Internet Engineering Task Force T. Harrison

Internet-Draft G. Michaelson

Updates: 9286 (if approved) APNIC

Intended status: Standards Track J. Snijders

Expires: 18 December 2025 16 June 2025

Resource Public Key Infrastructure (RPKI) Manifest Number Handling

draft-ietf-sidrops-manifest-numbers-05

Abstract

The Resource Public Key Infrastructure (RPKI) makes use of signed

Objects, called manifests. A manifest lists each file that a

publisher intends to include within an RPKI repository, and can be

used to detect certain forms of attack against a repository.

Manifests include a "manifest number" (manifestNumber), which the

publisher must increment whenever it issues a new manifest, and

Relying Parties (RPs) are required to verify that a newly-retrieved

manifest for a given Certification Authority (CA) has a higher

manifestNumber than the previously-validated manifest. However, the

manifestNumber field is 20 octets in length (i.e., bounded), and

no behaviour is specified for when a manifestNumber reaches the

largest possible value.

This document update RFC 9286 by specifying publisher and RP

behaviour for this scenario.

Status of This Memo

This Internet-Draft is submitted in full conformance with the

provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering

Task Force (IETF). Note that other groups may also distribute

working documents as Internet-Drafts. The list of current Internet-

Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months

and may be updated, replaced, or obsoleted by other documents at any

time. It is inappropriate to use Internet-Drafts as reference

material or to cite them other than as "work in progress."

This Internet-Draft will expire on 18 December 2025.

Copyright Notice

Copyright (c) 2025 IETF Trust and the persons identified as the

document authors. All rights reserved.

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

Provisions Relating to IETF Documents (https://trustee.ietf.org/

license-info) in effect on the date of publication of this document.

Please review these documents carefully, as they describe your rights

and restrictions with respect to this document. Code Components

extracted from this document must include Revised BSD License text as

described in Section 4.e of the Trust Legal Provisions and are

provided without warranty as described in the Revised BSD License.

Table of Contents

1. Introduction . . . . . . . . . . . . . . . . . . . . . . . . 2

1.1. Requirements Language . . . . . . . . . . . . . . . . . . 4

2. Manifest Number Handling . . . . . . . . . . . . . . . . . . 4

3. General Repository Handling . . . . . . . . . . . . . . . . . 5

4. Operational Considerations . . . . . . . . . . . . . . . . . 5

5. IANA Considerations . . . . . . . . . . . . . . . . . . . . . 5

6. Implementation status . . . . . . . . . . . . . . . . . . . . 6

7. Acknowledgements . . . . . . . . . . . . . . . . . . . . . . 6

8. References . . . . . . . . . . . . . . . . . . . . . . . . . 6

8.1. Normative References . . . . . . . . . . . . . . . . . . 6

8.2. Informative References . . . . . . . . . . . . . . . . . 7

Appendix A. Serial Number Arithmetic . . . . . . . . . . . . . . 8

Appendix B. Manifest thisUpdate . . . . . . . . . . . . . . . . 9

Authors' Addresses . . . . . . . . . . . . . . . . . . . . . . . 9

1. Introduction

The Resource Public Key Infrastructure (RPKI) [RFC6480] makes use of

signed objects [RFC6488], called manifests [RFC9286]. A manifest

lists each file that a publisher intends to include within an RPKI

repository [RFC6481], and can be used to detect certain forms of

attack against a repository. Manifests include a "manifest number"

(manifestNumber), which a publisher must increment by one whenever it

issues a new manifest, and Relying Parties (RPs) are required to

verify that a newly-retrieved manifest for a given Certification

Authority (CA) has a higher manifestNumber than the previously-

validated manifest (Section 4.2.1 of [RFC9286]).

However, the manifestNumber field is 20 octets in length (i.e., bounded), and no behaviour is specified for when a manifestNumber

reaches the largest possible value (2^159-1). When that value is

reached, some RP implementations will accept a new manifest for the

CA only once the current manifest has expired, while others will not

accept a new manifest at all. (For the purposes of [RFC9286], a "CA"

is represented by a CA certificate with a stable location and a

stable private key. Reissuing a CA certificate with changed

resources or a changed expiry date does not change the identity of

the CA such that the stored manifestNumber for the CA is reset, for

example.)

