IEEE WISEE 2026 · STINT WORKSHOP

On the Distribution of DTN Reachability Information

A bundle carries the name `ipn:13.1`. Somebody has to know where that agent lives. Should that knowledge be **pushed to every node in advance**, or **pulled when needed**? (assuming **existing** Internet protocols)

Juan A. Fraire · Inria, INSA Lyon, CITI, UR3720, Villeurbanne, France · D3TN GmbH, Dresden, Germany
Funded by the DARKSOL project · models and configurations are open

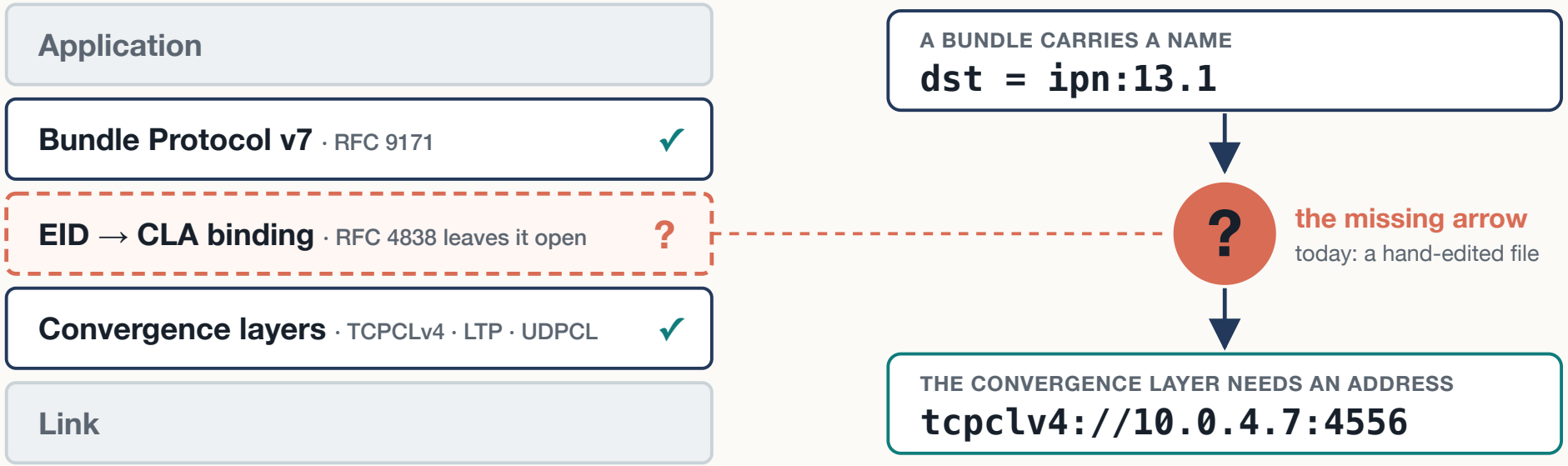
DARKSOL: operating a cloud in space

Space run like a data center. Satellites, rovers and probes shared by several tenants and operated with standard cloud tooling, Kubernetes included.

Delay and disruption are the default. DTN is the network underneath. A software update to a Mars robot takes minutes to arrive.

The network has to configure itself. Cloud tooling assumes it does. Future space computing will need such automation.

The missing arrow in the DTN stack



BPv7 and the convergence layers assume the EID-to-CLA binding is already there. RFC 4838 (the DTN architecture) never says how it gets there. IP routing and DNS do this job for the Internet; in DTN it is done by hand.

HOW THE BINDING IS PROVISIONED TODAY

Three stacks, one hand-written table

ION (NASA JPL)

Three admin files per node: `.bprc`, `.ipnrc`, `.ionrc`

```
a outduct tcp 10.0.4.7:4556 tcpclo  # .bprc
a plan 13 tcp/10.0.4.7:4556         # .ipnrc
a contact +1 +3600 2 13 100000      # .ionrc
a exit 100 199 ipn:13.0             # .ipnrc
```

µD3TN (D3TN)

Pushed at runtime over AAP 2.0 to the `config` endpoint; not persistent

```
aap2-config --socket ud3tn.aap2.socket \
  --schedule 1 3600 100000 \
  -r ipn:14.0 \
  ipn:13.0 mtcp:10.0.4.7:4224
```

HDTN (NASA Glenn)

Two JSON files per node: `hdtn_config.json` and a contact plan

```
"outductVector": [{
  "convergenceLayer": "tcpcl_v4",
  "nextHopNodeId": 13,
  "remoteHostname": "10.0.4.7",
  "remotePort": 4556 }]
"contacts": [{ "source": 2, "dest": 13,
  "startTime": 0, "endTime": 2000,
  "rateBitsPerSec": 1000000, "owlt": 1 }]
```

Different syntax, same model: an operator writes the binding and pre-places it on every node.

