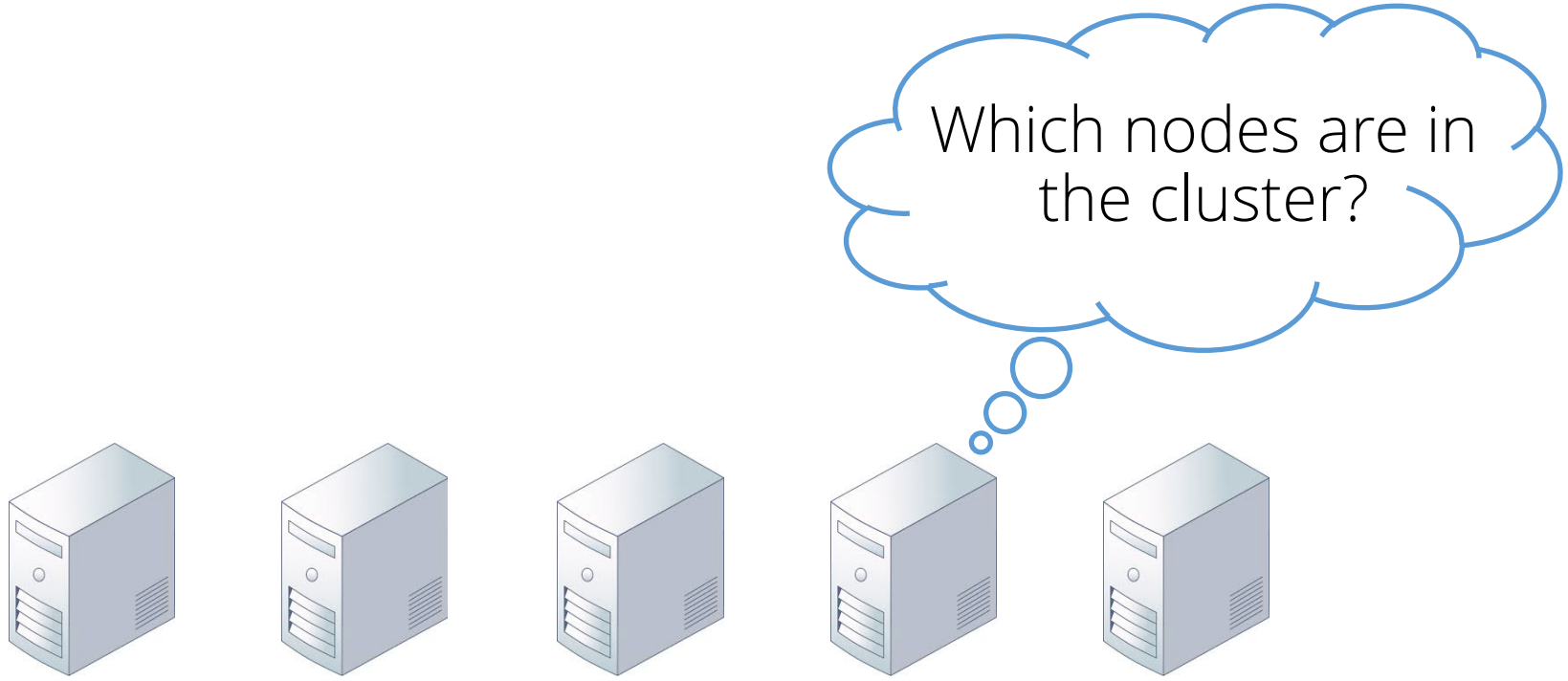
Stable and consistent membership at scale with Rapid

Lalith Suresh, Dahlia Malkhi, Parikshit Gopalan
Ivan Porto Carreiro[1], Zeeshan Lokhandwala[2]

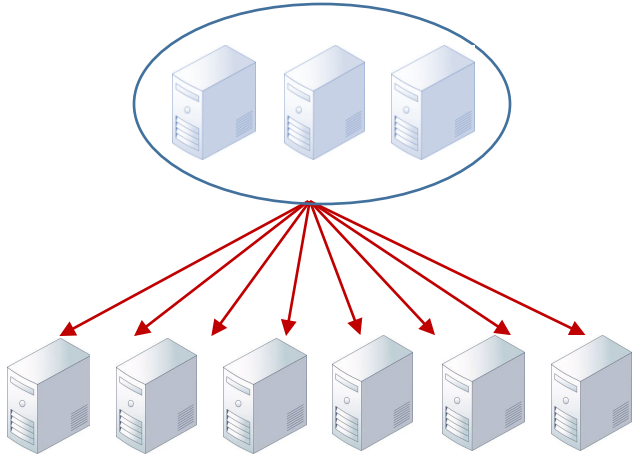
VMware Research [1]VMware [2]One Concern



