Internet Engineering Task Force
Internet-Draft
Updates: 9286 (if approved)
Intended status: Standards Track
Expires: 19 February 2026

T. Harrison
G. Michaelson
APNIC
J. Snijders

18 August 2025

Resource Public Key Infrastructure (RPKI) Manifest Number Handling draft-ietf-sidrops-manifest-numbers-08

Abstract

The Resource Public Key Infrastructure (RPKI) makes use of signed objects called manifests, each of which includes a "manifest number". This document updates RFC9286 by specifying issuer and RP behaviour when a manifest number reaches the largest possible value, a situation not considered in RFC9286.

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 19 February 2026.

Copyright Notice

Table of Contents

1. Introduction

The Resource Public Key Infrastructure (RPKI) [RFC6480] makes use of signed objects [RFC6488] called manifests [RFC9286]. A manifest lists each file that an issuer 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 an issuer 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.

While it is practically impossible for an issuer to reach the largest possible value under normal operating conditions (it would require that the issuer 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. For example:

* 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 issuer 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 issuer 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 issuers 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. This is an update to the requirements set out in Section 4.2.1 of [RFC9286].

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 were 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 such a change is necessary to address the invalidity problem described in this document. Similarly, an RP MUST alert its operators when a manifest filename changes for a given CA.

[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 this section 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.

3. Manifest Filenames

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 for the purposes of this document.

The corollary of the behaviour defined in the previous paragraph 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. If one of the manifest filenames from the previous CA certificate appears in the newly-issued CA certificate, then an RP that is using that manifest filename for comparisons will determine that the manifest filename for the CA has not changed, and will therefore not reset its stored manifestNumber for the CA.

4. Manifest SIA Verification

To avoid certain forms of replay attack, RPs MUST verify that the URI in the accessLocation in one of the id-ad-signedObject accessMethod instances 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. If this verification check is unsuccessful, then the fetch has failed, and the RP MUST proceed per Section 6.6 of [RFC9286].

5. RFC8488 Comparison

Section 3.2.1 of [RFC8488] describes a manifest selection approach for RPs that involves collecting all unexpired, valid manifests for a CA, and then selecting from that collection the manifest that has the highest manifestNumber. The approach set out in this document is different from that approach.

6. General Repository Handling

Section 2 contains a specific update to [RFC9286] for the handling of manifest numbers, in order 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 recognises 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.

7. 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.

8. 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]).

See Section 2, Paragraph 3 regarding the effect of skipping the manifestNumber check with respect to replay attacks. To protect against replay attacks in the absence of this check, RPs should ensure that they are verifying the thisUpdate value per the requirements of [RFC9286].

Section 4 describes an additional protection against certain forms of replay attack.

Although this document updates [RFC9286], the security considerations from [RFC9286] remain relevant.

9. IANA Considerations

This document has no actions for IANA.

10. 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".

10.1. rpki-client

* Responsible Organization: OpenBSD
* Location: https://www.rpki-client.org
* Description: This implementation supports the behaviour described in this document.
* Level of Maturity: This is a production implementation.
* Coverage: This implementation includes all of the features described in this specification.
* Contact Information: Job Snijders, job@sobornost.net

10.2. Routinator

* Responsible Organization: NLNetLabs
* Location: https://github.com/NLnetLabs/routinator
* Description: This implementation supports the behaviour described in this document.
* Level of Maturity: This is a production implementation.
* Coverage: This implementation supports the manifest number handling changes described in this specification.
* Contact Information: NLNetLabs, labs@nlnetlabs.nl

11. Acknowledgements

The authors would like to thank Theo Buehler, Ben Maddison, Rob Austein, Tim Bruijnzeels, Russ Housley, Mohamed Boucadair, Luigi Iannone, Daniele Ceccarelli, Darren Dukes, Barry Leiba, テ詠ic Vyncke, Gorry Fairhurst, and Andy Newton for their review and feedback on this document.

12. References

12.1. 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>.

[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>.

12.2. Informative References

[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>.

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