EdgeRDV: A Framework for Edge Workload Management at Scale

Gloire Rubambiza (gloire@cs.cornell.edu)

IEEE Edge - 07/04/2023 - Chicago, IL

Co-authors: Braulio Dumba (IBM Research), Andrew J. Anderson (IBM Research), Hakim Weatherspoon (Cornell University)

Motivation

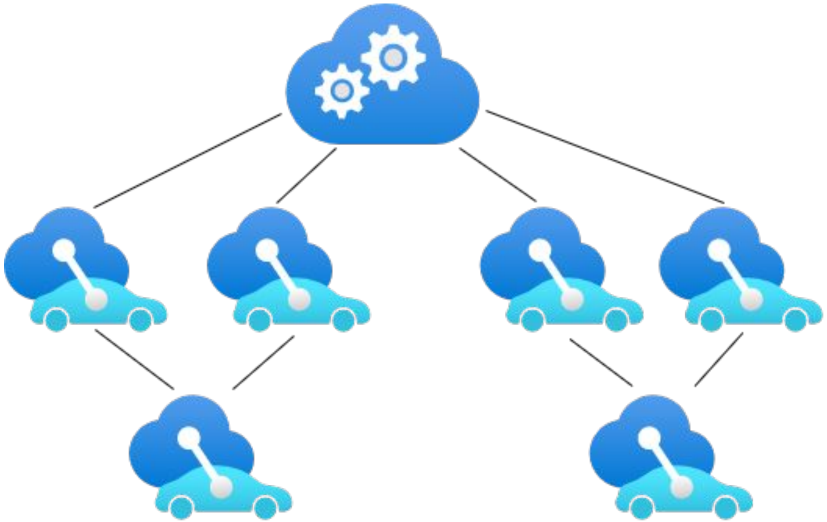
State-of-the-art

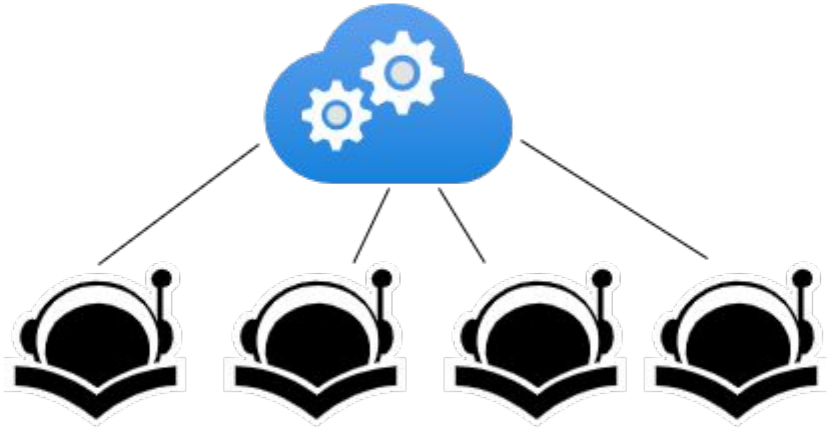
Methodology

Analysis/Evaluation

Motivation

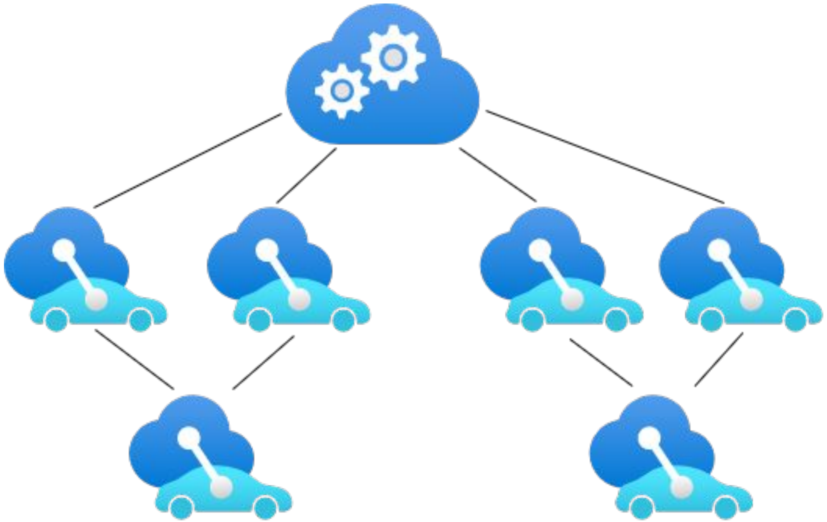
- Edge Computing
  - Supports manufacturing, smart homes, supply chains, etc
  - Reduces latency and bandwidth consumption
  - Novel applications such as AI inference at the edge

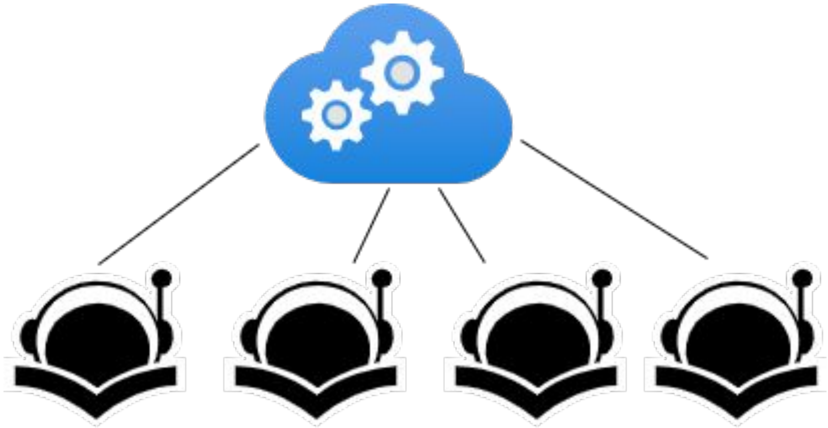




Edge Computing Challenges

- How do we manage application life-cycle at scale?
- Can we do so with limited cloud connectivity?
- How do we sustain operation when connectivity is lost?
- How do we gracefully recover from endpoint failures?





Edge Computing Challenges

- How do we manage application life-cycle at scale?
- Can we do so with limited cloud connectivity?
- How do we sustain operation when connectivity is lost?
- How do we gracefully recover from endpoint failures?

Research Question: how do we efficiently manage the application lifecycle on hundreds of thousands of endpoints?

Motivation

State-of-the-art

Methodology

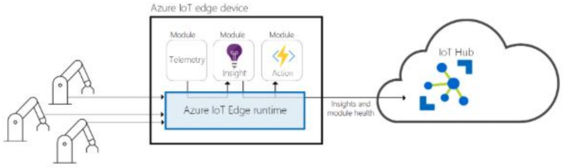
Analysis/Evaluation

Edge Workload Management in Resource-Constrained Environments

- Scaling Community Networks (CNs) with Community Cellular Manager. Hasan et al. NSDI. 2019.
- Energy-efficient Computing for Wildlife Tracking: Design Tradeoffs and Early Experiences with ZebraNet. Juang et al. ASPLOS. 2002.
- Visage: Enabling Timely Analytics for Drone Imagery. Jha et al. MobiCom. 2021.
- The Akamai Network: A Platform for High-Performance Internet Applications. Nygren et al. UMass Amherst. 2010.
- Experience in Implementing a Non-IP Routing Protocol VIRO in GENI. Dumba et al. 2014.

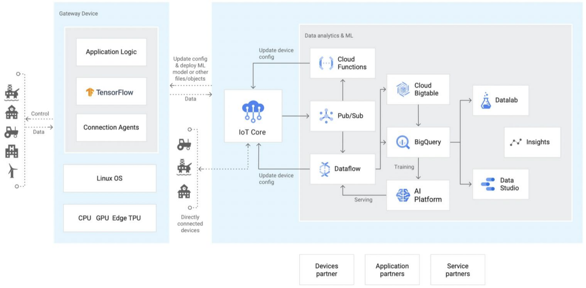
Industrial IoT Management Platforms

Azure IoT Edge



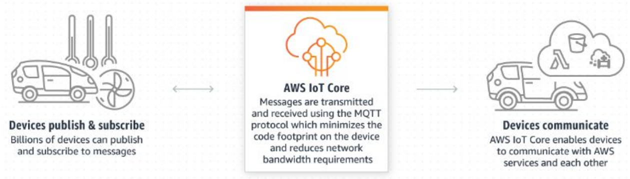
- Use cases
  - **Manufacturing**
  - **Electric**
  - **Automation**

Google IoT Core



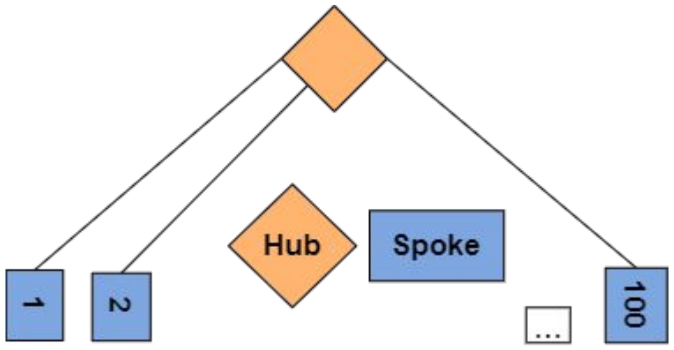
- Use cases
  - **Transportation**
  - **Utilities**
  - **Healthcare**

AWS IoT Core



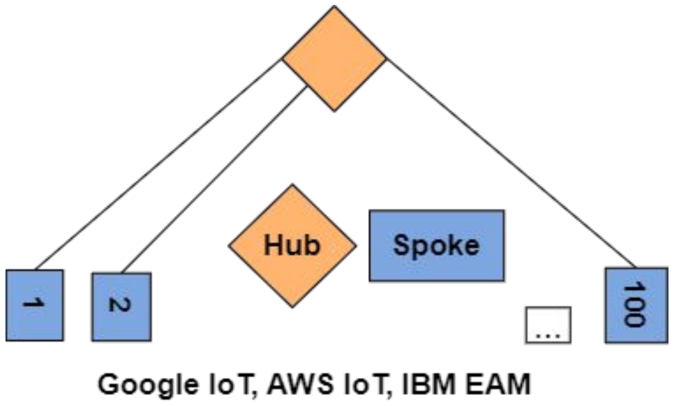
- Use cases
  - **Manufacturing**
  - **Smart homes**
  - **Supply chain**