Membership management and failure detection

Types of membership services

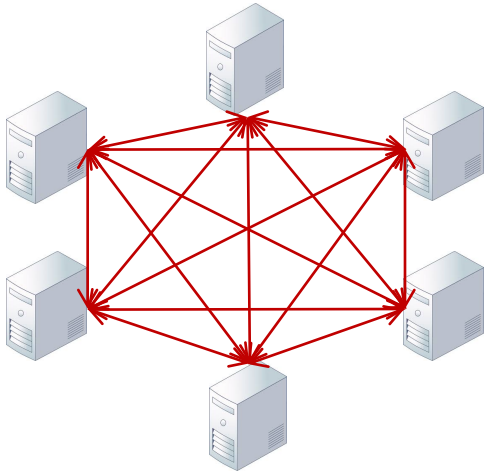
Centralized



Spark, HDFS,

Zookeeper-based systems

Gossip-based

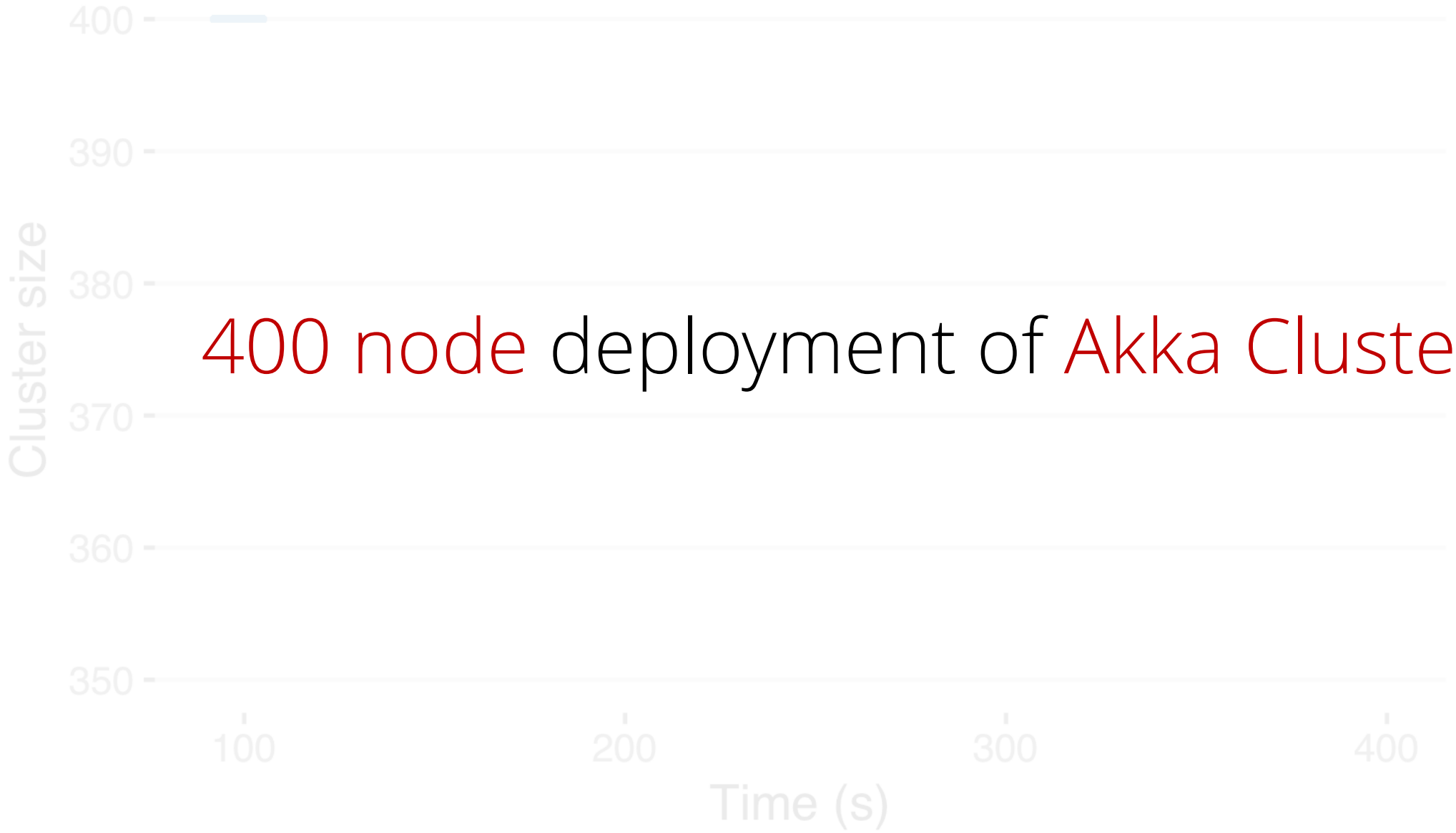


Cassandra, Akka,
Redis Cluster,
Dynamo

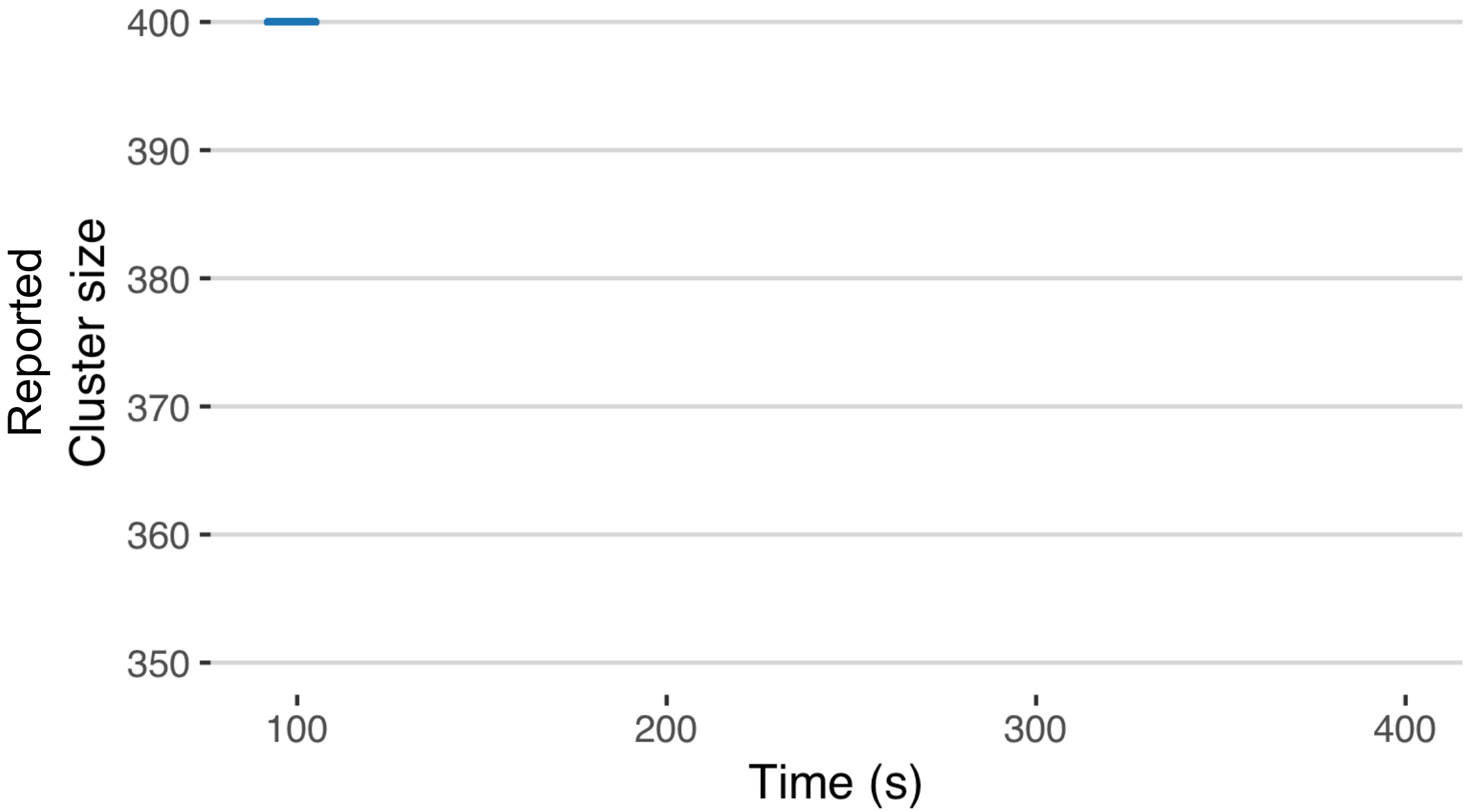
Existing solutions do not provide stability and consistency at scale

