IETF sidrops updates

Routing Security SIG

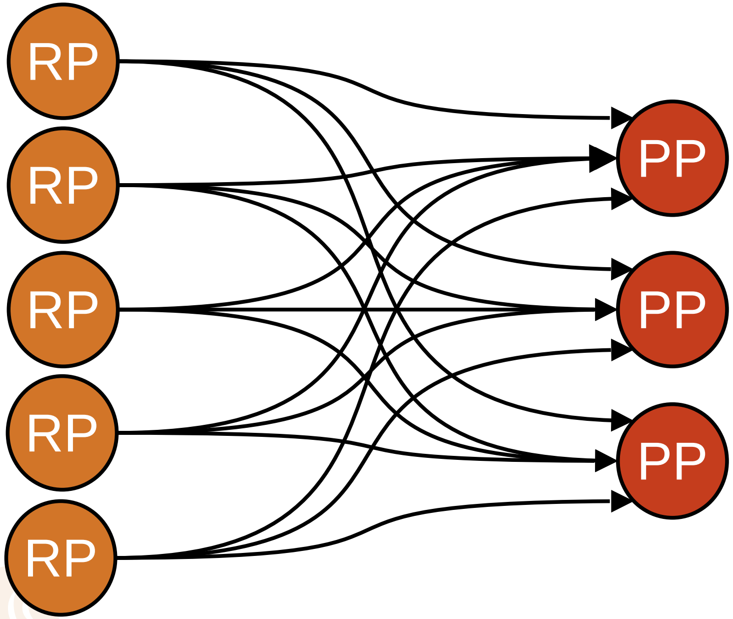
Topics

- Erik Synchronization Protocol
  - `draft-ietf-sidrops-rpki-erik-protocol`
- NRO Trust Anchor Constraints
  - `draft-nro-sidrops-ta-constraints`
- Publication Server Best Current Practices
  - `draft-ietf-sidrops-publication-server-bcp`

Erik Synchronization Protocol

- A new repository distribution protocol, for use in conjunction with existing rsync and RRDP servers
- Unlike existing servers, an Erik server (called an ‘Erik relay’) can be operated by anybody
- An Erik relay fetches RPKI content from one or more existing rsync/RRDP servers (**PP**s), and then makes that content available to relying parties (**RP**s) via the new protocol

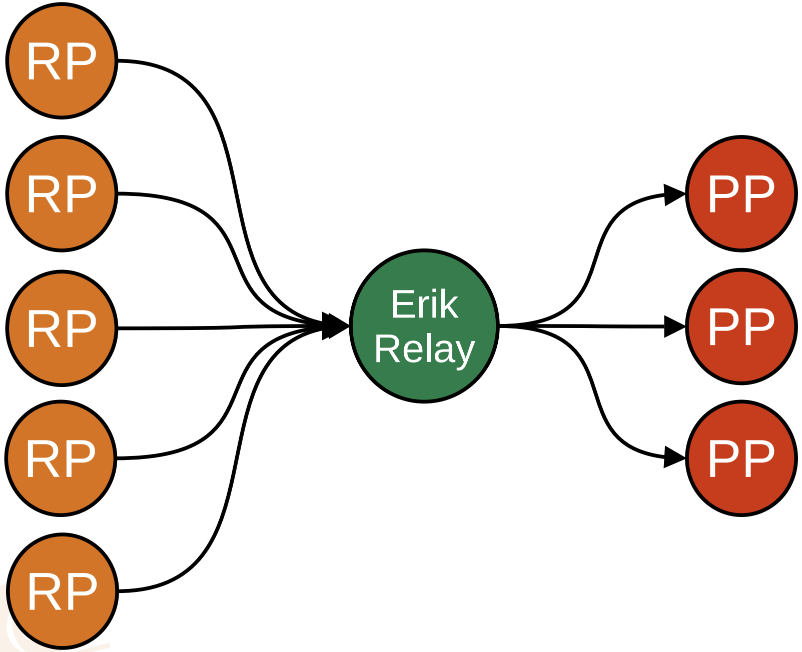
Problems with rsync/RRDP (1)



Each RP needs to fetch from each PP

- M:N problem that gets worse over time as more RPs are added
- Problems at one PP can hold up validation for an RP as a whole

Problems with rsync/RRDP (1)