Sources: ION man pages and test suite, µD3TN docs, HDTN repository.

Why the hand-written table stops working now

Cislunar buildout

LunaNet and Artemis: dozens of mobile assets from several agencies.

Multi-operator by construction

Nodes belonging to different organizations.

Bindings change

Radios swapped, relays retasked, gateways moved.

Space cloud Kubernetes-style operation on top: nodes join, move and leave with no operator editing a table.

Static provisioning is not adquate for large emerging space networks.

Two proposals in 2025, from opposite directions

Push it: BGP-like

Feldmann et al., WiSEE 2025. BGP UPDATE messages carry DTN EID reachability, reusing path-vector machinery operators already run.

Pull it: DNS-like

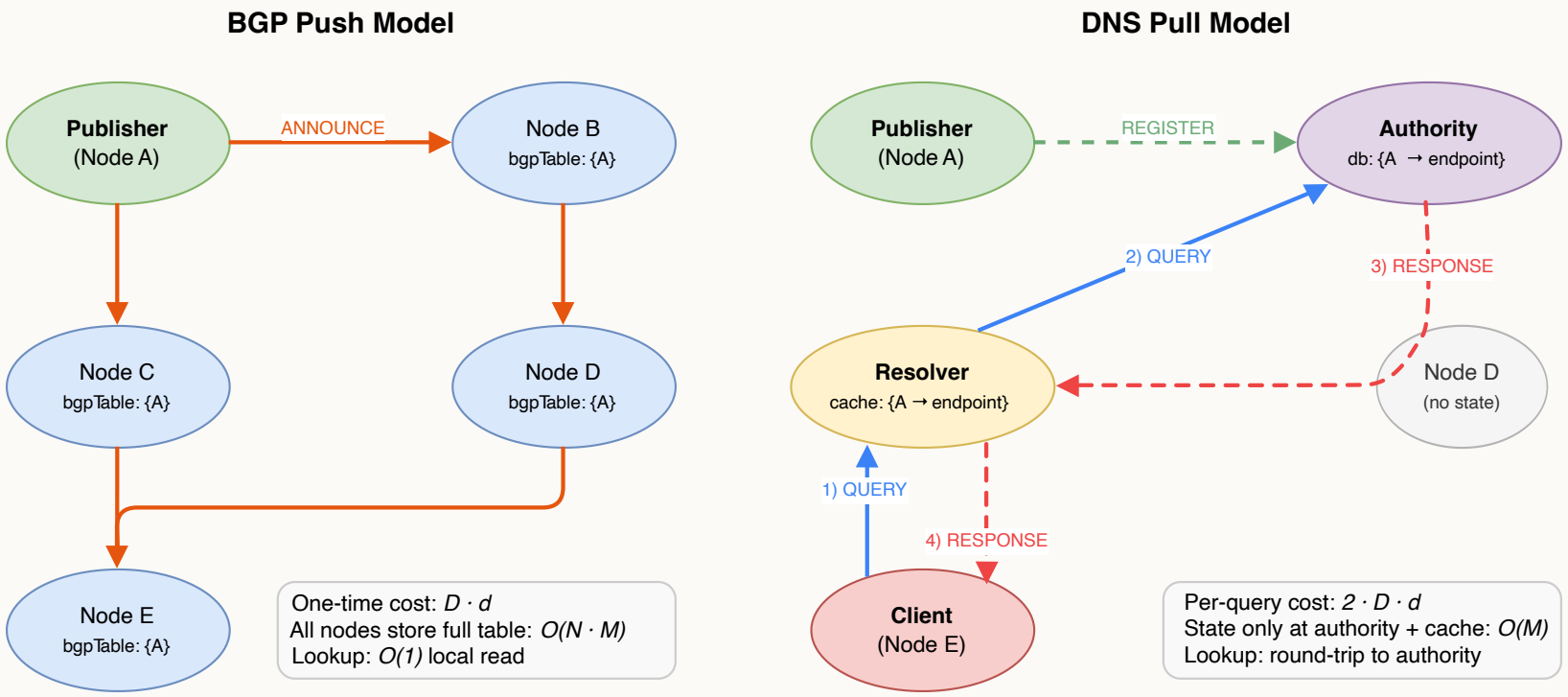
Kline, `draft-ek-dtn-ipn-arpa-00`, Nov 2025. An `ipn.arpa` zone resolves `ipn`-scheme EIDs through the DNS hierarchy the Internet already runs.

- The Internet runs both, side by side: **routes are pushed**, **names are pulled**.
- The direction is easy to guess. The numbers were unknown.

> **This paper puts numbers on the trade-off.**

Scope: Feldmann et al. target edge BPA configuration. We deliberately generalize that push paradigm to multi-hop.

Push and pull, side by side



Left: every binding is replicated to every node; a lookup is a local read. **Right:** bindings stay at an authority and every lookup is a round trip. Ordinary nodes such as D only relay the traffic and hold no binding state.

