Decentralized RPKI Repository Architecture

draft-li-sidrops-drr-architecture-00

Dan Li, **Yingying Su**, Li Chen
ZGC lab & Tsinghua university

July, 2025

IETF draft

IETF 123:

- draft-li-sidrops-drr-architecture-00
- present a new RPKI Repository architecture dRR
- design reliable, scalable and secure RPKI Repository architectrue

IETF 122:

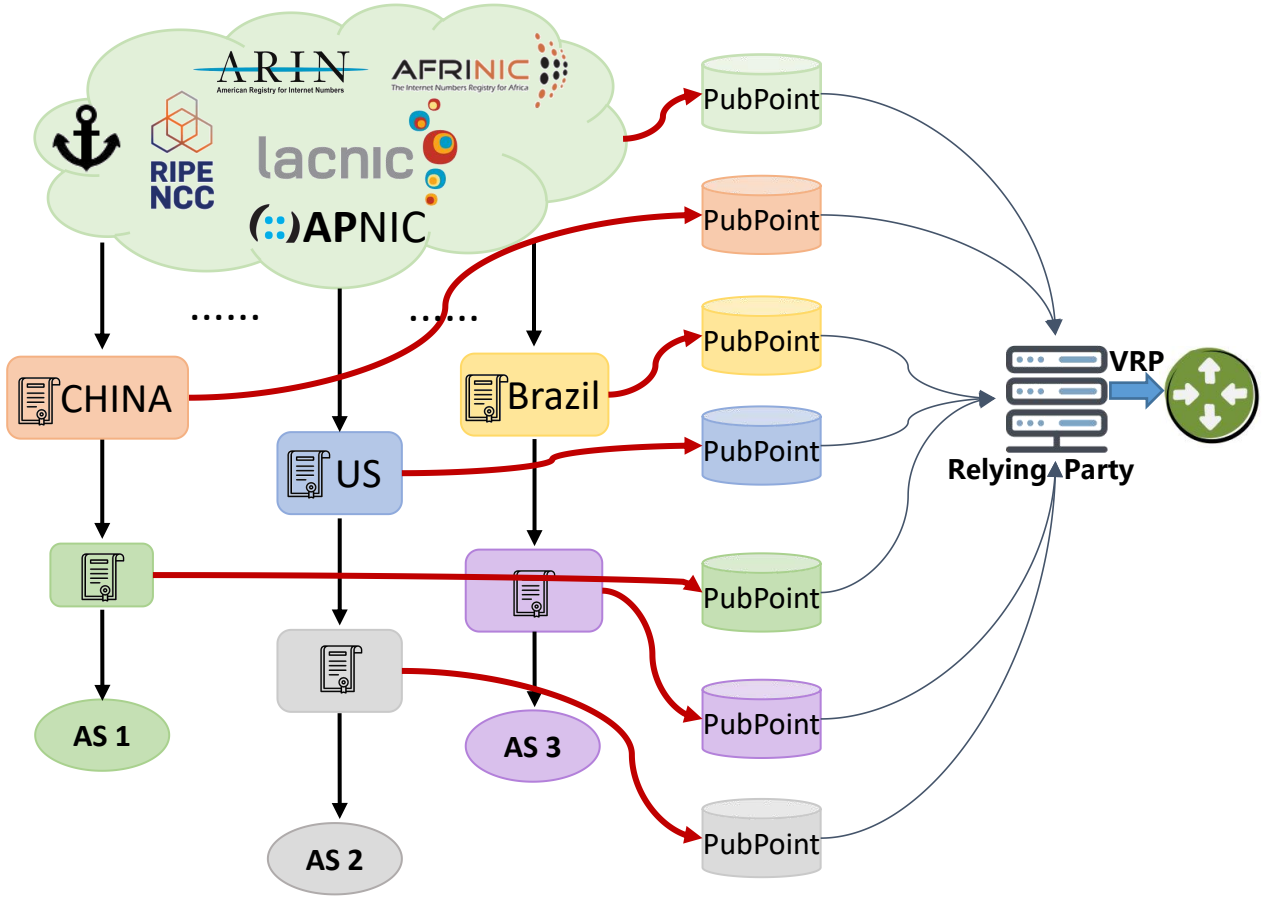
- draft-li-sidrops-rpki-repository-problem-statement-01
- introduce the requirements for an enhanced RPKI Repository

IETF 121:

- draft-li-sidrops-rpki-repository-problem-statement-00
- analyzed the problems of the current RPKI Repository

Current RPKI Repository Problems

The coupling between RPKI CAs and PPs introduces significant Reliability & Scalability issues.



P1. Poor Reliability

- Each RPKI object is hosted at the single **PP running** or **designating** by its **CA**
- The PP goes offline, RPs cannot retrieve its stored data
- CDN is not a fundamental solution

P2. Poor Scalability

- The number of PPs and RPs are expected to **increase**
- The **complex connections** between PPs and RPs and **pulling-based data fetching** prevent RPs from obtaining updated ROA quickly

P3. Lack Consistency & Auditability

- Lacks a globally consistent historical RPKI data view
- **Auditability** is important for **improving transparency, misconfiguration diagnosing** ...

key Idea of dRR

Separating RPKI object distribution from signing!