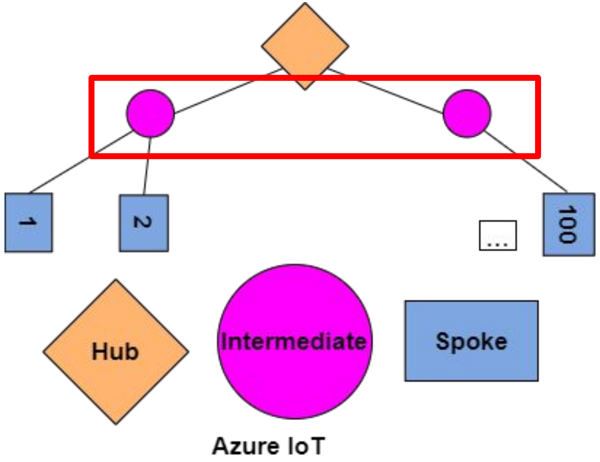
Industrial IoT Management Platforms



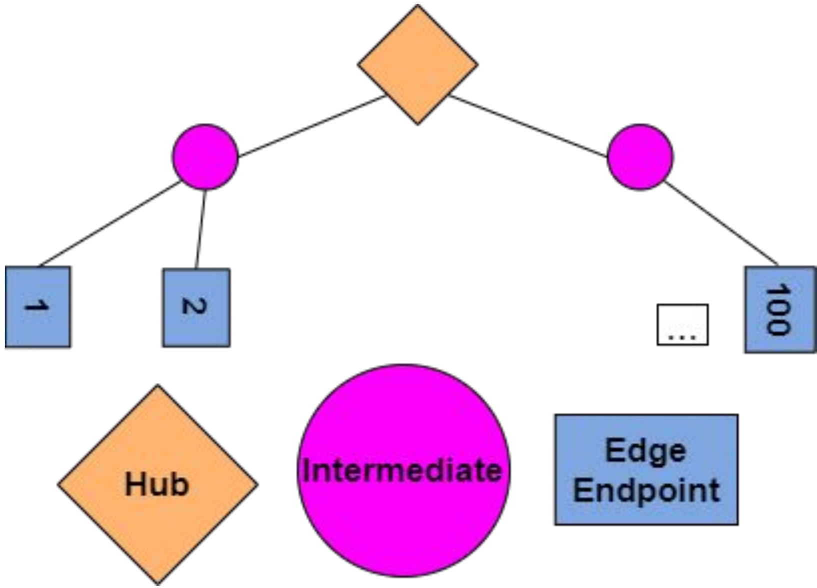
Google IoT, AWS IoT, IBM EAM

Industrial IoT Management Platforms



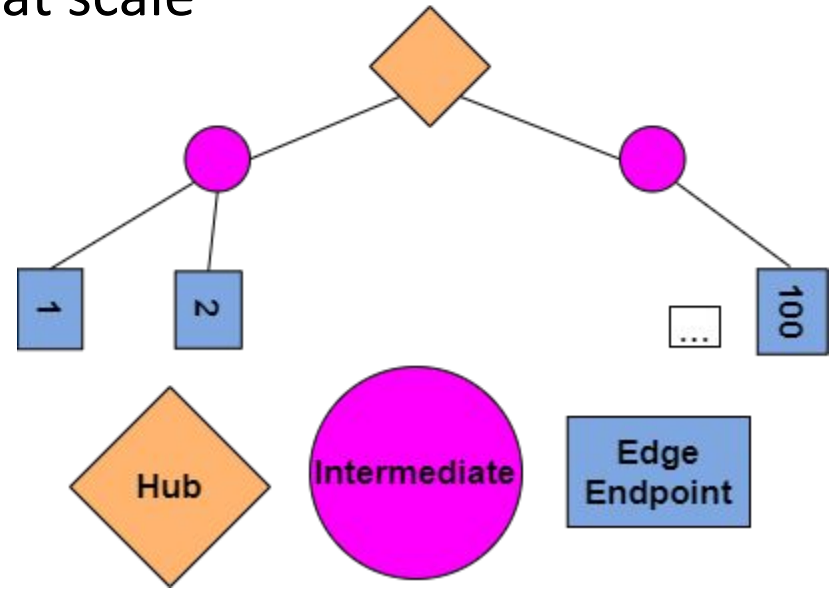


Industrial IoT Management Platforms



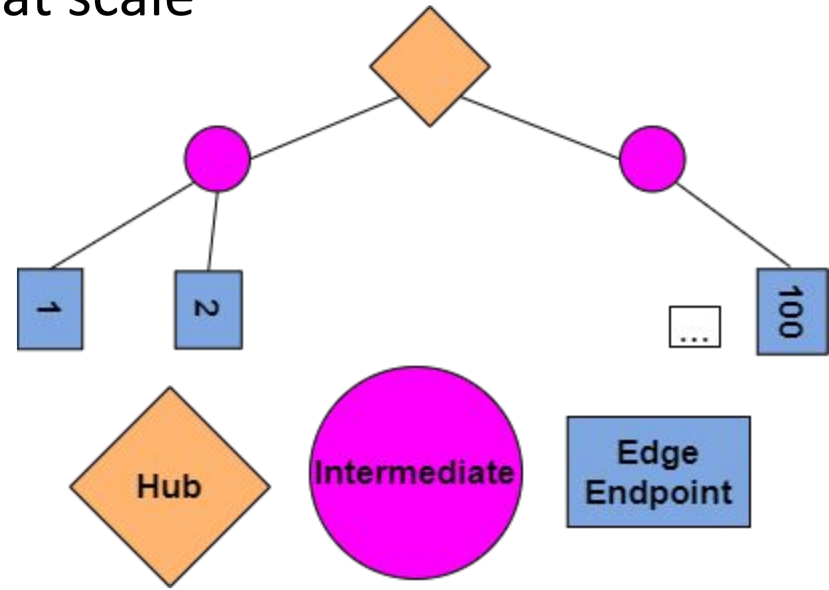
Industrial IoT Management Platforms

- Application life-cycle management at scale
- Limited connectivity
- Disconnected operation
- Failure recovery



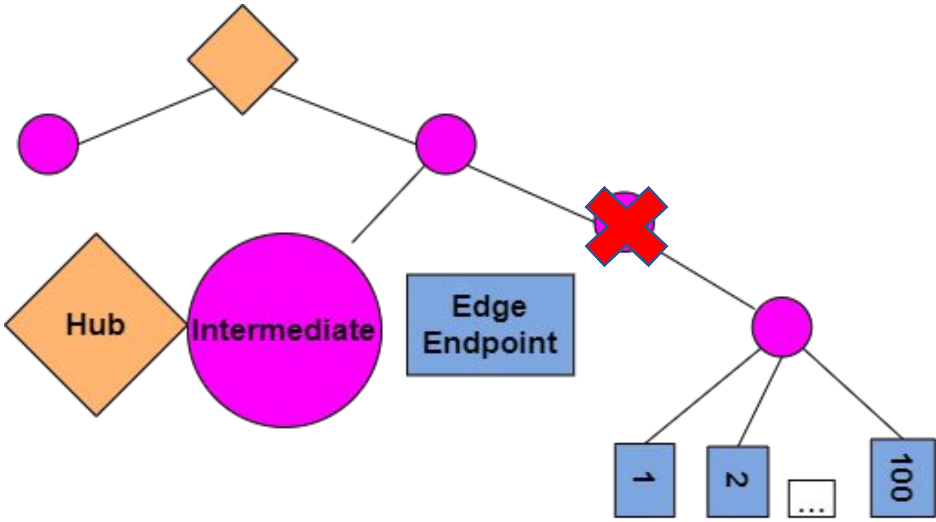
Industrial IoT Management Platforms

- Application life-cycle management at scale
  - **Bootstrapping IoT runtimes** 👍🏾
  - **Layered deployments** 👍🏾
- Limited connectivity
  - **Communication modalities** 👍🏾
- Disconnected operation
  - **Offline cache of messages** 👍🏾
- Failure recovery



Industrial IoT Management Platforms

- Application life-cycle management at scale
  - Bootstrapping IoT runtimes 👍
  - Layered deployments 👍
  - **Min. bandwidth consumption** ❌
- Limited connectivity
  - Communication modalities 👍
  - **Peer-to-peer communication** ❌
- Disconnected operation
  - Offline cache of messages 👍
- Failure recovery
  - **Caching workloads** ❌
  - **Avoid single point of failure** ❌



