

Building Strongly-Consistent Systems Resilient to Failures, Partitions and Slowdowns

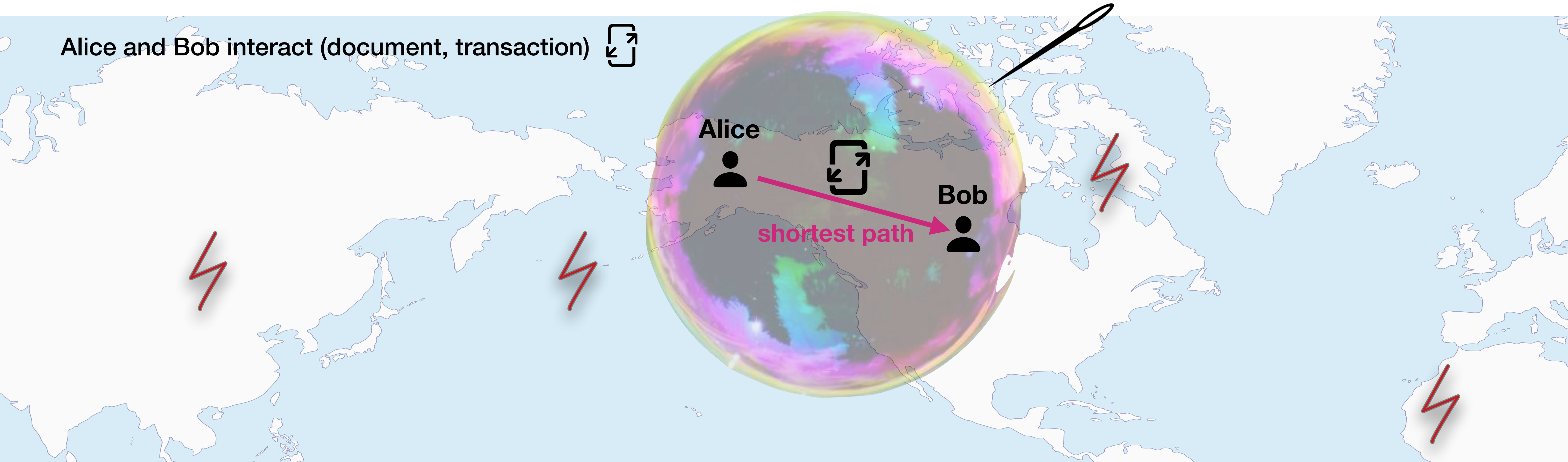
PhD Thesis Defense

26.05.2023

Cristina Bănescu

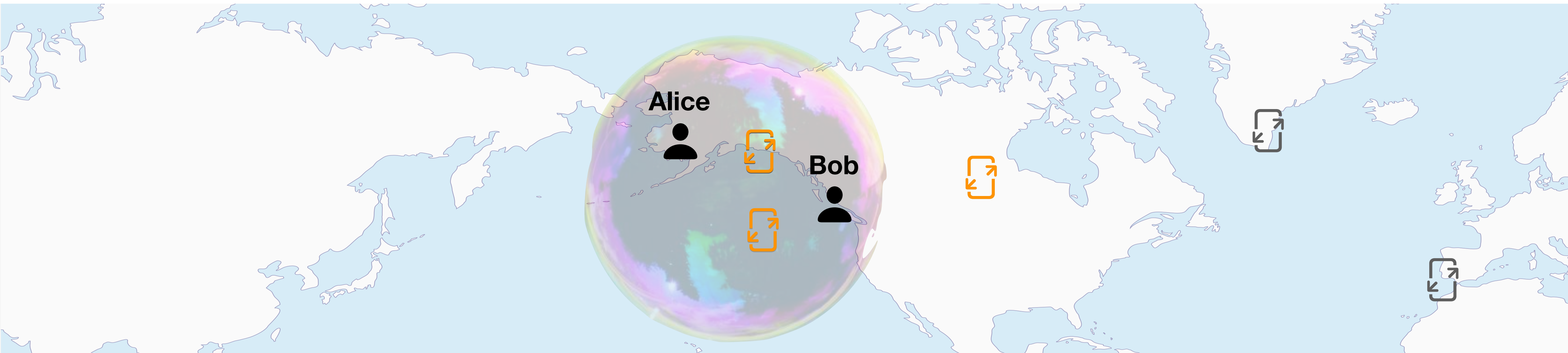
EPFL

An ideal situation



Goal: **resilience**

A more realistic view



Replication masks **individual replica failures**, but adds **dependency on quorum**

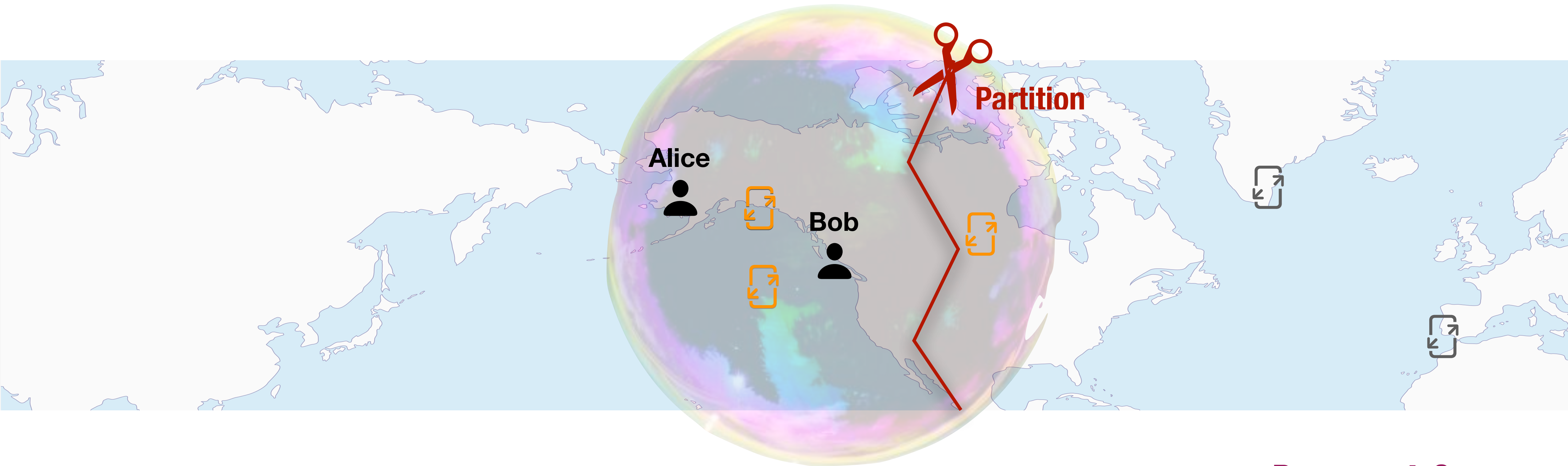

```
graph TD; A[Resilient strongly-consistent coordination] --> B[Resilience to distant failures]; A --> C[Resilience to network asynchrony];
```

Resilient strongly-consistent coordination

Resilience to distant failures

Resilience to network asynchrony

Challenge: Local resilience in a global world

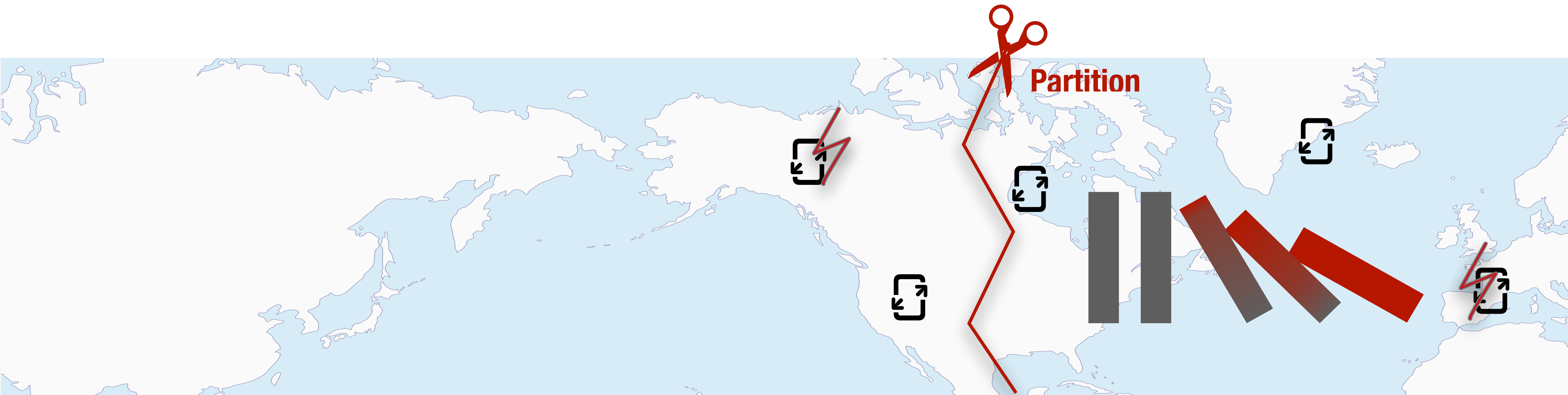


Rare events?
Naïve simplification

CAP theorem: pick two of strong consistency, availability, partition tolerance

* Seth Gilbert and Nancy Lynch. Brewer's conjecture and the feasibility of consistent, available, partition-tolerant web services. *ACM SIGACT News*, 33(2), 2002.

Gray failures, dependencies...



Resilience ⇔ withstanding **all failure types and combinations**

Evidence of grey failures in clouds

TECH

Amazon Finds the Cause of Its AWS Outage: A Typo

Company says disruption started at its Northern Virginia data centers

An outage Tuesday at Amazon Web Services disabled and slowed apps and websites from a wide section of U.S. companies.

PHOTO: ANDREW HARRER/BLOOMBERG NEWS

By [Laura Stevens](#)
March 2, 2017 4:35 pm ET

Amazon.com Inc. on Thursday blamed human error for an outage at its cloud-services unit that caused widespread disruption to internet traffic across the U.S. earlier this week.

Fastly says single customer triggered bug behind mass internet outage

Flaw was introduced in May and lay dormant until a customer updated their settings, firm says

An Analysis of Network-Partitioning Failures in Cloud Systems

Ahmed Alquraan, Hatem Takturi, Mohammed Alfatafta, Samer Al-Kiswany
University of Waterloo, Canada

OSDI '18

Gray Failure: The Achilles' Heel of Cloud-Scale Systems

Peng Huang

Microsoft Research
Johns Hopkins University

Chuanxiong Guo

Microsoft Research

Lidong Zhou

Microsoft Research

Jacob R. Lorch

Microsoft Research

Yingnong Dang

Microsoft Azure

Murali Chintalapati

Microsoft Azure

Randolph Yao


Microsoft Azure

HotOS '17

A Byzantine failure in the real world

27/11/2020

 Tom Lianza

 Chris Snook

An analysis of the Cloudflare API availability incident on 2020-11-02

When we review design documents at Cloudflare, we are always on the lookout for Single Points of Failure (SPOFs). Eliminating these is a necessary step in architecting a system you can be confident in. Ironically, when you're designing a system with built-in redundancy, you spend most of your time thinking about how well it functions when that redundancy is lost.

On November 2, 2020, Cloudflare had an incident that impacted the availability of the API and dashboard for six hours and 33 minutes. During this incident, the success rate

7

Real-world resilience risks in blockchains

- **Censorship / validators offline** \Rightarrow unavailability

SLASHING

Slashing is a more severe action that results in the forceful removal of a validator from the network and an associated loss of their staked ether. There are three ways a validator can be

Penalties

So far we have considered perfectly well-behaved validators, but what about validators that do not make timely head, source and target votes or do so slowly?

The penalties for missing the target and source votes are equal to the rewards the attester would have received had they submitted them. This means that instead of having the reward

- **Network routing attacks** \Rightarrow unavailability or forks

Application-layer and network-layer defenses are critical for fortifying routing attacks.

BY YIXIN SUN, MARIA APOSTOLAKI, HENRY BIRGE-LEE,
LAURENT VANBEVER, JENNIFER REXFORD,
MUNG CHIANG, AND PRATEEK MITTAL

Securing Internet Applications from Routing Attacks

Routing attacks on consensus. Network-level adversaries can perform routing attacks on bitcoin to partition the set of nodes into two (or more) disjoint components.² Consequently, the attacks disrupt the ability of the entire network to reach consensus. The adversary must divert and cut *all* the connections connecting the various components together. To do so, the adversary can perform an interception attack by hijacking the IP prefixes of each component and selectively dropping the

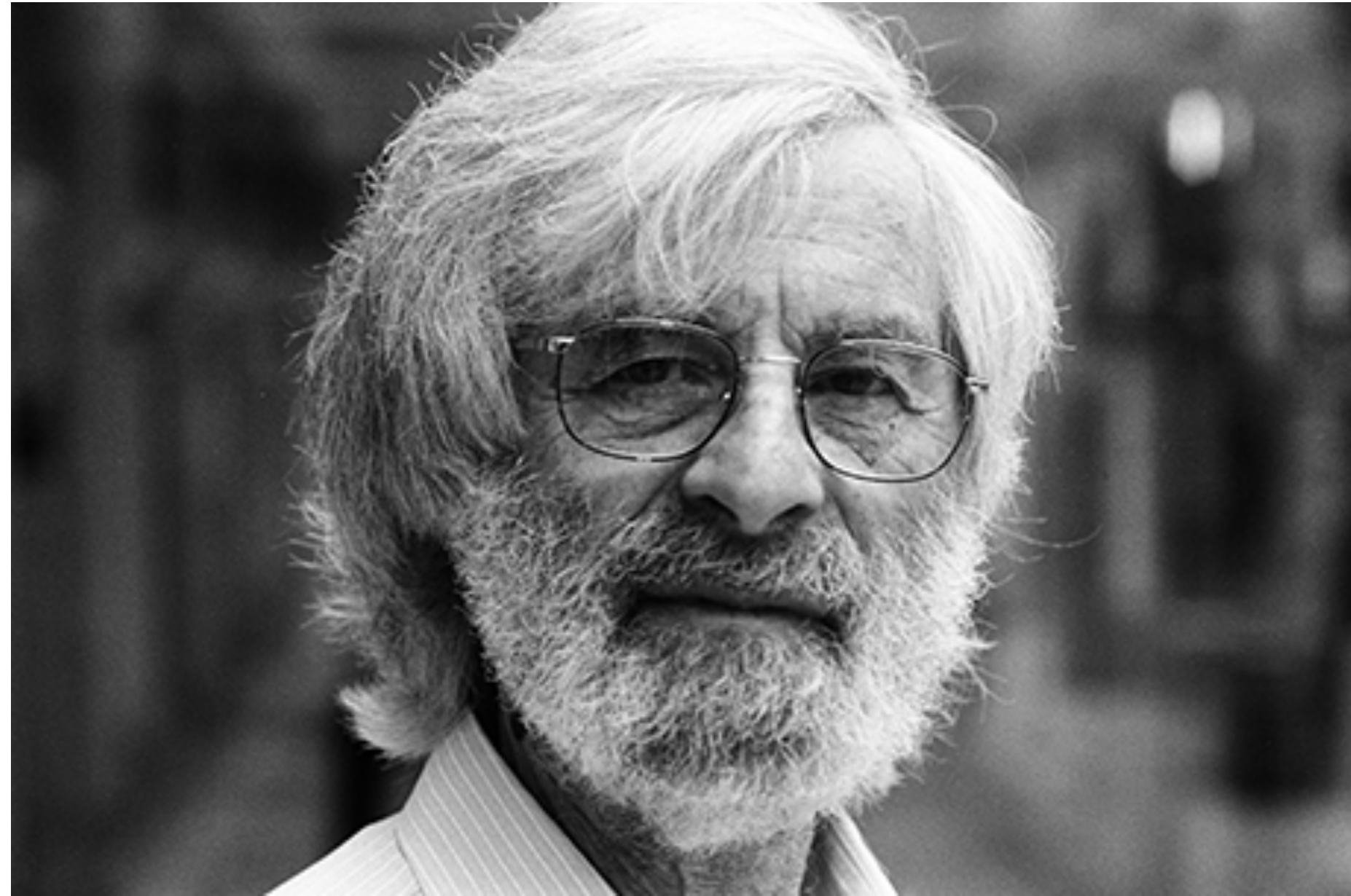
Hijacking Bitcoin: Routing Attacks on Cryptocurrencies

<https://btc-hijack.ethz.ch>

Maria Apostolaki
ETH Zürich
apmaria@ethz.ch

Aviv Zohar
The Hebrew University
avivz@cs.huji.ac.il

Laurent Vanbever
ETH Zürich
lvanbever@ethz.ch

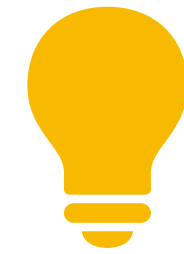


“A distributed system is one in which the failure of a computer you didn't even know existed can render your own computer unusable.”

(Leslie Lamport)

Can we protect Alice's access from distant failures, partitions and slowdowns?

Insight 1



**Alice cares that the service is
available for her access, not globally**

**Limit exposure:
local accesses have no global dependencies**

Resilient strongly-consistent coordination

```
graph TD; A[Resilient strongly-consistent coordination] --> B[Resilience to distant failures]; A --> C[Resilience to network asynchrony];
```

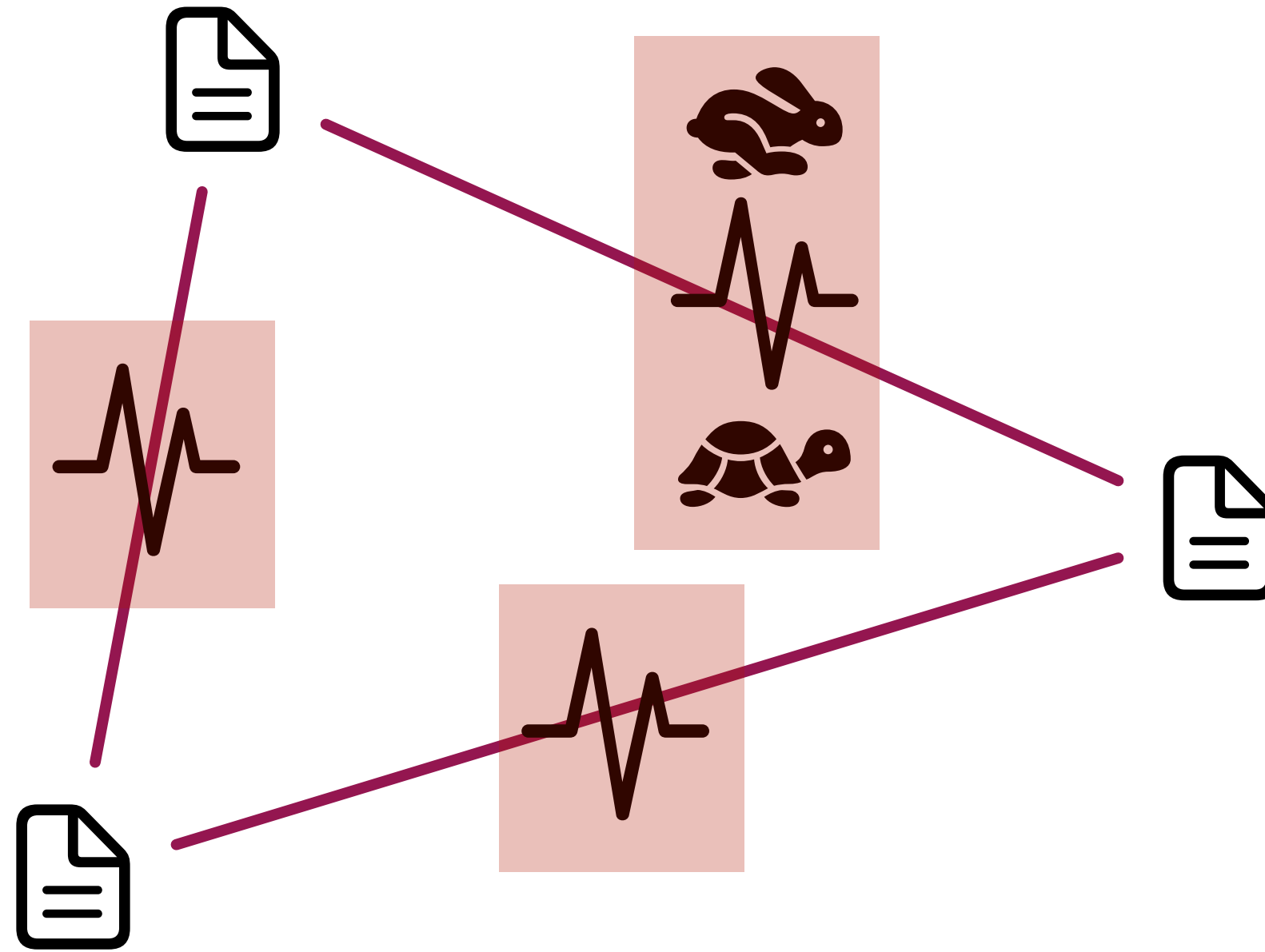
Resilience to distant failures

Resilience to network asynchrony



Limit exposure

Challenge 2: Resilience despite **asynchrony**



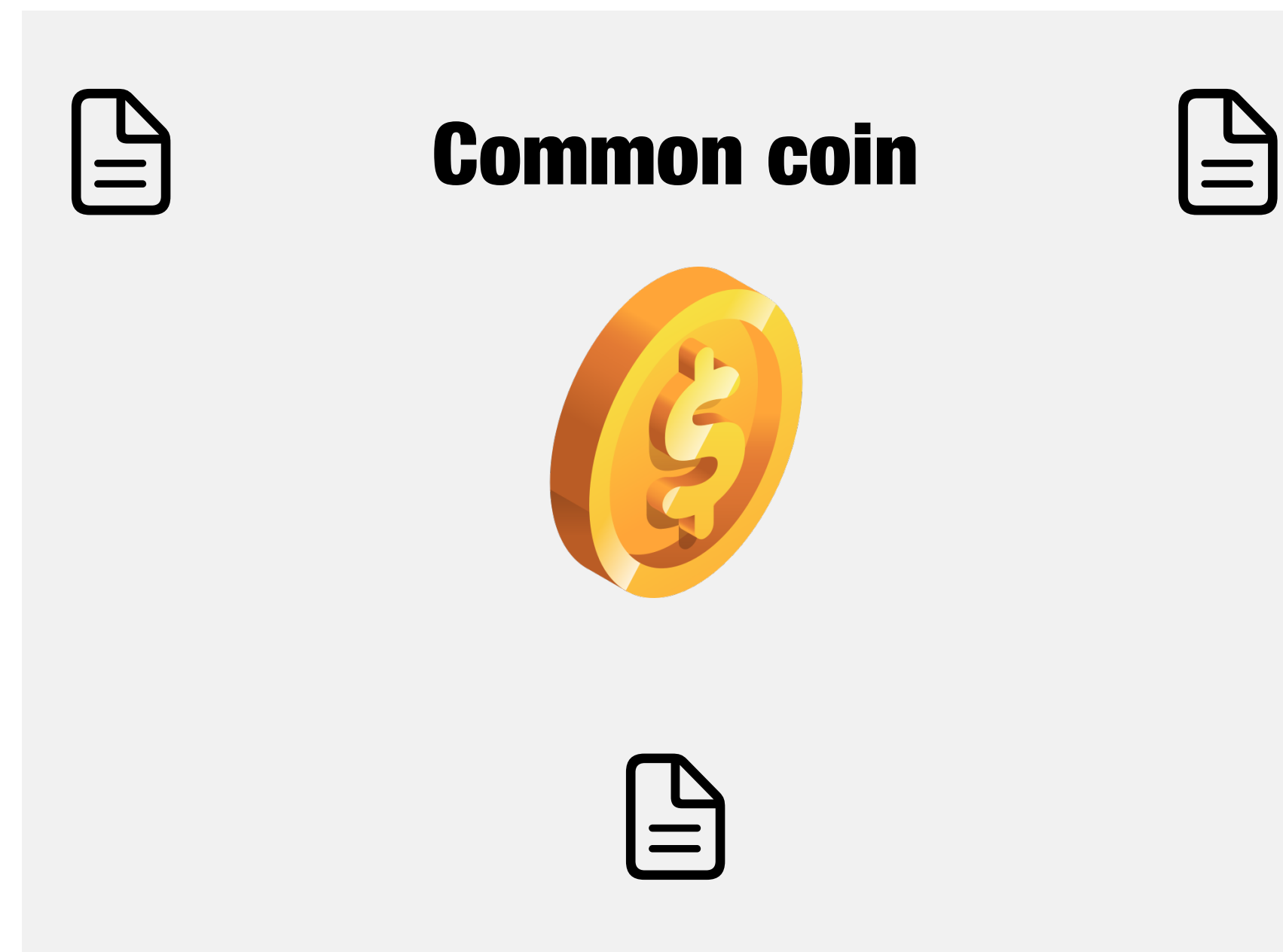
Asynchronous consensus is robust to asynchrony, but rarely deployed

Towards more practical asynchronous consensus

Asynchronous consensus requires randomness for progress (FLP*)

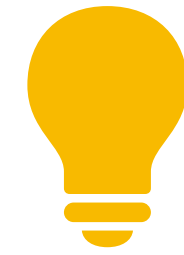
* Michael J Fischer, Nancy A Lynch, and Michael S Paterson. Impossibility of distributed consensus with one faulty process. *Journal of the ACM (JACM)*, 32(2):374–382, 1985.

Many algorithms typically use a common coin



Are common coins necessary for asynchronous consensus?

Insight 2



No

**Common coins not needed for asynchronous consensus
(crash-fault setting)**

**Though no asymptotic improvement
Open question: Byzantine setting**

First step towards more practical consensus

Resilient strongly-consistent coordination

```
graph TD; A[Resilient strongly-consistent coordination] --> B[Resilience to distant failures]; A --> C[Resilience to network asynchrony];
```

