Theoretical and practical worlds of failure detectors

Lena Hall

Principal Technologist, Microsoft

Twitter: @lenadroid

Lena Hall

Microsoft Azure

Engineering

- ✓ Architecture
- ✓ Cloud
- ✓ Data
- ✓ ML/AI

Questions? -> alehall [at] microsoft [dot] com

lenadroid

Acknowledgements

Natallia Dzenisenka for thoughtful collaboration on the material for theoretical failure detection.

Matthew Snider for reviewing the material and in-depth conversations around Service Fabric and its internals.

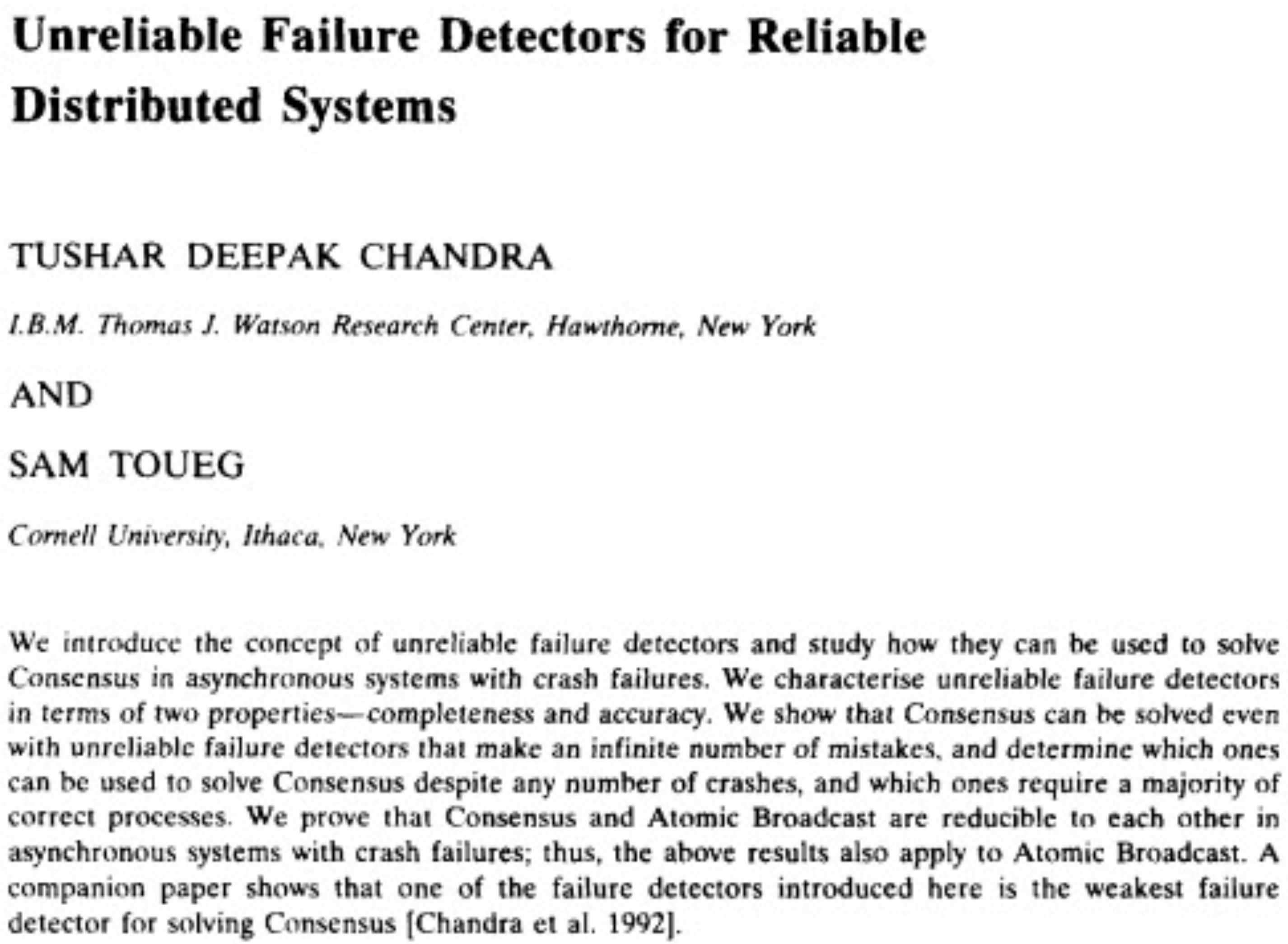
Mark Brown, Andrew Liu, Aleksey Charapko, Eddie Ailijiang for reviewing the material and answering a dozen of my questions about the core of Azure Cosmos DB.

Clemens Vasters for valuable insights on messaging.

Introduction

Can we trust our systems to never fail?

Why should you care?



Unreliable Failure Detectors for Reliable Distributed Systems

TUSHAR DEEPAK CHANDRA

I.B.M. Thomas J. Watson Research Center, Hawthorne, New York

AND

SAM TOUEG

Cornell University, Ithaca, New York

We introduce the concept of unreliable failure detectors and study how they can be used to solve Consensus in asynchronous systems with crash failures. We characterise unreliable failure detectors in terms of two properties—completeness and accuracy. We show that Consensus can be solved even with unreliable failure detectors that make an infinite number of mistakes, and determine which ones can be used to solve Consensus despite any number of crashes, and which ones require a majority of correct processes. We prove that Consensus and Atomic Broadcast are reducible to each other in asynchronous systems with crash failures; thus, the above results also apply to Atomic Broadcast. A companion paper shows that one of the failure detectors introduced here is the weakest failure detector for solving Consensus [Chandra et al. 1992].