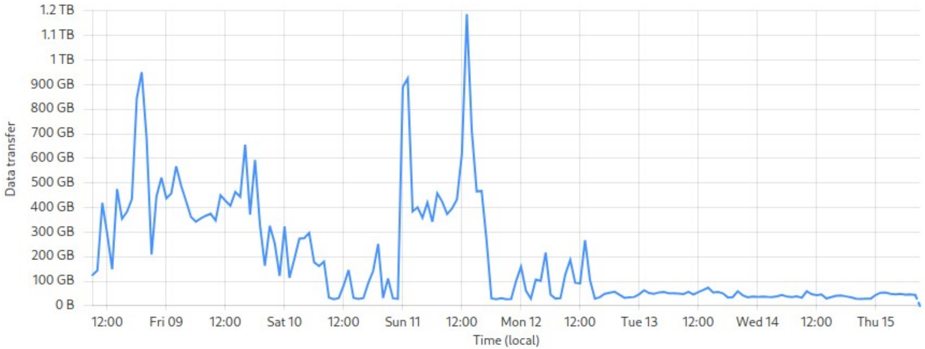
How Erik helps

- Erik relays simplify RP interactions – only one server to contact
- Relays are interchangeable – if one has problems, try another

Problems with rsync/RRDP (2)

Underprovisioning of services

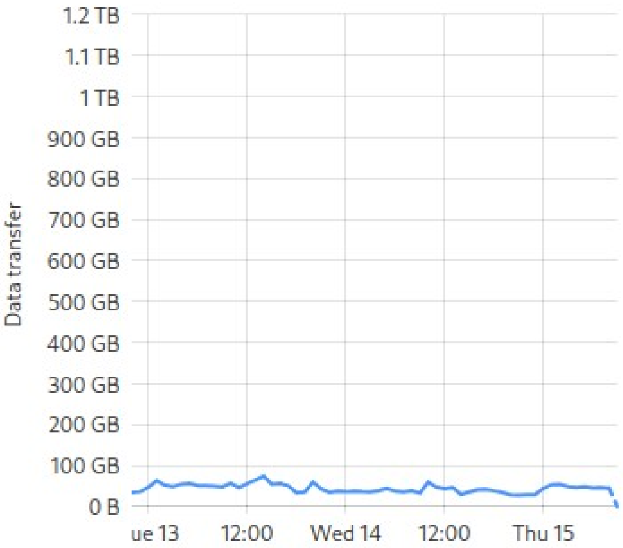
- Insufficient bandwidth/processing capacity
- Or capacity fine in most cases, but not resilient to problems



Problems with rsync/RRDP (2)

How Erik helps

- Instead of handling 2,000 RPs, PPs only have to handle ~20 relays (in full deployment)



Problems with rsync/RRDP (3)

```
<delta version="1"
       session_id="8dad0cc8-0bc8-4021-88ed-e75e295df946"
       serial="16355"
       xmlns="http://www.ripe.net/rpki/rrdp">
    <publish uri=".../Y_wvXeCUyDltyLyXRO1oL_SyOIE.crl"
             hash="3d751ce9dfaa4e84f9850ff0e2882739...">
        MIICGDCCAQACAQEwDQYJKoZIhvcNAQELBQAwRjER...
        ...
    </publish>
    <publish uri=".../Y_wvXeCUyDltyLyXRO1oL_SyOIE.mft"
             hash="26da47ee98a1fcbb2bf59d451786db57...">
        MIIIPQYJKoZIhvcNAQcCoIIILjCCCCoCAQMxDTAL...
        ...
    </publish>
    ...
</delta>
```

RRDP encoding introduces overhead

- Raw RPKI objects (DER-encoded) are first base64-encoded (33% increase in size) and then wrapped in XML (~10% increase in size)

Problems with rsync/RRDP (3)

```
  0:d=0  hl=4 l=10310 cons: SEQUENCE
  4:d=1  hl=2 l=  11 prim:  OBJECT            :1.2.840.113549.1.9.16.1.55
 17:d=1  hl=4 l=10293 cons:  cont [ 0 ]
 21:d=2  hl=4 l=10289 cons:   SEQUENCE
 25:d=3  hl=2 l=  13 prim:   IA5STRING         :rpki.ripe.net
 40:d=3  hl=2 l=  15 prim:   GENERALIZEDTIME   :20260108232054Z
 57:d=3  hl=2 l=  11 cons:   SEQUENCE
 59:d=4  hl=2 l=   9 prim:    OBJECT            :sha256
 70:d=3  hl=4 l=10240 cons:   SEQUENCE
 74:d=4  hl=2 l=  38 cons:    SEQUENCE
 76:d=5  hl=2 l=  32 prim:     OCTET STRING      [HEX DUMP]:B5E3...
110:d=5  hl=2 l=   2 prim:     INTEGER           :4278
114:d=4  hl=2 l=  38 cons:    SEQUENCE
116:d=5  hl=2 l=  32 prim:     OCTET STRING      [HEX DUMP]:76AB...
150:d=5  hl=2 l=   2 prim:     INTEGER           :4846
```

