RPKI Retrieval Delta Protocol - RRDP

Tim Bruijnzeels
Oleg Muravskiy
Bryan Weber
Rob Austein
David Mandelberg

IETF92 | Dallas | 23 March 2015

Update

- Draft accepted as working group document

- Some questions were raised in the working group
  - Design goals
  - HTTP vs other transport protocols
  - Quick overview

- Some questions were discussed between the authors
  - Implications of CAs sharing a publication server

- Implementation updates
  - Proof of concept code working since Honolulu (rcynic, rpkid, RIPE NCC)
  - RIPE NCC (almost) implemented validation
  - RIPE NCC planning to work open-source Publication server

Design ideas 1/2

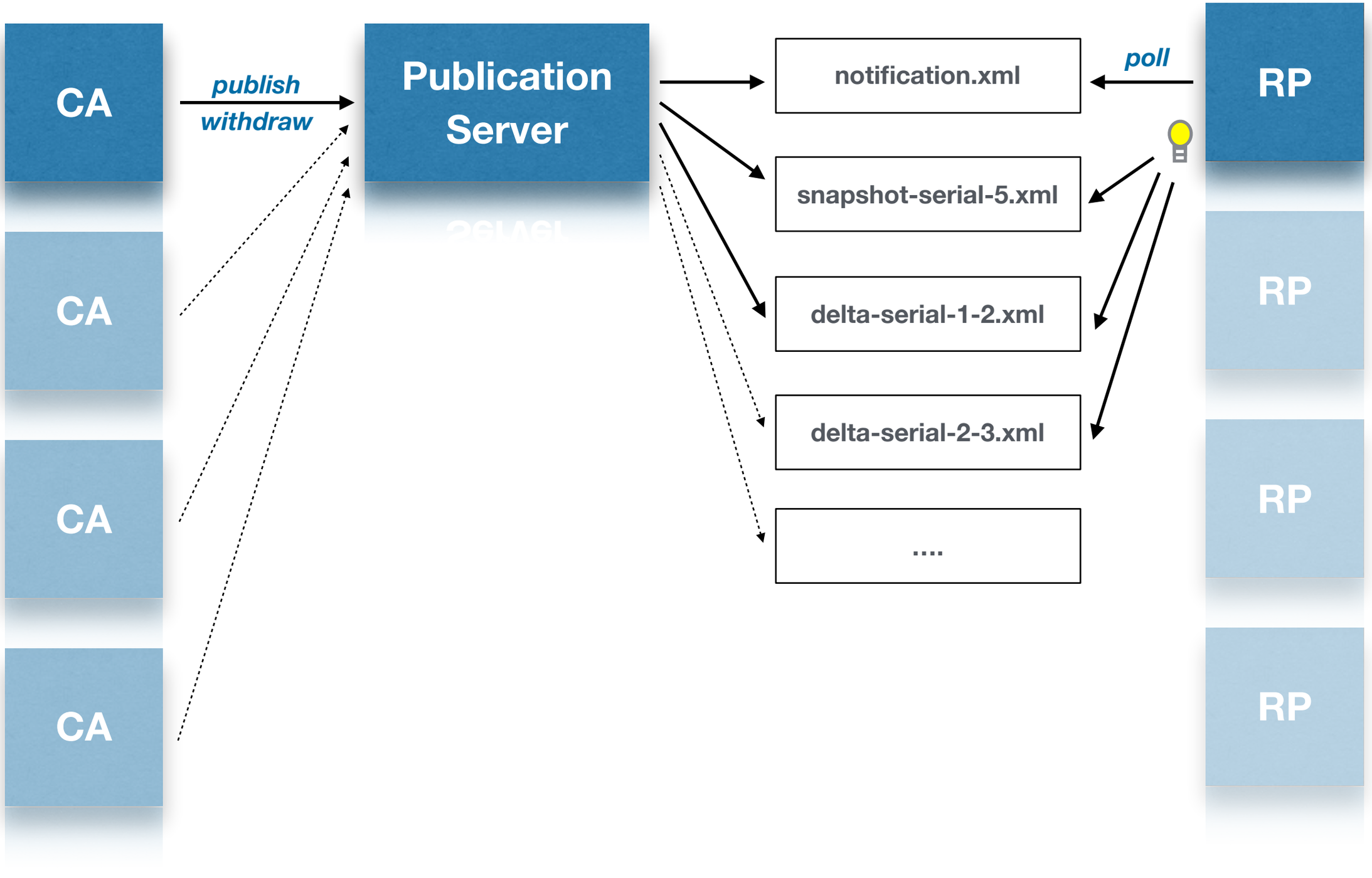
- Support publication servers
  - Many CAs share one publication server (compare recursive rsync)
  - Improve scaling: move expensive operations to the clients:
    - Server provides a stateless API, the client can choose what it needs to fetch
  - HTTP(S) as primary transport
    - Industry standard, many server and library implementations
    - Support scaling by use of existing global http based CDNs
    - Support scaling by use of local http caching servers (squid, apache, etc)