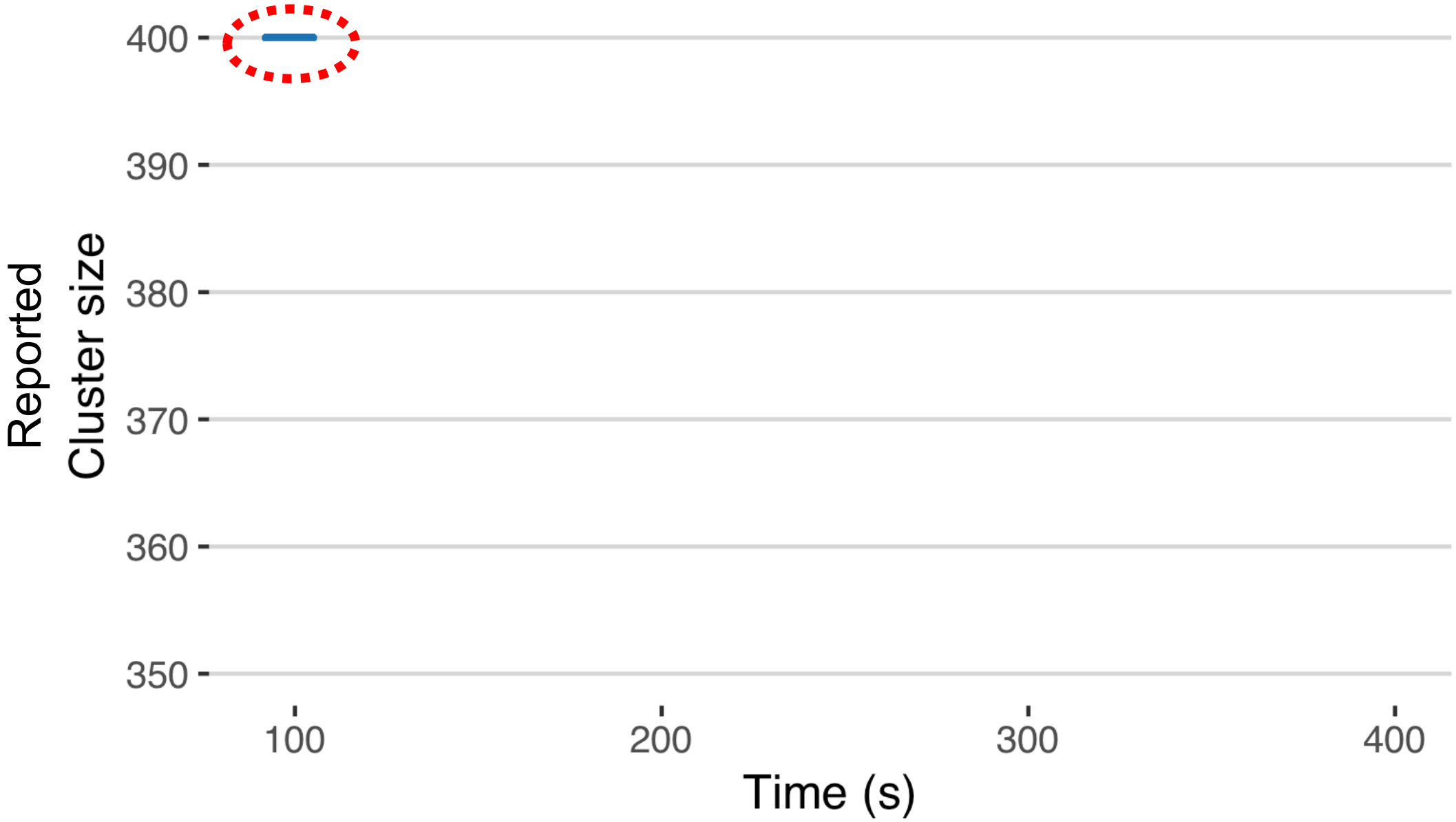
Centralized

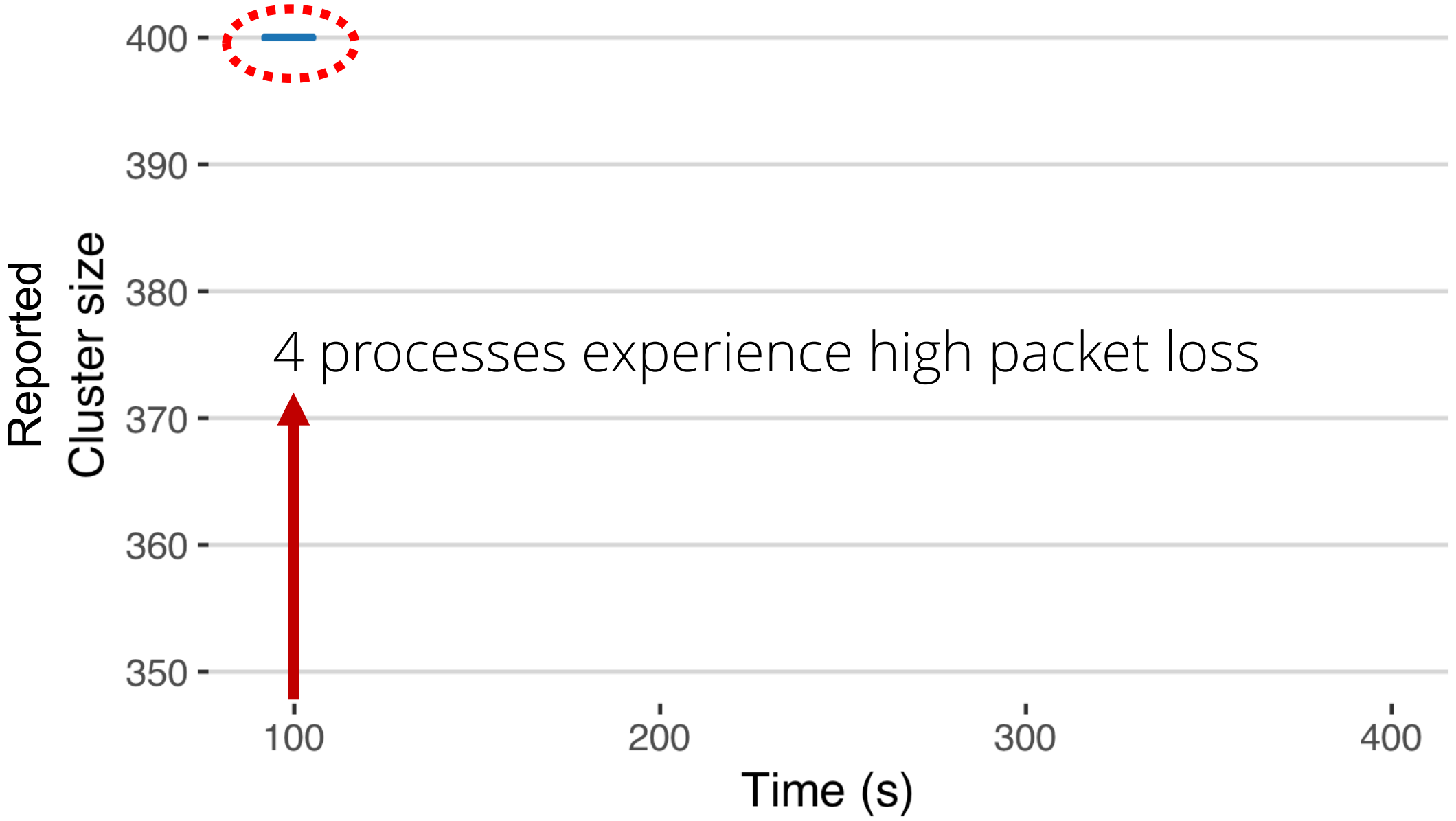
Gossip-based

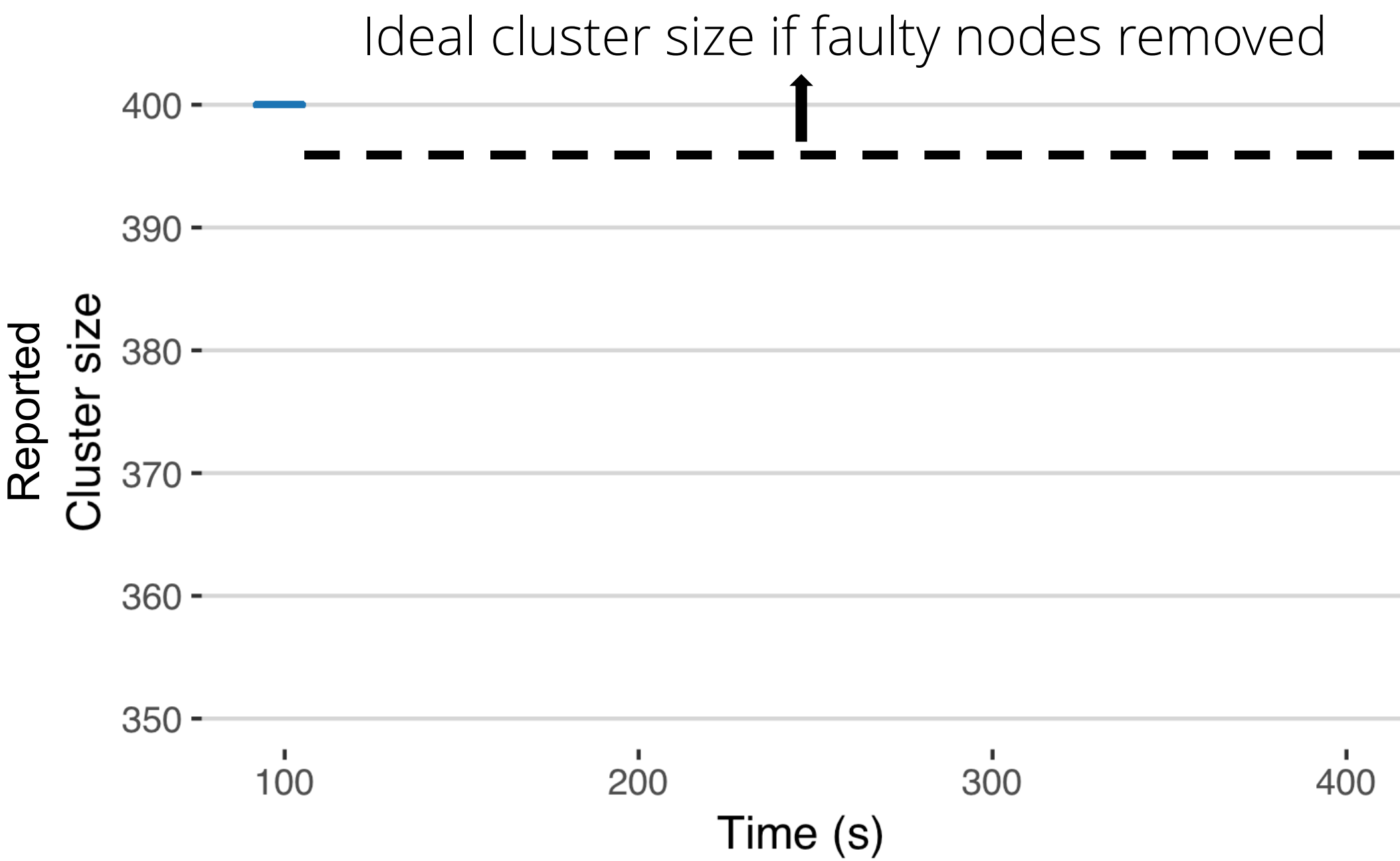