- Decouple PPs from RPKI CAs
- Design an independent repository for RPKI ⟶ **dRR**

dRR *vs* the current RPKI Repository

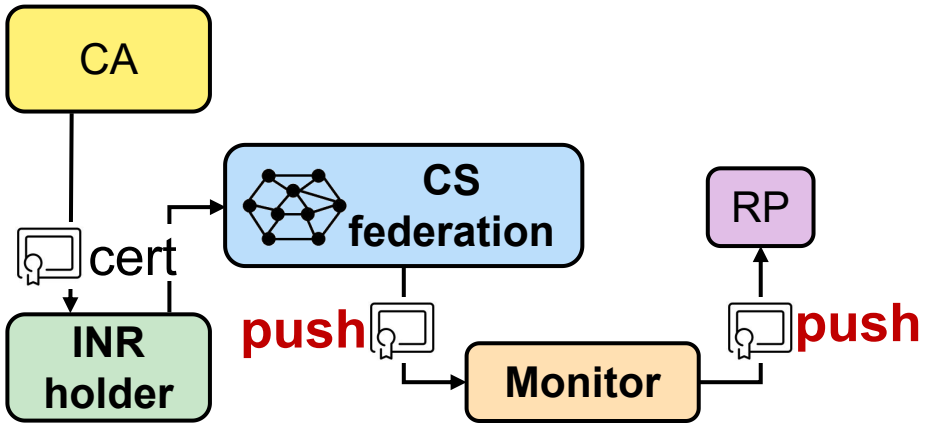


Fig. dRR architecture

VS

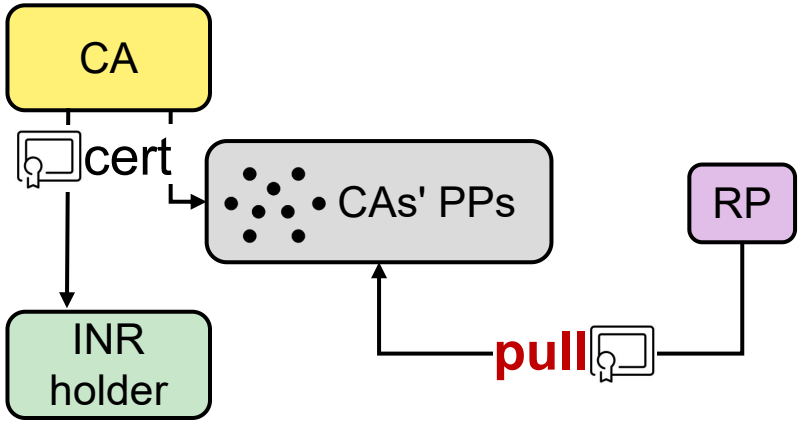


Fig. the current RPKI Repository

Two new entities for dRR: *CS federation* and *Monitor*

dRR *vs* the current RPKI Repository

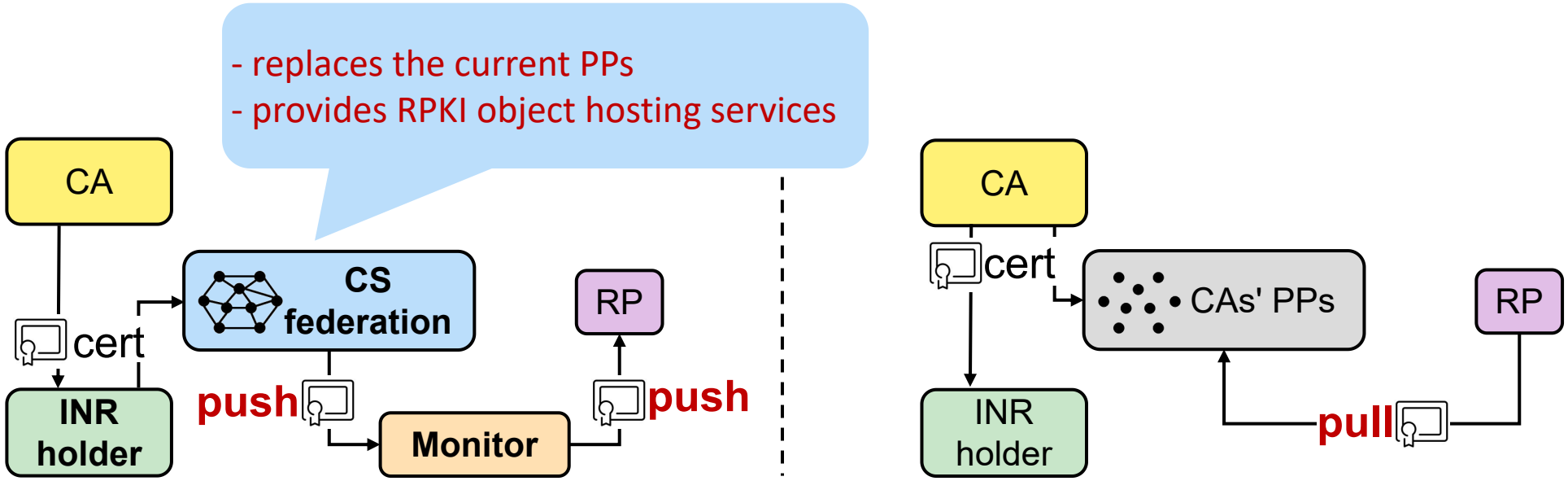


Fig. dRR architecture *VS* **Fig. the current RPKI Repository**