Failure Detectors

Applications of Failure Detectors

agreement problems

consensus

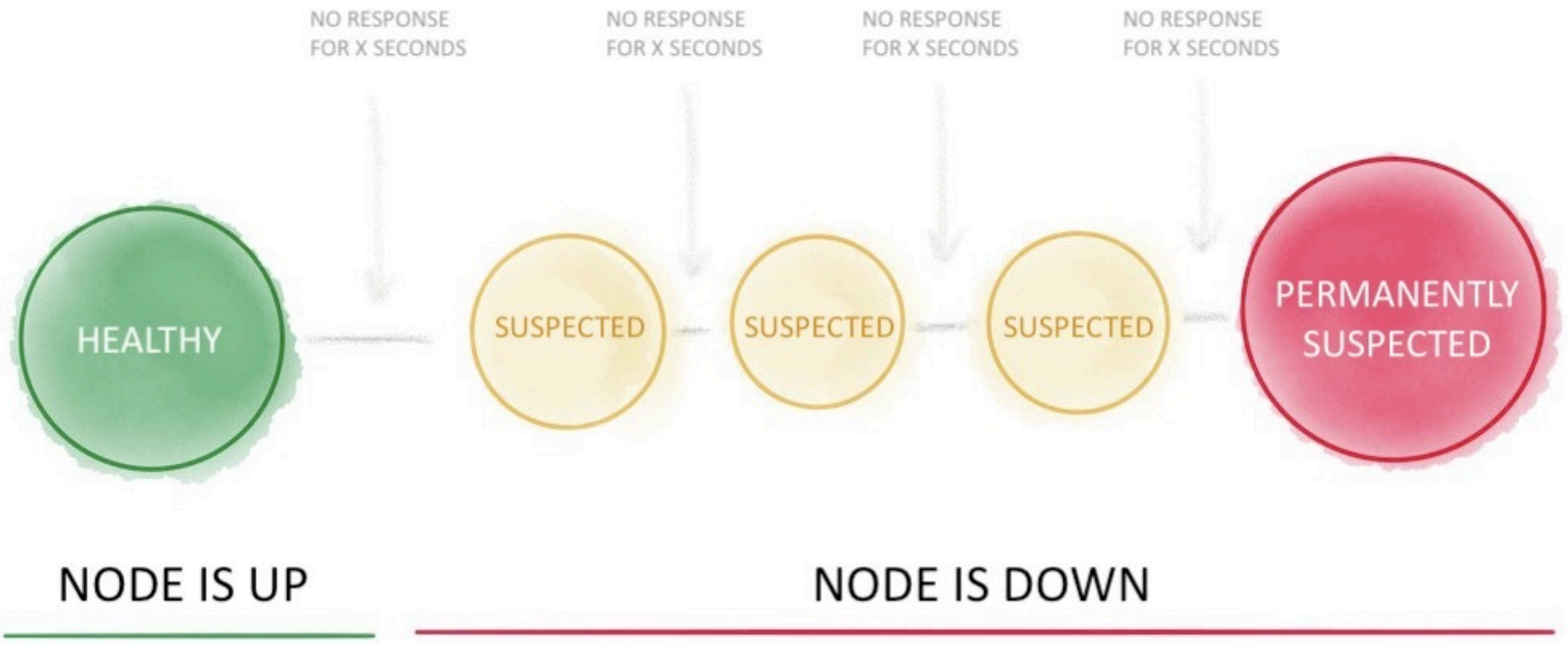
leader election

atomic broadcast

group membership

other distributed algorithms

Failure Suspicions



NO RESPONSE FOR X SECONDS

NO RESPONSE FOR X SECONDS

NO RESPONSE FOR X SECONDS

NO RESPONSE FOR X SECONDS

HEALTHY