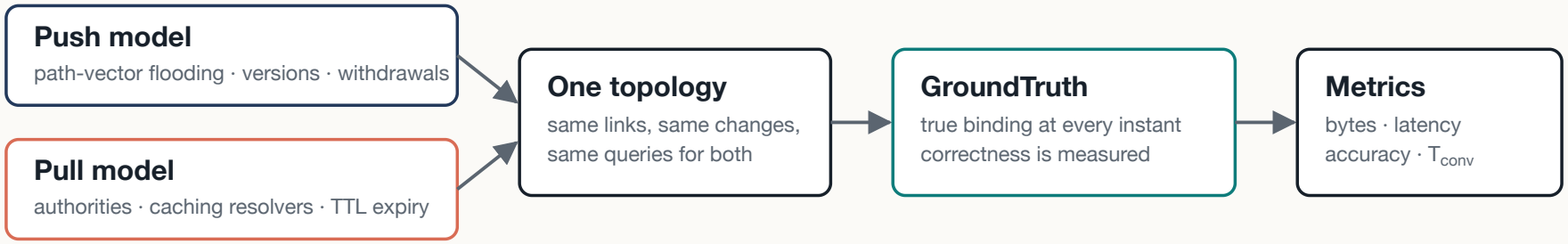
The two cost structures

	Push (BGP-like)	**Pull (DNS-like)**
Publish one binding	$O(N)$ messages, network-wide flood	$O(1)$, tell the authority
Look one up	local read, ≈ 0	$2 \cdot D \cdot d$, **every time**
State in the network	$O(N \cdot M)$, full replication	$O(M)$, one copy plus caches
A binding changes	flood again	update the authority; caches go stale
Is the answer current?	yes, after convergence	as fresh as the TTL allows

Push pays once, everywhere. Pull pays every time, only where needed.

N: nodes in the network · *M*: published EIDs (bindings) · *D*: network diameter in hops · *d*: one-way link delay. *D*·*d* is one crossing of the network, *2*·*D*·*d* a round trip to the authority.

What we built, and what we predicted



Fair by construction. Pull runs over a pre-routed underlay (RTT = 2·D·d), exactly as real DNS runs over routed IP. Routing is a different question.

13 experiments | 25–400 nodes | 10–500 EIDs | d = 10 ms → 20 s | 3 deployments | 3–10 seeds each | OMNeT++ 6.3

H1 · low delay
pull wins on overhead when `2·D·d < 1 s`

H2 · high delay
push wins after very few queries once `2·D·d` reaches tens of seconds

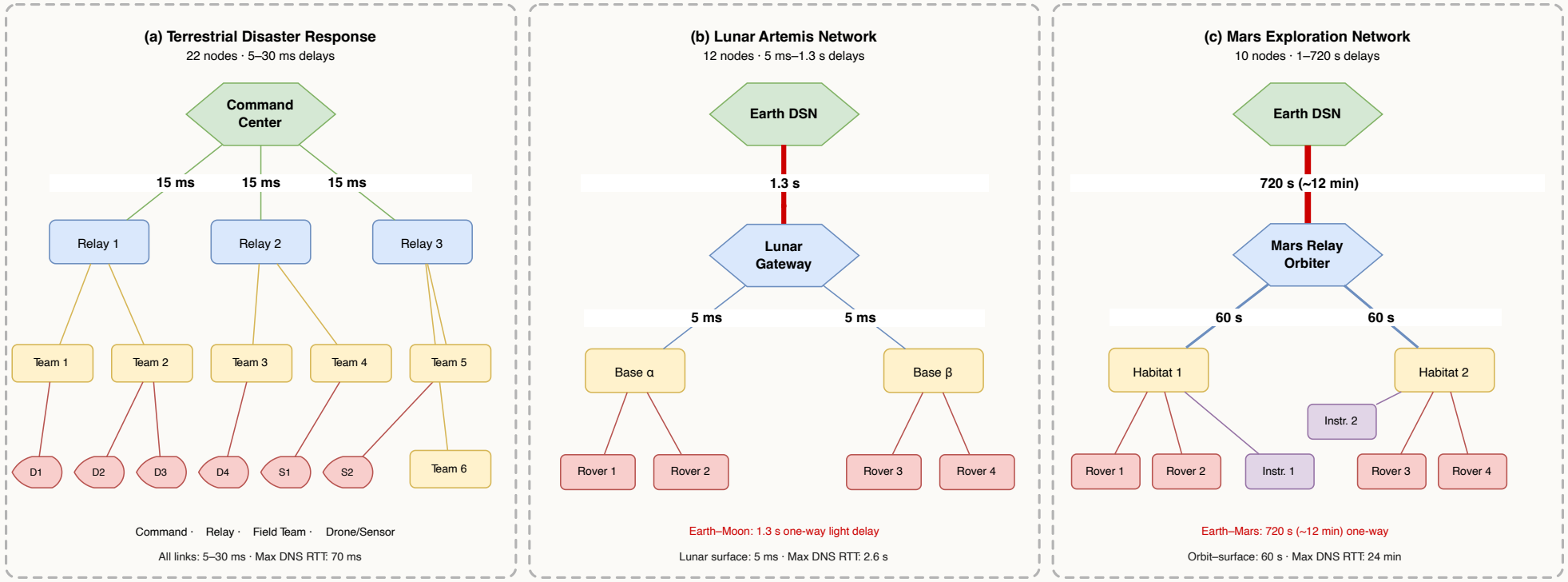
H3 · cislunar
a crossover near 1–2 s one-way: pull saves bytes, push already saves time