Two new entities for dRR: *CS federation* and *Monitor*

dRR *vs* the current RPKI Repository

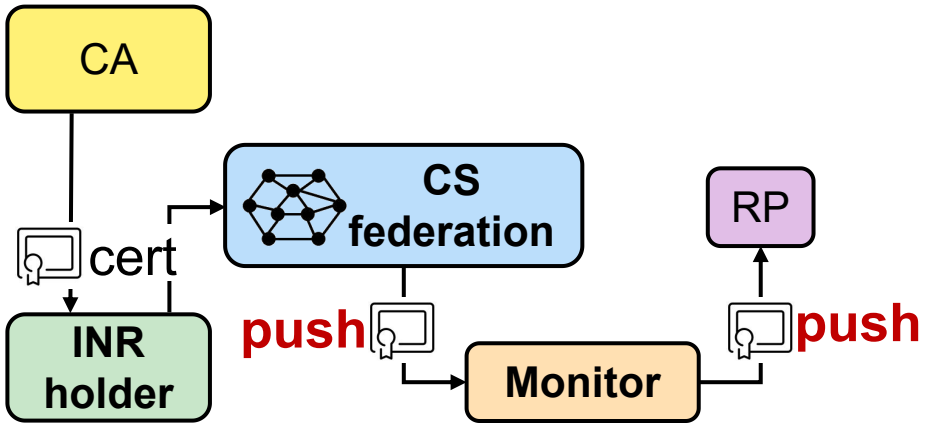


Fig. dRR architecture

VS

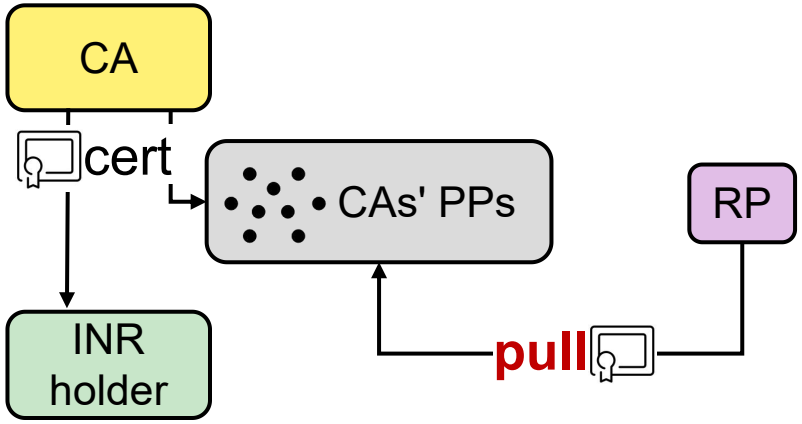
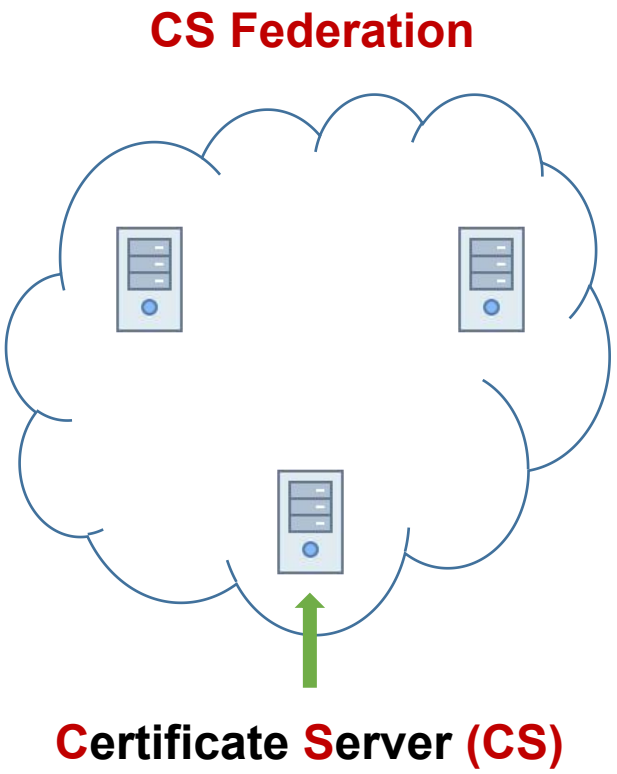


Fig. the current RPKI Repository

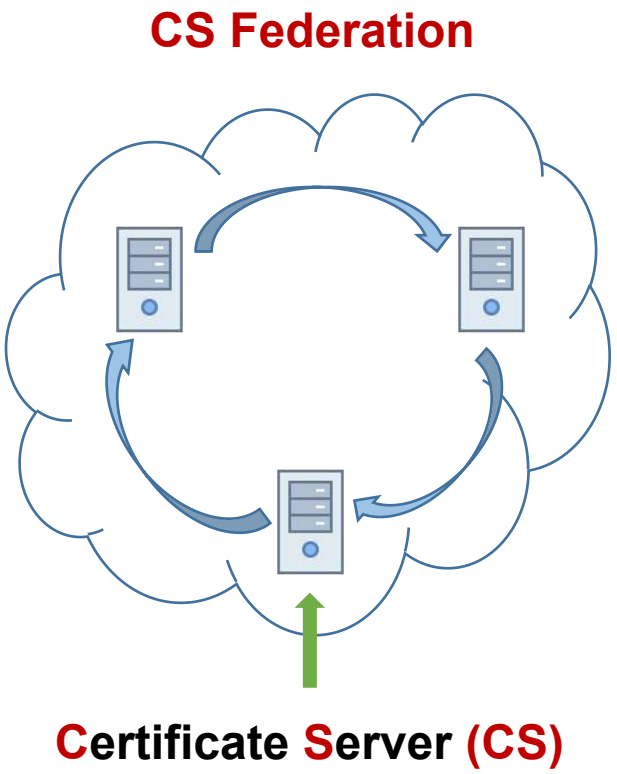
- act as middleware between federation and RPs
- responsible for retrieving RPKI updated data from the federation and pushing it to RPs.