SUSPECTED

SUSPECTED

SUSPECTED

PERMANENTLY SUSPECTED

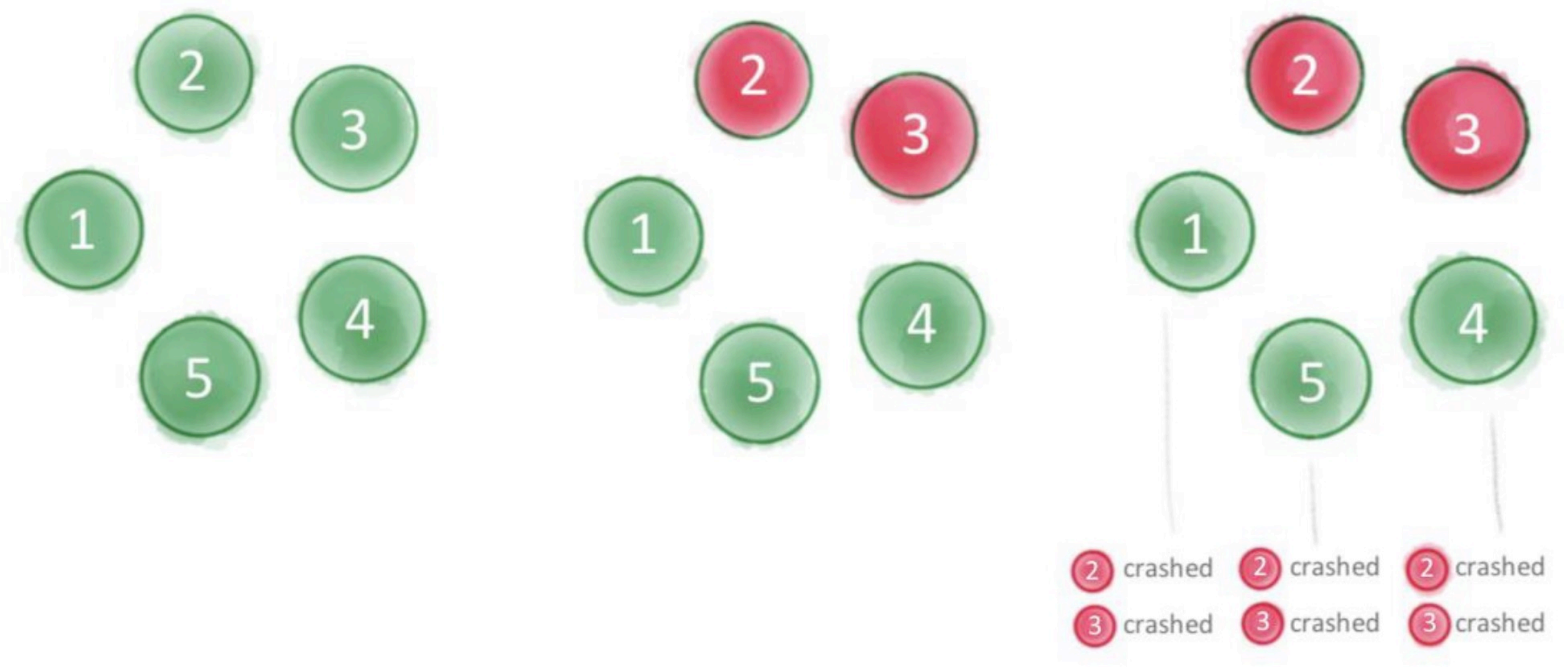
NODE IS UP

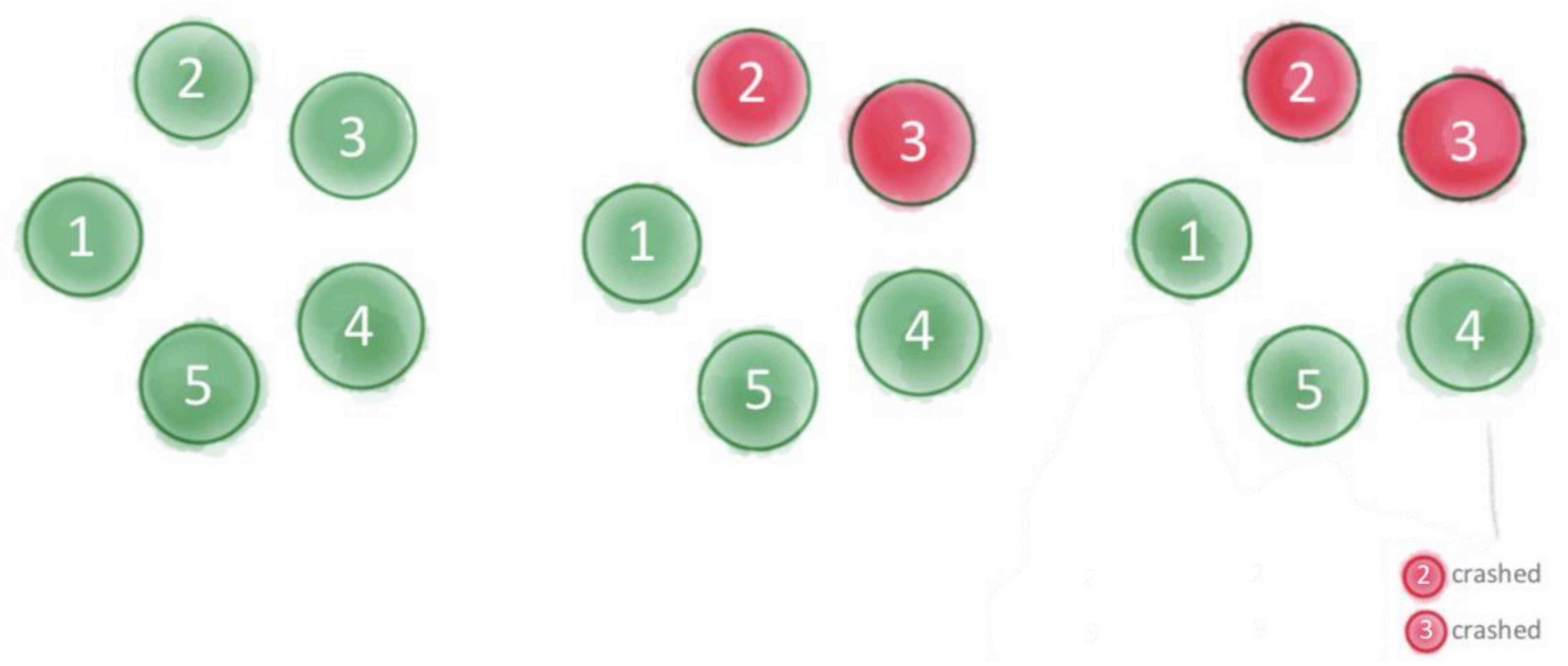
NODE IS DOWN

lenadroid

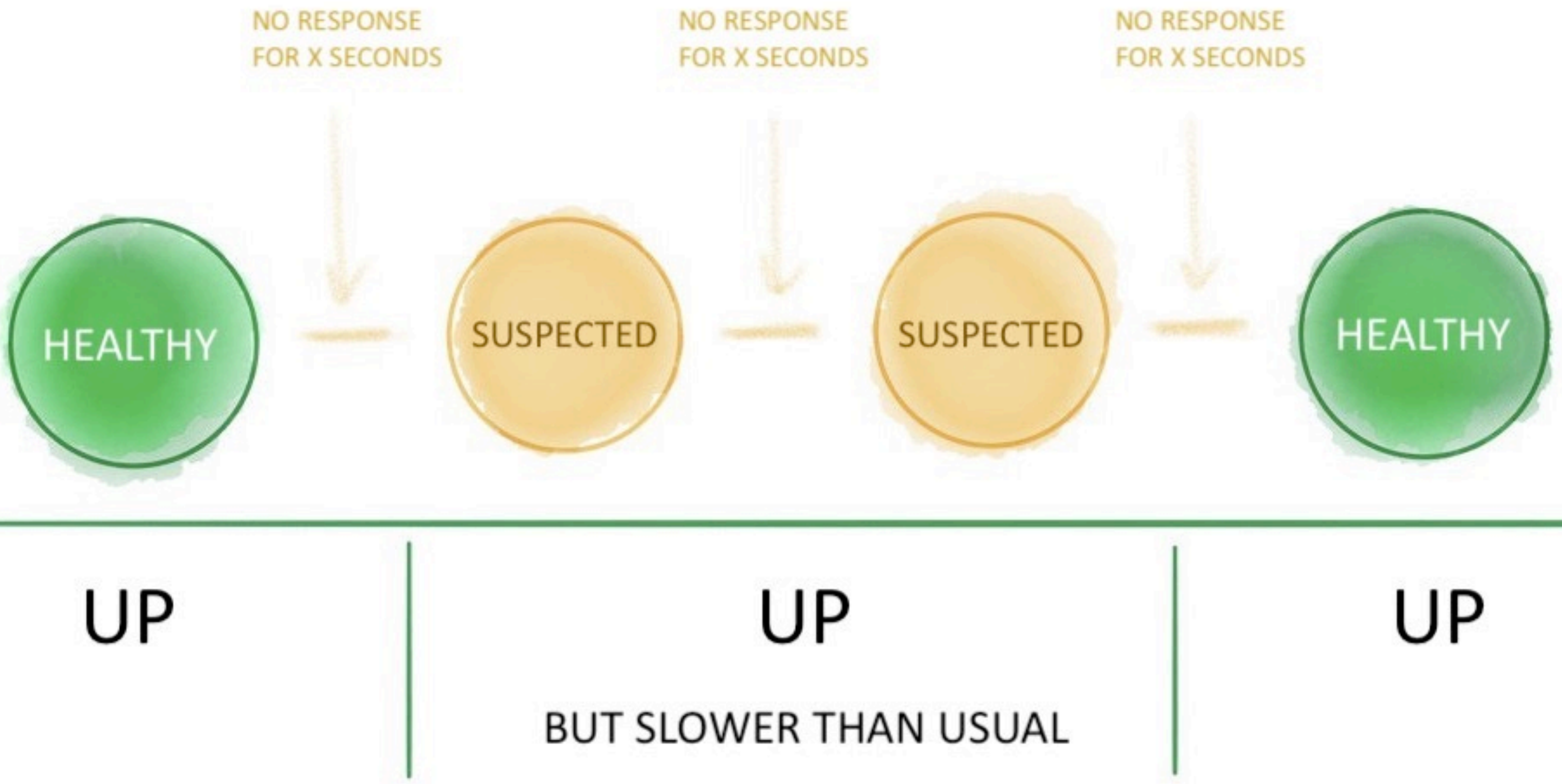
Properties of a Failure Detector

Completeness


2
3
1
4
5
2
3
1
4
5
2
3
1
5
4
2 crashed
3 crashed
2 crashed
3 crashed
2 crashed
3 crashed