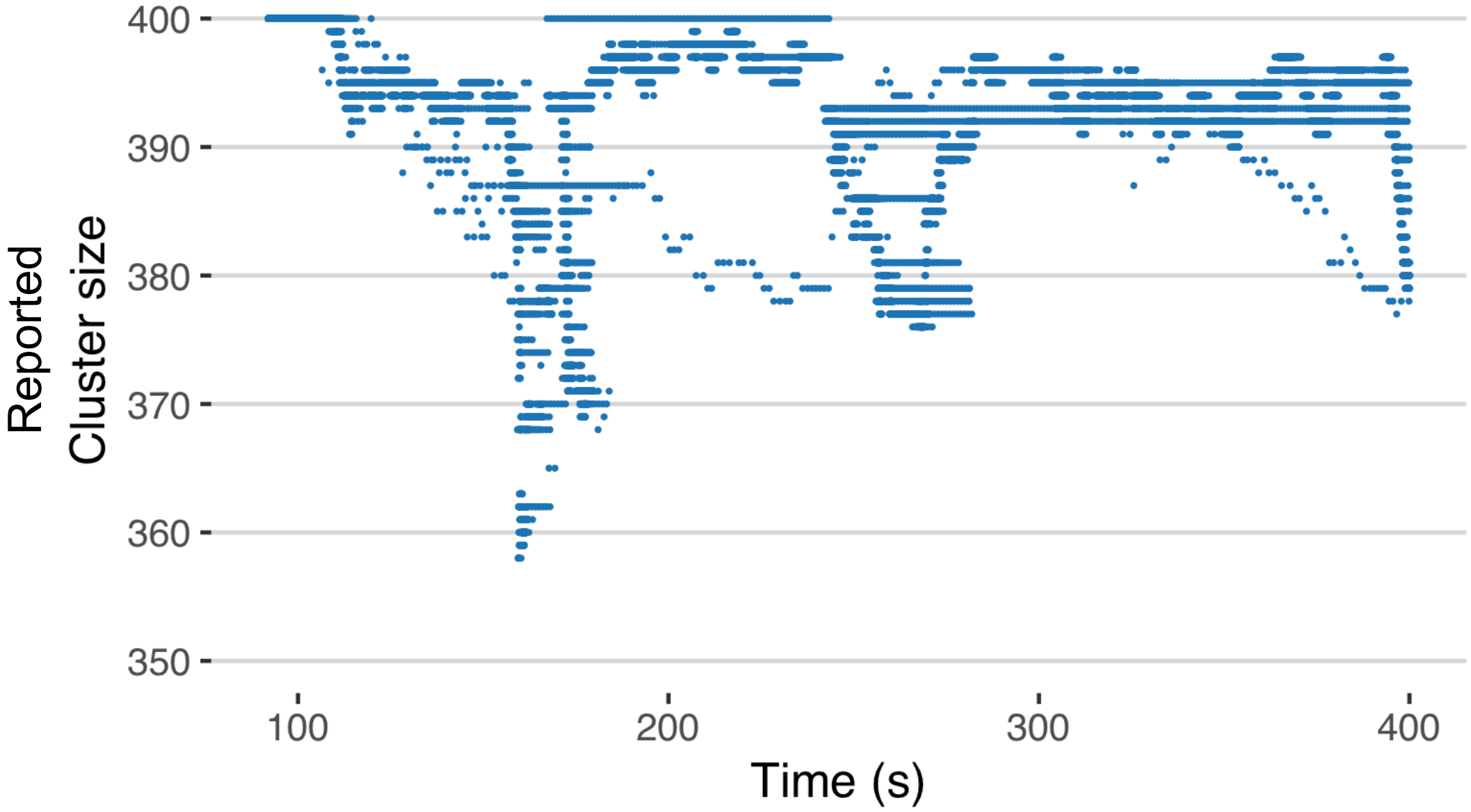
400 node deployment of Akka Cluster

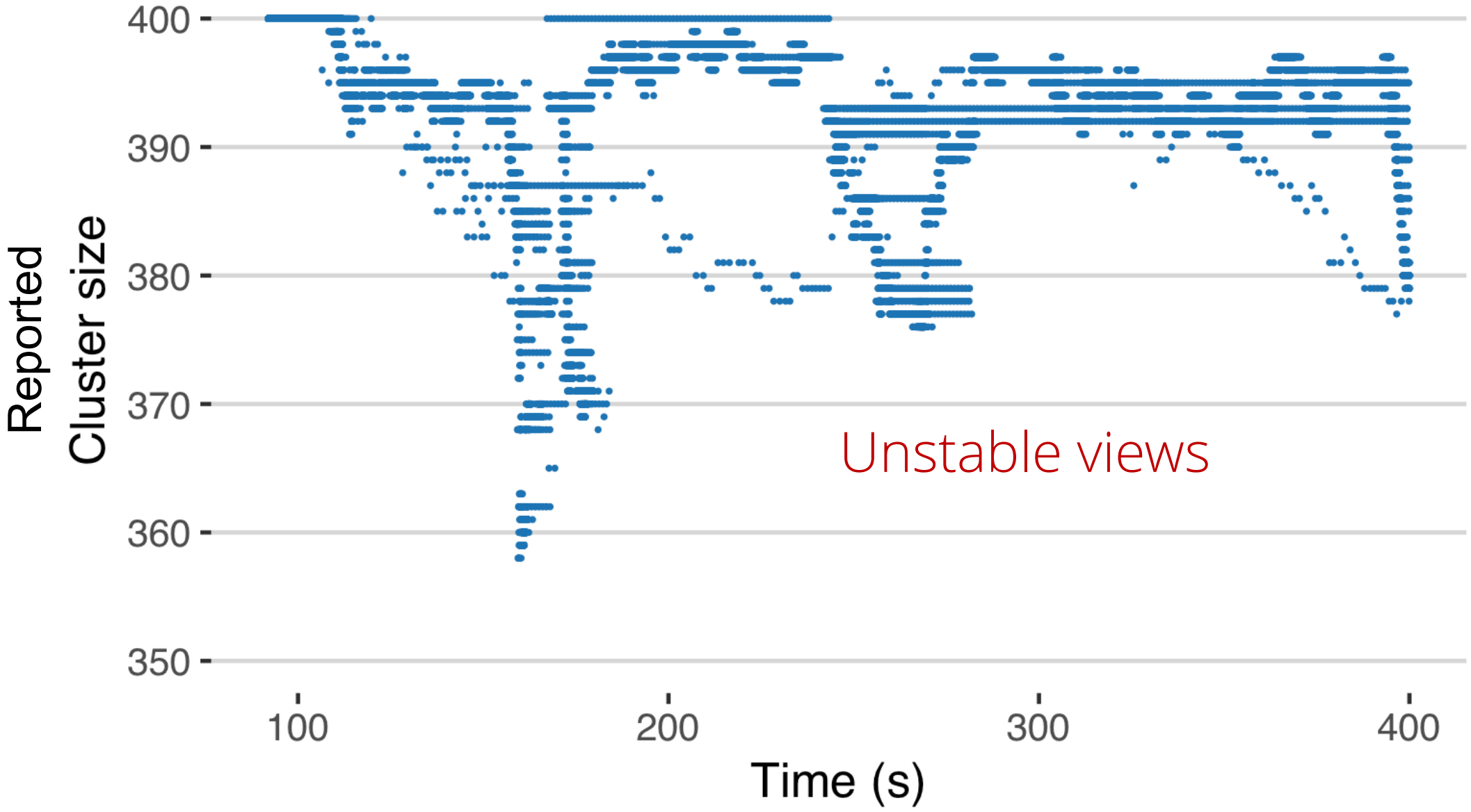

Reported
Cluster size
400
390
380
370
360
350
100
200
300
400
Time (s)

