8. If there is any difference in resources in the VRS-IP and the IP Address Delegation extension on certificate x, or the VRS-AS and the AS Identifier Delegation extension on certificate x, then:

   * If certificate x uses the Certificate Policy defined in Section 4.2.4.1 a warning listing the over-claiming resources for certificate x SHOULD be issued.

   * If certificate x uses the Certificate Policy defined in section 4.8.9 of [RFC6487], then certificate x MUST be rejected.

These rules allow a CA certificate to contain resources that are not present in (all of) the certificates along the path from the trust anchor to the CA certificate. If none of the resources in the CA certificate are present in all certificates along the path, no subordinate certificates could be valid. However, the certificate is not immediately rejected as this may be a transient condition. Not immediately rejecting the certificate does not result in a security problem because the associated VRS sets accurately reflect the resources validly associated with the certificate in question.

4.2.5. An alternative ROA validation RFC6482

Section 4 of [RFC6482] currently has the following text on the validation of resources on a ROA:

o The IP address delegation extension [RFC3779] is present in the end-entity (EE) certificate (contained within the ROA), and each IP address prefix(es) in the ROA is contained within the set of IP addresses specified by the EE certificate’s IP address delegation extension.

If the end-entity certificate uses the Certificate Policy defined in Section 4.2.4.1, then the following approach must be used instead.

- The amended IP address delegation extension described in Section 4.2.4.2 is present in the end-entity (EE) certificate (contained within the ROA), and each IP address prefix(es) in the ROA is contained within the VRS-IP set that is specified as an outcome of EE certificate validation described in Section 4.2.4.4.

Note that this ensures that ROAs can be valid only, if all IP address prefixes in the ROA are encompassed by the VRS-IP of all certificates along the path to the trust anchor used to verify it.

Operators MAY issue separate ROAs for each IP address prefix, so that the loss of one or more IP address prefixes from the VRS-IP of any certificate along the path to the trust anchor would not invalidate authorizations for other IP address prefixes.

4.2.6. An alternative to BGPSec Router Certificate Validation

If a BGPsec Router Certificate ([I-D.ietf-sidr-bgpsec-pki-profiles]) uses the Certificate Policy defined in Section 4.2.4.1, then in addition to the BGPsec Router Certificate Validation defined in section 3.3 of [I-D.ietf-sidr-bgpsec-pki-profiles], the following constraint MUST be met:

- The VRS-AS of BGPsec Router Certificates MUST encompass all ASNs in the AS Resource Identifier Delegation extension.

Operators MAY issue separate BGPsec Router Certificates for different ASNs, so that the loss of on ASN from the VRS-AS of any certificate along the path to the trust anchor would not invalidate router keys for other ASNs.

5. Validation examples

In this section we will demonstrate the outcome of RPKI validation performed using the algorithm and procedures described in Section 4.2.4.4, Section 4.2.5 and Section 4.2.6, under three deployment scenarios:

- An RPKI tree consisting of certificates using the old OIDs only
- An RPKI tree consisting of certificates using the new OIDs only
- An RPKI tree consisting of a mix of certificates using either the old or the new OIDs

In this context we refer to a certificate as using the 'old' OIDs, if the certificate uses a combination of the OIDs defined in section 4.8.9 of [RFC6487], section 2.2.1 of [RFC3779] and/or section 3.2.1 od [RFC3779]. We refer to a certificate as using the 'new' OIDS, if the certificate uses a combination of OIDs defined in Section 4.2.4.1, Section 4.2.2.1 and/or Section 4.2.2.3.

5.1. Example 1 - An RPKI tree using the old OIDs only

Consider the following example:

```
  Certificate 1 (trust anchor):
   Issuer: TA,
   Subject: TA,
   OIDs: OLD,
   Resources: 0/0, ::0, AS0-4294967295 (ALL Resources)

    Verified Resource Set: 0/0, ::0, AS0-4294967295 (ALL Resources)
    Warnings: none

  Certificate 2:
   Issuer: TA,
   Subject: CA1,
   OIDs: OLD,
   Resources: 192.0.2.0/24, 2001:db8::/32, AS64496

    Verified Resource Set: 192.0.2.0/24,
                           2001:db8::/32, AS64496
    Warnings: none

  Certificate 3 (invalid):
   Issuer: CA1,
   Subject: CA2,
   OIDs: OLD,
   Resources: 192.0.2.0/24, 198.51.100.0/24, AS64496

    Verified Resource Set: 192.0.2.0/24, AS64496

    Certificate 3 is considered invalid because "Resources:"
    contains 198.51.100.0/24 which is not found in the
    Verified Resource Set.

  ROA 1 (invalid):
   Embedded Certificate 4 (EE certificate invalid):
    Issuer: CA2,
    Subject: R1,
    OIDs: OLD,
    Resources: 192.0.2.0/24
```

```
         Prefix 192.0.2.0/24, Max Length 24, ASN 64496

       ROA1 is considered invalid because Certificate 3 is invalid.

     ROA 2 (invalid):
      Embedded Certificate 5 (EE certificate invalid):
       Issuer: CA2,
       Subject: R2,
       OIDs: OLD,
       Resources: 198.51.100.0/24
        Prefix 198.51.100.0/24, Max Length 24, ASN 64496

       ROA2 is considered invalid because Certificate 3 is invalid.

     BGPSec Certificate 1 (invalid):
      Issuer: CA2,
      Subject: ROUTER-64496,
      OIDs: NEW,
      Resources: AS64496

       BGPSec Certificate 1 is invalid because Certificate 3 is invalid.

     BGPSec Certificate 2 (invalid):
      Issuer: CA2,
      Subject: ALL-ROUTERS,
      OIDs: NEW,
      Resources: AS64496-AS64497

       BGPSec Certificate 2 is invalid because Certificate 3 is invalid.
```

5.2. Example 2 - An RPKI tree using the new OIDs only

Consider the following example under the amended approach:

```
     Certificate 1 (trust anchor):
      Issuer: TA,
      Subject: TA,
      OIDs: NEW,
      Resources: 0/0, ::0, AS0-4294967295 (ALL Resources)

       Verified Resource Set: 0/0, ::0, AS0-4294967295 (ALL Resources)
       Warnings: none

     Certificate 2:
      Issuer: TA,
      Subject: CA1,
      OIDs: NEW,
      Resources: 192.0.2.0/24, 2001:db8::/32, AS64496
```

```
       Verified Resource Set: 192.0.2.0/24,
                              2001:db8::/32, AS64496
       Warnings: none

     Certificate 3:
      Issuer: CA1,
      Subject: CA2,
      OIDs: NEW,
      Resources: 192.0.2.0/24, 198.51.100.0/24, AS64496

       Verified Resource Set: 192.0.2.0/24, AS64496
       Warnings: over-claim for 198.51.100.0/24

     ROA 1 (valid):
      Embedded Certificate 4 (EE certificate):
       Issuer: CA2,
       Subject: R1,
       OIDs: NEW,
       Resources: 192.0.2.0/24
        Prefix 192.0.2.0/24, Max Length 24, ASN 64496

        Verified Resource Set: 192.0.2.0/24
        Warnings: none

       ROA1 is considered valid because the prefix matches the Verified
       Resource Set on the embedded EE certificate.

     ROA 2 (invalid):
      Embedded Certificate 5 (EE certificate invalid):
       Issuer: CA2,
       Subject: R2,
       OIDs: NEW,
       Resources: 198.51.100.0/24
         Prefix 198.51.100.0/24, Max Length 24, ASN 64496

         Verified Resource Set: none (empty set)
         Warnings: 198.51.100.0/24

       ROA2 is considered invalid because the ROA prefix 198.51.100.0/24
       is not contained in the Verified Resource Set.

     BGPSec Certificate 1 (valid):
      Issuer: CA2,
      Subject: ROUTER-64496,
      OIDs: NEW,
      Resources: AS64496

       Verified Resource Set: AS64496
```

Warnings: none

BGPSec Certificate 2 (invalid):
Issuer: CA2,
Subject: ALL-ROUTERS,
OIDs: NEW,
Resources: AS64496-AS64497

Verified Resource Set: AS64496

BGPSec Certificate 2 is invalid because not all of its Resources are contained in the Verified Resource Set.

Note that this problem can be mitigated by issuing separate certificates for each AS number.

5.3. Example 3 - An RPKI tree using a mix of old and new OIDs

In the following example new OIDs are used only for CA certificates where the issuing CA anticipates that an over-claim could occur, and has a desire to limit the impact of this to just the over-claimed resources in question:

Certificate 1 (trust anchor):
Issuer: TA,
Subject: TA,
OIDs: OLD,
Resources: 0/0, ::0, AS0-4294967295 (ALL Resources)

Verified Resource Set: 0/0, ::0, AS0-4294967295 (ALL Resources)
Warnings: none

Note that a Trust Anchor certificate cannot be found to over-claim. So, using the new OIDs here would not change anything with regards to the validity of this certificate.

