sidrops
Internet-Draft
Intended status: Informational
Expires: 1 March 2026

D. Li
Tsinghua University
L. Chen
Y. Su
Zhongguancun Laboratory
Y. Fu
China Unicom

28 August 2025

RPKI Repository Problem Statement and Analysis
draft-li-sidrops-rpki-repository-problem-statement-02

Abstract

With the increasing deployment of Route Origin Authorizations (ROAs) and Route Origin Validation (ROV), the Resource Public Key Infrastructure (RPKI) has become a critical component for securing inter-domain routing. RPKI leverages cryptographic certificates to validate the authenticity and authorization of IP address and AS number allocations. All RPKI objects are stored in distributed repositories and periodically retrieved by relying parties (RPs). This document presents a data-driven analysis of the current RPKI repository system, including a global survey of AS administrators and an empirical evaluation of repository operations. The analysis reveals that the existing repository architecture suffers from limited scalability and suboptimal efficiency, and is highly sensitive to failures. As the number of ROAs and RPs increases, the growing number of point-to-point connections between repositories and RPs, coupled with the pull-based data synchronization model, significantly degrades the system’s scalability and performance. Moreover, a failure or attack on any Publication Point (PP) can prevent RPs from obtaining a complete and up-to-date view of RPKI data. This document further outlines the fundamental requirements for building a scalable, resilient,and efficient RPKI repository architecture to address these limitations.

Copyright Notice

Table of Contents

1. Introduction

To establish a trustworthy mapping between AS numbers and IP prefixes, RPKI arranges the Certificate Authorities (CAs) in a hierarchy that mirrors IP address allocation, and five RIRs are trust anchors. Each CA in RPKI holds a Resource Certificate (RC) issued by its parent authority, which attests to a binding of the entity's public key to a set of allocated Internet Number Resources (INRs, such as IP address blocks and AS numbers). CAs can issue subordinate RCs to reallocate their resources or issue ROAs (leaf nodes) to authorize ASes to originate specific IP prefixes.

As shown in Figure 1, each CA in RPKI will upload the objects it signs, such as manifests, CRLs, RCs, and ROAs, to the repository Publication Point (PP) it specifies. These PPs naturally form a tree structure that mirrors the RPKI certificate tree and collectively form the global RPKI Repository. Relying Parties (RPs), as entities that help ASes request RPKI data, periodically traverse RCs from five root RCs (held by RIRs) and access the PPs specified in their Subject Information Access (SIA) fields to fetch all RPKI objects, and then validate them. Finally, the ASes served by the RPs can use the verified ROAs to guide routers to perform ROV.