2
3
1
4
5
2
3
1
4
5
2
3
1
4
5
2 crashed
3 crashed

Accuracy



NO RESPONSE FOR X SECONDS

NO RESPONSE FOR X SECONDS

NO RESPONSE FOR X SECONDS

HEALTHY

SUSPECTED

SUSPECTED

HEALTHY

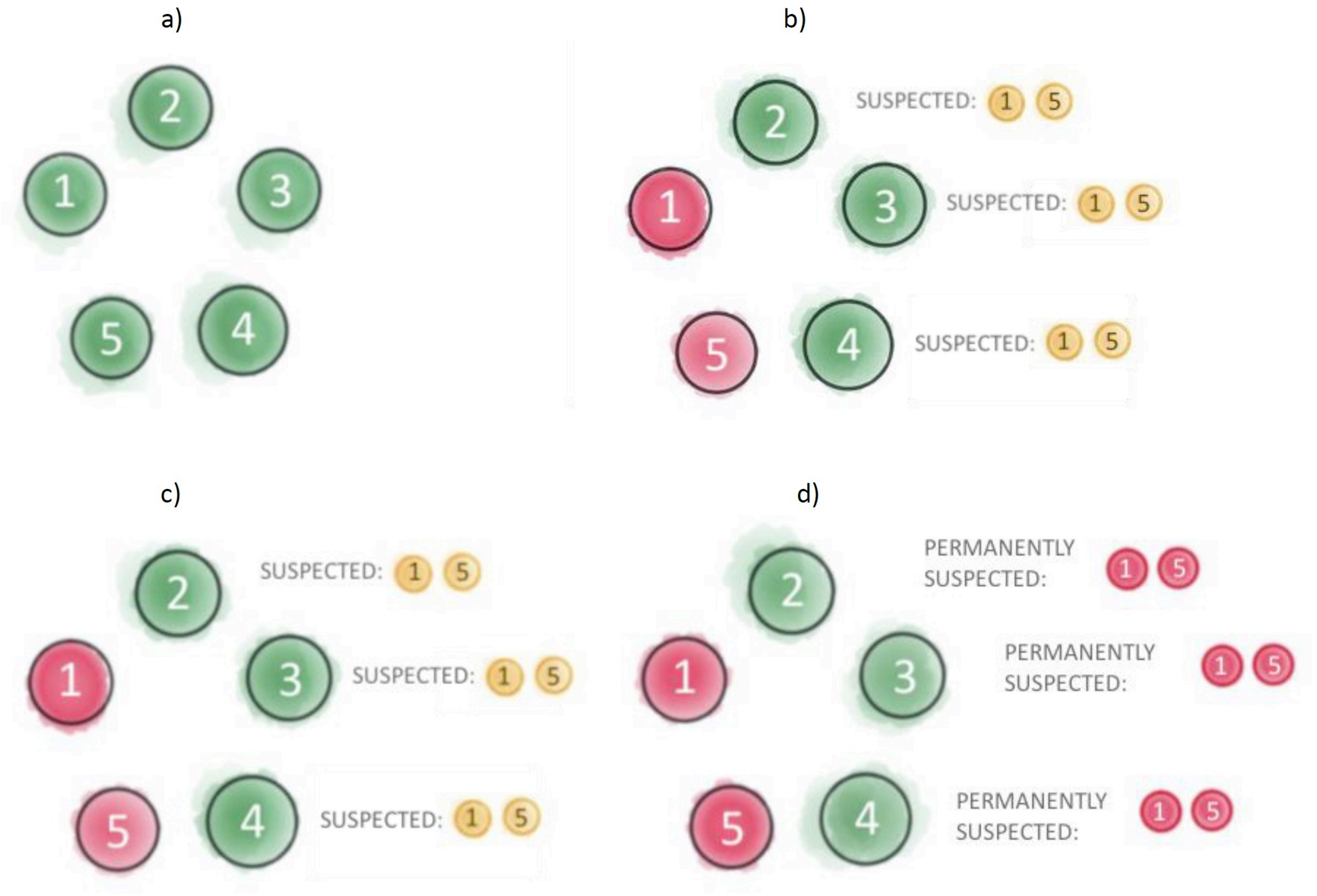
UP

UP

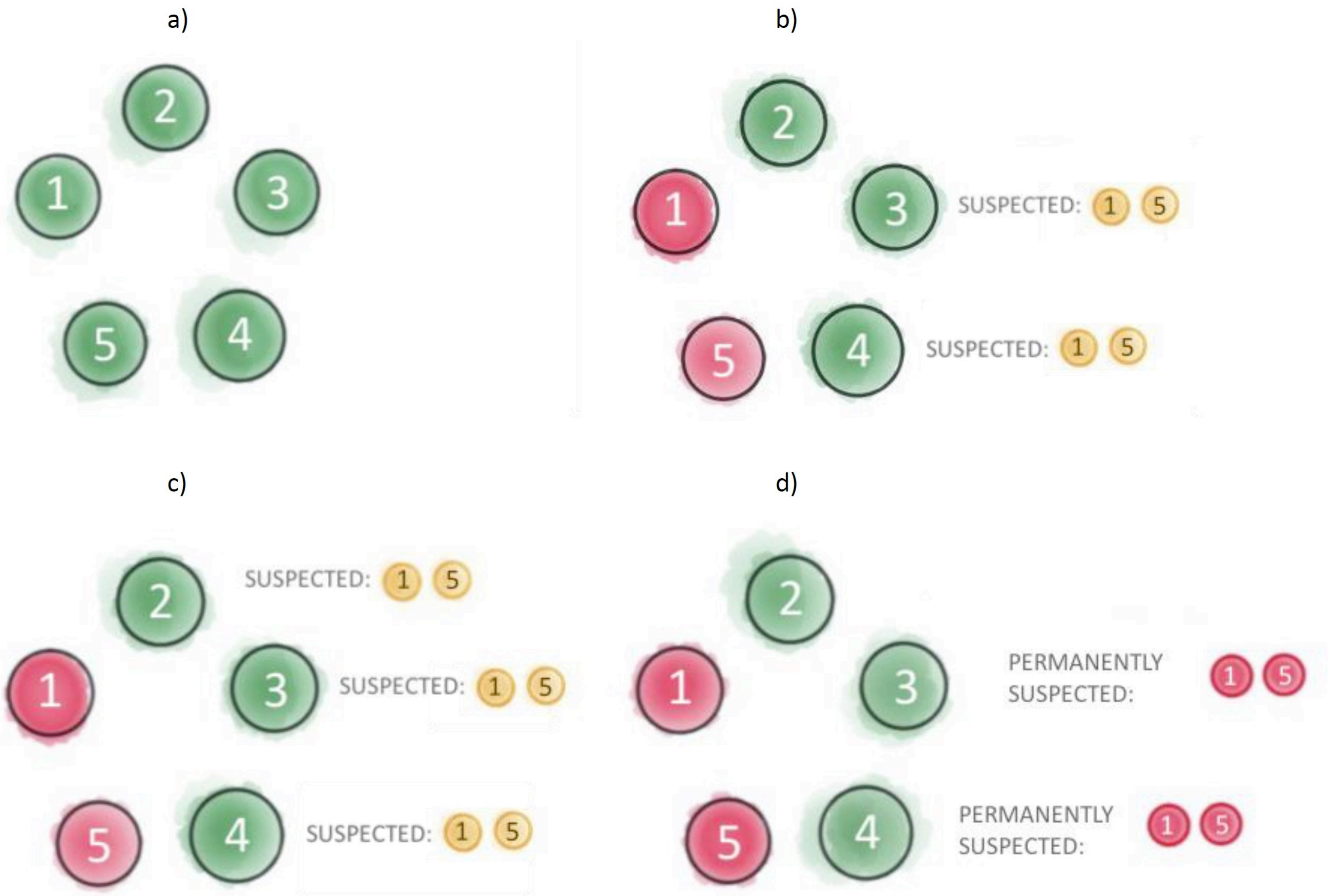
BUT SLOWER THAN USUAL

UP

lenadroid


a)
2
1
3
5
4
b)
2
SUSPECTED: 1 5
1
3
SUSPECTED: 1 5
5
4
SUSPECTED: 1 5
c)
2
SUSPECTED: 1 5
1
3
SUSPECTED: 1 5
5
4
SUSPECTED: 1 5
d)
2
PERMANENTLY SUSPECTED: 1 5
1
3
PERMANENTLY SUSPECTED: 1 5
5
4
PERMANENTLY SUSPECTED: 1 5

lenadroid


a)
2
1
3
5
4
b)
2
1
3
SUSPECTED: 1 5
5
4
SUSPECTED: 1 5
c)
2
SUSPECTED: 1 5
1
3
SUSPECTED: 1 5
5
4
SUSPECTED: 1 5
d)
2
1
3
PERMANENTLY SUSPECTED: 1 5
5
4
PERMANENTLY SUSPECTED: 1 5