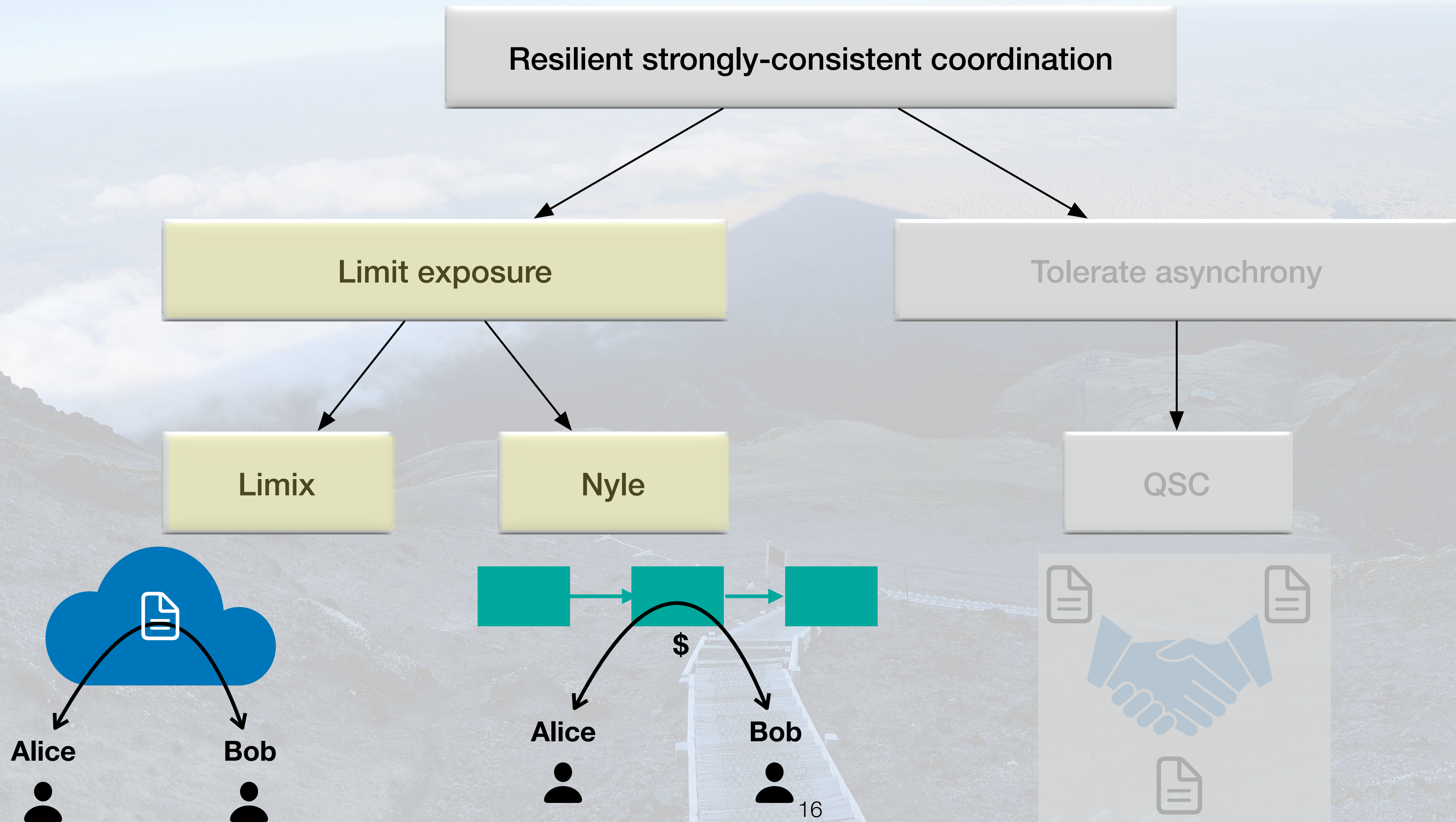
Resilience to distant failures

Resilience to network asynchrony

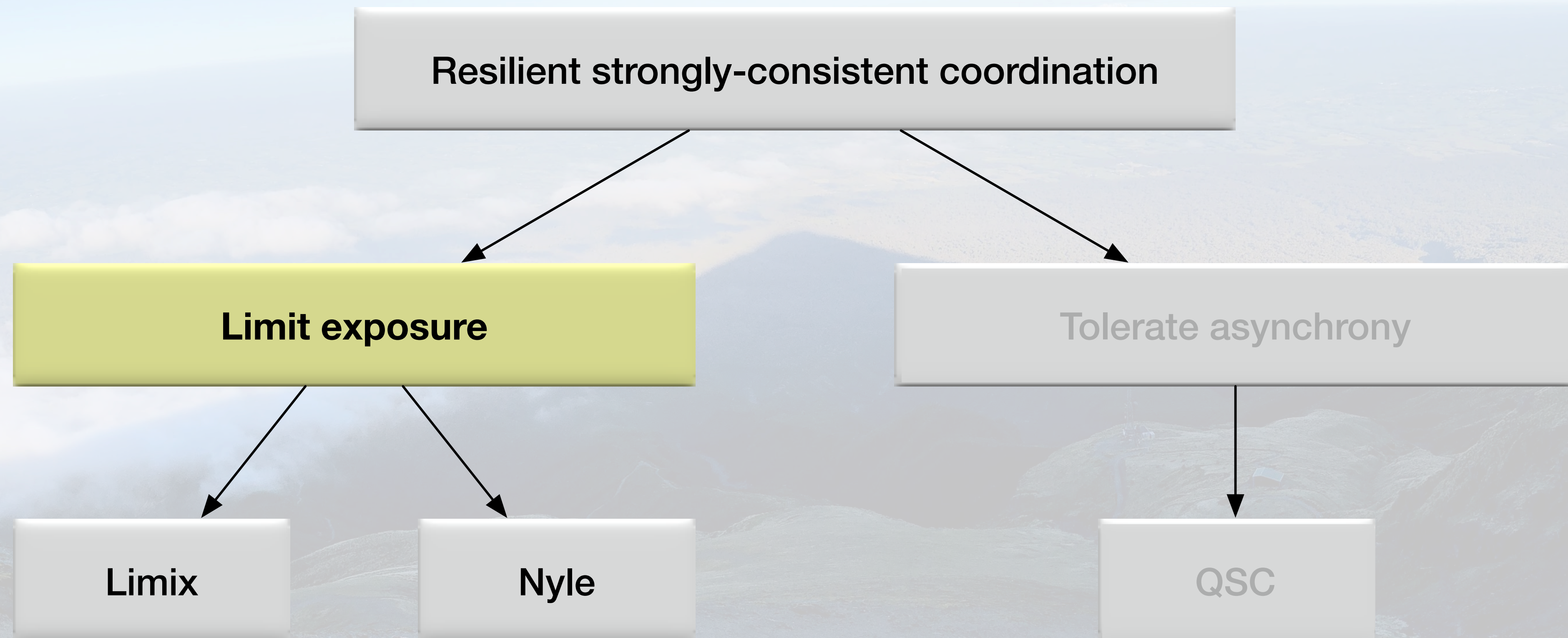


Limit exposure

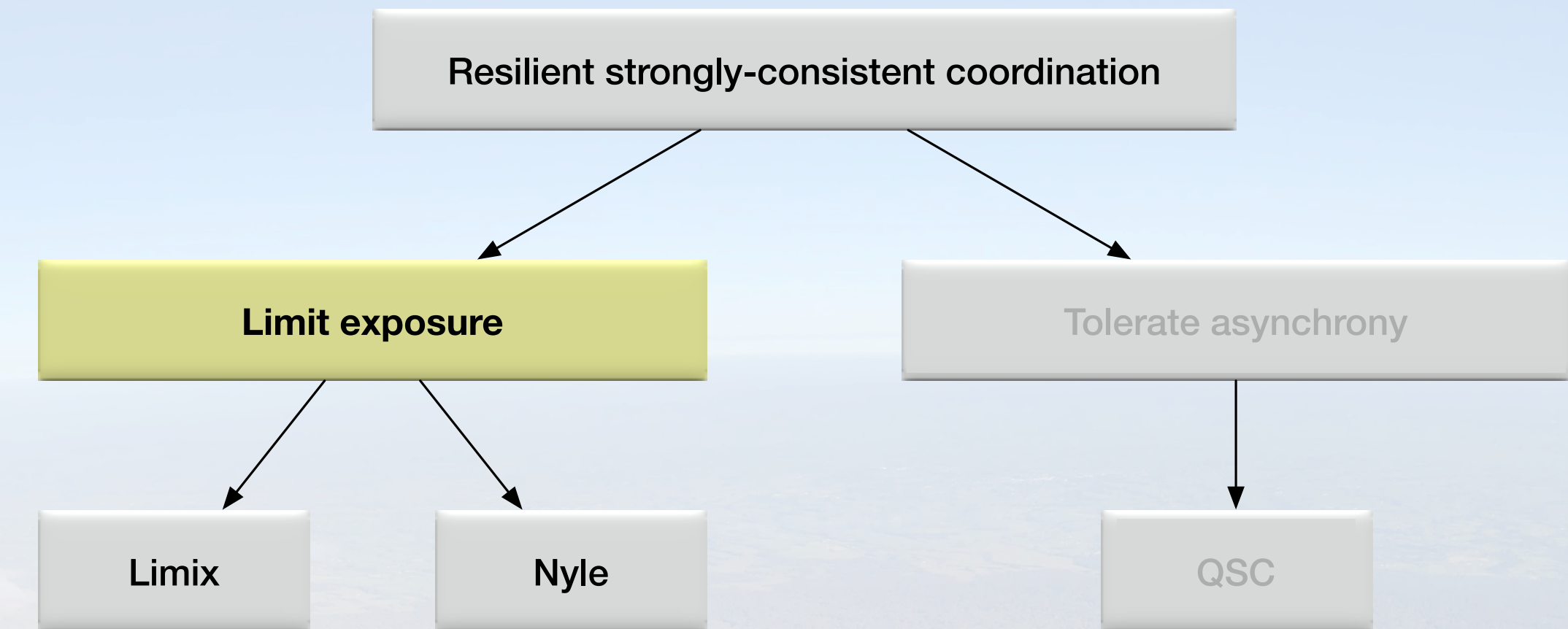
Talk roadmap



Thesis roadmap

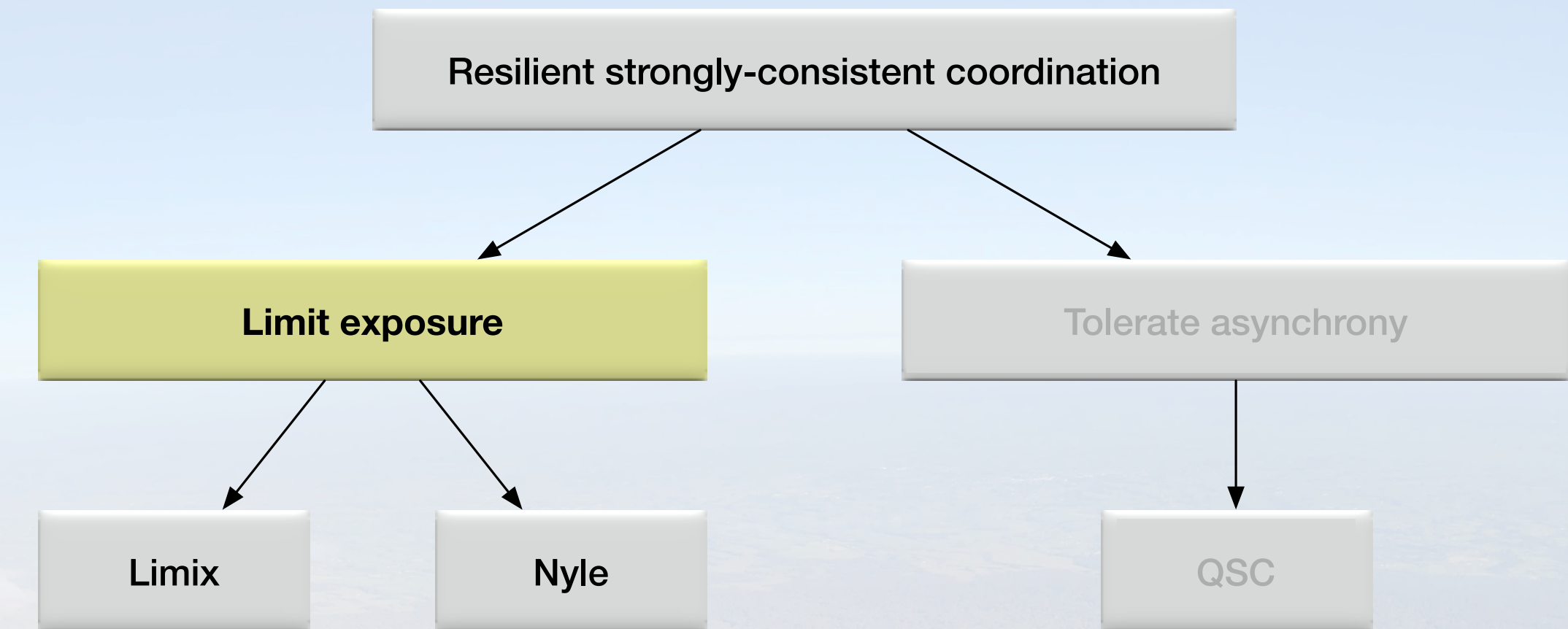


Limiting exposure roadmap



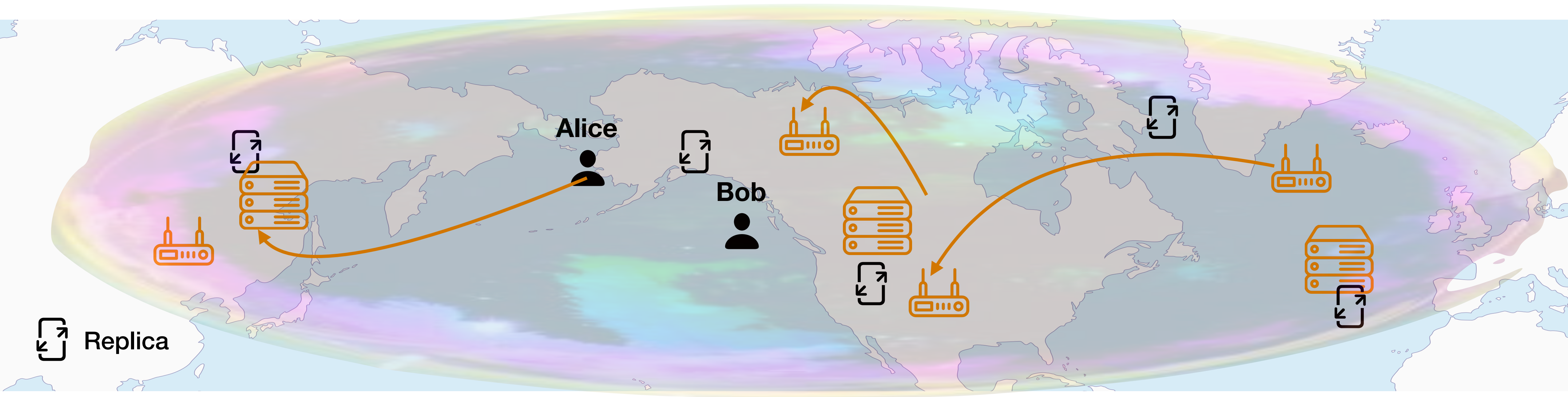
- Defining Lamport exposure
- Providing meaningful exposure guarantees
- Summary

Limiting exposure roadmap



- **Defining Lamport exposure**
- Providing meaningful exposure guarantees
- Summary

Defining Lamport exposure



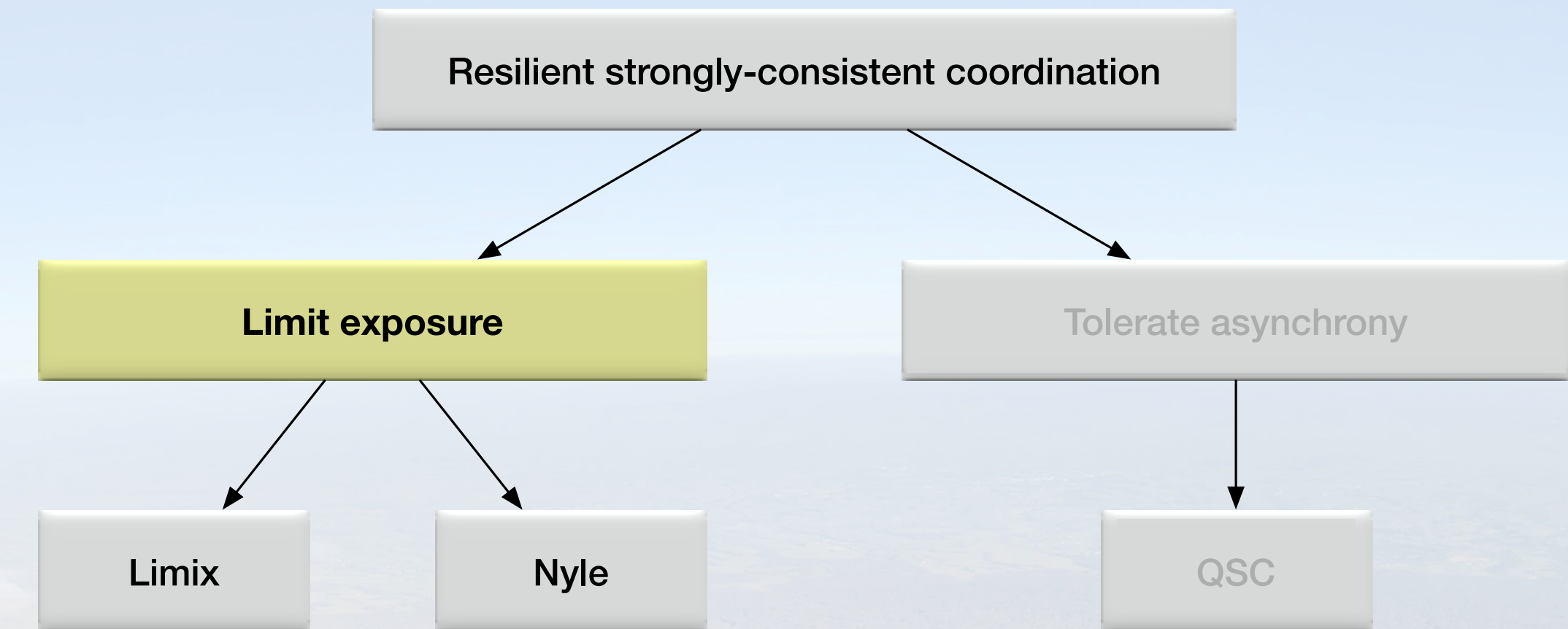
Lamport exposure for activity A of Alice

all distributed system components (servers, routers, network links, etc.)
whose failure **could** contribute to slowing or stopping activity A of a user.

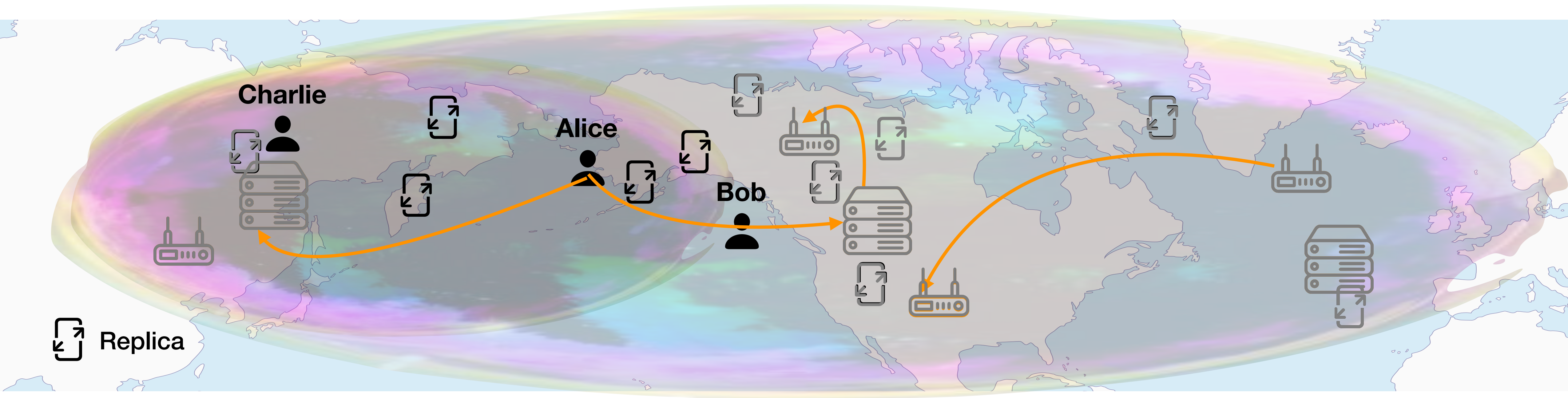
Global system: Lamport exposure is some global set the user is unaware of

Limiting exposure roadmap

- Defining Lamport exposure
- **Providing meaningful exposure guarantees**
 - **Goals**
 - Strawmen
 - Jurisdiction-based zoning
 - Metric-based zoning
- Summary



Limit exposure & preserve global manageability



Goals:

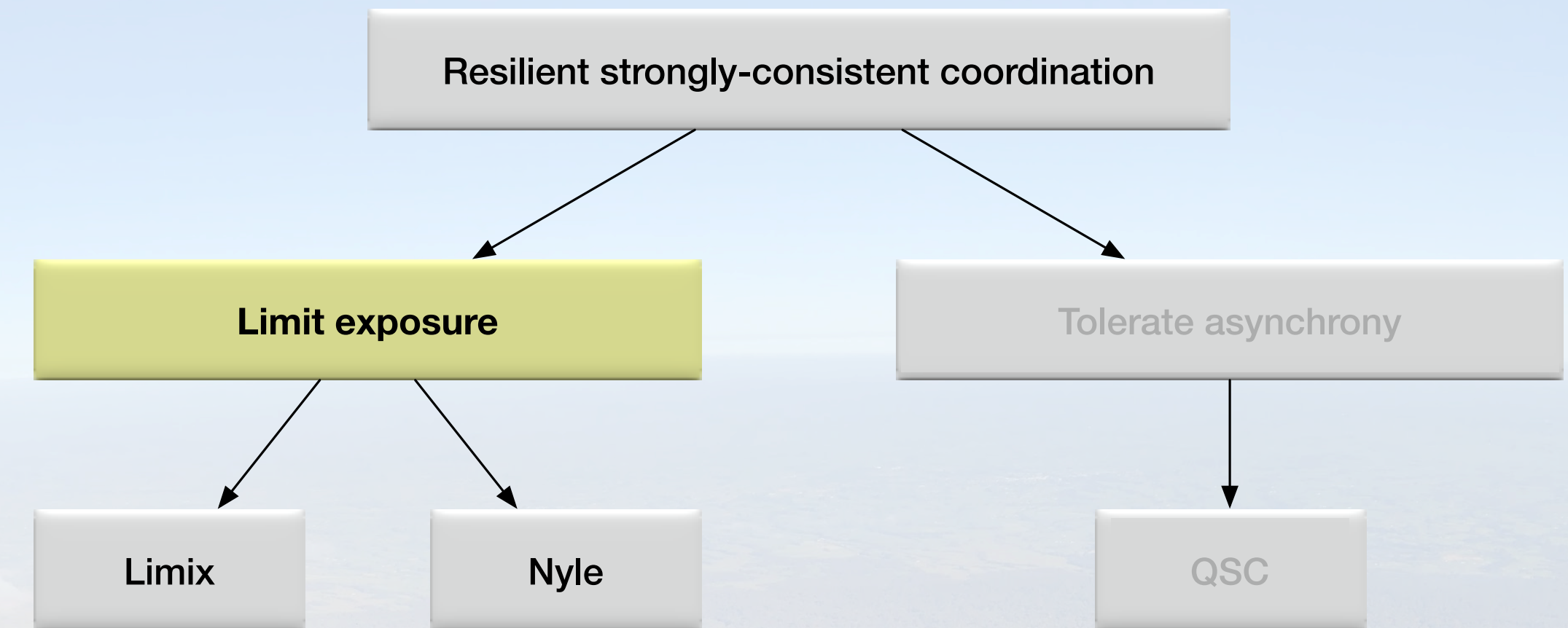
Alice has a lower exposure as Bob gets closer

Limiting exposure does not reduce durability

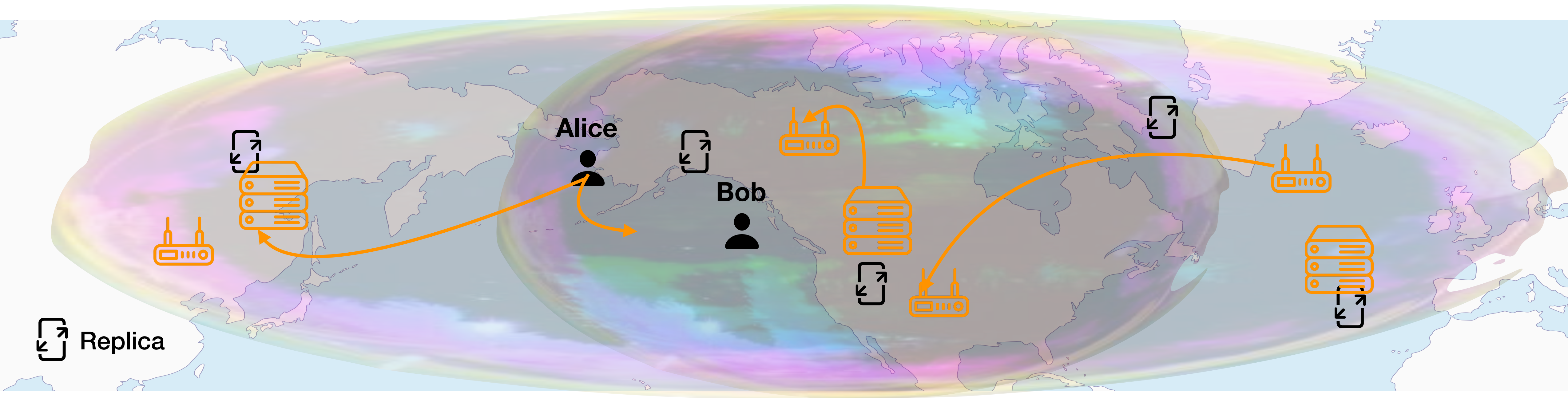
Exposure adjusts to new interactions and users changing location

Limiting exposure roadmap

- Defining Lamport exposure
- **Providing meaningful exposure guarantees**
 - Goals
 - **Strawmen**
 - Jurisdiction-based zoning
 - Metric-based zoning
- Summary



Strawman 1: localised infrastructure

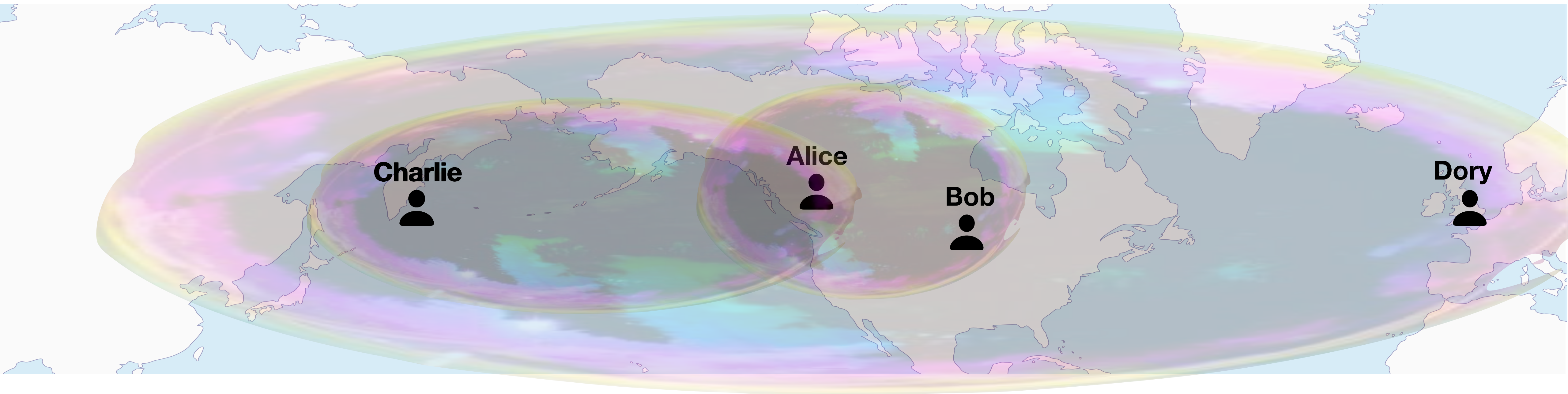


A localized deployment, e.g. local document service, instead of globalised one

✓ **Resilience:** Lower exposure for local users

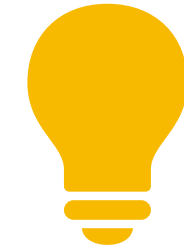
✗ **No global manageability:** migrating users have remote dependencies

Strawman 2: many zones



- ✓ **Resilience:** shield users from failures outside the zone
- ✓ **Global manageability:** dependencies follow usage pattern
- ✗ **Overhead:** replication inside each of the many zones

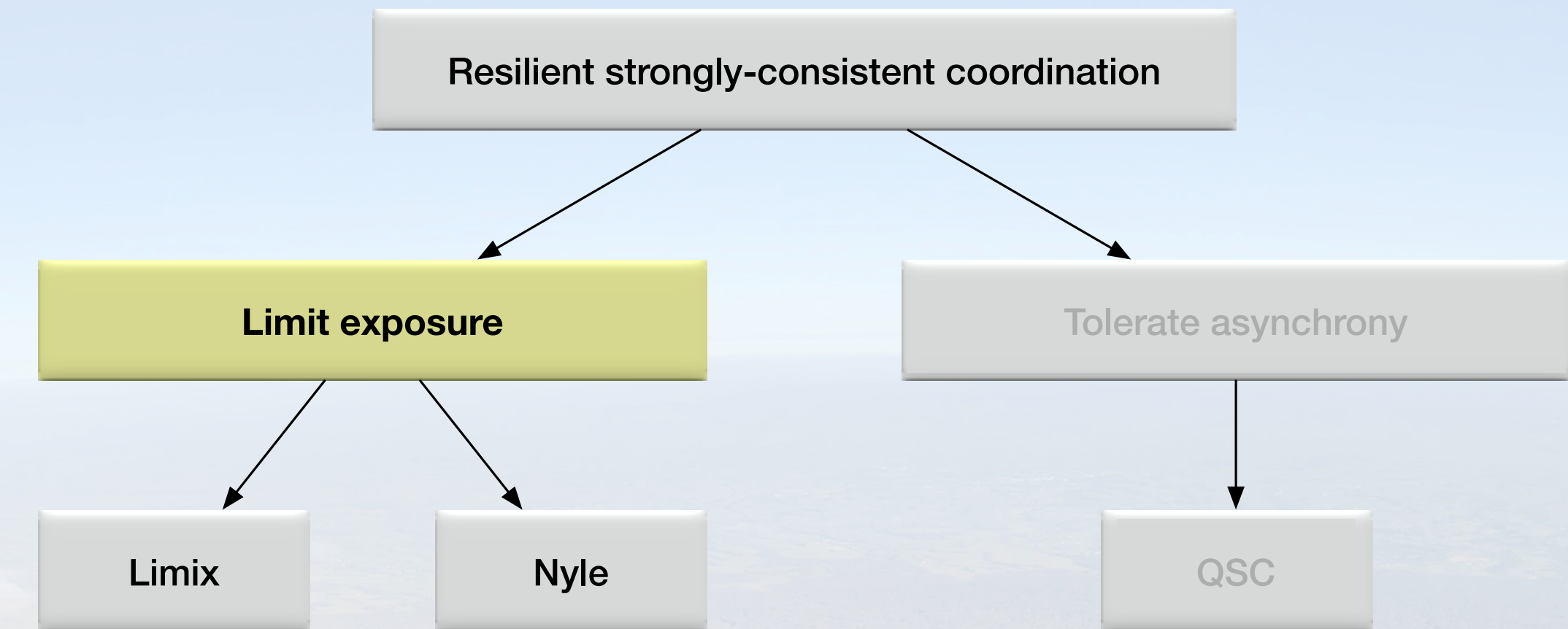
Insight



User interactions that
the application (document sharing, ledger) **aims to support**
determine the zoning policy

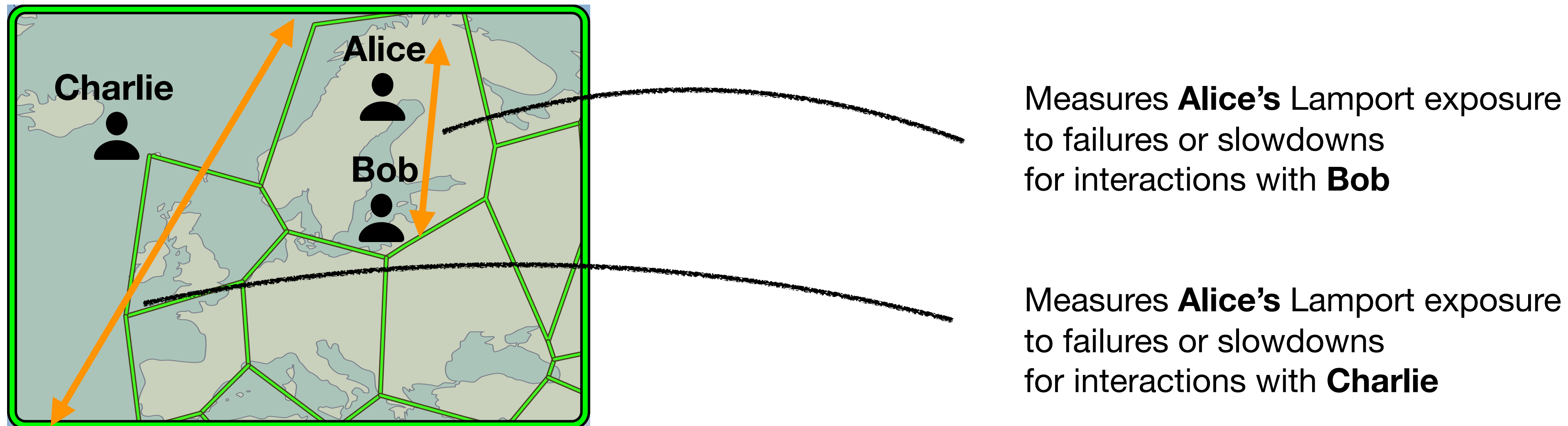
Limiting exposure roadmap

- Defining Lamport exposure
- **Providing meaningful exposure guarantees**
 - Goals
 - Strawmen
 - **Jurisdiction-based zoning**
 - Metric-based zoning
- Summary



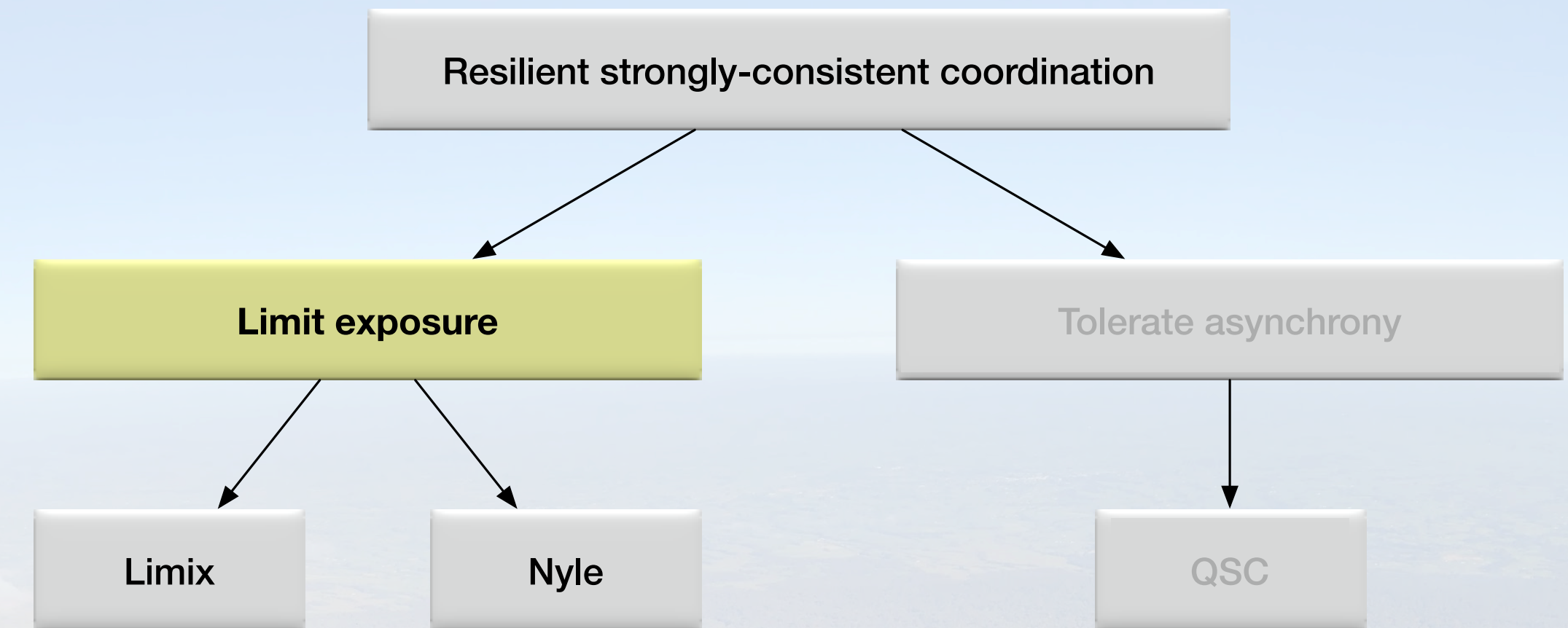
Jurisdiction-based zoning

- Administrative or legal boundaries (e.g., inside/outside the EU)
Use case: data sovereignty, enforcing legal frameworks for transactions
- Can be used in a **pay-as-you-go** fashion



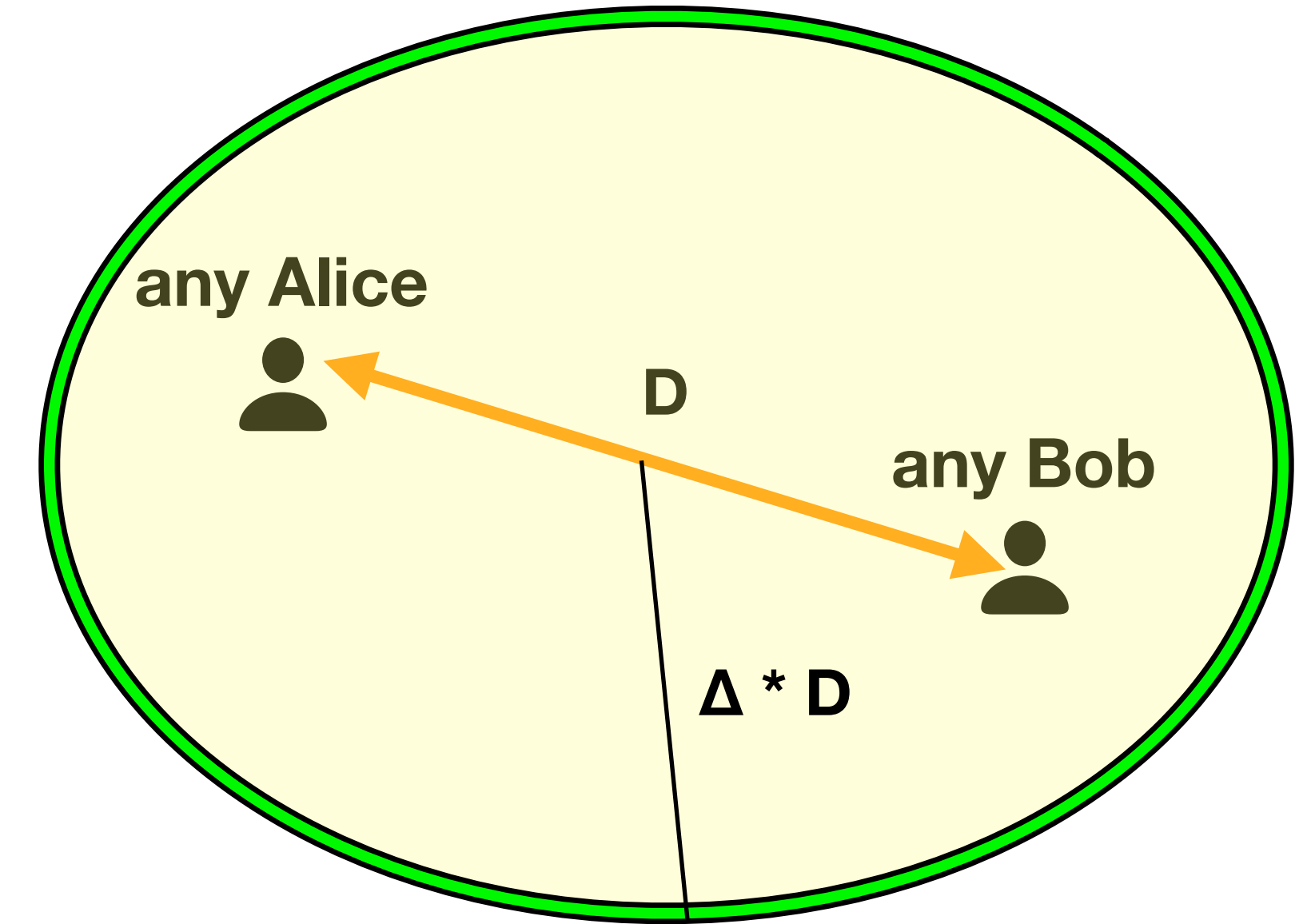
Limiting exposure roadmap

- Defining Lamport exposure
- **Providing meaningful exposure guarantees**
 - Goals
 - Strawmen
 - Jurisdiction-based zoning
 - **Metric-based zoning**
- Summary