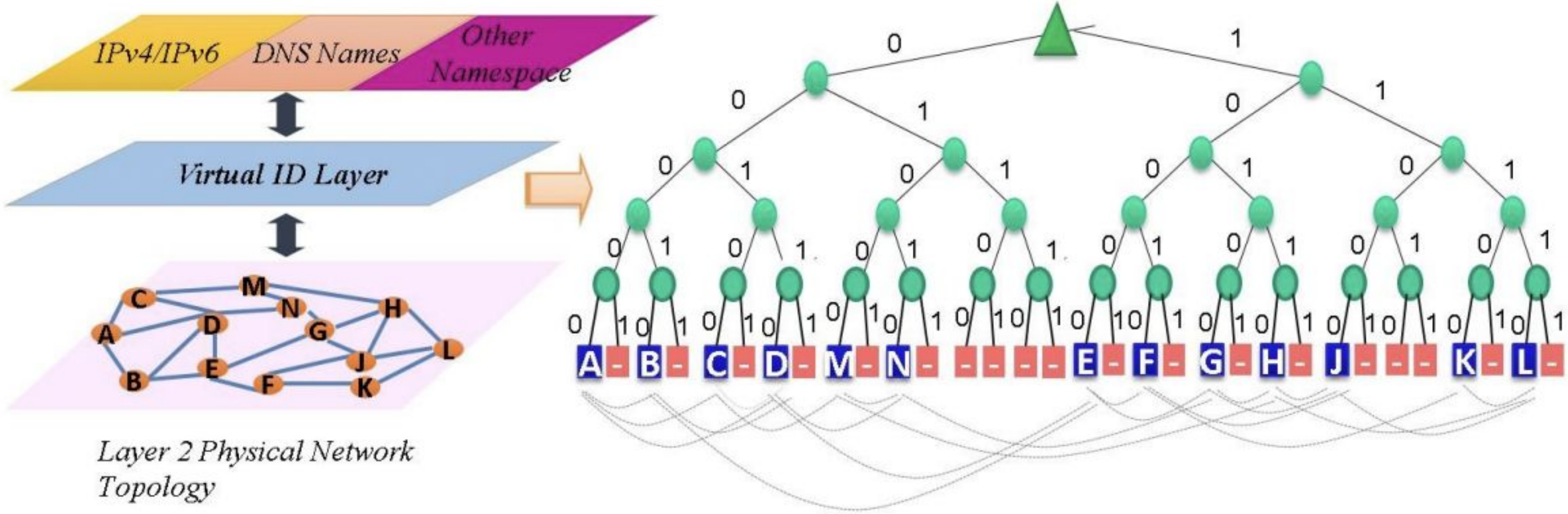
Motivation

State-of-the-art

Methodology

Analysis/Evaluation

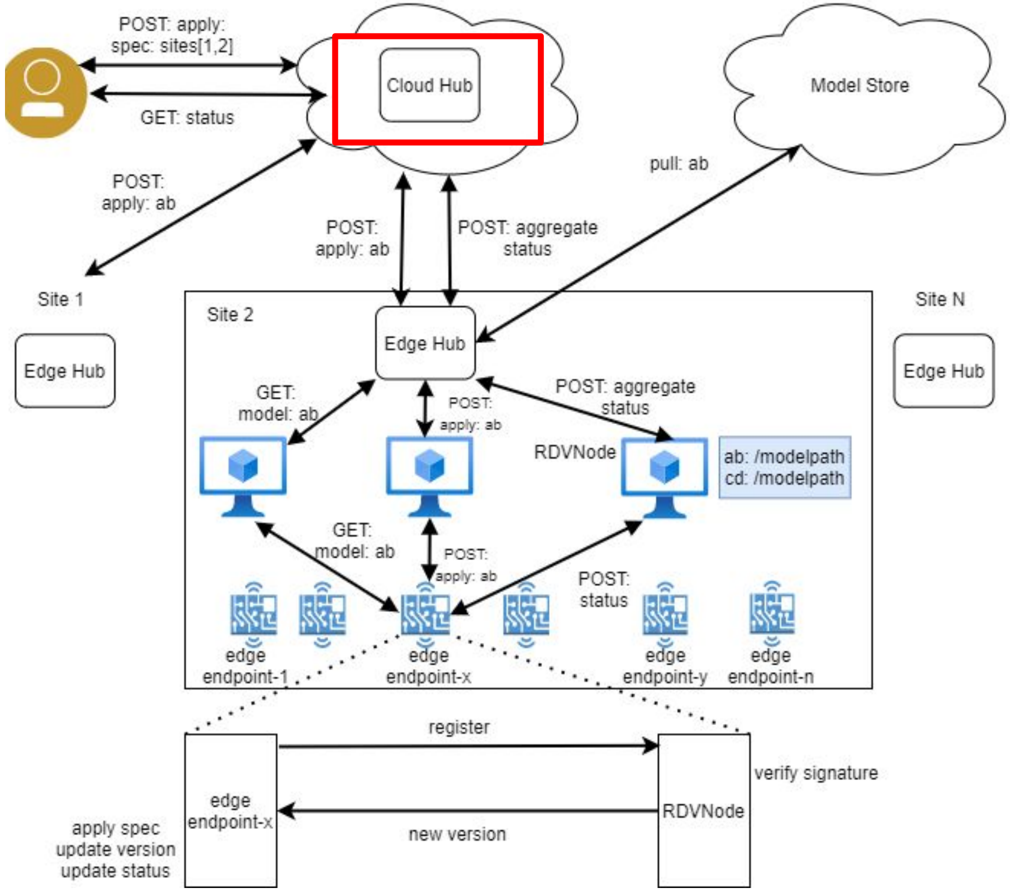
Proposed Solution – Key Idea
Scaling Edge Deployments using Rendezvous Nodes



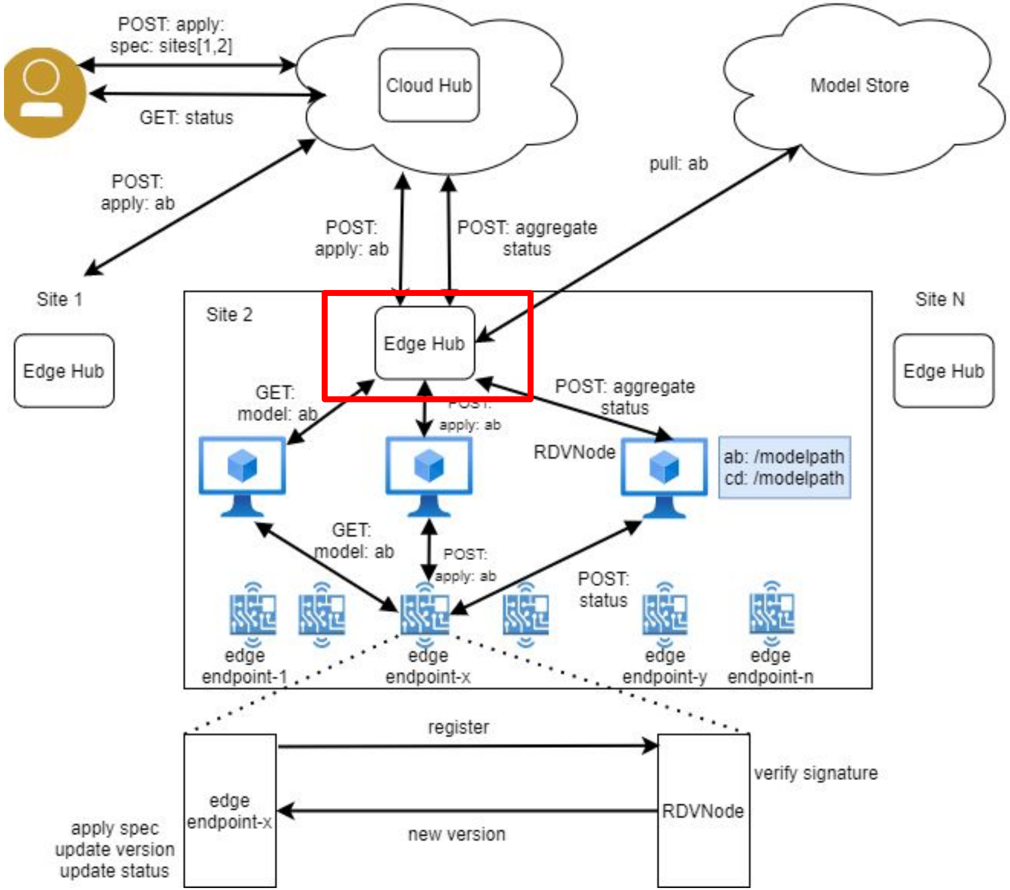
VIRO: A Scalable, Robust and Namespace Independent **V**irtual **I**d **RO**uting for Future Networks, Jain *et al.*, IEEE INFOCOMM, 2011

Experience in Implementing a Non-IP Routing Protocol VIRO in GENI, Dumba *et al.*, IEEE ICNP, 2014