dRR Architecture



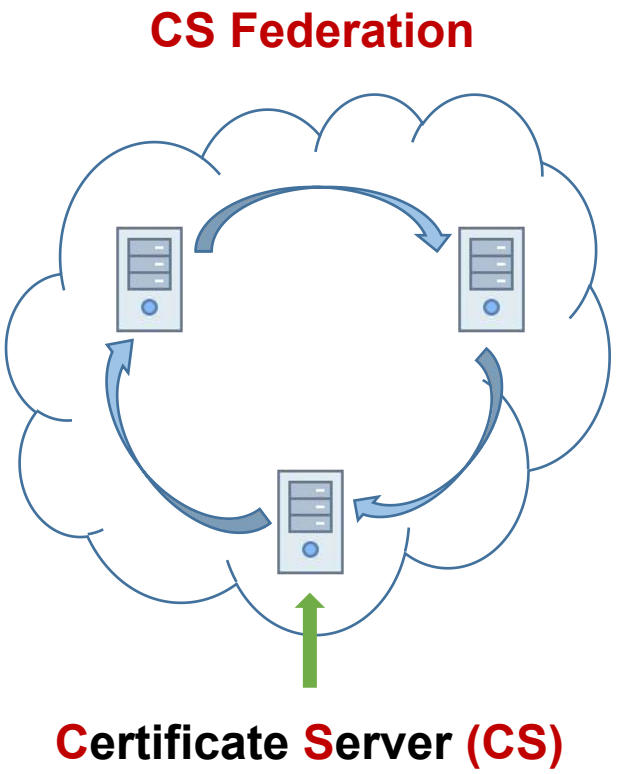
- CS federation is composed of **C**ertificate **S**erver nodes (CS nodes)
  - not operated or designated by RPKI CAs

dRR Architecture



- CS federation is composed of **C**ertificate **S**erver nodes (CS nodes)
  - not operated or designated by RPKI CAs
  - form an independent federation
  - has an access mechanism to select trustworthy data hosting nodes

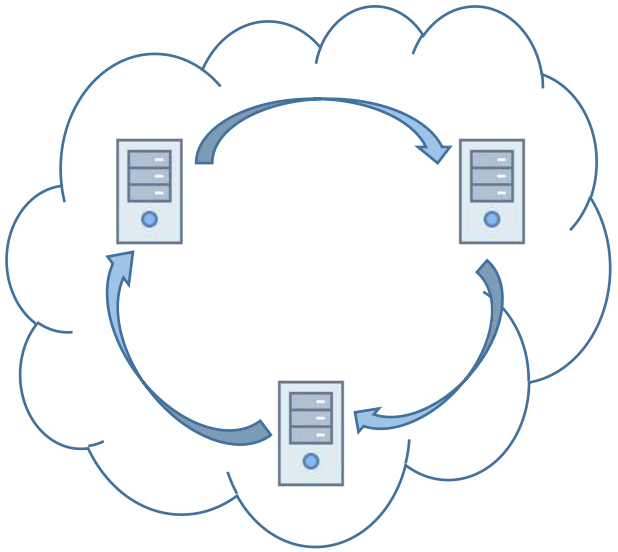
dRR Architecture



- Who can **run** dRR **CS nodes**?
  - a diverse set of orgs to operate CS nodes
  - RIRs, state-run institutions and large ISPs (e.g.,Akamai, Amazon, Cloudflare, etc.), ...
  - have reliable service infrastructures and strong reputations

dRR Architecture

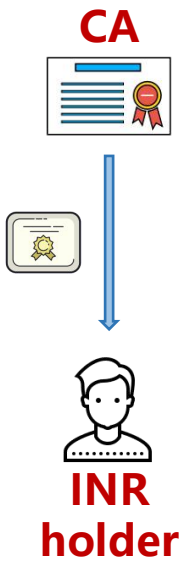
CS Federation

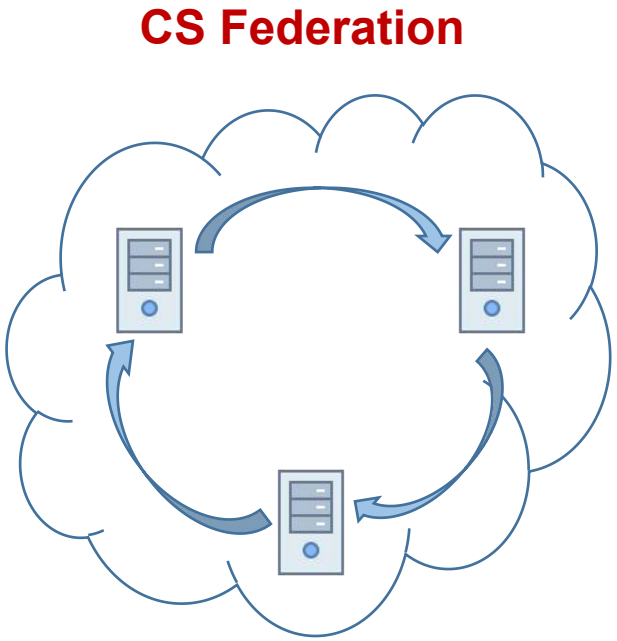


- Upper CA signs RPKI objects for the subordinate INR holder (step 0)

