SIDROPS
Internet-Draft
Intended status: Informational
Expires: 19 September 2026

L. Liu
Zhongguancun Laboratory
Z. Yan
CNNIC
L. Chen
Zhongguancun Laboratory
D. Li
Tsinghua University

18 March 2026

RPKI Repository Delta Protocol (RRDP) Delta File Retention Policy
draft-liu-sidrops-rrdp-delta-retention-policy-02

Abstract

This document updates RFC 8182 (The RPKI Repository Delta Protocol) by specifying an optimized delta file retention policy based on client access patterns. The proposed mechanism allows RRDP servers to maintain only the delta files required by active clients, reducing storage requirements while maintaining compatibility with existing clients. By tracking which serial numbers are being requested by active clients, the repository can determine the minimum serial number needed by any client and safely prune delta files that update from earlier serial numbers.

The proposed mechanism provides several benefits, including reduced storage requirements, smaller notification files, and more efficient use of bandwidth and processing resources. It also maintains backward compatibility with existing RRDP clients, requiring no changes to client implementations.

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 September 2026.

Table of Contents

1. Introduction

The RPKI Repository Delta Protocol (RRDP) [RFC8182] defines a mechanism for publishers to make available a set of current repository objects, and for relying parties to maintain a local copy of this repository that is periodically updated to match the published repository.

RFC 8182 specifies that repository must maintain a set of delta files that allow relying parties to update from any recent state to the current state. It proposes a size-based delta file retention strategy, stating that "Any older Delta Files that, when combined

with all more recent Delta Files, will result in the total size of deltas exceeding the size of the snapshot MUST be excluded to avoid that Relying Parties download more data than necessary.". However, as the number of existing RPKI objects grows and more are proposed, the snapshot file size increases significantly, requiring more delta files to be stored. This leads to higher storage costs and potential performance issues. Consequently, practical implementations have adopted various strategies to mitigate these issues, such as count-based limits (maintaining a fixed number of delta files) and time-based limits (retaining delta files only for a specified duration).

This document updates RFC 8182 by defining an adaptive delta file retention policy based on client access patterns. The key insight is that repositories need only maintain delta files that might be used by active relying parties. By tracking the minimum serial number accessed by active clients, repositories can safely prune older delta files that are no longer needed, while ensuring that all active clients can still perform incremental updates. This adaptive approach can be integrated with existing size-based delta file retention strategy [RFC8182] to provide a more comprehensive solution.

This approach provides several benefits:

1. Reduced storage requirements for RRDP servers;

2. Maintained backward compatibility with existing RRDP clients;

3. Potential performance improvements for notification file processing by limiting the size of notification file as suggested in Section 7.5 of [I-D.draft-ietf-sidrops-publication-server-bcp];

4. Better adaptation to actual client update patterns.

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. Background

This section reviews existing delta file retention strategies and illustrates their potential limitations.

2.1. Existing Delta File Retention Strategies

Section 3.3.2 of RFC 8182 mandates that servers limit the number of deltas in the notification file such that the combined size of the underlying delta files does not exceed that of the snapshot file. Servers are free to further limit the number of deltas included in the notification file, though. Two common strategies are:

1. *Count-based retention*: Maintaining a fixed number of most recent delta files (e.g., the last 500 delta files). This approach is simple to implement but does not account for varying delta file sizes or client access patterns, and may lead to frequently downloading the full snapshot for clients.

2. *Time-based retention*: Keeping delta files for a specified time period (e.g., 2 hours). This approach ensures clients that update regularly can use delta files, but may not be optimal for clients with irregular update schedules and may also lead to frequently downloading the full snapshot for clients.

These strategies are typically implemented as configuration options in RRDP server software, allowing repository operators to choose the approach that best fits their needs.

2.2. Challenges with Current Strategies

While the existing retention strategies provide some guidance, they have several limitations:

1. *Size-based retention*:

    * Does not consider client access patterns;

    * Can lead to inefficient storage use if many small delta files are retained when they are no longer needed. This issue may be exacerbated when the snapshot size grows along with the number increasement of existing objects and the addition of more new objects.

2. *Count-based retention*:

    * Arbitrary fixed limits may not match actual client needs;

    * Does not adapt to changing repository update frequencies;

    * May discard delta files that are still needed by active clients.

3. *Time-based retention*:

   * Does not account for varying client update frequencies and may retain too many files during periods of frequent updates or too few during periods of infrequent updates;

   * May retain files longer than necessary if no clients need them;

   * May discard delta files that are still needed by active clients.