How Erik helps

- Raw RPKI objects are returned as-is
- Supporting response types are also DER-encoded

Problems with rsync/RRDP (4)

RRDP does not support ‘repair’

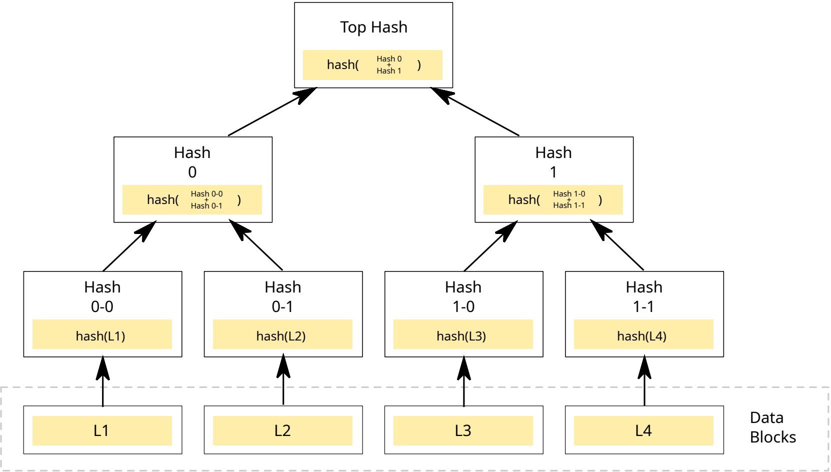
- If client not able to synchronise by way of delta (cheap), then client will reinitialise from snapshot (expensive, plus client will have most of the content already)
- ‘Thundering herd’ problem

```
$ wget https://rrdp.apnic.net/.../delta.xml
--2026-02-09 11:07:34--  https://rrdp.apnic.net/.../delta.xml
...
Saving to: ‘delta.xml’

delta.xml       66.51K  --.-KB/s    in 0.04s

2026-02-09 11:07:34 (1.48 MB/s) - ‘delta.xml’ saved [68102]

$ wget https://rrdp.apnic.net/.../snapshot.xml
--2026-02-09 11:06:30--  https://rrdp.apnic.net/.../snapshot.xml
...
Saving to: ‘snapshot.xml’

snapshot.xml      102.49M  4.24MB/s    in 24s

2026-02-09 11:06:55 (4.23 MB/s) - ‘snapshot.xml’ saved [107465798]
```

Problems with rsync/RRDP (4)

How Erik helps

- Uses Merkle trees to optimise the process of diffing the local repository with the remote repository
- Supports retrieval of individual files, like in rsync



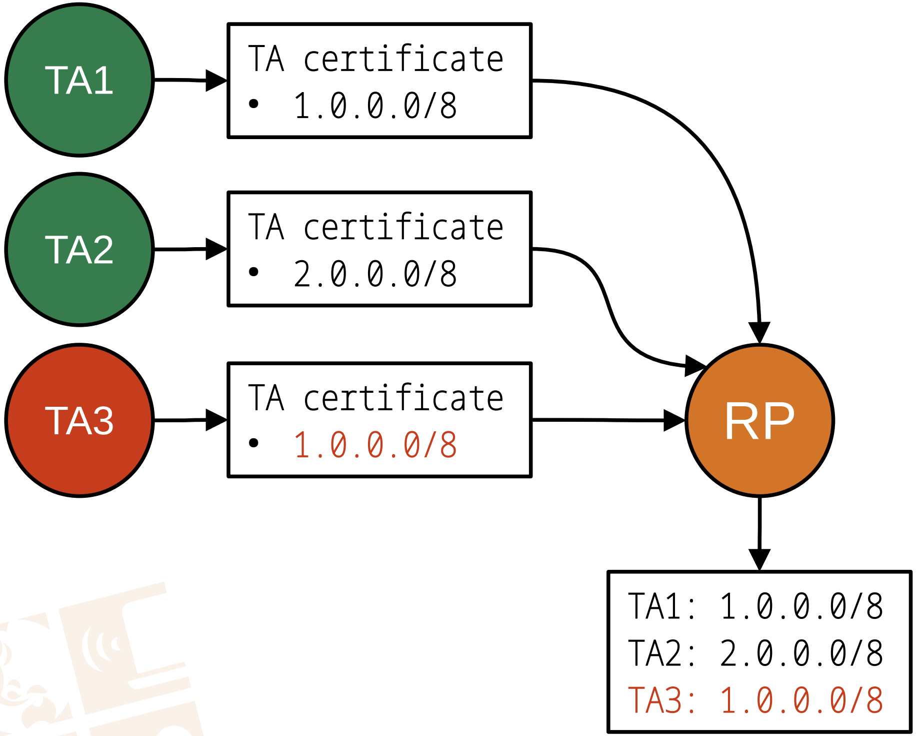
Current status

- Relay implementation
  - https://github.com/job/rpkitouch
  - https://relay.rpki-servers.org (see draft for others)
- Client implementation
  - https://github.com/bjpbakker/EPIC (by RIPE staff)
  - APNIC POC implementation (available soon)
- Adopted by WG – ongoing discussion/testing/etc.