CS federation enables redundant, multi-nodes storage of RPKI data at the architectural level to enhance overall system reliability.

dRR Architecture

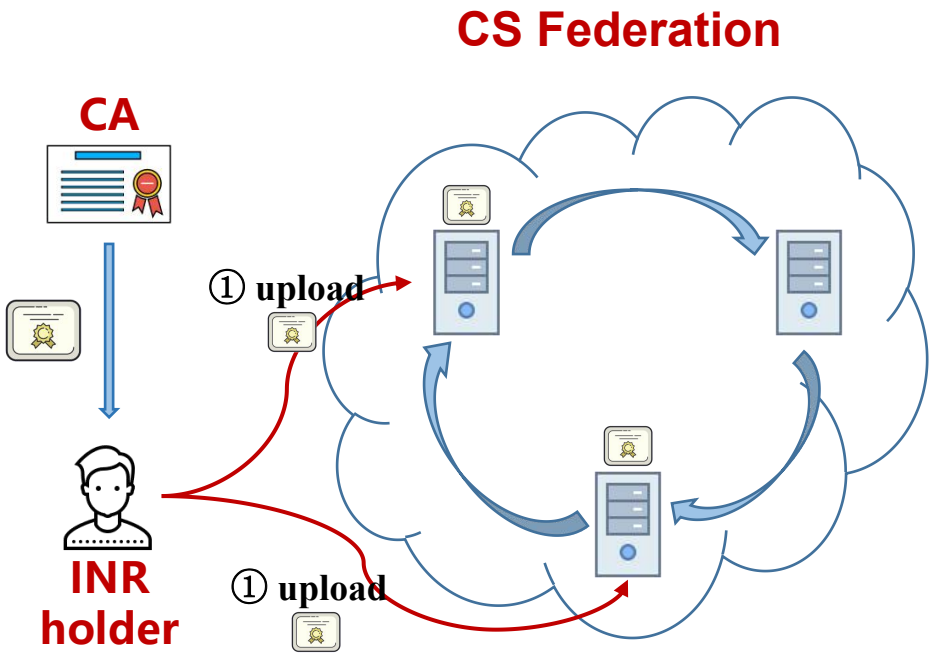




- Upper CA signs RPKI objects for the subordinate INR holder (step 0)

CS federation enables redundant, multi-nodes storage of RPKI data at the architectural level to enhance overall system reliability.

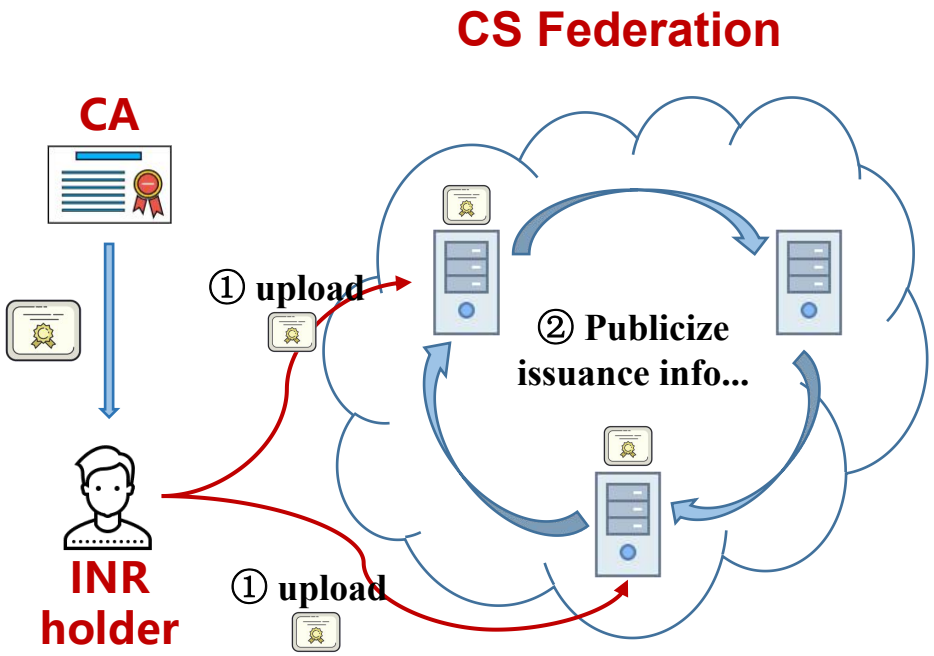
dRR Architecture



- INR holder can freely choose nodes it trusts to host its RPKI object. (step ①)
  - each object can be backed up across multiple physically isolated and independent CS nodes
  - providing more resilience against single point of failures or outages

CS federation enables redundant, multi-nodes storage of RPKI data at the architectural level to enhance overall system reliability.