While it is practically impossible for a publisher to reach the

largest possible value under normal operating conditions (it would

require that the publisher issue one manifest per second for

23,171,956,451,847,141,650,870 quintillion years), there is still a chance

that it could be reached due to bugs in the issuance or publication

systems or incorrect/inadvertent use of those systems. Examples of misbehaviors are:

* Incrementing by large values when issuing manifests, such that the

time to reach that largest value is reduced.

* Reissuing new manifests in an infinite delay-free loop, such that

the manifestNumber increases by a large value in a comparatively

short period of time.

* Inadvertently setting the manifestNumber to the largest possible

value, such that the publisher will no longer be able to publish

usable manifests for that repository.

These scenarios might also arise in combination and be more severe as a result. For example, a CA might increase the manifestNumber by a large value on reissuance, and also reissue the manifest more frequently than is necessary.

For a subordinate CA, the risk of repository invalidation due to such a

problem can be addressed by the publisher using the key

rollover process [RFC6489] to get a new CA certificate. RPs will

treat this new certificate as though it represents a distinct CA, and

the manifestNumber can be reset at that point.

However, this option is not available for RPKI Trust Anchors (TAs).

If a TA publishes a manifest with the largest-possible manifestNumber

value, then it is difficult to rely on the TA after that point, since

(as described previously) some RPs will not accept a new manifest until

the current one has expired, while others will reject all new

manifests indefinitely. Particularly in the case of TAs, the

manifest validity period may be quite long, too. Issuing a new TA

and distributing the associated Trust Anchor Locator (TAL) [RFC8630] to clients would involve a large

amount of work for TA operators and RPs. Additionally, depending on

the RP implementation being used, there would be a limited degree of

RPKI protection by way of that TA for the time between the issuance

of the problematic manifest and the installation of the new TAL.

In order to avoid these problems, this document updates [RFC9286] by defining how

publishers and RPs can handle this scenario in order to facilitate

ongoing use of an affected repository.

1.1. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT",

"SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and

"OPTIONAL" in this document are to be interpreted as described in

BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all

capitals, as shown here.

2. Manifest Number Handling

For a given CA, an RP MUST NOT reject a new manifest issued by that

CA on the basis of it not having a higher manifestNumber than a

previously-validated manifest if the new manifest has a different

filename from that of the previously-validated manifest. In other

words, an RP has to reset its stored manifestNumber for a given CA if

the CA changes the filename of its manifest.

With this behaviour, it is possible for a CA to be configured such

that any time it issues a new manifest, it uses a new filename for

that manifest. If a CA is configured in this way, the

manifestNumber validation set out in Section 4.2.1 of [RFC9286] would

have no purpose. To avoid this outcome, CAs SHOULD NOT use new

filenames for manifests except in situations where it is necessary to

ensure the ongoing validity of the CA or its repository. Similarly,

RP software SHOULD alert its operators when a manifest filename

changes for a given CA.

To avoid replay attacks, RPs MUST verify that the

URI in the accessLocation in one of the id-ad-signedObject

accessMethod objects in the manifest's Subject Information Access (SIA)

extension exactly matches the URI presented in the RPKI Repository

Delta Protocol (RRDP) [RFC8182] "publish" element or the path

presented by remote rsync servers.

Section 2.2 of [RFC6481] contains non-normative guidance for the

naming of manifest files in repositories. While a CA that supports

the behaviour described in this section cannot preserve the exact

filename suggested by that text (per Section 2.1 of [RFC4387]), the

CA SHOULD still ensure that the filename is a value derived from the

public key of the CA, per the more general guidance in that section.

A CA specifies its manifest URI by way of an SIA entry with an

accessMethod of id-ad-rpkiManifest (Section 4.8.8.1 of [RFC6487]). For the purposes of

this document, the manifest filename is the final segment of the path

of the accessLocation URI from that SIA entry.

Section 4.8.8.1 of [RFC6487] states that a CA may include in its

certificate multiple id-ad-rpkiManifest SIA entries. For

comparisons, an RP may use the filename from any one of the id-ad-

rpkiManifest SIA entries in the previously-validated CA certificate.