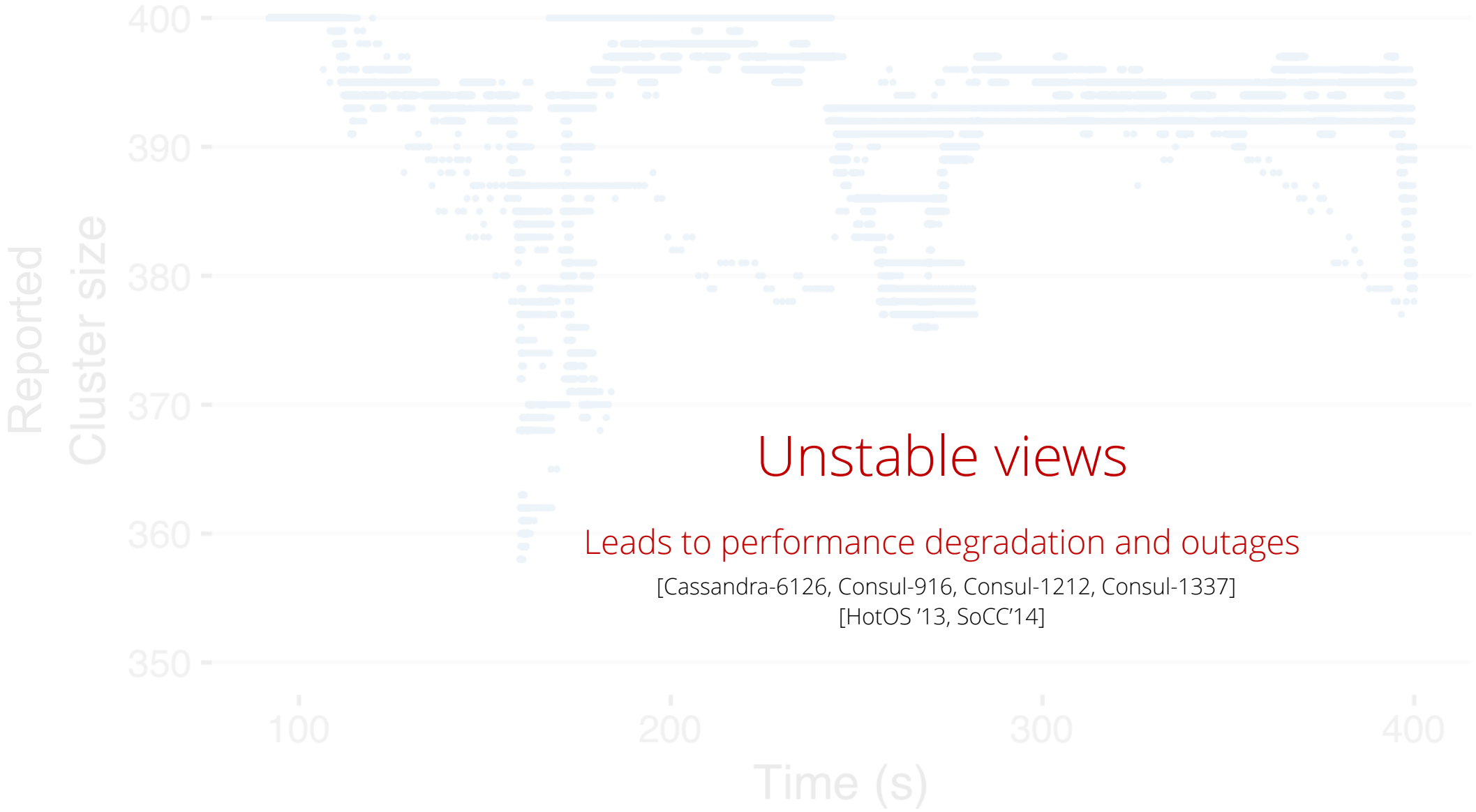

Reported
Cluster size
400
390
380
370
360
350
100
200
300
400
Time (s)


400
390
380
370
360
350
Reported
Cluster size
4 processes experience high packet loss
100
200
300
400
Time (s)


Ideal cluster size if faulty nodes removed
400
390
380
370
360
350
Reported
Cluster size
100
200
300
400
Time (s)


Reported
Cluster size
400
390
380
370
360
350
100
200
300
400
Time (s)

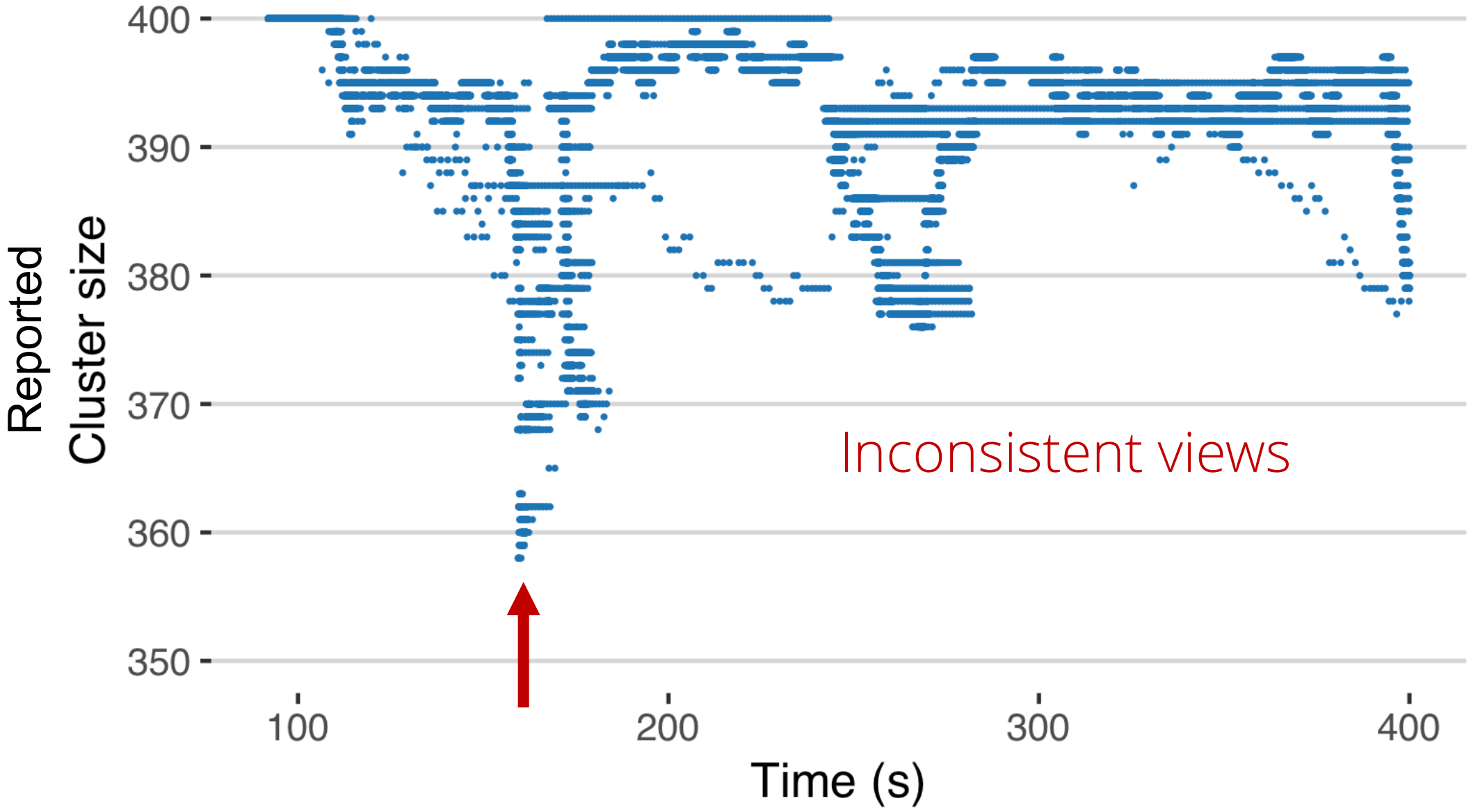

Reported Cluster size
400
390
380
370
360
350
100
200
300
400
Time (s)
Unstable views