Certificate 2:
Issuer: TA,
Subject: CA1,
OIDs: OLD,
Resources: 192.0.2.0/24, 2001:db8::/32, AS64496

Verified Resource Set: 192.0.2.0/24,
2001:db8::/32, AS64496
Warnings: none

Note that since the TA certificate claims all resources, it is impossible to issue a certificate below it that could be found

to be over-claiming. Therefore there is no benefit in using the new OIDs for Certificate 2.

```
  Certificate 3:
   Issuer: CA1,
   Subject: CA2,
   OIDs: NEW,
   Resources: 192.0.2.0/24, 198.51.100.0/24, AS64496

    Verified Resource Set: 192.0.2.0/24, AS64496
    Warnings: over-claim for 198.51.100.0/24
```

Note that CA1 anticipated that it might invalid Certificate 3 issued to CA2, if its own resources on Certificate 2 were modified and OLD OIDs would have been used on Certificate 3.

```
  ROA 1 (valid):
   Embedded Certificate 4 (EE certificate):
    Issuer: CA2,
    Subject: R1,
    OIDs: OLD,
    Resources: 192.0.2.0/24
     Prefix 192.0.2.0/24, Max Length 24, ASN 64496

     Verified Resource Set: 192.0.2.0/24
     Warnings: none
```

ROA1 is considered valid because the prefix matches the Verified Resource Set on the embedded EE certificate.

```
  ROA 2 (invalid):
   Embedded Certificate 5 (EE certificate invalid):
    Issuer: CA2,
    Subject: R2,
    OIDs: OLD,
    Resources: 198.51.100.0/24
      Prefix 198.51.100.0/24, Max Length 24, ASN 64496

      Verified Resource Set: none (empty set)
```

ROA2 is considered invalid because "Resources:" on its EE certificate contains 198.51.100.0/24, which is not contained in its Verified Resource Set.

Note that if new OIDs were used here (as in example 2) ROA 2 would be considered invalid because the Prefix is not contained in the Verified Resource Set.

So, if there is no difference in the validity outcome one could argue that using old OIDs here is clearest, because any over-claim of ROA prefixes MUST result in it being considered invalid (as described in section 4.2.5).

BGPSec Certificate 1 (valid):
Issuer: CA2,
Subject: ROUTER-64496,
OIDs: OLD,
Resources: AS64496

Verified Resource Set: AS64496
Warnings: none

BGPSec Certificate 2 (invalid):
Issuer: CA2,
Subject: ALL-ROUTERS,
OIDs: OLD,
Resources: AS64496-AS64497

Verified Resource Set: AS64496

BGPSec Certificate 2 is consider invalid because "Resources:" contains AS64497, which is not contained in its Verified Resource Set.

Note that if new OIDs were used here (as in example 2) BGPSec Certificate 2 would be considered invalid because the Prefix is not contained in the Verified Resource Set.

So, if there is no difference in the validity outcome one could argue that using old OIDs here is the clearest, because any over-claim on this certificate MUST result in it being considered invalid (as described in section 4.2.6).

Also note that as in example 2 this problem can be mitigated by issuing separate certificates for each AS number.

6. Deployment Considerations

This document defines an alternative RPKI validation algorithm, but it does not dictate how this algorithm will be deployed. This should be discussed as a separate effort. That said, the following observations may help this discussion.

Because this document introduces new OIDs and an alternative to the Profile for X.509 PKIX Resource Certificates described in [RFC6487], the use of such certificates in the global RPKI will lead to the

rejection of such certificates by Relying Party tools that do not (yet) implement the alternative profile described in this document.

For this reason it is important that such tools are updated before Certificate Authorities start to use this specification.

However, because the OIDs are defined in each RPKI certificate, there is no strict requirement for all Certificate Authorities to migrate to the new OIDs at the same time, or even for all the certificates they issue. The example in Section 5.3 illustrates a possible deployment where the new OIDs are used only when issuing CA certificates where an accidental over-claim may occur.

7. Security Considerations

The authors believe that the revised validation algorithm introduces no new security vulnerabilities into the RPKI, because it cannot lead to any ROA and/or Router Certificates to be accepted if they contain resources that are not held by the issuer.

8. IANA Considerations

IANA is to add the following to the SMI Security for PKIX Certificate Policies registry:

Decimal	Description	References
TBD1	id-cp-ipAddr-asNumber-v2	[section 4.2.1]

IANA is to add the following to the SMI Security for PKIX Certificate Extension registry:

Decimal	Description	References
TBD2	id-pe-ipAddrBlocks-v2	[section 4.2.2.1]
TBD3	id-pe-autonomousSysIds-v2	[section 4.2.2.3]

IANA is to add the following to the SMI Security for PKIX Module Identifier registry:

Decimal	Description	References
TBD4	id-mod-ip-addr-and-as-ident-v2	[section 4.2.2.7]
TBD5	id-mod-ip-addr-and-as-ident-2v2	[section 4.2.3]

9. Acknowledgements

The authors would like to thank Stephen Kent for reviewing and contributing to this document. We would like to thank Rob Austein for suggesting that separate OIDs should be used to make the behaviour of Relying Party tools deterministic, and we would like to thank Russ Hously, Sean Turner and Tom Petch for their contributions on OID and ASN.1 updates. Finally we would like to thank Tom Harrison for a general review of this document.

10. References

10.1. Normative References

[I-D.ietf-sidr-bgpsec-pki-profiles]
Reynolds, M., Turner, S., and S. Kent, "A Profile for BGPsec Router Certificates, Certificate Revocation Lists, and Certification Requests", draft-ietf-sidr-bgpsec-pki-profiles-21 (work in progress), January 2017.

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

[RFC3779] Lynn, C., Kent, S., and K. Seo, "X.509 Extensions for IP Addresses and AS Identifiers", RFC 3779, DOI 10.17487/RFC3779, June 2004, <https://www.rfc-editor.org/info/rfc3779>.

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

[RFC6482] Lepinski, M., Kent, S., and D. Kong, "A Profile for Route Origin Authorizations (ROAs)", RFC 6482, DOI 10.17487/RFC6482, February 2012, <https://www.rfc-editor.org/info/rfc6482>.

[RFC6484] Kent, S., Kong, D., Seo, K., and R. Watro, "Certificate Policy (CP) for the Resource Public Key Infrastructure (RPKI)", BCP 173, RFC 6484, DOI 10.17487/RFC6484, February 2012, <https://www.rfc-editor.org/info/rfc6484>.

[RFC6487] Huston, G., Michaelson, G., and R. Loomans, "A Profile for X.509 PKIX Resource Certificates", RFC 6487, DOI 10.17487/RFC6487, February 2012, <https://www.rfc-editor.org/info/rfc6487>.

10.2. Informative References

[RFC6268] Schaad, J. and S. Turner, "Additional New ASN.1 Modules for the Cryptographic Message Syntax (CMS) and the Public Key Infrastructure Using X.509 (PKIX)", RFC 6268, DOI 10.17487/RFC6268, July 2011, <https://www.rfc-editor.org/info/rfc6268>.

Authors' Addresses

Geoff Huston
Asia Pacific Network Information Centre
6 Cordelia St
South Brisbane, QLD 4101
Australia

Phone: +61 7 3858 3100
Email: gih@apnic.net

George Michaelson
Asia Pacific Network Information Centre
6 Cordelia St
South Brisbane, QLD 4101
Australia

Phone: +61 7 3858 3100
Email: ggm@apnic.net

Carlos M. Martinez
Latin American and Caribbean IP Address Regional Registry
Rambla Mexico 6125
Montevideo 11400
Uruguay

Phone: +598 2604 2222
Email: carlos@lacnic.net

Tim Bruijnzeels
RIPE Network Coordination Centre
Singel 258
Amsterdam 1016 AB
The Netherlands

Email: tim@ripe.net

Andrew Lee Newton
American Registry for Internet Numbers
3635 Concorde Parkway
Chantilly, VA 20151
USA

Email: andy@arin.net

Daniel Shaw
African Network Information Centre (AFRINIC)
11th Floor, Standard Chartered Tower
Cybercity, Ebene
Mauritius

Phone: +230 403 51 00
Email: daniel@afrinic.net

Secure Inter-Domain Routing
Internet-Draft
Intended status: Informational
Expires: January 9, 2017

X. Lee
X. Liu
Z. Yan
G. Geng
Y. Fu
CNNIC

July 8, 2016

RPKI Deployment Considerations: Problem Analysis and Alternative Solutions
draft-lee-sidr-rpki-deployment-02

Abstract

With the global deployment of RPKI, a lot of concerns about technical problems have been and will be raised. In this draft, we collect and analyze the problems that have appeared or that seem likely to appear during the process of RPKI deployment, and suggest some solutions to address or mitigate these problems.

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 January 9, 2017.

Table of Contents

1. Introduction

1.1. RPKI Architecture

In RPKI, CAs (Certification Authorities) are organized in a hierarchical structure which is aligned to the existing INR (Internet Number Resources) allocation hierarchy (including IP prefixes and AS numbers). Each INR allocation requires corresponding resource certificates to attest to it, for security. In RPKI, two types of resource certificates [RFC6480] are generated as adjuncts to this allocation process: CA certificates and EE (End-entity) certificates. CA certificates attest to the INR holdings; EE certificates are primarily used for ROAs (Route Origin Authorizations) [RFC6482] and Router Certificates. ROAs are used to bind IP prefixes to the ASes

that is permitted to originate routes for these IP prefixes. Manifests [RFC6486] are also validated using EE certificates. Manifests are used to ensure the integrity of the RPKI repository system.

The process of using the RPKI to verify the origin of a route is as follows.

1. CAs, including IANA (Internet Assigned Numbers Authority), five RIRs (Regional Internet Registries), NIRs (National Internet Registries) and ISPs (Internet Service Providers), publish authoritative objects (including resource certificates, ROAs, Manifest and so on) into their repositories.

2. RPs (Relying Parties) all over the world collect (using rsync or RRDP protocol [I-D.ietf-sidr-delta-protocol]) and verify (using rcynic or RPSTIR) the RPKI objects from these repositories, and provide the results of verification to BGP border routers or other routing practices such as RPSL-based.

3. Finally, BGP border routers can make use of these results to verify the route origin information in the BGP update messages they receive. This may be done by generating route filters from the validated RPKI data, or by using the RPKI-to-router protocol [RFC6810].