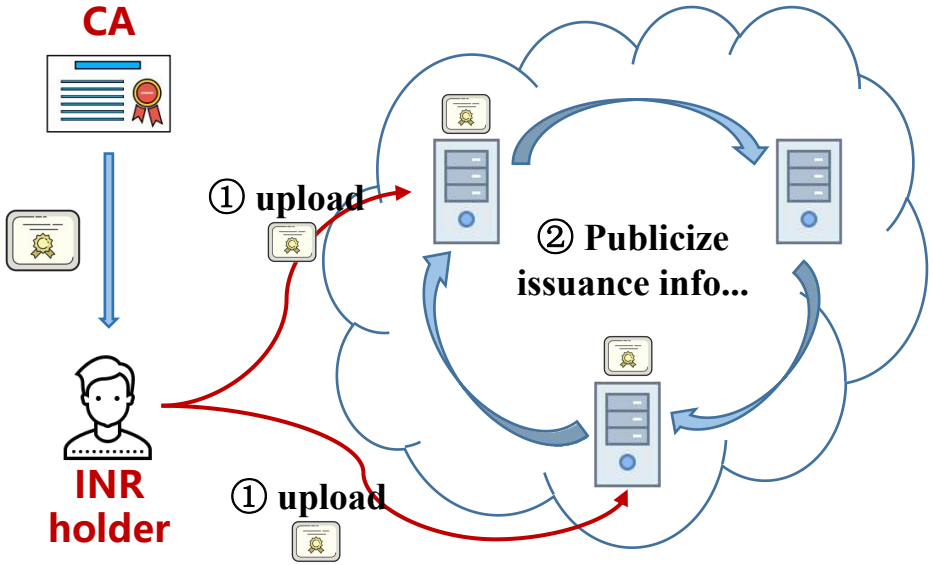
dRR Architecture



- Publication and revocation will be announced within the CS federation. (step ②)
  - allowing all federation members to establish a globally consistent RPKI data view...

CS federation enables redundant, multi-nodes storage of RPKI data at the architectural level to enhance overall system reliability.