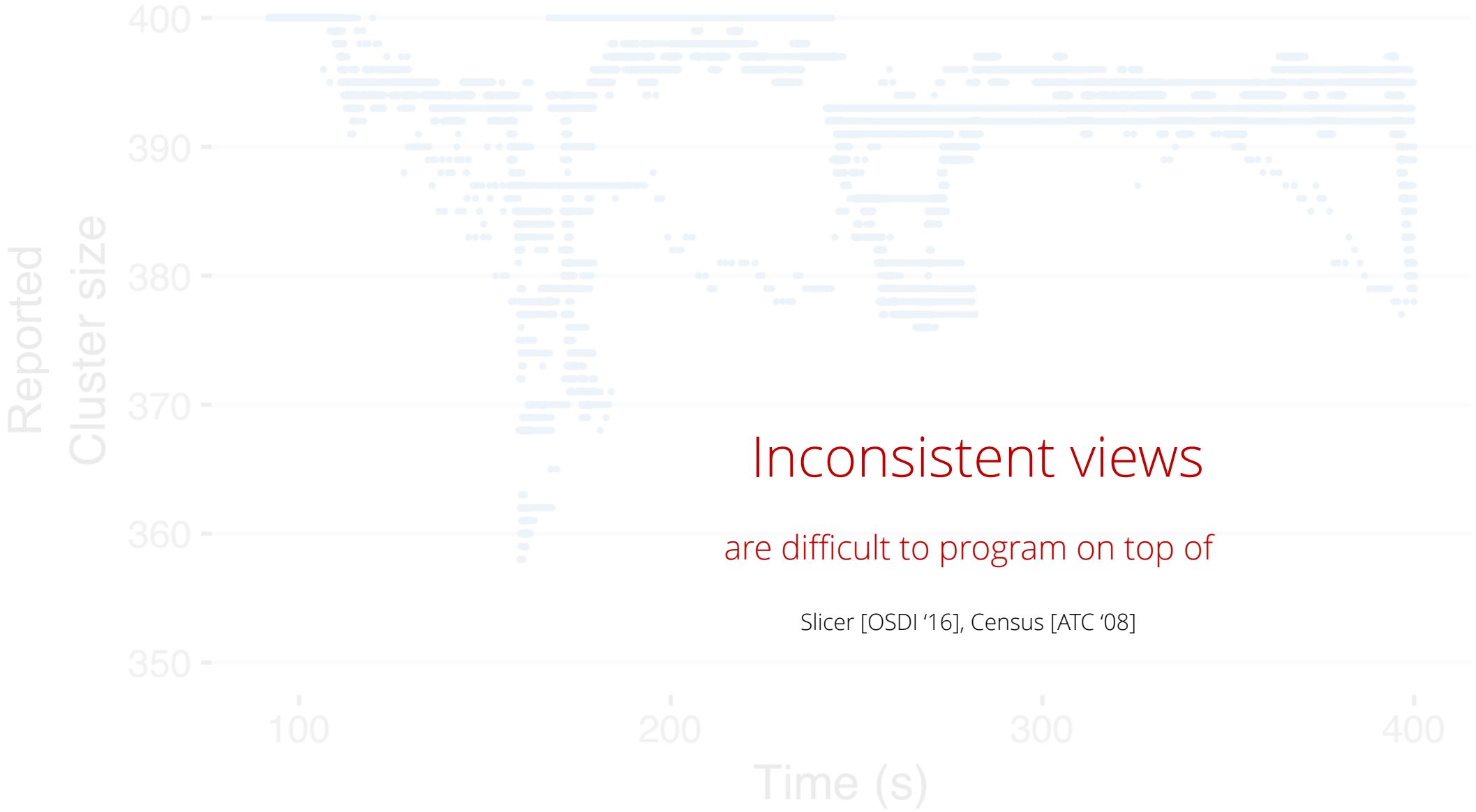


Unstable views

Leads to performance degradation and outages

[Cassandra-6126, Consul-916, Consul-1212, Consul-1337]
[HotOS '13, SoCC'14]


Reported Cluster size
400
390
380
370
360
350
100
200
300
400
Time (s)
Inconsistent views



Inconsistent views

are difficult to program on top of

Slicer [OSDI ‘16], Census [ATC ‘08]

Rapid

Stable and consistent membership at scale

Rapid

Stable and consistent membership at scale

Robust against asymmetric network failures, flip-flops, packet loss etc.

Rapid

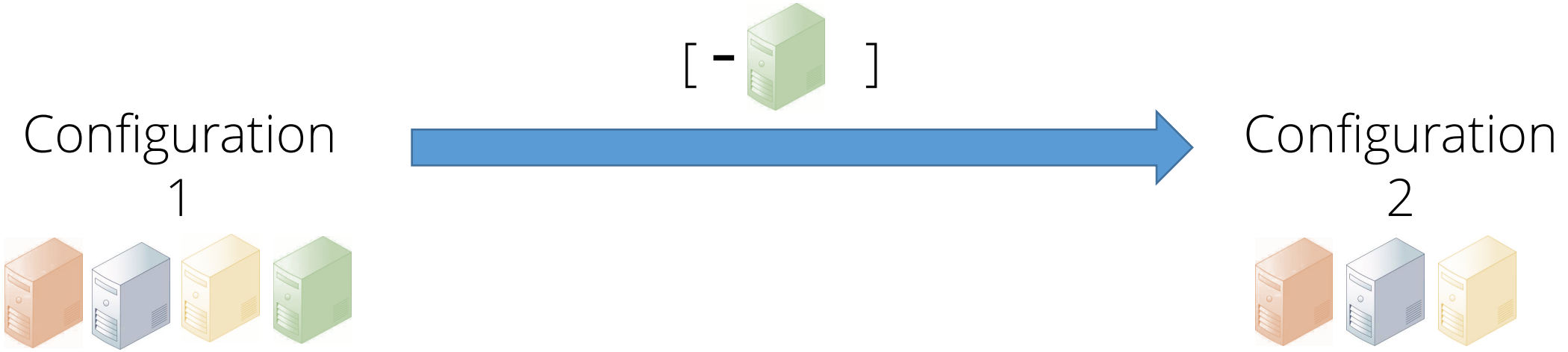
Stable and consistent membership at scale

Processes see the same sequence of membership changes

Rapid

Stable and consistent membership at scale

Bootstraps 2000 nodes 2-5x faster than Zookeeper and Memberlist



Monitoring Overlay → Membership change proposal → View change with consensus

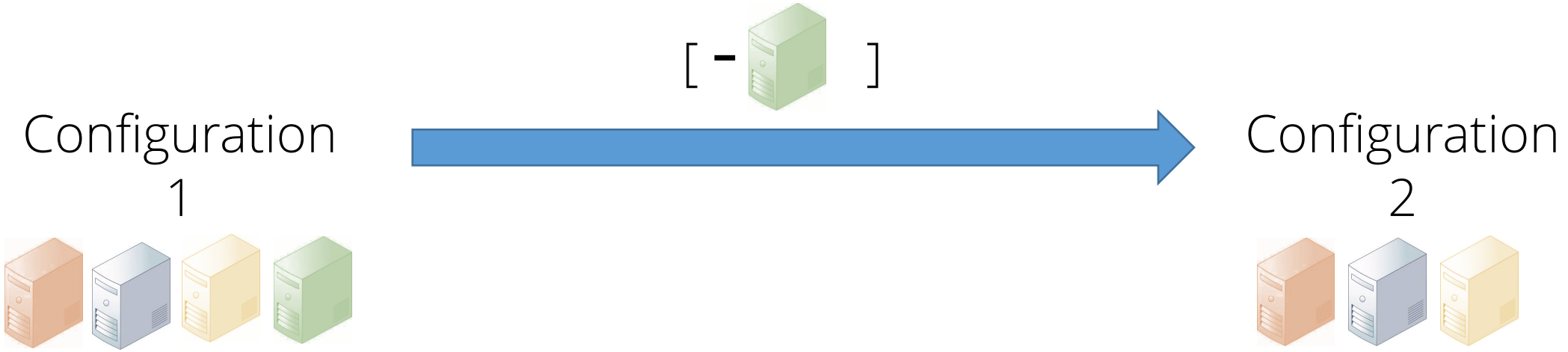
Expander graph-based monitoring

Multi-process cut detection

Fast path to consensus

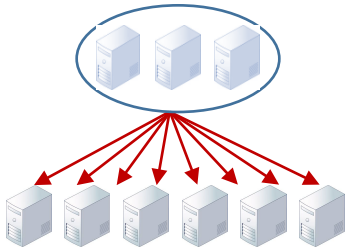
Stability

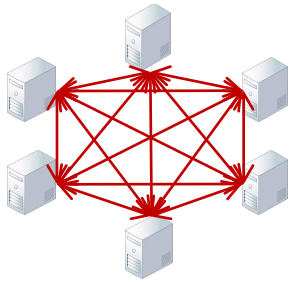
Consistency

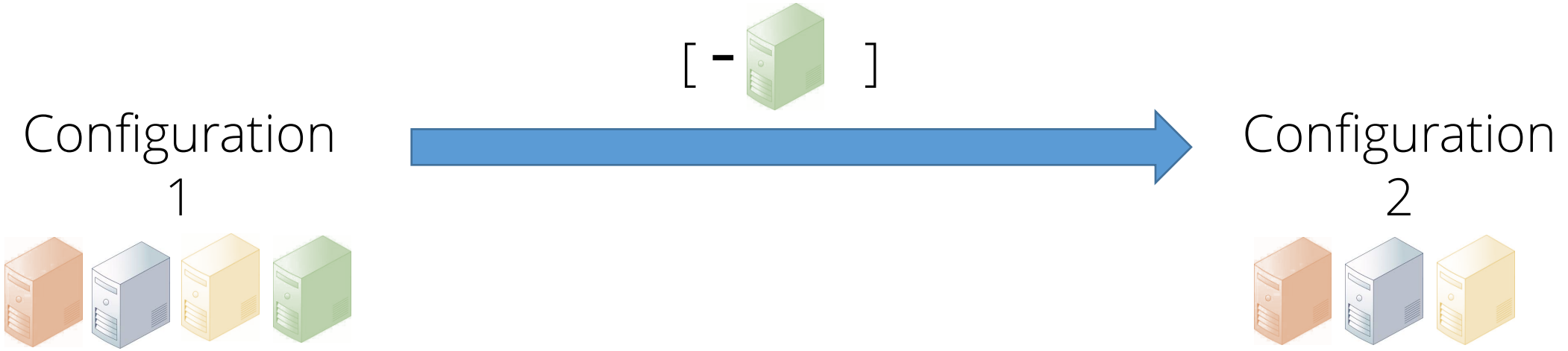


Monitoring Overlay → Membership change proposal → View change with consensus

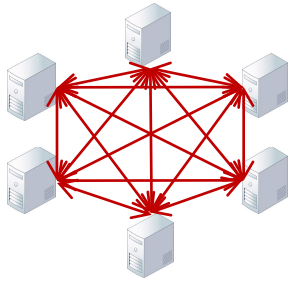
Rapid runs in both centralized and decentralized configurations

