August 28

KEYSPACE

Amsterdam

Demystifying Valkey Clustering: Architecture, Fault Tolerance, and Scalability

Harkrishn Patro
Senior Software Engineer - AWS/Valkey Maintainer

Valkey

Agenda

- Overview
- Operation mode
- Cluster mode components
- Cluster coordination mechanisms
- Recent improvements
- Benchmark
- Future work

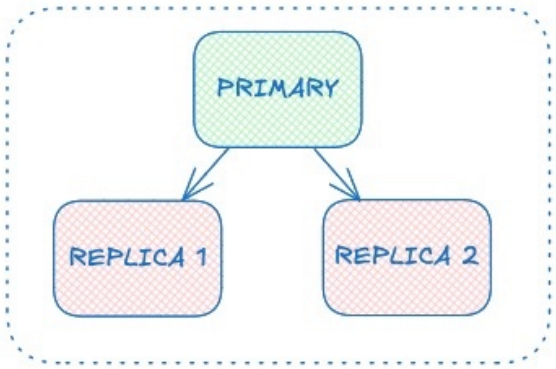
Valkey

Introducing the Valkey project

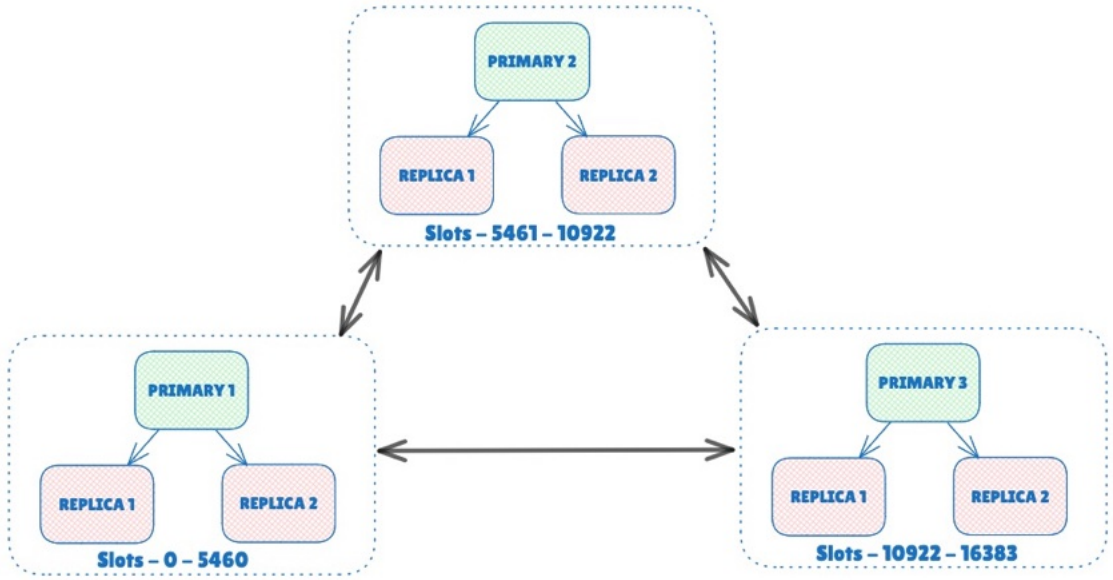
- In-memory key value store
- Supports over 200 commands and multiple data structures like hashes, sets, sorted sets
- Used for caching, session management, leaderboard, message broker

Valkey

Operation mode



STANDALONE

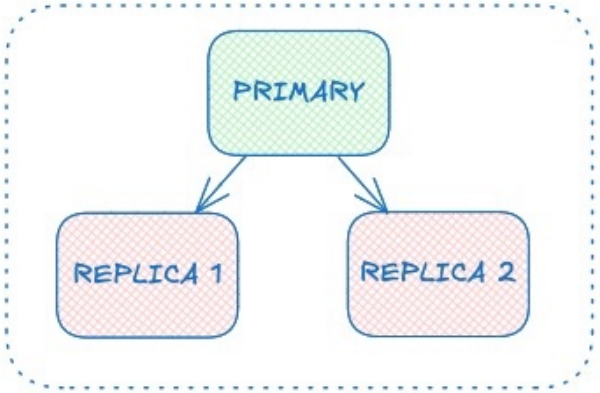


CLUSTER

Valkey

Valkey Standalone Overview

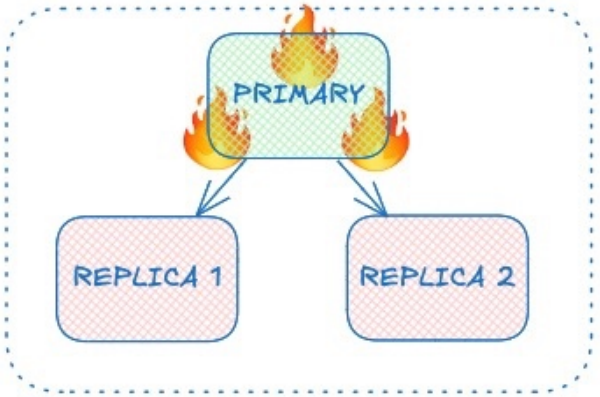
- Single keyspace
- Asynchronous replication
- Scalable reads



Valkey

Valkey Standalone - Summary

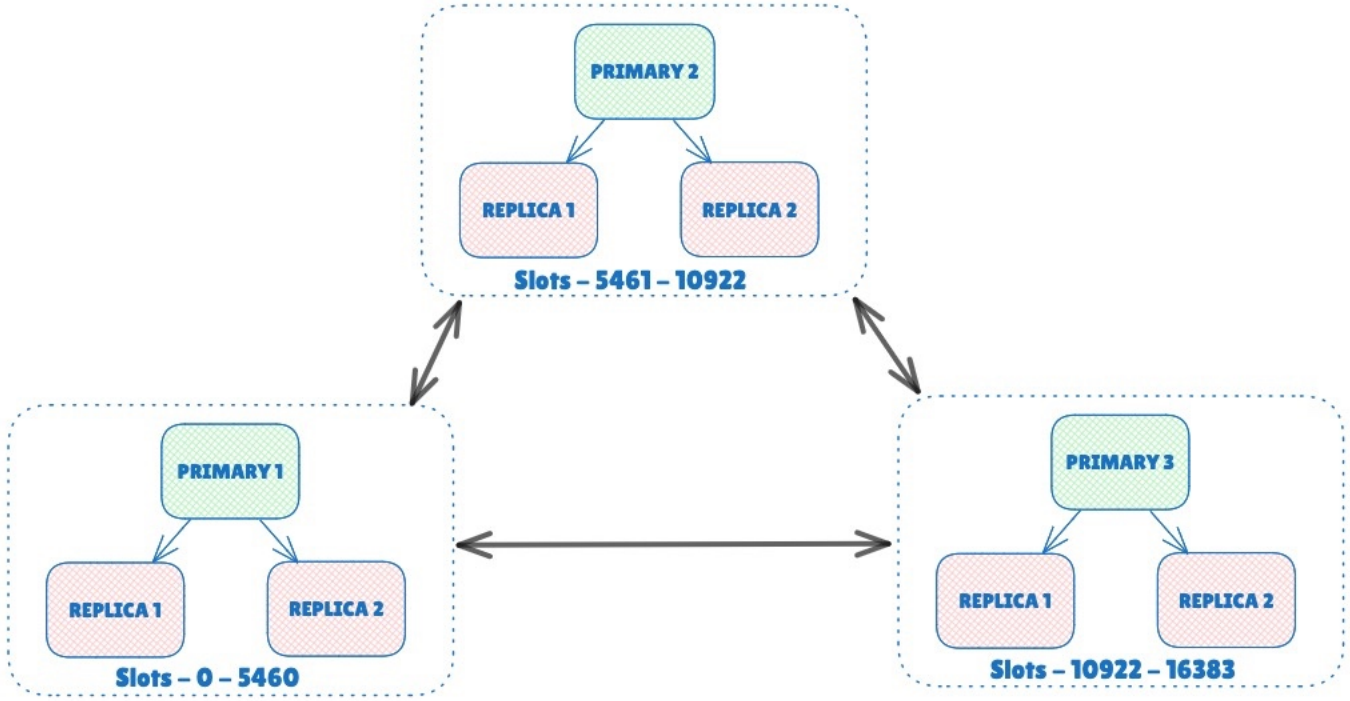
- Easy to setup
- Lack of fault tolerance
- Memory and write throughput bottleneck



Valkey

Valkey Cluster Overview

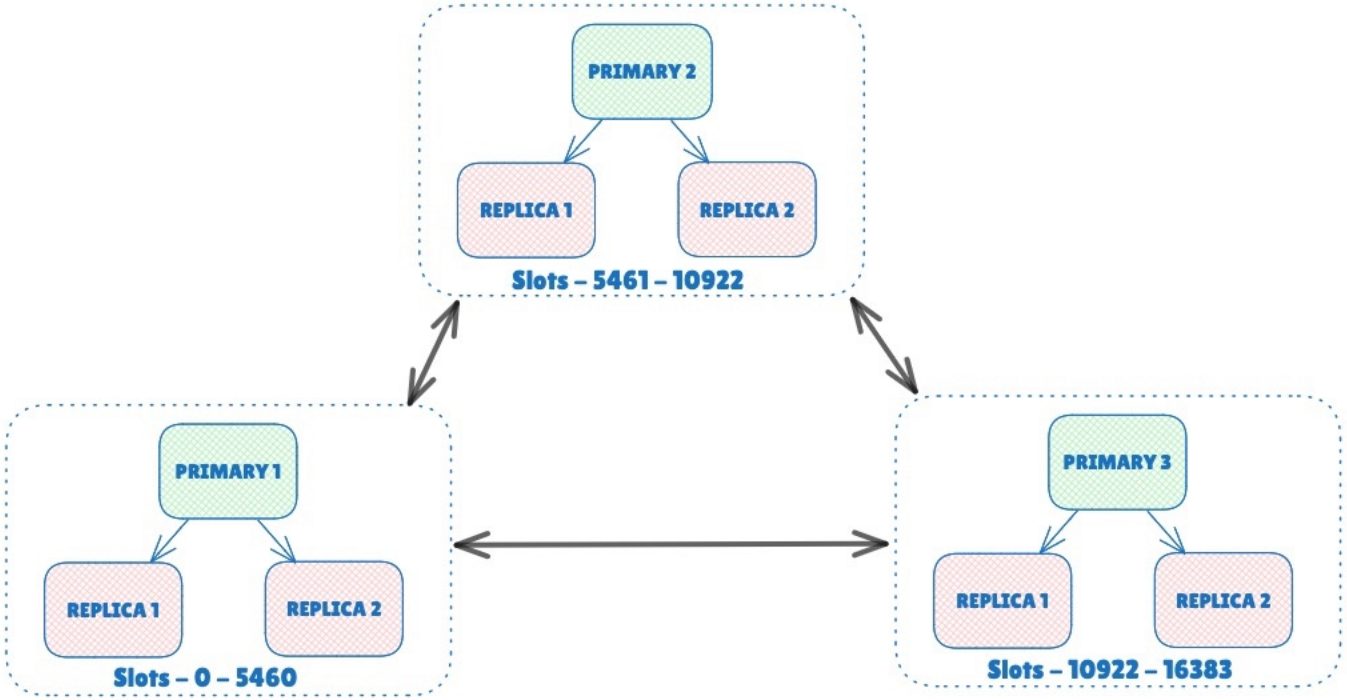
- Horizontally scalable
- Homogenous setup
- Server enabled data sharding
- Automatic client redirection
- In-built fault tolerance



Valkey

Cluster Mode Components

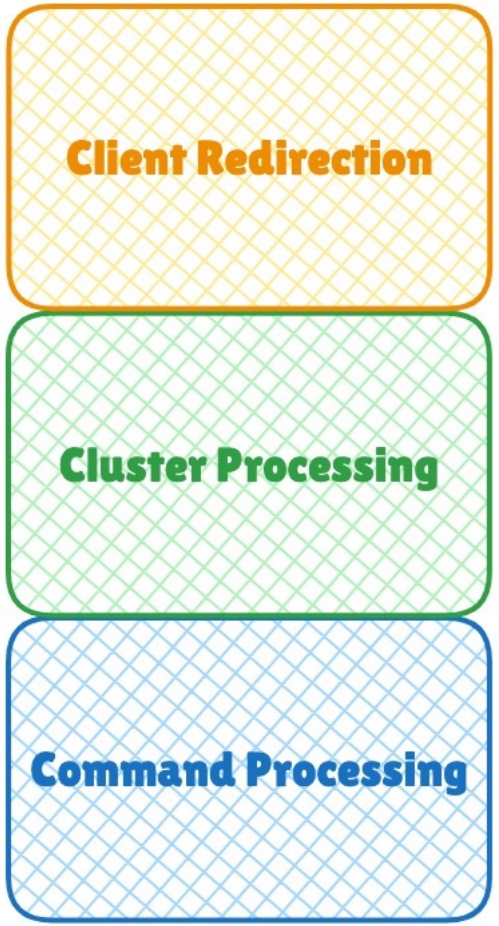
- Cluster Node
- Cluster Slots
- Cluster Bus



Valkey

Cluster Node

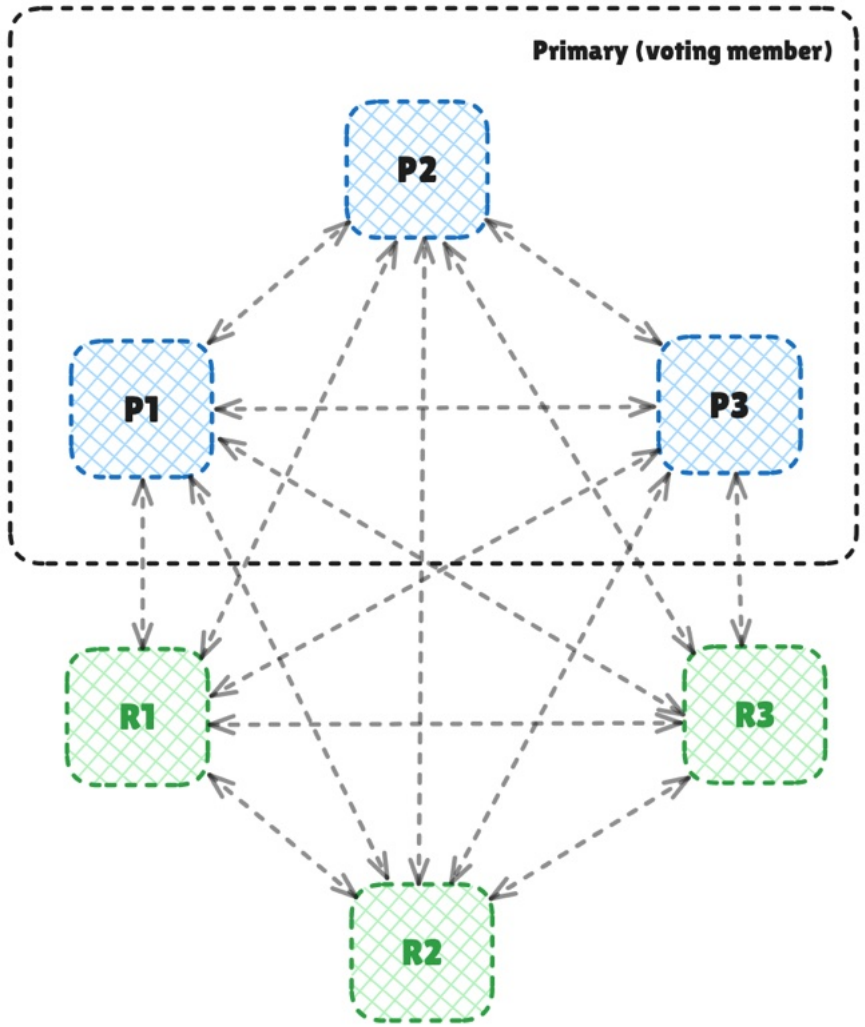
- Serves data
- Health detection
- Serves topology information



Valkey

Cluster Node (Contd.)