1.2. Status of RPKI Deployment

Each of the five RIRs has initiated the deployment of RPKI, and each now offers RPKI services to its members. A number of countries (Ecuador, Japan, Bangladesh, etc.) have also started to test and deploy RPKI internally. In order to promote the deployment of RPKI, ICANN (Internet Corporation for Assigned Names and Numbers), the five RIRs, many NIRs and companies have making continuous efforts to solve the existing problems and improve the corresponding policies and technical standards.

However, RPKI is still in its early stages of global deployment. According to the data provided by RPKI Dashboard as of July 2016, the current routing table holds about 659,271 IP prefixes in total, and the RPKI validation state has been determined for 44983 IP prefixes, which means that only 6.82% of the prefixes in the routing table can be validated using the RPKI. Table 1 details of the RPKI "adoption rate" (the percentage of members deployed RPKI) in each of the RIRs.

RIR	AFRINIC	APNIC	ARIN	LACNIC	RIPE NCC
Adoption Rate	0%	3.44%	1.22%	20.66%	12.14%

Table 1.Adoption rate of RPKI in 5 RIRs

As we can see from Table 1, LACNIC has the highest adoption rate, which is about 20.66%. While the adoption rates in ARIN and AFRINIC are much lower, which are only 1.22% and 0% respectively.

RIPE NCC provides some statistics regarding the number of resource certificates and ROAs in each RIR. From these statistics we find a good sign that the global deployment status of RPKI rises gradually, and with its further evolution, the global adoption rate of RPKI should achieve a faster growth.

2. 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].

3. Considerations of RPKI Deployment

During the process of incremental deployment of RPKI, several technical problems have appeared and others may appear. In this section, we attempt to collect and analyze the problems that seem most critical.

3.1. More than One TA

A TA (Trust Anchor) is an authoritative entity represented by a public key and its associated data [RFC5914]. The public key is used to verify digital signatures and the associated data describes the types of information and actions for which the TA is authoritative. There are multiple TAs in the RPKI architecture today, for example, the five RIRs are generally viewed as default TAs.

With more than one TA, there is no technical mechanism to prevent two or more TAs from asserting control over the same set of INRs accidentally or maliciously, which means that certificates might be issued for allocations of the overlapping INRs. This, in turn, may lead to inconsistent and conflicting assertions about to whom the specific INRs have been allocated. This kind of problem obviously may cause resource conflicts on the Internet.

3.2. Problems of CAs

3.2.1. Operational Errors

Operatioanl errors by CAs are inevitable and may cause significant impact on Internet routing. Thus such errors by CAs in RPKI constitute a risk to widespread deployment.

Operatioanl errors by CAs in the RPKI may lead to serious consequences similar to those caused by malicious attacks (black-hole routes, traffic interception, and denial-of-service attacks). For example, an error in using a ROA (such as adding a new erroneous ROA or whacking an existing ROA) may cause all routes covered by the (original) ROA to become invalid (or to assume an "unknown" security status [RFC6483]). Note that, if the old validating ROA still matches (not just covers) the announce prefix, the announcement will still be marked as valid.

3.2.2. Unilateral Resource Revocation

In the RPKI architecture, there is a risk that CAs have the power to unilaterally revoke the INRs that have been allocated to their descendants, e.g., by revoking corresponding CA certificates [RFC6480].

This is a natural aspect of PKIs and it is a necessary capability for CAs as they manage re-allocation of resources within their domains. However, if revocation occurs accidentally, or because the CA has been compelled by authorities, the results can be significant. Specifically, all RPs will view the origin assertions by the CA (and its descendants) to be not found. This may cause ISPs to depreference routes to the affected prefixes.

3.3. Mirror World Attacks

In mirror world attacks, a malicious CA presents one view of the RPKI repository (that it manages) to some RPs, and a different view to others. (Because repository data may be cached by ISPs, it may not be possible for a malicious CA to provide erroneous results to a narrowly targeted set of RPs.)

Since a CA in the RPKI controls everything in its own repository, it may be easy for a malicious CA to perform such a attack. For example, a malicious CA presents the correct view of its repository to some RPs, but a forged view (e.g., the CA adds a specific, erroneous ROA) to the others. When these deceived RPs offer their validation results to BGP routers, the routers may abandon the

legitimate routes that are considered to be invalid according to the (erroneous) validation results they have received.

3.4. Data Synchronization

It is required in [RFC6480] that all repositories must be accessible via rsync protocol which is used by RPs to get the RPKI objects in the global distributed repositories. However, the rsync protocol is considered to be controversial with respect to the following disadvantages:

1. Lack of standards and non-modular implementation: Although rsync is widely adopted in backup, restore, and file transfer, it has not been standardized by IETF. And the rsync implementation is non-modular, making it difficult to use its source code.

2. Underlying overhead caused by repository updates during active data transmissions: During data transmissions between RPs and the repository, a new update to the repository may cause data inconsistency between them. In order to rectify this inconsistency, extra overhead costs (such as performing the synchronization once more) are required.

3. This is being solved by the new RRDP protocol, now in deployment.

3.5. Problems of Staged and Incomplete Deployment

Since the global deployment of RPKI is an incremental and staged process, unexpected problems may appear during this process. Let’s take an example to explain why the incomplete deployment of RPKI may cause legitimate routes to be misclassified into invalid. In Fig.1, we make the following assumptions:

1. CNNIC, ISP1 and ISP2 have deployed the RPKI, but ISP3 has not yet. ISP1 and ISP2 received allocations form CNNIC, and ISP3 received its allocation from ISP1.

2. CNNIC allocated IP prefix 218.241.104.0/22 to ISP1 and 218.241.108.0/22 to ISP2.

3. Three ROAs (ROA1, ROA2, ROA3) are issued respectively by CNNIC, ISP1 and ISP2.

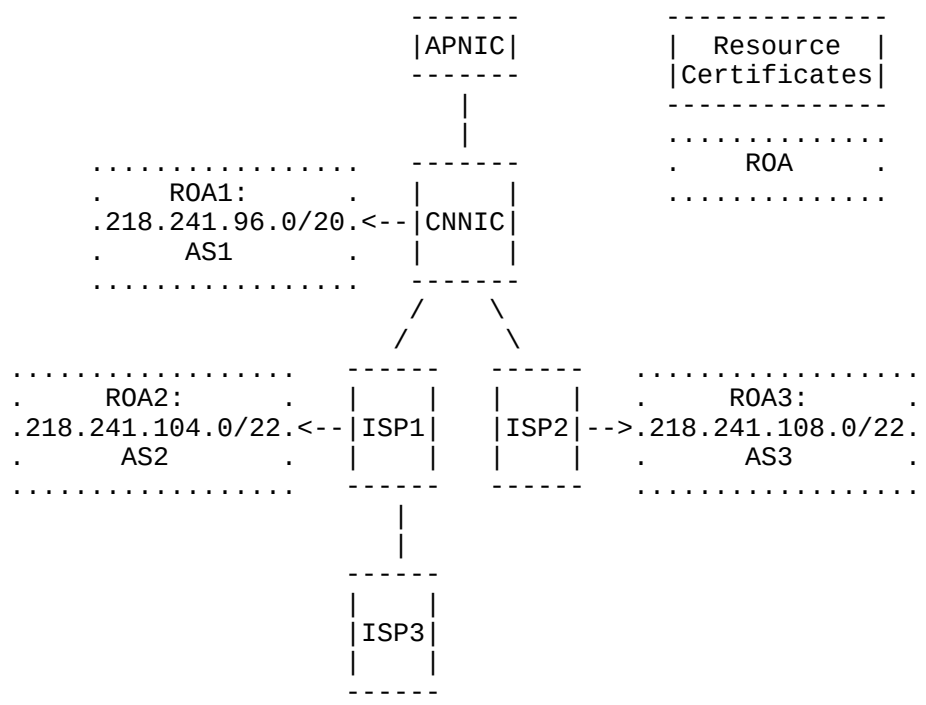

Fig.1: An example of incomplete deployment

Now ISP3 announces to be the origin of 218.241.106.0/23. When other entities receive this announcement, they can validate it with ROAs information. Since prefix 218.241.104.0/22 described in ROA2 encompasses prefix 218.241.106.0/23 and no matching ROA describes 218.241.106.0/23 could be found [RFC6483], the announcement for prefix 218.241.106.0/23 will be considered to be invalid. This example illustrates why careful coordination is needed when (non-leaf) ISPs incrementally deploy the RPKI. This example illustrates why careful coordination is needed when (non-leaf) ISPs incrementally deploy the RPKI.

Therefore, if an ISP knows its customer is not creating a ROA, it is the ISP’s responsibility to create that ROA, just as it is that ISP’s responsibility to do 42 other things for their ’customer’.

3.6. Low Validation Coverage

The route origin validation coverage refers to the percentage of valid routes attested to by the RPKI. i.e., Coverage =

number_of_valid_routes / (number_of_valid_routes + number_of_invalid_routes).

As we can see from Table 2, the coverage of route origin validation in the five RIRs differs a lot. LACNIC and RIPE NCC have the highest validation coverage and both of them are over 90%, while the coverage in APNIC is less than 70%. Many reasons may account for the low validation coverage, such as misconfigurations, low RPKI adoption rates, etc.

RIR	Total	Valid	Invalid	Unknown	Accuracy	Adoption Rate
AFRINIC	14948	242	5	14701	97.98%	1.65%
APNIC	158020	3332	1564	153124	68.06%	3.1%
ARIN	219779	1911	337	217531	85.01%	1.02%
LACNIC	76841	13379	736	62726	94.79%	18.37%
RIPE NCC	159256	16771	1307	141178	92.77%	11.35%

Table 2. Route Origin Validation Accuracy in 5 RIRs

4. Alternative Solutions to RPKI Deployment Problems

In this section, we propose and analyze the alternative solutions and strategies to solve or mitigate the problems mentioned in Section 3.