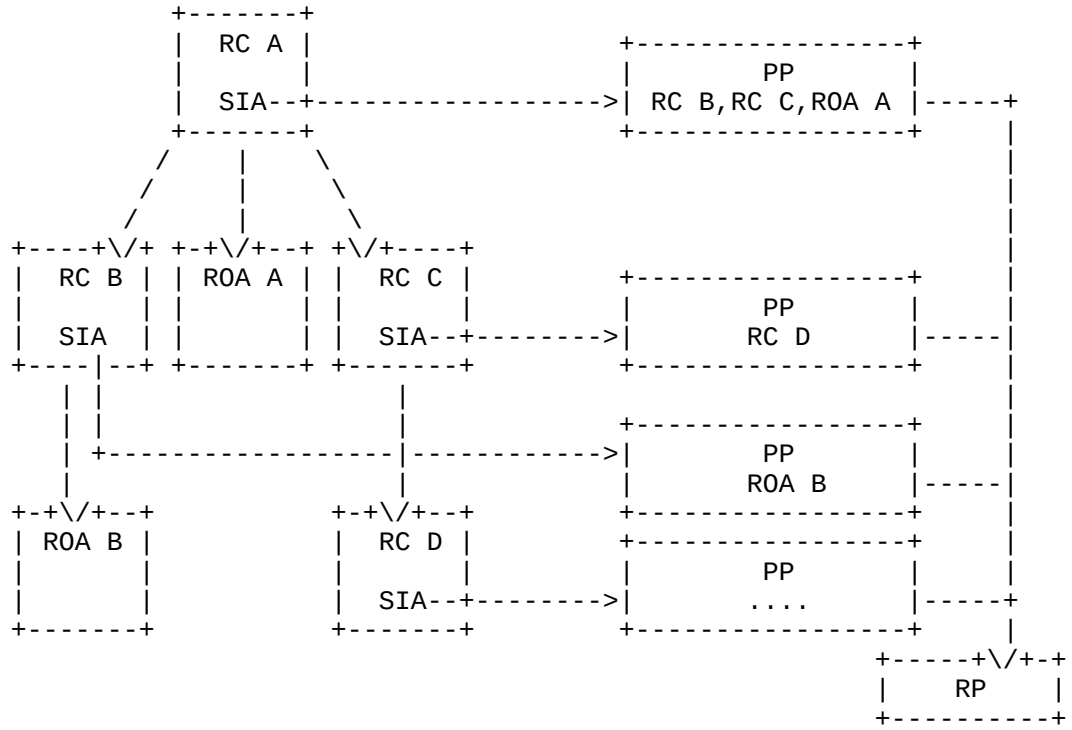

Figure 1: The RPKI certificate tree and RPKI Repository structure.

With the increasing deployment of RPKI, as of June 2024, ROAs cover 42.40% of active IPv4 address space and 57.97% of active IPv6 address space. Meanwhile, the number of independent repository instances has grown to 63. This document presents a measurement-based analysis of the reliability and scalability of the current RPKI repository architecture. Our study includes a large-scale survey of AS administrators from 2,500 randomly selected ROA-enabled ASes, along with an empirical assessment of repository deployment and operational status. The results indicate that the current architecture lacks the capability to provide reliable and resilient data access to RPs. Specifically, the RPKI system mandates that RPKI objects issued by a Certification Authority (CA) must be hosted at the PP operated by the same CA. As a result, any failure or unavailability of a PP can prevent RPs from accessing complete and up-to-date RPKI data. As delegated RPKI becomes more widespread, an increasing number of CAs are operating their own repository instances. However, many of these instances lack secure, highly available infrastructure, further impacting system reliability. In addition, the growing number of repository instances poses scalability and efficiency challenges: RPs are required to proactively traverse all known PPs and retrieve RPKI data in order to maintain and refresh their local caches.

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.

1.2. Terminology

It is assumed that the reader is familiar with the terms and concepts described in "A Profile for Resource Certificate Repository Structure" [RFC6481], "The RPKI Repository Delta Protocol (RRDP)" [RFC8182], and "An Infrastructure to Support Secure Internet Routing" [RFC6480].

2. Scalability and Efficient Analysis

In current RPKI infrastructure, refreshing the local cache of each RP involves traversing all repositories to fetch the updated RPKI objects. However, RPKI Repository has grown from 5 repositories run by five RIRs to more than 60 repositories and is expected to increase dramatically with the further deployment of ROA.

On the one hand, with a deeper understanding of RPKI, ROA deployers prefer to adopt delegated RPKI to flexibly control their RPKI objects. In the survey, 45.3% of the AS administrators using hosted RPKI say they have plans to run their own repositories for flexible certificate signing and control. The result shows that delegated RPKI will emerge as a trend, and the number of independent repositories will inevitably increase. The work [beyond-limits] predicts that when ROA is fully deployed, the number of repositories will reach 10K and there will be 140K active RPs, the RPKI snapshot download would easily exceed 1 hour (on the premise that each repository is well accessible). Since RPs are required to check the updates of all repositories when refreshing their local caches, the increasing number of repositories will result in higher refresh costs. It may prevent RPs from obtaining routing information in a timely manner, and also prohibits the release of use cases such as temporary ROAs to enable ISPs to perform tactical traffic engineering to ease congestion.