Metric-based zoning

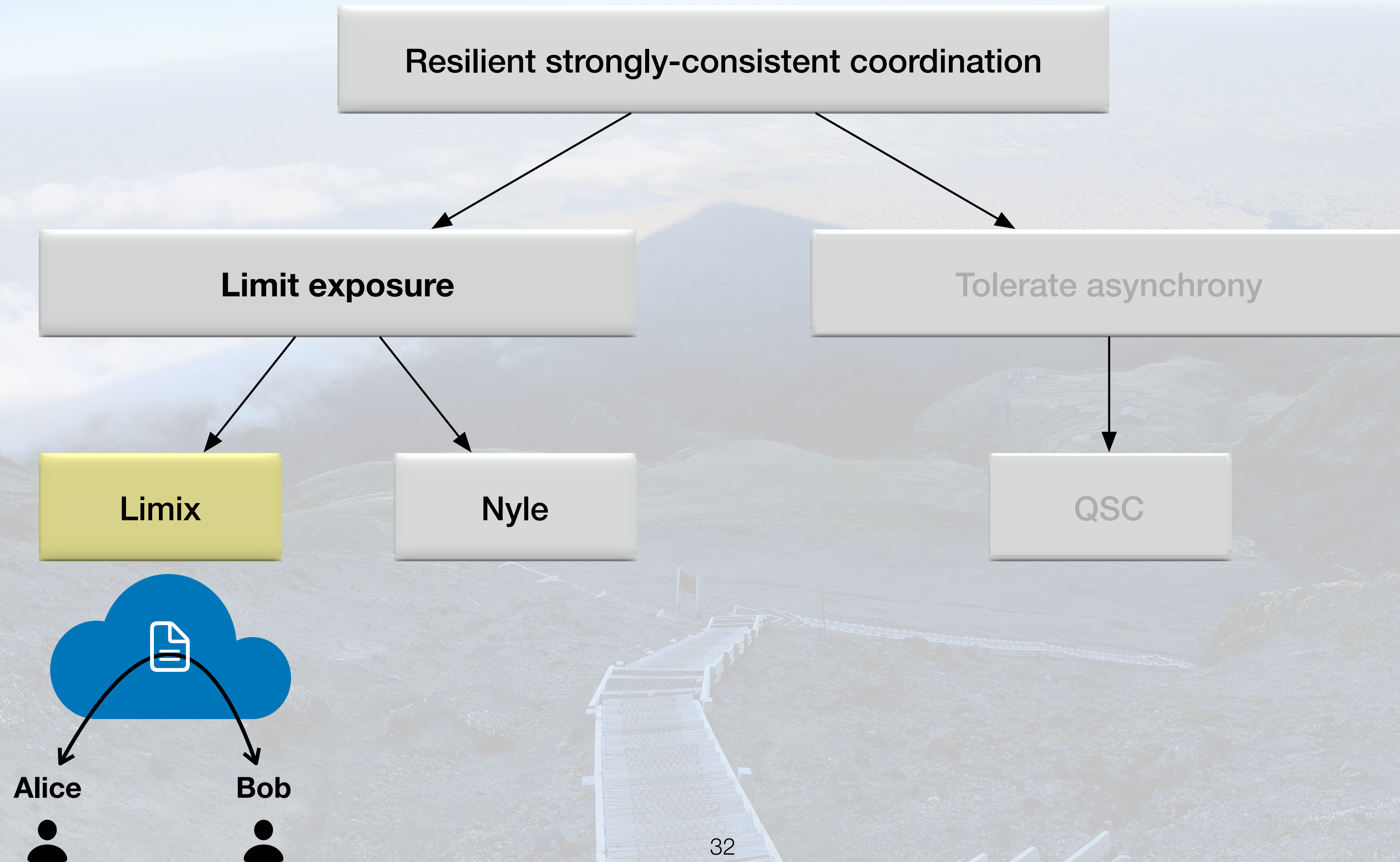
- **Autozoning for systematically limiting exposure**
- **Dynamic local and global interactions**
 - Limit exposure to a small Δ around **any two users**
 - Metrics for D : geographic distance, latency, etc
- **Use compact-graph approximation theory**
 - Strong guarantees
 - **Overhead:** # zones logarithmic in # deployment nodes



Limiting exposure summary

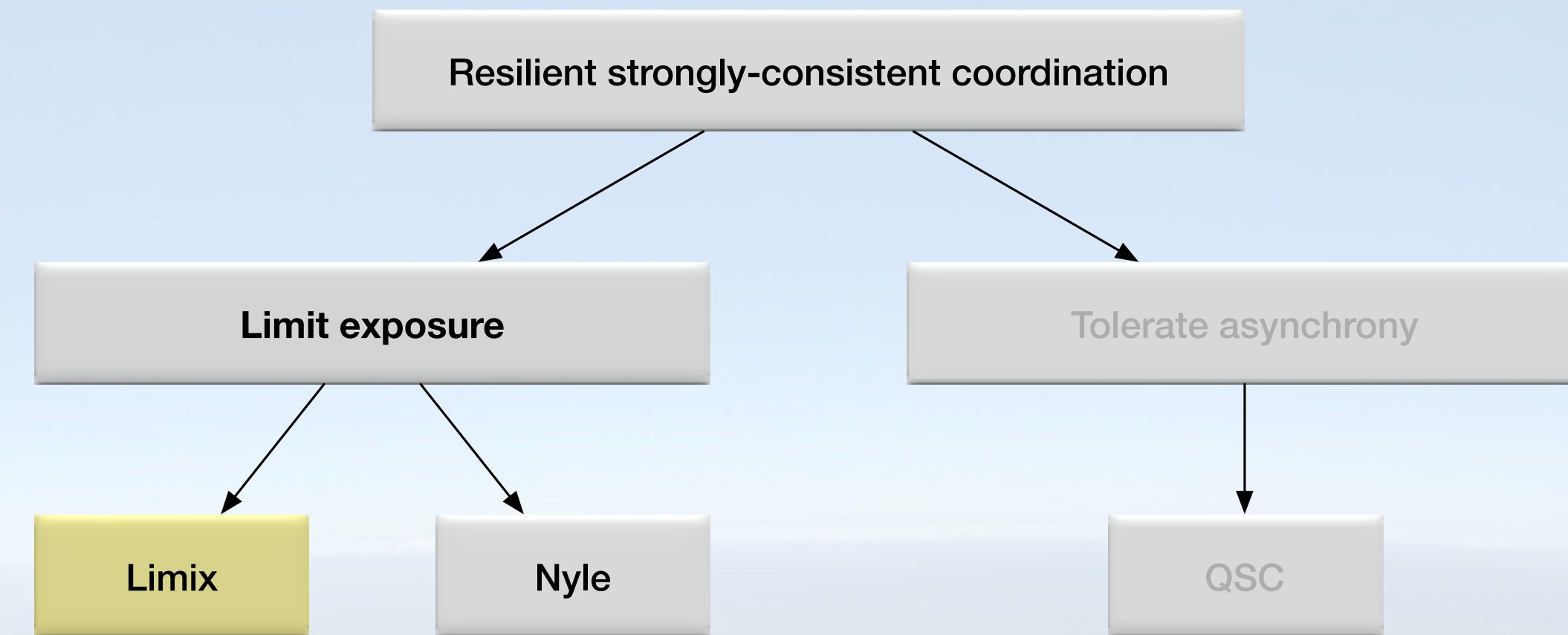
- Defined **Lamport exposure** to measure resilience to remote failures
- **Practical metrics** for Lamport exposure
- **Zoning** for building **globalised systems** that enforce **localized resilience**

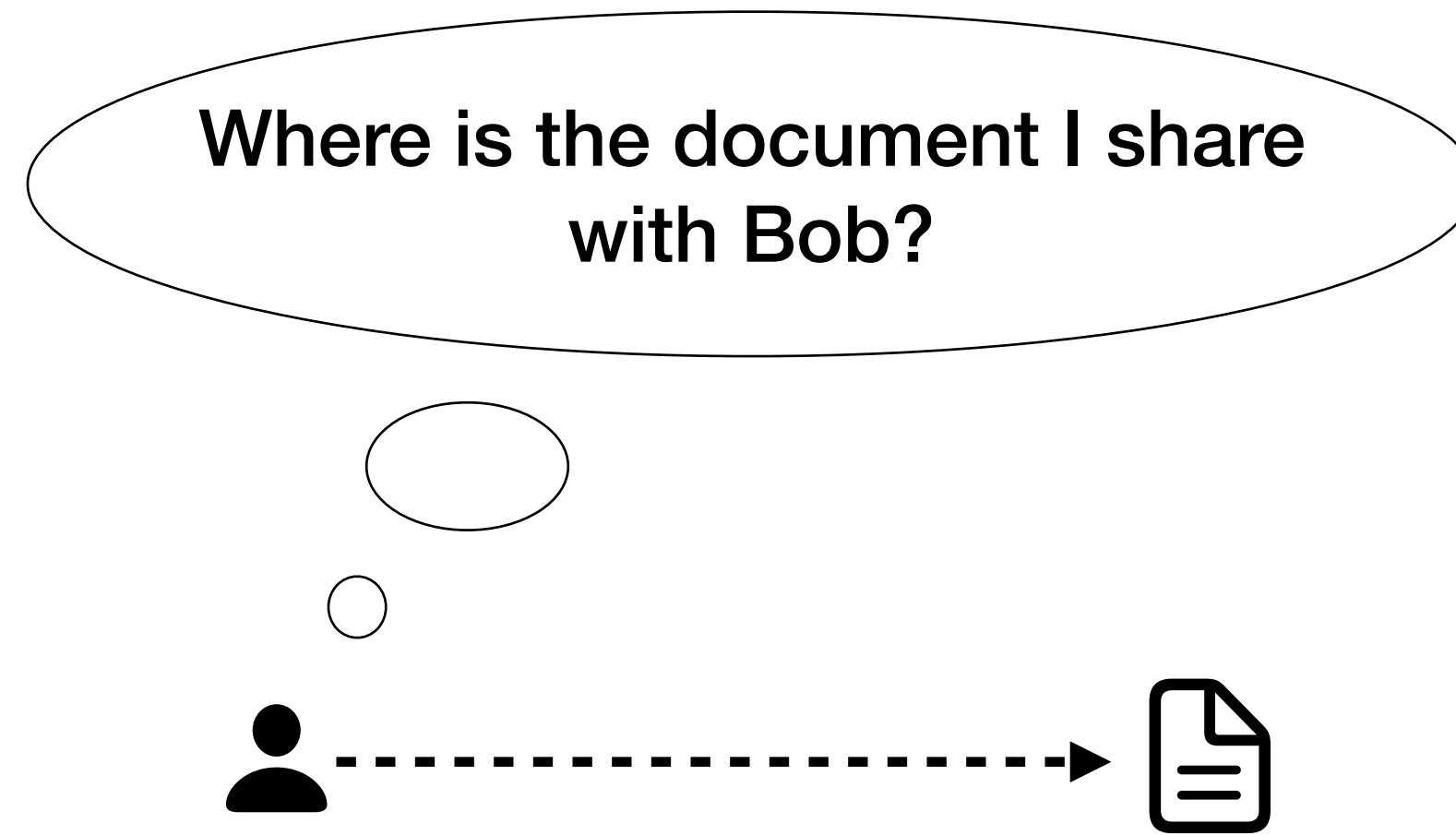
Thesis roadmap



Limix roadmap

- **The problem: exposure of metadata**
- Design of exposure-limiting metadata service
- Challenges: strongly-consistent item lookup
- Architecture
- Evaluation



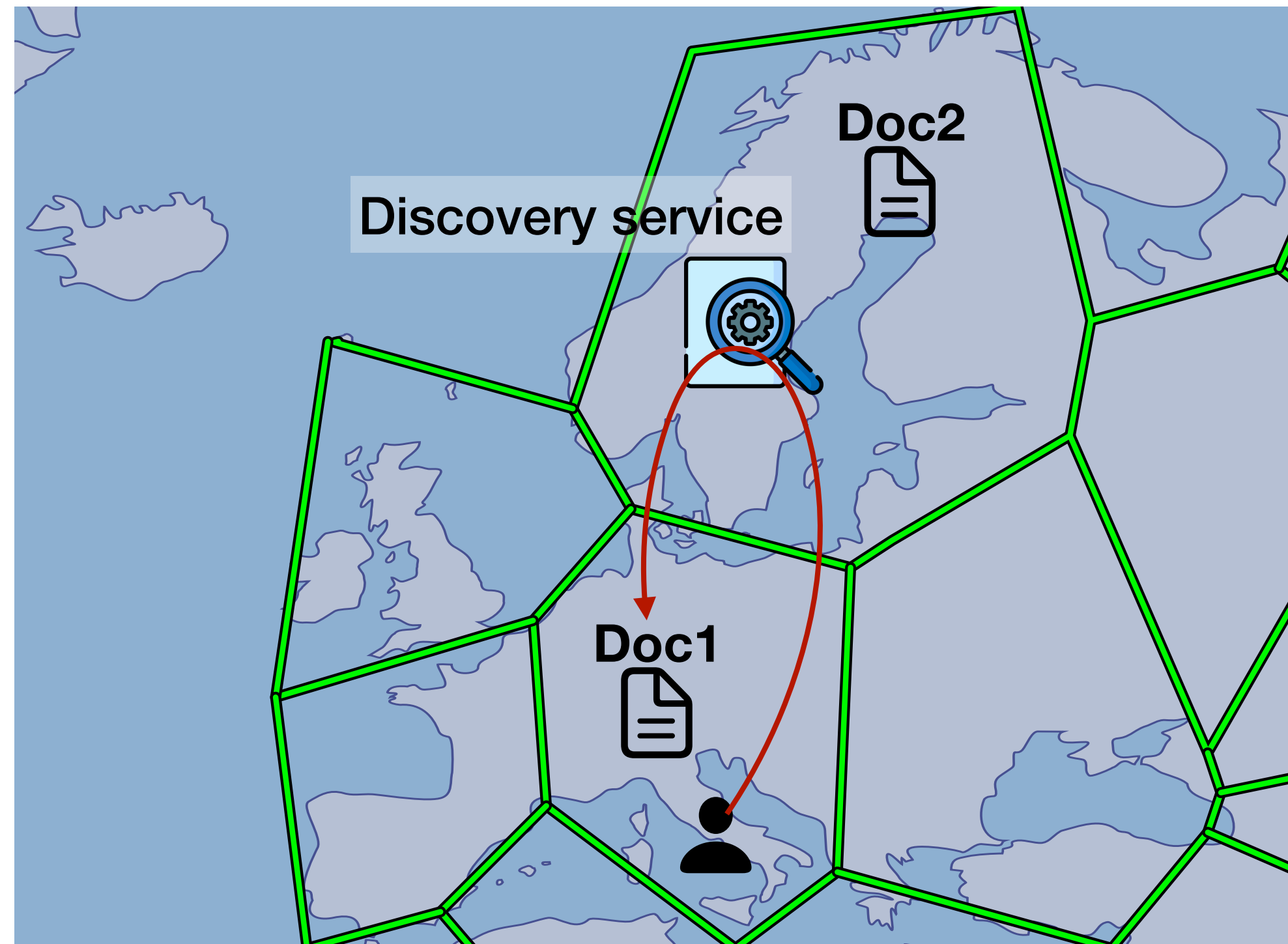


Limix focuses on the metadata problem:

**How to find a localized strongly-consistent item
when the location metadata is usually globalized?**

Geo-replication is not enough

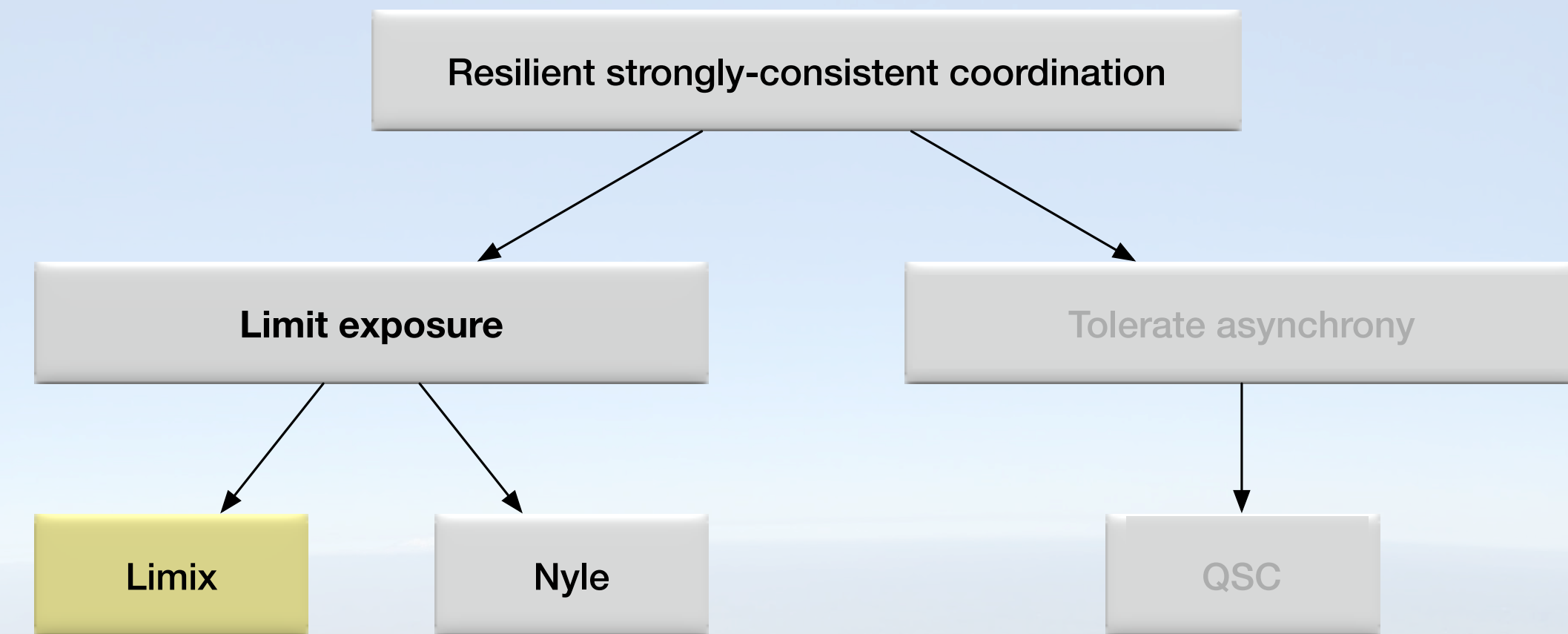
Two documents in a geo-replicated system
(replicas not depicted)



X Challenge 1:
Maintain Lamport exposure while performing file lookup

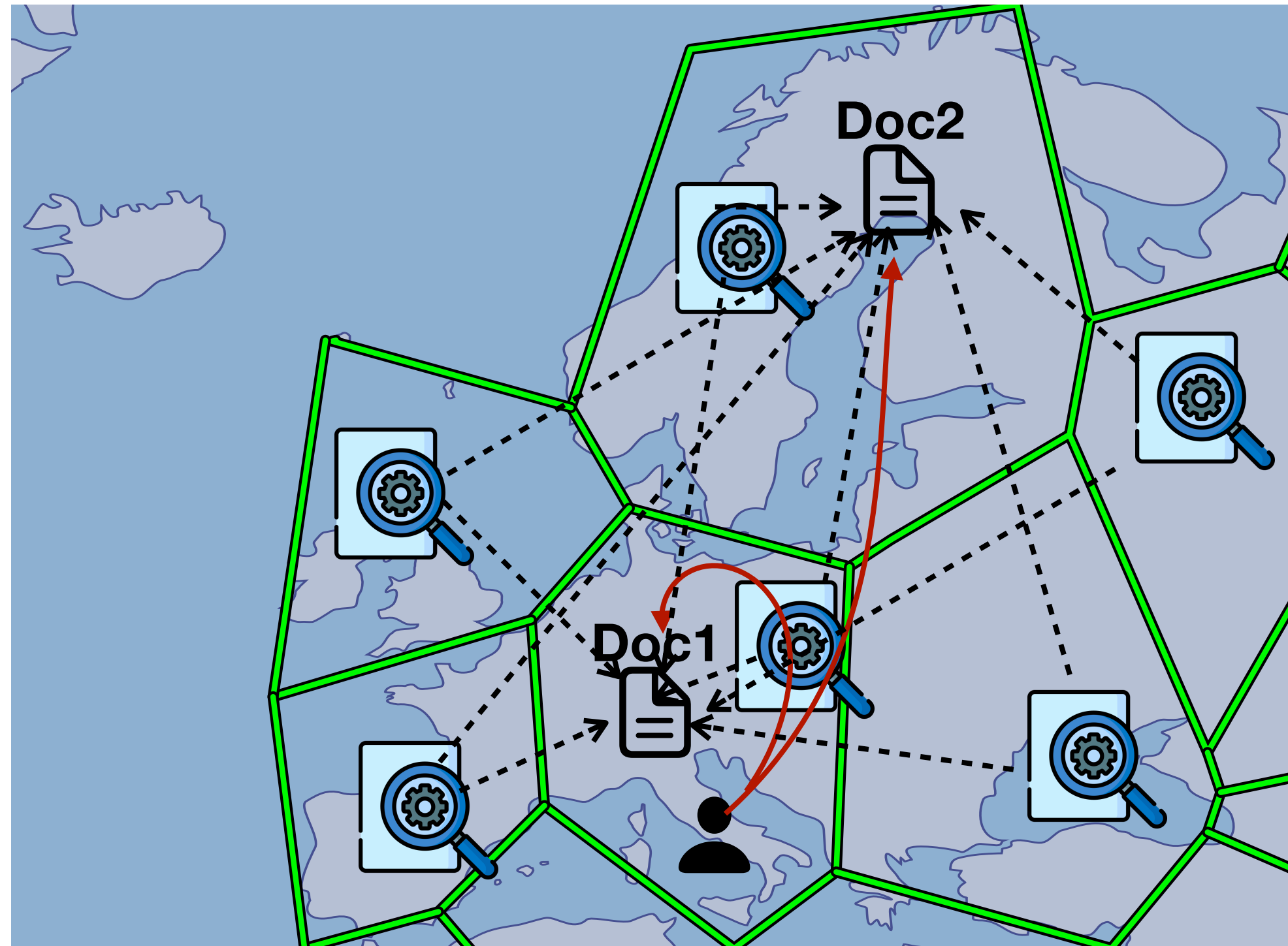
Limix roadmap

- The problem: metadata exposure
- **Design of exposure-limiting metadata service**
- Challenges: strongly-consistent item lookup
- Architecture
- Evaluation



Strawman 1: limits exposure but does not scale

Two documents in a geo-replicated system
(replicas not depicted)

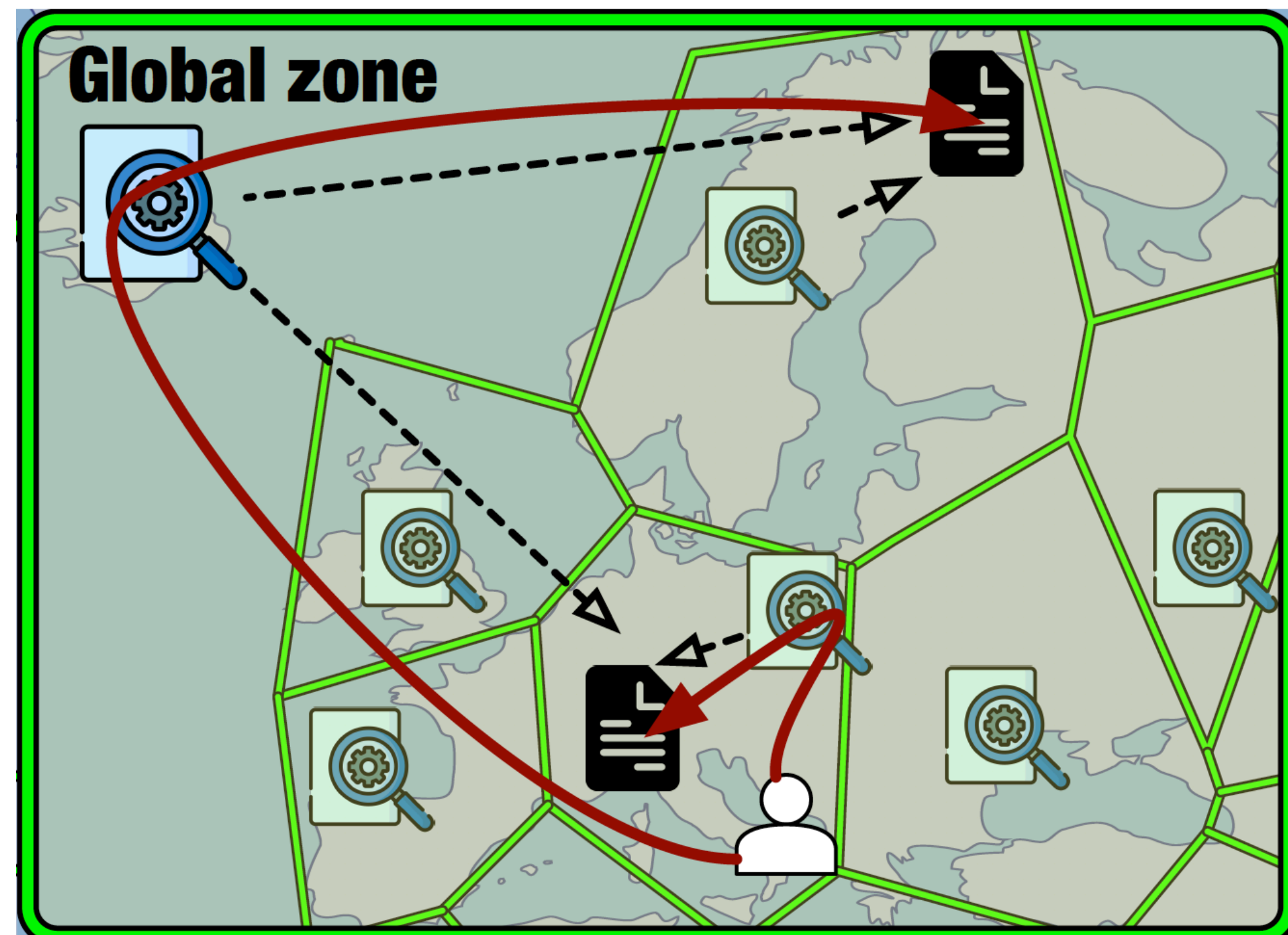


--> Data pointer

- ✓ **Challenge 1:**
Maintain Lamport exposure while performing file lookup
Solution:
Replicate the discovery service
Per-zone discovery service, each service points to all data
- ✗ **Challenge 2:**
Scalable discovery service enforcing Lamport exposure

Strawman 2: scales but provides coarse-grained exposure

Two documents in a geo-replicated system
(replicas not depicted)



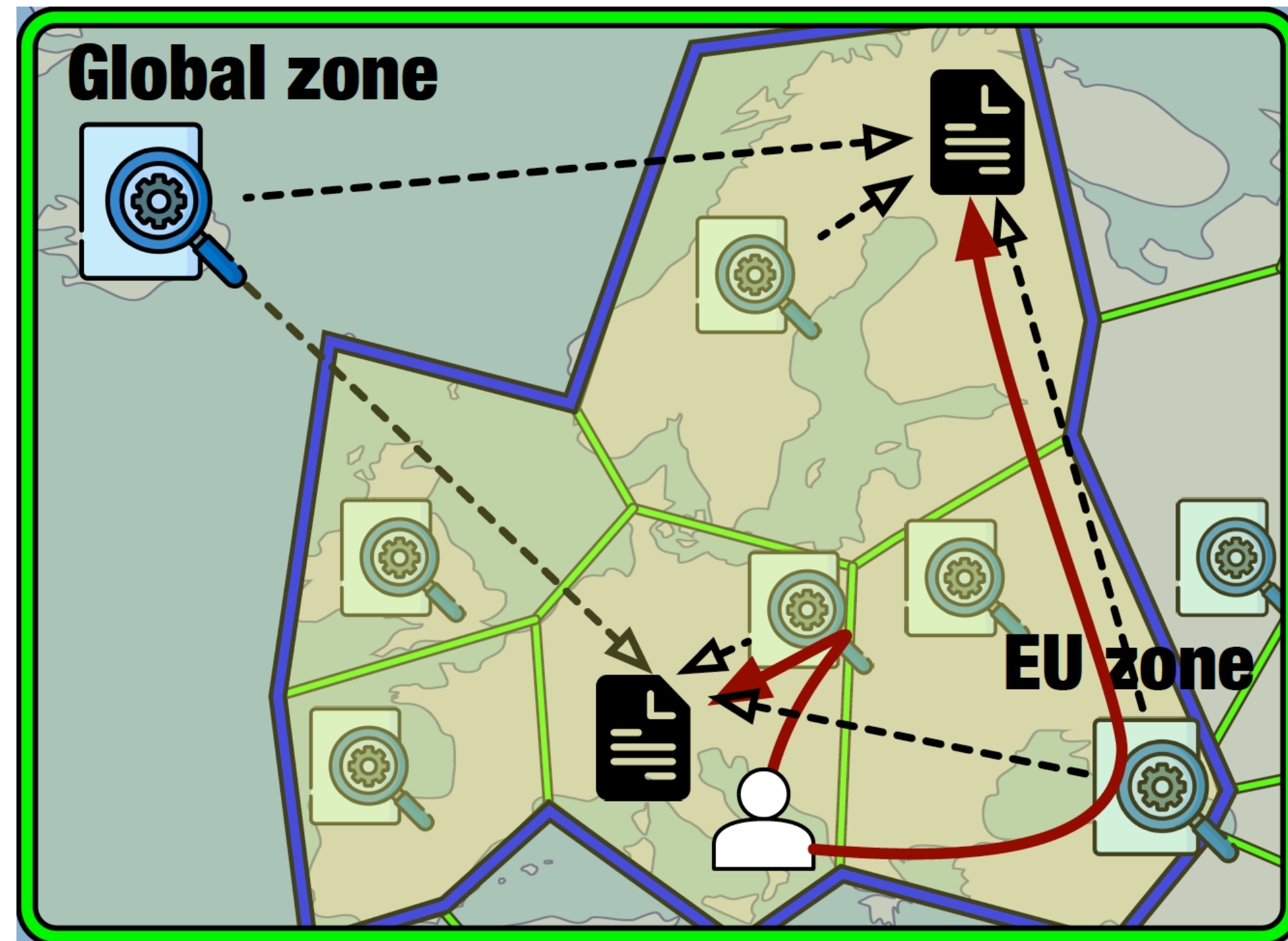
✓ **Challenge 1:**
Maintain Lamport exposure while performing file lookup
Solution:
Replicate the discovery service

✓ **Challenge 2:**
Scalable discovery service enforcing Lamport exposure
Solution:
Add a global zone
The per-zone discovery service points to local data only