4.1. Solutions to Multiple TAs

The RIRs say they are trying to continually evolve RPKI. ICANN (IANA) and RIRs have developed a technical testbed with an RPKI GTA. It’s assumed that there must be a single root trust anchor eventually. With this single root trust anchor deployed, the risks of resource conflicts (at the level of RIR certificates) could be significantly reduced.

However, this solution cedes more power to ICANN (IANA) and thus might exacerbate the risk of "Unilateral Resource Revocation" (power imbalance) mentioned in Section 3.2.2.

4.2. Solutions to Misbehaving CAs

S. Kent et al. put forward a collection of mechanisms named "Suspenders". "Suspenders" is designed to address the adverse effects on INR holders which were caused by CAs' accidental or deliberate misbehavior or attacks on CAs and repositories. This mechanism imports two new objects: an INRD (Internet Number Resource Declaration) file and a LOCK object. The INRD file is external to the RPKI repository, and it contains the most recent changes that were made by the INR holder. The LOCK object is published in the INR holder's repository. It contains a URL which points to the INRD file, and a public key used to verify the signature of INRD file. Whenever the RPs detect the inconsistencies between the actual changes and the INRD file, they can determine individually whether to accept these changes or not. (This proposal is being revised to address operational concerns, but it is anticipated that a subsequent version of Suspenders will preserve the primary features noted above.)

4.3. Solutions to Data Synchronization

A number of alternative protocols have been presented to take the place of "rsync" protocol due to its shortcomings mentioned above.

1) RRDP

T. Bruijnzeels et al. have proposed an alternative protocol (RRDP, RPKI Repository Delta Protocol) for RPs to keep their local caches in sync with the repository system [I-D.ietf-sidr-delta-protocol]. This new protocol is based on notification, snapshot and delta files. When RPs query a repository for updates, they will use delta files (and snapshot files as needed) to keep their local caches updated. Moreover, RRDP protocol can work with the existing rsync URIs.

Compared with rsync protocol, RRDP is considered to be effective as a way to eliminate a number of consistency related issues, help to reduce the load on publication servers, and have improved scalability.

RRDP is in current RIPE and DRL software.

2) Improved Rsync Protocol

CNNIC also proposed an improved rsync mechanism which transfers the work of checksums calculation to RPs in order to reduce the computation load on the rsync server side. The mechanism also offered a NOTIFY method that send NOTIFY message to make some important RPs to actively fetch the updated RPKI objects in time.

4.4. Solutions to Incomplete Deployment and Low Validation Coverage

Both of the two problems (incomplete deployment and low validation accuracy) are caused by the partial deployment of RPKI. With the widely deployment of RPKI in the near future, these two problems ought to be mitigated.

5. Security Considerations

TBD

6. IANA Considerations

This draft does not request any IANA action.

7. Acknowledgements

The authors would like to thanks the valuable comments made by Stephen Kent and other members of sidr WG.

This document was produced using the xml2rfc tool [RFC2629].

8. References

8.1. Normative References

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

[RFC6480] Lepinski, M. and S. Kent, "An Infrastructure to Support Secure Internet Routing", RFC 6480, DOI 10.17487/RFC6480, February 2012, <http://www.rfc-editor.org/info/rfc6480>.

[RFC6482] Lepinski, M., Kent, S., and D. Kong, "A Profile for Route Origin Authorizations (ROAs)", RFC 6482, DOI 10.17487/RFC6482, February 2012, <http://www.rfc-editor.org/info/rfc6482>.

[RFC6483] Huston, G. and G. Michaelson, "Validation of Route Origination Using the Resource Certificate Public Key Infrastructure (PKI) and Route Origin Authorizations (ROAs)", RFC 6483, DOI 10.17487/RFC6483, February 2012, <http://www.rfc-editor.org/info/rfc6483>.

[RFC6486] Austein, R., Huston, G., Kent, S., and M. Lepinski, "Manifests for the Resource Public Key Infrastructure (RPKI)", RFC 6486, DOI 10.17487/RFC6486, February 2012, <http://www.rfc-editor.org/info/rfc6486>.

8.2. Informative References

[I-D.ietf-sidr-delta-protocol] Bruijnzeels, T., Muravskiy, O., Weber, B., and R. Austein, "RPKI Repository Delta Protocol", draft-ietf-sidr-delta-protocol-03 (work in progress), July 2016.

[RFC2629] Rose, M., "Writing I-Ds and RFCs using XML", RFC 2629, DOI 10.17487/RFC2629, June 1999, <http://www.rfc-editor.org/info/rfc2629>.

[RFC5914] Housley, R., Ashmore, S., and C. Wallace, "Trust Anchor Format", RFC 5914, DOI 10.17487/RFC5914, June 2010, <http://www.rfc-editor.org/info/rfc5914>.

[RFC6810] Bush, R. and R. Austein, "The Resource Public Key Infrastructure (RPKI) to Router Protocol", RFC 6810, DOI 10.17487/RFC6810, January 2013, <http://www.rfc-editor.org/info/rfc6810>.

Authors' Addresses

Xiaodong Lee
CNNIC
No.4 South 4th Street, Zhongguancun
Beijing, 100190
P.R. China

Email: xl@cnnic.cn

Xiaowei Liu
CNNIC
No.4 South 4th Street, Zhongguancun
Beijing, 100190
P.R. China

Email: liuxiaowei@cnnic.cn

Zhiwei Yan
CNNIC
No.4 South 4th Street, Zhongguancun
Beijing, 100190
P.R. China

Email: yanzhiwei@cnnic.cn

Guanggang Geng
CNNIC
No.4 South 4th Street, Zhongguancun
Beijing, 100190
P.R. China

Email: gengguanggang@cnnic.cn

Yu Fu
CNNIC
No.4 South 4th Street, Zhongguancun
Beijing, 100190
P.R. China

Email: fuyu@cnnic.cn

SIDR
Internet-Draft
Intended status: Informational
Expires: October 14, 2016

D. Ma
ZDNS
S. Kent
BBN

April 12, 2016

Requirements for Resource Public Key Infrastructure (RPKI) Relying Parties
draft-madi-sidr-rp-00

Abstract

This document provides a single reference point for requirements for Relying Party (RP) software for use in the Resource Public Key Infrastructure (RPKI). It cites requirements that appear in several RPKI RFCs, making it easier for implementers to become aware of these requirements.

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 October 14, 2016.

Copyright Notice

Table of Contents

1. Introduction

Relying party software is used by network operators and others to acquire and verify Internet Number Resource (INR) data stored in the RPKI repository system. RPKI data, when verified, allows an RP to verify assertions about which Autonomous Systems (ASes) are authorized to originate routes for IP address prefixes. RPKI data also establishes binding between public keys and BGP routers, and indicates the AS numbers that each router is authorized to represent.

The follow sections present requirements imposed on RPs as defined in the following RFCs:

RFC 6480 (RPKI Architecture)
RFC 6481 (Repository Structure)
RFC 6482 (ROA format)
RFC 6485 (Algorithms)
RFC 6486 (Manifests)
RFC 6487 (Certificate and CRL profile)
RFC 6488 (RPKI Signed Objects)
RFC 6489 (Key Rollover)
RFC 6810 (RPKI to Router Protocol)
RFC 6916 (Algorithm Agility)
RFC 7730 (Trust Anchor Locator)
RFC XXXX (Router Certificates)

This document will be update to reflect new or changed requirements as these RFCs are updated, or new RFCs are written.

2. Fetching and Caching RPKI Repository Objects

RP software uses synchronization mechanisms supported by targeted repositories (e.g., [rsync]) to download all RPKI changed data objects in the repository system and cache them locally. The software validates the RPKI data and uses it to generate authenticated data identifying which ASes are authorized to originate routes for address prefixes, and which routers are authorized to sign BGP updates on behalf of ASes.

2.1. TAL Acquisition and Processing

In the RPKI, each relying party (RP) chooses its own set of trust anchors (TAs). Consistent with the extant INR allocation hierarchy, the IANA and/or the five RIRs are obvious candidates to be default TAs for the RP.

An RP does not retrieve TAs directly. A set of Trust Anchor Locators (TALs) is used by each RP to retrieve and verify the authenticity of each trust anchor.

TAL acquisition and processing are specified in Section 3 of [RFC7730].

2.2. Locating RPKI Objects Using Authority and Subject Information Extensions

The RPKI repository system is a distributed one, consisting of multiple repository instances. Each repository instance contains one or more repository publication points. An RP discovers publication points using the SIA and AIA extensions from (validated) certificates.

Section 5 of [RFC6481] specifies how an RP locates all RPKI objects by using the SIA and AIA extensions. Detailed specifications of SIA and AIA extensions in a resource certificate are described in section 4 of [RFC6487].

2.3. Dealing with Key Rollover

An RP takes the key rollover period into account with regard to its frequency of synchronization with RPKI repository system.

RP requirements in dealing with key rollover are described in section 3 of [RFC6489].

2.4. Dealing with Algorithm Transition

The set of cryptographic algorithms used with the RPKI is expected to change over time. Each RP is expected to be aware of the milestones established for the algorithm transition and what actions are required at every juncture.

RP requirements for dealing with algorithm transition are specified in section 4 of [RFC6916].

2.5. Strategies for Efficient Cache Maintenance

Each RP is expected to maintain a local cache of RPKI objects. The cache needs to be as up to date and consistent with repository publication point data as the RP's frequency of checking permits.

The last paragraph of section 5 of [RFC6481] provides guidance for maintenance of a local cache.

3. Certificate and CRL Processing

The RPKI make use of X.509 certificates and CRLs, but it profiles these standard formats [RFC6487]. The major change to the profile established in [RFC5280] is the mandatory use of a new extension to X.509 certificate [RFC3779].