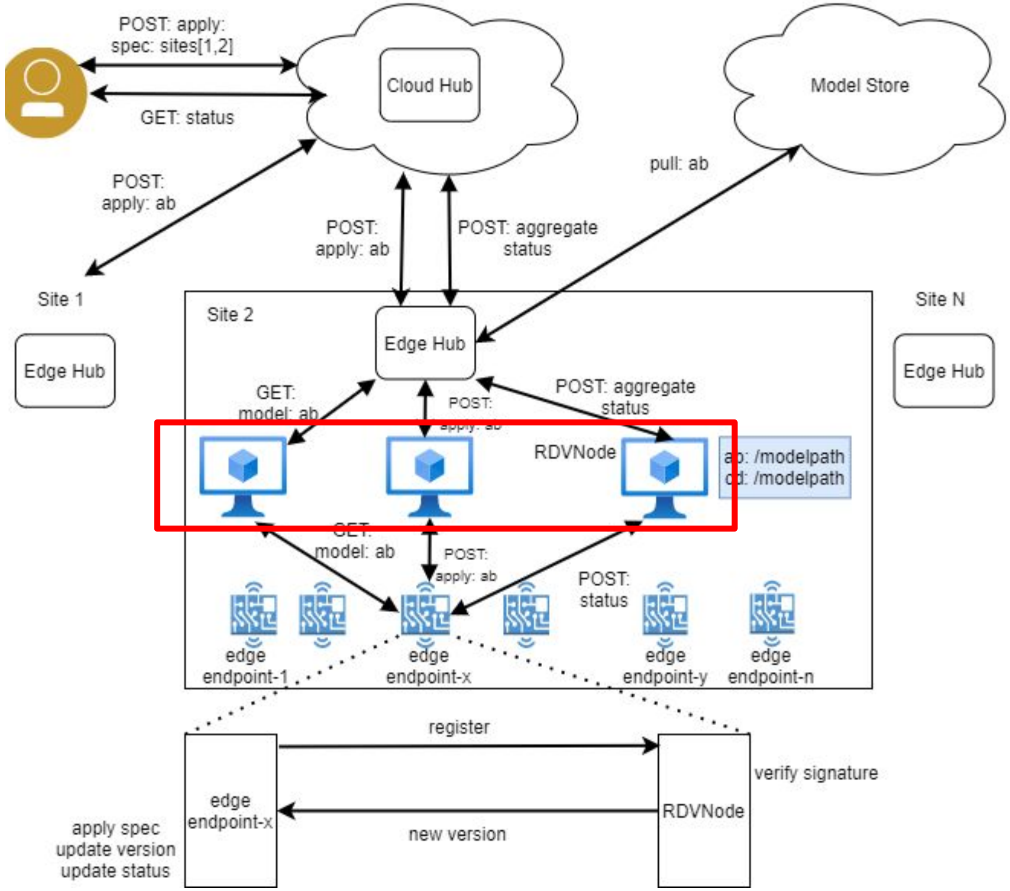
EdgeRDV Benefits: Scalability



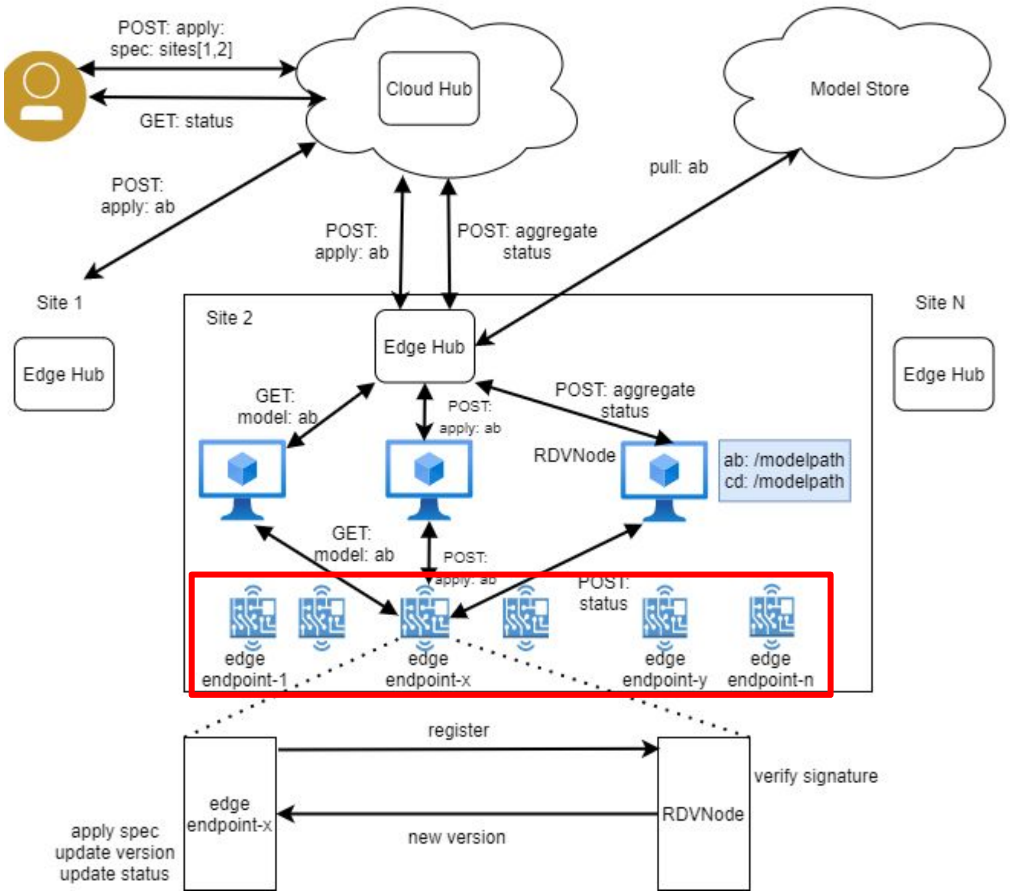
EdgeRDV Benefits: Scalability



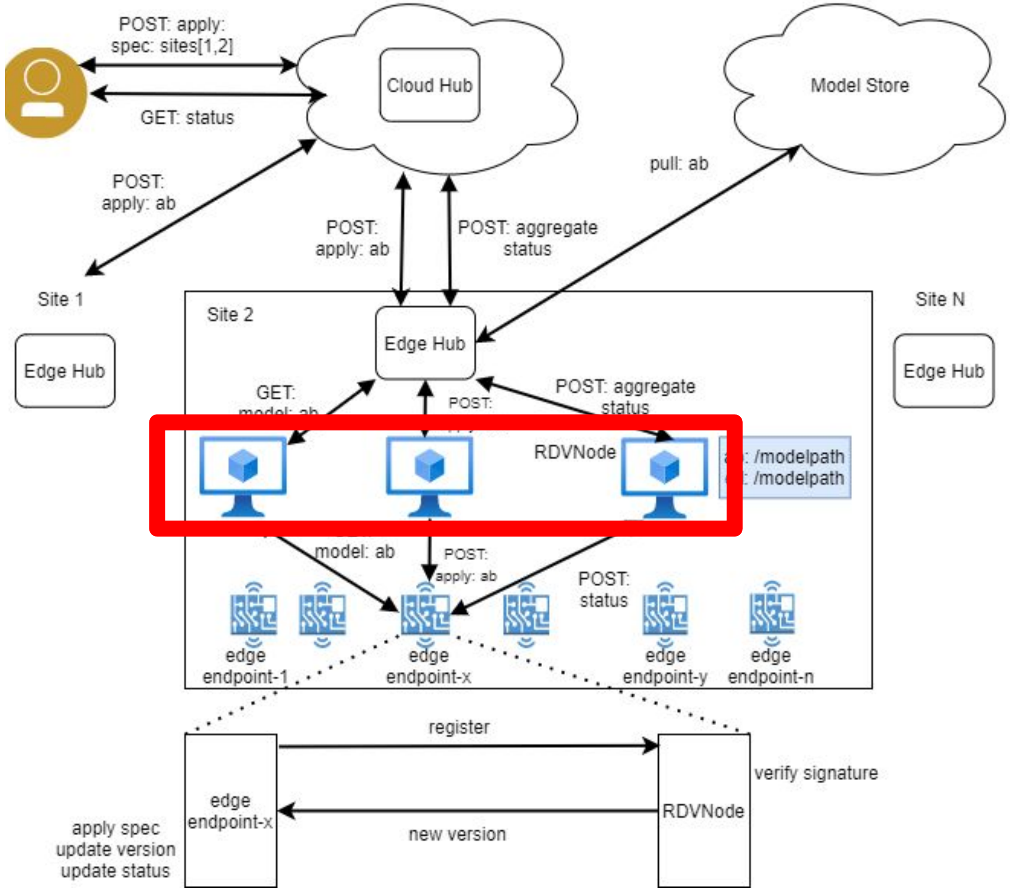
EdgeRDV Benefits: Scalability



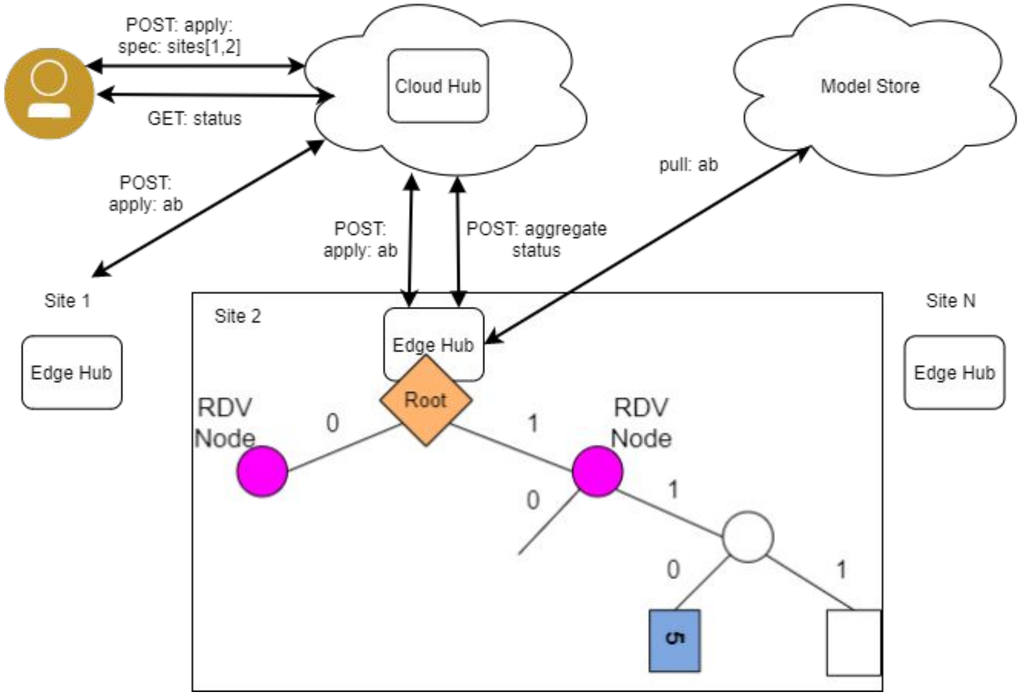
EdgeRDV Benefits: Scalability



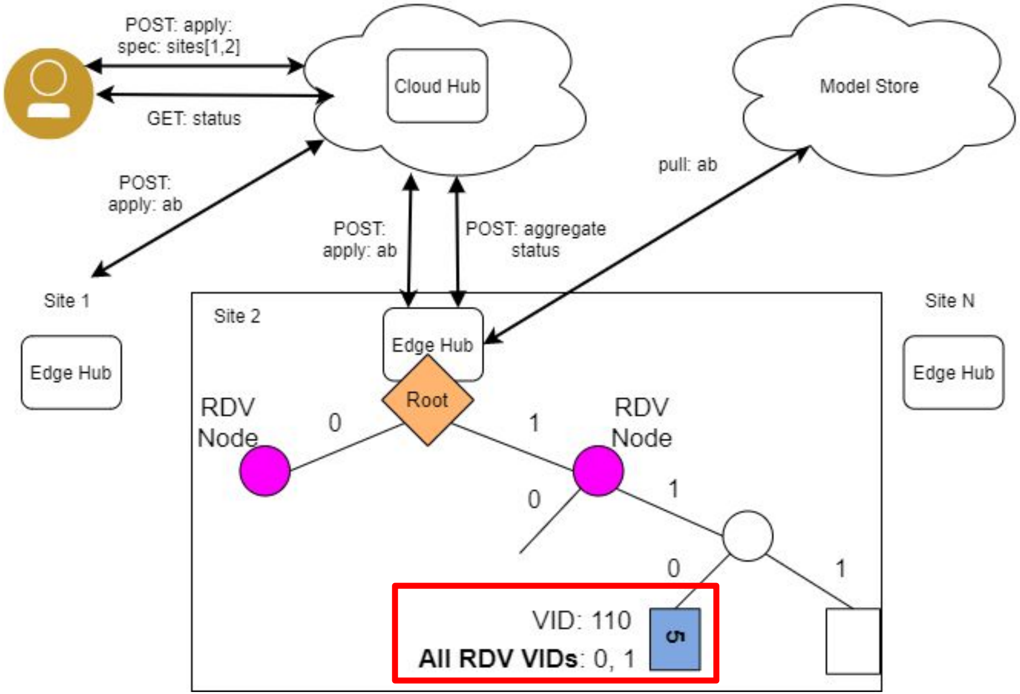
EdgeRDV Benefits: Scalability



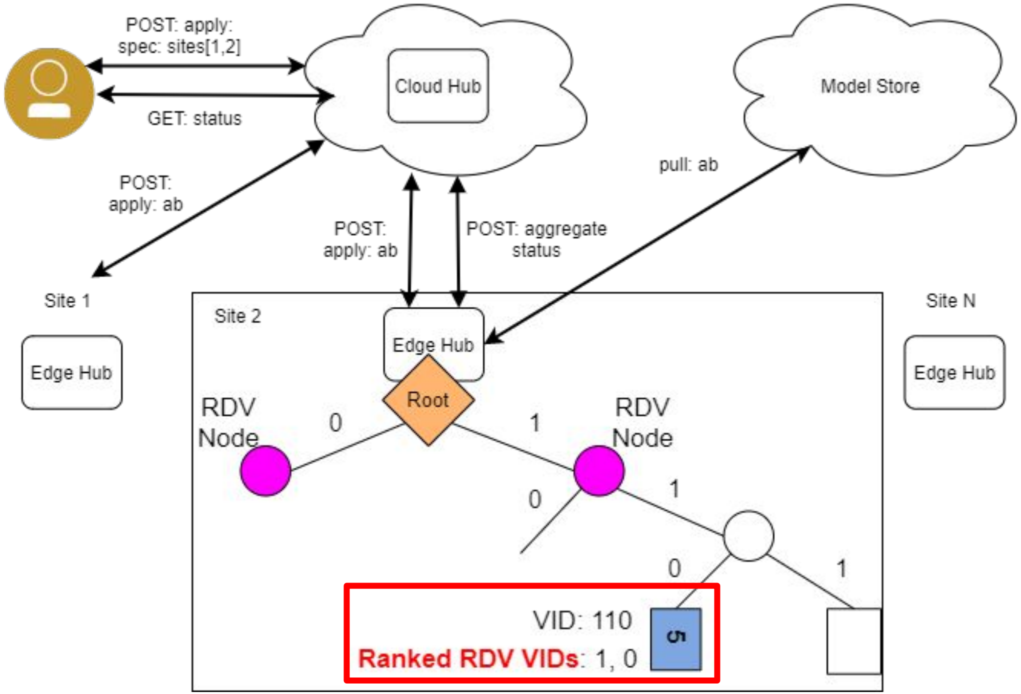
EdgeRDV Benefits: Scalability



EdgeRDV Benefits: Scalability

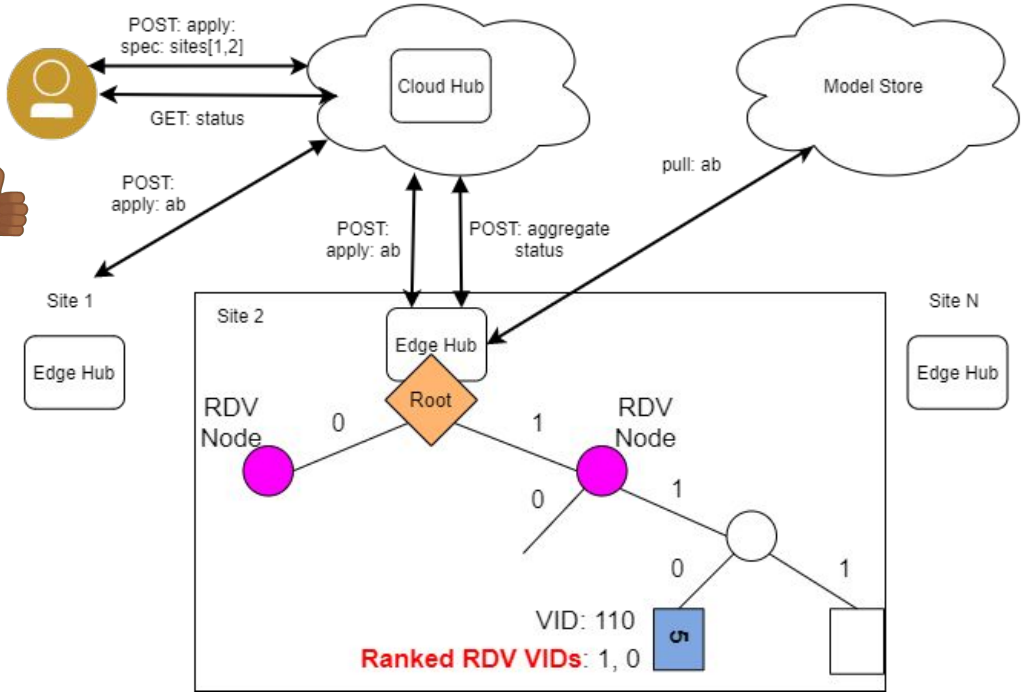