✗ **Challenge 3**
Fine-grained overlapping Lamport exposure

Limix: our proposal for limiting Lamport exposure

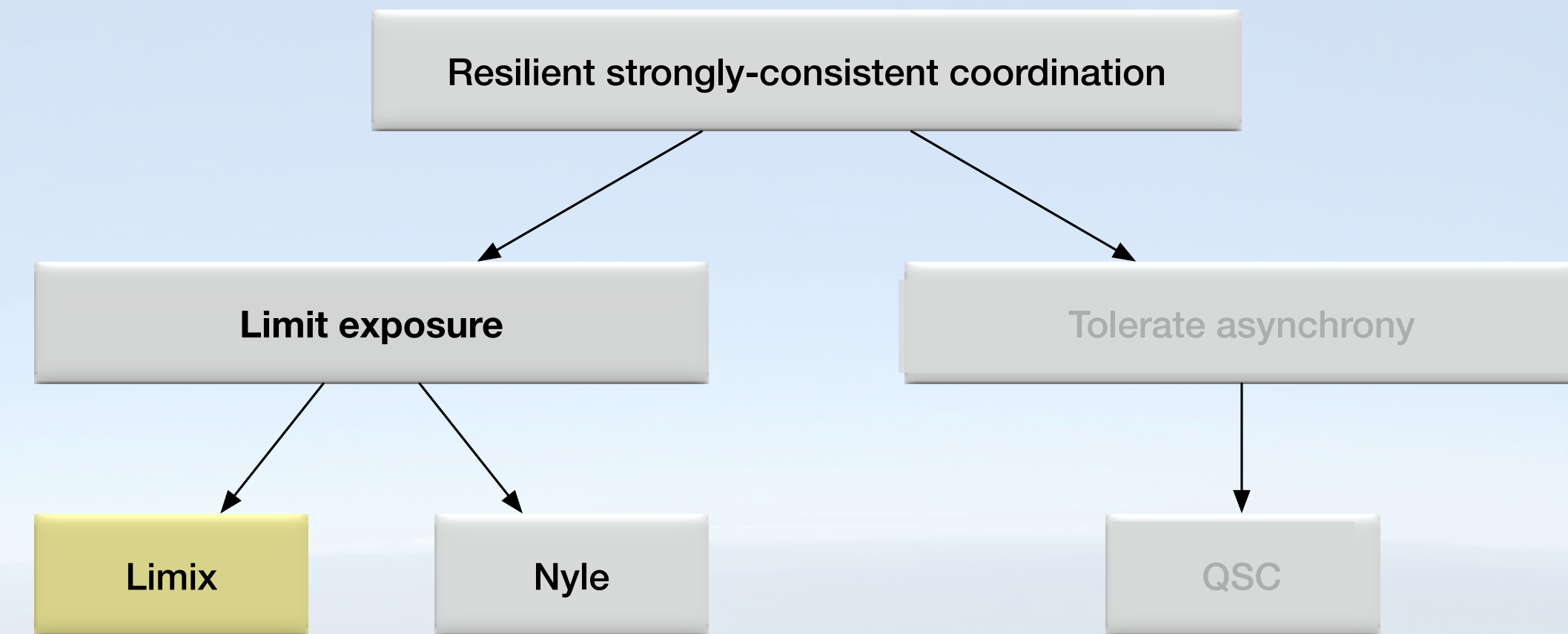
Two documents in a geo-replicated system
(replicas not depicted)



- ✓ **Challenge 1:**
Maintain Lamport exposure while performing file lookup
Solution:
Replicate the discovery service
- ✓ **Challenge 2:**
Scalable discovery service enforcing Lamport exposure
Solution:
Add a global zone
The per-zone discovery service points to local data only
- ✓ **Challenge 3**
Fine-grained overlapping Lamport exposure

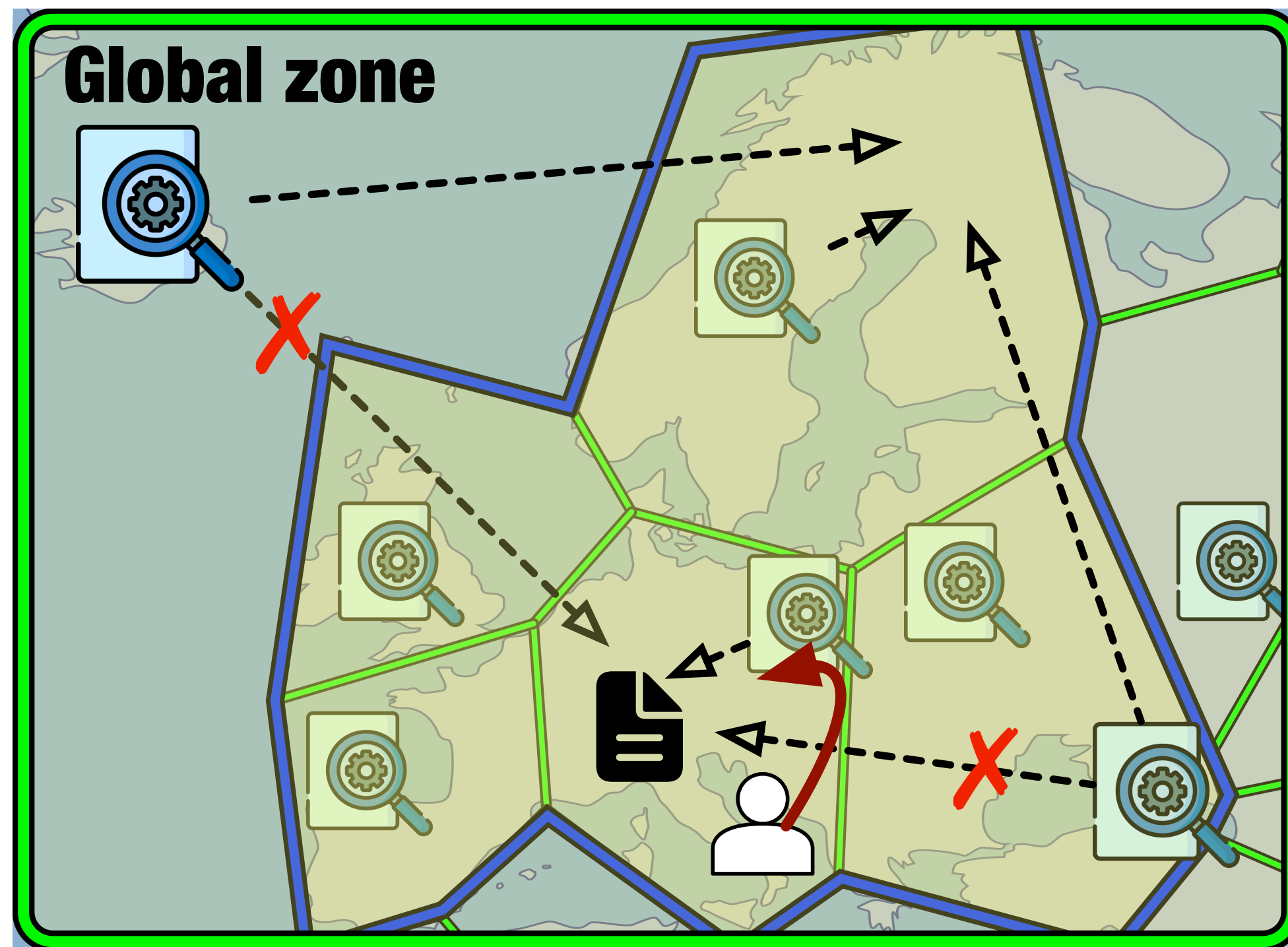
Limix roadmap

- The problem: metadata exposure
- Design of exposure-limiting metadata service
- **Challenges: strongly-consistent item lookup**
- Architecture
- Evaluation



The devil is in the details

Recall **manageability**: support for migrating strongly-consistent data



Challenge 4: Inconsistent metadata during document migration

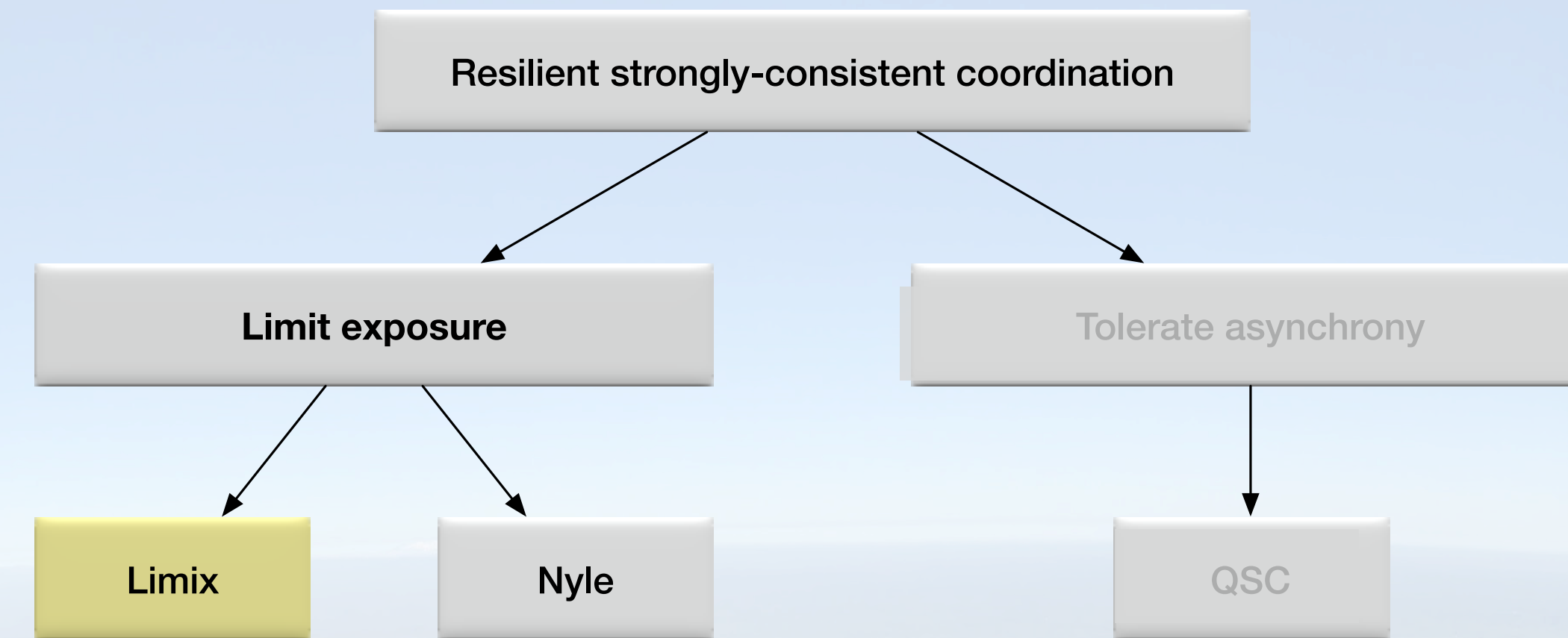
Solution: Update all relevant document pointers

Challenge 5: Updates on different pointers (and different objects) in general) not atomic

Challenge 6: Limit exposure during document migration

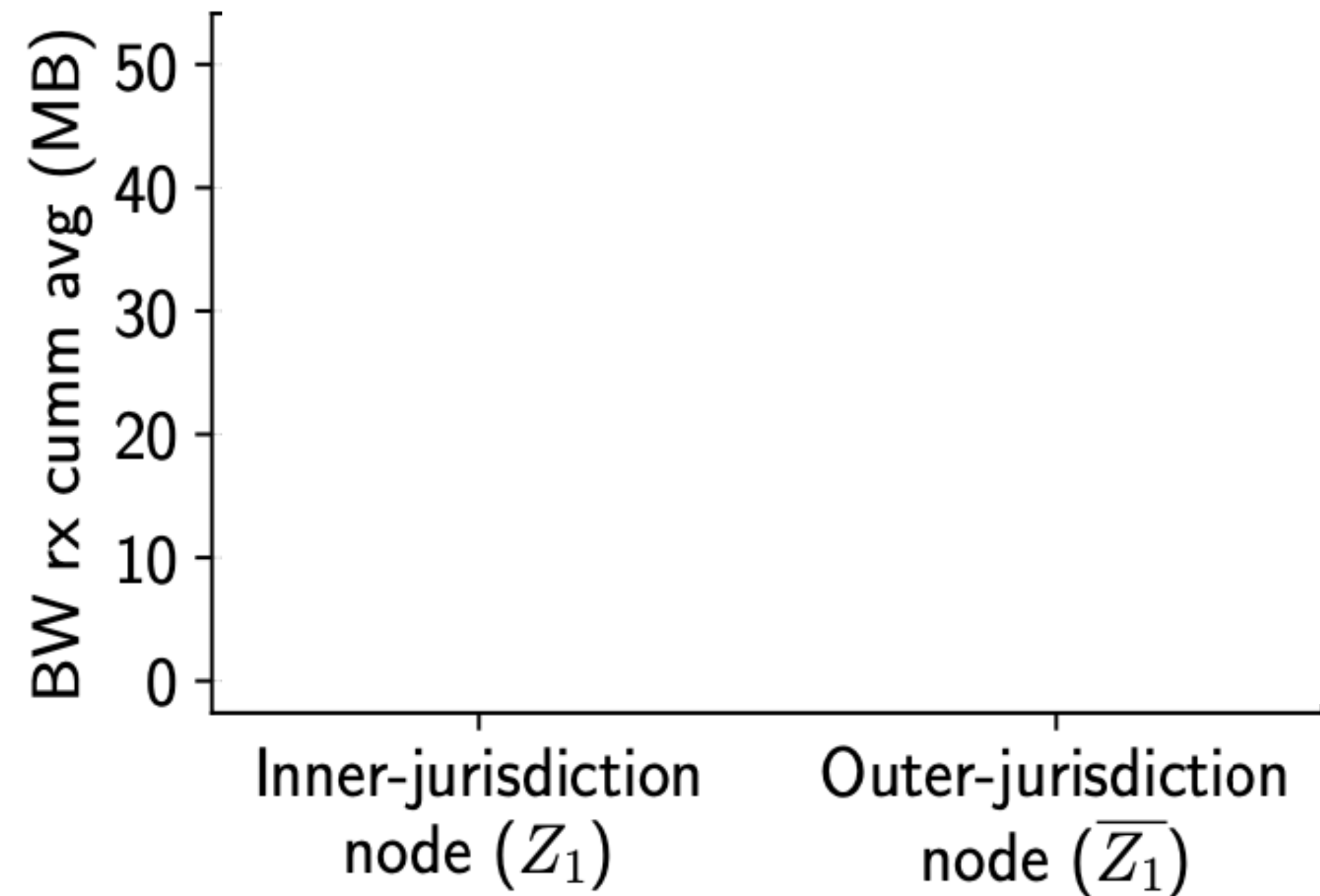
Limix roadmap

- The problem: metadata exposure
- Design of exposure-limiting metadata service
- Challenges: strongly-consistent item lookup
- Architecture
- **Evaluation**
 - **Jurisdictions: availability and costs per zone**
 - Latency-based autozoning: availability microbenchmark of Limix vs Physalia
 - Latency-based autozoning: realistic workloads Limix vs Physalia



Administrative zoning

- Cost per extra region deployed

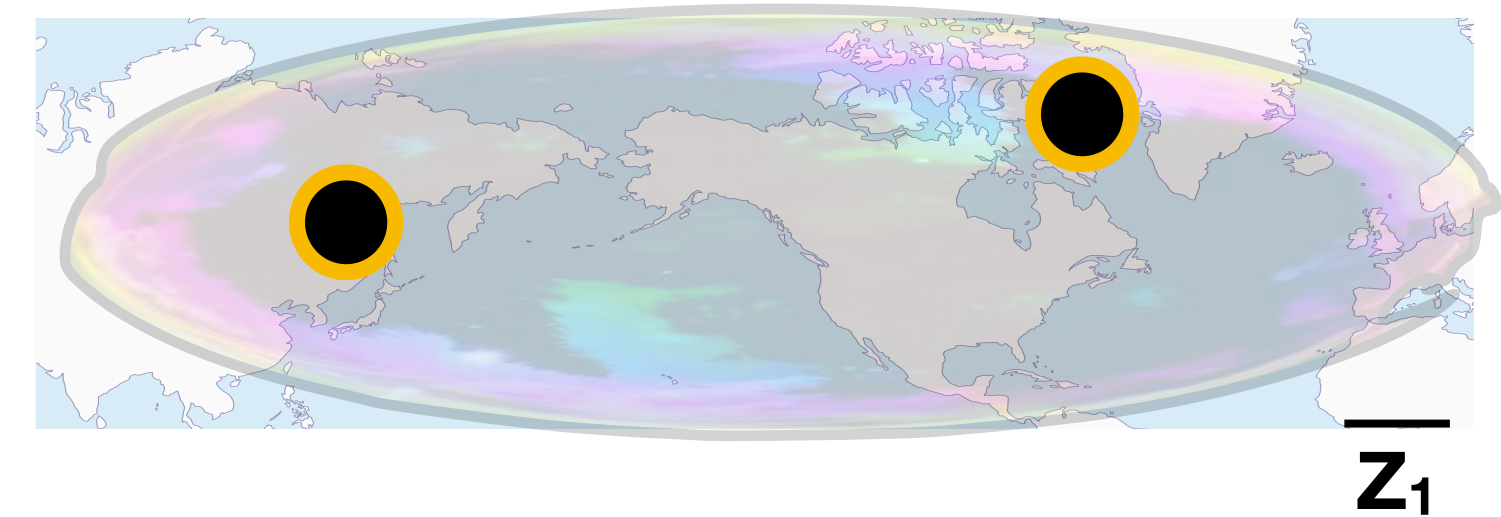


Suitable for a “pay-as-you-go” model

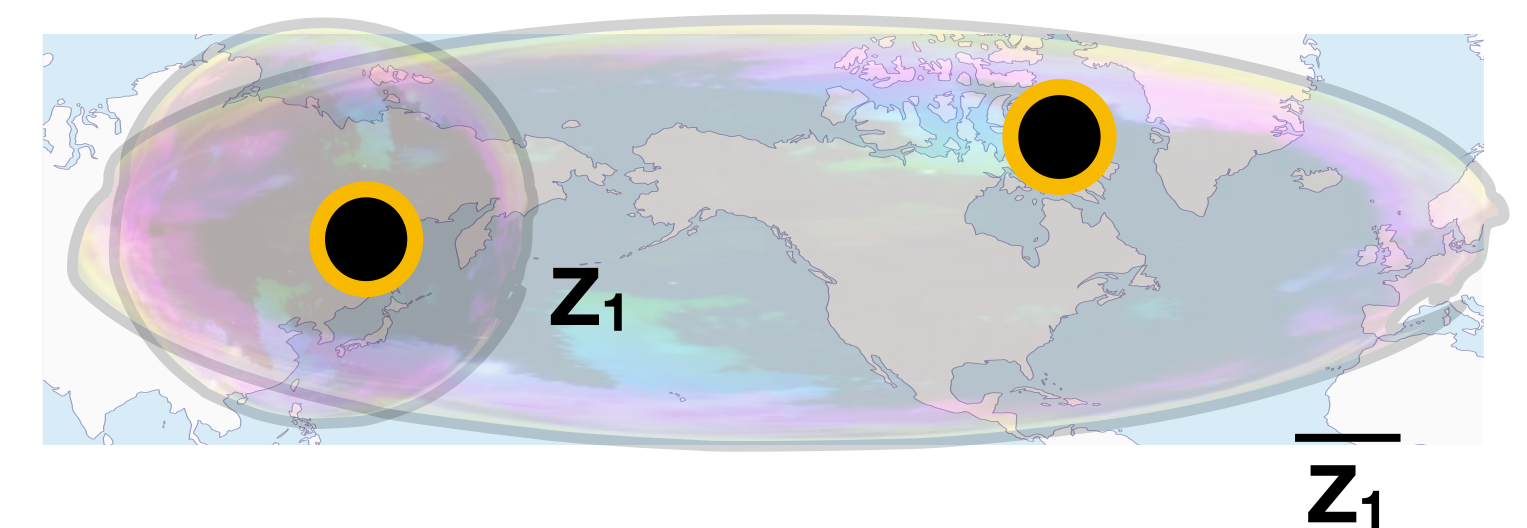
Private cloud



Geo replication

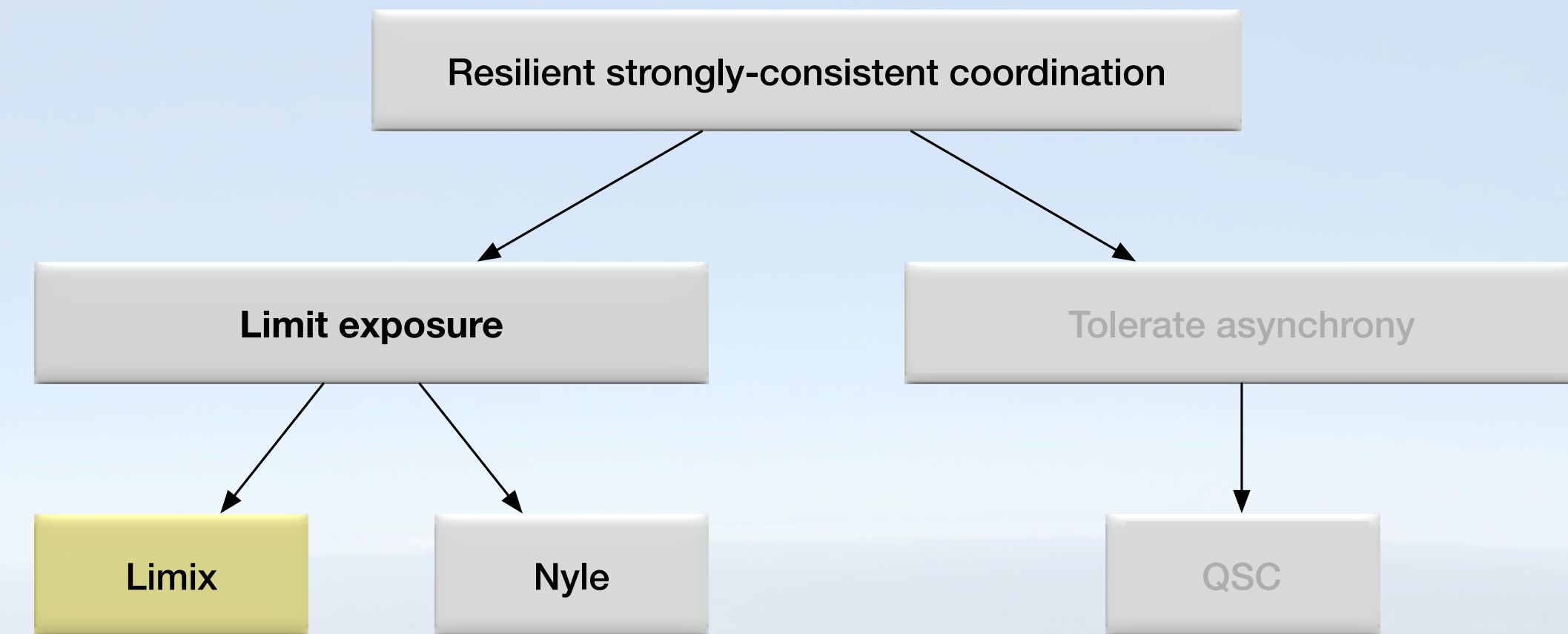


Limix



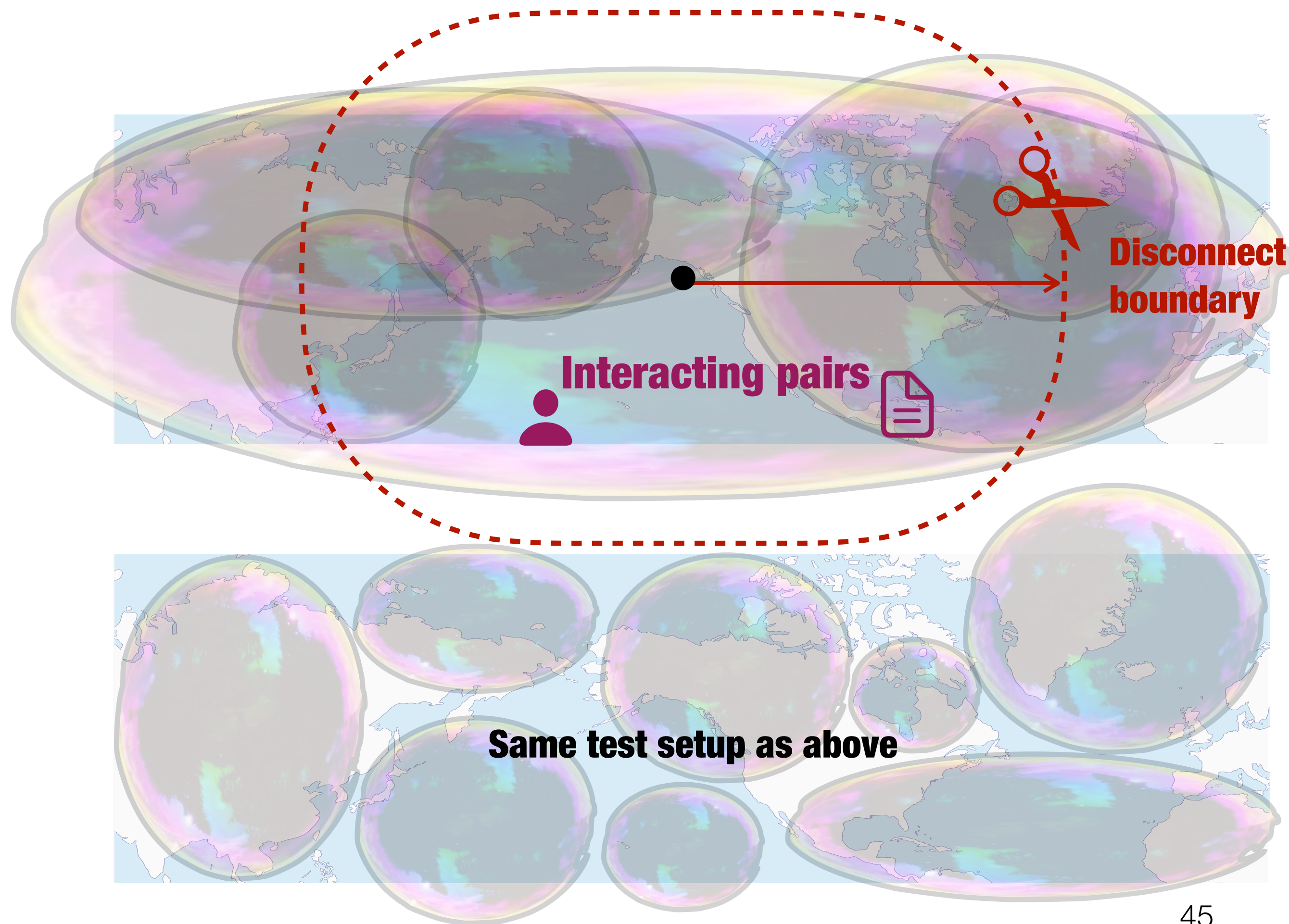
Limix roadmap

- The problem: exposure of data lookup
- Design of exposure-limiting metadata service
- Challenges: strongly-consistent item lookup
- Architecture
- **Evaluation**
 - Jurisdictions: availability and costs per zone
 - **Latency-based autozoning: microbenchmark of Limix vs Physalia**
 - Latency-based autozoning: realistic workloads Limix vs Physalia



Limix vs Physalia setup

- Do far away failures affect accesses?
- **20 AWS sites:** N America, Europe and Asia-Pacific