3.1. Verifying Resource Certificate and Syntax

Certificates in the RPKI are called resource certificates, and they are required to conform to the profile [RFC6487]. An RP is required to verify that a resource certificate adheres to the profile established by [RFC6487]. This means that all extensions mandated by [RFC6487] must be present and value of each extension must be within the range specified by this RFC. Moreover, any extension excluded by [RFC6487] must be omitted.

Section 7.1 of [RFC6487] gives the procedure that the RP should follow to verify resource certificate and syntax.

3.2. Certificate Path Validation

In the RPKI, issuer can only assign and/or allocate public INRs belong to it, thus the INRs in issuer's certificate are required to encompass the INRs in the subject's certificate. This is one of necessary principles of certificate path validation in addition to cryptographic verification i.e., verification of the signature on each certificate using the public key of the parent certificate).

Section 7.2 of [RFC6487] gives the procedure that the RP should follow to perform certificate path validation.

3.3. CRL Processing

The CRL processing requirements imposed on CAs and RP are described in [RFC6487]. CRLs in the RPKI are tightly constrained; only the AuthorityKeyIndetifier and CRLNumber extensions are allowed, and they MUST be present. No other CRL extensions are allowed, and no CRLEntry extensions are permitted. RPs are required to verify that these constraints have been met. Each CRL in the RPI MUST be verified using the public key from the certificate of the CA that issued the CRL.

In the RPKI, RPs are expected to pay extra attention when dealing with a CRL that is not consistent with the Manifest associated with the publication point associated with the CRL.

Processing of a CRL that is not consistent with a manifest is a matter of local policy, as described in the fourth paragraph of Section 6.6 of [RFC6486].

4. Processing RPKI Repository Signed Objects

4.1. Basic Signed Object Syntax Checks

Before an RP can use a signed object from the RPKI repository, the RP is required to check the signed object syntax.

Section 3 of [RFC6488] lists all the steps that the RP is required to execute in order to validate the top level syntax of a repository signed object.

Note that these checks are necessary, but not sufficient. Additional validation checks must be performed based on the specific type of signed object.

4.2. Syntax and Validation for Each Type of Signed Object

4.2.1. Manifest

To determine whether a manifest is valid, the RP is required to perform manifest-specific checks in addition to those specified in [RFC6488].

Specific checks for a Manifest are described in section 4 of [RFC6486]. If any of these checks fails, indicating that the manifest is invalid, then the manifest will be discarded and treated as though no manifest were present.

4.2.2. ROA

To validate a ROA, the RP is required perform all the checks specified in [RFC6488] as well as the additional ROA-specific validation steps. The IP address delegation extension [RFC3779] present in the end-entity (EE) certificate (contained within the ROA), must encompass each of the IP address prefix(es) in the ROA.

More details for ROA validation are specified in section 2 of [RFC6482].

4.2.3. Ghostbusters

The Ghostbusters Record is optional; a publication point in the RPKI can have zero or more associated Ghostbuster Records. If a CA has at least one Ghostbuster Record, RP is required to verify that this Ghostbusters Record conforms to the syntax of signed object defined in [RFC6488].

The payload of this signed object is a (severely) profiled vCard. An RP is required to verify that the payload of Ghostbusters conforms to format as profiled in [RFC6493].

4.2.4. Verifying BGPsec Router Certificate

A BGPsec Router Certificate is a resource certificate, so it is required to comply with [RFC6487]. Additionally, the certificate must contain an AS Identifier Delegation extension, and must not contain an IP Address Delegation extension. The validation procedure used for BGPsec Router Certificates is identical to the validation procedure described in Section 7 of [RFC6487], but using the constraints applied come from specification of section 7 of [ID.sidr-bgpsec-pki-profiles].

Note that the cryptographic algorithms used by BGPsec routers are found in [ID.sidr-bgpsec-algs]. Currently, the algorithms specified in [ID.sidr-bgpsec-algs] and [ID.sidr-rfc6485bis] are different. BGPsec RPs will need to support algorithms that are used to validate BGPsec signatures as well as the algorithms that are needed to validate signatures on BGPsec certificates, RPKI CA certificates, and RPKI CRLs.

4.3. How to Make Use of Manifest Data

For a given publication point, the RP ought to perform tests to determine the state of the Manifest at the publication point. A Manifest can be classified as either valid or invalid, and a valid Manifest is either current and stale. An RP decides how to make use of a Manifest based on its state, according to local (RP) policy.

If there are valid objects in a publication point that are not present on a Manifest, [RFC6486] does not mandate specific RP behavior with respect to such objects. However, most RP software ignores such objects and this document recommends that this behavior be adopted uniformly.

In the absence of a Manifest, an RP is expected to accept all valid signed objects present in the publication point. If a Manifest is stale (see [RFC6486]) and an RP has no way to acquire a more recent Manifest, the RP is expected to (TBD).

4.4. What to Do with Ghostbusters Information

An RP may encounter a stale Manifest or CRL, or an expired CA certificate or ROA at a publication point. An RP is expected to use the information from the Ghostbusters record to contact the maintainer of the publication point where any stale/expired objects were encountered. The intent here is to encourage the relevant CA and/or repository manager to update the slate or expired objects.

5. Delivering Validated Cache to BGP Speakers

On a periodic basis, BGP speakers within an AS request updated validated origin AS data and router/ASN data from the RP’s cache. The RP passes this information to BGP speakers to enable them to verify the authenticity of routing announcements. The specification of the protocol designed to deliver validated cache data from an RP to a BGP Speaker is provided in [RFC6810].

6. Security considerations

TBD

7. IANA Considerations

This document has no actions for IANA.

8. Acknowledgements

The authors thank David Mandelberg and Wei Wang for their review, feedback and editorial assistance in preparing this document.

9. References

9.1. Normative References

[ID.sidr-bgpsec-algs]
Turner, S., "BGPsec Algorithms, Key Formats and Signature Formats", work-in-progress, <draft-ietf-sidr-bgpsec-algs>.

[ID.sidr-bgpsec-pki-profiles]
Turner, S., "A Profile for BGPsec Router Certificates, Certificate Revocation Lists, and Certification Requests", work-in-progress, <draft-ietf-sidr-bgpsec-pki-profiles>.

[ID.sidr-rfc6485bis]
Huston, G. and G. Michaelson, "The Profile for Algorithms and Key Sizes for use in the Resource Public Key Infrastructure", work-in-progress, <draft-ietf-sidr-rfc6485bis>.

[RFC3779] Lynn, C., Kent, S., and K. Seo, "X.509 Extensions for IP Addresses and AS Identifiers", RFC 3779, DOI 10.17487/RFC3779, June 2004, <http://www.rfc-editor.org/info/rfc3779>.

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

[RFC6481] Huston, G., Loomans, R., and G. Michaelson, "A Profile for Resource Certificate Repository Structure", RFC 6481, DOI 10.17487/RFC6481, February 2012, <http://www.rfc-editor.org/info/rfc6481>.

[RFC6482] Lepinski, M., Kent, S., and D. Kong, "A Profile for Route Origin Authorizations (ROAs)", RFC 6482, DOI 10.17487/RFC6482, February 2012, <http://www.rfc-editor.org/info/rfc6482>.

[RFC6486] Austein, R., Huston, G., Kent, S., and M. Lepinski, "Manifests for the Resource Public Key Infrastructure (RPKI)", RFC 6486, DOI 10.17487/RFC6486, February 2012, <http://www.rfc-editor.org/info/rfc6486>.

[RFC6487] Huston, G., Michaelson, G., and R. Loomans, "A Profile for X.509 PKIX Resource Certificates", RFC 6487, DOI 10.17487/RFC6487, February 2012, <http://www.rfc-editor.org/info/rfc6487>.

[RFC6488] Lepinski, M., Chi, A., and S. Kent, "Signed Object Template for the Resource Public Key Infrastructure (RPKI)", RFC 6488, DOI 10.17487/RFC6488, February 2012, <http://www.rfc-editor.org/info/rfc6488>.

[RFC6489] Huston, G., Michaelson, G., and S. Kent, "Certification Authority (CA) Key Rollover in the Resource Public Key Infrastructure (RPKI)", BCP 174, RFC 6489, DOI 10.17487/RFC6489, February 2012, <http://www.rfc-editor.org/info/rfc6489>.

[RFC6493] Bush, R., "The Resource Public Key Infrastructure (RPKI) Ghostbusters Record", RFC 6493, DOI 10.17487/RFC6493, February 2012, <http://www.rfc-editor.org/info/rfc6493>.

[RFC6810] Bush, R. and R. Austein, "The Resource Public Key Infrastructure (RPKI) to Router Protocol", RFC 6810, DOI 10.17487/RFC6810, January 2013, <http://www.rfc-editor.org/info/rfc6810>.

[RFC6916] Gagliano, R., Kent, S., and S. Turner, "Algorithm Agility Procedure for the Resource Public Key Infrastructure (RPKI)", BCP 182, RFC 6916, DOI 10.17487/RFC6916, April 2013, <http://www.rfc-editor.org/info/rfc6916>.

[RFC7730] Huston, G., Weiler, S., Michaelson, G., and S. Kent, "Resource Public Key Infrastructure (RPKI) Trust Anchor Locator", RFC 7730, DOI 10.17487/RFC7730, January 2016, <http://www.rfc-editor.org/info/rfc7730>.

9.2. Informative References

[RFC6480] Lepinski, M. and S. Kent, "An Infrastructure to Support Secure Internet Routing", RFC 6480, DOI 10.17487/RFC6480, February 2012, <http://www.rfc-editor.org/info/rfc6480>.

[rsync] "rsync web page", <http://rsync.samba.org/>.