EdgeRDV Benefits: Scalability



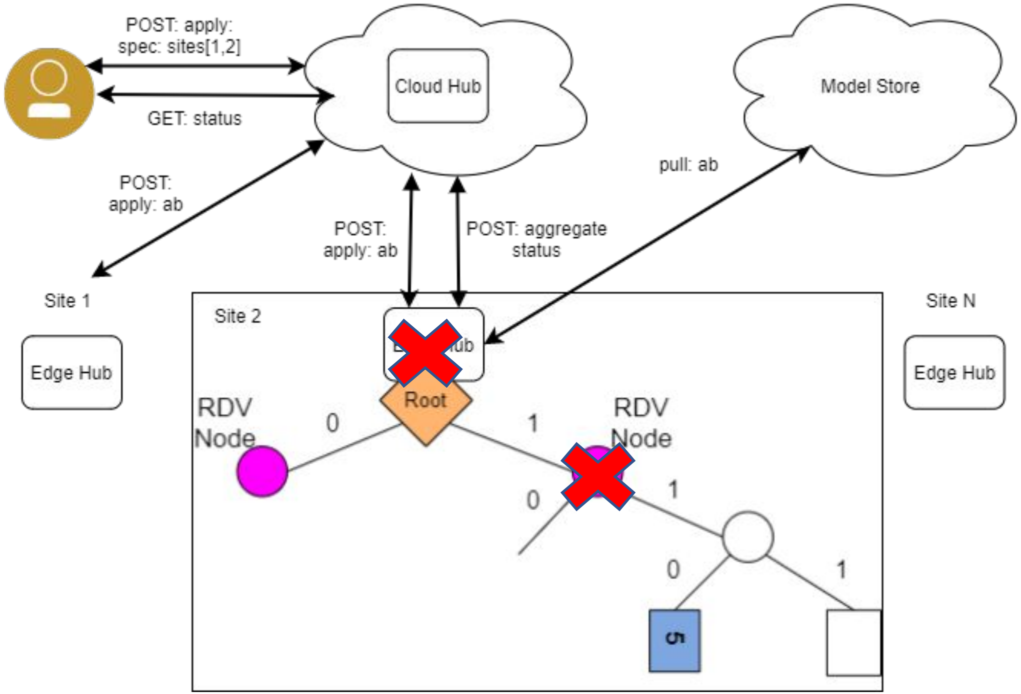
EdgeRDV Benefits: Scalability

- Priority list of RDV nodes 👍🏾
- Minimal edge hub overhead 👍🏾

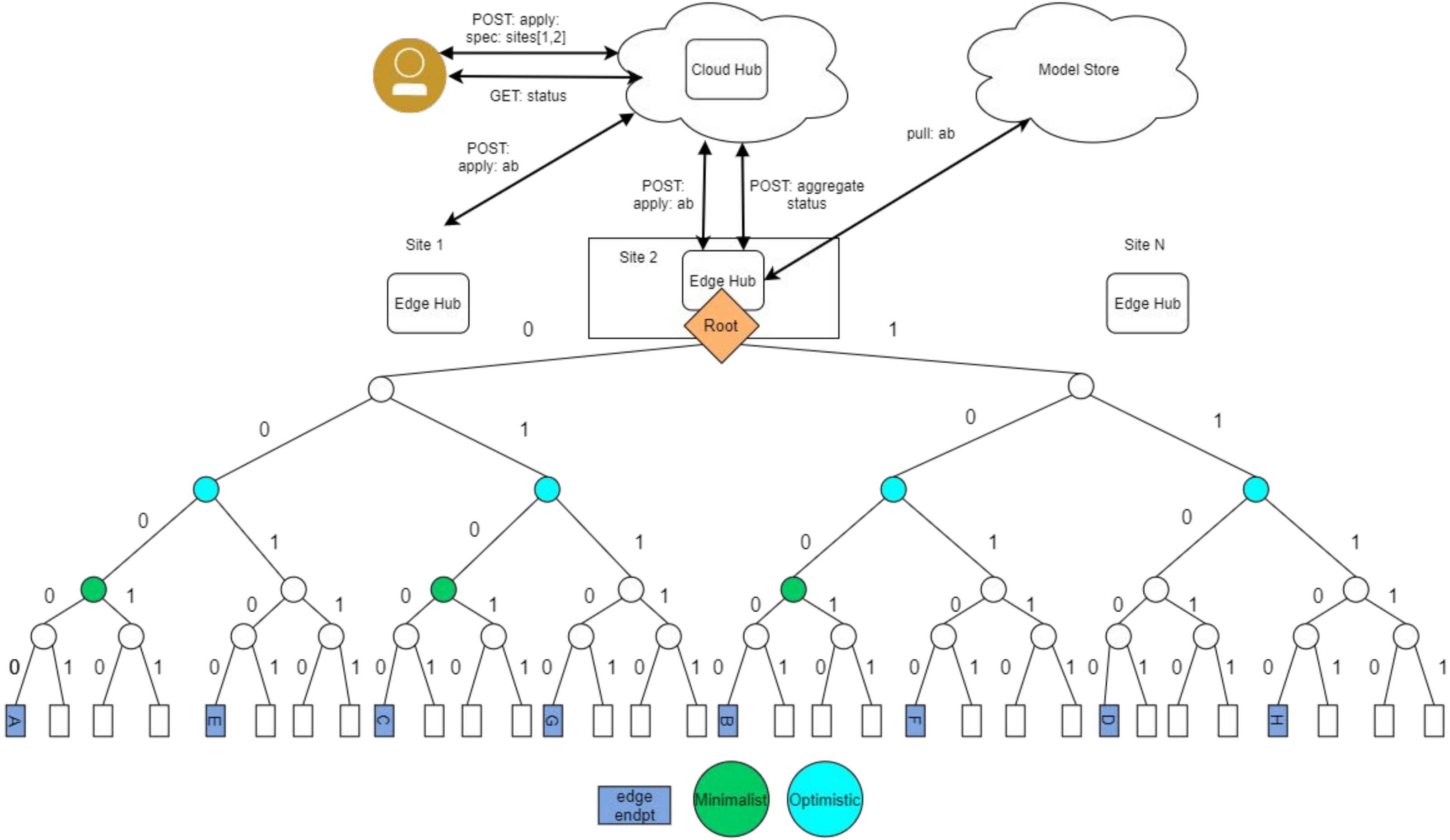


EdgeRDV Benefits: Multi-level Failure Resilience

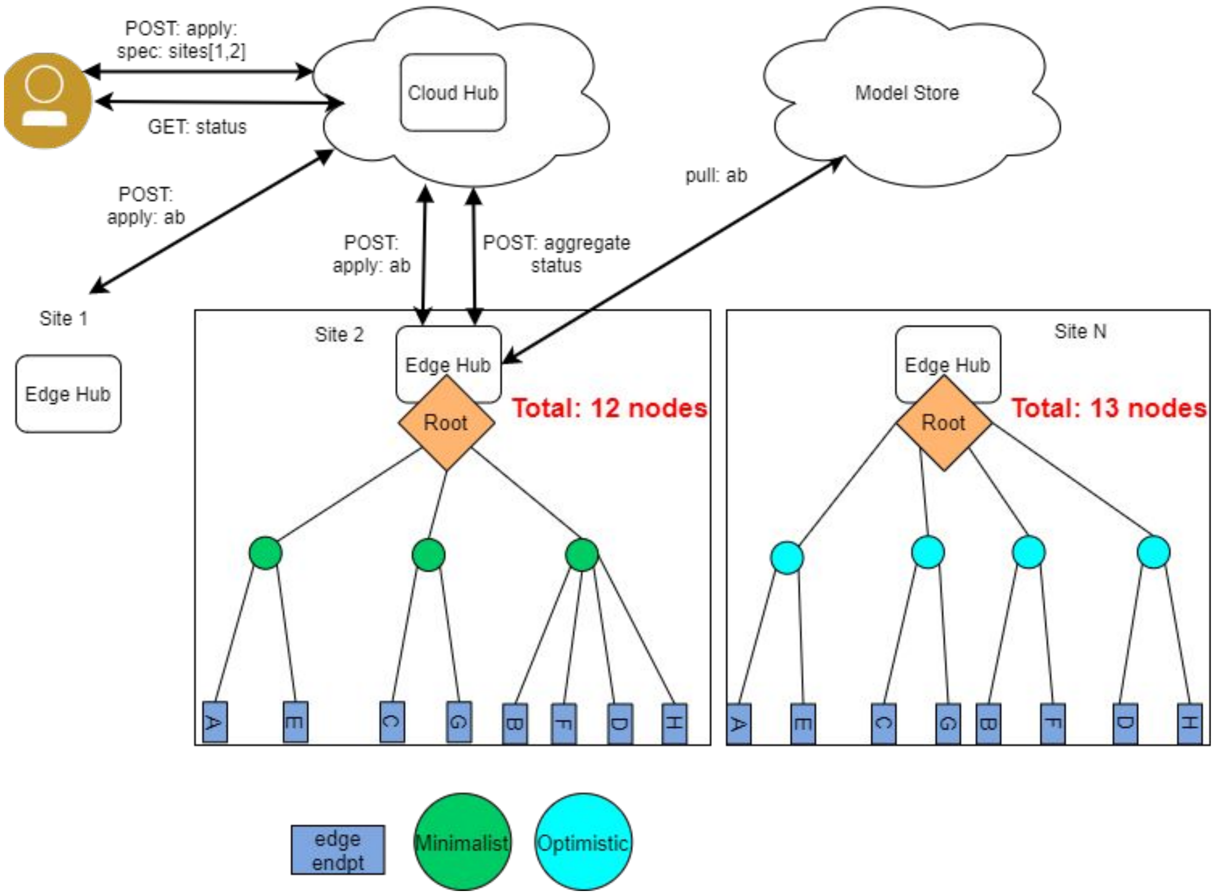
- Multi-level caching 👍🏾
- Multiple RDV nodes 👍🏾
- Ranked list of backup RDVs 👍🏾



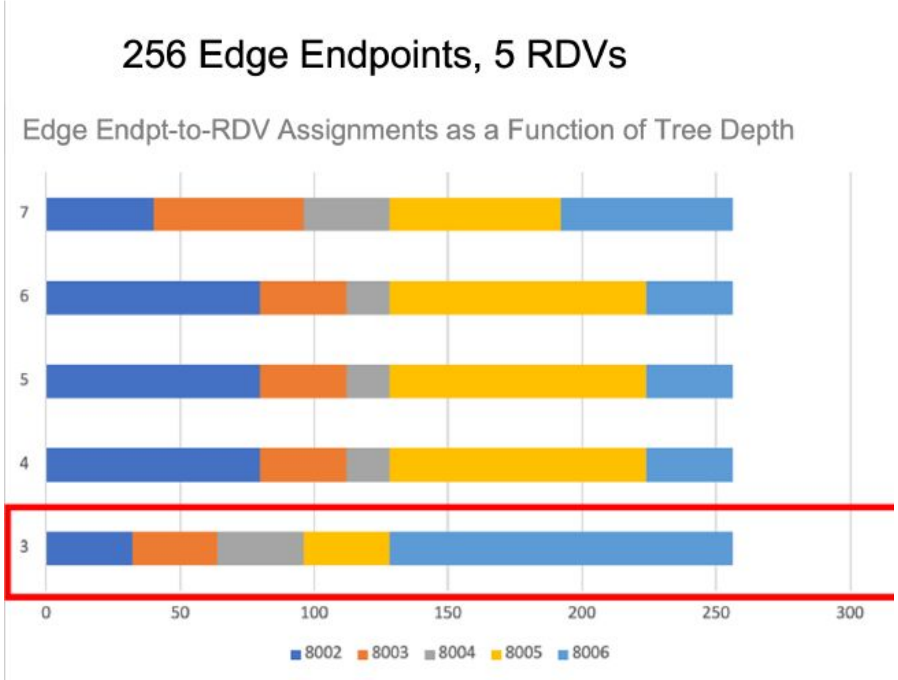
Challenge with RDV nodes - How many and where?