Types Of Failure Detectors

- **Perfect Failure Detector:** Strong Completeness, Strong Accuracy
- **Eventually Perfect Failure Detector:** Strong Completeness, Eventual Strong Accuracy
- **Strong Failure Detector:** Strong Completeness, Weak Accuracy
- **Eventually Strong Failure Detector:** Strong Completeness, Eventual Weak Accuracy
- **Weak Failure Detector:** Weak Completeness, Weak Accuracy
- **Eventually Weak Failure Detector:** Weak Completeness, Eventual Weak Accuracy
- **Quasi-Perfect Failure Detector:** Weak Completeness, Strong Accuracy
- **Eventually Quasi-Perfect Failure Detector:** Weak Completeness, Eventual Strong Accuracy

Failure Detectors In Asynchronous Environment

From Theory To Practice

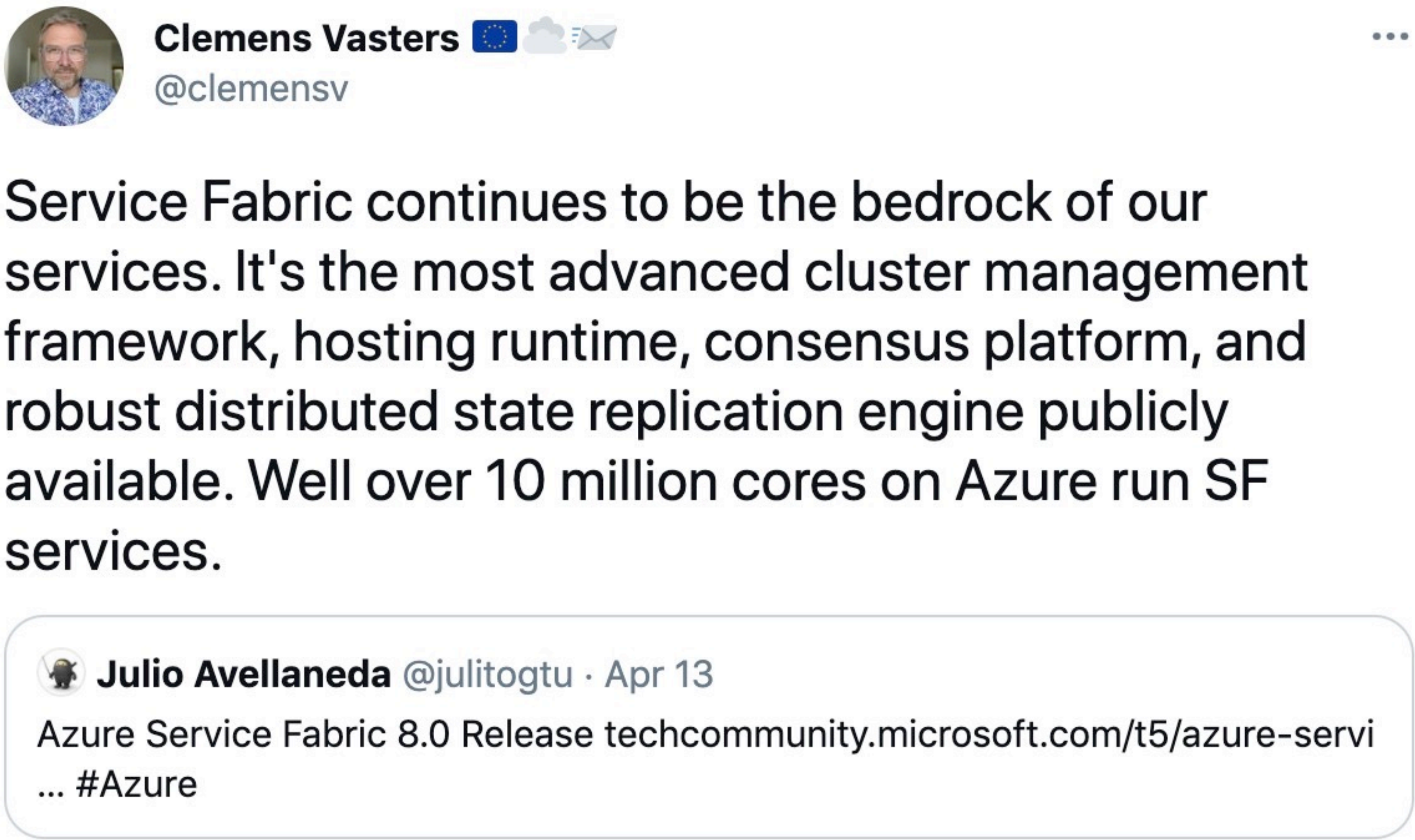
Detecting Failures in the Wild

lenadroid

Service Fabric

”When people ask what is the core replication or consensus algorithm - when in the raft paper it’s mentioned that a certain optimization is left out - Service Fabric has it. It’s a fighter jet that you don’t need to take to go to the grocery store”