Design ideas 2/2

- Support Relying Parties
  - Many mature native http(s) client libraries
  - Minimise number of fetches and amount of data

- Avoid *unnecessary* Complexity
  - Minimal protocol, clearly defined and versioned (no svn, git, etc)
  - No big attempts to secure the fetch protocol: trust object security (just like rsync)
  - Messages based on existing publication protocol draft

Quick overview

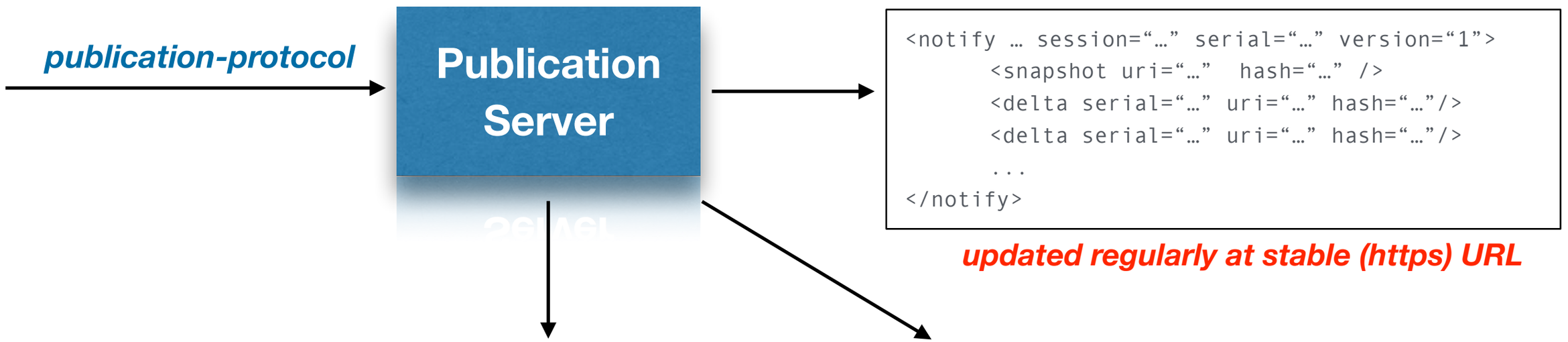


Quick overview - Publication

publication protocol

```
<messages>
      <publish uri="rsync://host/ca1/key.mft" hash="…">
            base64
      </publish>
      <publish uri="rsync://host/ca1/key.crl" hash="…">
            base64
      </publish>
      <publish uri="rsync://host/ca1/roa2.roa">
            base64
      </publish>
      <withdraw uri="rsync://host/ca1/roa1.roa" hash="…"/>

      ...

</messages>
```