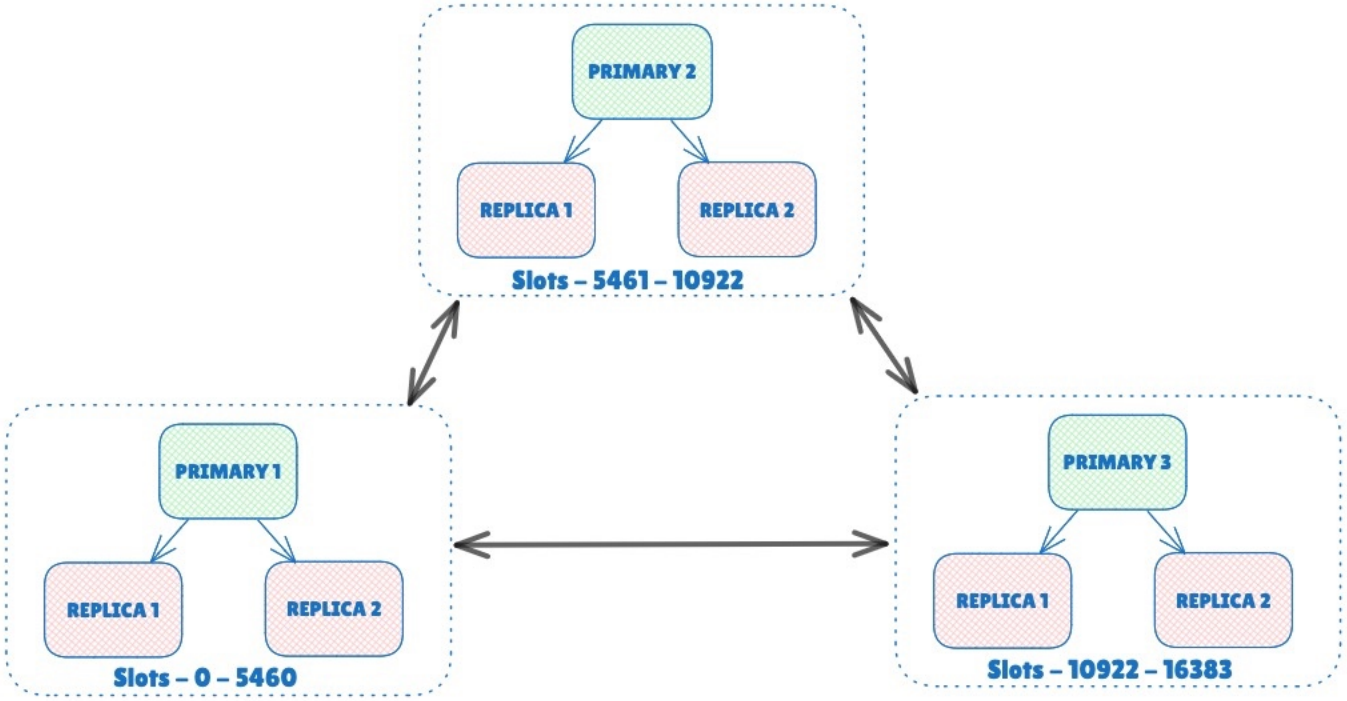
- Primary
  - Quorum - Voting member
- Replica
  - Election candidate



Valkey

Cluster Slots

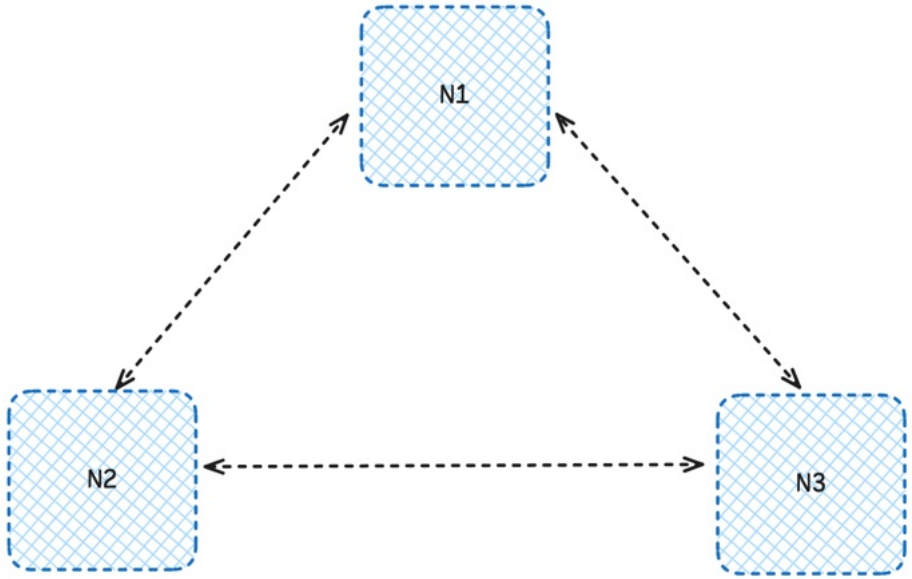
- Hashes key to a slot
- Slot to a node
- Key distribution – 16384 slots
- Algorithm - CRC16(key) % 16384
- Primary – one or more slots



Valkey

Cluster Bus

- Bidirectional persistent TCP connection between nodes
- Mesh topology
- Custom message protocol
- Gossip – piggyback information
- Supports Pub/Sub traffic



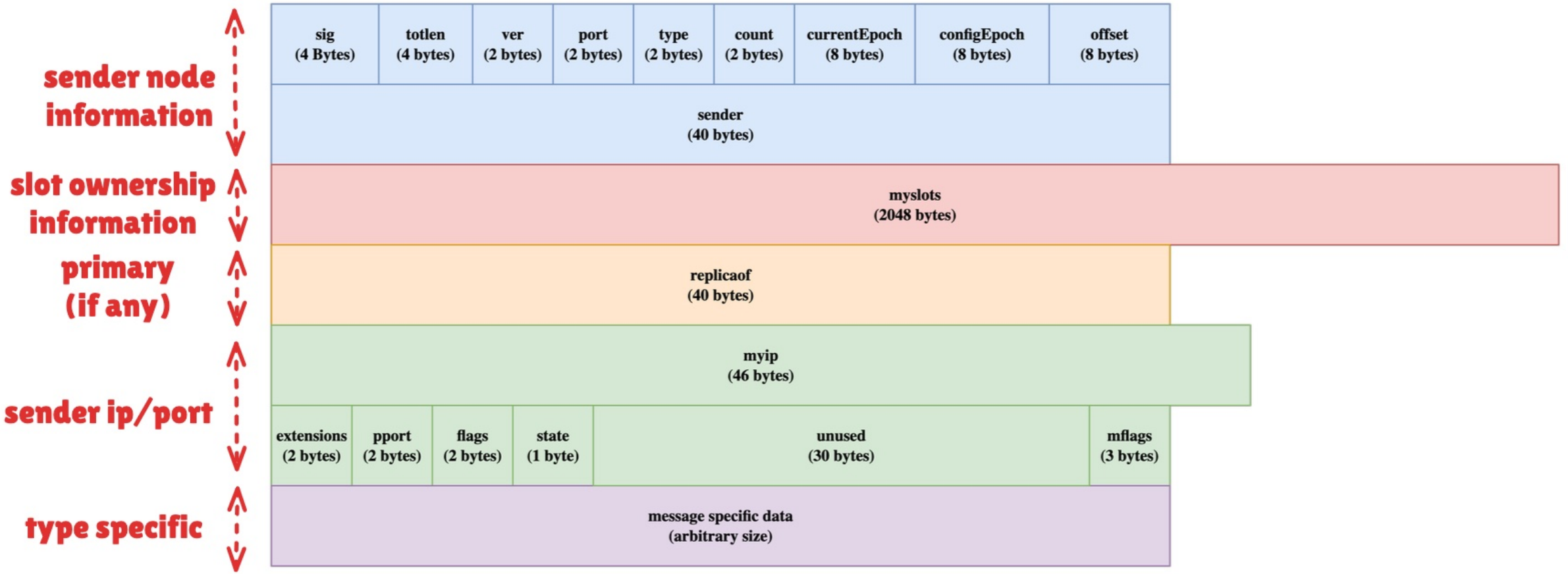
Valkey

Cluster Bus – Message Type

- New node discovery - MEET
- Heart beat – PING / PONG
- Node failure – FAIL / UPDATE
- PubSub – PUBSUB / PUBSUBSHARD

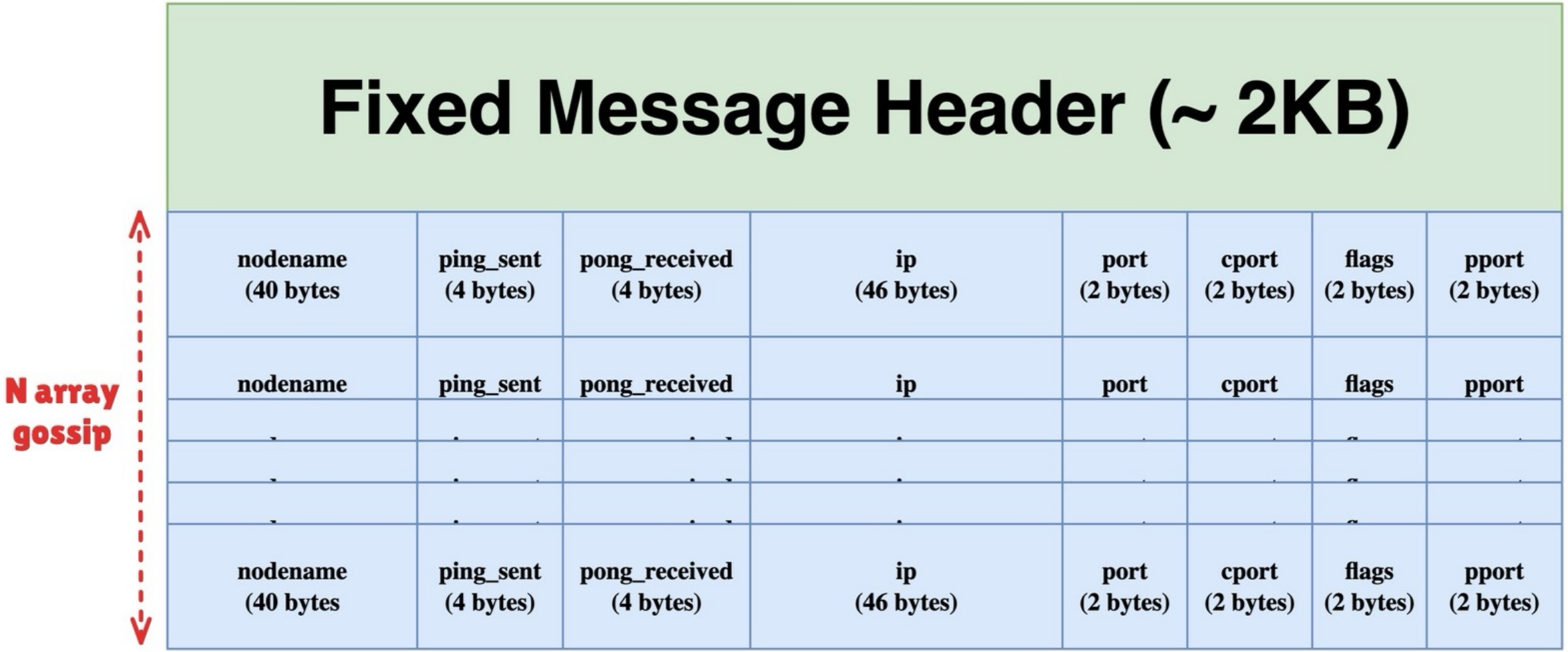
Valkey

Cluster Bus - Message Header Format



Valkey

Cluster Bus - Message Header (PING/PONG/MEET)



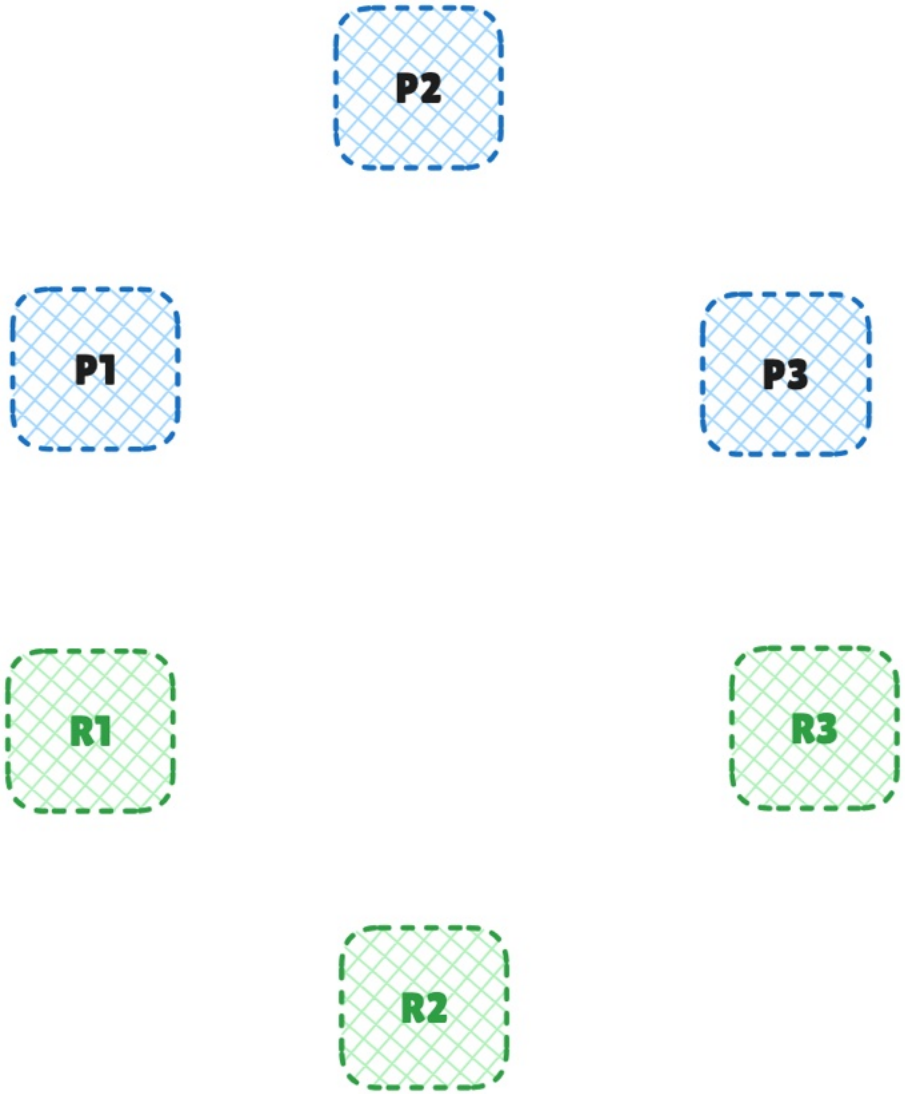
Valkey

Cluster Coordination Mechanism

- Membership
- Redirection - Client
- Failure detection
- Failover
- Versioning – Conflict Resolution