On the other hand, RPKI Repository is designed without a strict admission mechanism, allowing entities to apply for RC from RIRs or other RPKI CA entities, register as delegated CAs, and then operate repositories with a small fee and identity verification. Therefore, some repositories for unwanted purposes (measurement, attack experiments, etc.) are gradually emerging. Furthermore, the low barriers to running repositories and joining RPKI Repository will introduce unforeseen risks to RPs. For example, a malicious CA can create a large number of descendant RCs and operate numerous repositories to make RPs endlessly retrieve repositories, thus exhausting and paralyzing RPs. Although the most popular RP software, Routinator [routinator], has set the default value for the maximum searchable RPKI-tree depth to 32 in its latest version 0.14.1, and the RPKI-client has set the default value to 12, neither has provided a value for RPKI-tree width (as it is difficult to set a reasonable threshold). Similarly, it is also challenging for RP softwares to clearly limit the maximum number of accessible repositories. In addition, a series of recent academic works have emerged that exploit repository vulnerabilities to cause repository downtime, thereby preventing it from providing normal services to RPs [behind-the-scenes]. There are also works [beyond-limits][stalloris] that manipulate malicious repositories to attack RPs, thus obstructing the synchronization of RPKI data.

In addition, with the increasing rate of ROV deployment, more and more RPs will connect to RPKI Repository. It will undoubtedly increase the burden on the repositories. In summary, as RPKI deployment progresses further, RPs will need to access more repositories, and repositories will need to serve more RPs. This increase in the number of bidirectional connections and pull-based data synchronization mode will threaten the scalability and efficiency of the RPKI system.

3. Reliability Analysis

Although the current RPKI Repository is globally distributed, each CA only stores the RPKI objects it issues in the unique PP it runs. If any PP fails (malfunctions or is attacked), RPs will not be able to fetch the RPKI objects issued by its CA, and the integrity of the global RPKI object view cannot be guaranteed.

As of August, 2024, there are 42,950 RCs and 303,918 ROAs in RPKI Repository. For each RC, we extracts the SIA field that records the URI of its HTTPS-based RRDP file (Since all PPs now support RRDP and serve RPs through RRDP files, the analysis of the RPKI Repository focuses primarily on RRDP Repository). The result shows that there are currently 63 independent repository instances (SIA fields of RCs held by the same entity may share the same PP, therefore, the number of repositories is much lower than that of RCs).

Next, the measurement focuses on the ASes where the repositories are located, their IP addresses, and whether they utilize CDNs. We uses 2,000 globally distributed DNS resolvers to resolve repositories' domain names, with the aim of finding the CNAME and all IP address records for each repository. Then, we use the IP-to-AS mapping information maintained by RIPE NCC [routing-history] to obtain the AS where each PP is located.

Typically, CDN service providers will display their information in the domain name and CNAME record, such as including "cdn.cloudflare.net" or specify the CDN in the X-Via-CDN field, cache status field, or Server field in the HTTPS headers. In addition, if CDN acceleration is used, the latency of requesting HTTPS services from different geographic locations around the world will be relatively low. Therefore, it can determine whether the PP services are hosted on CDNs from the following aspects: (1) Whether the domain name and CNAME record contain information about CDN service providers; (2) The number of IP addresses returned by DNS resolvers and the geographical distribution of these IP addresses; (3) Whether the HTTPS request headers contain information about mainstream CDN providers; (4) The latency of accessing RRDP files from different geographic locations around the world.

Measurement results show that although RRDP enables the use of Content Distribution Networks (CDNs) infrastructure for resilient service, only 9 out of 63 repositories are hosted in CDNs, with 8 being hosted on Cloudflare AS13335 and 1 on Amazon AS16509. It also shows that out of 63 repositories, 60 of them are hosted in a single AS. It means that the accessibility of these repositories is highly dependent on the reachability of a single AS. Worse still, among the repositories whose corresponding ASes have deployed ROAs, there are 14 of them carry the ROA of the ASes in which they are located. It means that once the repository goes down and the RPs cannot fetch its ROAs, the route of the AS that the repository locates in may be downgraded by ROV adopters. Then, even if the repository is restored, those ROV adopters still cannot access the repository, in which case the access of the repository depends on the reachability of the AS it locates in, while the reachability of the AS also depends on the access of the repository.