Quick overview - RRDP snapshot / deltas

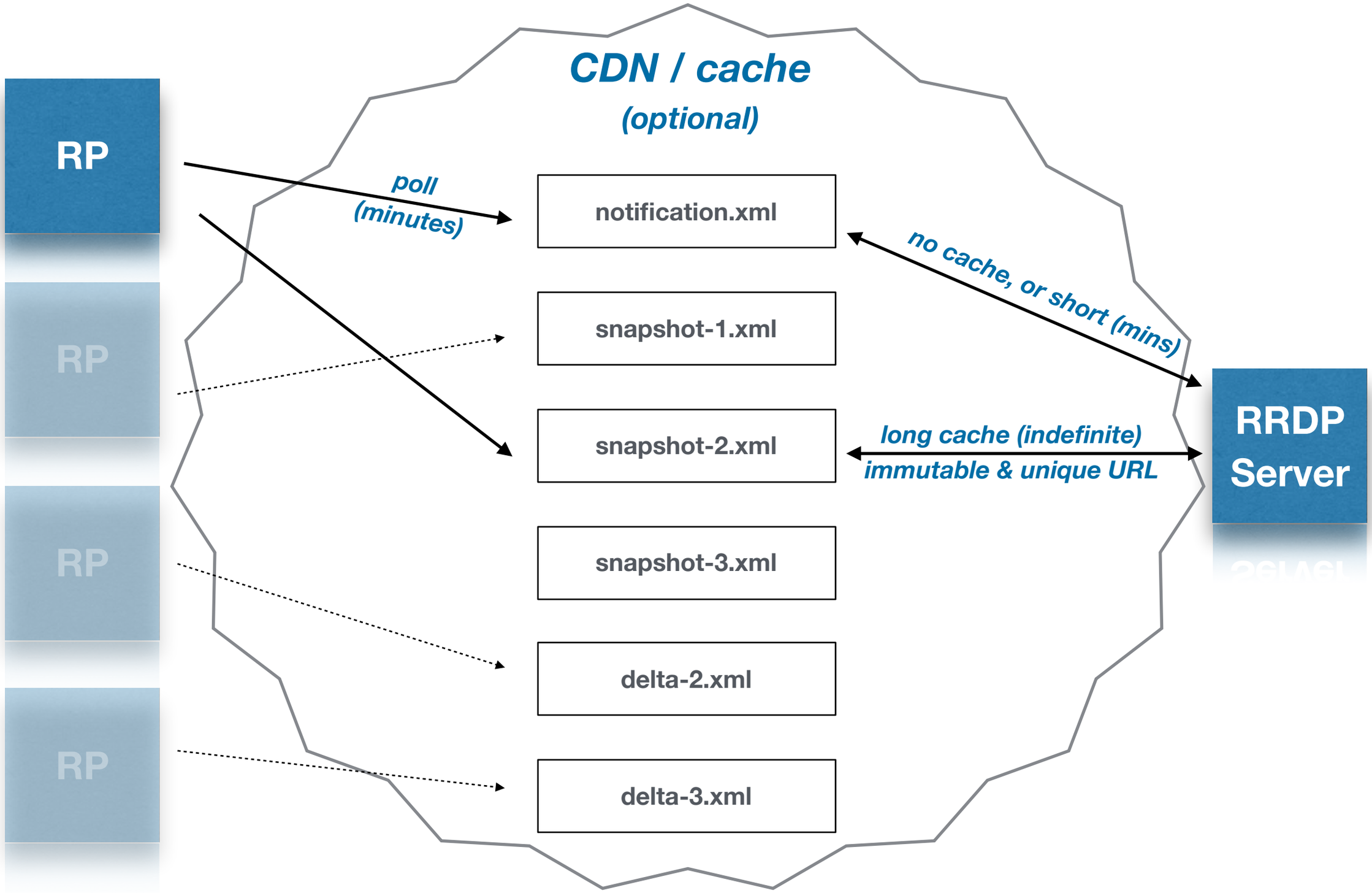



```
<delta session="…" serial="…" version="1">
      <publish uri="rsync://host/ca1/key.mft" hash="…">
            base64
      </publish>
      <publish uri="rsync://host/ca1/key.crl" hash="…">
            base64
      </publish>
      <publish uri="rsync://host/ca1/roa2.roa">
            base64
      </publish>
      <withdraw uri="rsync://host/ca1/roa1.roa" hash="…"/>

      ...

</delta>
```

```
<snapshot session="…" serial="…" version="1">
      <publish uri="rsync://host/ca1.cer">
            base64
      </publish>
      <publish uri="rsync://host/ca2.cer">
            base64
      </publish>
      <publish uri="rsync://host/ca1/roa2.roa">
            base64
      </publish>
      <publish uri="rsync://host/ca2/roa1.roa">
            base64
      </publish>
      ...
</snapshot>
```

Immutable data for a given session and version - published at unique (http) URLs to allow caching (CDN)

Quick overview - Validator polling / fetching



Implementation Update - Validation

- Proof of concept code works (RIPE NCC and rcynic)

- Production grade code almost done for RIPE NCC validator
  - Validation and retrieval separated, handles rsync and rrdp
  - Relying on key identifiers for discovery
  - Still solving issues regarding forgetting about old objects (if not on mft and revoked, or not on mft and expired?)
  - Plans to write an informational document describing full validation and retrieval algorithm

Implementation Update - Publication

- Proof of concept code works (RIPE NCC and rpkid)

- RIPE NCC planning to work on open-source publication server
  - First version using publication protocol semantics
  - But not the CMS just yet (can easily be added later)
  - Supporting publishing to both RRDP and rsync
    (transparent to CA as long as it includes the right URIs)
  - Aim to have running code by IETF93

Current discussions

- Implications of CAs sharing a publication server
  - Snapshots and deltas include objects issued by all CAs using this publication server
  - The SIA just points to notification.xml. How does an RP find the manifest and products signed by ***this*** CA certificate? And what about AIA and CRLDP?

- Will continue discussions and work
  - Comments on list welcome
  - Or talk to us during the week
  - Will publish new version when buffer fills up

Questions? Comments?