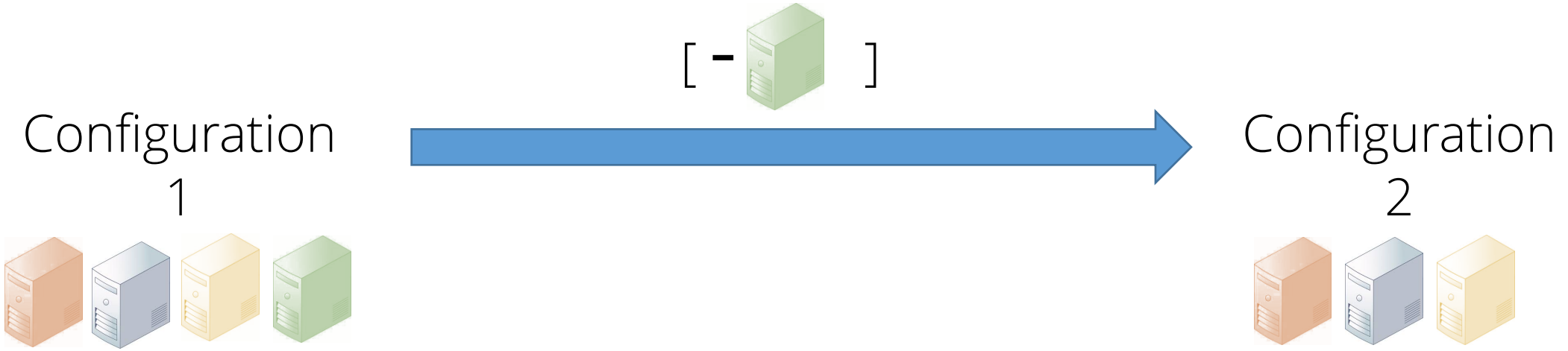






This Talk: decentralized design and failures





Monitoring Overlay

Membership change proposal

View change with consensus

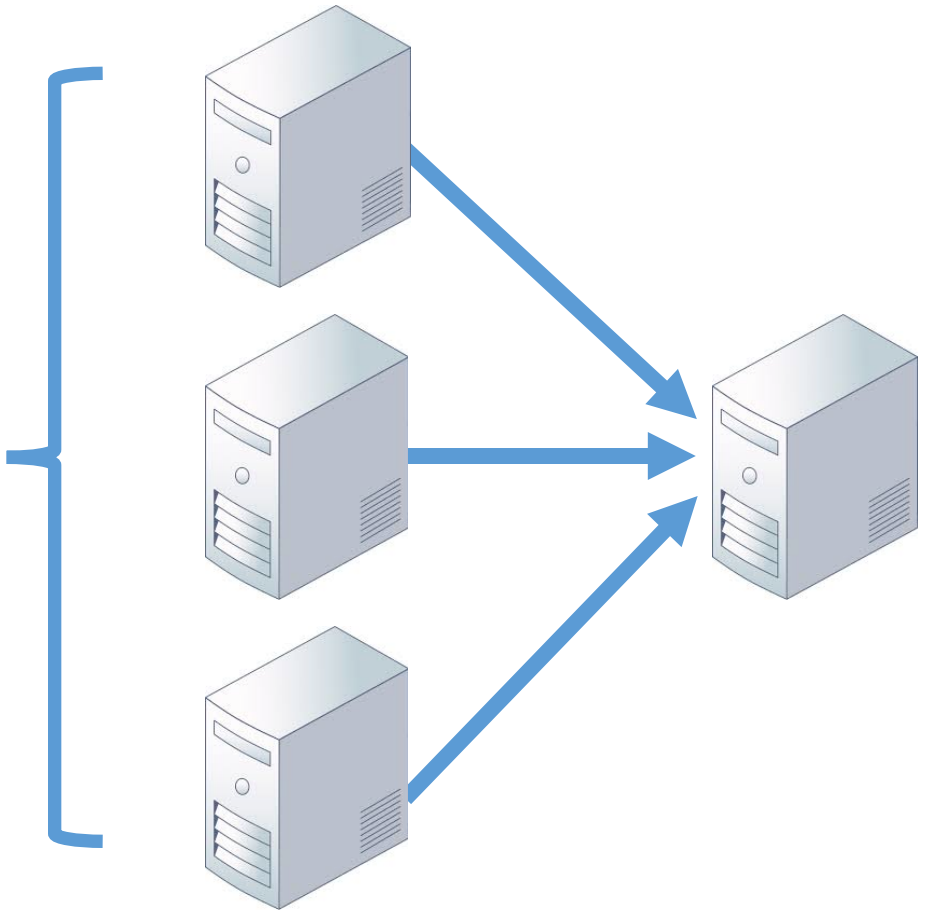
Expander graph-based monitoring

Multi-process cut detection

Fast path to consensus

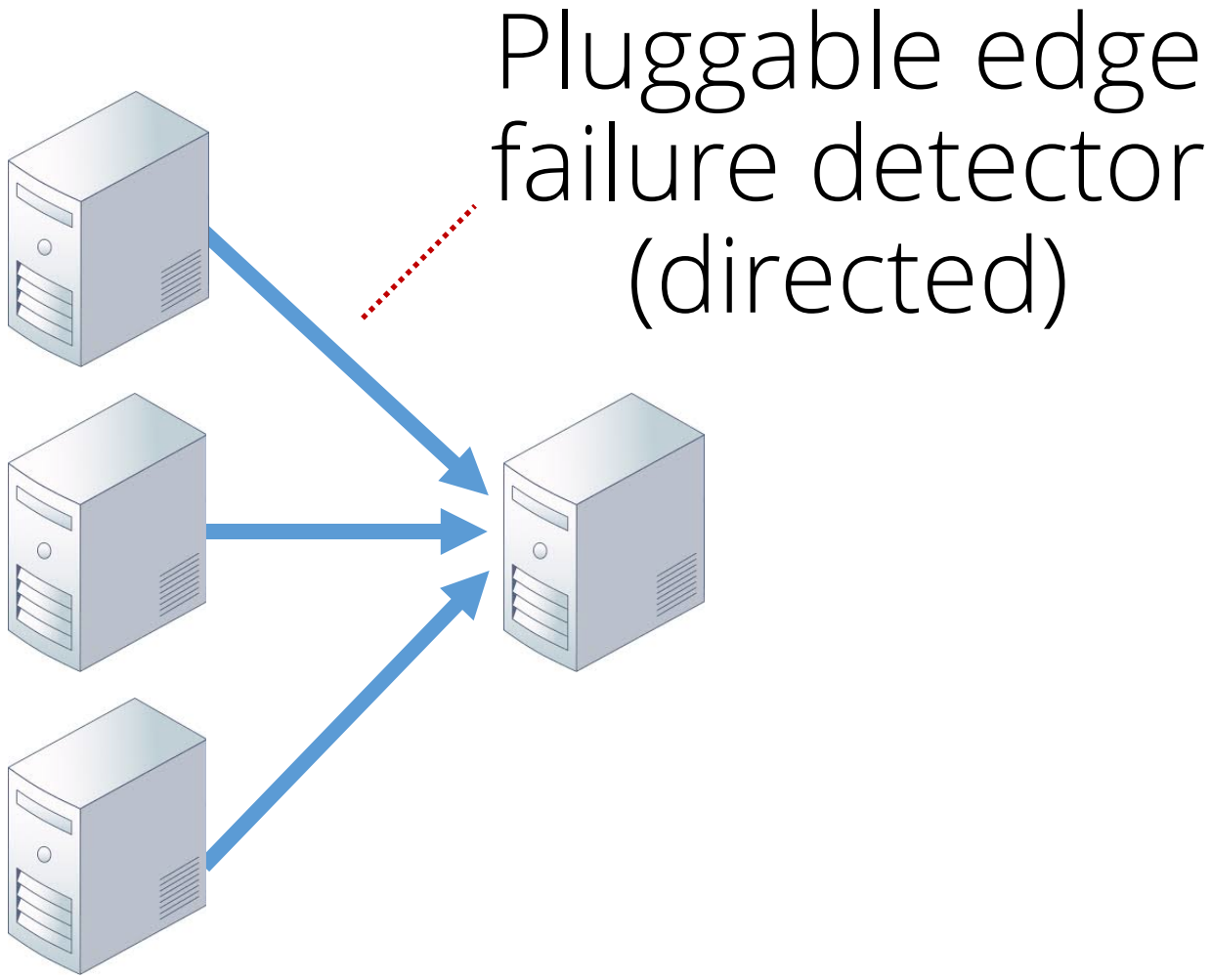
Expander-graph based monitoring

K observers per node



Subject

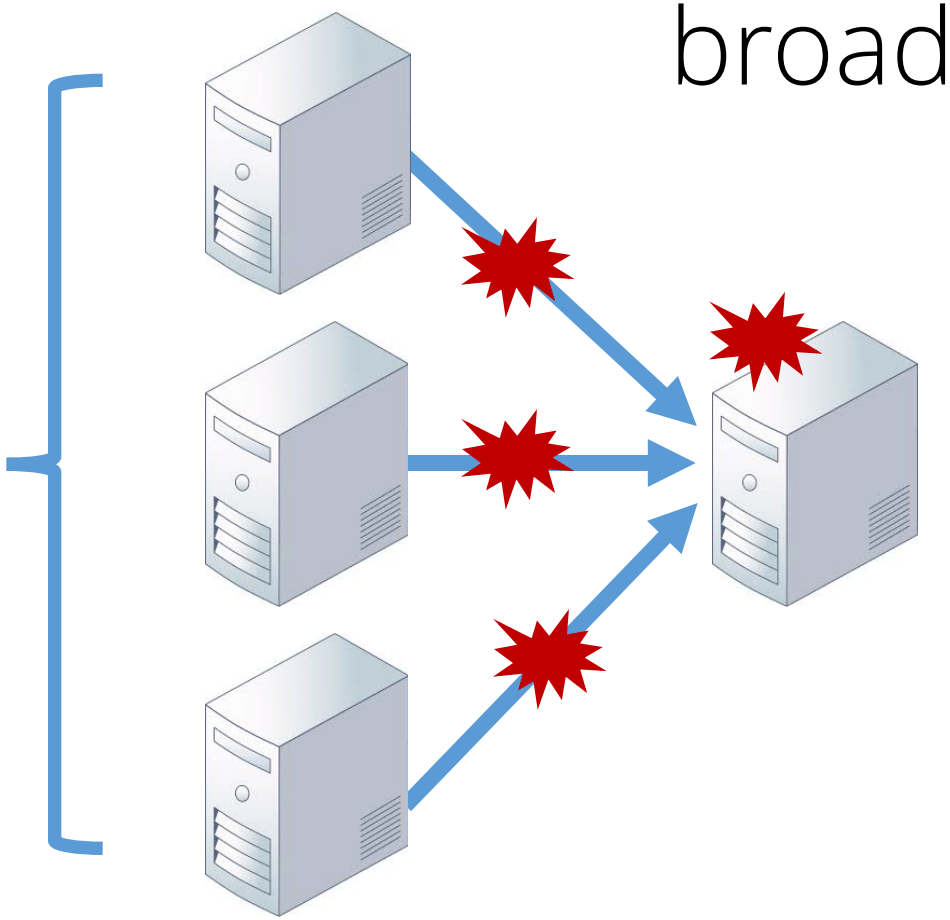
Expander-graph based monitoring