Real-world incidents of repository breakdowns do occur at times. On 6 April 2020, the repository maintained by RIPE NCC suffered a sudden increase in connection to service [RIPE-downtime], resulting in it appearing as down to many RPs for 7 hours (RIPE’s services are already hosted on CDNs). On 15 May 2020, the Japan operated repository was out of service for 10 hours due to hardware failure [service-outage], and between 26 Jan 2022 and 2 Feb 2022, due to full disk space, all ROAs in its repository again became invalid [disk-outage]. During 10 July 2024 to 12 July 2024, we also found that RPKI data synchronized through Routinator [routinator] once again had missing parts from RIPE NCC.

4. Summary: Problems of the current RPKI Repository

Based on the measurement and analysis described in the previous section, this section summarizes the main problems of the current RPKI Repository.

4.1. P1: RPKI Repository is costly in RP refreshing.

Refreshing the local cache of each RP involves traversing all repositories to fetch the updated RPKI objects. However, the binding of the repository PP to the CA causes the number of repository instances to increase dramatically with the further deployment of ROA (as the number of CAs that hold RCs increase). Furthermore, as AS administrators gain a deeper understanding of RPKI technology, delegated RPKI has emerged as a trend, CAs are more willing to maintain the repository PP themselves to achieve more flexible certificate signing and management. The growth in the number of repository instances increases the cost of RP refreshes, the growth in the number of RPs brings burdens to repositories, and both will

threaten the scalability of RPKI.

4.2. P2: Every RPKI object is a singleton in RPKI Repository.

Although the current RPKI Repository is globally distributed, RPKI does not provide distributed storage for RPKI objects. The binding of PP and CA causes each RPKI objects to be stored only in the PP run by the CA that issued the object, instead of being stored in multiple repository nodes. Therefore, the current RPKI Repository cannot guarantee that RP can obtain complete RPKI data when some repository nodes fail. Even worse, the singleton nature of RPKI objects also introduces unwanted interdependence between the accessibility of a CA's PP and the reachability of the AS that the PP locates. Since RPKI ensures the routing security and reachability of ASes, it is fair to expect PPs to remain accessible during any incident when corresponding ASes become unreachable.

5. New Requirements for RPKI Repository

The following requirements are identified for a reliable, scalable and secure RPKI Repository architecture.

5.1. Requirement 1: Push-based Data Synchronization Mode

A push-based data synchronization mode is needed for RPKI Repository to efficiently serve tens of thousands of RPs in a timely and scalable manner.

In the current pull-based model, RPs periodically fetch RPKI data from all repository PPs, which introduces unnecessary overhead and synchronization delays. In contrast, push-based synchronization allows repository nodes or intermediate proxies to proactively send updated RPKI objects to RPs once new data becomes available. This significantly reduces the synchronization interval, avoids redundant polling connections, and ensures that routing infrastructure can respond to critical changes (e.g., ROA issuance or revocation) with minimal delay. Moreover, timely RPKI update delivery is essential for ensuring routing correctness and enabling responsive policy enforcement, such as temporary ROA deployment for traffic engineering or emergency response. When RPs depend solely on periodic polling, such timely updates cannot be guaranteed, potentially leading to stale routing decisions. Therefore, adopting a push-based model helps decouple update propagation from RP polling schedules, enabling RPKI to better support high-speed and dynamic inter-domain routing environments.

5.2. Requirement 2: Truly Distributed Storage

A truly distributed storage model is needed for RPKI Repository to provide reliable services for tens of thousands of RPs.

It means that Resource Certificate (RC) and ROAs are no longer only stored at the repository operated by the CA that issued them, but can be stored at multiple RPKI Repository nodes (PP or other names). The truly distributed storage architecture breaks the singleton nature of RPKI objects. It ensures that any single point of failure in the RPKI Repository nodes will not affect the integrity of RPKI snapshot retrieval by RPs. Additionally, since RPKI Repository nodes are located in different ASes and a single RPKI object can be stored on multiple nodes, the unwanted interdependence between the accessibility of a PP's objects and the reachability of the PP's AS will be broken. Since RPKI ensures the routing security and reachability of ASes, it is fair to expect RPKI objects to remain accessible during any incident when the Repository node or its corresponding AS become unreachable.

5.3. Requirement 3: Admission Mechanism

An admission mechanism is needed to effectively limit the unconstrained expansion of RPKI repository instances and enhance the scalability of RPKI infrastructure.