H4 · binding changes
pull accuracy falls as $R = T_{conv} / \Delta_{churn}$ grows; push does not care

Hypotheses written before the sweeps. All four held; the interest is where the boundaries fall.

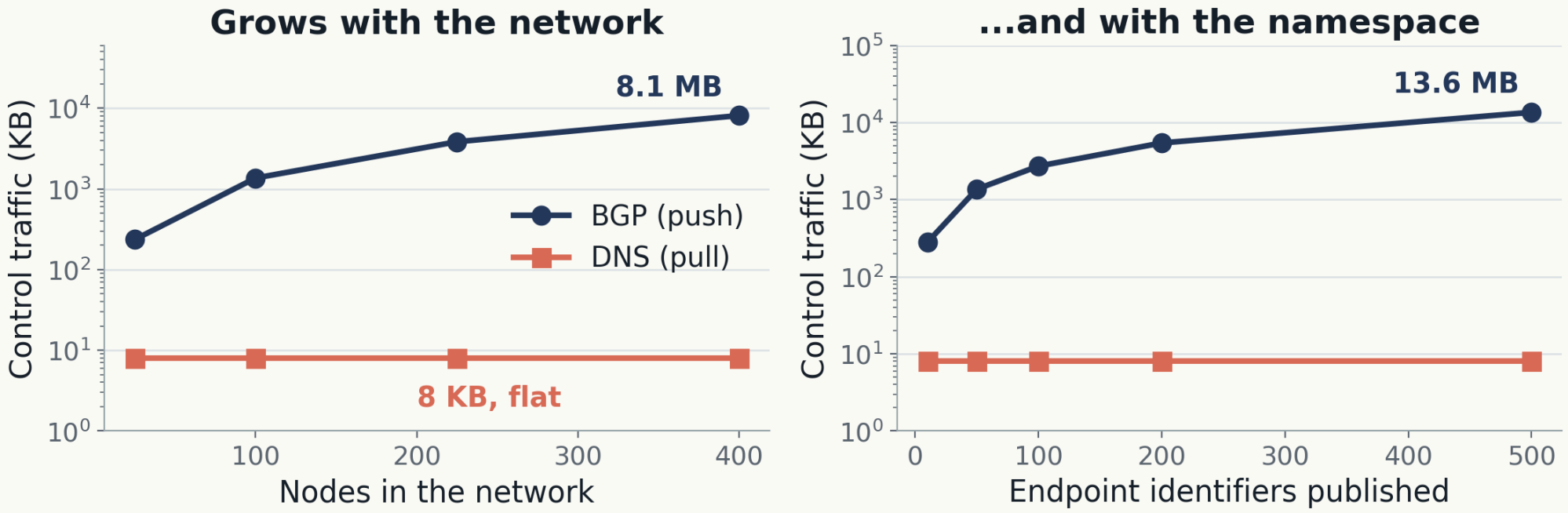
Where we ran them

- *Synthetic grids* and three *scenarios*:



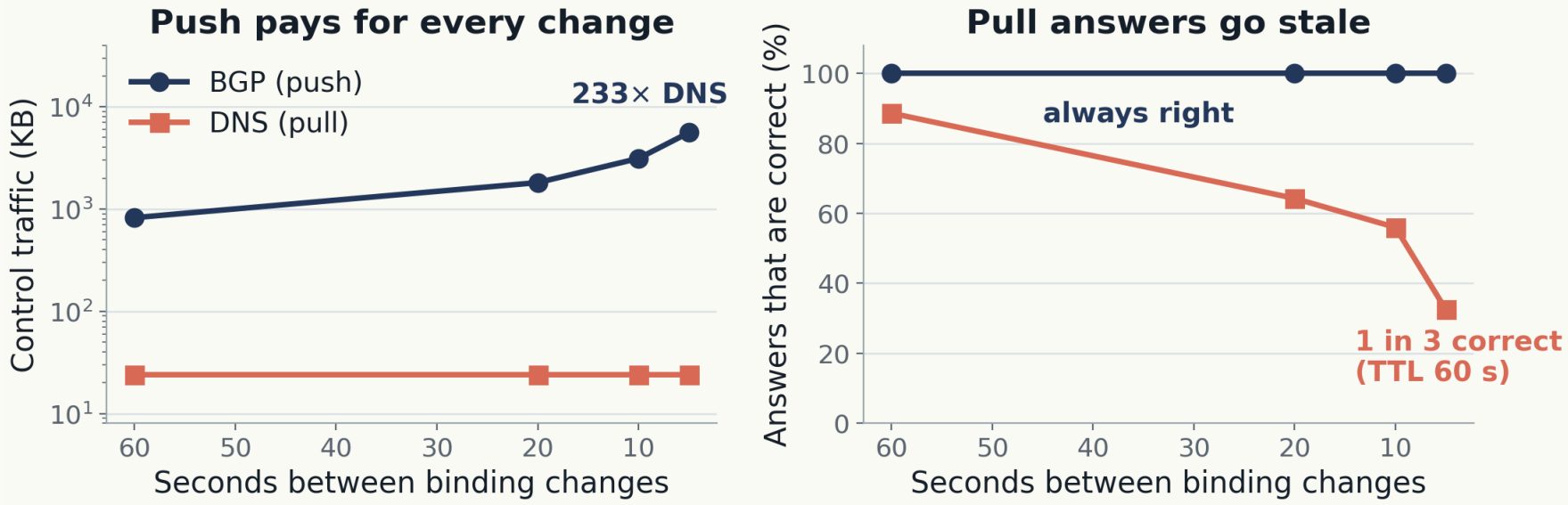
Terrestrial disaster response (22 nodes, 5–30 ms) · **lunar Artemis** (12 nodes, a 1.3 s Earth–Moon trunk) · **Mars exploration** (10 nodes, a 720 s trunk over 60 s orbit-to-surface hops). Synthetic grids carry the controlled sweeps.

Result 1: push does not scale quietly



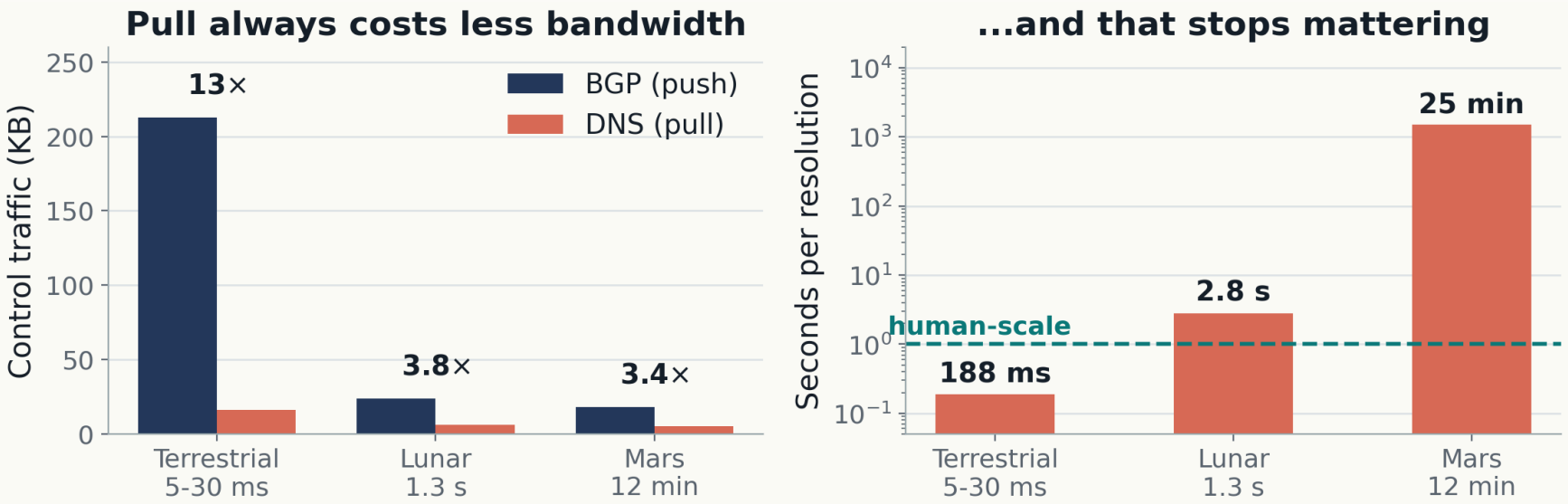
On a constrained space link, the push table is the traffic.