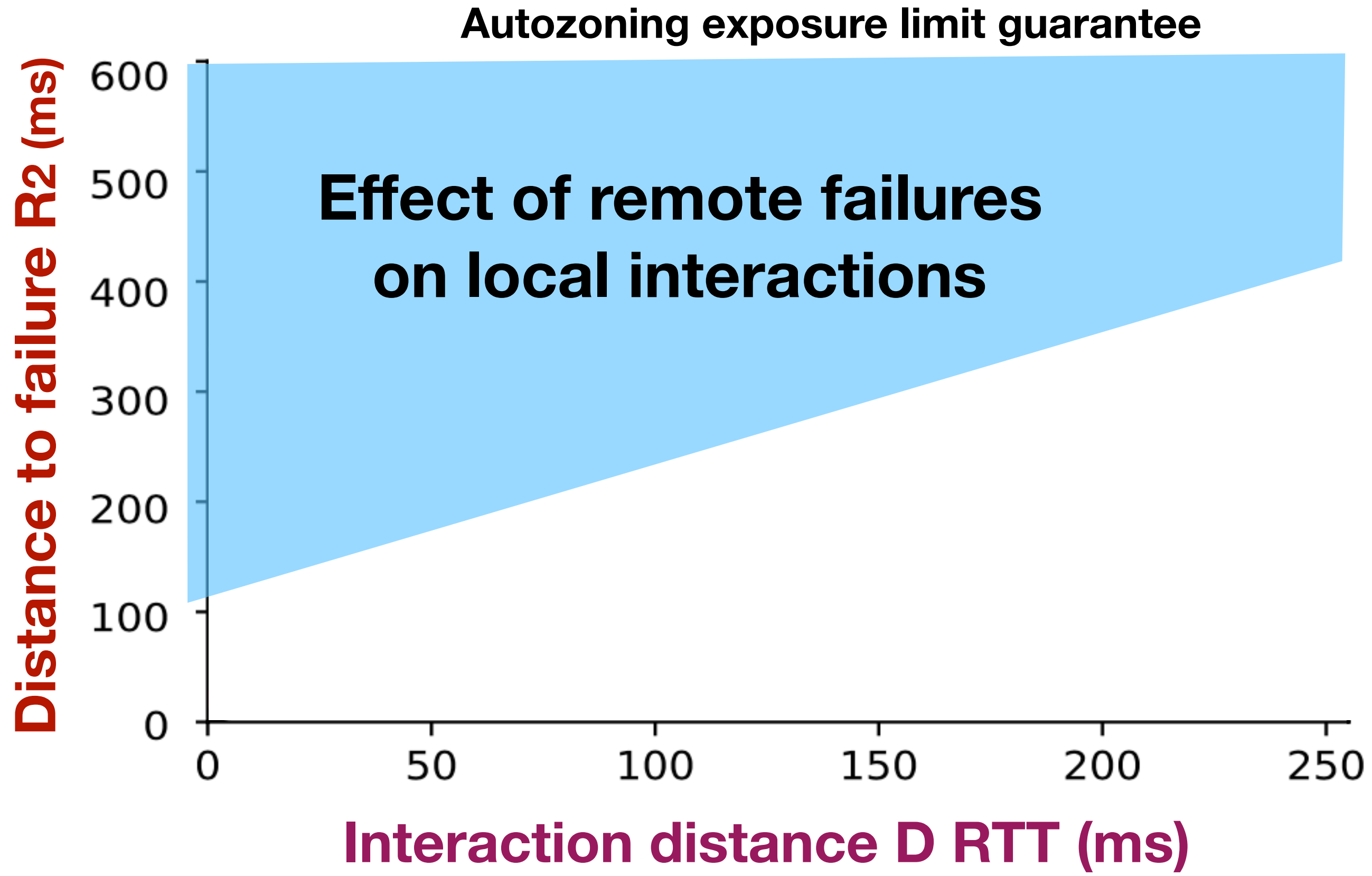
- **Limix latency-based autozoning**

- **Physalia***

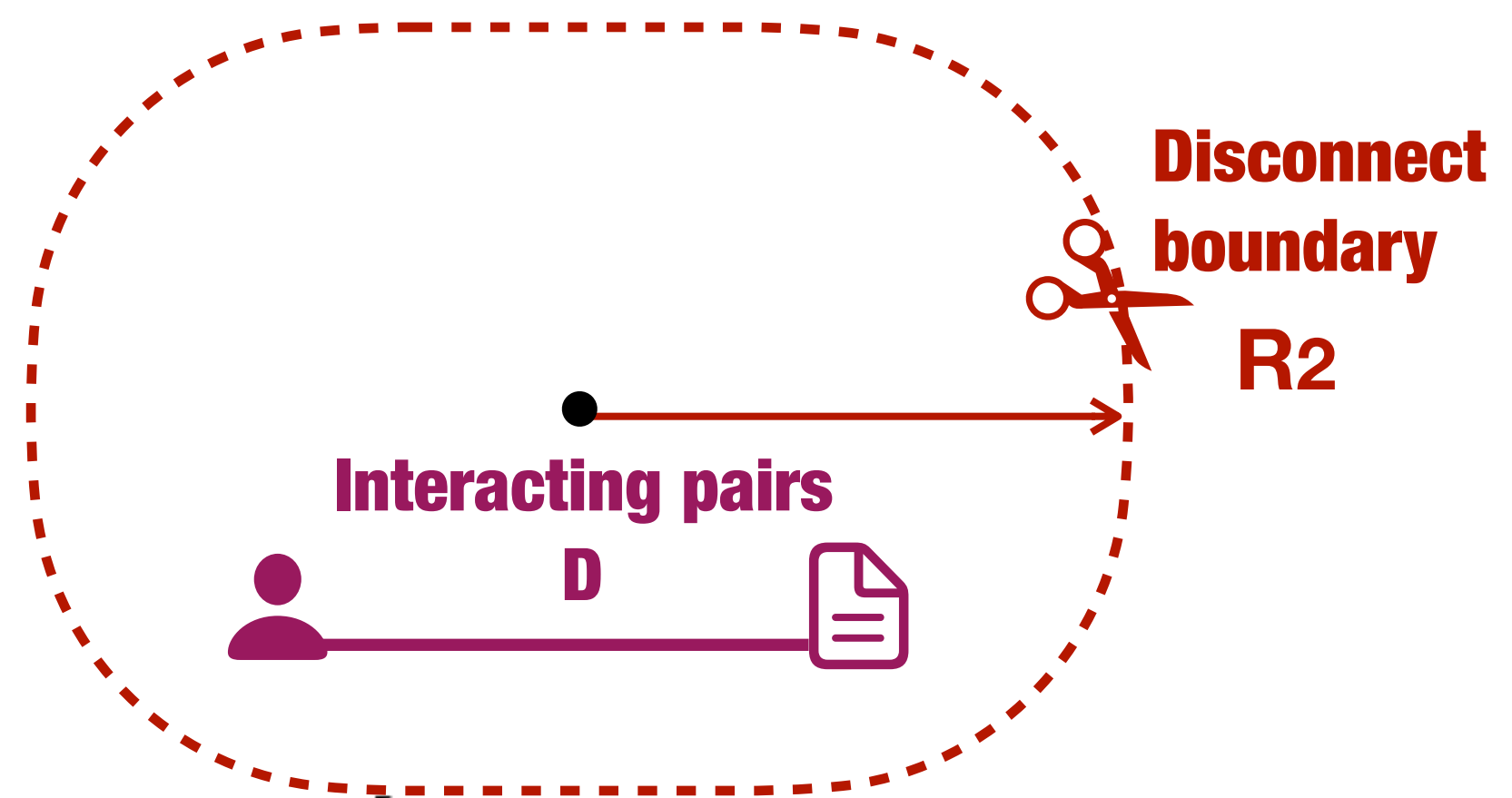
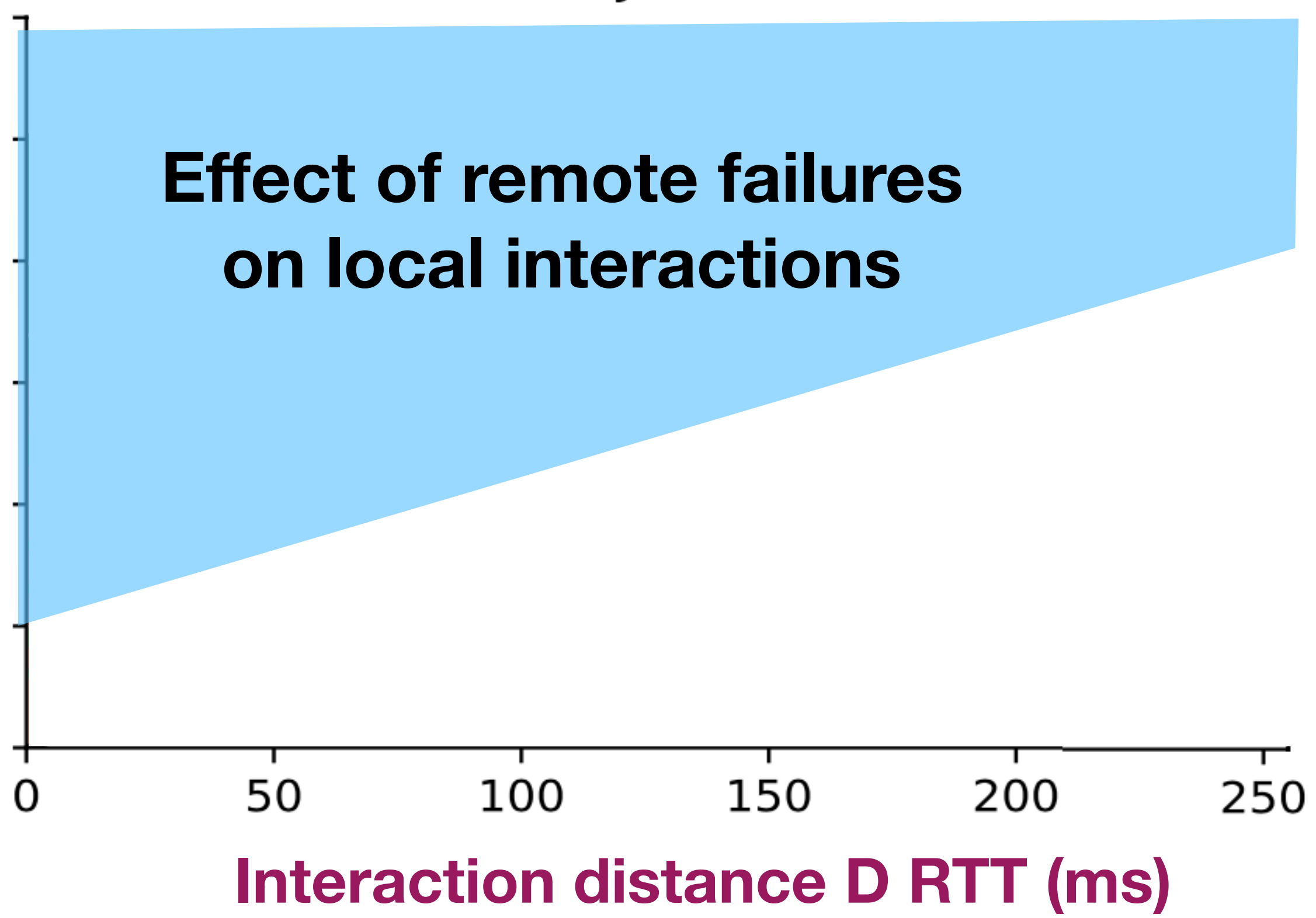
*Marc Brooker, Tao Chen, and Fan Ping. Millions of tiny databases. In *Conference on Networked Systems Design and Implementation (NSDI)*, 2020.

Limix vs Physalia: availability

Limix



Physalia

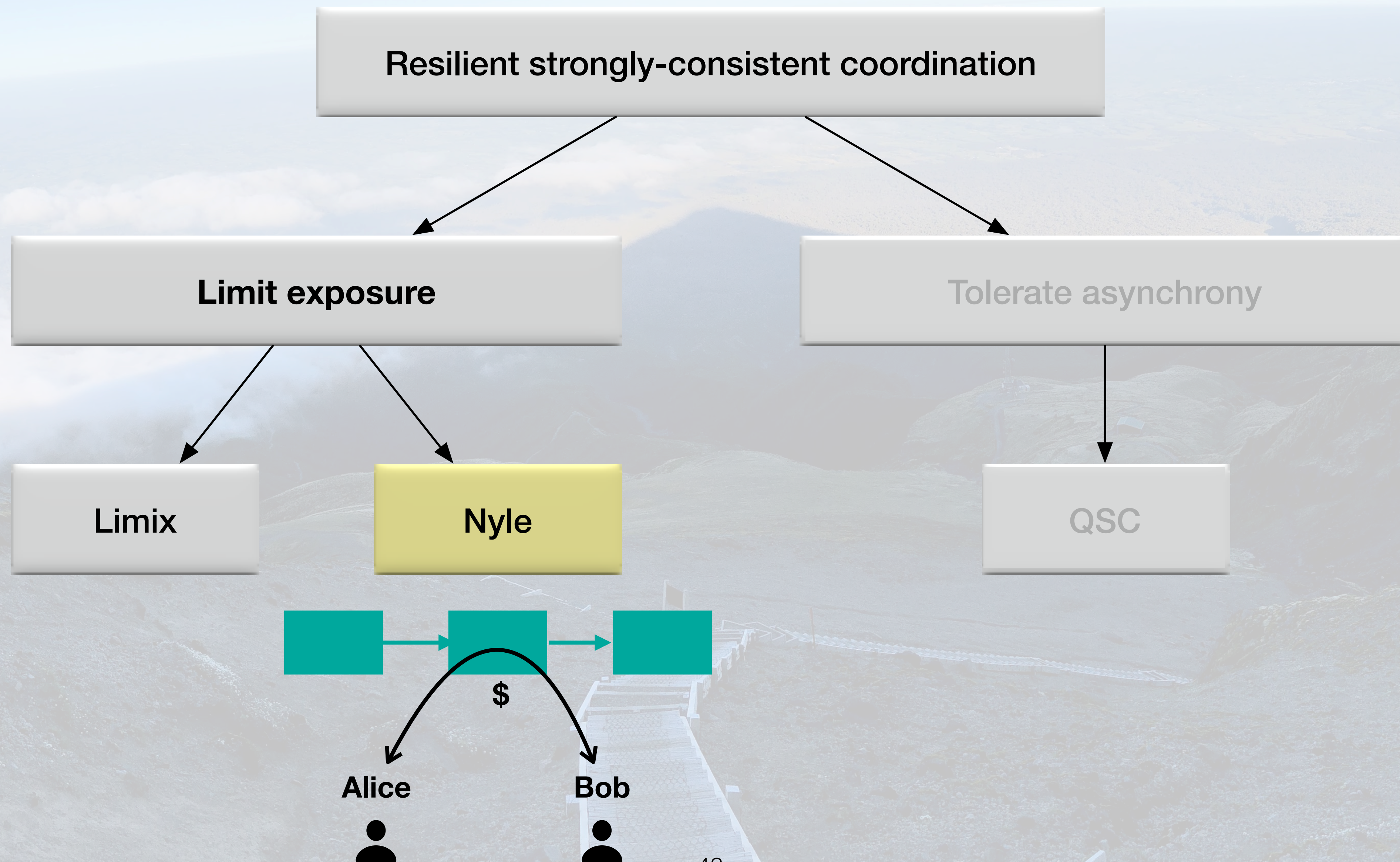


- succeeded
- failed

Limix summary

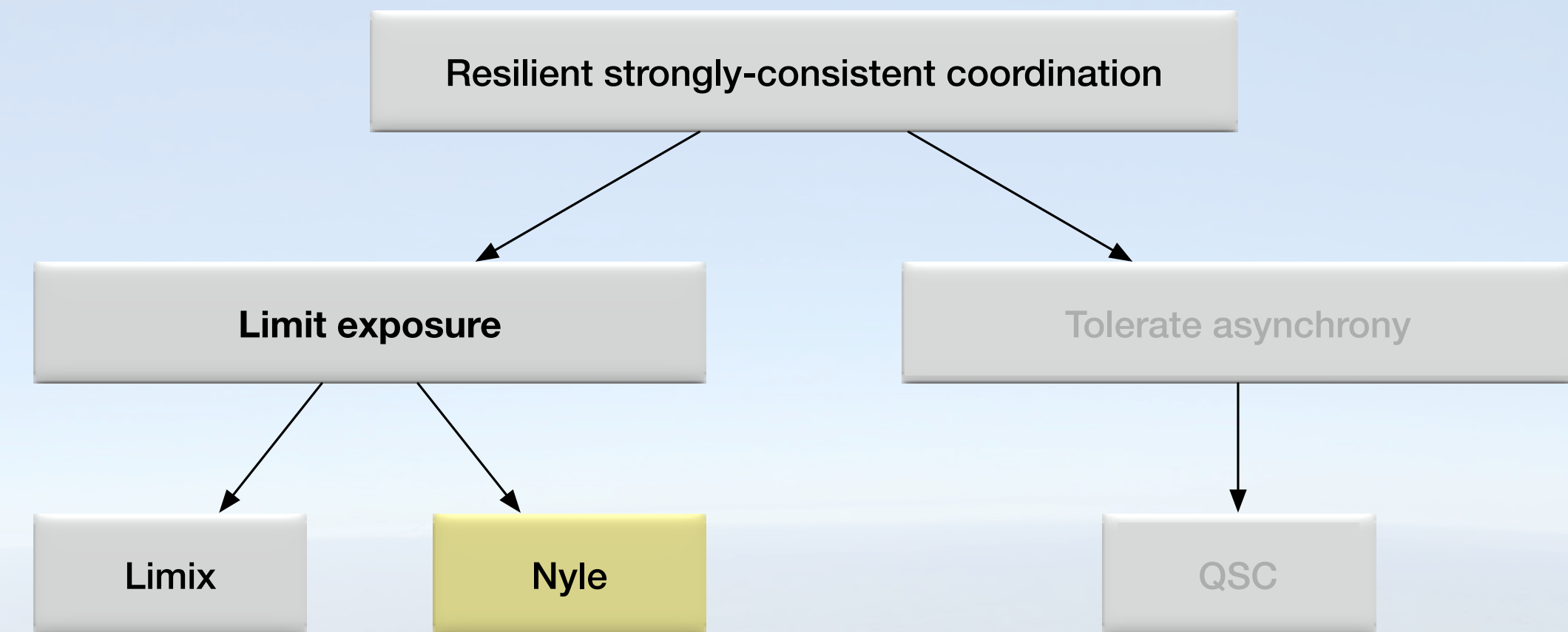
- **Global metadata service that protects local accesses from distant failures**
 - Suitable for strongly consistent data planes
- **Pay-as-you-go administrative zone deployment**
- **Latency-based autozoning outperforms related work**

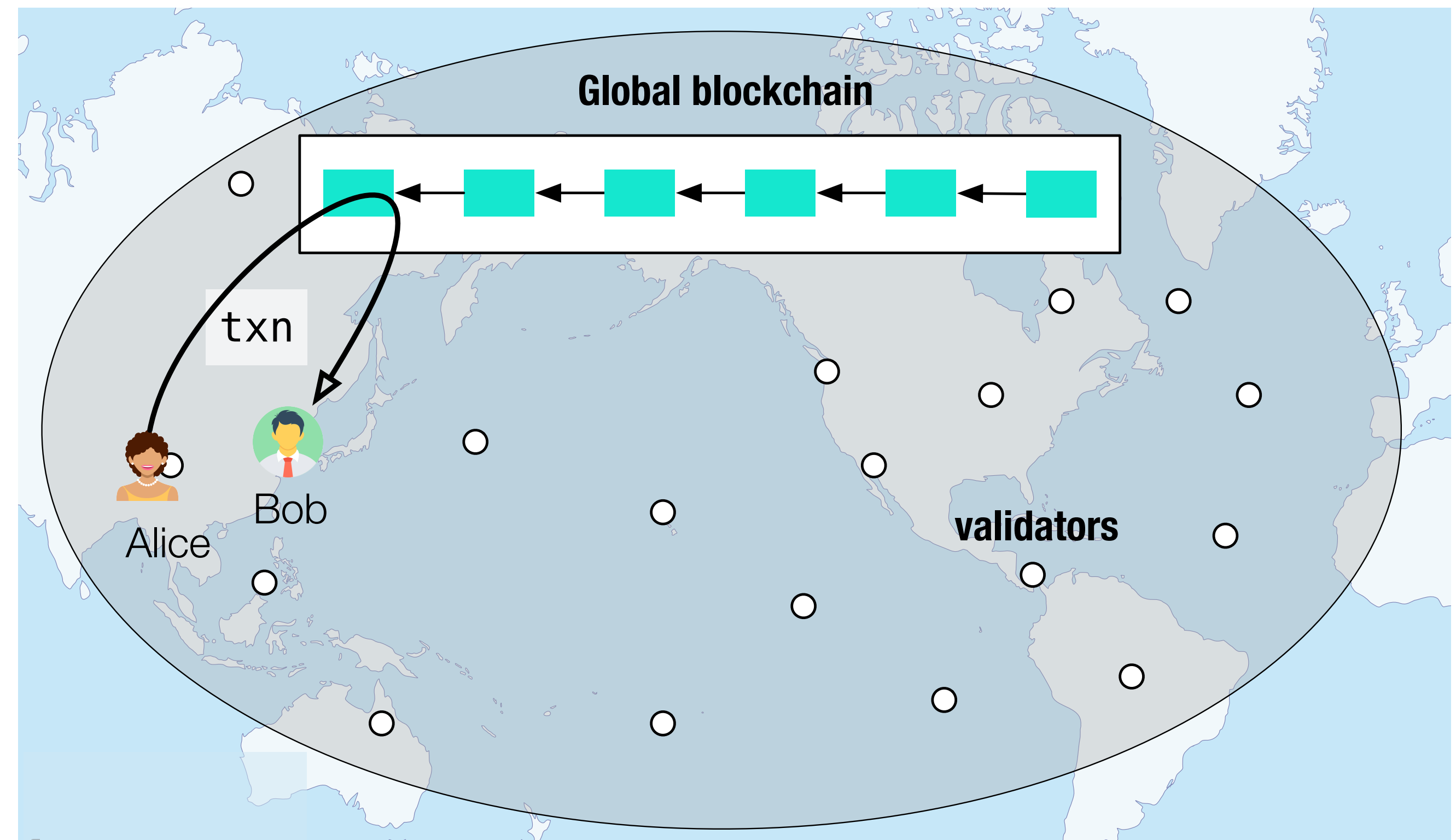
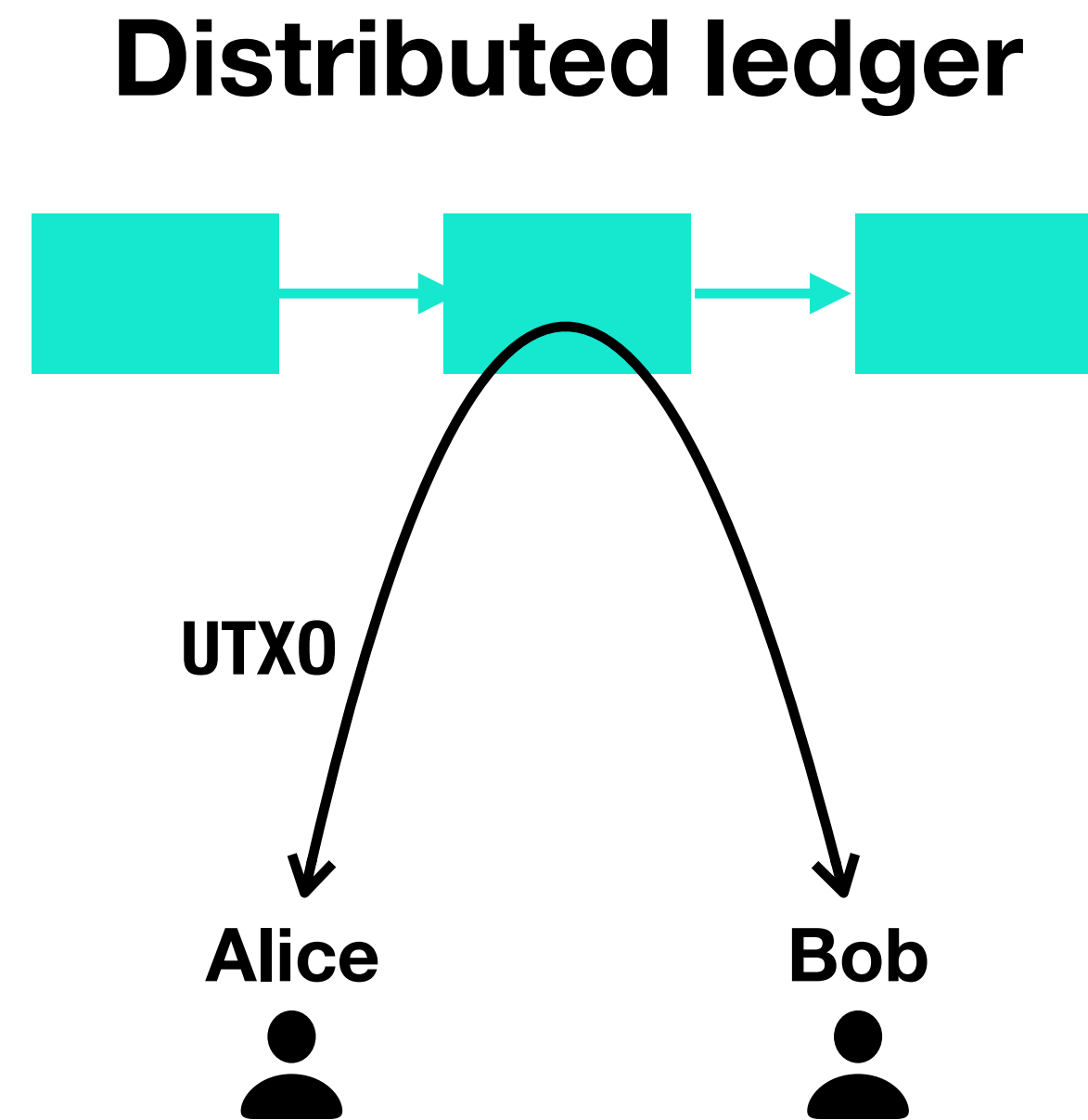
Thesis roadmap



Nyle roadmap

- **Goal: exposure-limiting distributed ledger**
- A trust-but-verify design for transaction validation
- Secure zoning with Byzantine validators
- Challenges: clearance zone
- Simulation





- Do we need round-the-world consensus for local payments?
- Many payments are localized

Setup and goals

- Same availability goal as Limix, but ...

- **Users and validators (< 1/3 globally) can be Byzantine**

Client attacks

Double spend

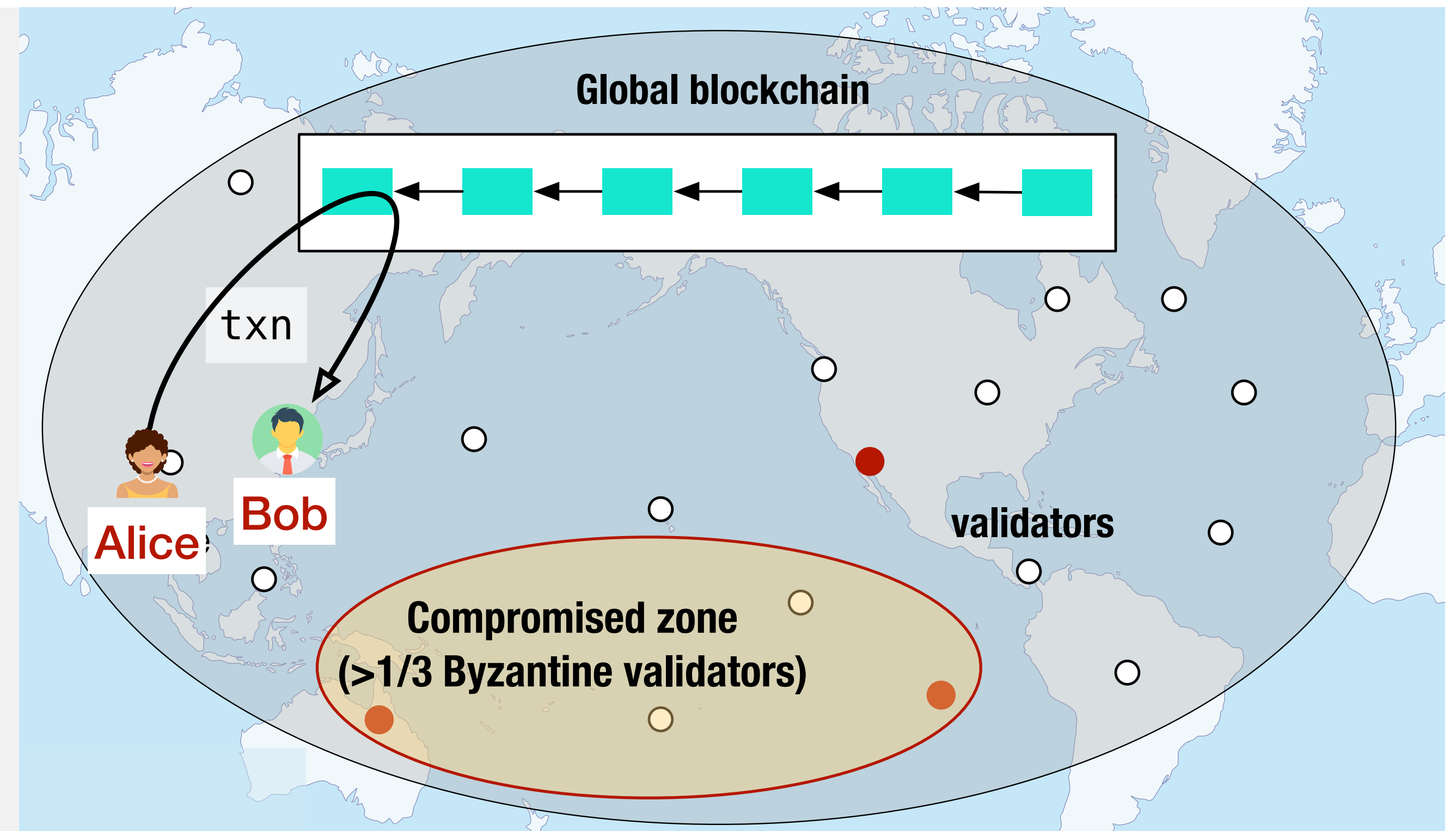
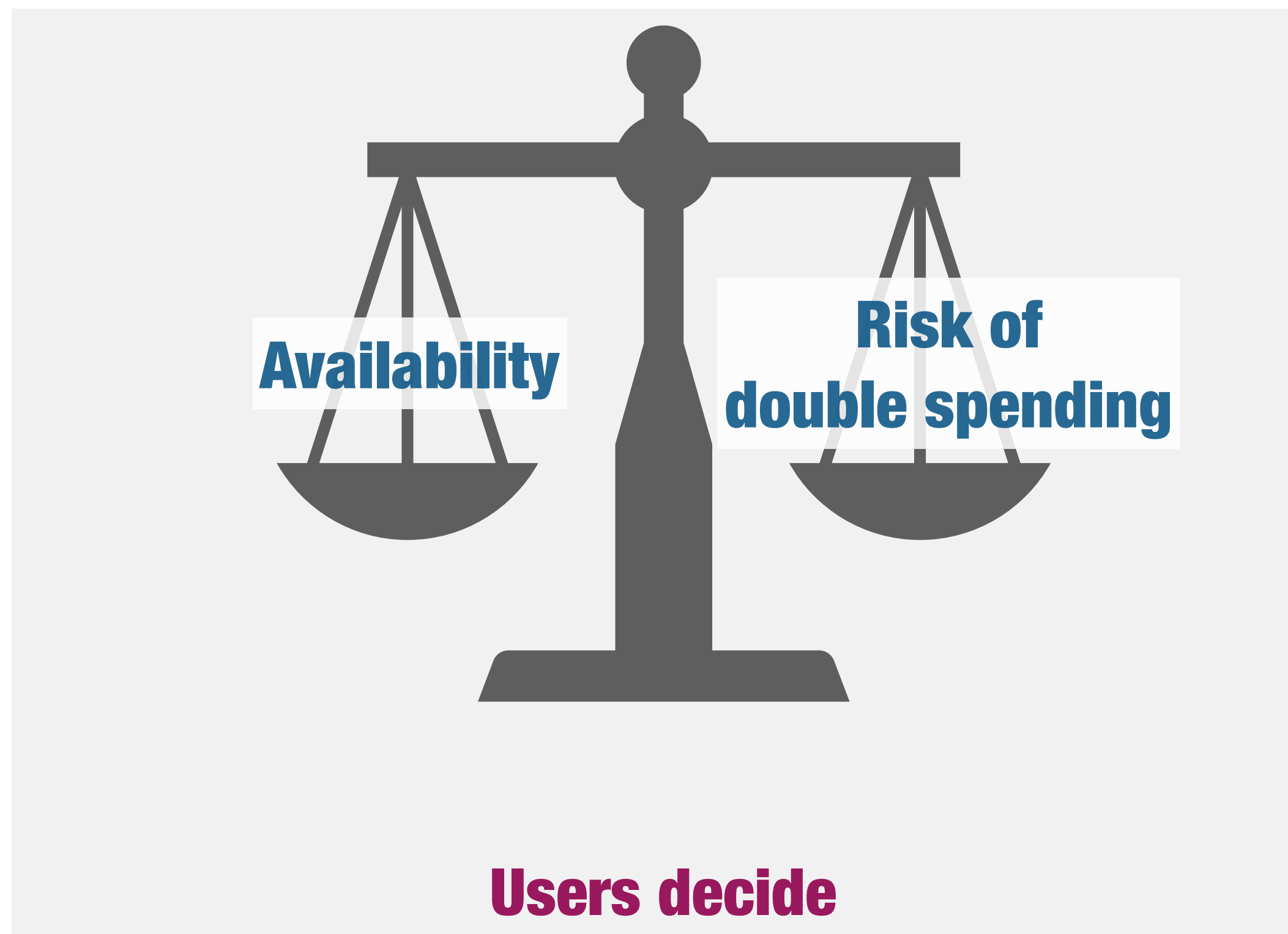
Zone overload

Validator attacks

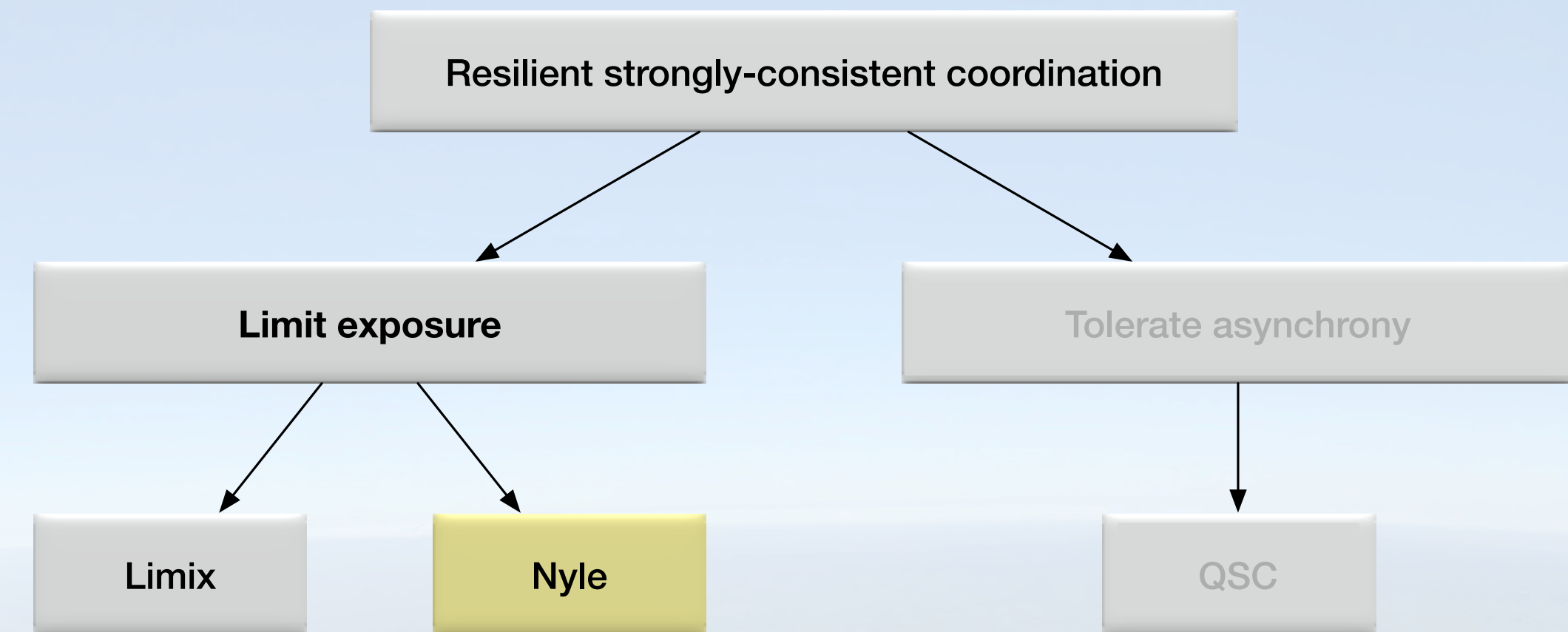
Approve double spend

Censor clients

Craft its location

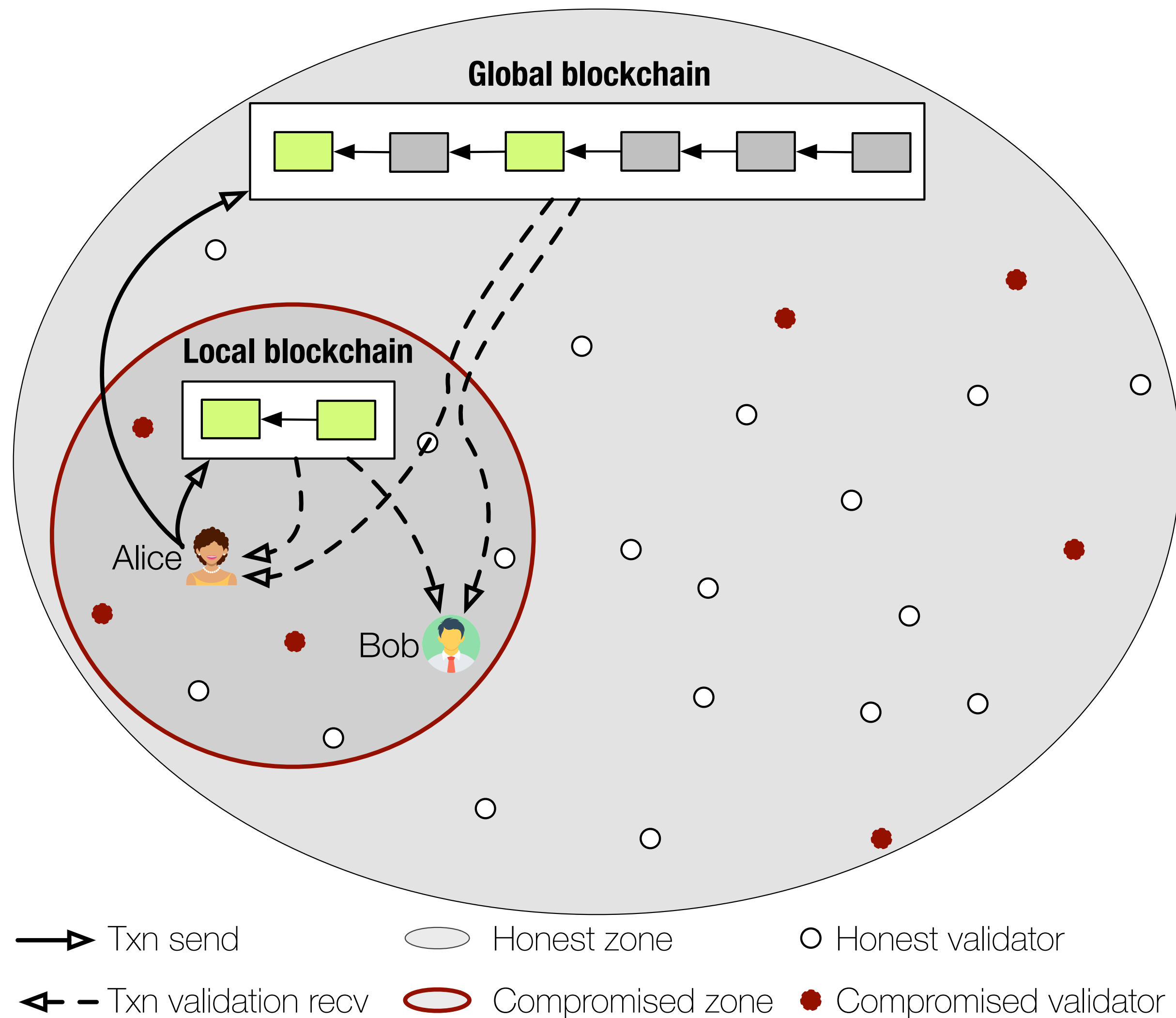


Nyle roadmap



- Goal: exposure-limiting distributed ledger
- **A trust-but-verify design for transaction validation**
- Secure zoning with Byzantine validators
- Challenges: clearance zone
- Simulation