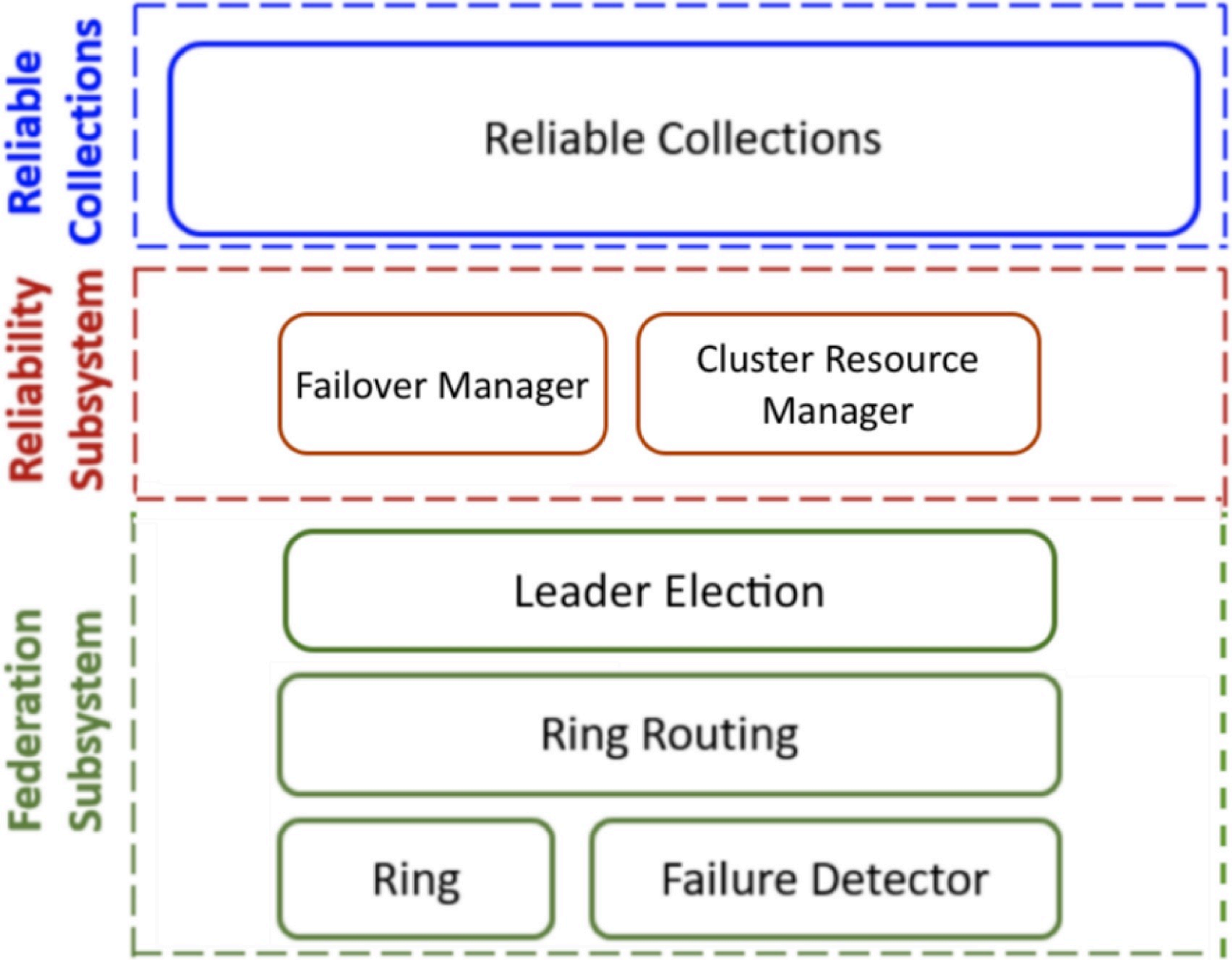
- *Matthew Snider*



Clemens Vasters 🇪🇺☁️📨
@clemensv

Service Fabric continues to be the bedrock of our services. It's the most advanced cluster management framework, hosting runtime, consensus platform, and robust distributed state replication engine publicly available. Well over 10 million cores on Azure run SF services.

Julio Avellaneda @julitogtu · Apr 13
Azure Service Fabric 8.0 Release techcommunity.microsoft.com/t5/azure-servi … #Azure