Result 2: pull is cheap because it may be wrong



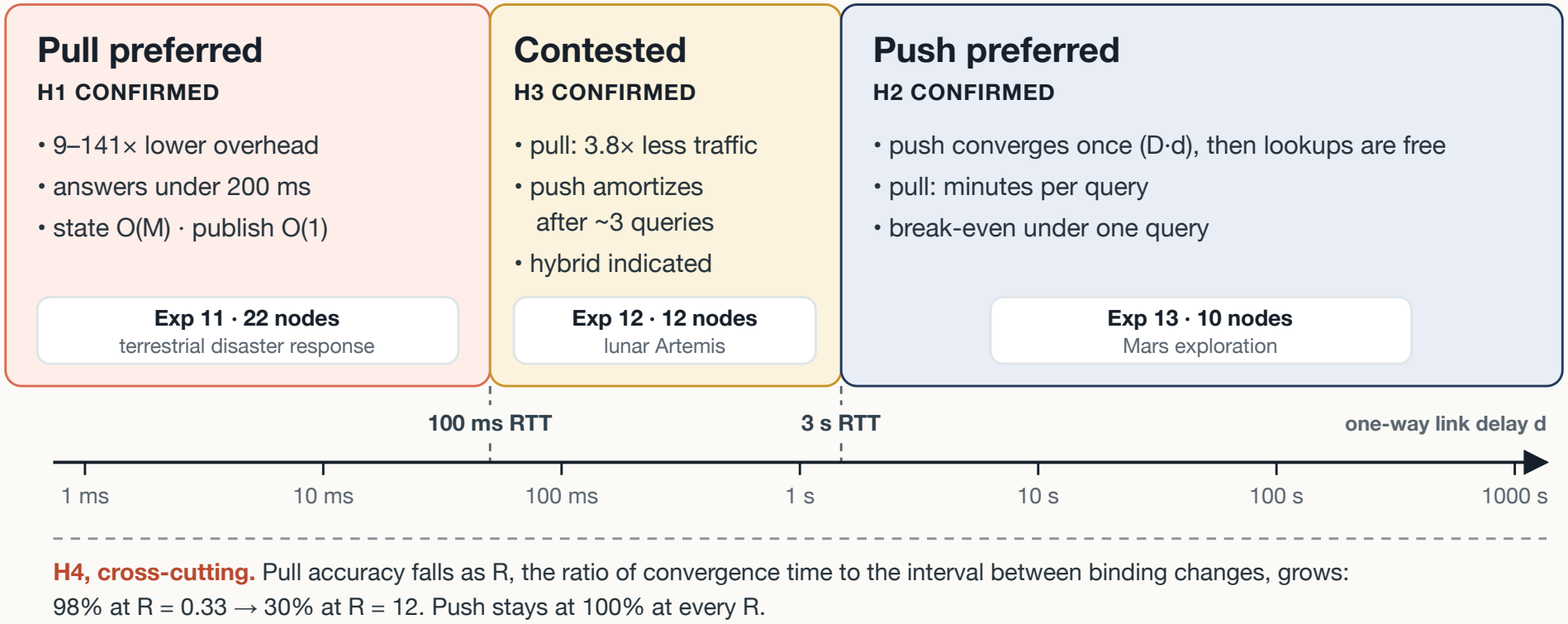
Choosing pull means choosing an accuracy budget.

Three deployments, two dimensions



Pull's bandwidth edge stays constant. Its latency cost moves five orders of magnitude.

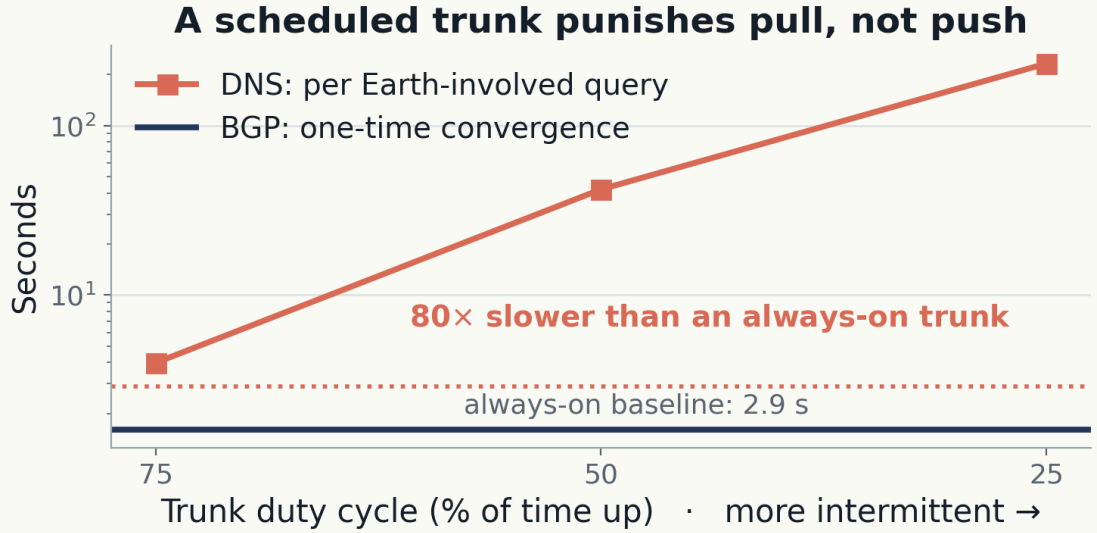
The deliverable: a delay-driven regime map