Trust-but-verify design



Global blockchain

Prevents double spending

No local availability

Local blockchain validations

Local availability if local zone honest

Censorship and double spending if local zone compromised

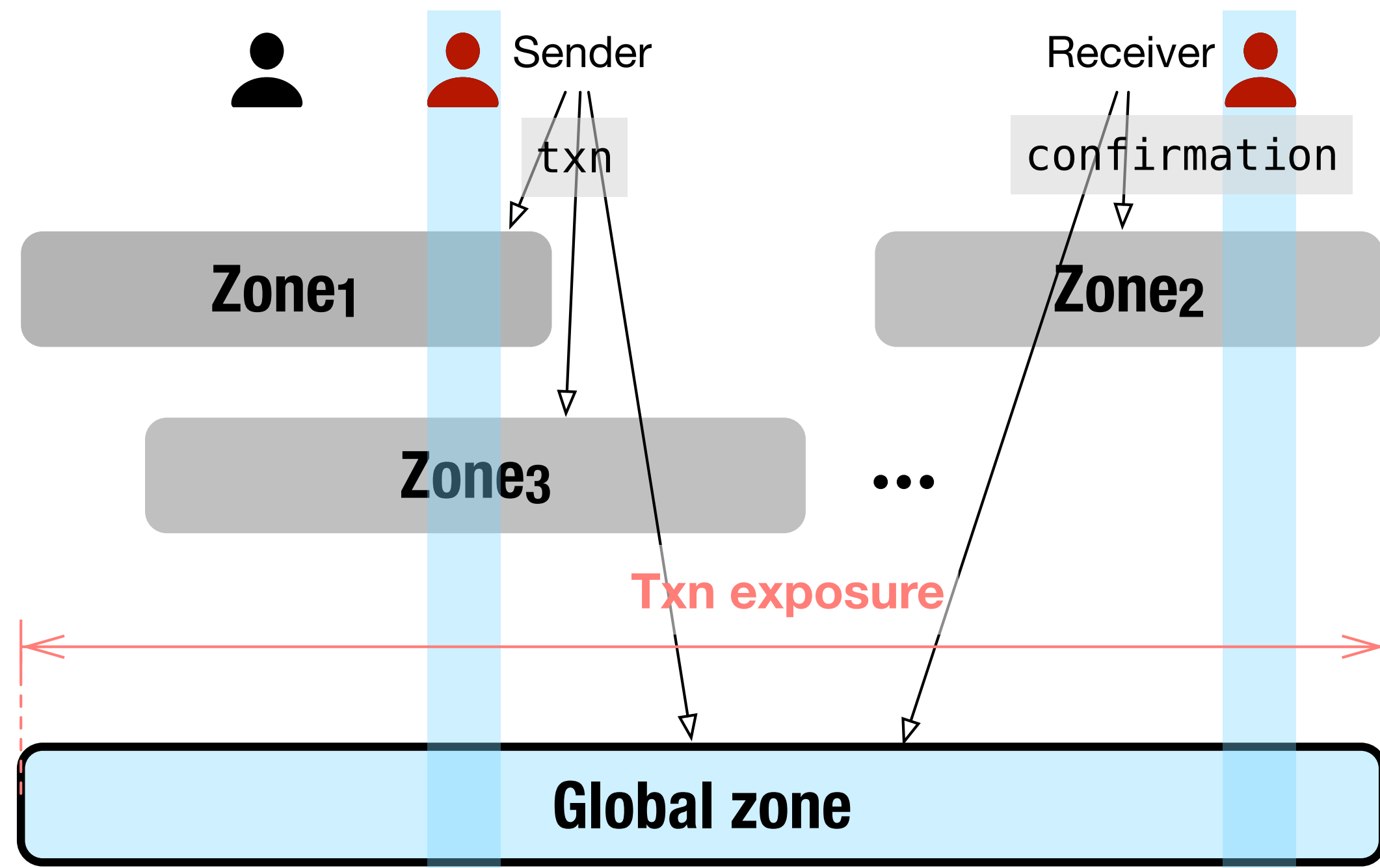
Trust-but-verify

Local availability if local zone honest

Retroactively detect local double spending

User's choice

Transaction validation

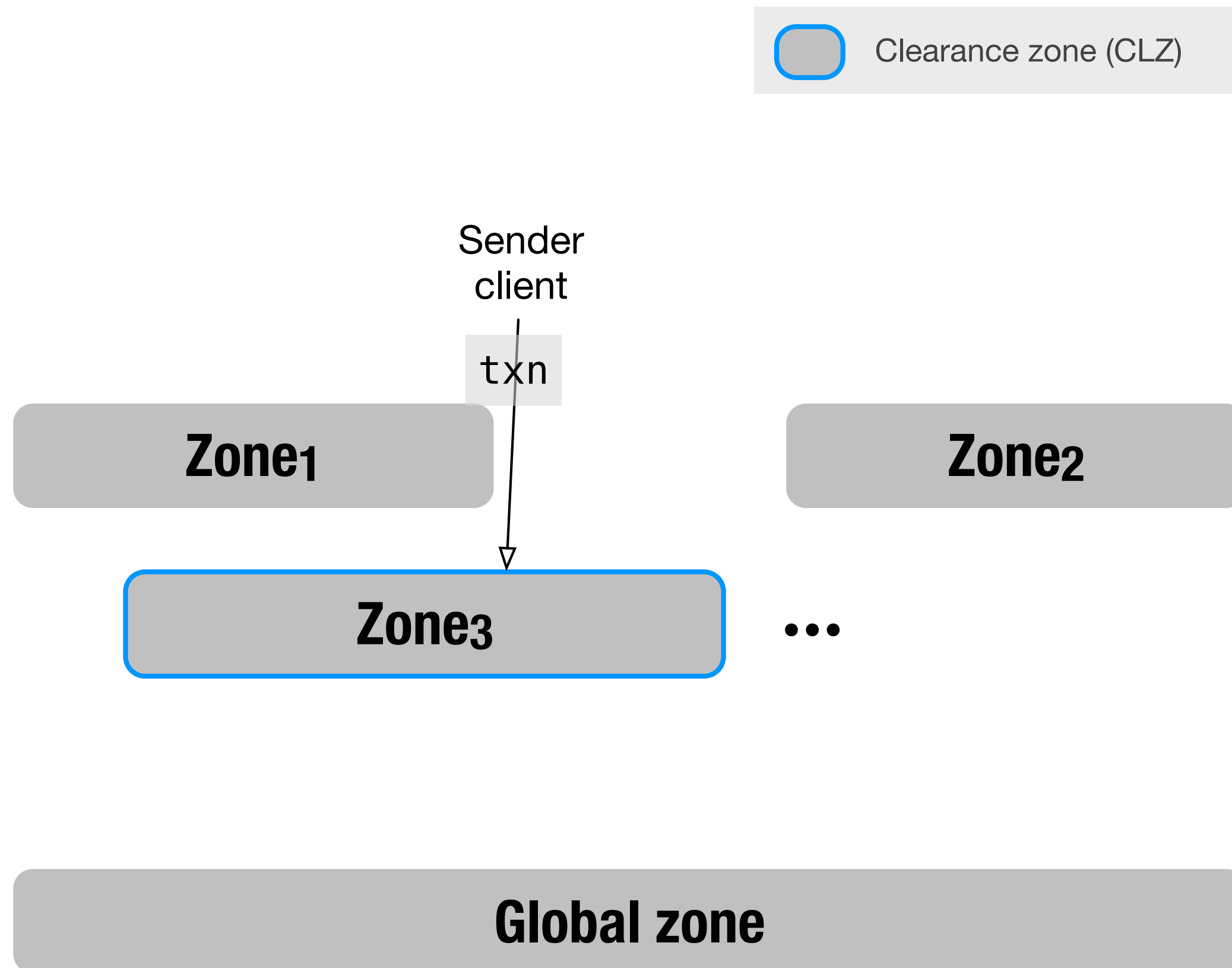


Challenge: Zones can be compromised

Solution: Enable users to decide and retroactively verify

Challenge: Users can be malicious

Transaction validation: attacks



Challenge 1: Sender S double spends via more zones

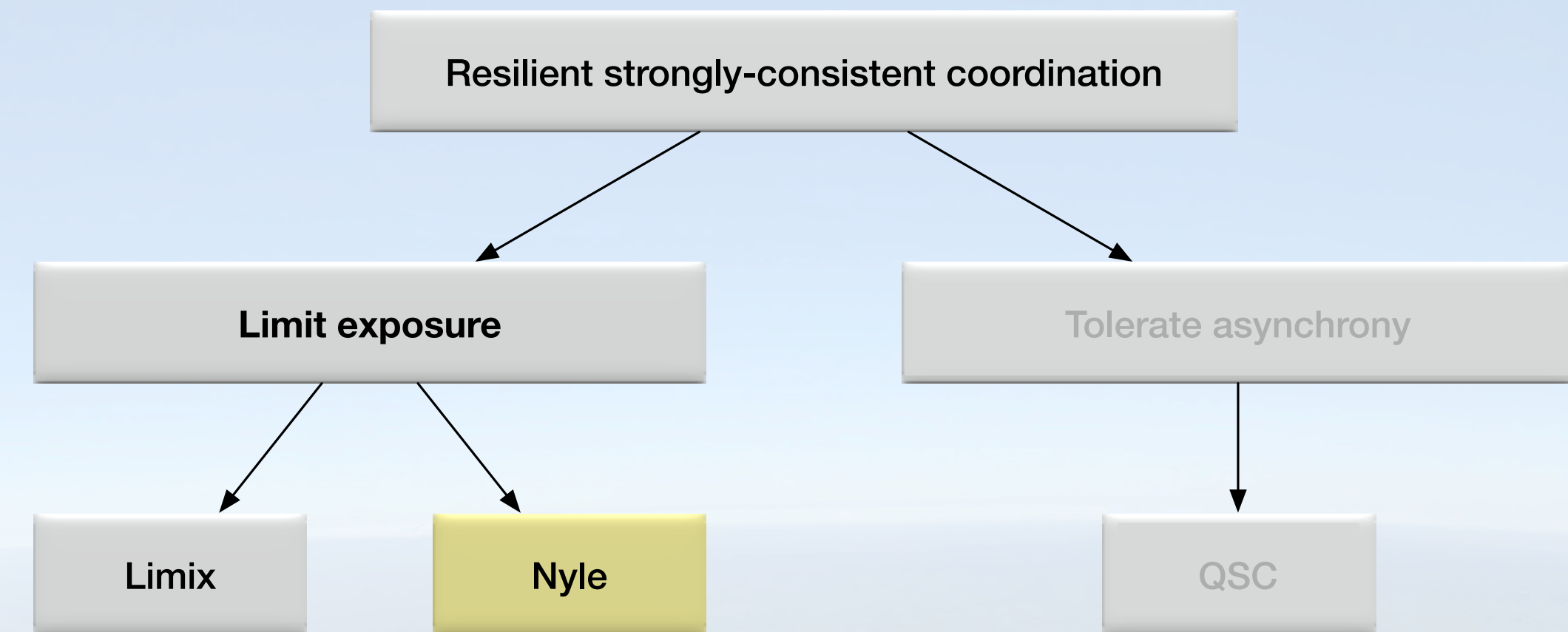
Solution: One clearance zone (CLZ) per UTXO approves spend

Challenge 2: Sender S overloads zones

Challenge 3: Joint sender-receiver control over exposure

Challenge 4: CLZ trusted by owner of UTXO

Nyle roadmap

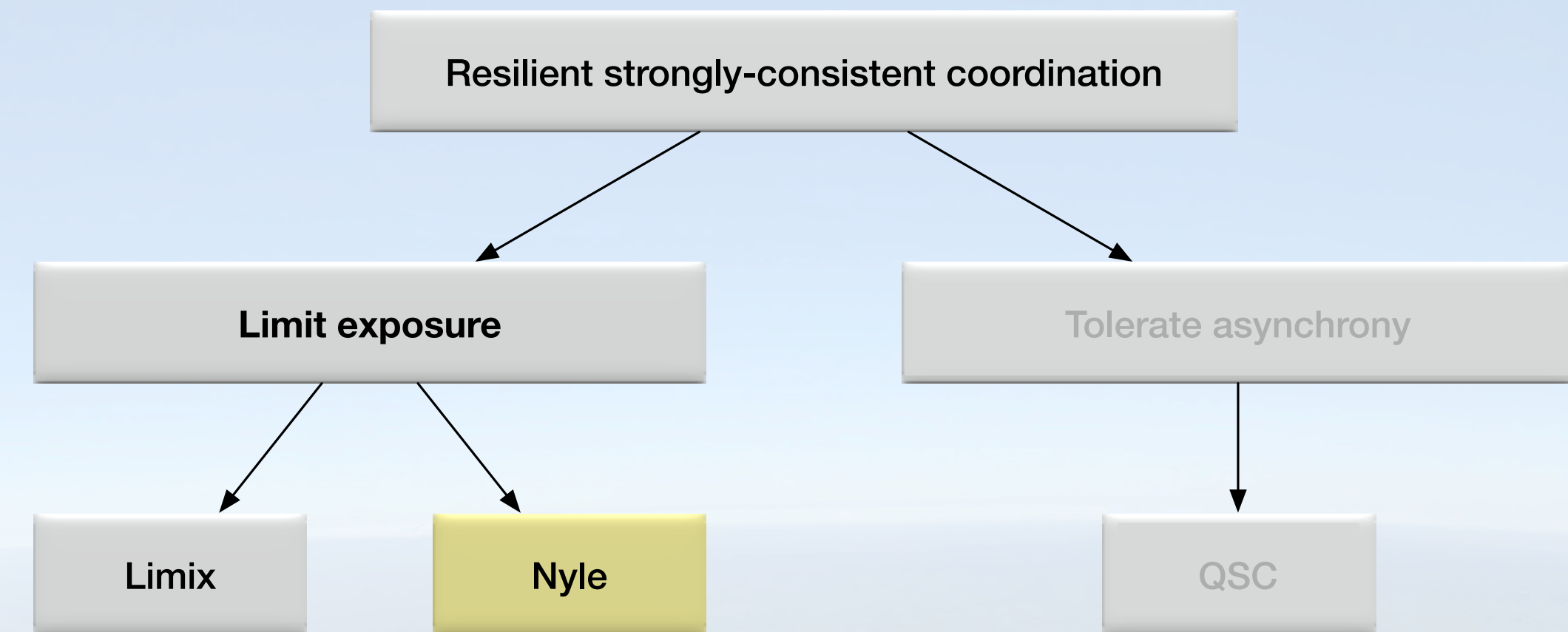


- Goal: exposure-limiting distributed ledger
- A trust-but-verify design for transaction validation
- Secure zoning with Byzantine validators
- **Challenges: clearance zone**
- Simulation

Clearance zone (CLZ) plays a central role

- **First point of contact for a transaction**
- **Might need to change**
 - To **adapt exposure** when user changes location
 - For **multi-input transactions**, the user transfers funds to a single zone
 - User trust in zone / regulations change
- **CLZ censorship of transactions**

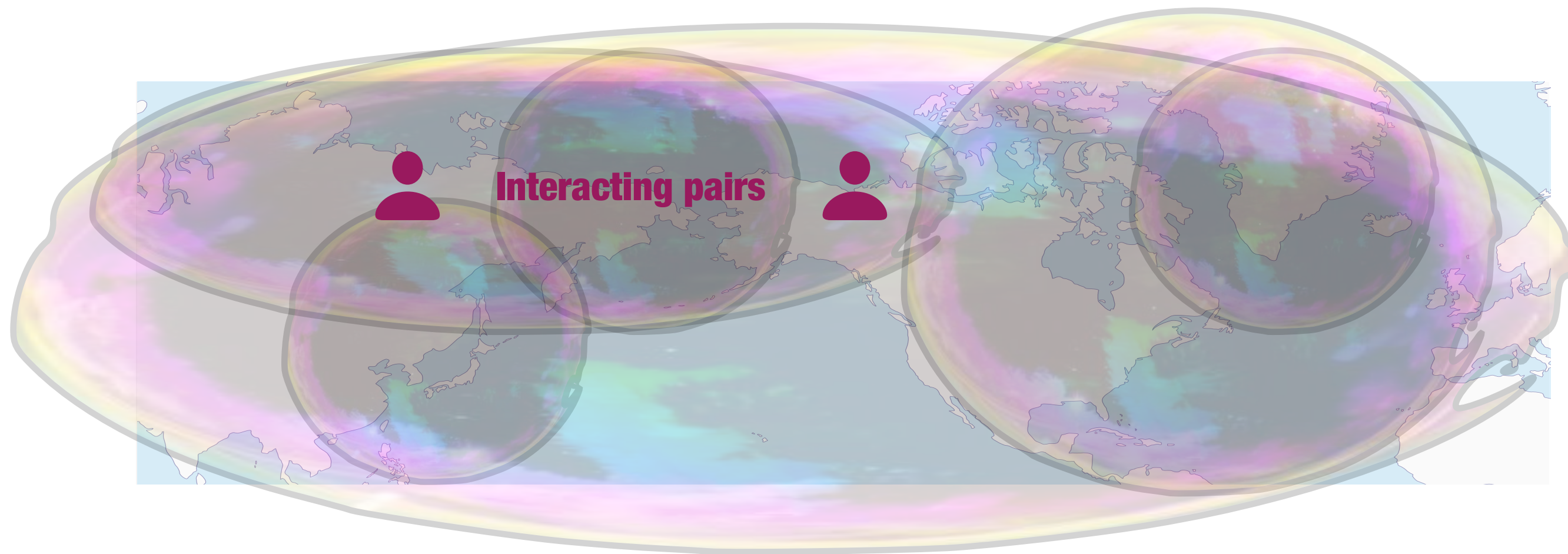
Nyle roadmap



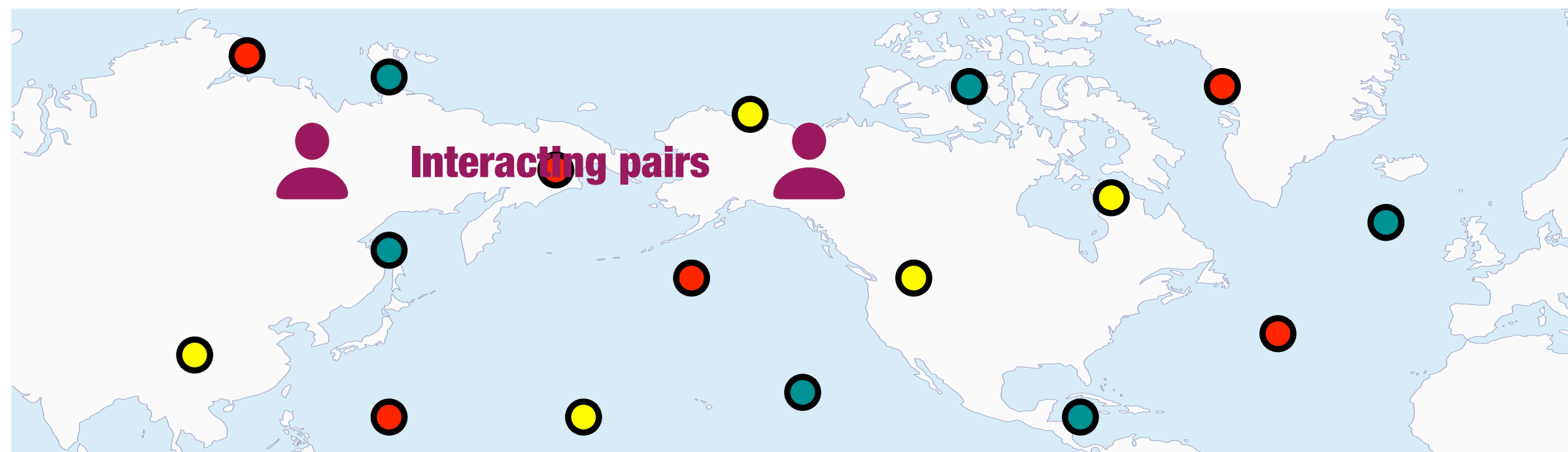
- Goal: exposure-limiting distributed ledger
- A trust-but-verify design for transaction validation
- Secure zoning with Byzantine validators
- Challenges: clearance zone
- **Simulation**

Simulation setup

- Do far away failures affect transactions?



- Nyle latency-based autozoning

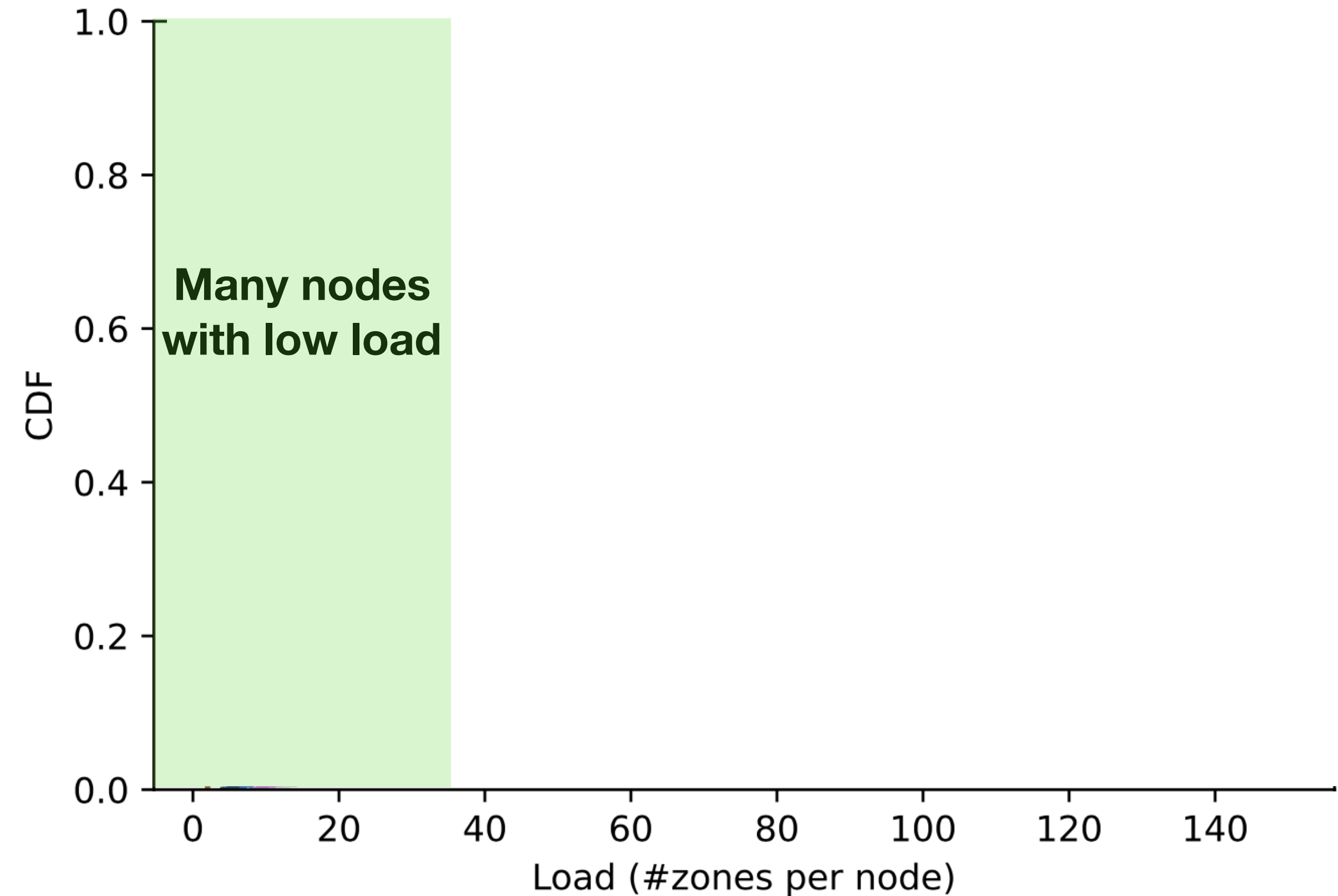
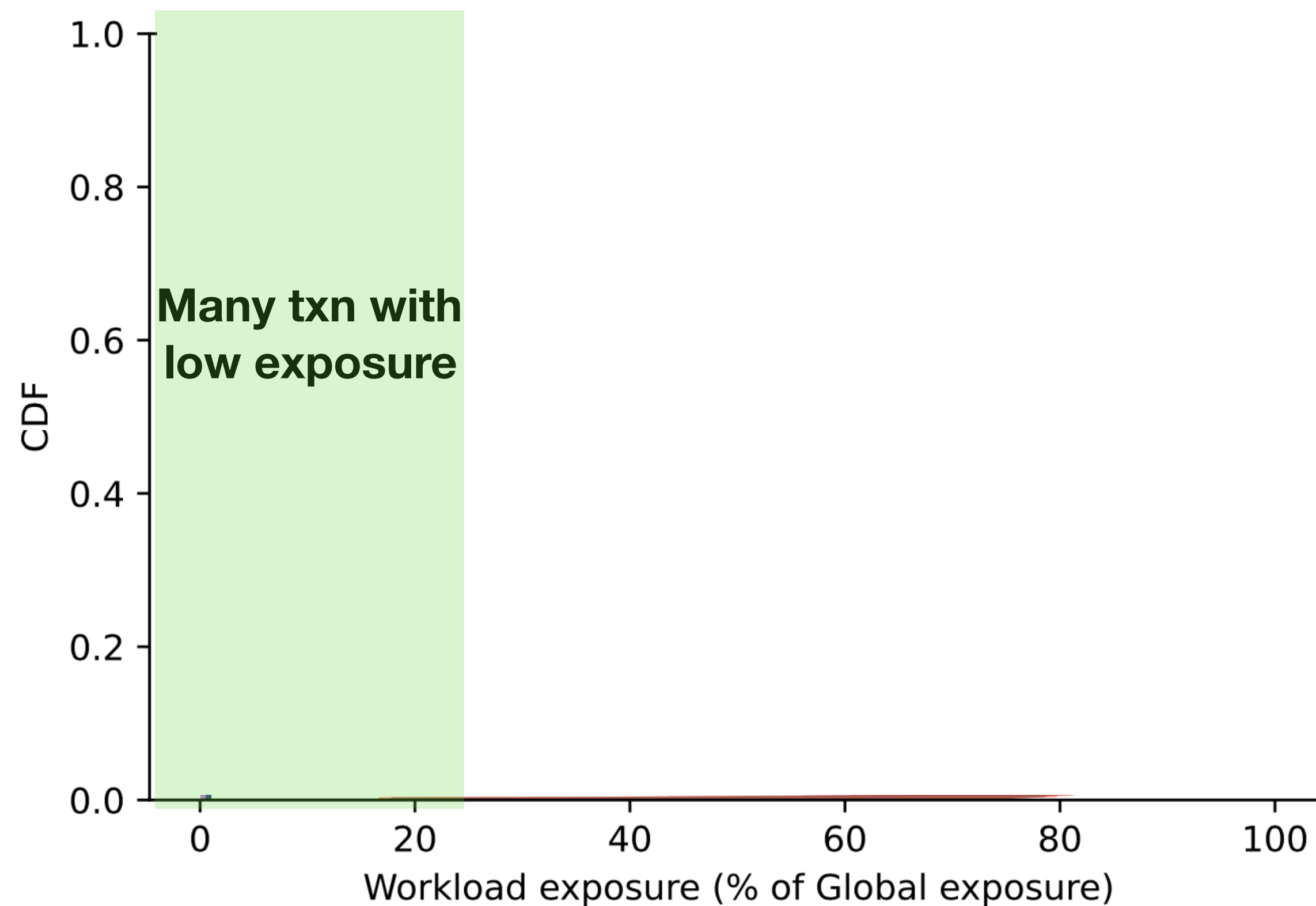


- Omniledger*

* Eleftherios Kokoris-Kogias, Philipp Jovanovic, Linus Gasser, Nicolas Gailly, Ewa Syta, and Bryan Ford. OmniLedger: A Secure, Scale-Out, Decentralized Ledger via Sharding. In *39th IEEE Symposium on Security and Privacy (SP)*, 2018.