Challenge with RDV nodes - How many and where?

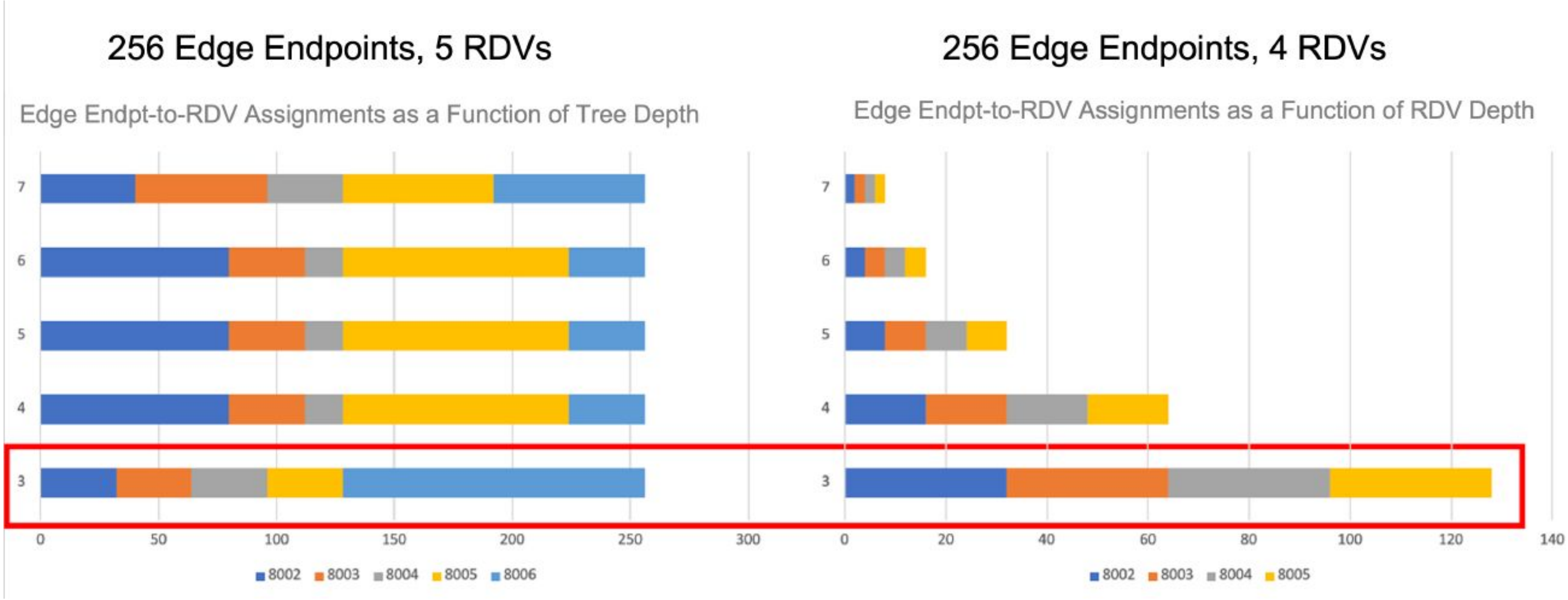


Simulation Results
Keeping the network balanced



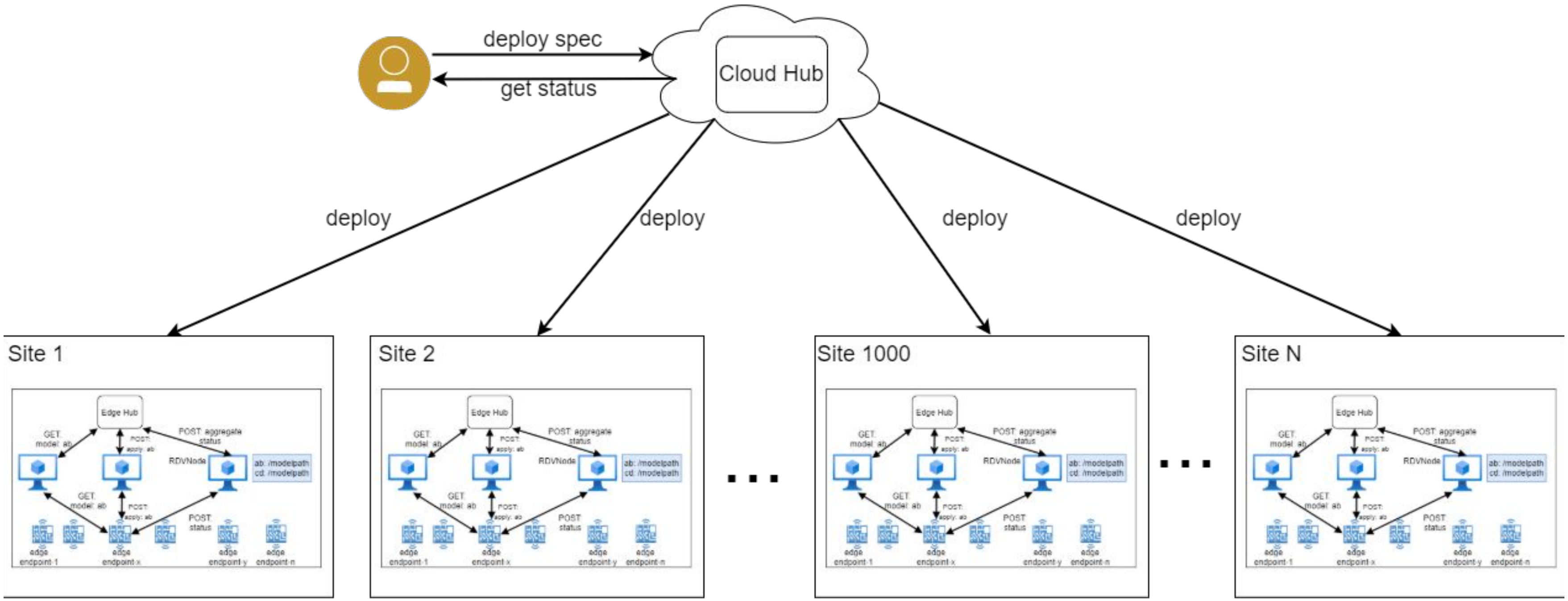
Minimalist

Simulation Results
Keeping the network balanced

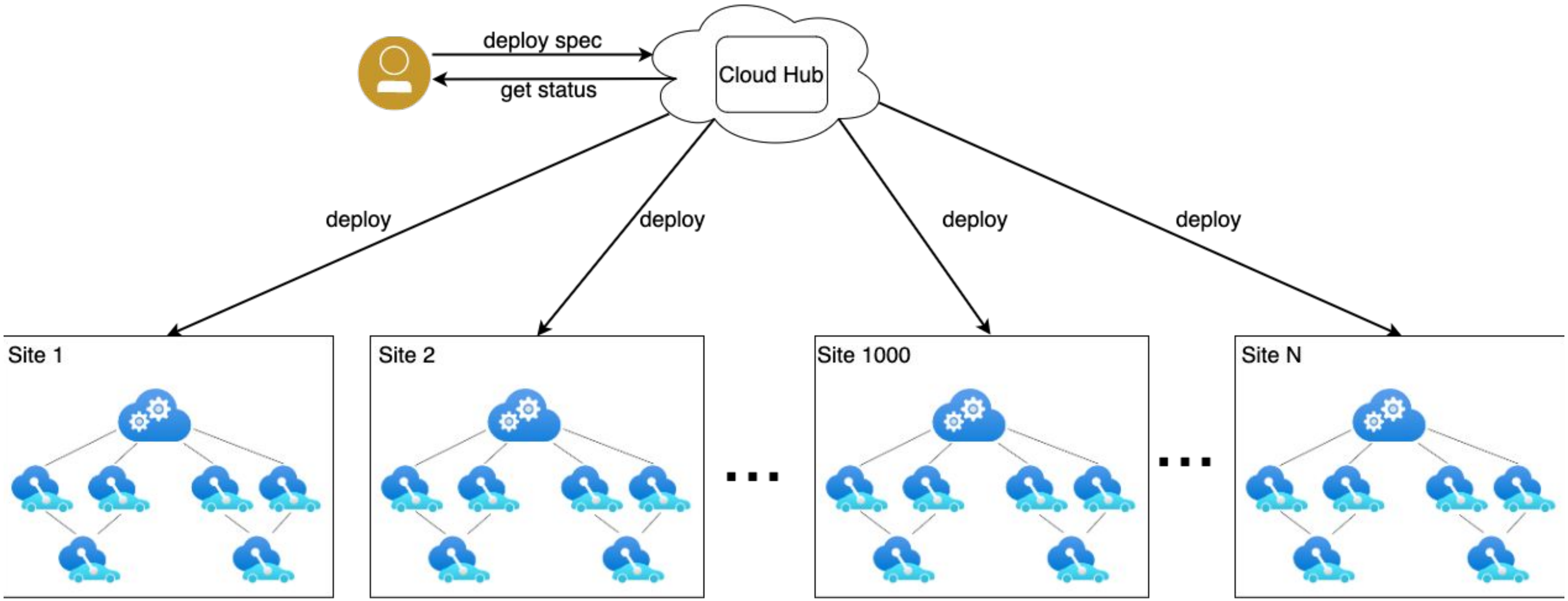


Minimalist **Optimistic**

EdgeRDV at Scale



EdgeRDV at Scale



Motivation

State-of-the-art

Methodology

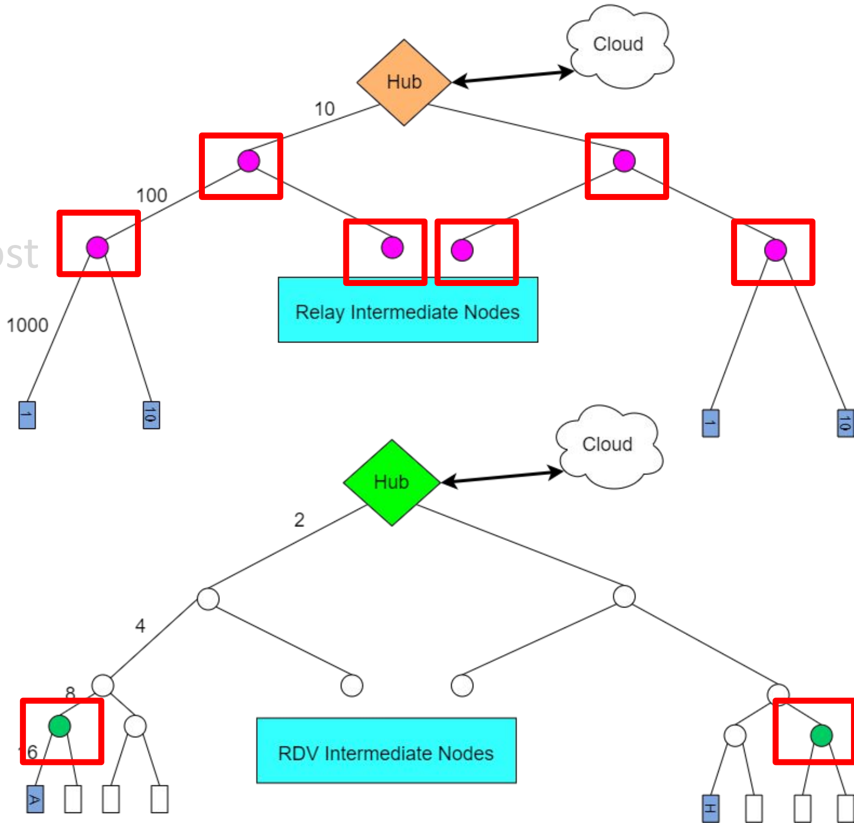
Analysis/Evaluation

Experiments

- **Setup**
  - Docker Linux container (Ubuntu 20.04)
  - RDV node coverage: 10%
  - Run components as processes on the same host
  - Scalability analysis up to 667K endpoints

Experiments

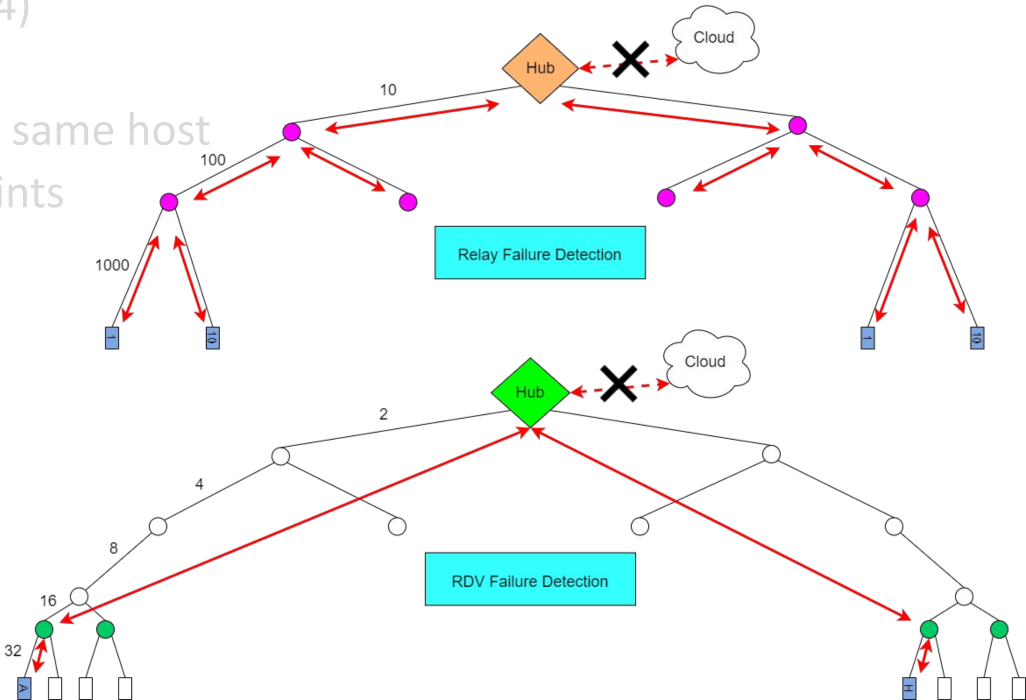
- **Setup**
  - Docker Linux container (Ubuntu 20.04)
  - RDV node coverage: 10%
  - Run components as processes on the same host
  - Scalability analysis up to 667K endpoints
- **Metrics**
  - **Intermediate nodes**



Experiments

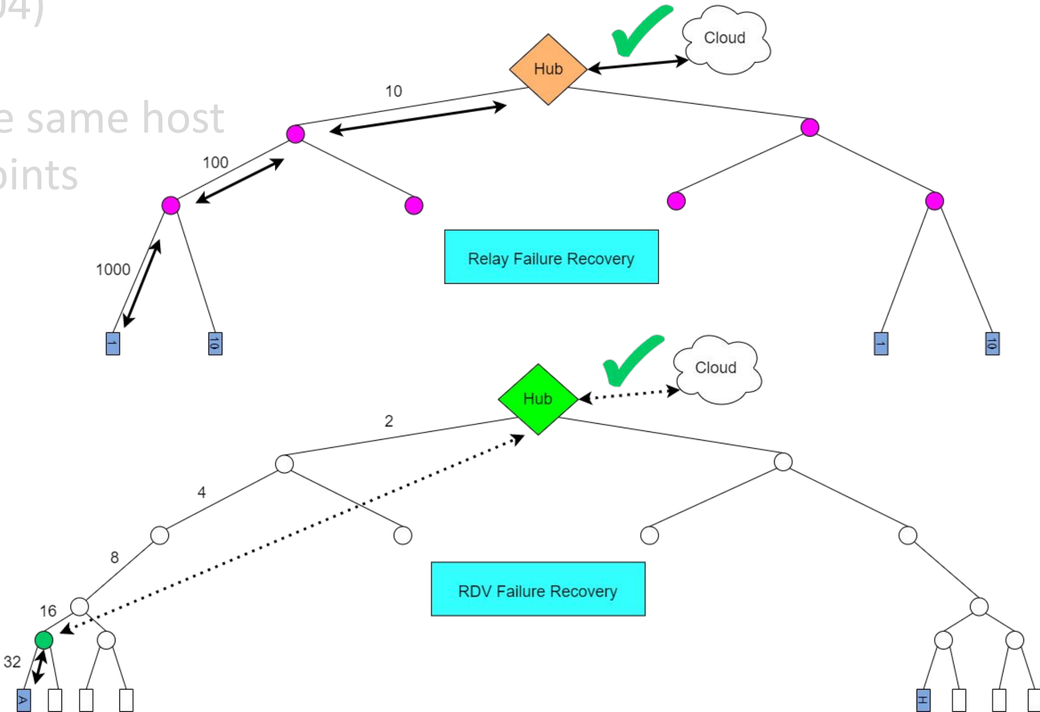