If that filename does not appear in any of the id-ad-rpkiManifest SIA

entries in the CA certificate that is currently being validated, then

the manifest filename has changed.

The corollary of this behaviour is that a CA that includes multiple

id-ad-rpkiManifest SIA entries in its certificate and wants to rely

on the behaviour defined in this document MUST ensure that none of

the manifest filenames in the previous CA certificate appear in the

newly-issued CA certificate.

The approach set out in this section is different from that

described in Section 3.2.1 of [RFC8488].

3. General Repository Handling

Section 2 contains a specific update to [RFC9286] for the handling of

manifest numbers to address one potential permanent

invalidity scenario. RPs that encounter other permanent invalidity

scenarios SHOULD also consider how those can be addressed such that

the scenario does not require the relevant CA or TA to perform a key

rollover operation. For example, in the event that an RP recognizes

that a permanent invalidity scenario has occurred, the RP could alert

the operator and provide an option to the operator to stop relying on

cached data for the affected repository, so that the CA can rectify

the problem.

4. Operational Considerations

CA software may opt to support the manifest number reset

functionality in various ways. For example, it could change the

manifest filename when the manifestNumber reaches a certain

threshold, or it could alert the operator in this scenario and

request confirmation that the filename should be changed.

5. IANA Considerations

This document has no actions for IANA.

5. Security Considerations

The RPKI primarily exists to support and improve security of the global Internet routing system.

Reliability improvements to the RPKI itself, such as outlined in this document, strengthen its dependability (see Section 8 of [RFC6480]).

[RFC9286] requires that RPs perform two replay-related checks on newly-retrieved manifests: firstly, that the purported new manifest has a greater manifestNumber than the cached manifest, and secondly, that the purported new manifest has a more recent thisUpdate than the cached manifest. An RP that implements the behaviour in Section 2 will momentarily omit the manifestNumber check following a manifest filename change. So long as the RP still performs the second check described above, it will be protected against replay attacks.

6. Implementation status

This section is to be removed before publishing as an RFC.

This section records the status of known implementations of the

protocol defined by this specification at the time of posting of this

Internet-Draft, and is based on a proposal described in [RFC7942].

The description of implementations in this section is intended to

assist the IETF in its decision processes in progressing drafts to

RFCs. Please note that the listing of any individual implementation

here does not imply endorsement by the IETF. Furthermore, no effort

has been spent to verify the information presented here that was

supplied by IETF contributors. This is not intended as, and must not

be construed to be, a catalog of available implementations or their

features. Readers are advised to note that other implementations may

exist.

According to [RFC7942], "this will allow reviewers and working groups

to assign due consideration to documents that have the benefit of

running code, which may serve as evidence of valuable experimentation

and feedback that have made the implemented protocols more mature.

It is up to the individual working groups to use this information as

they see fit".

\* OpenBSD [rpki-client]

\* Routinator [routinator]

7. Acknowledgements

The authors would like to thank Theo Buehler, Ben Maddison, Rob

Austein, Tim Bruijnzeels, and Russ Housley for their review and

feedback on this document.

8. References

8.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate

Requirement Levels", BCP 14, RFC 2119,

DOI 10.17487/RFC2119, March 1997,

<https://www.rfc-editor.org/info/rfc2119>.

[RFC4387] Gutmann, P., Ed., "Internet X.509 Public Key

Infrastructure Operational Protocols: Certificate Store

Access via HTTP", RFC 4387, DOI 10.17487/RFC4387, February

2006, <https://www.rfc-editor.org/info/rfc4387>.

[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>.

[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,

<https://www.rfc-editor.org/info/rfc6488>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC

2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174,

May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[RFC8182] Bruijnzeels, T., Muravskiy, O., Weber, B., and R. Austein,

"The RPKI Repository Delta Protocol (RRDP)", RFC 8182,

DOI 10.17487/RFC8182, July 2017,

<https://www.rfc-editor.org/info/rfc8182>.

[RFC9286] Austein, R., Huston, G., Kent, S., and M. Lepinski,

"Manifests for the Resource Public Key Infrastructure

(RPKI)", RFC 9286, DOI 10.17487/RFC9286, June 2022,

<https://www.rfc-editor.org/info/rfc9286>.