As AS administrators gain a deeper understanding of RPKI technology, delegated RPKI has emerged as a trend, the number of repository instances has grown more than 12 times in the past five years. The rapid growth in the number of repositories increases the cost of RP refreshes and threatens the scalability of RPKI. Furthermore, since each repositories connects to all RPs in the world, the low barriers to running repositories and joining RPKI Repository will introduce unforeseen risks to RPs because some repositories may emerge with unexpected intentions. Regardless, RPs should ensure they establish connections with trusted and reliable nodes. Therefore, a secure and scalable RPKI Repository needs an admission mechanism to raise the threshold for operating repository nodes. For example, the addition of a new repository node should be reviewed by existing node members or other trusted entities. The review can include verifying node's real identity, the reputation of the node operator, and whether it can provide a robust, secure, and reliable service to RPs, among other factors.

To implement an admission mechanism, it may be necessary to first implement "Decoupled from RPKI Authorities", because once repository nodes are bound to CAs, each CA has the right or need to operate its own repository.

5.4. Requirement 4: Be Compatible with Current RPKI

Since the current RPKI is mature and widely deployed, the new RPKI Repository should not modify the RPKI hierarchical certificate issuance architecture and should not affect the existing RPKI certificate validation or the ROV process. It also should be compatible with current RPKI Repository architecture and supports incremental deployment.

6. Ethical Considerations

This section will outline the ethical considerations regarding the RPKI Repository measurement and the worldwide survey. First, we strictly limited the rate and number of DNS resolving packets (sending 130,000 packets over 24 hours) to avoid causing much burden on the Internet. For IP-to-AS mapping, we used open-source data provided by RIPE NCC. We conducted access experiments on the RPKI Repository from six globally distributed probes (India, Dubai, Silicon Valley, Frankfurt, Beijing, and S達o Paulo), with each RPKI repository being accessed fewer than 10 times to avoid DDoS attacks on the repositories. For the survey, we obtained the contact information of AS administrators from open-source WHOIS mailing lists and provided an option for administrators to opt out of the survey. We did not disclose any additional information about the participating administrators or their feedback and ensured that their responses would be used solely for research.

7. Security Considerations

TBD

8. IANA Considerations

This document has no IANA requirements.

9. References

9.1. Normative References

[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/rfc/rfc6481>.

[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/rfc/rfc8182>.

[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/rfc/rfc6480>.

[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/rfc/rfc2119>.

[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/rfc/rfc8174>.

9.2. Informative References

[routing-history] RIPE NCC, "routing history", 2024, <https://stat.ripe.net/docs/02.data-api/routing-history.html>.

[service-outage] JAPNIC, "Service outage Roaweb and rpki repository (resolved)", 2020, <https://www.nic.ad.jp/en/topics/2020/20200521-01.html>.

[disk-outage] JAPNIC, "Service outage Disk full caused lost roa validity", 2022, <https://www.nic.ad.jp/en/topics/2022/20220202-01.html>.

[routinator] NLnet Labs, "Routinator", 2021, <https://github.com/NLnetLabs/routinator>.

[beyond-limits] "Beyond Limits How to Disable Validators in Secure Networks", 2023, <https://dl.acm.org/doi/abs/10.1145/3603269.3604861>.

[stalloris] "Stalloris RPKI Downgrade Attack", 2022, <https://www.usenix.org/system/files/sec22fall_hlavacek.pdf>.

[behind-the-scenes] "Behind the Scenes of RPKI", 2022, <https://dl.acm.org/doi/pdf/10.1145/3548606.3560645>.

[RIPE-downtime]
RIPE NCC, "Rsync rpki repository", 2022,
<https://www.ripe.net/ support/service-announcements/
rsync-rpki-repository-downtime>.

Authors' Addresses

Dan Li
Tsinghua University
Beijing
China
Email: tolidan@tsinghua.edu.cn

Li Chen
Zhongguancun Laboratory
Beijing
China
Email: lichen@zgclab.edu.cn

Yingying Su
Zhongguancun Laboratory
Beijing
China
Email: suyy@mail.zgclab.edu.cn

Yu Fu
China Unicom
Beijing
China
Email: fuy186@chinaunicom.cn