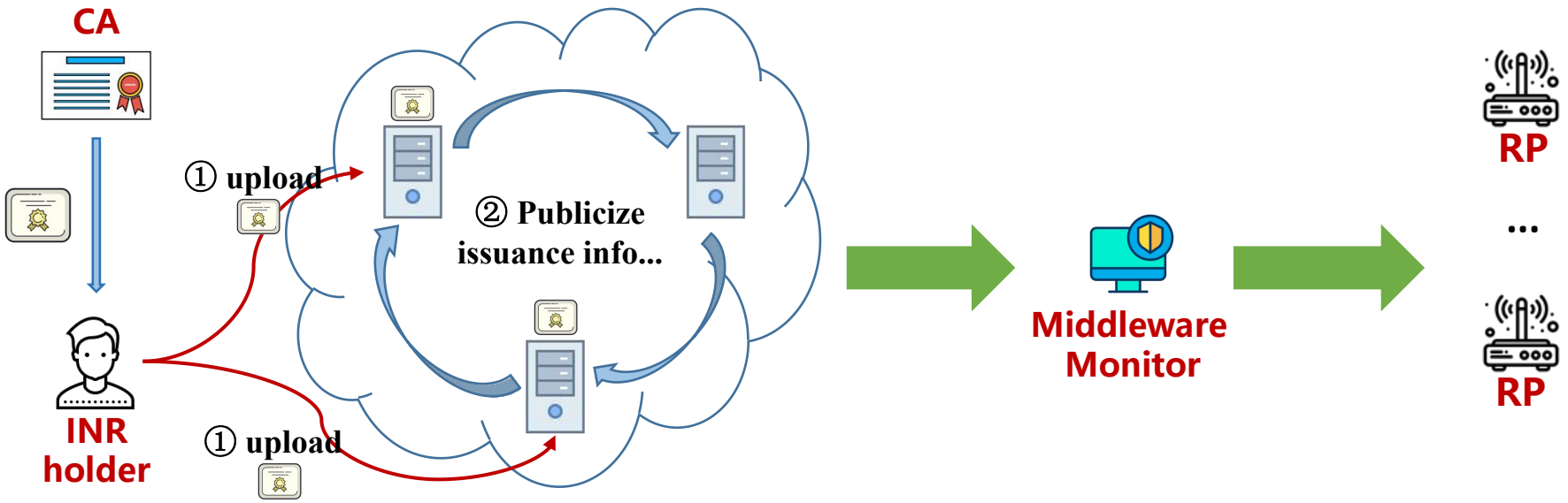
dRR Architecture



Middleware
Monitor

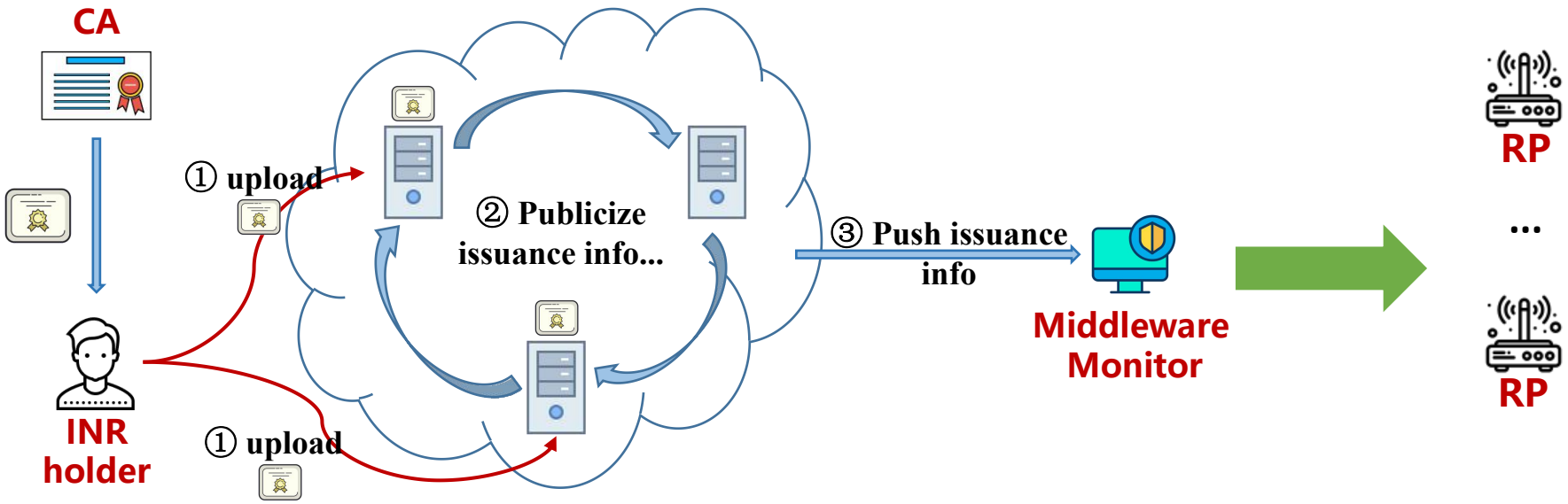
- Two purpose for Middleware Monitor
  - (1) to alleviate the connection burden between the CS nodes and the thousands of RPs; and
  - (2) to proactively push updated RPKI data from the CS nodes to the RPs

dRR Architecture



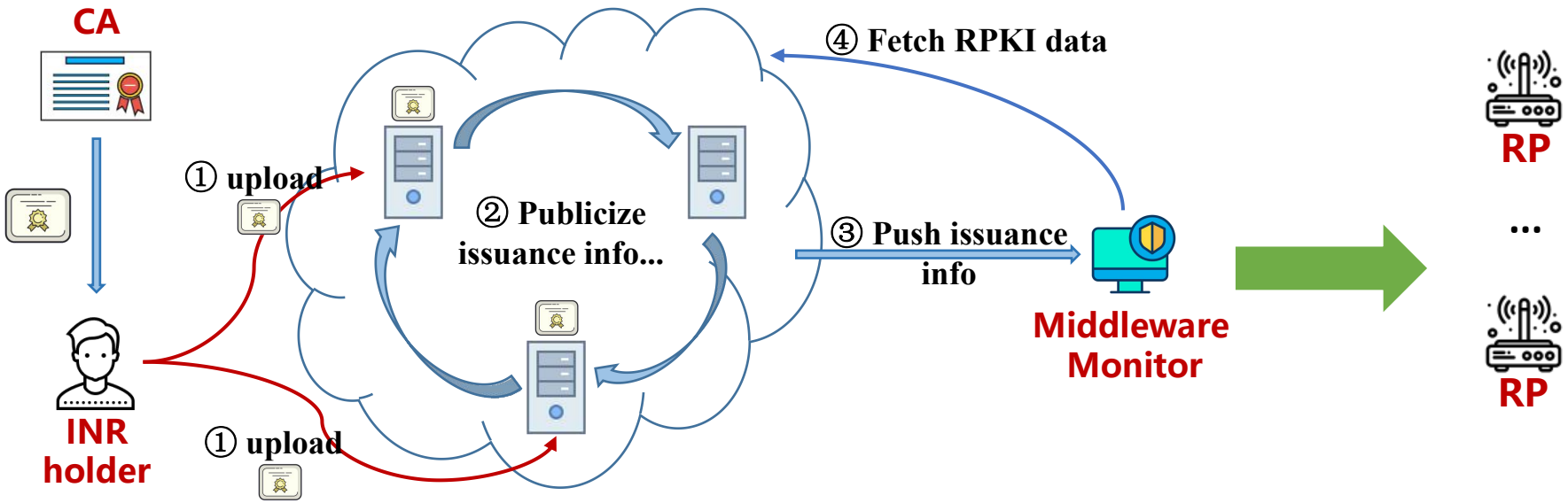
- Two purpose for Middleware Monitor
  - (1) to alleviate the connection burden between the CS nodes and the thousands of RPs; and
  - (2) to proactively push updated RPKI data from the CS nodes to the RPs