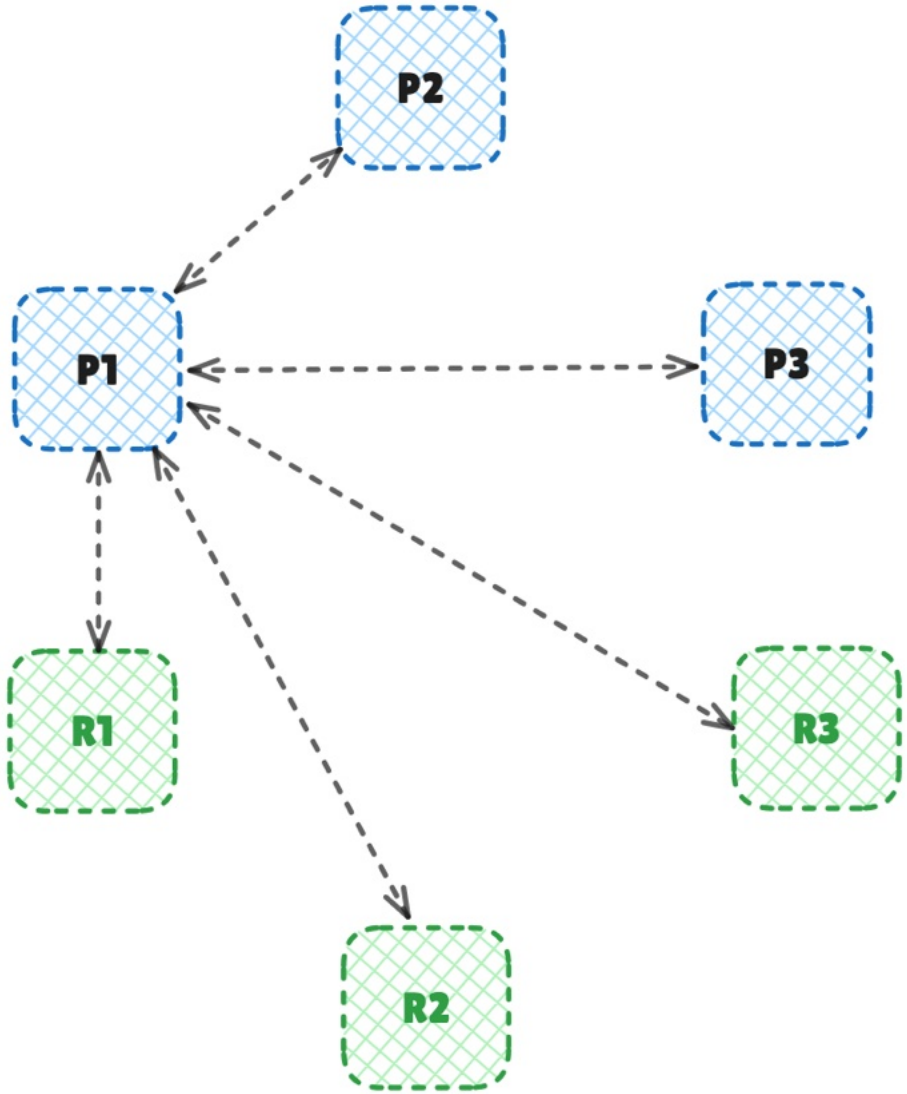
Valkey

Cluster Node Membership



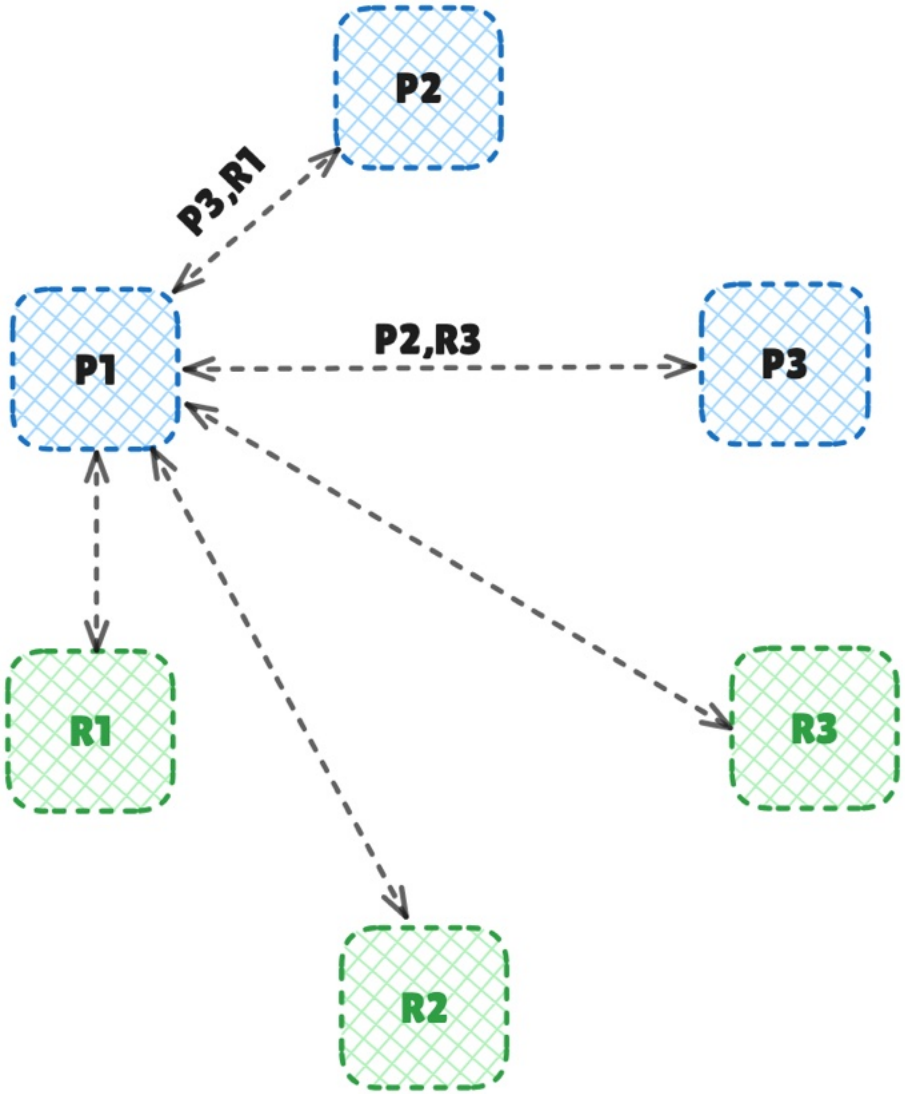
Valkey

Node Membership – Seed node discovery



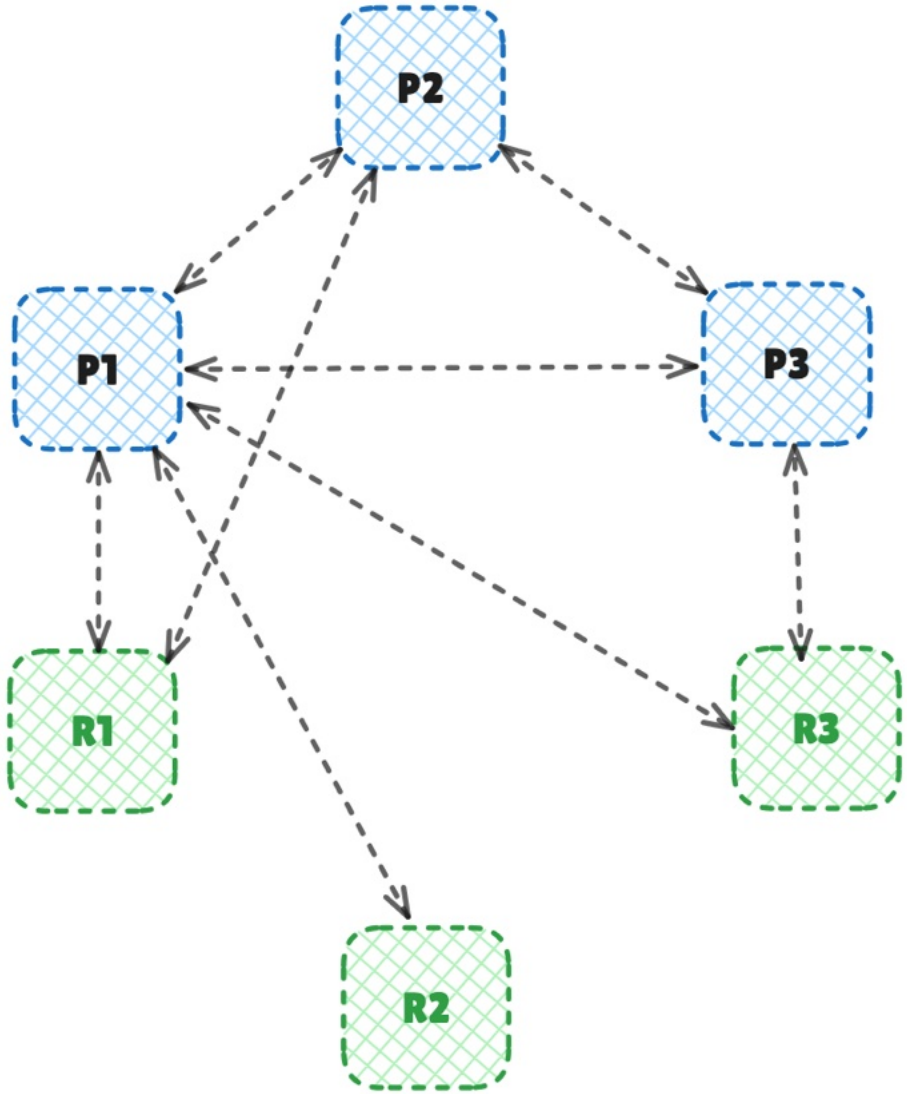
Valkey

Node Membership – Gossip



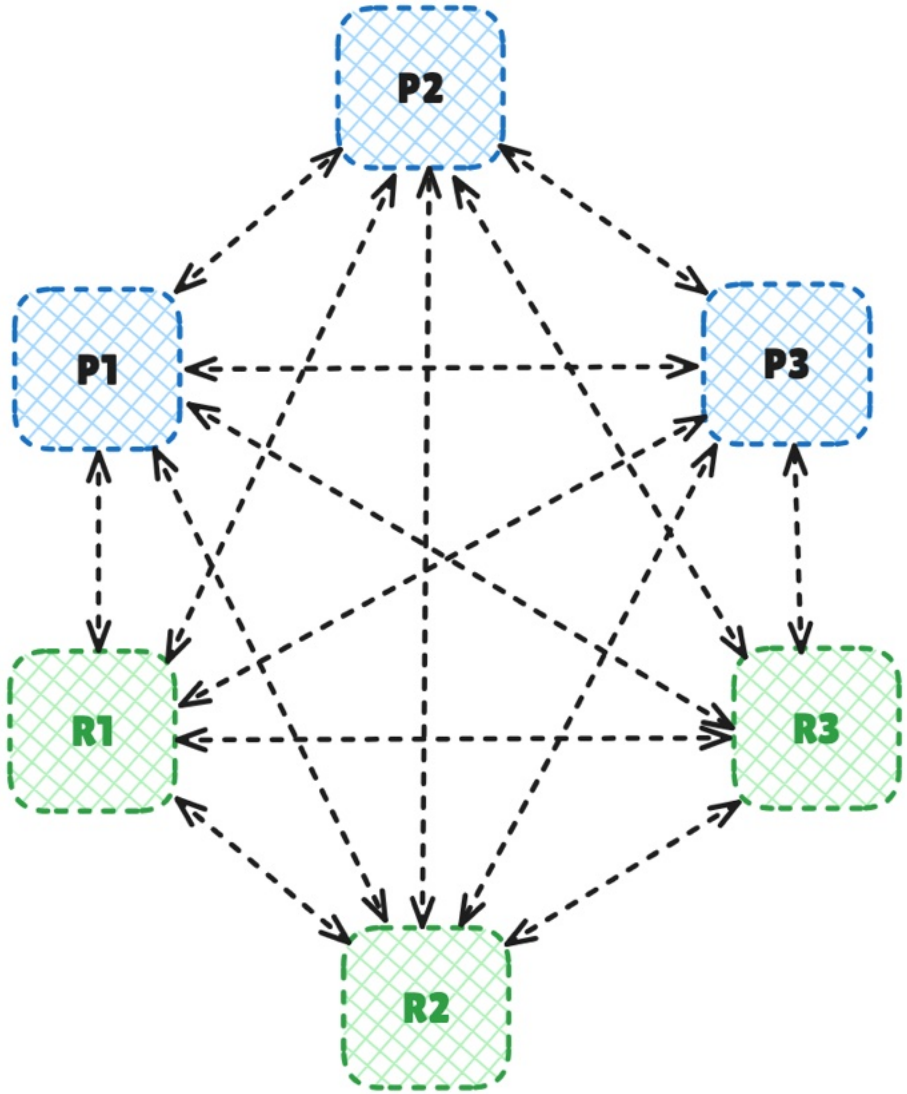
Valkey

Node Membership– Discovery via gossip



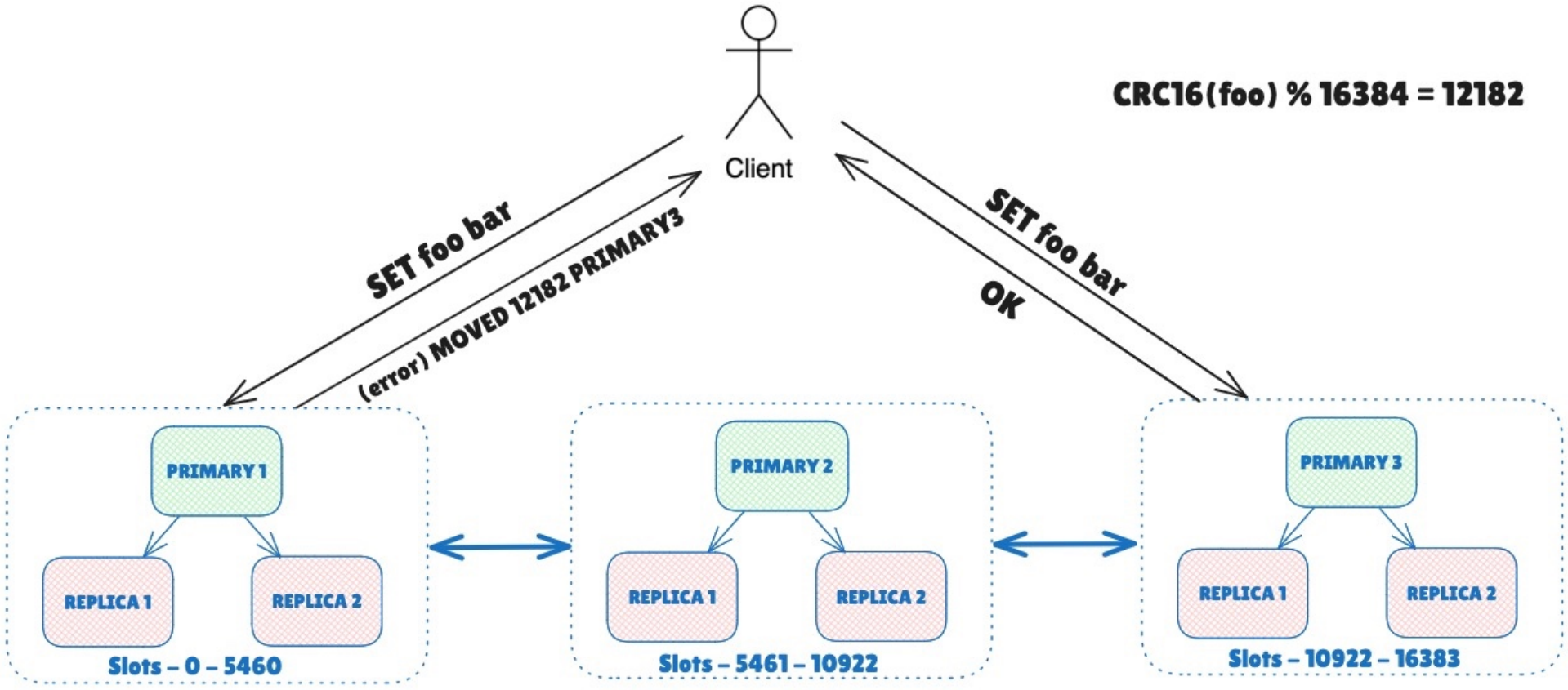
Valkey

Node Membership – Fully connected



Valkey

Node Redirection - Client



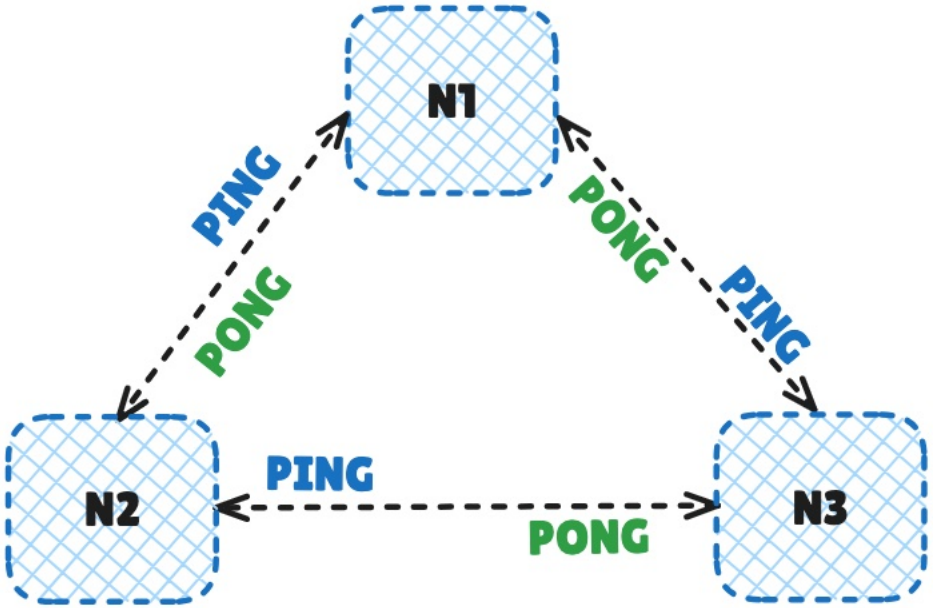
Valkey

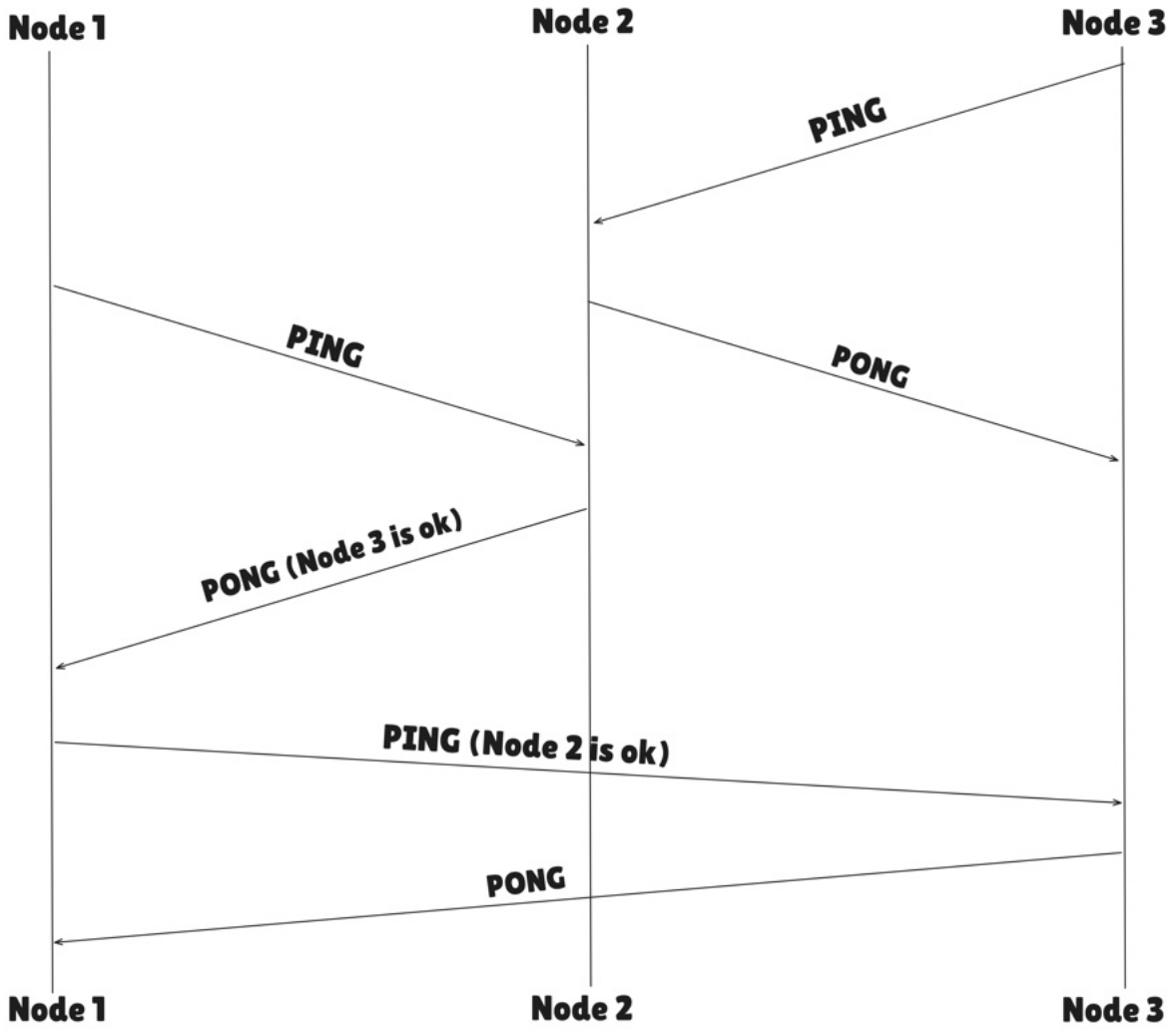
Failure Detection – PING/PONG

- ping-sent
- pong-received
- Partial failure – node-timeout / 2
- Complete failure (Quorum) – node-timeout
- node-timeout is configurable