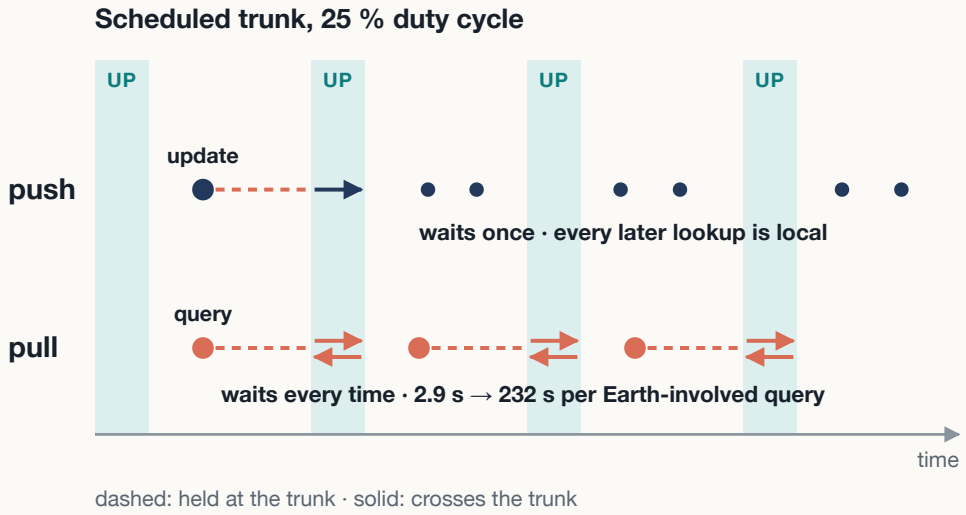


H4, cross-cutting. Pull accuracy falls as R, the ratio of convergence time to the interval between binding changes, grows: 98% at R = 0.33 → 30% at R = 12. Push stays at 100% at every R.

In the contested band the hybrid is concrete: push what is queried across the trunk, pull what stays local.

The caveat we can quantify: links are scheduled





Everything so far assumed links that are up. On a lunar trunk at 25% duty, Earth-involved pull queries go **2.9 s → 232 s**; push overhead is byte-identical and accuracy stays at 100%.

> **Intermittency moves the contested band toward push. The always-on results are pull's best case.**

What to take away

1. **The binding layer needs a mechanism.** It is the last hand-maintained table in an otherwise automated stack.
2. **Link delay picks the mechanism.** It predicts the winner better than network size or how often bindings change.
3. **Rule of thumb:** pull below ~100 ms RTT · push beyond ~3 s · hybrid in between
4. **Next:** hierarchical authorities, contact-plan-aware resolution, and the cost of securing it with BPSec.
5. **Old DTN wisdom, now with numbers.** Request-response is a phone call, publish-subscribe a broadcast, and delay favors the broadcast. Measured here for EID reachability with the Internet's own protocols: DTN can bootstrap on what already connects us.

Paper, models, all 13 experiment configurations and the analysis scripts: `github.com/darksol-cloud/bgp-dns-omnetpp`