Authors' Addresses

Di Ma
ZDNS
4 South 4th St. Zhongguancun
Haidian, Beijing 100190
China

Email: madi@zdns.cn

Stephen Kent
BBN
10 Moulton St
Cambridge, MA 02138-1119
USA

Email: kent@bbn.com

Internet Engineering Task Force
Internet-Draft
Intended status: Informational
Expires: January 21, 2017

A. Newton, Ed.
ARIN
C. Martinez-Cagnazzo, Ed.
LACNIC
D. Shaw
AFRINIC
T. Bruijnzeels
RIPE NCC
B. Ellacott
APNIC
July 20, 2016

RPKI Multiple "All Resources" Trust Anchors Applicability Statement
draft-rir-rpki-allres-ta-app-statement-01

Abstract

This document provides an applicability statement for the use of multiple, over-claiming 'all resources' (0/0) RPKI certificate authorities (CA) certificates used as trust anchors (TAs) operated by the Regional Internet Registry community to help mitigate the risk of massive downstream invalidation in the case of transient registry inconsistencies.

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 January 21, 2017.

Copyright Notice

Table of Contents

1. 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].

2. Introduction

The RPKI is a hierarchical cryptologic system that uses X.509 certificates to match and validate holdership of Internet number resources. This validation follows the allocation change from IANA to an RIR, to an NIR or LIR, and ending with end users who make use of the address block. Since these allocations can be cryptographically validated, this can then be tied to assertions made by the holder of those number resources. As an improvement of this system, the RPKI was updated to add validation of origin routing announcements via ROAs. These ROAs can then be independently and cryptographically validated by third parties to assure themselves that the origin of the announcement as seen in the actual routing system is valid.

Since this system is envisioned to be used by network operators and ISPs to determine their routing decisions, there is a goal to be 100% correct 100% of the time. This goal could be achieved if the system was contained in a static environment where there is little or no movement of holdership changes from one organization to another of number resources. Unfortunately, this state cannot be achieved today, as movement of number resouces from from organization to organization is becoming common largely due to IPv4 scarcity.

Unfortunately, this state of 100% correctness at all times is infeasible in a model where separate entities are operating independently, yet rely critically on each others' perfect synchronisation at all times.

Because the current validation mechanism is all-or-nothing, any inconsistency at all at a high apex CA has the potential to invalidate a large number of additional Internet Number Resources. The higher the apex, and the larger the total set of INRs maintained by the CA, the greater the impact of even a small inconsistency.

As resources do change at high apex CAs for a variety of reasons, the likelihood of a small inconsistency is non-zero. And the likelihood of a transitional inconsistency is moderate. Due to the distributed nature of the RPKI repository mechanism, even if all CAs were able to operate in perfect synchronicity at all times, there is a reasonable likelihood that a given validating client may witness a temporarily inconsistent state of the system as a whole. A risk of wide-spread invalidity therefore exists as a very high impact and moderate likelihood event.

This brittleness in the RPKI validation rules has been identified and presented by the current RPKI TA operators to the IETF. A solution has also been proposed ([I-D.ietf-sidr-rpki-validation-reconsidered]), a solution that would allow for accidental over-claiming only to invalidate the resource that is incorrectly listed and allow the remaining to continue to be valid. As the implementation and deployment of solutions to this problem will occur according to timelines outside the control of the current TA operators, the workaround proposed in the present draft provides an acceptable trade-off.

3. Applicability to reduce overclaiming possibilities

The consequences of an RIR over-claiming are grave given that every ISP within their certificate would be invalidated. If routing was to be reliant on RPKI at this point, all routes announced by those ISPs below the affected RIR certificate would cease to work.

To mitigate risk and alleviate this threat, each RIR will move from a Trust Anchor that reflects their current holdings only, to one that reflects all holdings (e.g. 0/0). This will then ensure that over-claiming can not occur at a RIR level when dealing with transfers from one RIR to another. RPKI validators will not see the five Trust anchors from the RIRs as over-claiming and validation can proceed normally.

For those who may want to audit the RIRs to ensure that RIRs are not allocating the same IP addresses in separate regions, this can be done by matching the inventory of each RIR ([NROSTATS]) that is provided by the RIRs with the certificates issued by the RIRs within the RPKI.

Note that there will be minor changes from time to time to account for movements from IP address holdings that are in flight from one RIR to another and that transient overlaps can, and probably will, occur as inter-RIR transfers become more and more common.

4. Normative References

[I-D.ietf-sidr-rpki-validation-reconsidered]
Huston, G., Michaelson, G., Martinez, C., Bruijnzeels, T., Newton, A., and D. Shaw, "RPKI Validation Reconsidered", draft-ietf-sidr-rpki-validation-reconsidered-06 (work in progress), July 2016.

[NROSTATS]
"NRO Extended Stats File", July 2016, <https://www.nro.net/wp-content/uploads/apnic-uploads/delegated-extended>.

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

Authors' Addresses

Andrew Newton (editor)
ARIN
Chantilly VA
United States

Email: andy@arin.net

Carlos Martinez-Cagnazzo (editor)
LACNIC
Montevideo
Uruguay

Email: carlos@lacnic.net

Daniel Shaw
AFRINIC
Cybercity Ebene
Republic of Mauritius

Email: daniel@afrinic.net

Tim Bruijnzeels
RIPE NCC
Amsterdam
Netherlands

Email: tim@ripe.net

Byron Ellacott
APNIC
Brisbane
Australia

Email: bje@apnic.net

Internet Engineering Task Force
Internet-Draft
Intended status: Informational
Expires: January 19, 2018

A. Newton, Ed.
ARIN
C. Martinez-Cagnazzo, Ed.
LACNIC
D. Shaw
AFRINIC
T. Bruijnzeels
RIPE NCC
B. Ellacott
APNIC
July 18, 2017

RPKI Multiple "All Resources" Trust Anchors Applicability Statement
draft-rir-rpki-allres-ta-app-statement-02

Abstract

This document provides an applicability statement for the use of multiple, over-claiming 'all resources' (0/0) RPKI certificate authorities (CA) certificates used as trust anchors (TAs) operated by the Regional Internet Registry community to help mitigate the risk of massive downstream invalidation in the case of transient registry inconsistencies.

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 January 19, 2018.

Table of Contents

1. 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].

2. Introduction

The RPKI is a hierarchical cryptologic system that uses X.509 certificates to match and validate holdership of Internet number resources. This validation follows the allocation change from IANA to an RIR, to an NIR or LIR, and ending with end users who make use of the address block. Since these allocations can be cryptographically validated, this can then be tied to assertions made by the holder of those number resources. As an improvement of this system, the RPKI was updated to add validation of origin routing announcements via ROAs. These ROAs can then be independently and cryptographically validated by third parties to assure themselves that the origin of the announcement as seen in the actual routing system is valid.

Since this system is envisioned to be used by network operators and ISPs to determine their routing decisions, there is a goal to be 100% correct 100% of the time. This goal could be achieved if the system was contained in a static environment where there is little or no movement of holdership changes from one organization to another of number resources. Unfortunately, this state cannot be achieved today, as movement of number resouces from from organization to organization is becoming common largely due to IPv4 scarcity.

Unfortunately, this state of 100% correctness at all times is infeasible in a model where separate entities are operating independently, yet rely critically on each others' perfect synchronisation at all times.

Because the current validation mechanism is all-or-nothing, any inconsistency at all at a high apex CA has the potential to invalidate a large number of additional Internet Number Resources. The higher the apex, and the larger the total set of INRs maintained by the CA, the greater the impact of even a small inconsistency.

As resources do change at high apex CAs for a variety of reasons, the likelihood of a small inconsistency is non-zero. And the likelihood of a transitional inconsistency is moderate. Due to the distributed nature of the RPKI repository mechanism, even if all CAs were able to operate in perfect synchronicity at all times, there is a reasonable likelihood that a given validating client may witness a temporarily inconsistent state of the system as a whole. A risk of wide-spread invalidity therefore exists as a very high impact and moderate likelihood event.

This brittleness in the RPKI validation rules has been identified and presented by the current RPKI TA operators to the IETF. A solution has also been proposed ([I-D.ietf-sidr-rpki-validation-reconsidered]), a solution that would allow for accidental over-claiming only to invalidate the resource that is incorrectly listed and allow the remaining to continue to be valid. As the implementation and deployment of solutions to this problem will occur according to timelines outside the control of the current TA operators, the workaround proposed in the present draft provides an acceptable trade-off.

3. Applicability to reduce overclaiming possibilities

The consequences of an RIR over-claiming are grave given that every ISP within their certificate would be invalidated. If routing was to be reliant on RPKI at this point, all routes announced by those ISPs below the affected RIR certificate would cease to work.

To mitigate risk and alleviate this threat, each RIR will move from a Trust Anchor that reflects their current holdings only, to one that reflects all holdings (e.g. 0/0). This will then ensure that over-claiming can not occur at a RIR level when dealing with transfers from one RIR to another. RPKI validators will not see the five Trust anchors from the RIRs as over-claiming and validation can proceed normally.

For those who may want to audit the RIRs to ensure that RIRs are not allocating the same IP addresses in separate regions, this can be done by matching the inventory of each RIR ([NROSTATS]) that is provided by the RIRs with the certificates issued by the RIRs within the RPKI.

Note that there will be minor changes from time to time to account for movements from IP address holdings that are in flight from one RIR to another and that transient overlaps can, and probably will, occur as inter-RIR transfers become more and more common.

4. Normative References

[I-D.ietf-sidr-rpki-validation-reconsidered]
Huston, G., Michaelson, G., Martinez, C., Bruijnzeels, T., Newton, A., and D. Shaw, "RPKI Validation Reconsidered", draft-ietf-sidr-rpki-validation-reconsidered-06 (work in progress), July 2016.