dRR Architecture



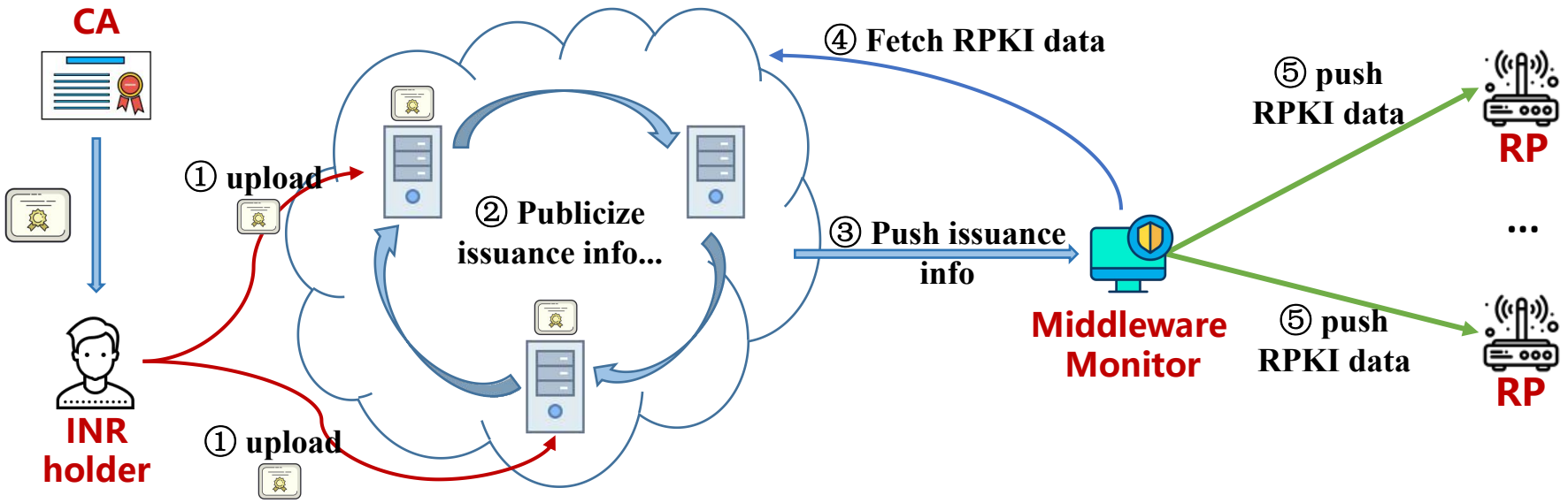
- Once the CS federation confirms a batch of publication & revocation information
- Each CS node will proactively push the confirmed RPKI update information to its connected Monitors (step ③)

dRR Architecture



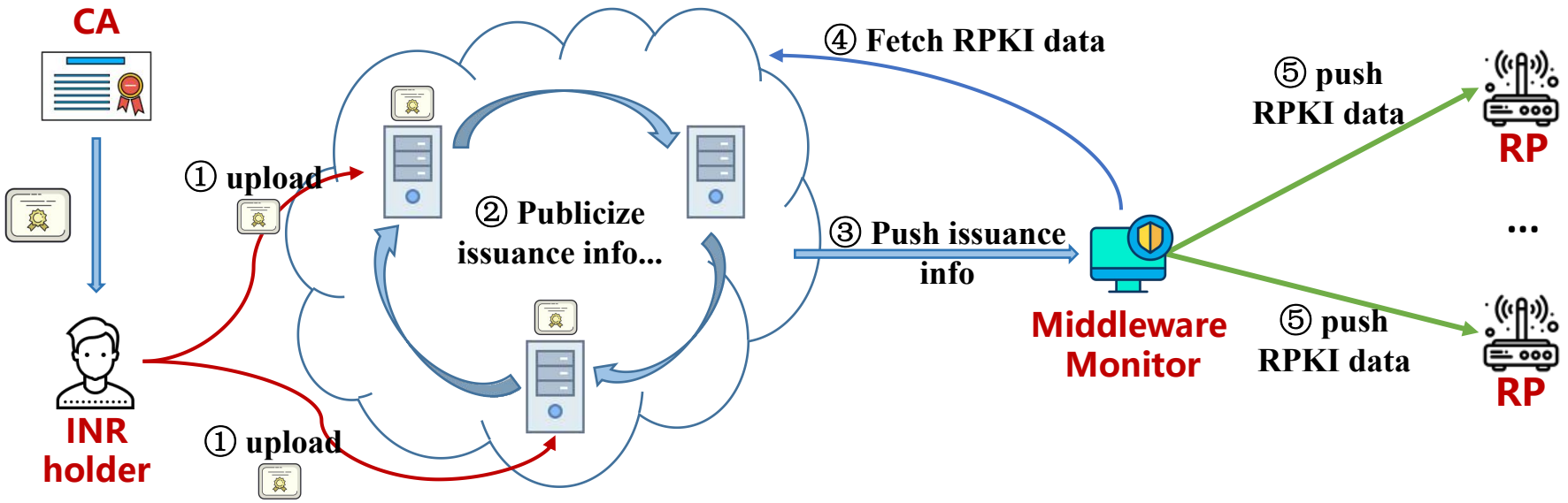
- Monitor then fetch newly added RPKI objects from the specified CS nodes according to the publication info (step ④)

dRR Architecture



- Then Monitor will proactively forward both update information and newly added RPKI objects to its connected RPs (step ⑤)

dRR Architecture



- Achieve push-based data synchronization model, allow RPs and BGP routers to timely receive update RPKI data
- Who can run Monitor
  - regional networks or large ISPs operate Monitors to serve local RPs

dRR properties

- **0. High Compatibility**
  - Not modify the current hierarchical RPKI certificate issuance architecture
  - Be compatible with current RPKI Repository and Allow the ROA deployers to freely choose between two repositories...
- **1. High Reliability**
  - Distributed data storage, enable a single RPKI object to be backed up across multiple nodes
  - The integrity of the RPKI data accessed by RPs will not be affected even if a node experiences failure...
- **2. High Scalability**
  - Simplifies the connections between repository nodes and RPs
  - Enables proactive pushing of RPKI data from repository to RPs
- **3. Auditability & Consistency**
  - Maintains a complete and consistent RPKI historical data view, provide audit function for RPKI
  - Auditability is important for improving transparency, misconfiguration diagnosing

Developing a mature and stable dRR prototype

Welcome you to join us!

Thank you!