8.2. Informative References

[I-D.ietf-sidrops-rpki-crl-numbers]

Snijders, J., Maddison, B., and T. Buehler, "Handling of

Resource Public Key Infrastructure (RPKI) Certificate

Revocation List (CRL) Number Extensions", Work in

Progress, Internet-Draft, draft-ietf-sidrops-rpki-crl-

numbers-05, 22 May 2025,

<https://datatracker.ietf.org/doc/html/draft-ietf-sidrops-

rpki-crl-numbers-05>.

[RFC1982] Elz, R. and R. Bush, "Serial Number Arithmetic", RFC 1982,

DOI 10.17487/RFC1982, August 1996,

<https://www.rfc-editor.org/info/rfc1982>.

[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>.

[RFC6480] Lepinski, M. and S. Kent, "An Infrastructure to Support

Secure Internet Routing", RFC 6480, DOI 10.17487/RFC6480,

February 2012, <https://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,

<https://www.rfc-editor.org/info/rfc6481>.

[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,

<https://www.rfc-editor.org/info/rfc6489>.

[RFC7942] Sheffer, Y. and A. Farrel, "Improving Awareness of Running

Code: The Implementation Status Section", BCP 205,

RFC 7942, DOI 10.17487/RFC7942, July 2016,

<https://www.rfc-editor.org/info/rfc7942>.

[RFC8488] Muravskiy, O. and T. Bruijnzeels, "RIPE NCC's

Implementation of Resource Public Key Infrastructure

(RPKI) Certificate Tree Validation", RFC 8488,

DOI 10.17487/RFC8488, December 2018,

<https://www.rfc-editor.org/info/rfc8488>.

[RFC8630] Huston, G., Weiler, S., Michaelson, G., Kent, S., and T.

Bruijnzeels, "Resource Public Key Infrastructure (RPKI)

Trust Anchor Locator", RFC 8630, DOI 10.17487/RFC8630,

August 2019, <https://www.rfc-editor.org/info/rfc8630>.

[routinator]

NLnet Labs, "Routinator", June 2024,

<https://www.nlnetlabs.nl/projects/routing/routinator/>.

[rpki-client]

OpenBSD Project, "rpki-client", January 2024,

<https://www.rpki-client.org/>.

Appendix A. Serial Number Arithmetic

Serial number arithmetic [RFC1982] is an approach that has been used

in the DNS context (among others) to permit the indefinite use of a

finite number space. At least in theory, it would be possible to use

a similar approach with the manifestNumber field as well.

However, unlike the corresponding DNS context with Start of Authority

(SOA) resource records, an RPKI CA does not have visibility into or

control over RPKI RPs generally. This means that it is not possible

to select an updated manifestNumber value or to manage the relevant

state transitions so as to guarantee that all RPs will have valid

state at the end of the process. The approach proposed in Section 2

does not have this problem.

Appendix B. Manifest thisUpdate

The thisUpdate field in the manifest object is of type

GeneralizedTime, defined in Section 4.1.2.5.2 of [RFC5280]. This

type has a maximum value of 99991231235959Z (i.e. 31 December 9999

23:59:59 GMT). Section 4.2.1 of [RFC9286] requires that "[e]ach RP

MUST verify that this field value is greater (more recent) than the

most recent manifest it has validated", so it would appear to be

subject to the same problem as for manifest numbers. However, during

validation, if the RP detects that the current time is not between

the manifest thisUpdate and nextUpdate values, the RP must treat the

fetch as a failed fetch. Therefore, the RP will not cache a manifest

with a current date far in the future, and the CA can rectify the

problem here by reissuing the relevant manifest with the correct

date.

Authors' Addresses

Tom Harrison

Asia Pacific Network Information Centre

6 Cordelia St

South Brisbane QLD 4101

Australia

Email: tomh@apnic.net

George G. Michaelson

Asia-Pacific Network Information Centre

6 Cordelia St

South Brisbane QLD 4101

Australia

Email: ggm@apnic.net

Job Snijders

Amsterdam

Netherlands

Email: job@sobornost.net