[NROSTATS]
"NRO Extended Stats File", July 2016, <https://www.nro.net/wp-content/uploads/apnic-uploads/delegated-extended>.

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

Authors' Addresses

Andrew Newton (editor)
ARIN
Chantilly VA
United States

Email: andy@arin.net

Carlos Martinez-Cagnazzo (editor)
LACNIC
Montevideo
Uruguay

Email: carlos@lacnic.net

Daniel Shaw
AFRINIC
Cybercity Ebene
Republic of Mauritius

Email: daniel@afrinic.net

Tim Bruijnzeels
RIPE NCC
Amsterdam
Netherlands

Email: tim@ripe.net

Byron Ellacott
APNIC
Brisbane
Australia

Email: bje@apnic.net

Secure Inter-Domain Routing
Internet-Draft
Intended status: Informational
Expires: November 7, 2016

Z. Yan
Y. Fu
X. Liu
G. Geng
CNNIC

May 6, 2016

Problem Statement and Considerations for ROA Mergence
draft-yan-sidr-roa-mergence-00

Abstract

The address space holder needs to issue an ROA object when it authorizes one or more ASes to originate routes to multiple prefixes. During the process of ROA issuance, the address space holder needs to specify an origin AS for a list of IP prefixes. Besides, the address space holder has a free choice to put multiple prefixes into a single ROA or issue separate ROAs for each prefix based on the current specification. This memo analyzes and presents some operational problems which may be caused by the misconfigurations of ROAs containing multiple IP prefixes. Some suggestions and considerations also have been proposed.

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 November 7, 2016.

Table of Contents

1. Introduction

Route Origin Authorization (ROA) is a digitally signed object which is used to identify that a single AS has been authorized by the address space holder to originate routes to one or more prefixes within the address space[RFC6482].If the address space holder needs to authorize more than one ASes to advertise the same set of address prefixes, the holder must issue multiple ROAs, one per AS number. However, at present there are no mandatory requirements in any RFCs describing that the address space holders must issue a separate ROA for each prefix or a ROA for multiple prefixes.

Each ROA contains an "asID" field and an "ipAddrBlocks" field. The "asID" field contains one single AS number which is authorized to originate routes to the given IP address prefixes. The "ipAddrBlocks" field contains one or more IP address prefixes to which the AS is authorized to originate the routes. The ROA mergence is a common case that each ROA contains exactly one AS number but may contain multiple IP address prefixes in the operational process of ROA issuance.

2. 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].

3. Problem statement and Analysis

3.1. Statistical analysis of ROA mergence

As mentioned above, the address space holder needs to issue an ROA object when it authorizes one or more ASes to originate routes to multiple prefixes. During the process of ROA issuance, the address space holder needs to specify an origin AS for a list of IP prefixes. Besides, the address space holder has a free choice to put multiple prefixes into a single ROA or issue separate ROAs for each prefix based on the current specification.

On our RPKI testbed, the Trust Anchor Locator (TAL) files configured by RP correspond to the five RIRs' RPKI Trust Anchors. By using these TAL files, all the ROA objects issued in each region (the five RIRs) around the world are collected and validated with the RPKI Relying Party tools provided by rpki.net. According to the analysis on these data, some statistical results are described in Table. 1.

The total number of ROAs	The number of ROAs with a single prefix	The number of ROAs with multiple prefixes
5027	2341	2686

Table.1 Statistical results of all ROAs

As shown in Table. 1, by now (as of April 19, 2016), the total number of ROA objects issued around the world is about 5027. The result is in accordance with the statistics provided by RIPE NCC and Internet Multifeed Co. (MF). Based on the further analysis on these ROA objects, it is found that: the number of ROAs containing only one prefix is about 2341 (account for 46.6% of all ROA objects), and the number of ROAs containing two or more prefixes is about 2686 (account for 53.4% of all ROA objects).

In the 2686 ROA objects which each one contains two or more prefixes, the number of IP address prefixes are calculated and analyzed. The statistical results are shown in Table. 2.

The number of prefixes	The number of ROAs	The average number of prefixes in each ROA
20379	2686	7.59

Table. 2 Statistical results of the 2686 ROAs

As described in Table. 2, there are 20379 IP address prefixes in the 2686 ROA objects. And the average number of prefixes in each ROA is 7.59 (20379/2686). In addition, four types of ROAs are analyzed and calculated in the 2686 ROAs: ROAs each contains 2-10/11-50/51-100/>100 IP address prefixes. The statistical results are presented in Table. 3.

ROA types	ROA with 2-10 prefixes	ROA with 11-50 prefixes	ROA with 51-100 prefixes	ROA with >100 prefixes	Total
The number of ROAs	2316	325	29	16	2686
The ratio of ROAs	86.22%	12.10%	1.08%	0.60%	100.00%
The number of prefixes	8849	6563	1917	3050	20379
The ratio of prefixes	43.42%	32.20%	9.41%	14.97%	100.00%

Table. 3 Statistical results of four types of ROAs

As shown in Table. 3, taking the first type of ROA as an example, there are 2316 ROAs (account for 86.22% of the 2628 ROA objects) which each contains 2-10 IP address prefixes, and the total number of IP prefixes in these 2316 ROAs is 8849 (account for 43.42% of the 20379 prefixes).

According to the third row (the ratio of ROAs) in Table. 3, it shows the trend that the address space holders tend to issue each ROA object with fewer IP prefixes (more than 98% of ROAs containing less

than 50 prefixes), but they still tend to put multiple prefixes into one single ROA.

It should also be paid more attention that among all the ROAs issued today, a single ROA may contain a large number of IP address prefixes. In the statistical results, it is found that there exists two ROAs (corresponding to ASN 3215 and ASN 9299) which each contains more than 300 IP address prefixes (324 and 375 respectively).

3.2. Experimental analysis of ROA mergence

A large number of experiments for the process of ROA issuance have been made on our RPKI testbed, it is found that the misconfigurations during the issuance may cause the ROAs which have been issued to be revoked. The corresponding scenarios are as follows.

AS shown in Fig. 1, an ISP needed to issue two ROA objects respectively to authorize ASN 64500 to originate routes to IP prefixes 192.0.2.128/28 and ASN 64501 to originate routes to IP prefixes 198.51.100.128/28. The operations are simulated on our RPKI testbed.

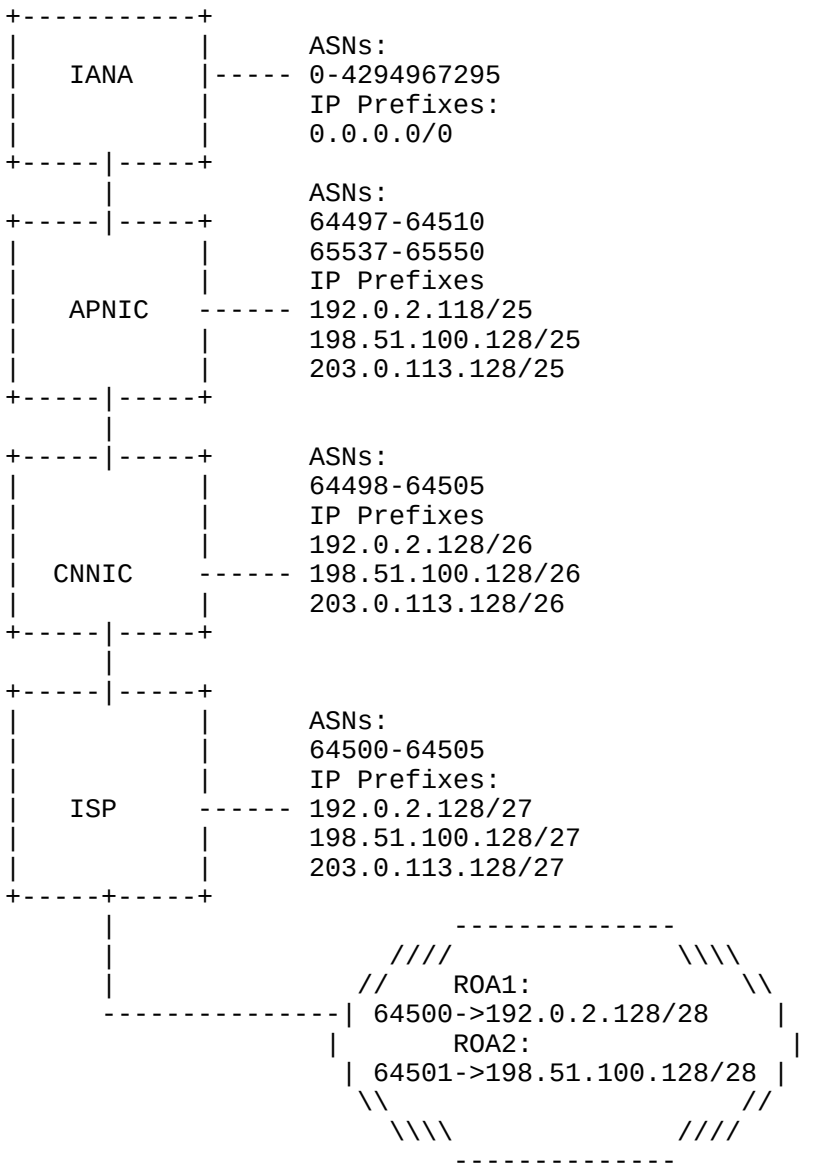

Fig. 1 Scenario of ROA issuance

The ROA objects issued by ISP could be checked with the "show_published_objects" command. And as shown in Fig. 2, ISP has issued two ROA objects M74Rq1am9m4YUairntkXTRAx6Wg.roa and vulw_jMZBy7-ktn7nyhlpchBKZY.roa to respectively authorize ASN 64500