Reliable Collections

Reliable Collections

Reliability Subsystem

Failover Manager

Cluster Resource Manager

Federation Subsystem

Leader Election

Ring Routing

Ring

Failure Detector

lenadroid

Concepts

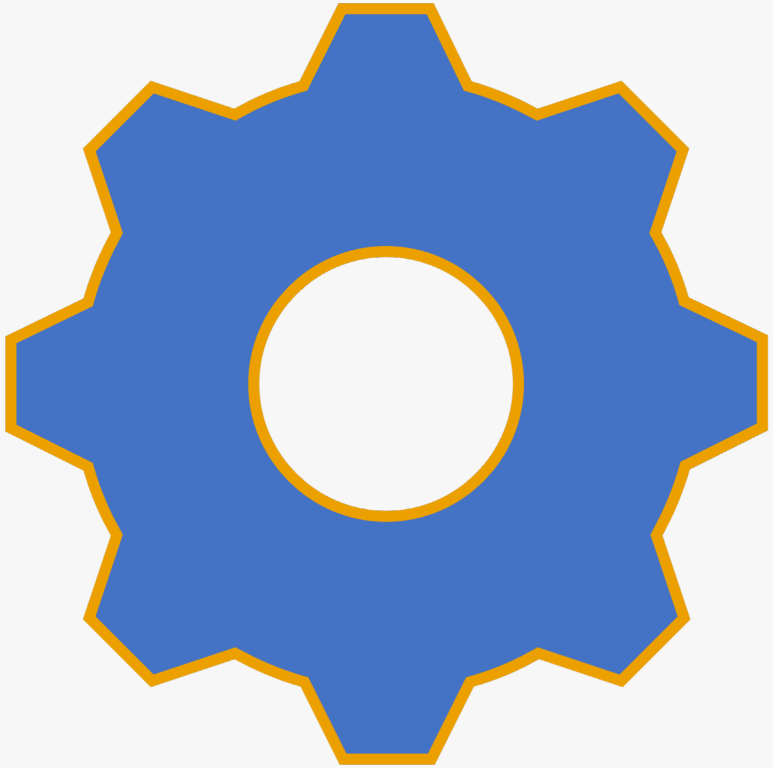
Virtual Ring

Neighborhood

Lease Mechanism

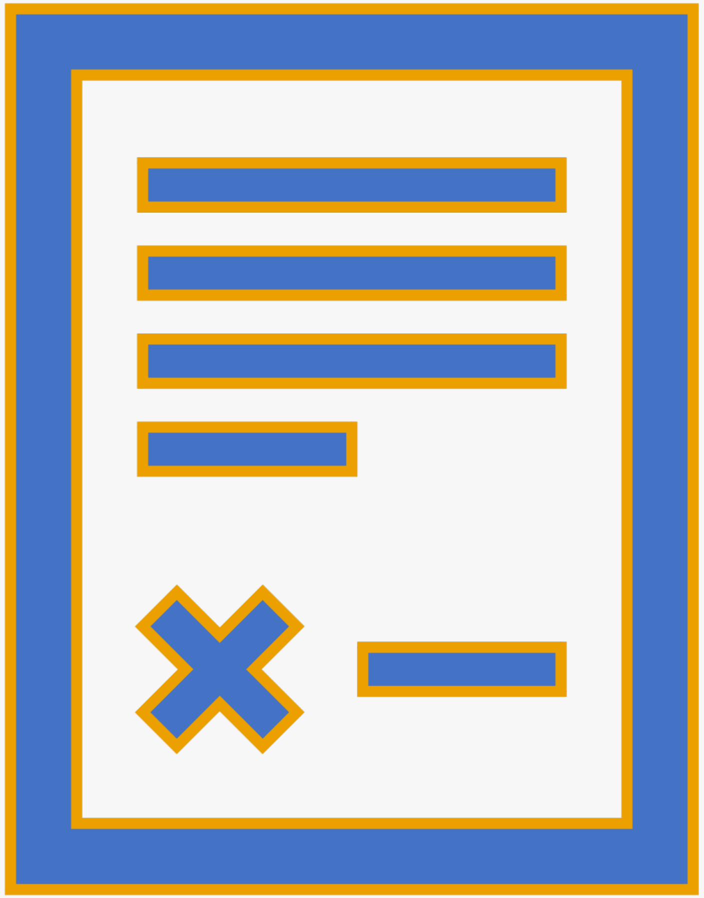
Arbitration

Virtual Ring



Neighborhood

Lease Mechanism



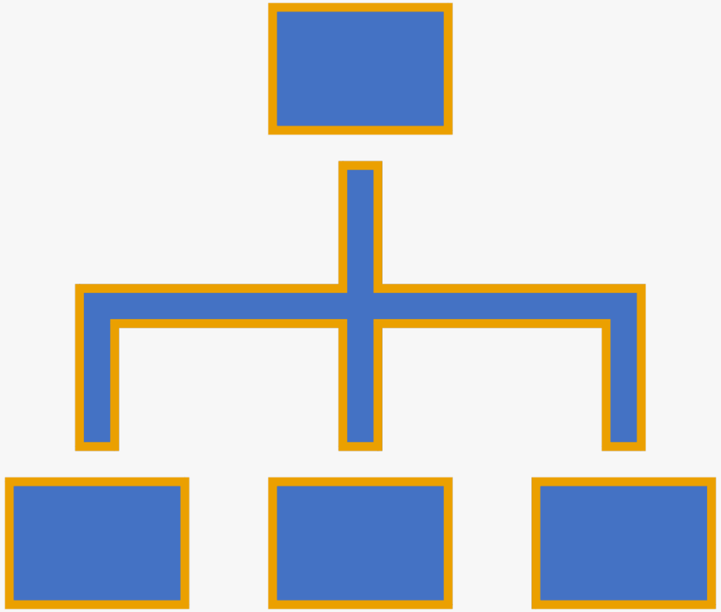
lenadroid

Arbitration

lenadroid

Partitions

Failover Manager

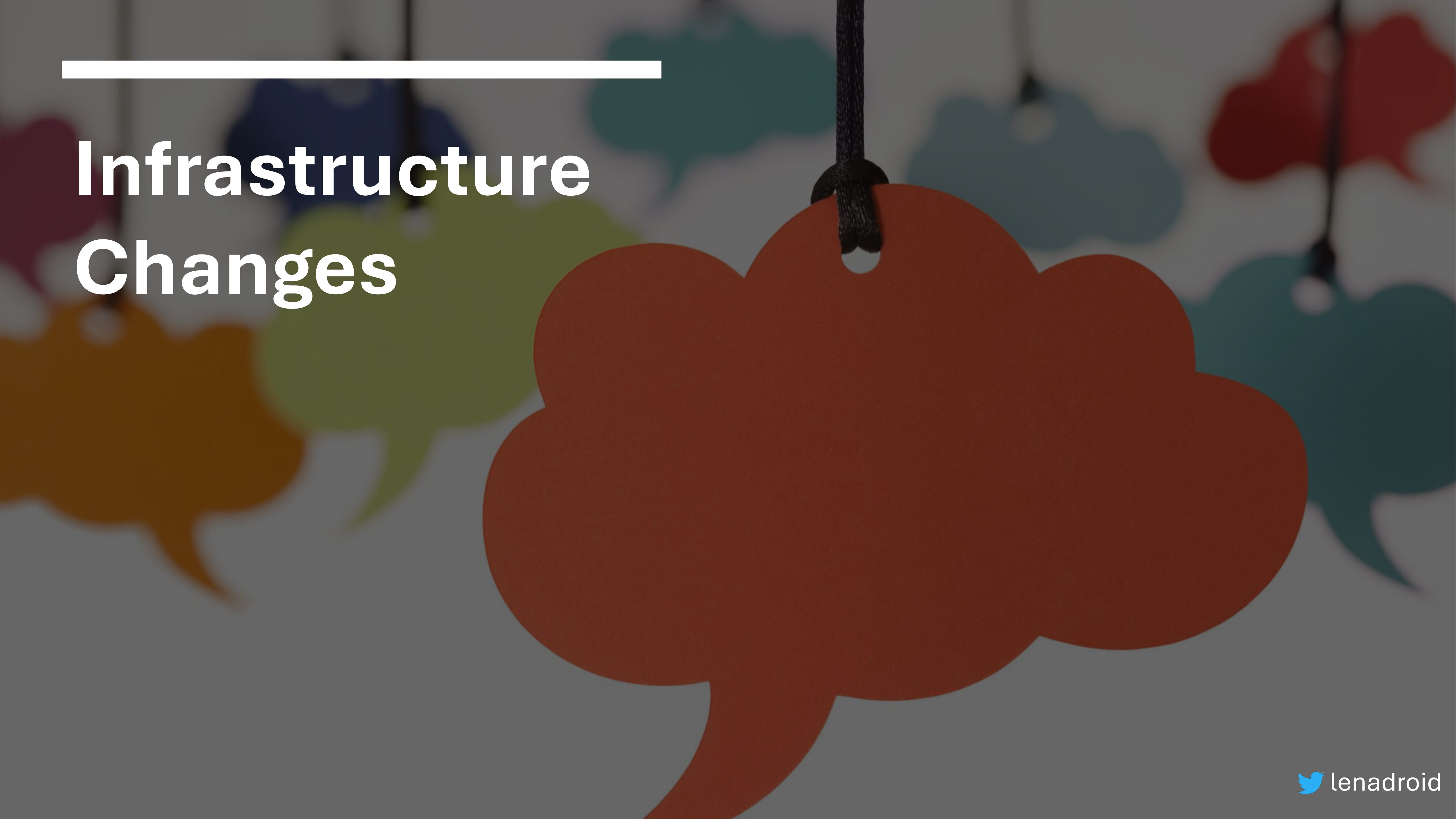


lenadroid

Is It Slow or Is It Dead?