Simulation results

- Workload: 10000 random pairs with RTTs from a realistic trace-based distribution

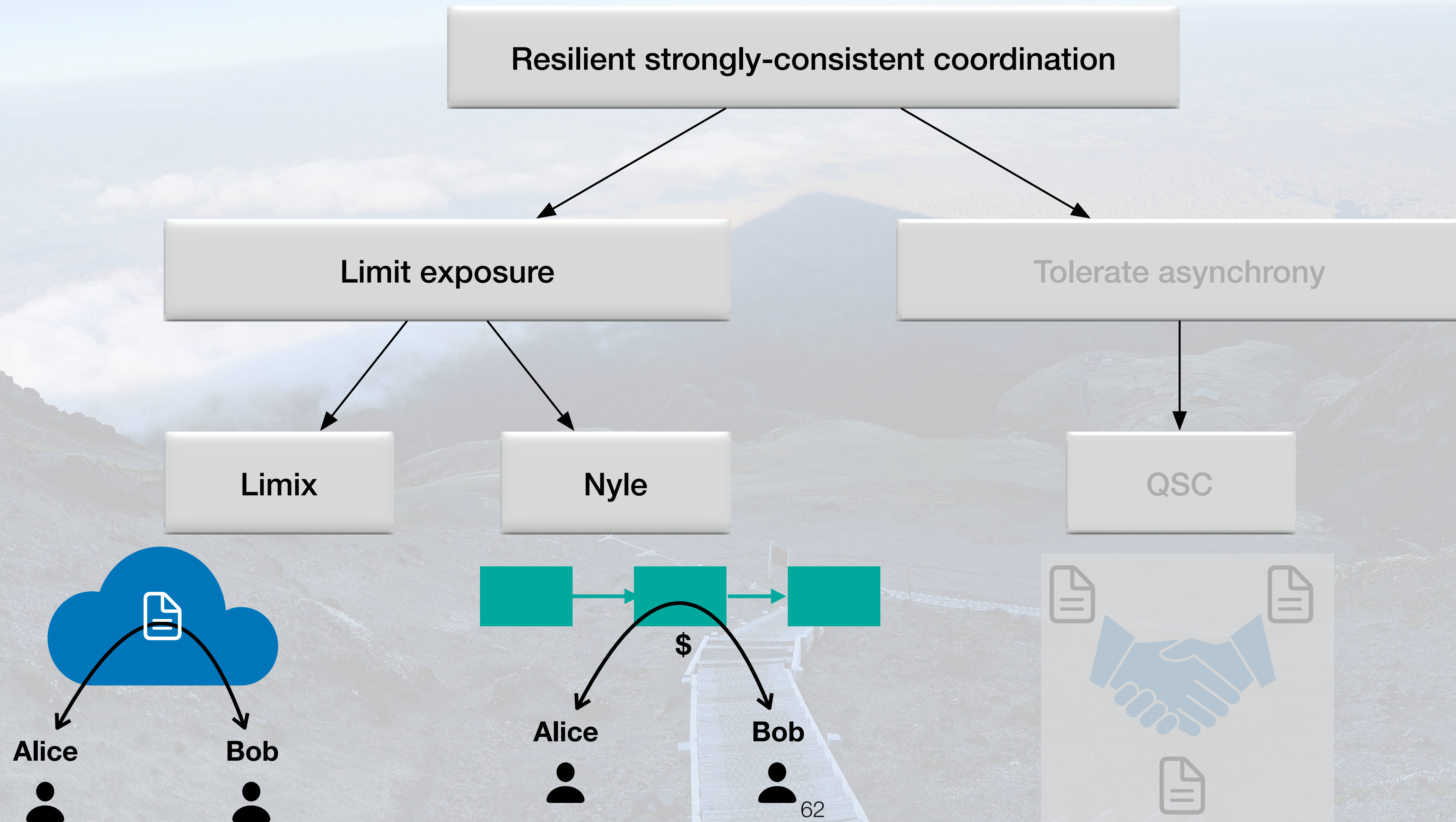


Lower exposure is better, because it guarantees higher availability

Nyle summary

- **Exposure-limiting architecture for BFT-style distributed ledgers**
 - Global zone is a secure finaliser, whereas local zones are opportunistic finalisers
- **Per-transaction user-controlled exposure**
- **Nyle significantly reduces exposure compared to Omniledger, at a 10x cost**

Thesis roadmap

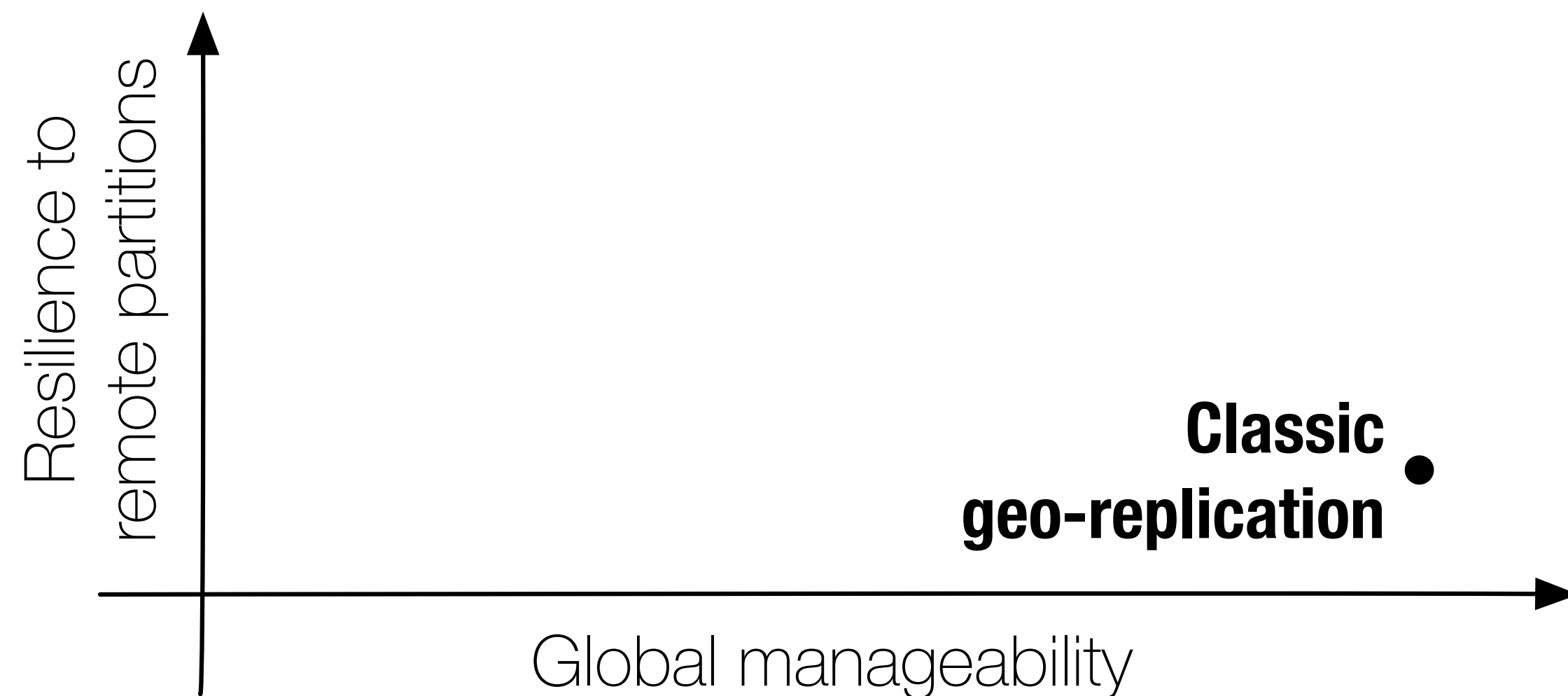


Conclusion

- **Resilience in strongly consistent coordination**
- **Limiting exposure results in systems resilient to distant failures**
 - Limix reduces exposure in metadata services for cloud systems
 - Nyle reduce exposure in BFT-style blockchains through a trust-but-verify approach
- **Tolerating network asynchrony could become more practical**
 - QSC solves asynchronous consensus for crash faults without requiring common coins

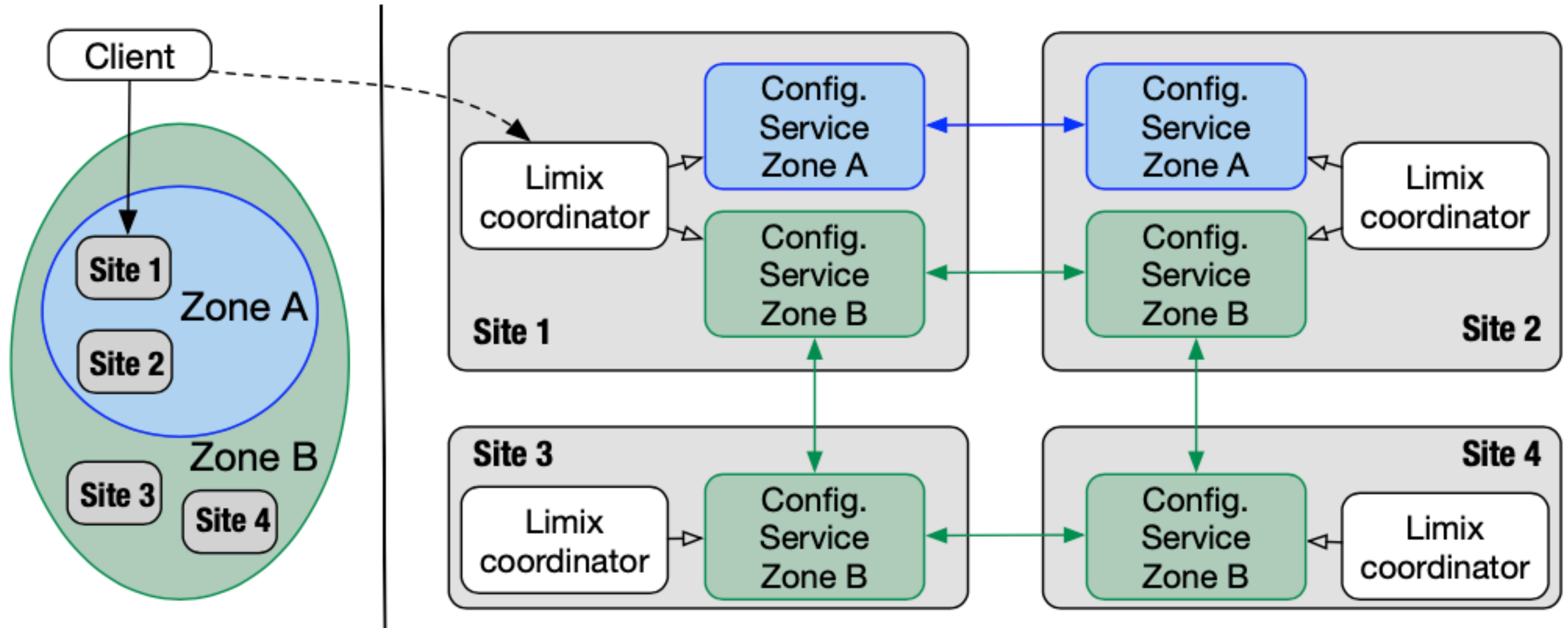
Backup slides

Limix: limit exposure to remote partitions



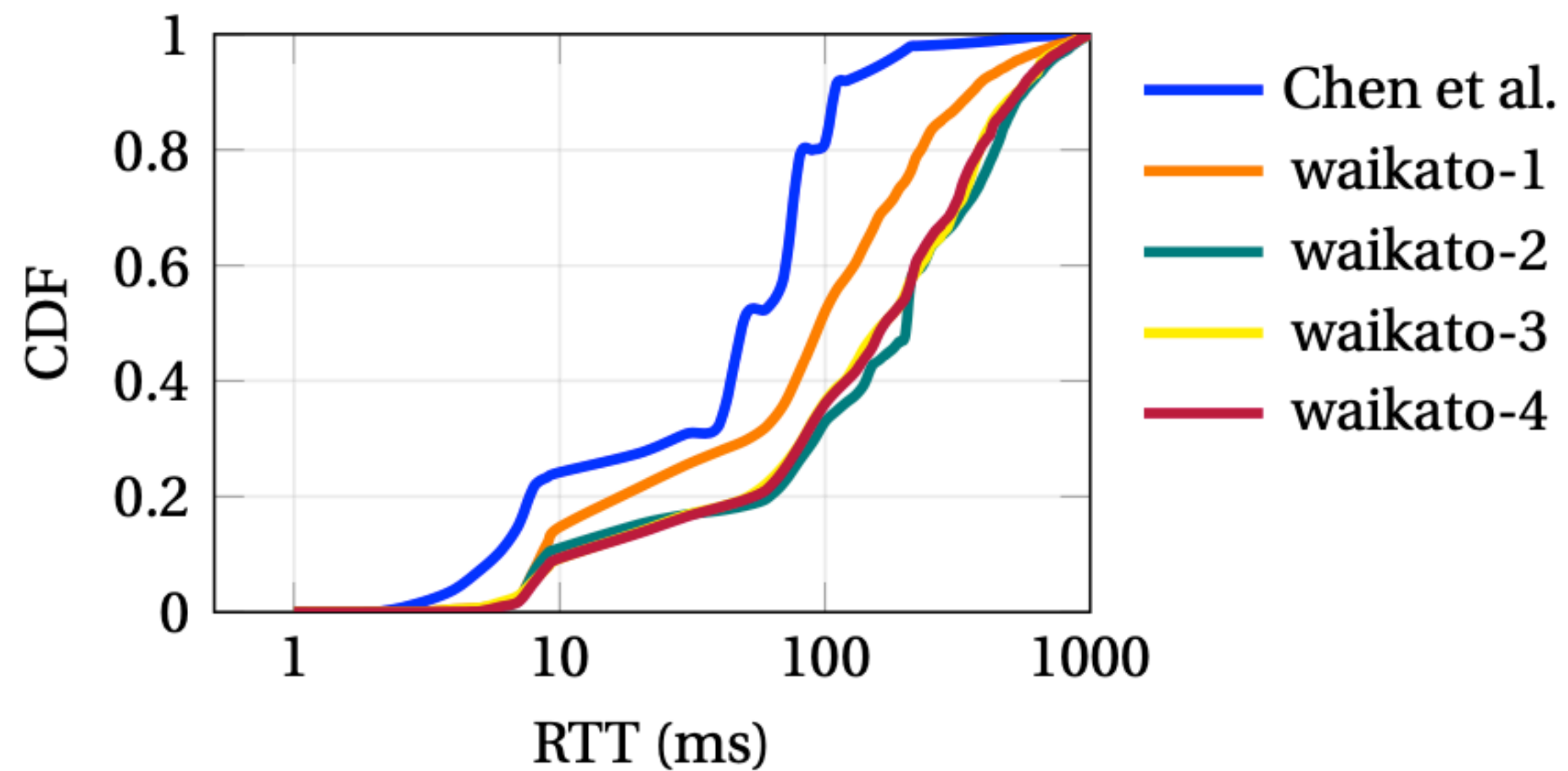
Goal: Limit metadata exposure for all users

Limix architecture

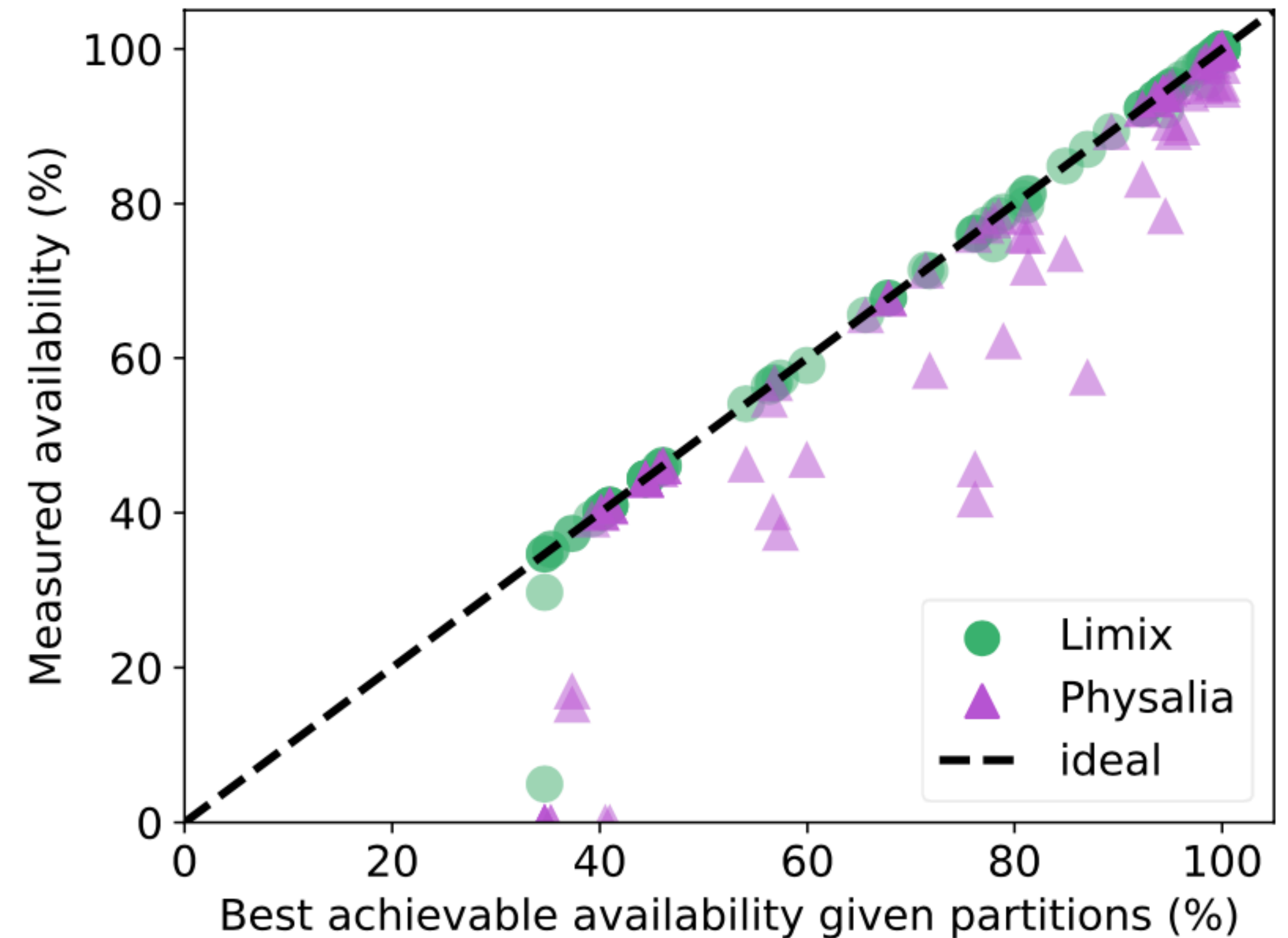


Limix realistic workload experiment

RTT between W-W pairs



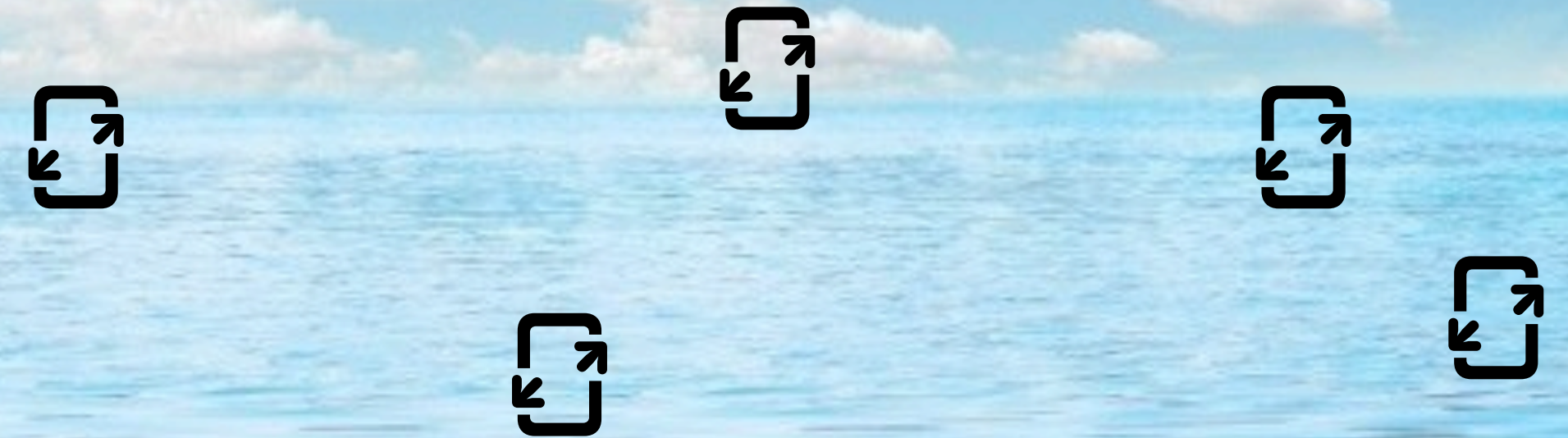
Availability



Dependencies across layers

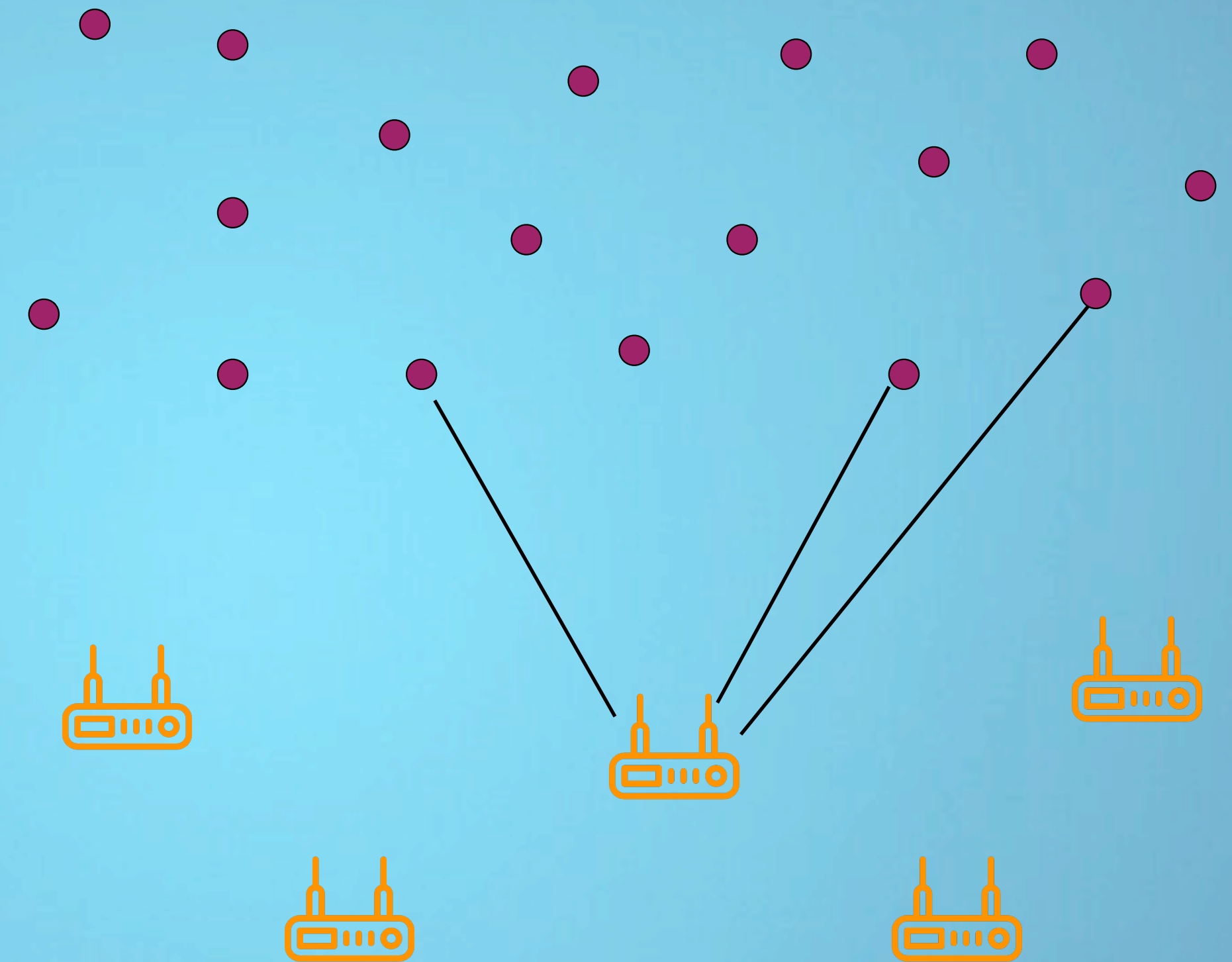
Application layer

Detect double spending, run consensus



Validators layer

Ensure validator uptime, withstand Byzantine behaviour



Interconnect layer

Correlated failures and attacks
Are validators up? From whose perspective?

Real-world resilience risks in blockchains

- Censorship / validators offline or slow \Rightarrow unavailability

Penalties

So far we have considered perfectly well-behaved validators, but what about validators that do not make timely head, source and target votes or do so slowly?

The penalties for missing the target and source votes are equal to the rewards the attester would have received had they submitted them. This means that instead of having the reward

SLASHING

Slashing is a more severe action that results in the forceful removal of a validator from the network and an associated loss of their staked ether. There are three ways a validator can be

Nyle zoning

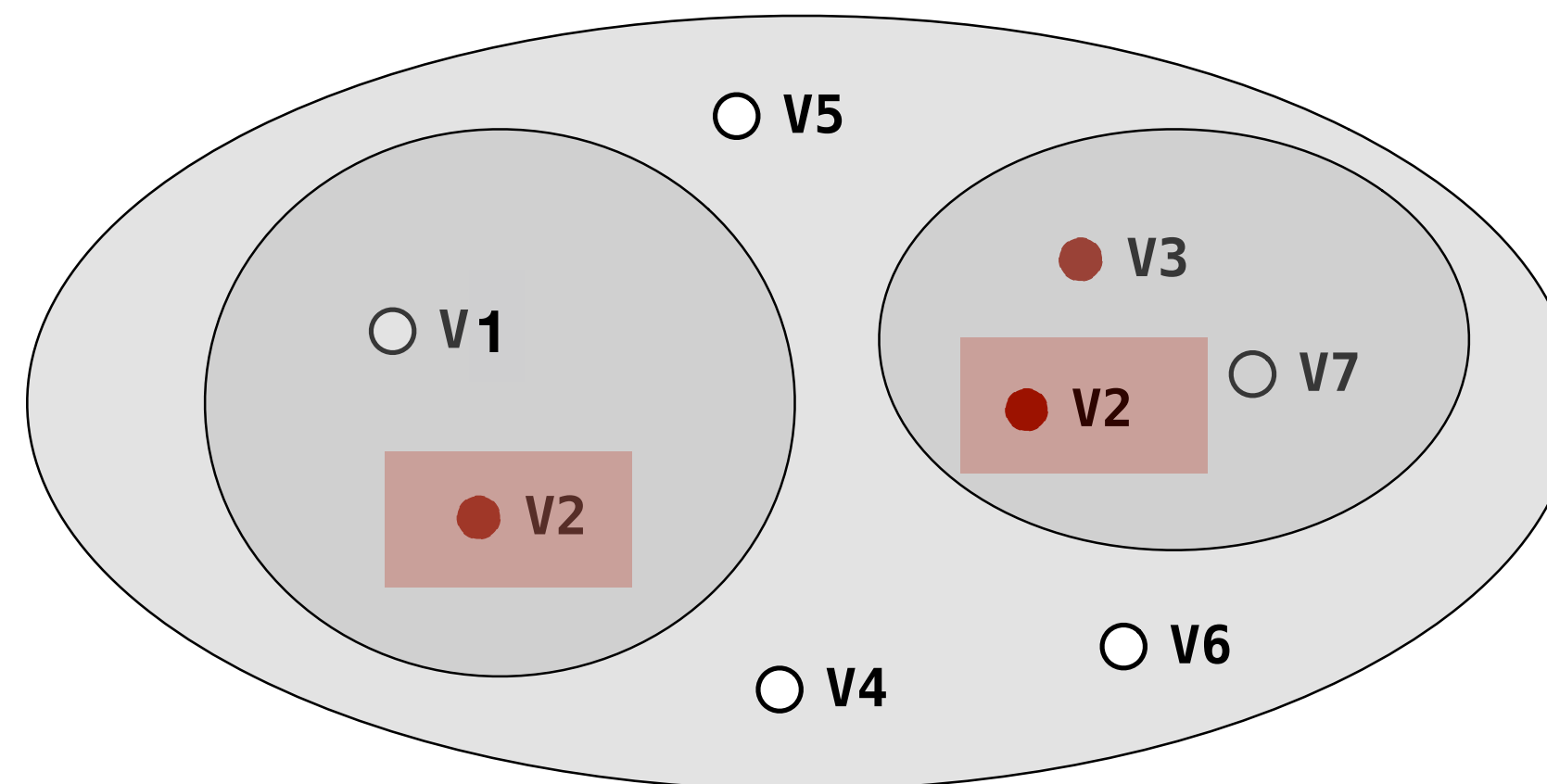
- ✓ **Challenge 1:**
Open membership resilient to Sybil attacks
(globally $< \frac{1}{3}$ Byzantine validators)

Solution:

Use existing* proof-of-work approaches

- ✗ **Challenge 2:**
Validator joins two disjoint zones

Global membership ($< \frac{1}{3}$ Byzantine)



* Eleftherios Kokoris-Kogias, Philipp Jovanovic, Nicolas Gailly, Ismail Khoffi, Linus Gasser, and Bryan Ford. Enhancing Bitcoin Security and Performance with Strong Consistency via Collective Signing. In *Proceedings of the 25th USENIX Conference on Security Symposium*, 2016.

Nyle zoning

- ✓ **Challenge 1:**
Open membership resilient to Sybil attacks
(globally $< \frac{1}{3}$ Byzantine validators)

Solution:

Use existing proof-of-work approaches

- ✓ **Challenge 2:**
Validator joins two disjoint zones

Solution:

Validators make a **location claim**

Global membership ($< \frac{1}{3}$ Byzantine)

○ V1: PubK₁, Auth_{loc1}
● V2: PubK₂, Auth_{loc2}
● V3: PubK₃, Auth_{loc3}
⋮
○ Vn: PubK_n, Auth_{locn}

Zoning alg
(exposure metrics)

Zone₁ membership (Byzantine can be $\geq \frac{1}{3}$)

○ V1 ● V3 ...

Zone_k membership (Byzantine can be $\geq \frac{1}{3}$)

○ V1 ● V2 ● V7 ...