Valkey

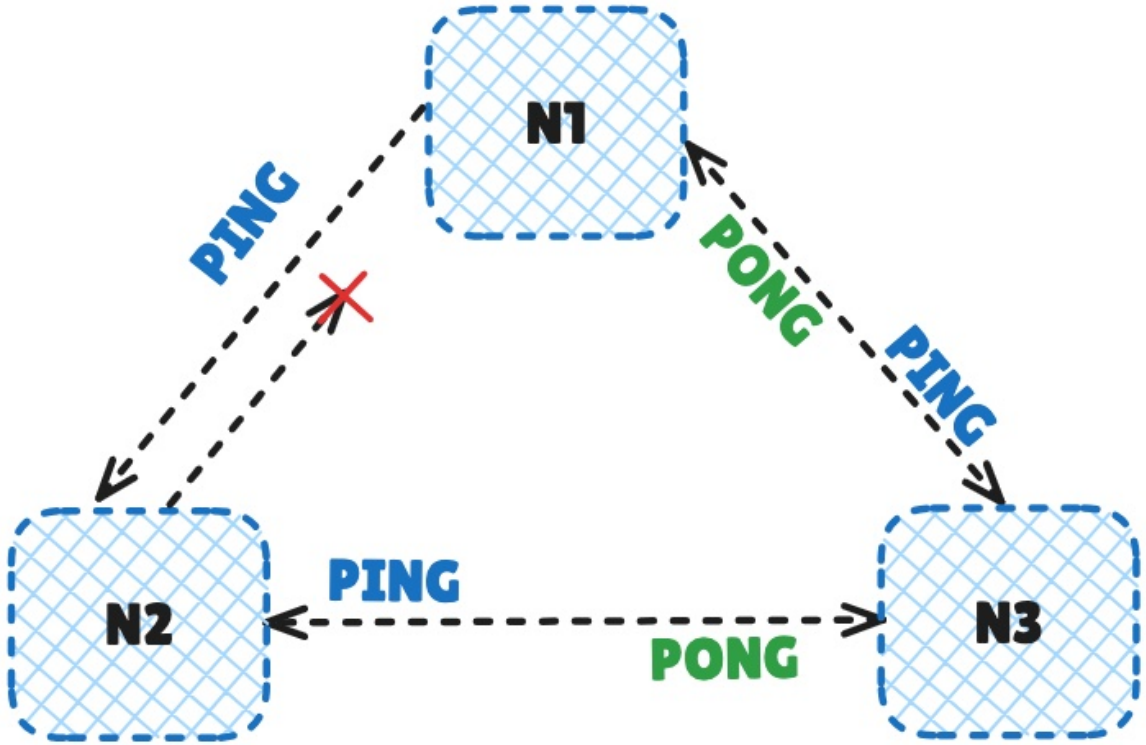
Failure Detection – Healthy State





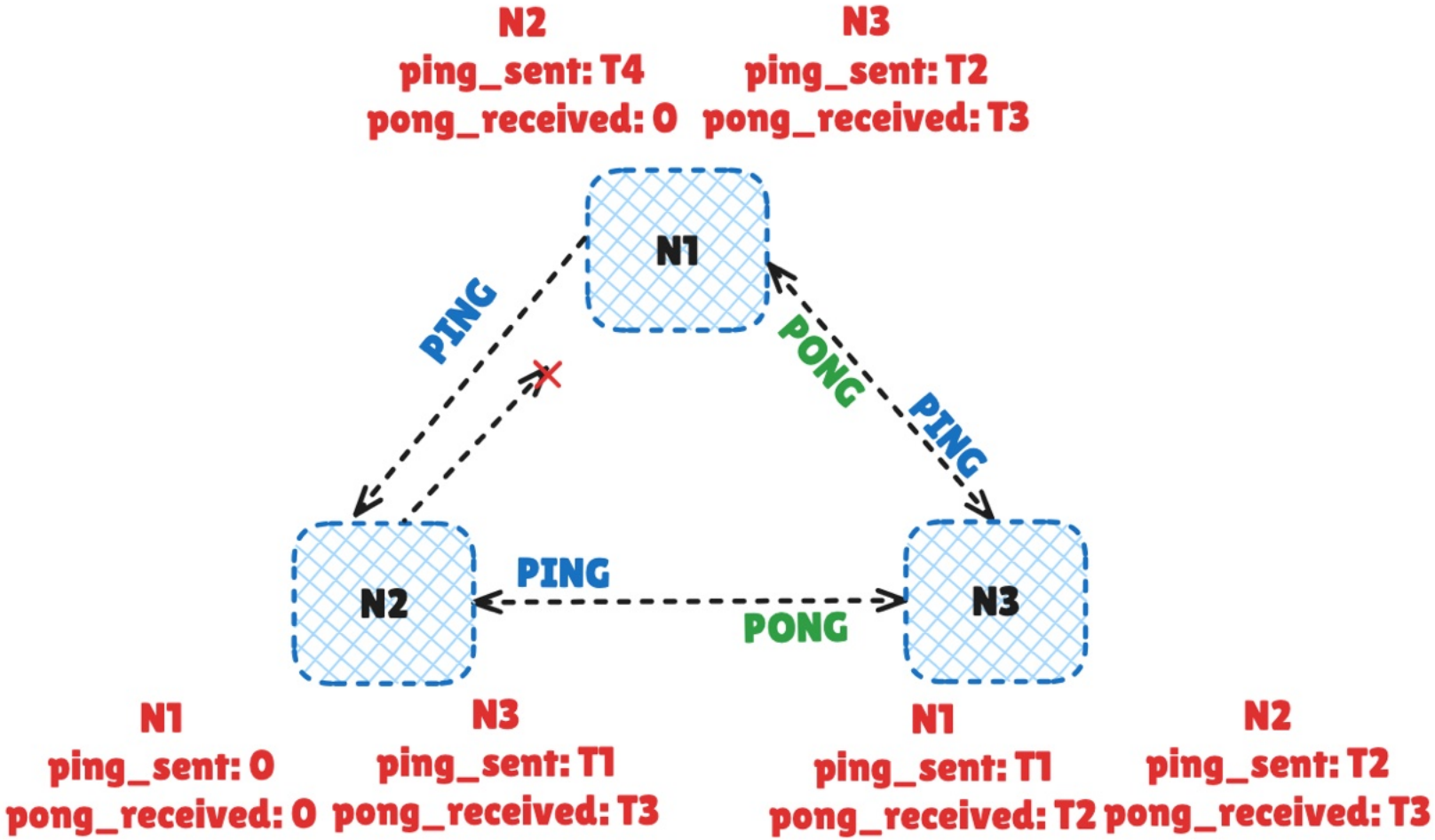
Valkey

Failure Detection – Partial failure



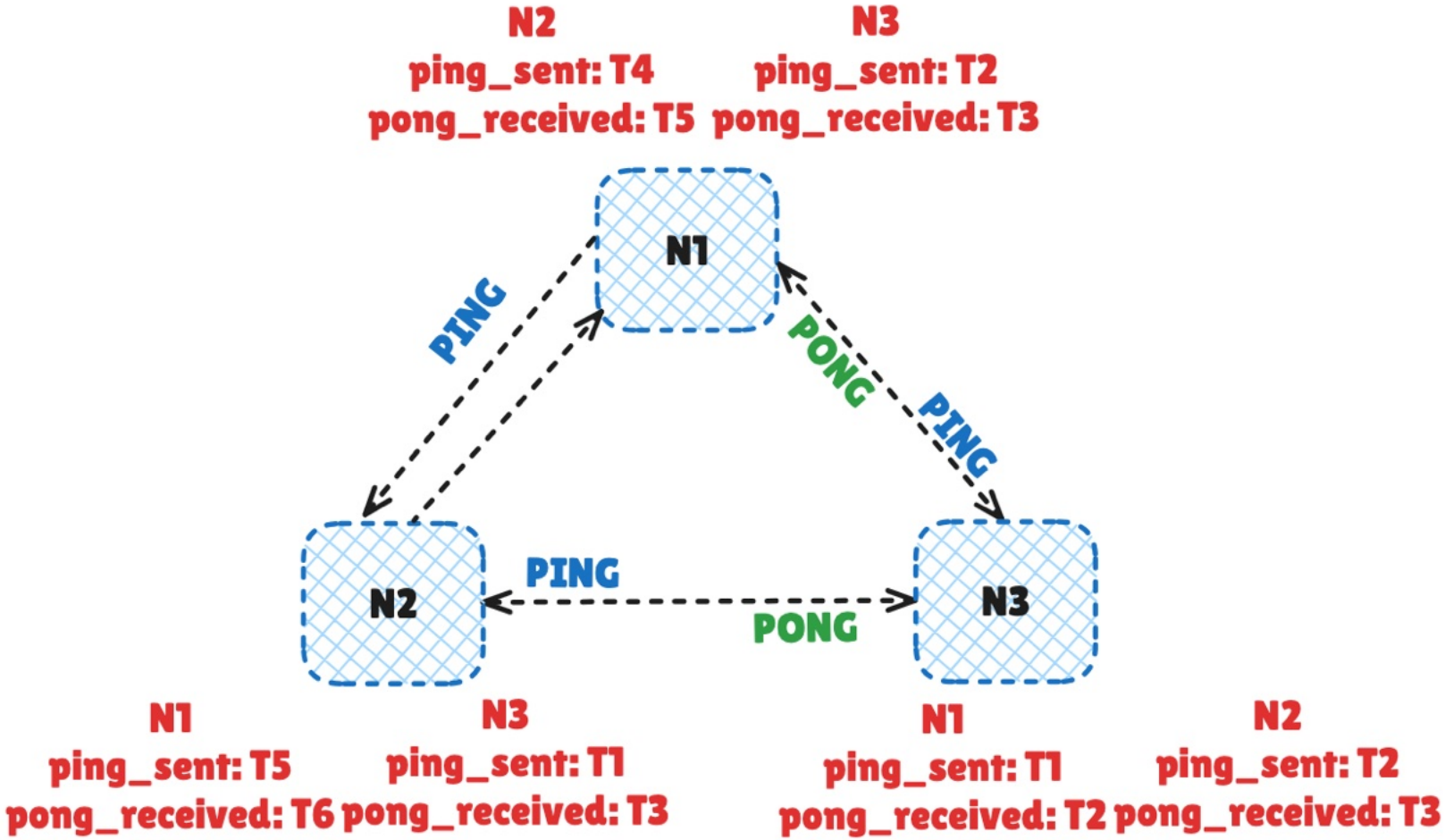
Valkey

Failure Detection – Partial failure



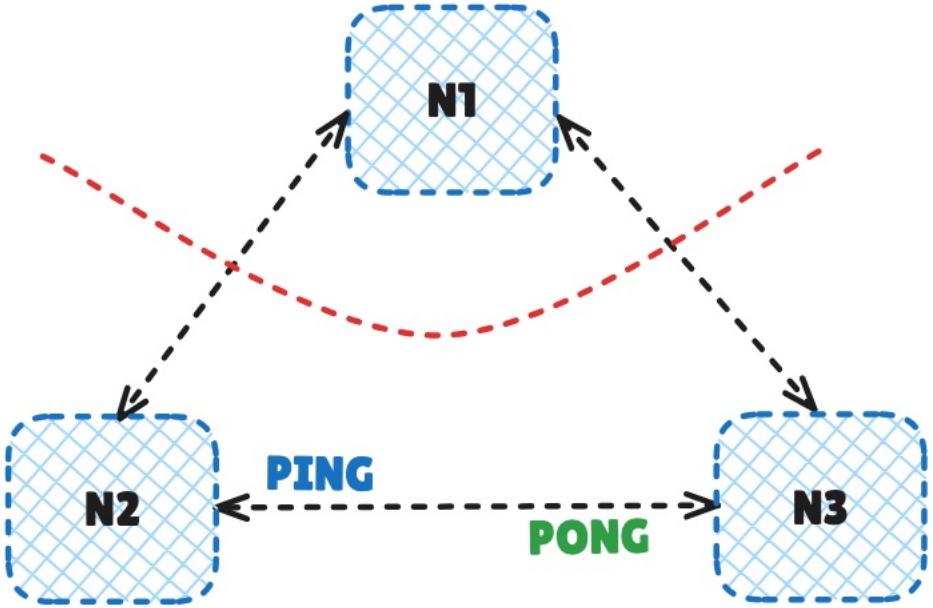
Valkey

Failure Detection – Healthy



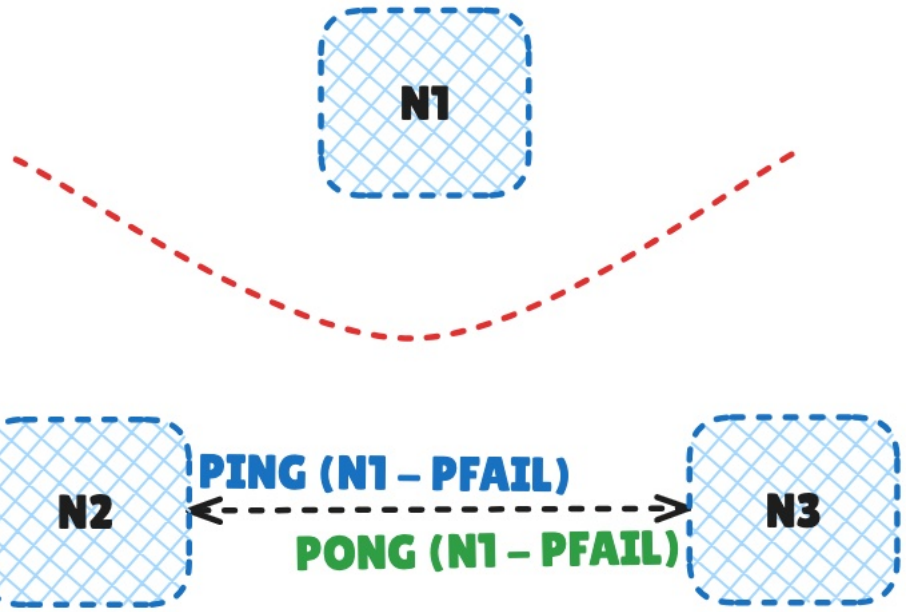
Valkey

Failure Detection - Partition



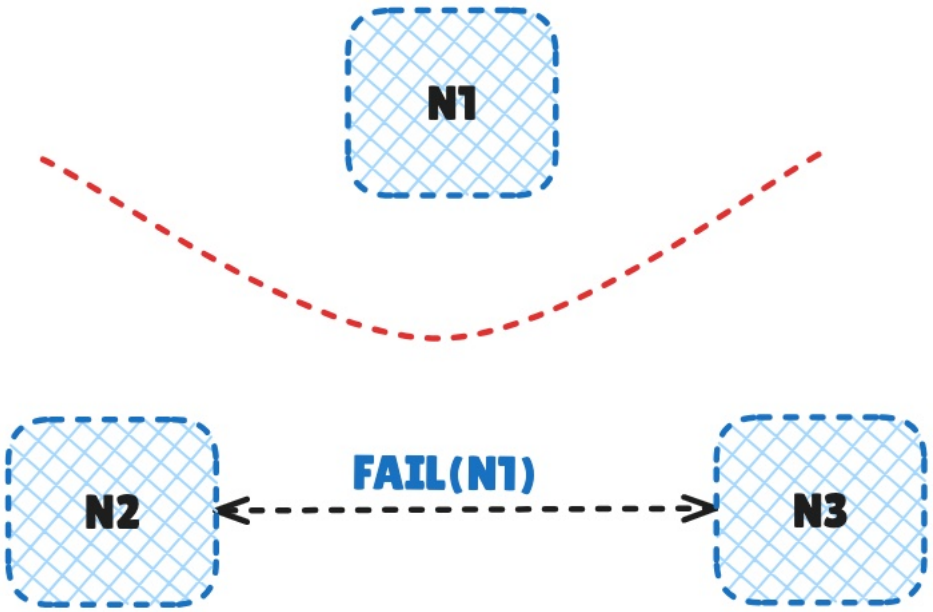
Valkey

Failure Detection – Time out gossip



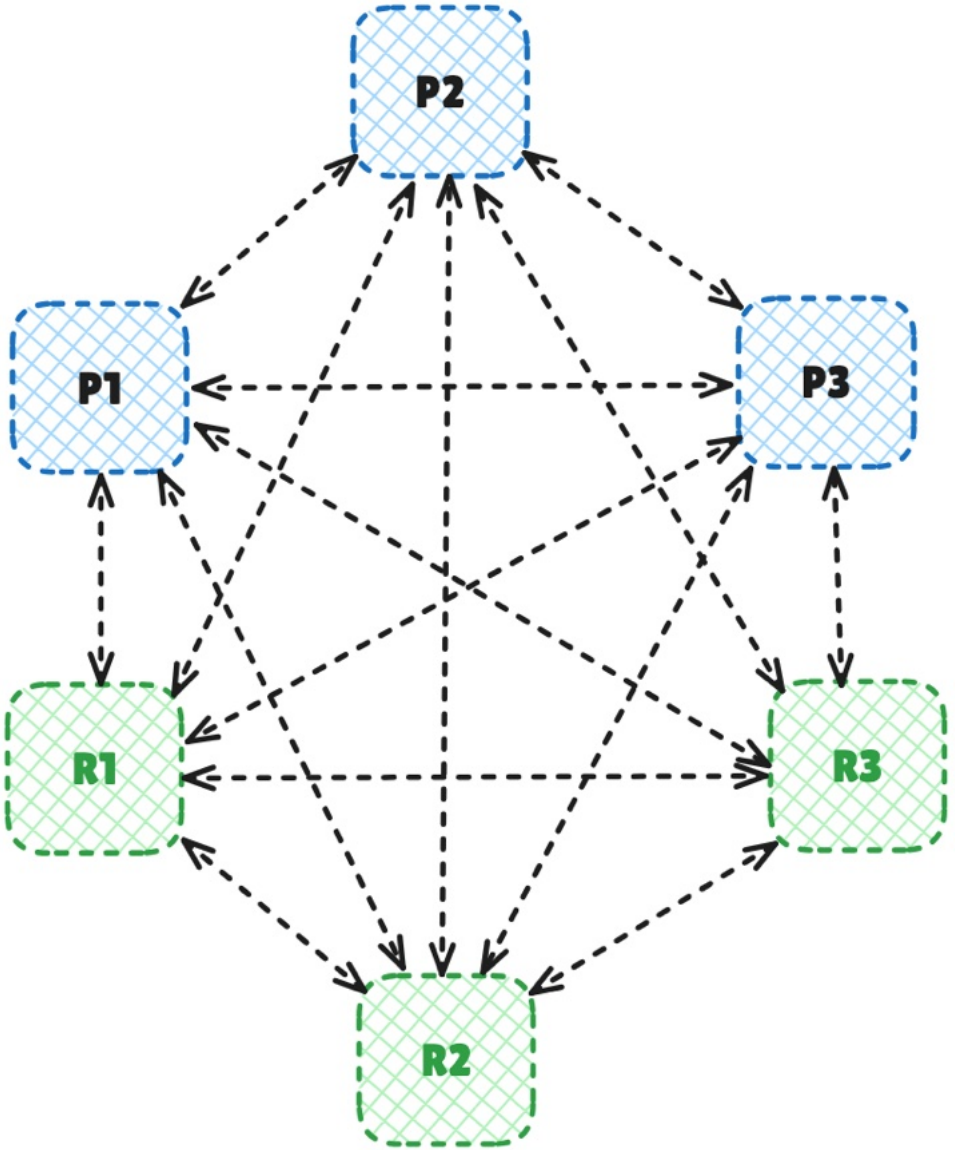
Valkey