These challenges highlight the need for a more adaptive approach to delta file retention that considers actual client behavior while maintaining compatibility with existing client implementations.

3. Adaptive Delta File Retention Policy

This document defines an adaptive delta file retention policy based on client access patterns. The key principle is that repositories need only maintain delta files that might be used by active relying parties. This approach would not like to replace the existing size-based retention strategy proposed in RFC 8182 and can be integrated with it to make RRDP work in a more efficient and adaptive manner.

The policy consists of three main components:

1. A client tracking mechanism that records the serial numbers accessed by clients;

2. A method for determining the minimum serial number needed by active clients;

3. An algorithm for pruning delta files that are no longer needed.

By implementing this policy, RRDP repositories can reduce storage requirements while ensuring that all active clients can perform incremental updates.

3.1. Client Tracking Mechanism

To implement the adaptive delta file retention policy, RRDP repositories MUST track the serial numbers accessed by clients. This can be accomplished by monitoring client requests for delta files.

When a client requests a delta file, it typically does so by first retrieving the notification file, identifying the appropriate delta file based on its current local serial number, and then requesting that specific delta file.

Repositories SHOULD maintain a data structure that records:

1. Client identifiers (e.g., IP addresses or anonymized identifiers);

2. The serial numbers requested by each client;

3. Timestamps of the most recent requests.

Figure 1 shows an example for the data structure to record RRDP clients.

Client ID	Serial Number	Last Access Time
Client A	42	March 17, 2026, at 12:00PM UTC
Client B	37	March 17, 2026, at 08:30AM UTC
Client C	45	March 17, 2026, at 14:15PM UTC

Figure 1: An example for the data structure to record RRDP clients.

Repositories MAY implement more sophisticated tracking mechanisms, such as:

* Using probabilistic data structures (e.g., Bloom filters) to efficiently track large numbers of clients;

* Implementing privacy-preserving techniques to avoid storing identifiable client information.

Repositories SHOULD periodically clean this data structure to remove entries for clients that have not been seen for a configurable period (e.g., 7 days). This helps ensure that the minimum serial number calculation is based only on active clients.

3.2. Minimum Serial Number Determination

Based on the client tracking data, repositories can determine the minimum serial number needed by active clients. This is the smallest serial number that any active client might need to update from.

The algorithm for determining the minimum serial number is as follows:

1. Initialize min_serial to the current repository serial number;

2. For each active client in the tracking data: a. If the client's serial number is less than min_serial, update min_serial to the client's serial number;

3. Return min_serial.

For example, using the client tracking data from the previous section:

* Current repository serial number: 50;

* Client A's serial number: 42;

* Client B's serial number: 37;

* Client C's serial number: 45;

* Minimum serial number: 37.

Repositories SHOULD recalculate the minimum serial number:

* Whenever a new delta file is created;

* Periodically (e.g., daily) to account for client tracking data cleanup;

* Before any delta file pruning operation.

Repositories MAY implement a safety margin by subtracting a small value from the calculated minimum serial number. This helps ensure that clients that have recently become active but have not yet requested a delta file can still perform incremental updates.

3.3. Delta File Pruning Algorithm

Once the minimum serial number has been determined, repositories can safely prune delta files that are no longer needed. A delta file is no longer needed if it allows updating from a serial number that is less than the minimum serial number.

The algorithm for delta file pruning is as follows:

1. Determine the minimum serial number (min_serial) as described in Section 3.2;

2. For each delta file: a. Extract the "from" serial number (from_serial) from the delta file metadata; b. If from_serial < min_serial, mark the delta file for deletion;

3. Delete all marked delta files;

4. Update the notification file to remove references to the deleted delta files.

For example, if the minimum serial number is 37, delta files that update from serial numbers 1-36 can be safely deleted.

Repositories MAY implement the following safeguards:

* Never delete the most recent N (e.g., 5) delta files, regardless of the minimum serial number calculation;

* Maintain delta files for a minimum time period before considering them for deletion;

* Implement a gradual pruning strategy to avoid sudden changes in available delta files.

Publishers MUST ensure that after pruning, the notification file still contains at least one delta element, unless the current serial number is 1.

3.4. Integration with Existing Size-based Retention Policy

The adaptive delta file retention policy described in this document MUST be integrated with the size-based retention policy specified in [RFC8182].