Nyle zoning

- ✓ **Challenge 1:**
Open membership resilient to Sybil attacks
(globally $< \frac{1}{3}$ Byzantine validators)

Solution:

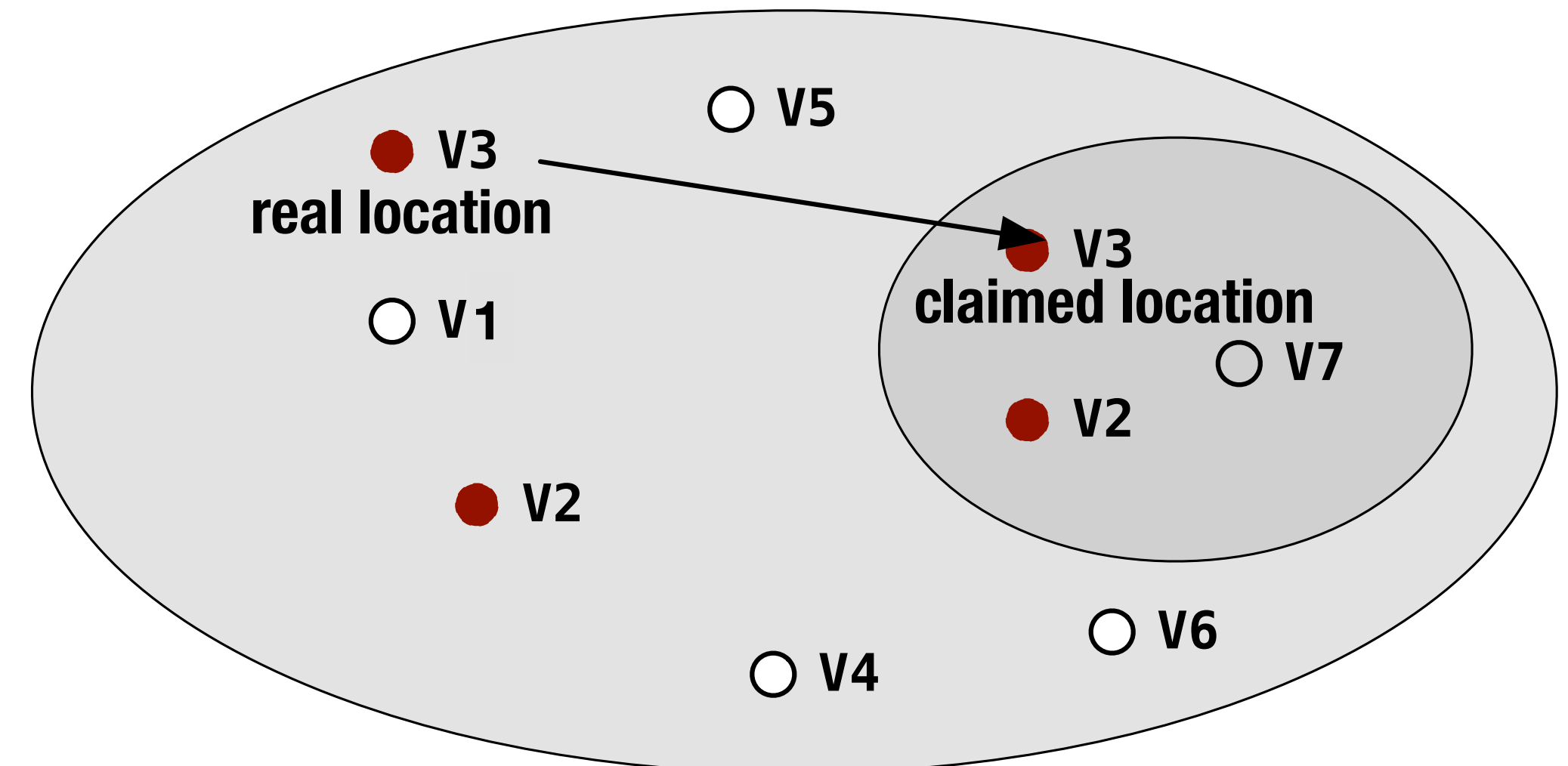
Use existing proof-of-work approaches

- ✓ **Challenge 2:**
Validator joins two disjoint zones

Solution:

Validators make a **location claim**

- ✗ **Challenge 3:**
Compromised validator makes bogus location claim



Nyle zoning

- ✓ **Challenge 1:**
Open membership resilient to Sybil attacks
(globally $< \frac{1}{3}$ Byzantine validators)

Solution:

Use existing proof-of-work approaches

- ✓ **Challenge 2:**
Validator joins two disjoint zones

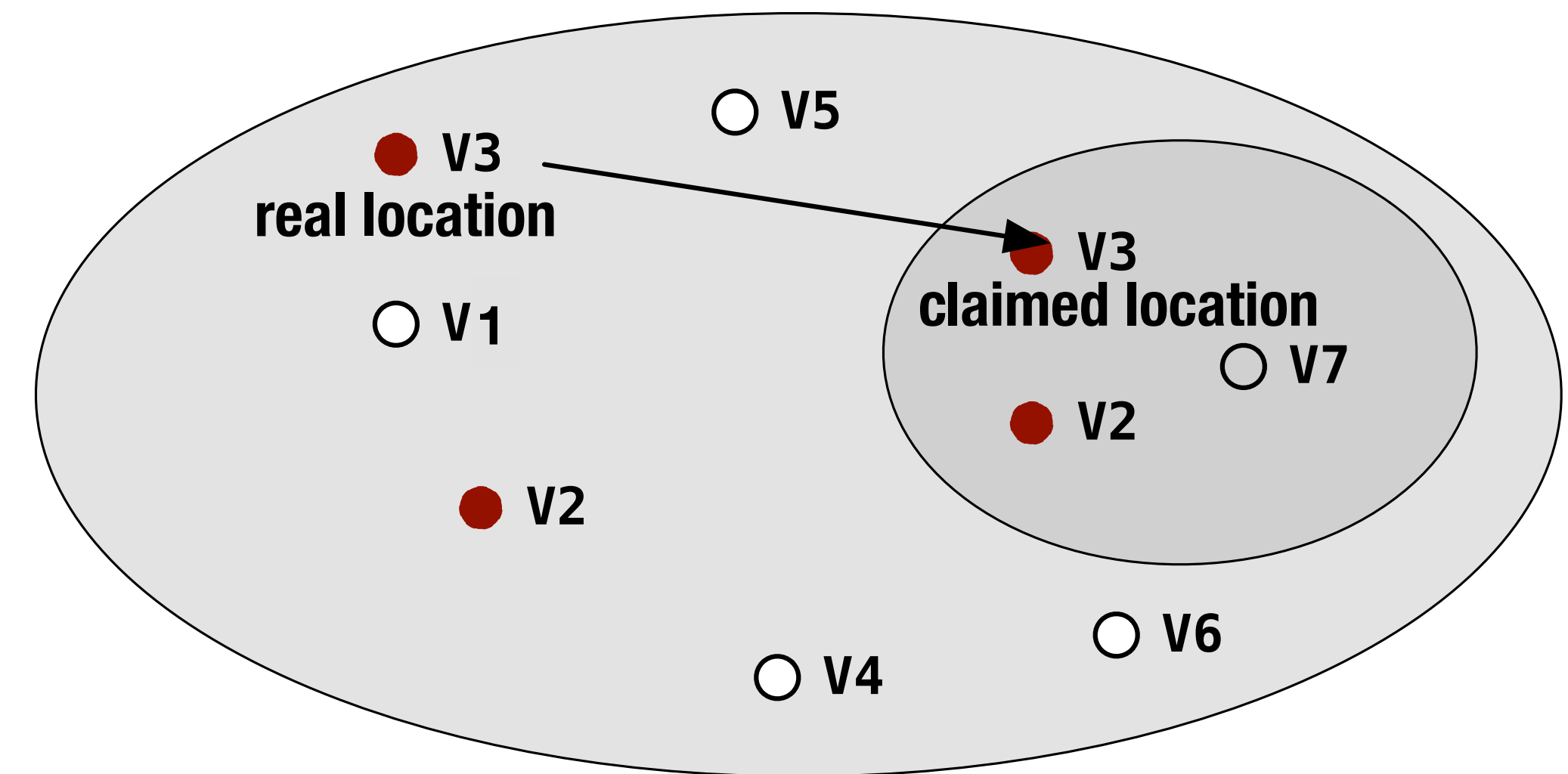
Solution:

Validators make a **location claim**

- ✓ **Challenge 3:**
Malicious validator makes wrong location claim

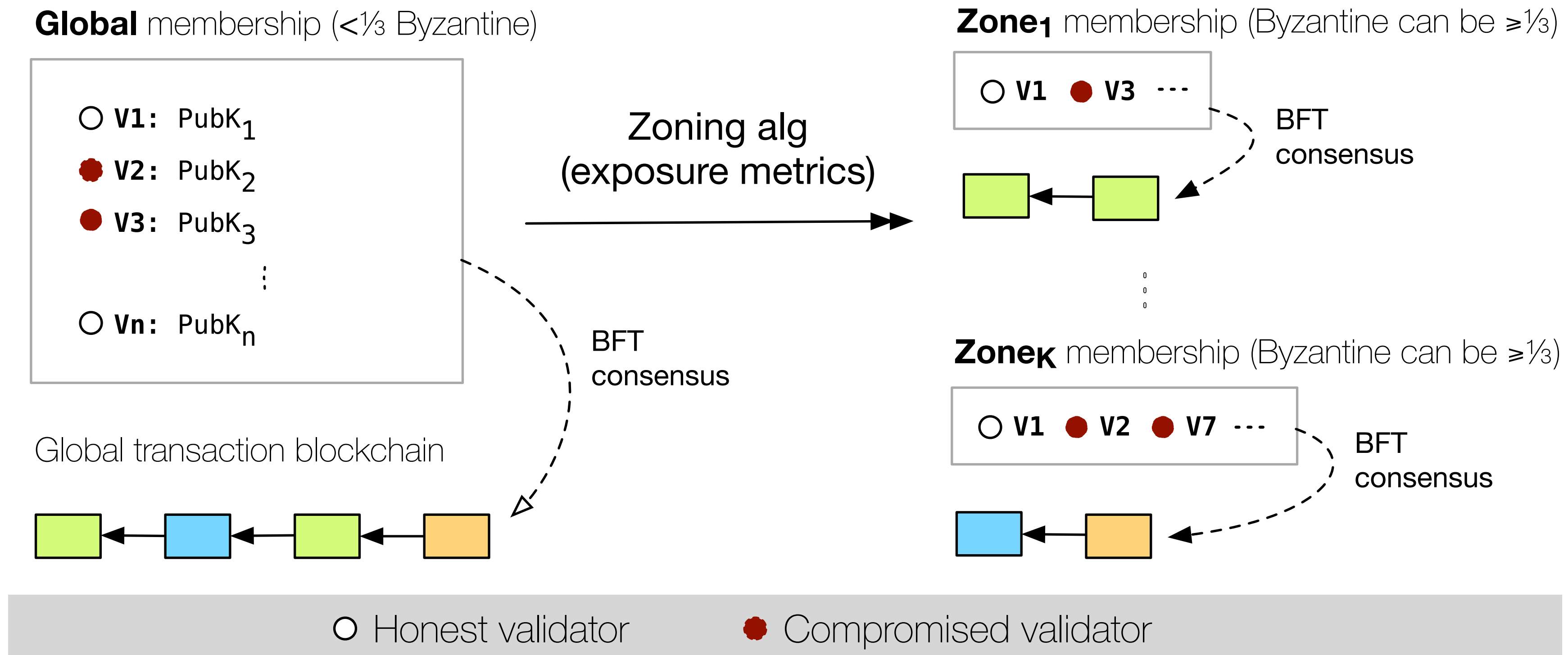
Solution:

Validate location claims via existing mechanisms*



* Katharina Kohls and Claudia Diaz. VerLoc: Verifiable localization in decentralized systems. In *USENIX Security Symposium (USENIX Security)*, 2022

Nyle zoning



Nyle transaction validation security

#	Participant honest(✓) / compromised(×) / unknown(*)				Nyle property guarantee yes(✓) / no(×) / N/A [reason]						
	Sender	Recv	CLZ	SEZ	Safety			Liveness*			Exposure
					Sender	Recv.	Zones	Sender	Recv.	Zones	
1.	✓	*	✓	✓	✓	✓	✓	✓	✓	✓	SEZ
2.				×		✓ [S2]		✓ [L1]			Global
3.	✓	*	×	✓	✓ [S1]	✓	✓	✓	✓ [L2]	✓ [L2]	SEZ
4.				×		✓ [S2]		✓ [L1,L2]			Global
5.	×	*	✓	✓	N/A	✓	✓	✓	✓	✓	SEZ
6.				×		✓ [S4]		✓ [L1]			Global
7.	×	*	×	✓	N/A	✓ [S3]	✓ [S3]	✓	✓ [L1,L3]	✓ [L1,L3]	SEZ
8.				×		✓ [S3]		✓ [L1]			Global

Reason ID	Description
S1	The sender can forward an already validated transaction to cancel CLZ history rewrite.
S2	The transaction should be accepted, the receiver detects the bogus reject of a compromised SEZ.
S3	Zones and receiver notified of double spending by the Global zone.
S4	The transaction must not be accepted, the receiver detects the bogus accept of a compromised SEZ.
L1	The Global zone eventually replies with an authentic validation.
L2	The sender replaces a censoring CLZ via a CLZ transfer (Section 3.6).
L3	The zones contact the Global zone to intervene if the CLZ is unresponsive.

Binary consensus using common core

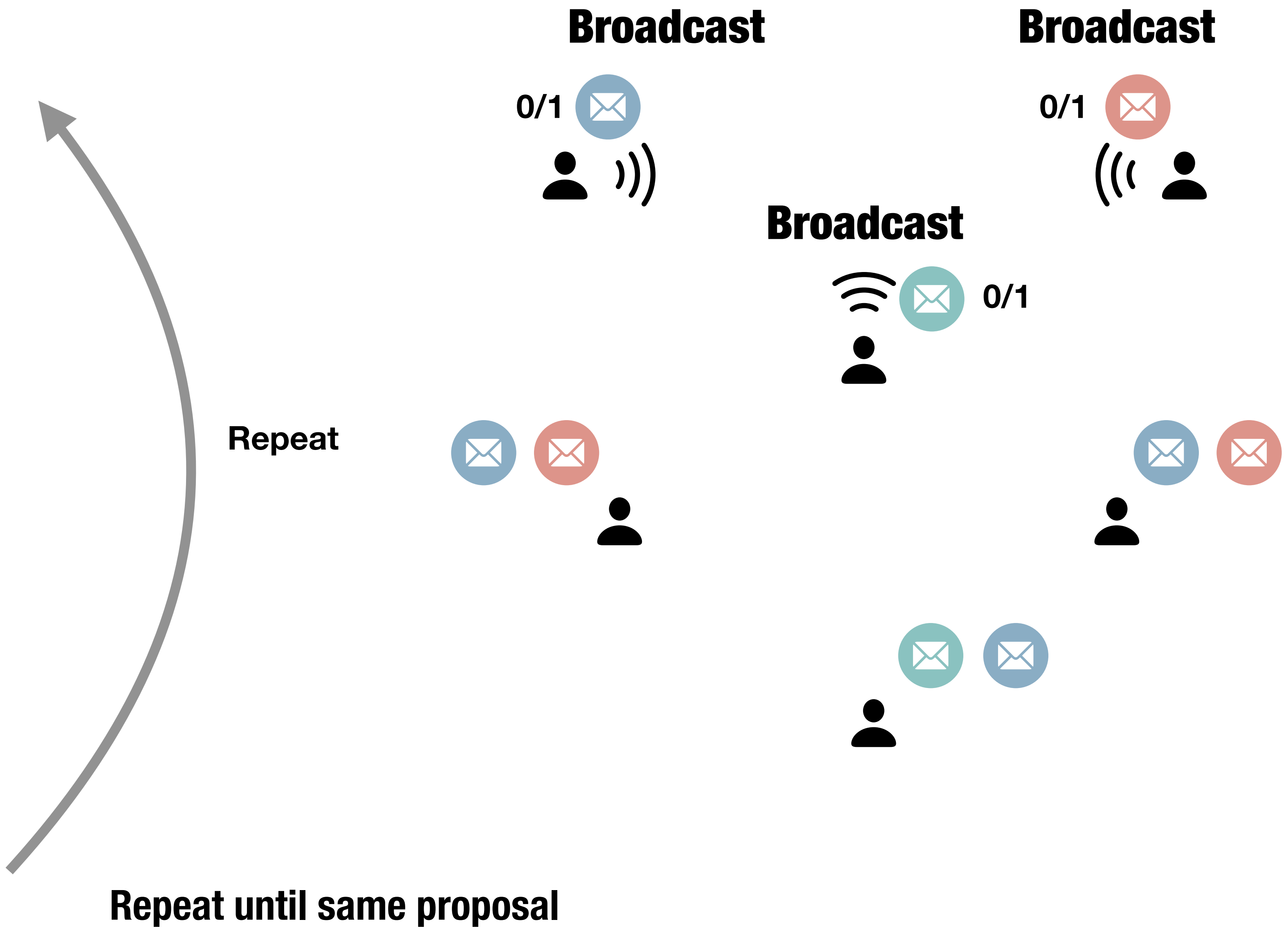
Exchange of information

Voting

Wait for N-f

Can I decide?
If not, how do I continue?

Observe a majority vote?



Consensus using reliable broadcast

Agreement on arbitrary values

Cannot use majority vote

Use reliable broadcast

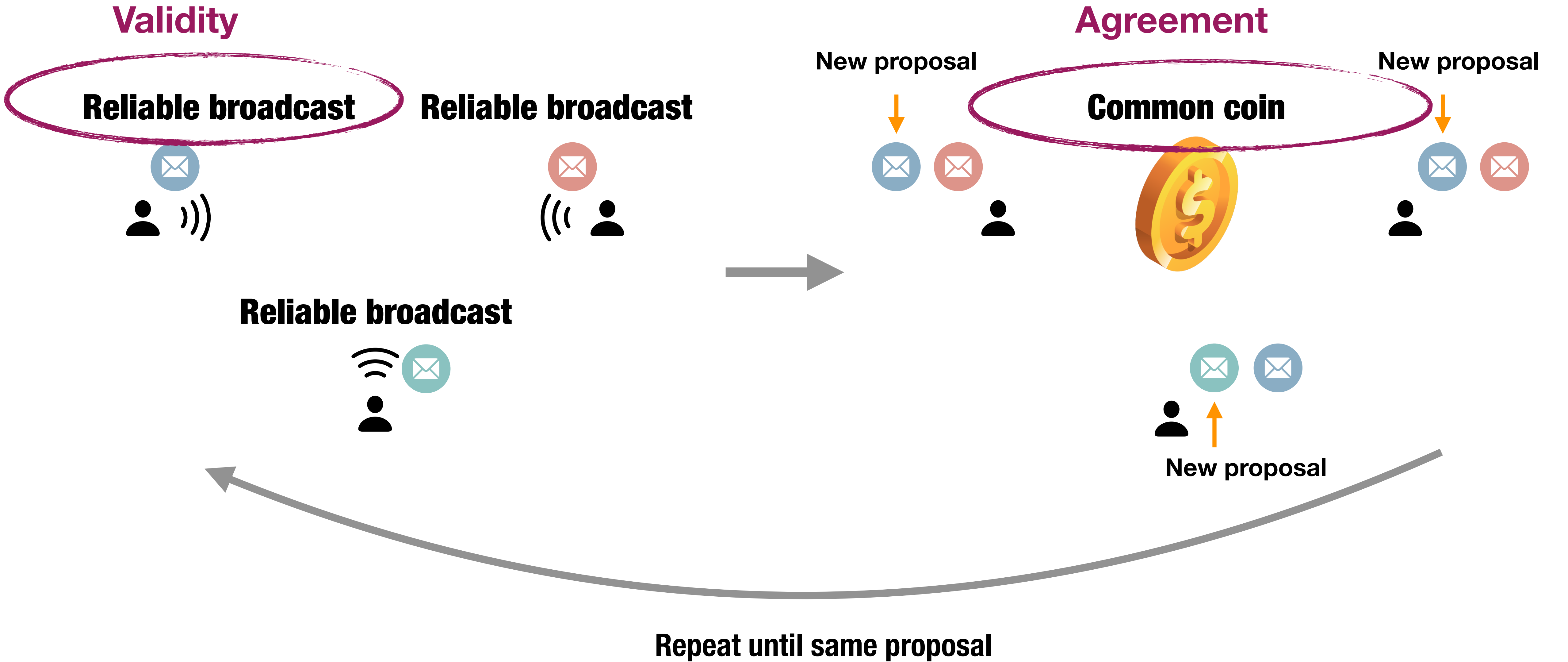
Eventually all live nodes receive the same proposals. But when?

Different signals possible

A multi-valued common coin

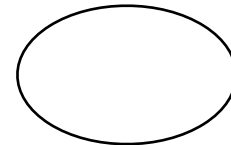
N blocks of binary consensus

Consensus with common coins and reliable broadcast



Validity

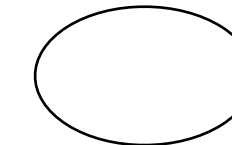
Alice



What values are there?

Agreement

Alice



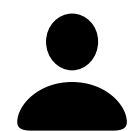
Do all others know and can
decide the same value?

QSC insights

Validity

What values are there?

Alice



Use a novel broadcast primitive (FSTSB)

Helper property:
Bound uncertainty

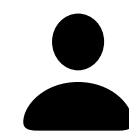
Alice



Agreement

Do all others know and can
decide the same value?

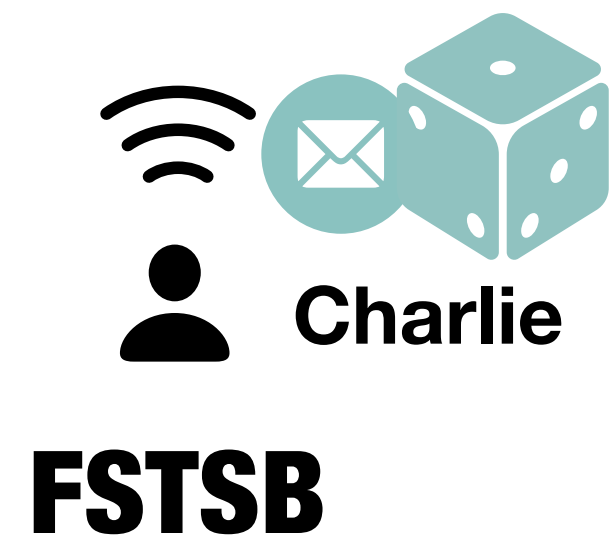
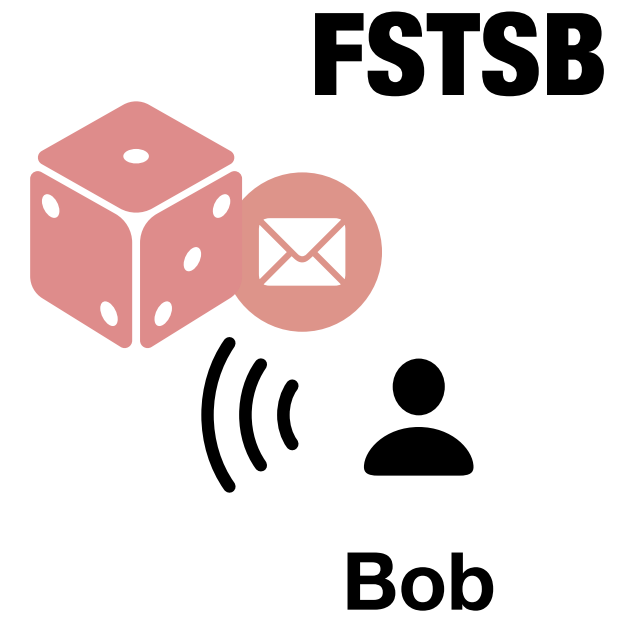
Alice



Private randomness together with values

What do the others know?

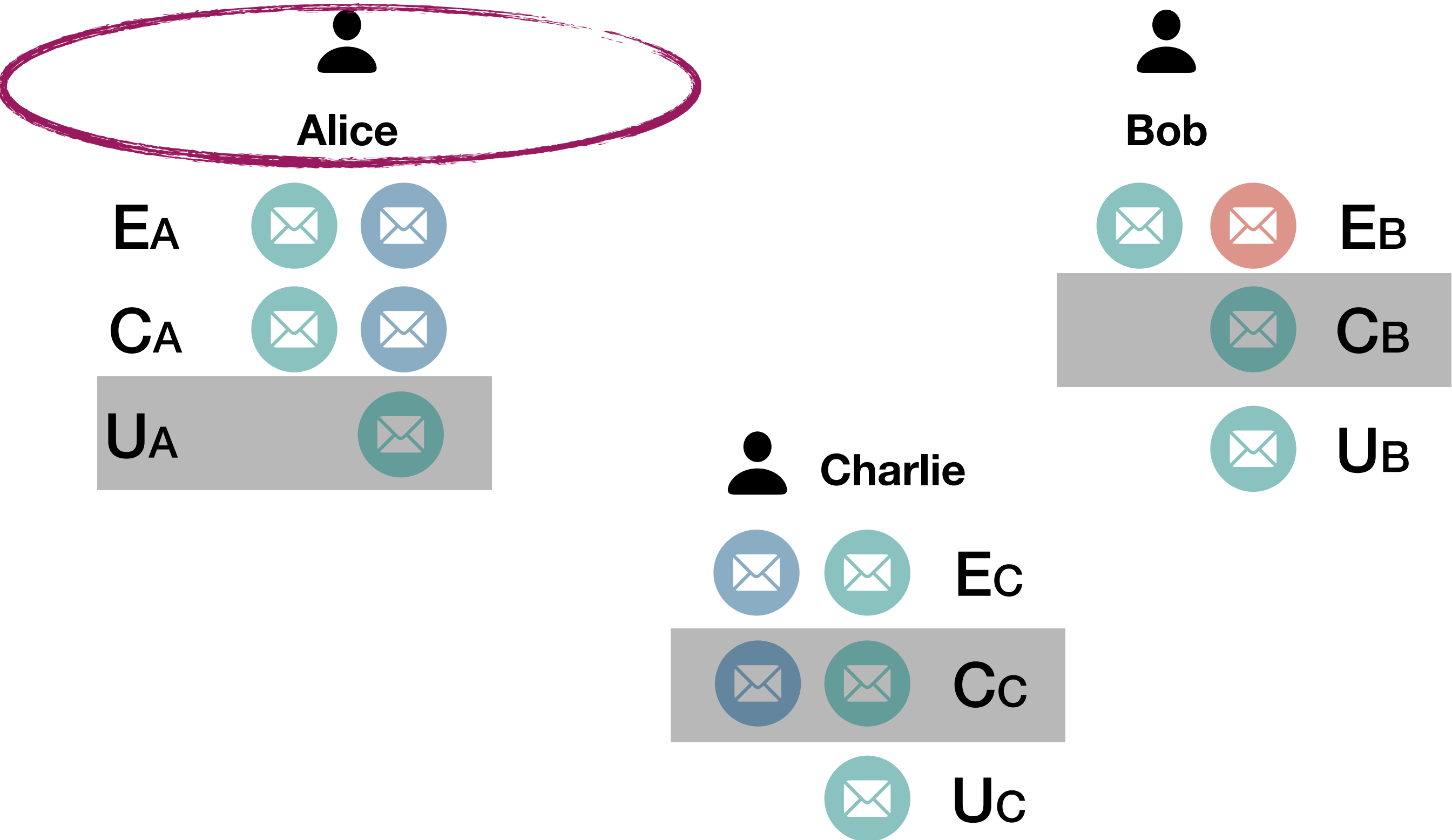
Validity



- No other proposal is ever introduced
- Though random values change

Bounding uncertainty by approximating others' knowledge

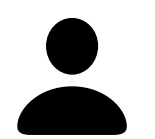
FSTSB \Rightarrow sets E,C,U



C any-other \supseteq **U** Alice

Bounding uncertainty by approximating others' knowledge

FSTSB \Rightarrow sets E,C,U



Bob



EB



UB



Charlie



EC



UC



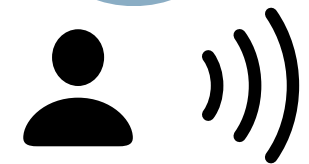
E Alice \supseteq **C** any-other \supseteq **U** Alice



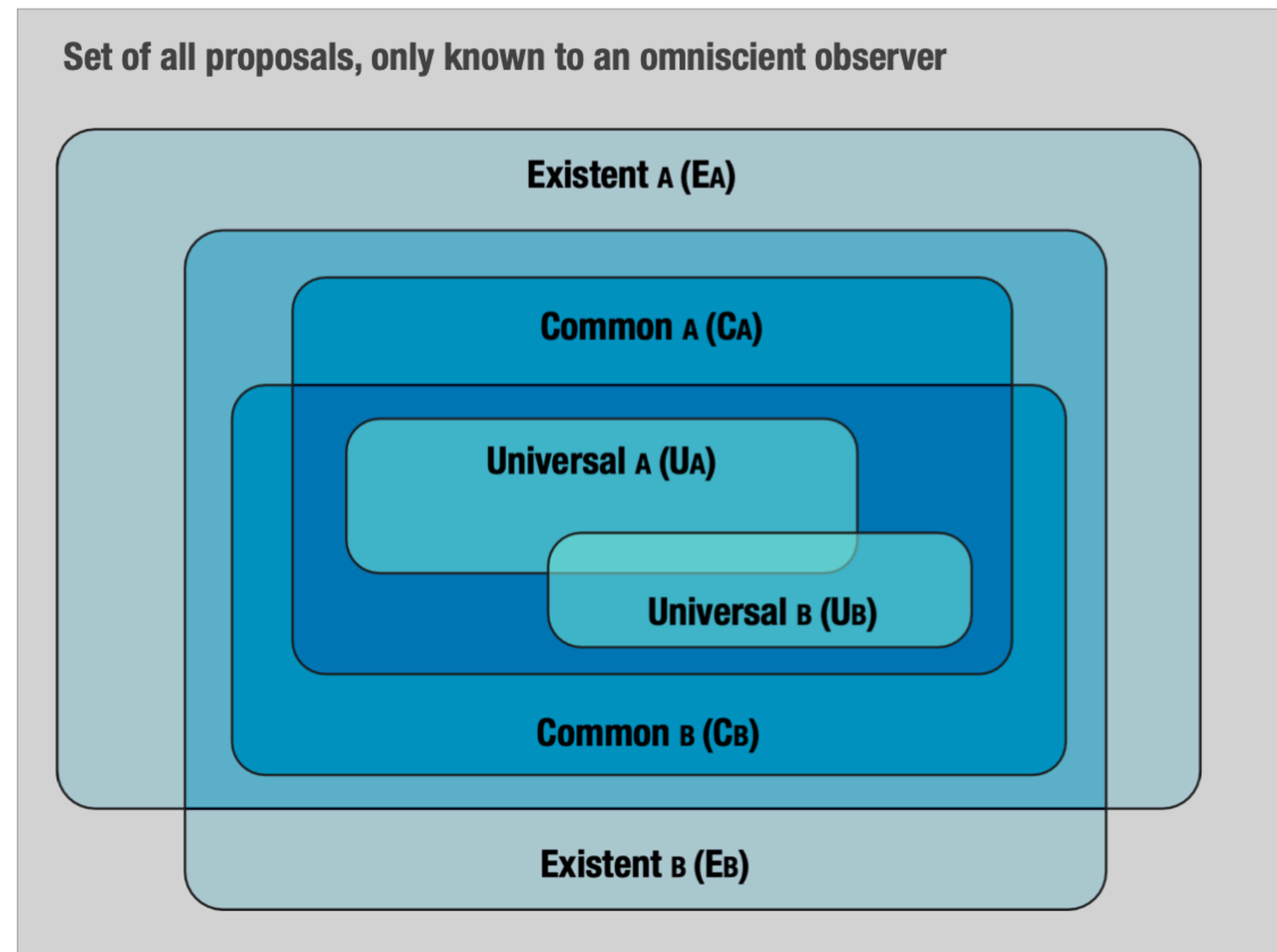
Bounding uncertainty by approximating others' knowledge

FSTSB \Rightarrow sets E,C,U

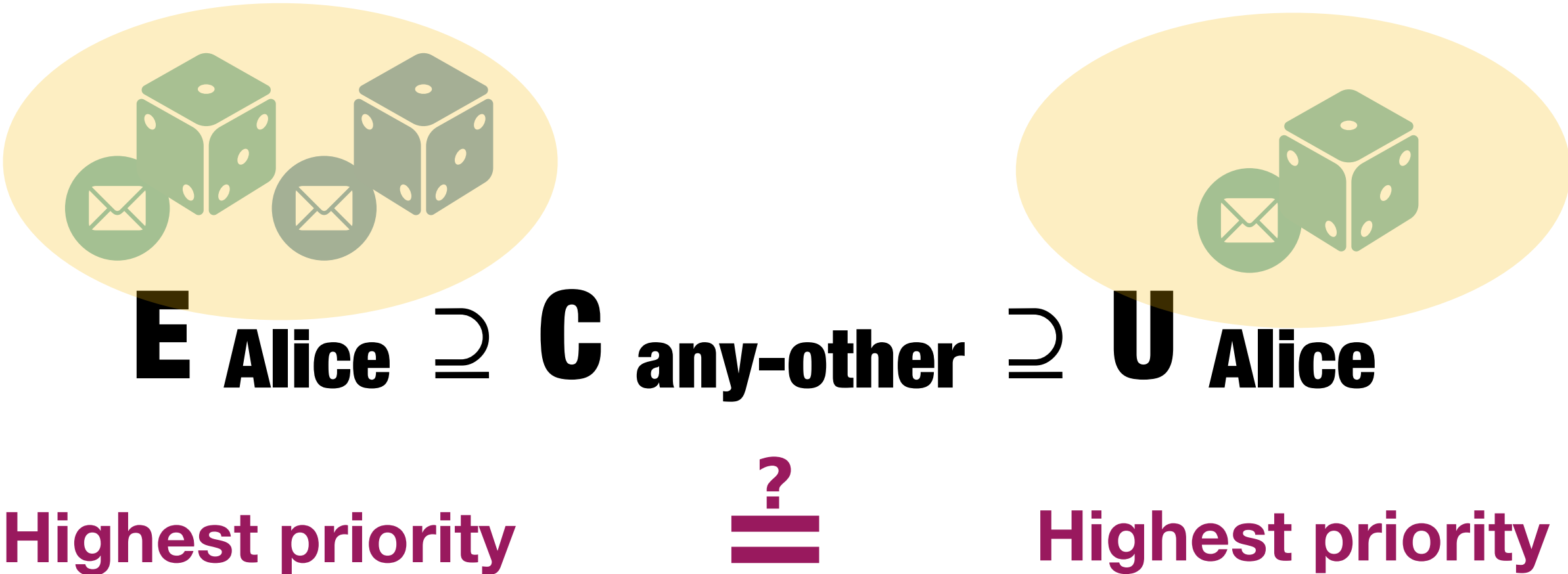
$$\mathbf{E}_{\text{Alice}} \supseteq \mathbf{C}_{\text{any-other}} \supseteq \mathbf{U}_{\text{Alice}}$$



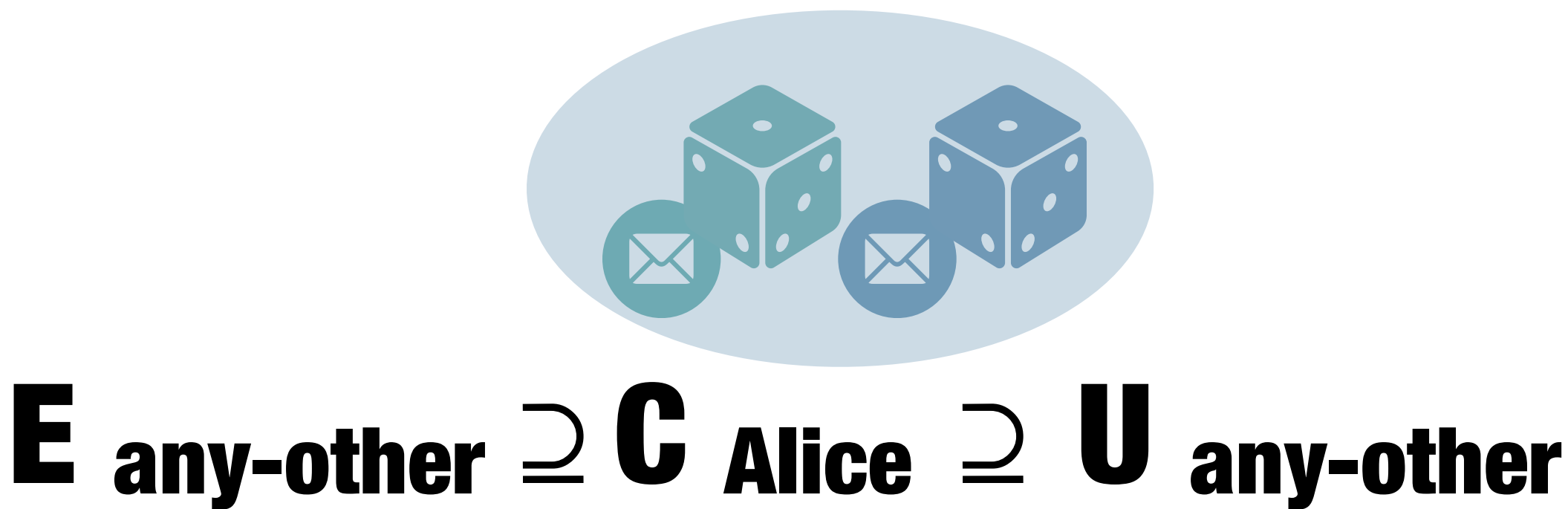
Alice



Agreement



Y: Decide that value



N: Continue with highest priority

Agreement

Continue with highest priority