Failure Detection - Broadcast



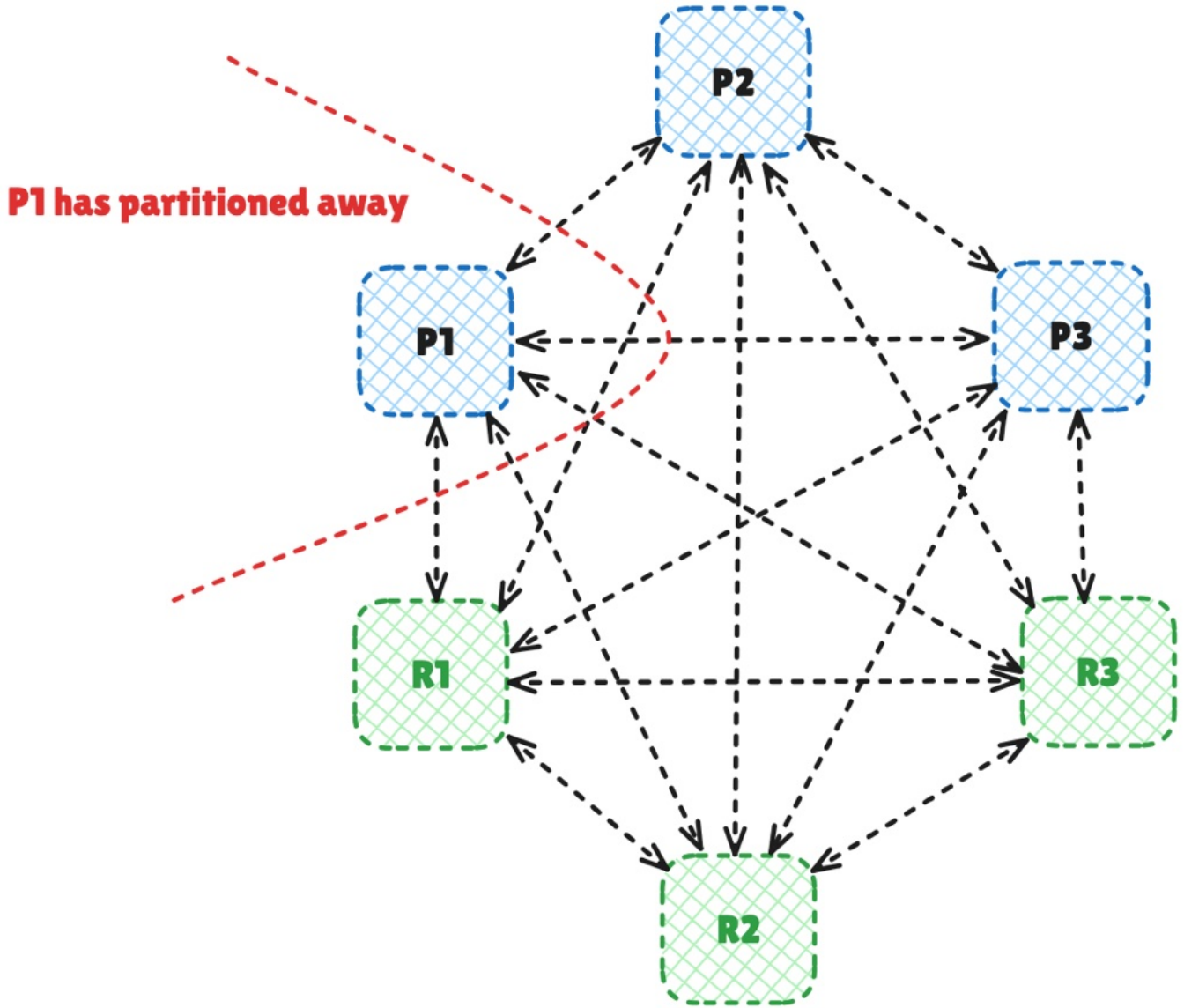
Valkey

Failover



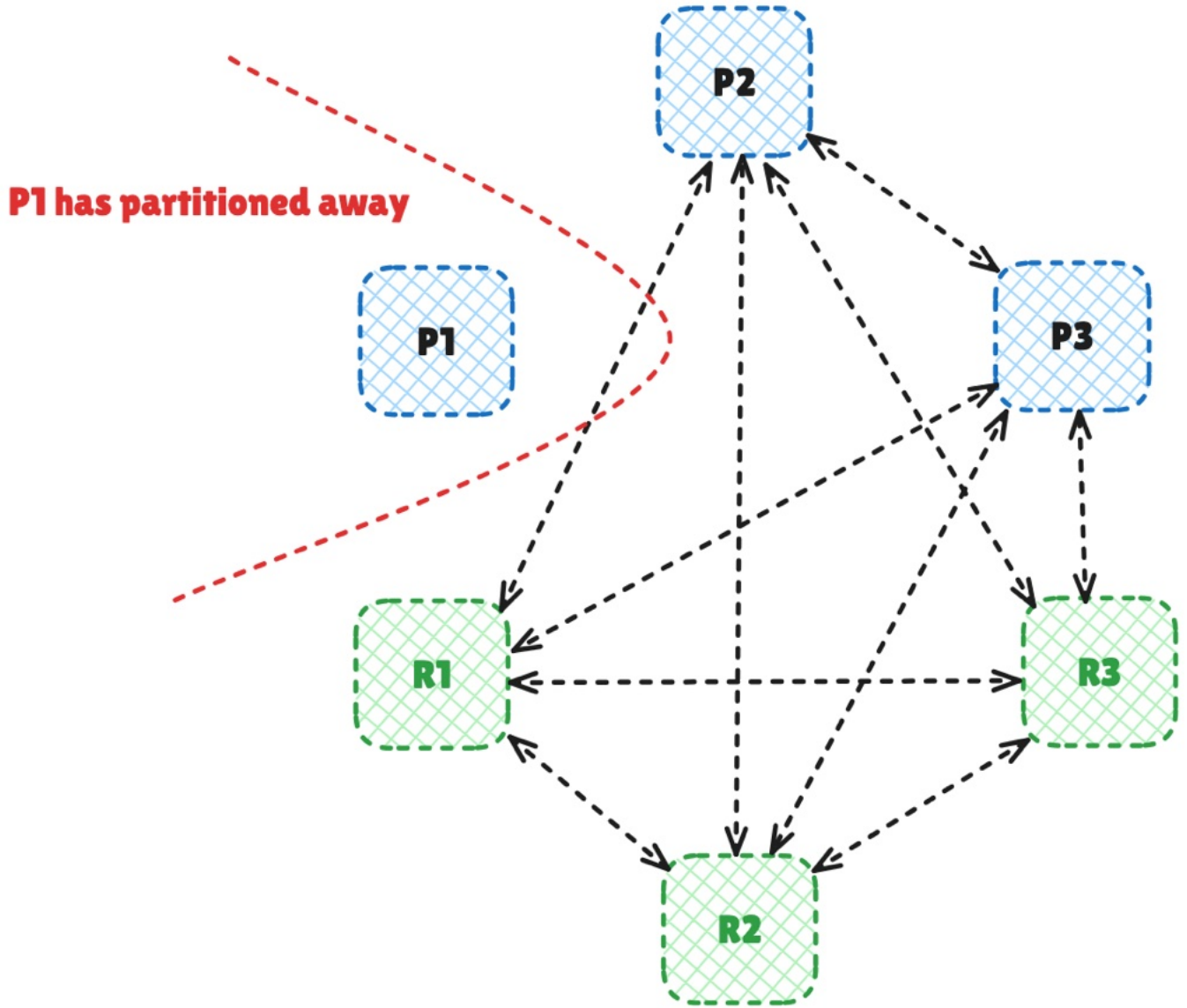
Valkey

Failover – Failure detection



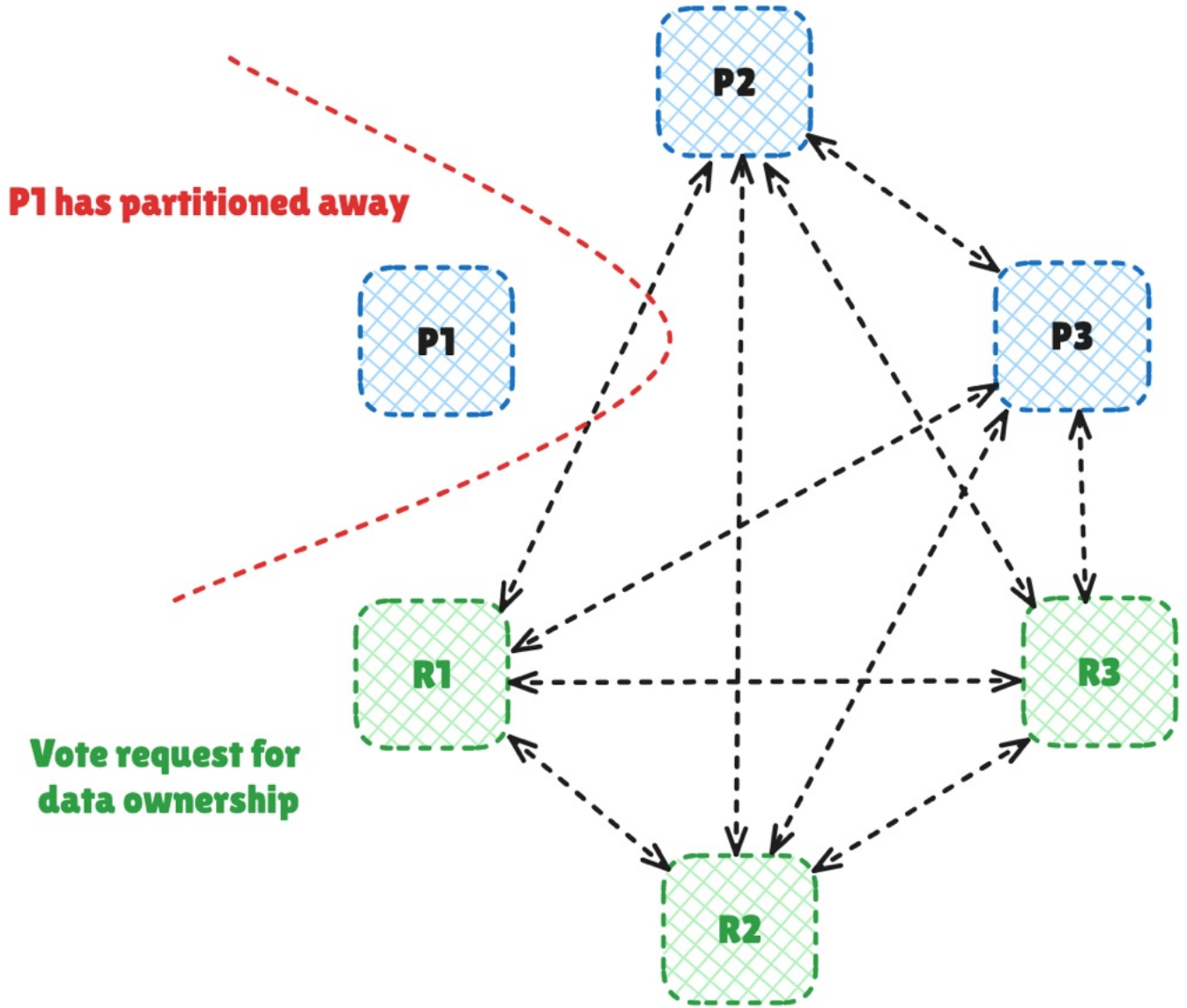
Valkey

Failover – Link disconnection



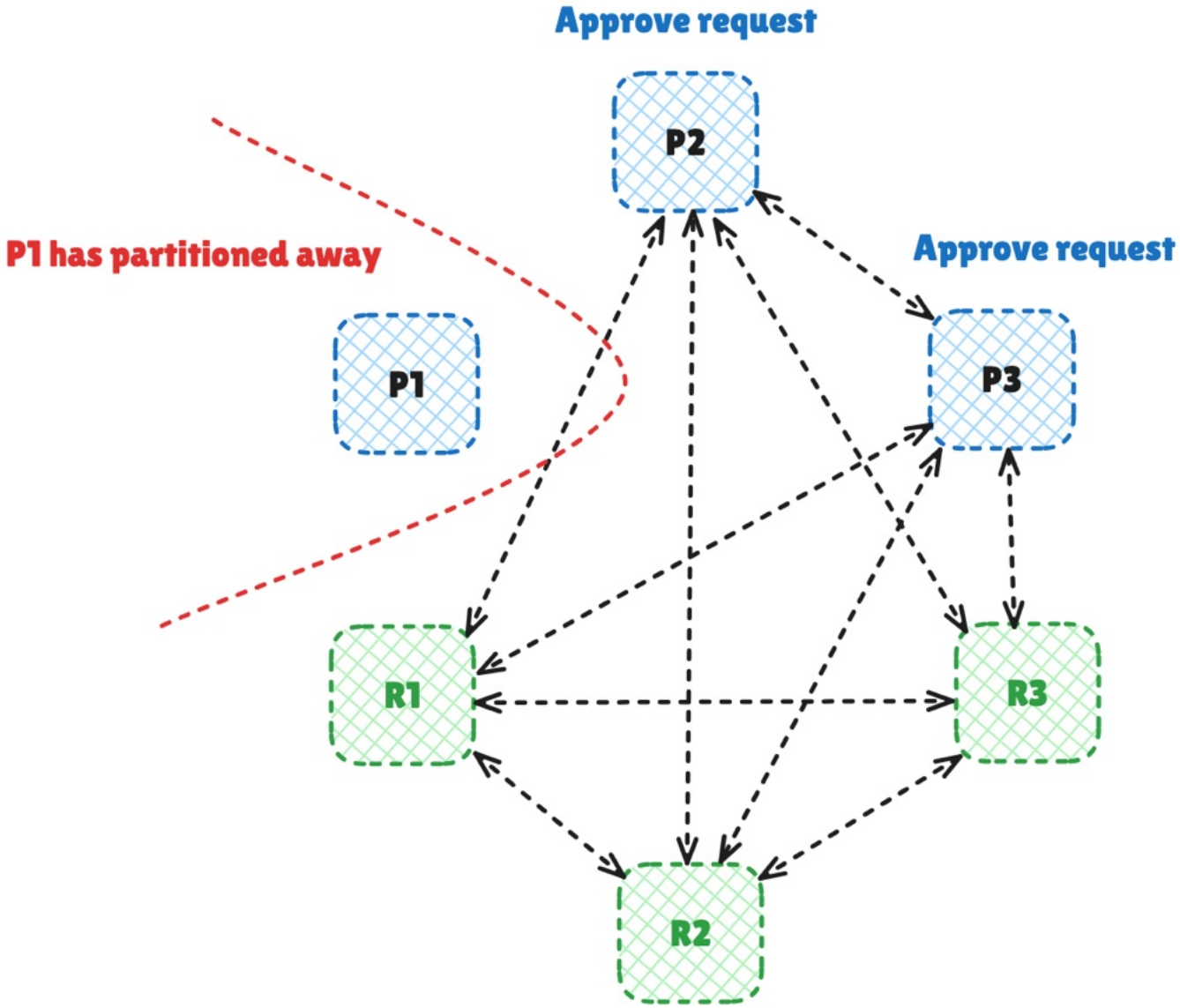
Valkey

Failover – Replica promotion request



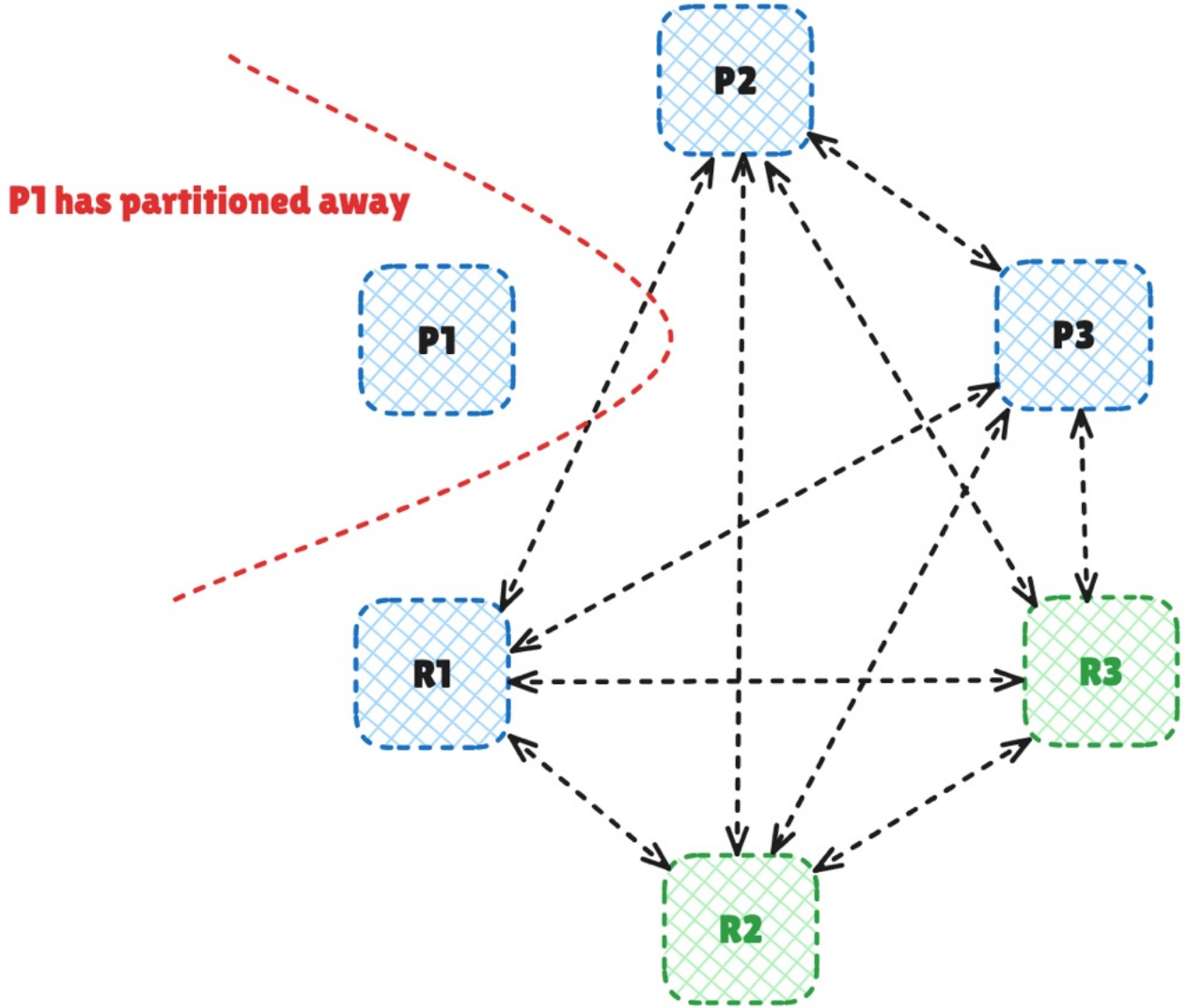
Valkey

Failover – Replica promotion request



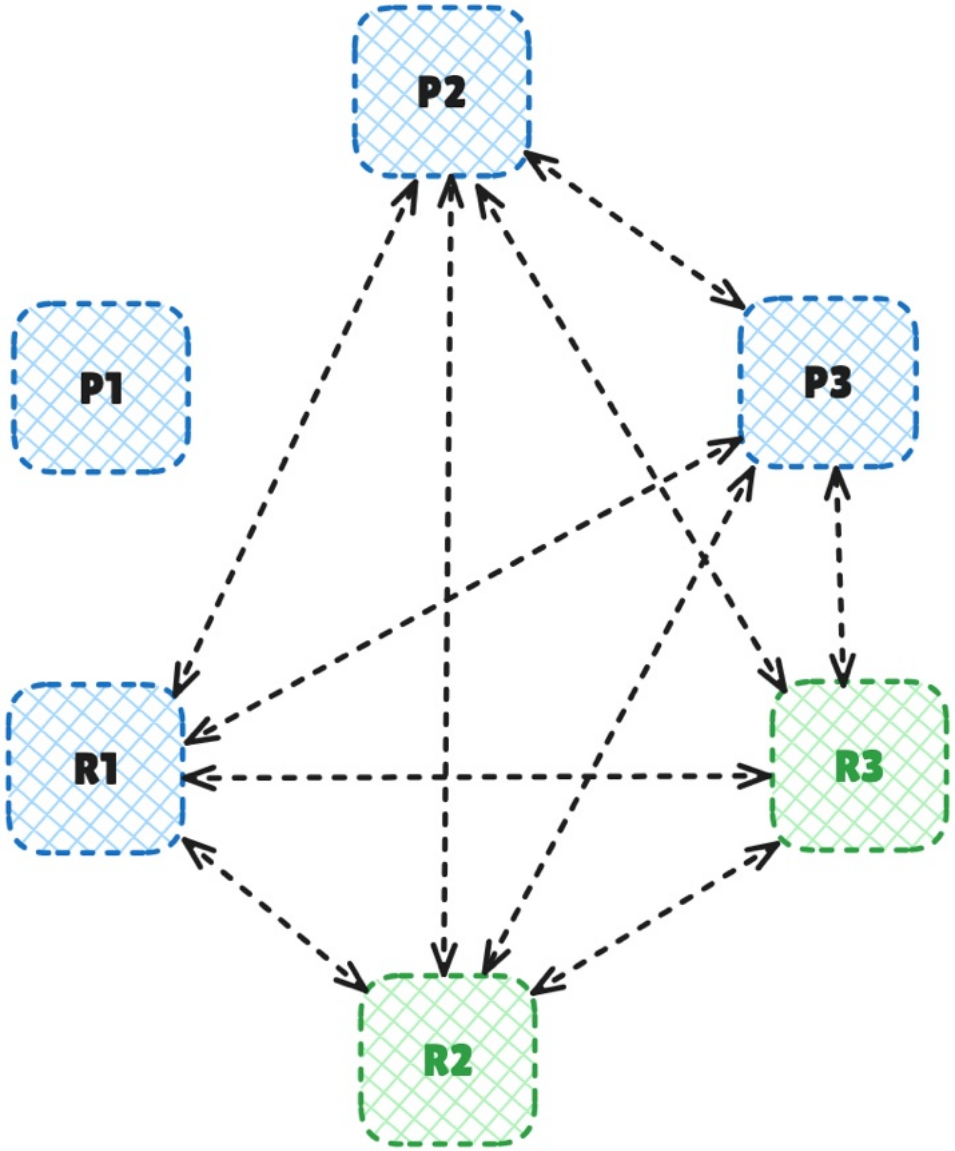
Valkey

Failover – Replica promoted



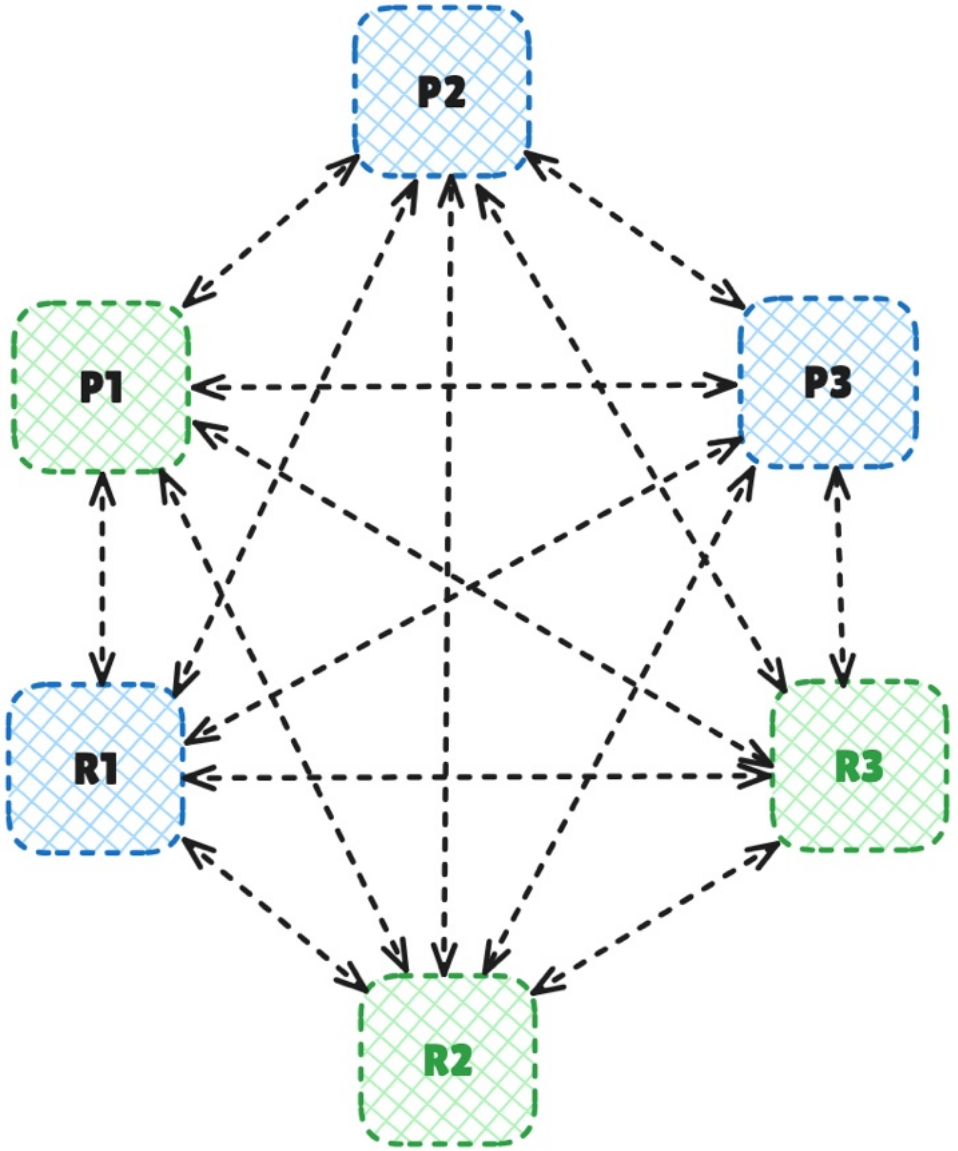
Valkey