lenadroid

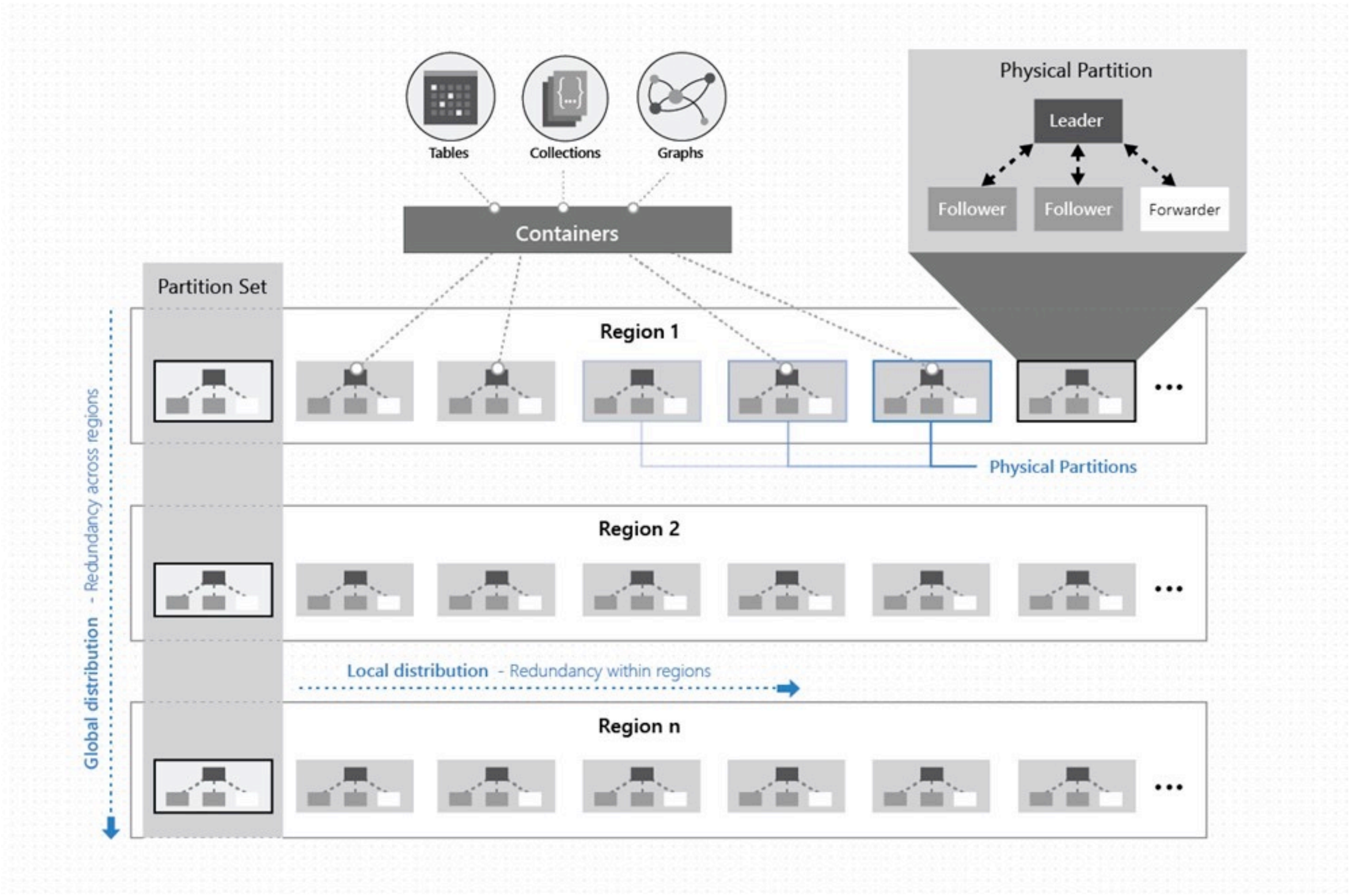
Infrastructure Changes



Azure Cosmos DB

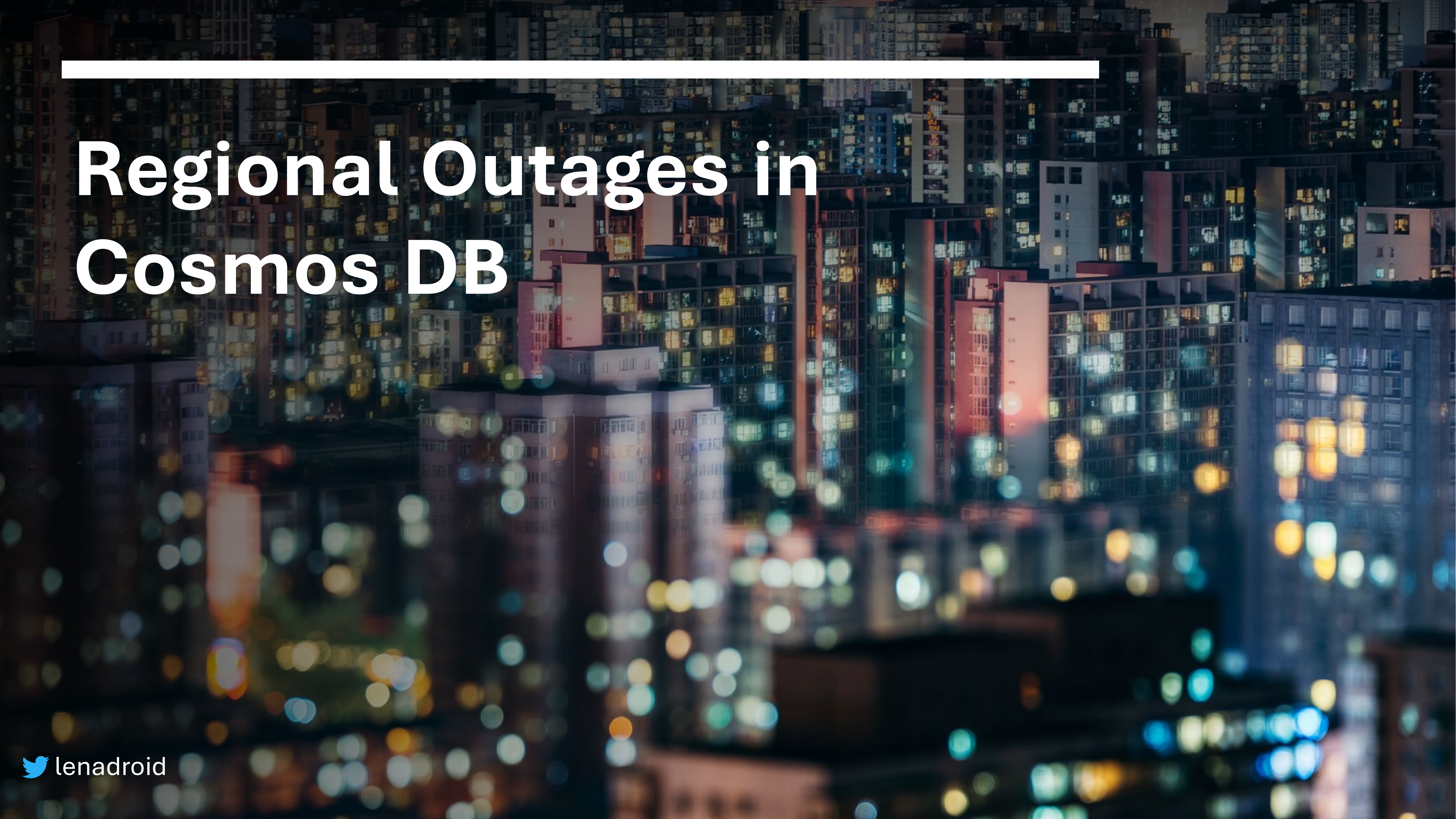
lenadroid

Global data distribution with Azure Cosmos DB - under the hood



lenadroid

Cosmos DB Relies on Service Fabric



Regional Outages in Cosmos DB

lenadroid

Detecting Failures in Cosmos DB Central Replica-Set Hub

USING HEARTBEATS

VOTING PHASES

INITIATED FAILOVER

Cosmos DB’s Failure Detector Properties

Weak Completeness

Eventual Weak Accuracy

Server-Side Recovery

FAILURE IN A
READ REGION

FAILURE IN A
WRITE REGION

Other Types of Failure Recovery

Client-side redirection

Some Numbers (example)

24 hours

278 suspected failures – missed heartbeats

47 temporarily revoked the lease

Thank You!

Questions -> alehall [at] microsoft [dot] com

Follow -> @lenadroid

Transcript -> aka.ms/failure-detection-talk