to originate routes to IP prefixes 192.0.2.128/28 and ASN 64501 to originate routes to IP prefixes 198.51.100.128/28.

```
test@~$cat ISPROA.csv
192.0.2.128/28  64500 Group1
198.51.100.128/28  64501 Group2
test@~$ rpkic -i ISP load_roa_requests ISPROA.csv
test@~$ rpkic -i ISP show_published_objects
rsync://ubuntu/rpki/IANA/APNIC/CNNIC/ISP/duPylfF7Hv31rpOa4dVVCZnRkmk.crl
2016-04-19T10:34:04Z 594CB167AF4E81424EBEA7C1A5FD8DDE216D5C69
rsync://ubuntu/rpki/IANA/APNIC/CNNIC/ISP/duPylfF7Hv31rpOa4dVVCZnRkmk.mft
2016-04-19T10:34:04Z 17C98CBFB179D60D9D0A6D52C2629B7A8DEA8A9C
rsync://ubuntu/rpki/IANA/APNIC/CNNIC/ISP/M74Rq1am9m4YUairntkXTRAx6Wg.roa
2016-04-19T09:20:20Z 0CFD927D1522BF43FC52B748F274646387569222
64500 192.0.2.128/28
rsync://ubuntu/rpki/IANA/APNIC/CNNIC/ISP/vulw_jMZBY7-KTN7nyhlpchBKZY.roa
2016-04-19T10:34:04Z 305866D0c4ee5e156ebeda811d3540bf0e094043
64501 198.51.100.128/28
```

Fig. 2 Check the ROAs issued by ISP

Afterwards, ISP wanted to authorize ASN 64501 to originate routes to another IP prefixes 203.0.113.128/28, so it modified the ISPROA.csv file and operated the "load_roa_requests" command again.

```
test@~$cat ISPROA.csv
192.0.2.128/28  64500 Group1
198.51.100.128/28  64501 Group2
203.0.113.128/28  64501 Group2
test@~$ rpkic -i ISP load_roa_requests ISPROA.csv
test@~$ rpkic -i ISP show_published_objects
rsync://ubuntu/rpki/IANA/APNIC/CNNIC/ISP/duPylfF7Hv31rpOa4dVVCZnRkmk.crl
2016-04-19T10:38:03Z 2606EAA75AB60BE7785AE0CB0599D984AFD5BDB5
rsync://ubuntu/rpki/IANA/APNIC/CNNIC/ISP/duPylfF7Hv31rpOa4dVVCZnRkmk.mft
2016-04-19T10:38:03Z 10F3F9249F0A6A636BF8143075693681B45A4BC2
rsync://ubuntu/rpki/IANA/APNIC/CNNIC/ISP/M74Rq1am9m4YUairntkXTRAx6Wg.roa
2016-04-19T09:20:20Z 0CFD927D1522BF43FC52B748F274646387569222
64500 192.0.2.128/28
rsync://ubuntu/rpki/IANA/APNIC/CNNIC/ISP/vO3whtjMpYxxyva4BxRqI2H8eqA.roa
2016-04-19T10:38:03Z 4B85FDBABEC567A9DD8DA5745B34A201390F4530
64501 198.51.100.128/28,203.0.113.128/28
```

Fig. 3 Add a new authorization

As shown in Fig. 3, after processing the above operations, a new ROA object vO3WhtjMpYxxyva4BxRqI2H8eqA.roa which contained two IP prefixes was issued. One thing which needs to be noticed is that in the ISPROA.csv file the third column of the last two lines (with

respect to ASN 64501) are set as the same label "Group2" to make sure that the authorizations to the two IP prefixes will be issued into a single ROA.

Now, ISP wants to authorize ASN 64500 to originate routes to IP prefixes 203.0.113.128/28 as well, but when it modifies the ISPROA.csv file, it appends 204.0.113.128/28 (or any prefixes that do not belong to ISP) instead of 203.0.113.128/28 into the ISPROA.csv file by mistake. And then, when it operates the "load_roa_requests" command, something unexpected will happen.

```
test@~$cat ISPROA.csv
192.0.2.128/28  64500 Group1
204.0.113.128/28 64500 Group1
198.51.100.128/28  64501 Group2
203.0.113.128/28  64501 Group2
test@~$ rpkic -i ISP load_roa_requests ISPROA.csv
test@~$ rpkic -i ISP show_published_objects
rsync://ubuntu/rpki/IANA/APNIC/CNNIC/ISP/duPylfF7Hv31rpOa4dVVCZnRkmk.crl
2016-04-19T12:39:47Z 2DD037213237D72AF6CE95F8F37D1F08E8B49A37
rsync://ubuntu/rpki/IANA/APNIC/CNNIC/ISP/duPylfF7Hv31rpOa4dVVCZnRkmk.mft
2016-04-19T12:39:47Z 735D9723B8C6D8214DA78117D27E529AA47E14B6
rsync://ubuntu/rpki/IANA/APNIC/CNNIC/ISP/vO3whtjMpYxxyva4BxRqI2H8eqA.roa
2016-04-19T10:38:03Z 4B85FDBABEC567A9DD8DA5745B34A201390F4530
64501 198.51.100.128/28,203.0.113.128/28
```

Fig. 4 Add an incorrect authorization by mistake

As shown in Fig. 4, a legitimate ROA object was revoked because of ISP’s misconfiguration. Obviously, this misconfiguration may lead to some serious consequences to RPKI (such as legitimate BGP routes are misclassified as "invalid").

3.3. Problem statement

It concludes that the misconfigurations of ROAs containing multiple IP address prefixes may lead to much more serious consequences than ROAs with fewer IP address prefixes. According to the above statistical and experimental analysis, misconfigurations of the ROAs which contain more than 300 IP address prefixes may cause a large-scale network interruption.

Another potential influence of misconfigurations of ROAs containing multiple IP prefixes on BGP routers may be considered. For the ROA containing multiple prefixes, once increase or delete one <AS, ip_prefix> pair in it, this ROA will be reissued. Through sychronization with repository, RPs fetch a new ROA object and then notify and send all the <AS, ip_prefix> pairs in this ROA to BGP

routers. That is to say, the update of the ROA containing multiple IP address prefixes will lead to redundant transmission between RP and BGP routers . So frequent update of these ROAs will increase the convergency time of BGP routers and reduce their performance obviously.

4. Suggestions and Considerations

Based on the statistical and experimental analysis, following considerations should be considered during the process of ROA issuance:

1) The issuance of ROAs containing a large number of IP prefixes may lead to misconfigurations more easily than ROAs with fewer IP prefixes.

A ROA which contains a large number of IP prefixes is more vulnerable to misconfigurations, because any misconfiguration of these prefixes may cause the legitimate ROA to be revoked. Besides, since the misconfigurations of ROAs containing a larger number of IP address prefixes may lead to much more serious consequences (a large-scale network interruption) than ROAs with fewer IP address prefixes, it is suggested to avoid issuing ROAs with a large number of IP address prefixes.

2) The number of ROAs containing multiple IP prefixes should be limited and the number of IP prefixes in each ROA should also be limited.

The extreme case (a single ROA can only contain one IP address prefix) may lead to too much ROA objects globally, which may in turn become a burden for RPs to synchronize and validate all these ROA objects with the fully deployment of RPKI. So a tradeoff between the number of ROAs and the number of IP prefixes in a single ROA should be considered.

3) A safeguard scheme is essential to protect the process of ROA issuance

Considering the misconfigurations during the process of ROA issuance are inevitable and the serious consequences they may lead to, a safeguard scheme to protect and monitor the process of ROA issuance should be considered.

5. Security Considerations

TBD.

6. IANA Considerations

This draft does not request any IANA action.

7. Acknowledgements

The authors would like to thanks the valuable comments made by XXX and other members of sidr WG.

This document was produced using the xml2rfc tool [RFC2629].

8. References

8.1. Normative References

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

[RFC6480] Lepinski, M. and S. Kent, "An Infrastructure to Support Secure Internet Routing", RFC 6480, DOI 10.17487/RFC6480, February 2012, <http://www.rfc-editor.org/info/rfc6480>.

[RFC6482] Lepinski, M., Kent, S., and D. Kong, "A Profile for Route Origin Authorizations (ROAs)", RFC 6482, DOI 10.17487/RFC6482, February 2012, <http://www.rfc-editor.org/info/rfc6482>.

[RFC6486] Austein, R., Huston, G., Kent, S., and M. Lepinski, "Manifests for the Resource Public Key Infrastructure (RPKI)", RFC 6486, DOI 10.17487/RFC6486, February 2012, <http://www.rfc-editor.org/info/rfc6486>.

8.2. Informative References

[RFC2629] Rose, M., "Writing I-Ds and RFCs using XML", RFC 2629, DOI 10.17487/RFC2629, June 1999, <http://www.rfc-editor.org/info/rfc2629>.

[RFC5914] Housley, R., Ashmore, S., and C. Wallace, "Trust Anchor Format", RFC 5914, DOI 10.17487/RFC5914, June 2010, <http://www.rfc-editor.org/info/rfc5914>.

Authors' Addresses

Zhiwei Yan
CNNIC
No.4 South 4th Street, Zhongguancun
Beijing, 100190
P.R. China

Email: yanzhiwei@cnnic.cn

Yu Fu
CNNIC
No.4 South 4th Street, Zhongguancun
Beijing, 100190
P.R. China

Email: fuyu@cnnic.cn

Xiaowei Liu
CNNIC
No.4 South 4th Street, Zhongguancun
Beijing, 100190
P.R. China

Email: liuxiaowei@cnnic.cn

Guanggang Geng
CNNIC
No.4 South 4th Street, Zhongguancun
Beijing, 100190
P.R. China

Email: gengguanggang@cnnic.cn