- **Setup**
  - Docker Linux container (Ubuntu 20.04)
  - RDV node coverage: 10%
  - Run components as processes on the same host
  - Scalability analysis up to 667K endpoints

- **Metrics**
  - Intermediate nodes
  - **Total physical edges**



Experiments

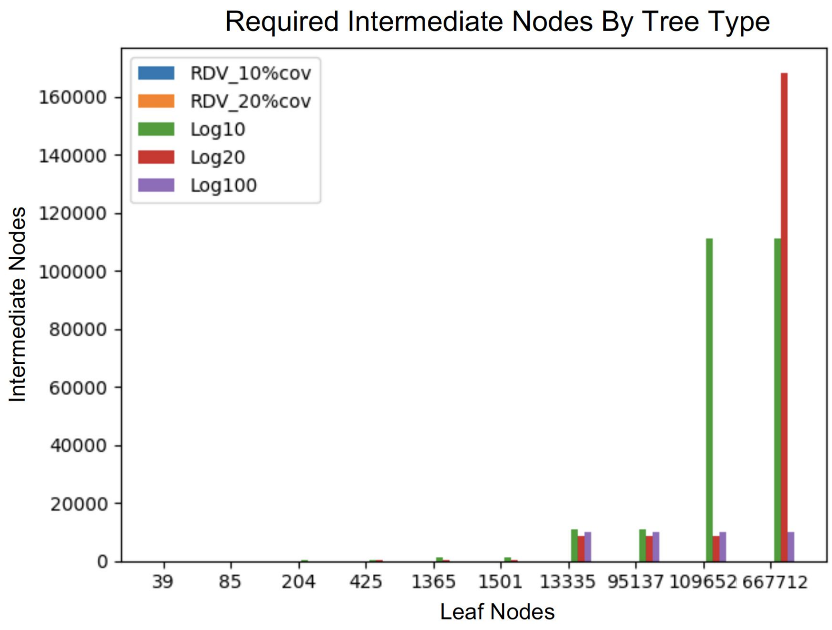
- **Setup**
  - Docker Linux container (Ubuntu 20.04)
  - RDV node coverage: 10%
  - Run components as processes on the same host
  - Scalability analysis up to 667K endpoints
- **Metrics**
  - Intermediate nodes
  - Total physical edges
  - **Control/data messages**



Analysis: Minimizing Intermediate Nodes

Minimizing intermediate nodes

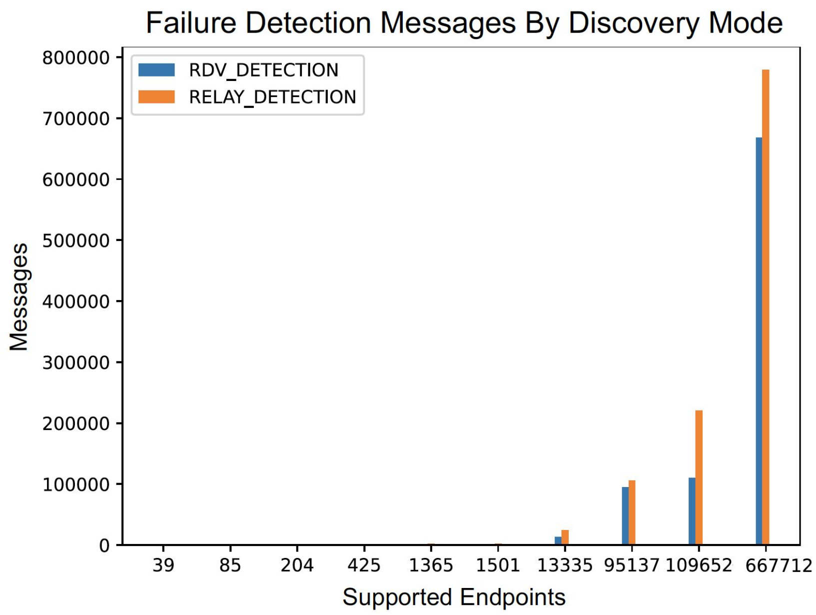
- **Experiment**:
  - To estimate the number of intermediate nodes required to get messages to the hub
- **Observation(s)**:
  - RDV method has **10-1000x fewer intermediate nodes** depending on targeted coverage
  - RDV selection algorithm is **sensitive** to coverage and costs



