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

Di Ma & Stephen Kent

IETF 99, Prague

Background (1/2)

- Requirements imposed on Relying Parties (RPs) are scattered throughout numerous (12) RFCs:
  - cert/CRL profile
  - ROA specification
  - Manifests specification
  - TAL specification, etc.
- This makes it hard for an implementer to be confident that he/she has addressed all of these generalized requirements.

Implementers need to know
what are those requirements on RP.

Background (2/2)

- Software engineering calls for how to segment the RP system into components with orthogonal functionalities, so that those components could be distributed across the operational timeline of the user.

- Taxonomy of generalized RP requirements is going to help have 'RP role' well framed.

Implementers need to know how to organize all of these requirements on RP.

Overview

- We propose to consolidate RP requirements in one document, with pointers to all the relevant RFCs.
  - This document provides a single reference point for requirements for RP software for use in the RPKI.
  - It cites requirements that appear in several RPKI RFCs, making them segmented with orthogonal functionalities in different sections.

What would this doc be like?

- To DEFINE the RP?
  - No. There is no standards language (e.g., MUST, SHOULD, MAY, ...) in this doc, as it is just POINTING to the docs that have the real requirements
- To collect data in one place?
  - Yes. This doc outlines the RP functions, summarizes them and then gives reference to those precise sections or paragraphs.
  - This document will be updated to reflect new or changed requirements as these RFCs are updated, or new RFCs are written.
- To provide implementation guidance?
  - This document could be a good place to provide guidance based on experience from RP implementers.

Outline (1/2)

- Fetching and Caching RPKI Repository Objects
  - TAL acquisition and processing
  - Locating RPKI objects using Authority and Subject Information Extensions
  - Dealing with Key Rollover
  - Dealing with algorithm transition
  - Strategies for efficient cache maintenance (implementation guidance?)

- Certificate and CRL Processing
  - Verifying Resource Certificate and Syntax
  - Certificate Path Validation
  - CRL Processing

Outline (2/2)

- Processing RPKI Repository Signed Objects
  - Basic signed object syntax checks
  - Syntax and validation for each type of signed object
    - Manifest
    - ROA
    - Ghostbusters
    - BGPsec router certificate
  - How to Make Use of Manifest Data (implementation guidance?)
  - What to Do with Ghostbusters Information (implementation guidance?)

- Delivering Validated Cache Data to BGP Speakers

Clarifications

- 1) We are not suggesting that implementers skip reading those RFCs in full.
  - Our draft is born to be a guide to help implementers get the essentials of RP functionalities scattered in different RFCs. Anyone who wants to comprehend RPKI cannot be exempted from reading all the RPKI RFCs, let alone the implementers.
  - One might see the RP requirements document as a "Manifest" for all necessary RP functions ☺
- 2) Implementers need to know more than what RP requirements are.
  - They need to know how to reflect these functions as they are making software design.
  - To that end, this draft has generalized RP requirements that are segmented with orthogonal functionalities in different sections.

sidrops-rpki-tree-validation vs. madi-sidrops-rp

- RPKI tree validation describes how the RIPE RP software works, while the RP requirements document is generic.

- The tree validation doc is expressly a description of one particular RP implementation. Thus it is an example of how that implementation tries to meet the RP requirements, not a general characterization of RP requirements.

It is appropriate to proceed with both documents.

QUESTIONS?