Failover – Partition heal start



Valkey

Failover – New topology



Valkey

Conflict Resolution - Epoch

- Decentralized cluster – no single source of truth
- Monotonically increasing counter
- Conflict resolution
  - config epoch, node id to break ties safely

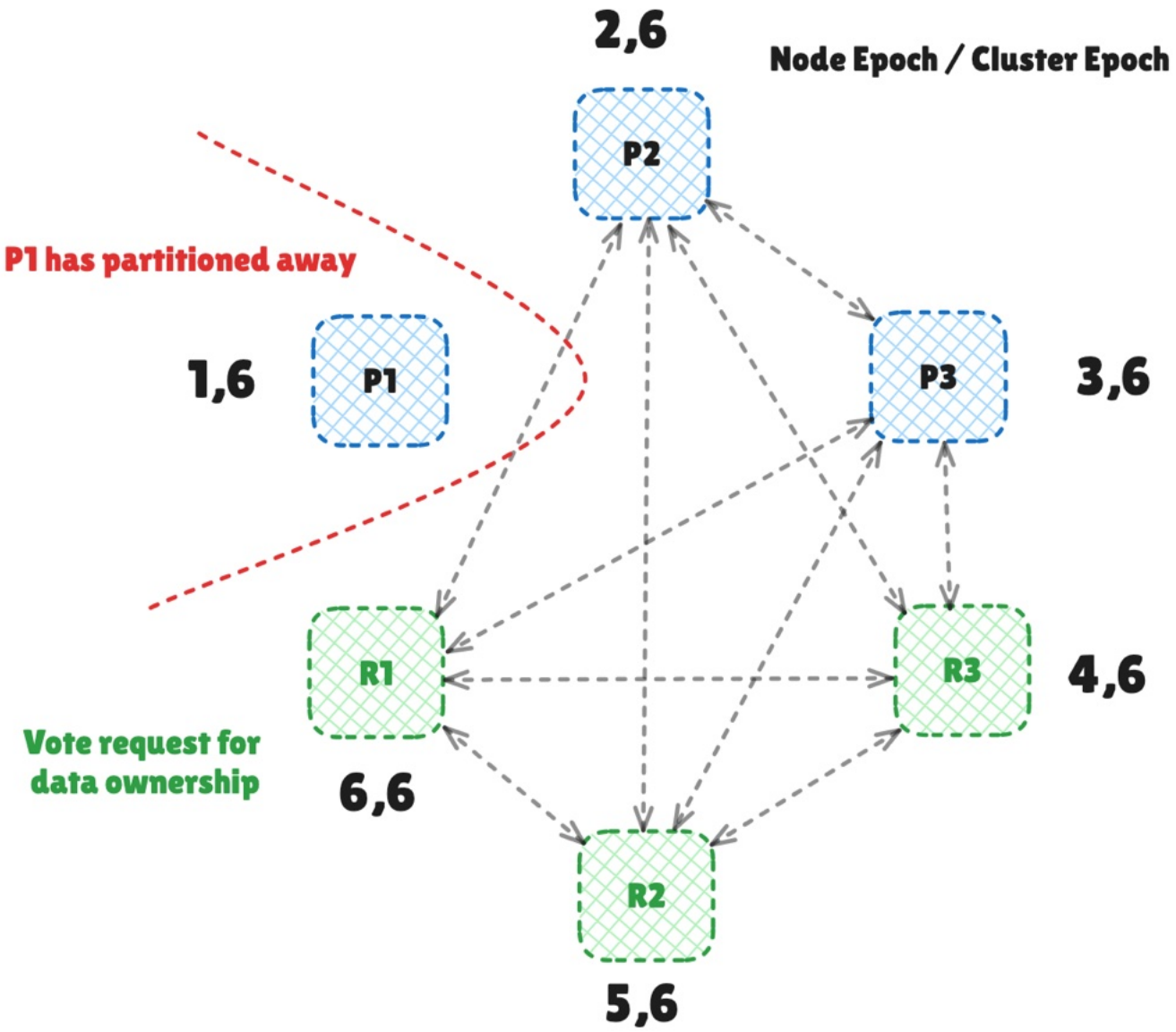
Valkey

Conflict Resolution – Epoch Types

- Node level epoch – config epoch
  - Per node epoch indicating **authority over slots**
- Cluster wide epoch – cluster epoch
  - Global for the **entire cluster**
  - Incremented during failovers to assign unique epochs