The integration SHOULD be implemented as follows:

* First, determine the minimum set of delta files required by active clients using the adaptive policy;

* Then, if the total size of these delta files exceeds the snapshot file size, remove older delta files to ensure the total size of retained delta files doe not exceed the size of the snapshot file, as required by [RFC8182].

4. Implementation Considerations

4.1. Server Implementation

Repositories implementing the adaptive delta file retention policy SHOULD follow these guidelines:

1. Client Tracking:
    * Use efficient data structures for client tracking as described in Section 3.1;
    * Regularly clean up tracking data for inactive clients.
2. Configuration Options:
    * Allow configuration of the client inactivity threshold (default: 7 days);
    * Allow configuration of the safety margin for minimum serial number calculation (default: 5).
3. Operational Features:
    * Provide monitoring and alerting for delta file management;
    * Log delta file pruning operations;
    * Expose metrics on storage savings and client serial number distribution.
4. Fallback Mechanisms:
    * Implement mechanisms to recover if delta files are accidentally pruned too aggressively;
    * Consider maintaining an archive of pruned delta files that can be restored if needed.

Repositories MAY implement the following optimizations:

* Batch delta file pruning operations to reduce the frequency of notification file updates;
* Implement predictive algorithms to anticipate client behavior and adjust the safety margin accordingly.

4.2. Client Behavior

No changes to client behavior are required to benefit from the adaptive delta file retention policy. Existing RRDP clients will continue to function as specified in RFC 8182.

5. Privacy Considerations

The client tracking mechanism described in Section 3.1 involves collecting information about client behavior, which raises privacy concerns. Repositories implementing this mechanism SHOULD take steps to protect client privacy:

1. Anonymization:
    * Use anonymized identifiers instead of storing actual IP addresses;
    * Consider using techniques such as IP address hashing with a rotating salt;
    * Ensure that the anonymization method still allows tracking the same client across multiple requests.
2. Data Minimization:
    * Store only the minimum information needed (client identifier, serial number, timestamp);
    * Do not store additional information such as User-Agent strings or other HTTP headers unless necessary for debugging.
3. Data Retention:
    * Implement strict data retention policies;
    * Delete tracking data for clients that have not been seen for a configurable period (e.g., 7 days).

Repositories MAY implement more sophisticated privacy-preserving techniques, such as differential privacy or secure multi-party computation, to further protect client privacy while still benefiting from the adaptive delta file retention policy.

6. Security Considerations

The adaptive delta file retention policy introduces several security considerations:

1. Denial of Service:

   * An attacker could potentially manipulate the minimum serial number calculation described in Section 3.2 by repeatedly requesting delta files with very old serial numbers

   * To mitigate this, repositories SHOULD:

     - Implement rate limiting for delta file requests;

     - Use anomaly detection to identify suspicious patterns;

     - Consider weighting client importance based on factors such as reputation or update frequency;

     - Implement the safeguards described in Section 3.3 to provide additional protection.

2. Information Leakage:

   * The client tracking mechanism could potentially leak information about client update patterns;

   * Repositories SHOULD implement the privacy considerations described in Section 5 to mitigate this risk.

3. Consistency:

   * Inconsistent delta file availability across multiple servers could lead to confusion or errors;

   * Repositories SHOULD ensure consistent implementation across all servers.

4. Fallback Reliability:

   * Clients falling back to snapshot updates more frequently could increase load on servers and networks;

   * Repositories SHOULD monitor fallback frequency and adjust retention policy parameters accordingly.

These security considerations do not introduce new vulnerabilities beyond those already present in the RRDP protocol, but they should be carefully addressed in any implementation of the adaptive delta file retention policy.

7. IANA Considerations

This document has no IANA actions.

8. Contributors

Tom Harrison
APNIC
Email: tomh@apnic.net
Thank you very much for the constructive revision suggestions and sharing practical operation experience.

9. Normative References

[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>.

[I-D.draft-ietf-sidrops-publication-server-bcp] Bruijnzeels, T., de Kock, T., Hill, F., Harrison, T., and J. Snijders, "Best Practises for Operating Resource Public Key Infrastructure (RPKI) Publication Services", Work in Progress, Internet-Draft, draft-ietf-sidrops-publication-server-bcp-06, 2 March 2026, <https://datatracker.ietf.org/doc/html/draft-ietf-sidrops-publication-server-bcp-06>.

[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>.

Authors' Addresses

Libin Liu
Zhongguancun Laboratory
Beijing
China
Email: liulb@zgclab.edu.cn

Zhiwei Yan
CNNIC
Beijing
China
Email: yanzhiwei@cnnic.cn

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

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