NRO Trust Anchor Constraints

- An updated approach to RPKI validation, where a set of TAs jointly attests the resource holdings of each other TA in the set, by way of new signed objects
  - State object: what each TA has at the moment
  - Transfer objects: issued when a TA needs to move resources to another TA
- An RP can process these new objects in order to limit the resources that a given TA may claim

Problem with current validation



TA certificate
- 1.0.0.0/8

TA certificate
- 2.0.0.0/8

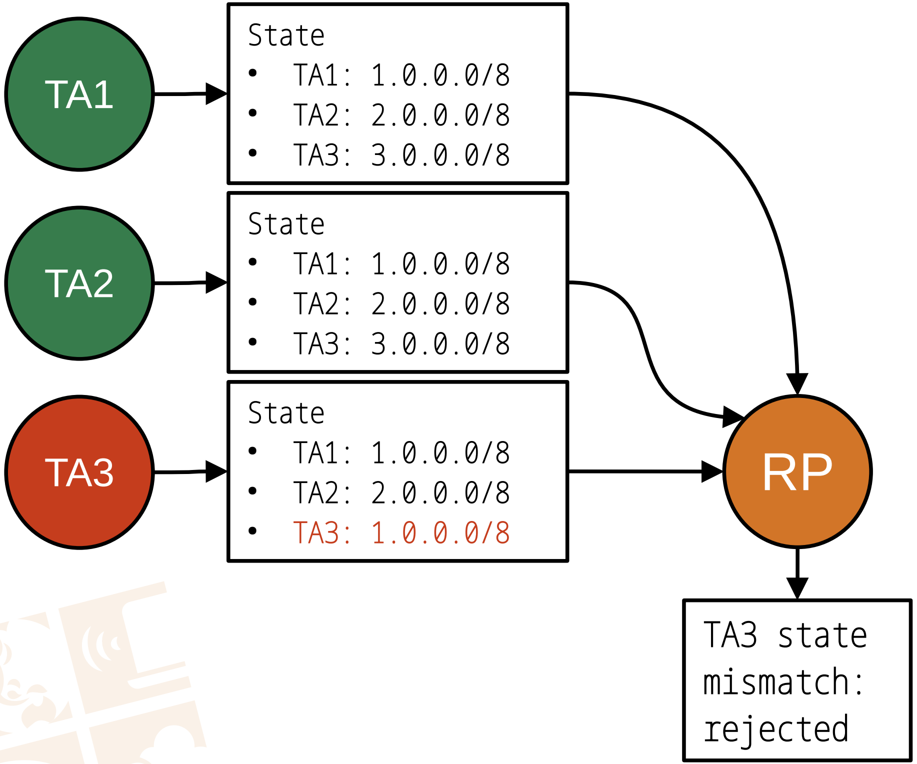
TA certificate
- 1.0.0.0/8

```
TA1: 1.0.0.0/8
TA2: 2.0.0.0/8
TA3: 1.0.0.0/8
```

RPs unable to prevent overclaiming

- RPs currently trust what each TA tells them
- No good way to prevent TA3 from claiming resources that actually belong to TA1

Problem with current validation



State
- TA1: 1.0.0.0/8
- TA2: 2.0.0.0/8
- TA3: 3.0.0.0/8

State
- TA1: 1.0.0.0/8
- TA2: 2.0.0.0/8
- TA3: 3.0.0.0/8

State
- TA1: 1.0.0.0/8
- TA2: 2.0.0.0/8
- TA3: 1.0.0.0/8

TA3 state mismatch: rejected

How TA constraints help

- RP first checks that the state files match
- If they do, then those statements limit what can be claimed
- If they don’t (e.g.), then the client can rely on those that match to limit what can be claimed

Current status

- APNIC POC implementation (internal)
- Substantial feedback off-list
- Iterating on the design, and will be presenting again at the next IETF meeting in March

Publication Server BCP

- A single reference point for best practice as to running an RPKI publication server and associated PPs
- Having a definitive, authoritative document on this topic helps when responding to questions about how systems should be configured, problems handled, and so on
- Adopted by working group, and likely to be completed within the next few months

2026 APRICOT

APNIC 61

JAKARTA, INDONESIA
4 – 12 February 2026

#apricot2026