Analysis: Detecting and Adapting to Failures

Time to acquiesce/adapt to failures

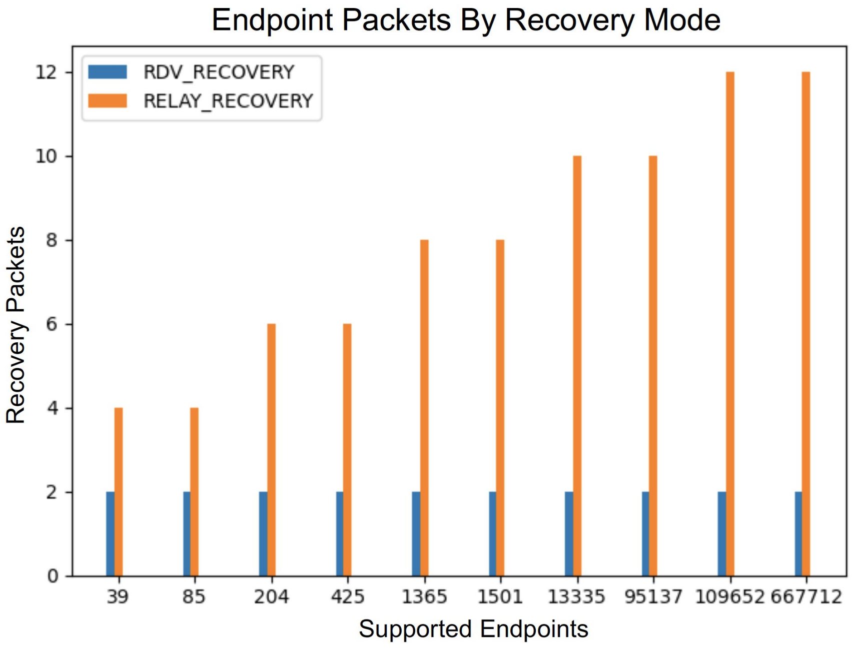
- **Experiment**:
  - To understand the number of messages required to update all endpoints of single point of failure (i.e., the hub)
- **Observation(s)**:
  - RDV failure messages **scale gracefully** with number of endpoints



Analysis: Recovering from Failures

Model download in failure recovery

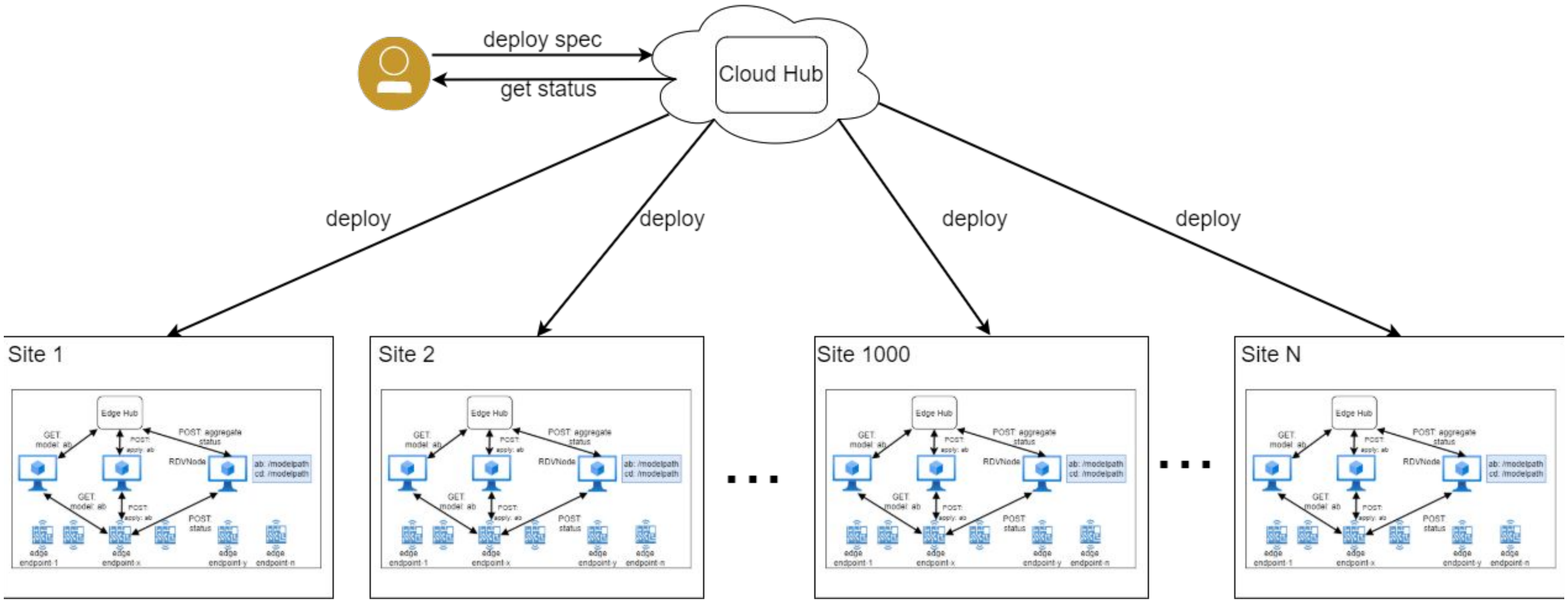
- **Experiment**:
  - To understand the number of control/data messages required to pull a new update during failure recovery
- **Observation(s)**:
  - RDV method **scales constantly** with number of endpoints



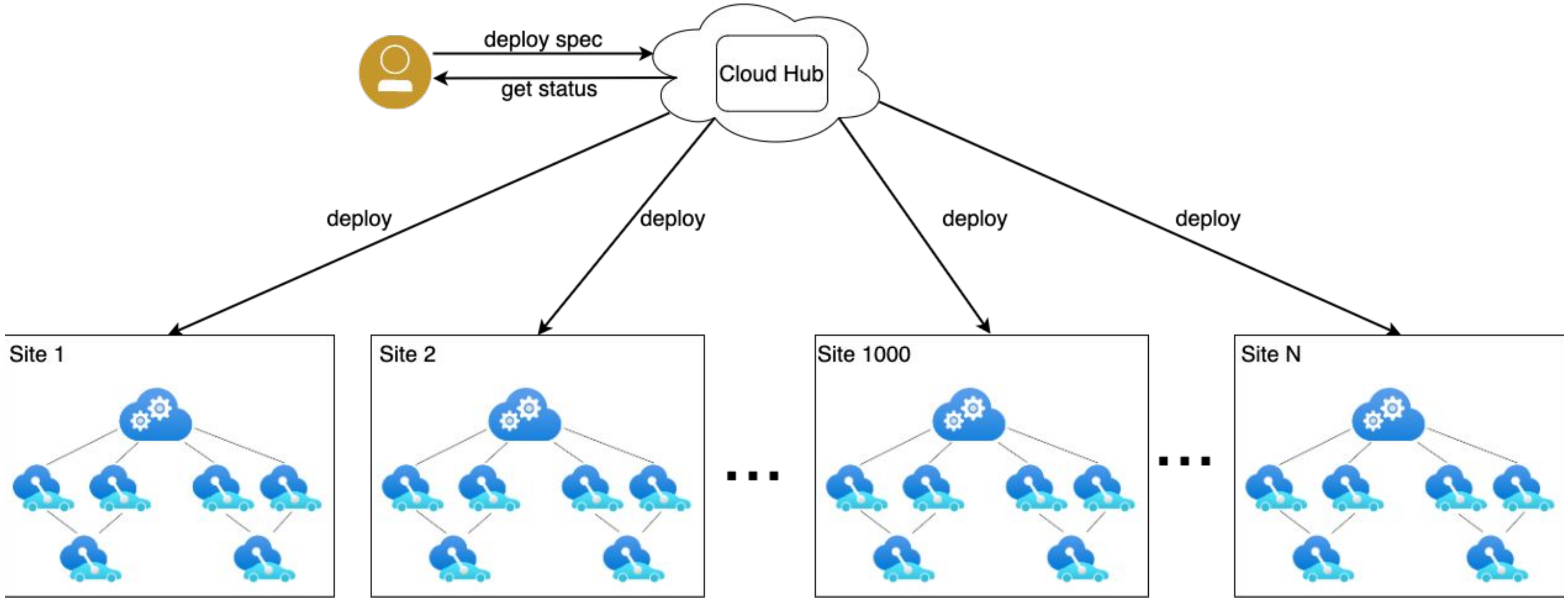
Next Steps

- Hardware implementation
  - Scalability
  - Bandwidth consumption
  - Data transfer
  - RTT optimization
- Novel applications
  - Digital agriculture

Conclusion

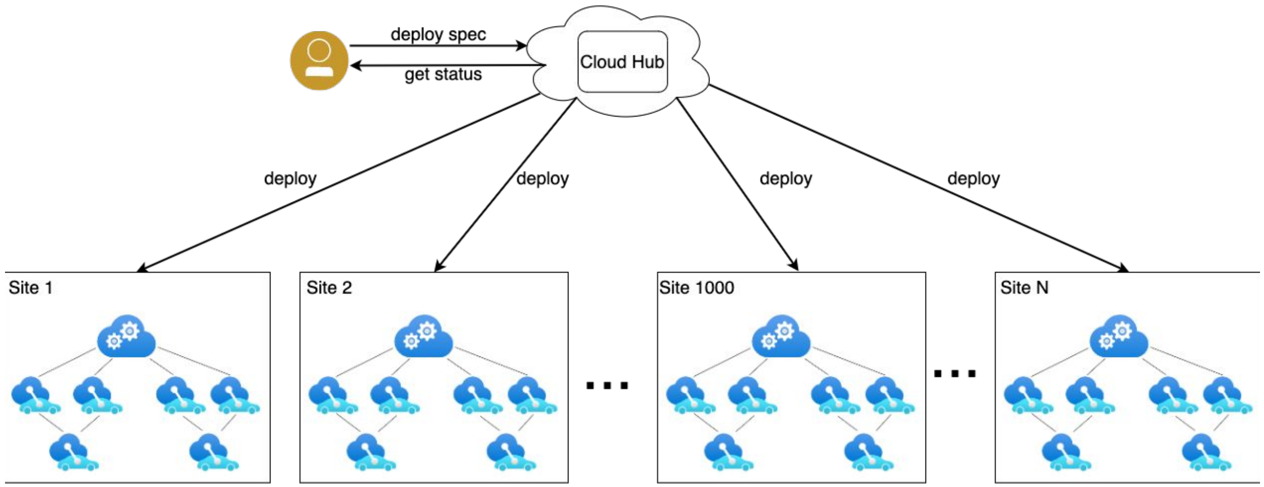

deploy spec
get status
Cloud Hub
deploy
deploy
deploy
deploy
Site 1
Site 2
Site 1000
Site N
Edge Hub
GET model: ab
POST: apply: ab
POST: aggregate status
RDVNode
ab: /modelpath
cd: /modelpath
POST: status
edge endpoint-1
edge endpoint-x
edge endpoint-y
edge endpoint-n

Conclusion



deploy spec
get status
Cloud Hub
deploy
deploy
deploy
deploy
Site 1
Site 2
...
Site 1000
...
Site N

Conclusion



- 1-3 orders of magnitude fewer intermediate nodes
- Scalable infrastructure bootstrapping
- Adjustable network resilience
- Efficient resource usage

Thank You

Gloire Rubambiza
gloire@cs.cornell.edu
https://rubambiza.github.io