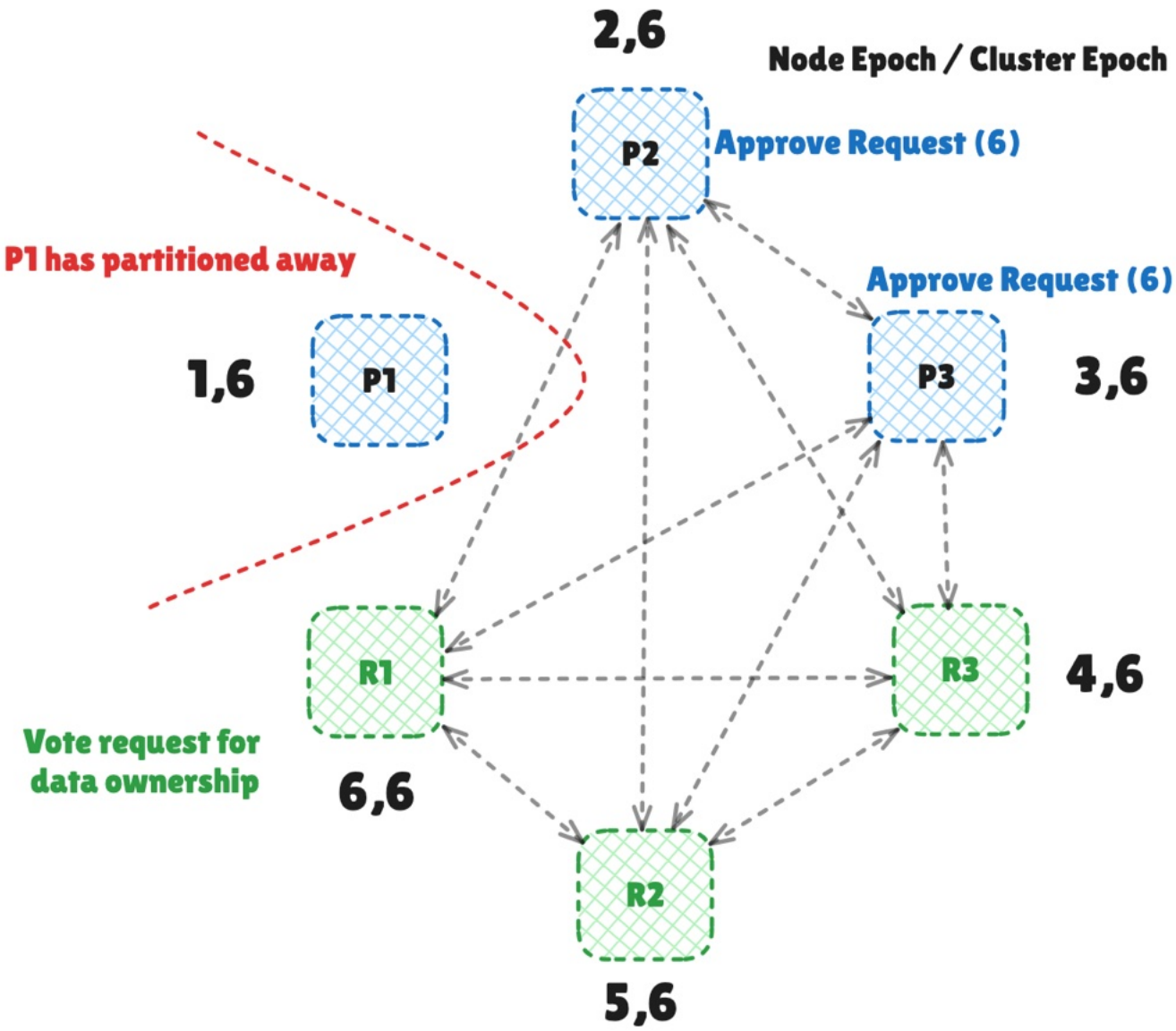
Valkey

Conflict Resolution - Epoch



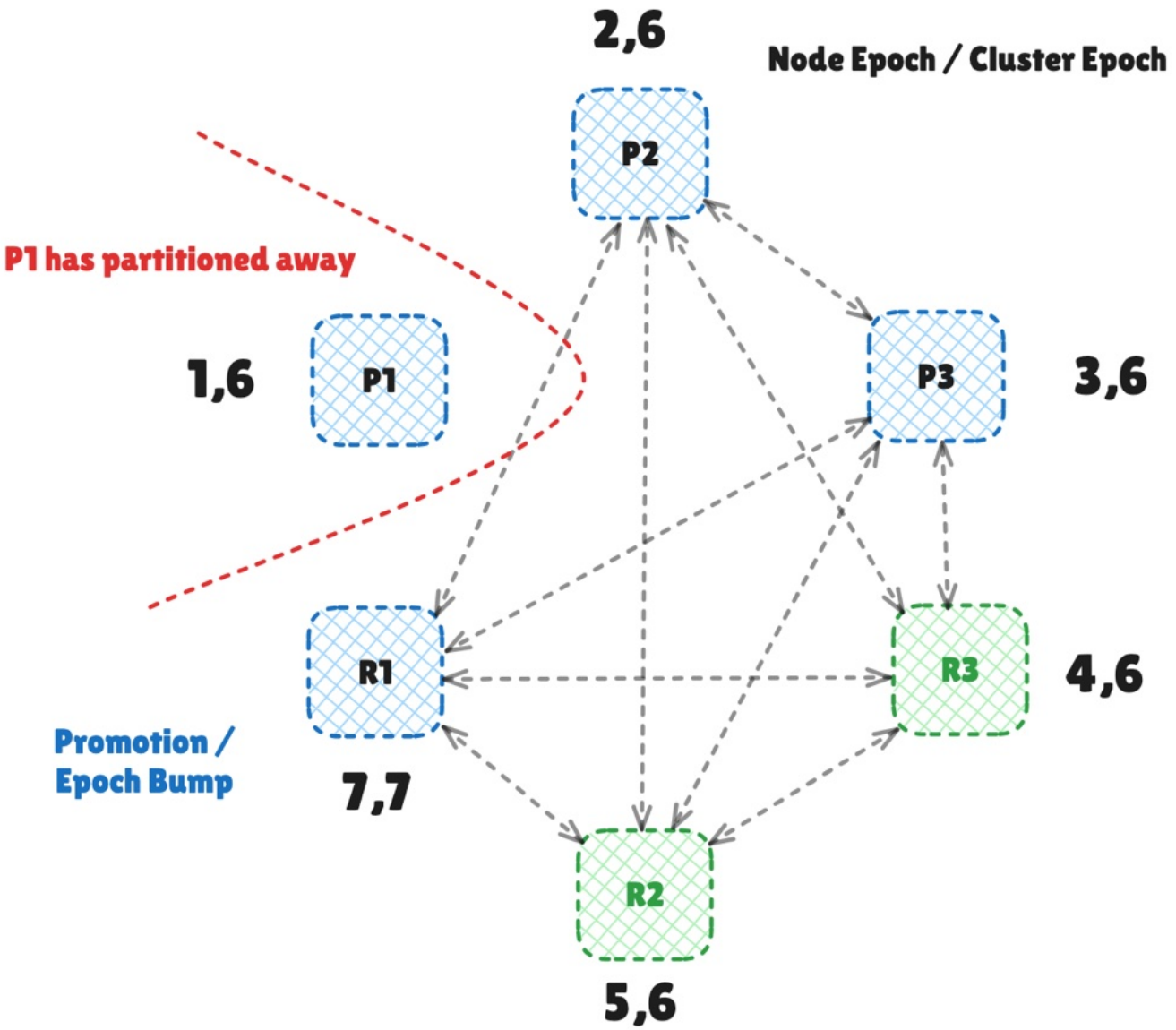
Valkey

Epoch – Vote Request



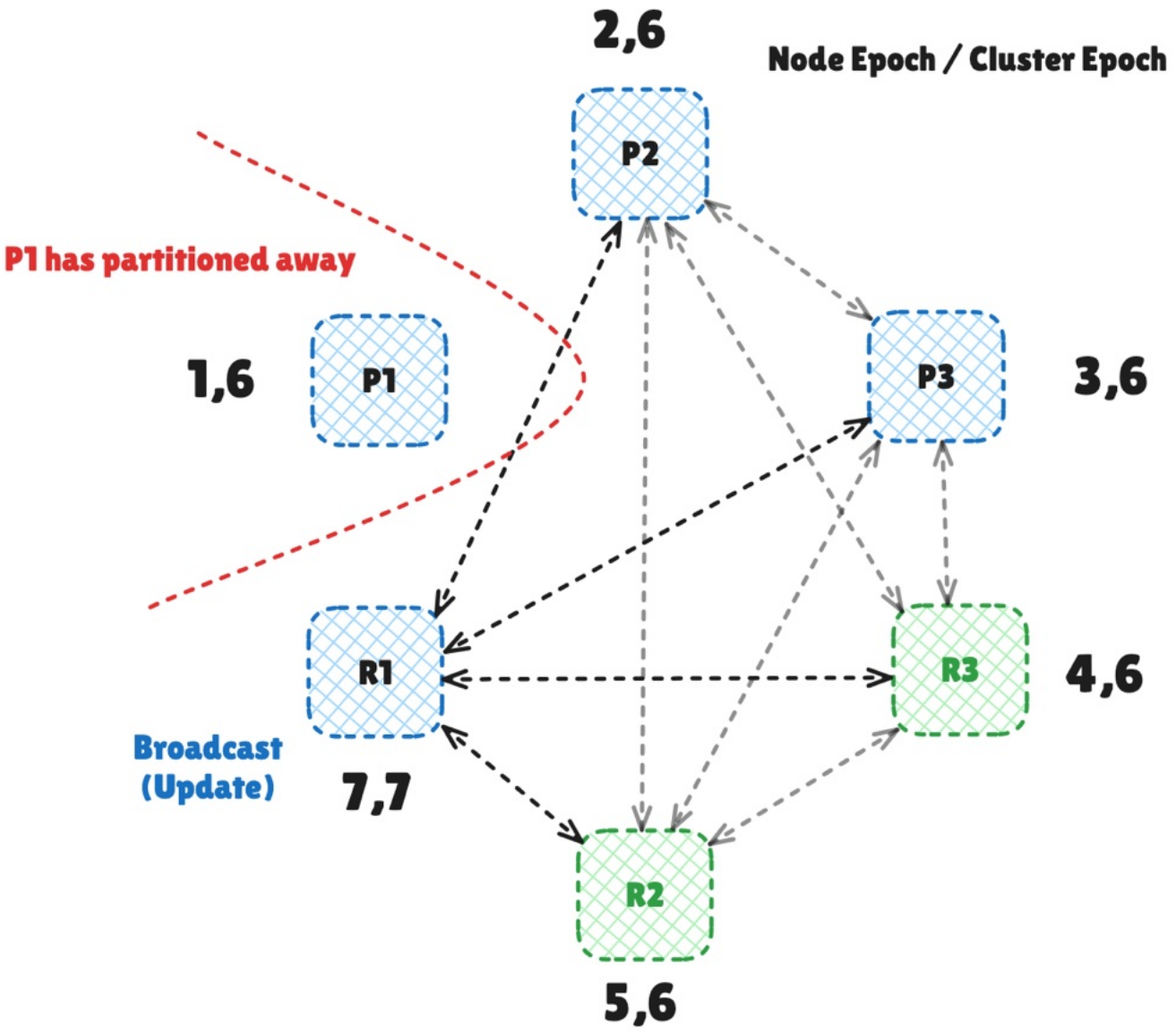
Valkey

Epoch Bump



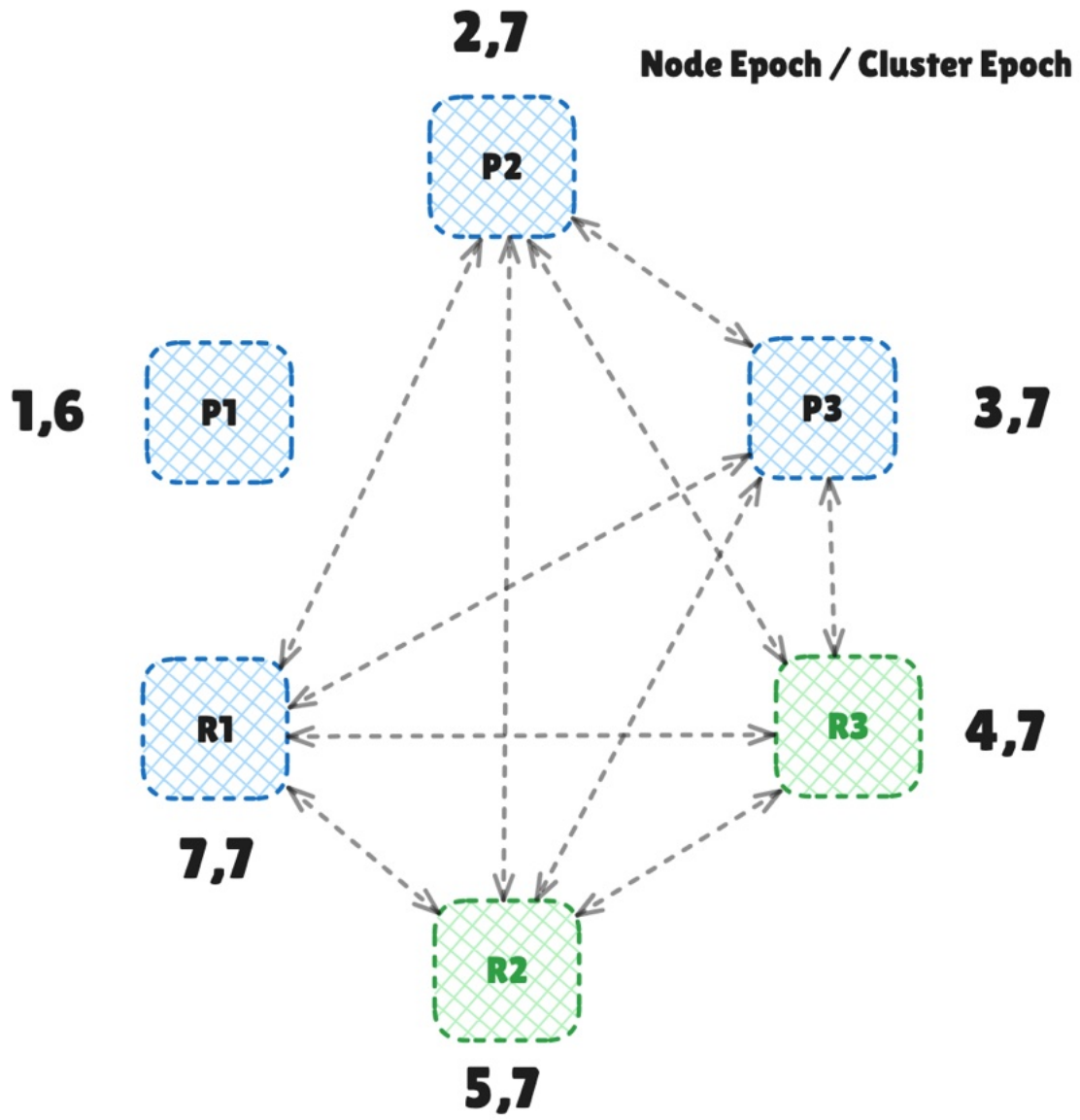
Valkey

Epoch Bump - Broadcast



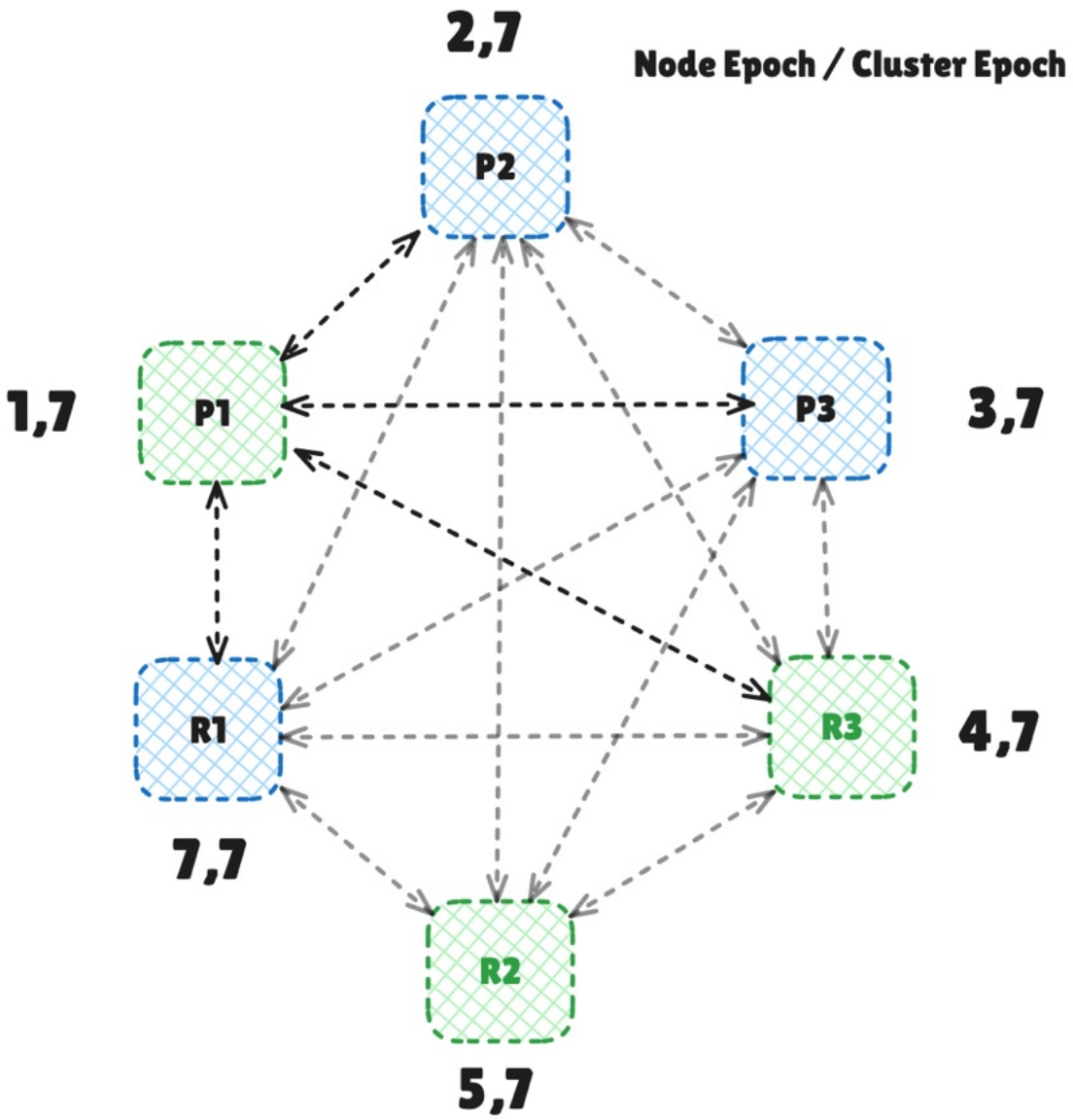
Valkey

Epoch Bump – New Topology



Valkey

Epoch Bump – Conflict Resolution



Valkey

Valkey Clustering improvements

- Fast failover with multiple primary failures
- Light message header type (~30 bytes)
- Connection rate limiting

Valkey

Can it scale?

- Peer to peer health detection, does it scale?
- Message transfer rate is high during ideal state ?
- Too many votes (quorum of primaries) for failover ?

Valkey

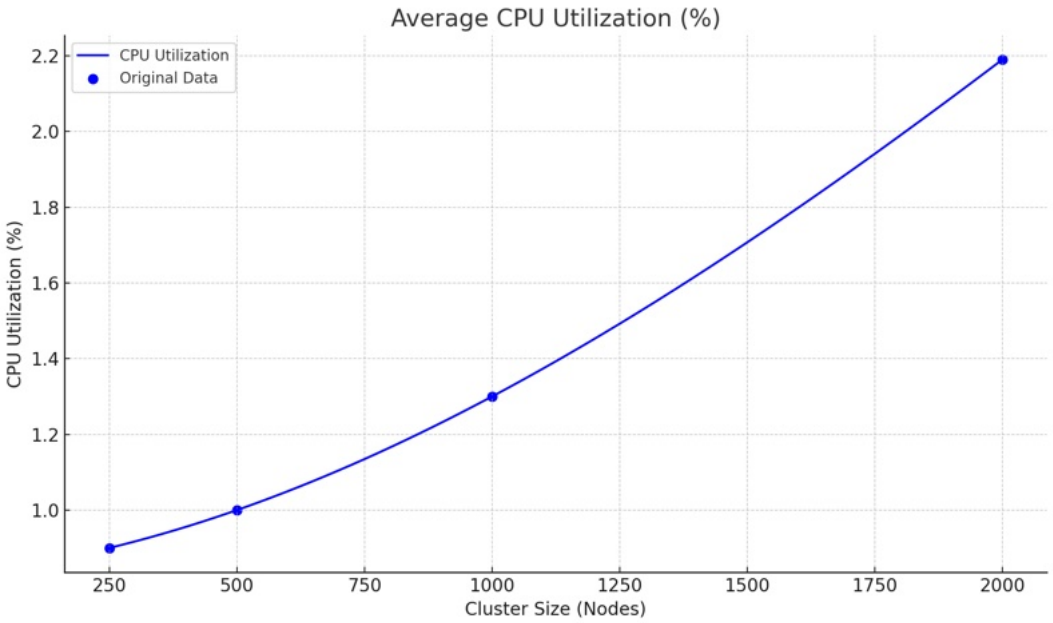
Benchmark - Results

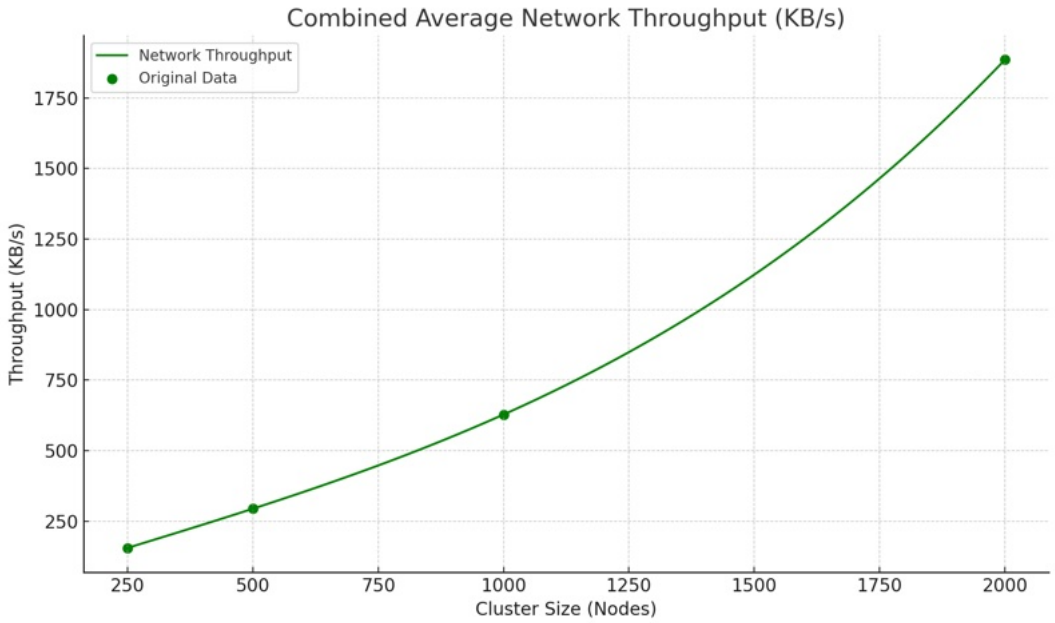
- Scales well upto 2000 nodes cluster
- Node timeout - 15 seconds
- 10% node failure detection and failover time ~ 20 seconds

```
127.0.0.1:6379> CLUSTER INFO
cluster_state:ok
cluster_slots_assigned:16384
cluster_slots_ok:16384
cluster_slots_pfail:0
cluster_slots_fail:0
cluster_nodes_pfail:0
cluster_nodes_fail:0
cluster_voting_nodes_pfail:0
cluster_voting_nodes_fail:0
cluster_known_nodes:1998
cluster_size:666
cluster_current_epoch:1997
cluster_my_epoch:180
cluster_stats_messages_ping_sent:1442617
cluster_stats_messages_pong_sent:1461312
cluster_stats_messages_meet_sent:671
cluster_stats_messages_sent:2904600
cluster_stats_messages_ping_received:1457438
cluster_stats_messages_pong_received:1448050
cluster_stats_messages_meet_received:934
cluster_stats_messages_received:2906422
total_cluster_links_buffer_limit_exceeded:0
127.0.0.1:6379>
```

Valkey

Benchmark - Results





Valkey

What’s next?

- Reduce steady state CPU utilization
- Offload cluster message processing
- Additional observability information
- Fuzzy testing

Valkey

Get involved in the project

valkey.io

valkey/valkey

github.com/valkey-io

Valkey

Thank you!

Harkrishn Patro
Software Engineer – AWS
github.com/hpatro
linkedin.com/in/harkrishn-patro