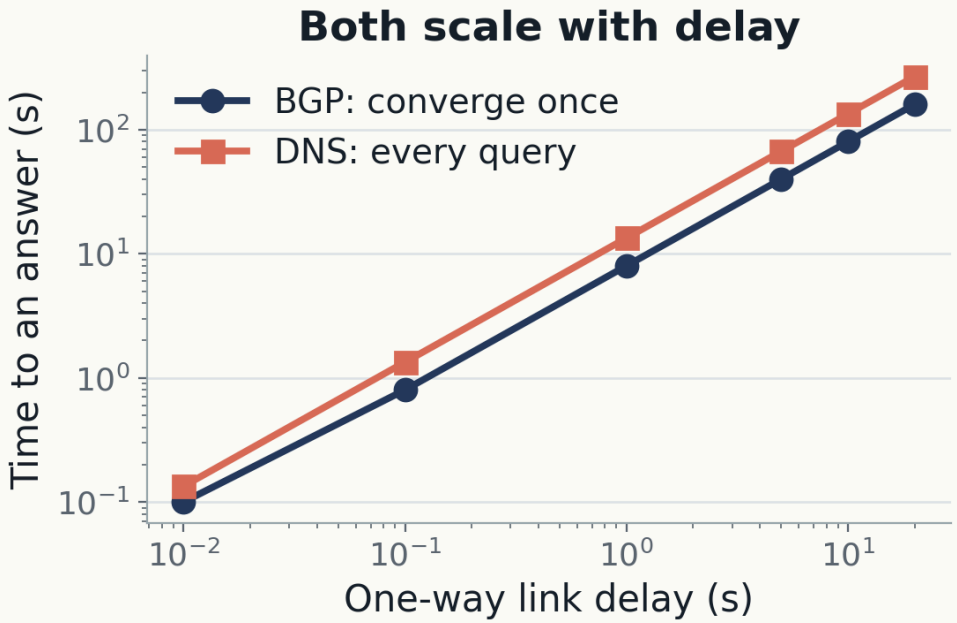
IEEE WISEE 2026 · STINT WORKSHOP

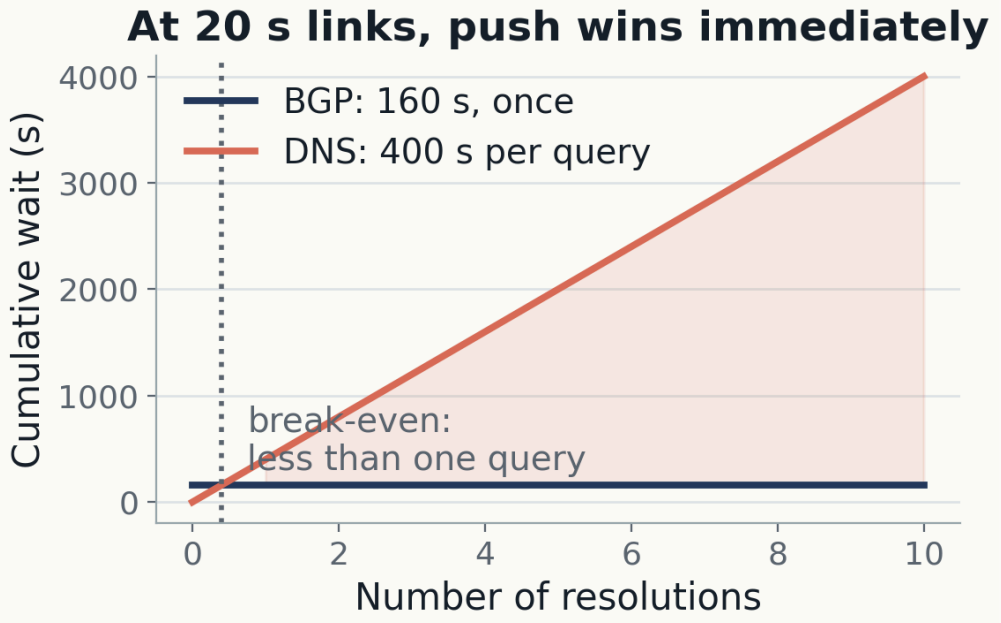
On the Distribution of DTN Reachability Information

Thanks for listining! Questions?

Juan A. Fraire · Inria, INSA Lyon, CITI, UR3720, Villeurbanne, France · D3TN GmbH, Dresden, Germany
Funded by the DARKSOL project · models and configurations are open

Past a few seconds, delay flips the answer





- At 20 s links push converges once in **160 s**; pull pays **400 s per query**, forever. Break-even is under one query.
- Caching helps (45% hits, 41% lower mean latency) and still leaves 235 s per resolution.

Pay once and use forever beats pay every time.

BACKUP · DESTINATION OR NEXT HOP?

Two lookups, stacked

Lookup	Question	Who answers it today
Routing	destination EID → next-hop node	contact plan + CGR, exits, static routes
Binding	next-hop node → CLA address	egress plan · outduct · contact (ION · HDTN · µD3TN)

Push carries both

The flooded entry holds the destination's endpoint **and a path vector**, as BGP carries NLRI, AS_PATH and NEXT_HOP. One table answers both lookups in a multi-hop mesh.

Pull returns the destination only

An endpoint, as DNS returns an address; an underlay delivers there. Inside an IP island the destination **is** the next hop. Across islands a gateway must resolve: hierarchical authorities, next step.

IPND finds neighbors; push and pull find destinations

	IPND	Push / pull (this paper)
Question	who is next to me?	where is `ipn:13.1`?
Scope	one BP hop	network-wide
Needs	a live IP link and periodic beacons	store-and-forward; works over scheduled links (Exp. 14)
Protocol	new and DTN-specific; IRTF draft, expired 2016	BGP and DNS machinery and tooling that already exist
Administration	autonomous, no policy handle, no security in the draft	pull: an authority with registration, TTL, delegation · push: origin decides what it announces, policy at peerings

`draft-irtf-dtnrg-ipnd-03`: UDP beacons (broadcast, multicast or unicast) carrying the sender's EID and its CLAs.
Implemented in ION, which auto-creates a plan, a duct and a discovered contact for TCP/UDP neighbors.

> **IPND can fill the neighbor table that push floods from. On a scheduled trunk it has nothing to discover.**