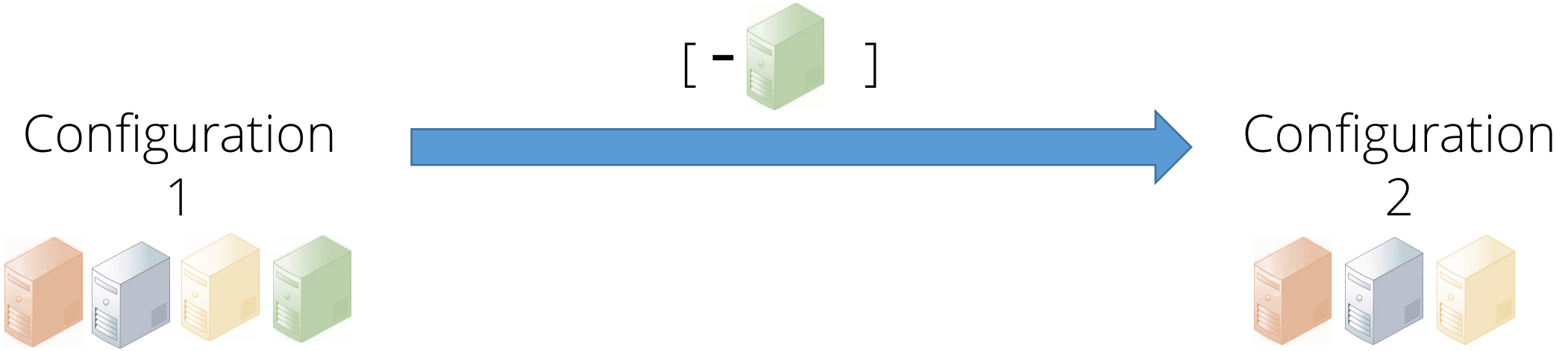
Expander-graph based monitoring

Up to K, edge alerts broadcasted during failures

K observers per node



Subject



Monitoring Overlay

Membership change proposal

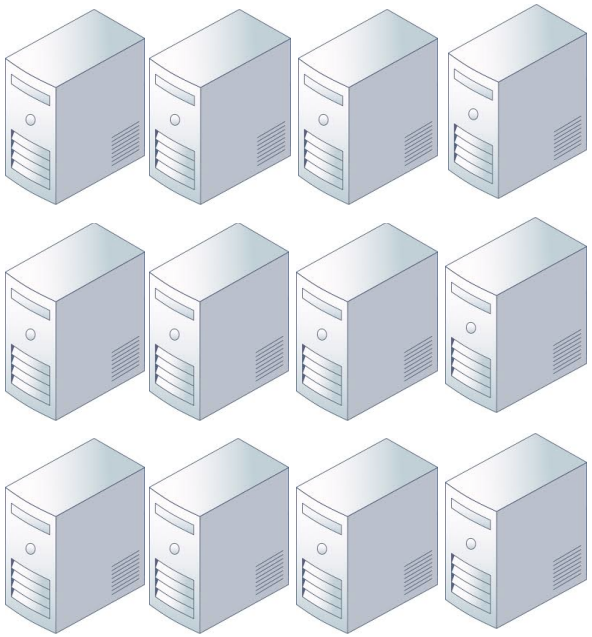
View change with consensus

Expander graph-based monitoring

Multi-process cut detection

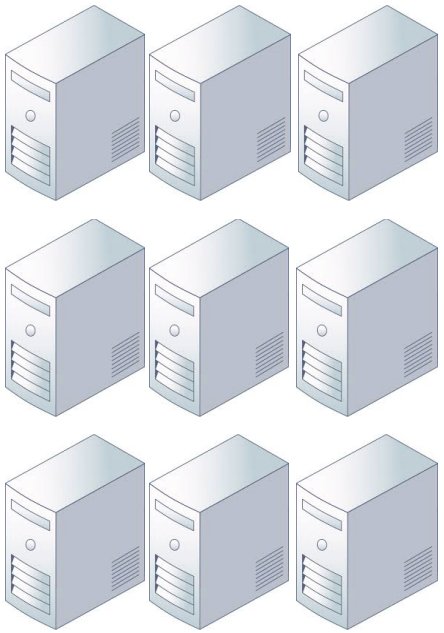
Fast path to consensus

Multi-process cut detection

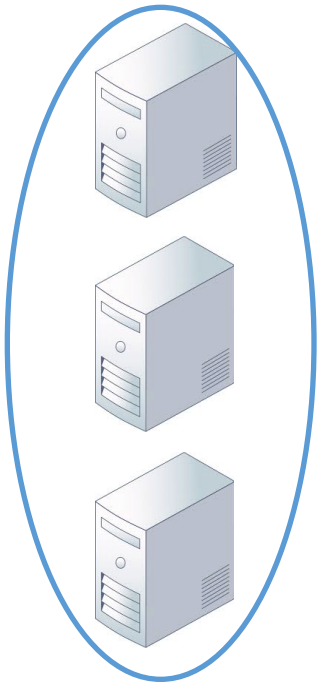


Expander-based monitoring overlay

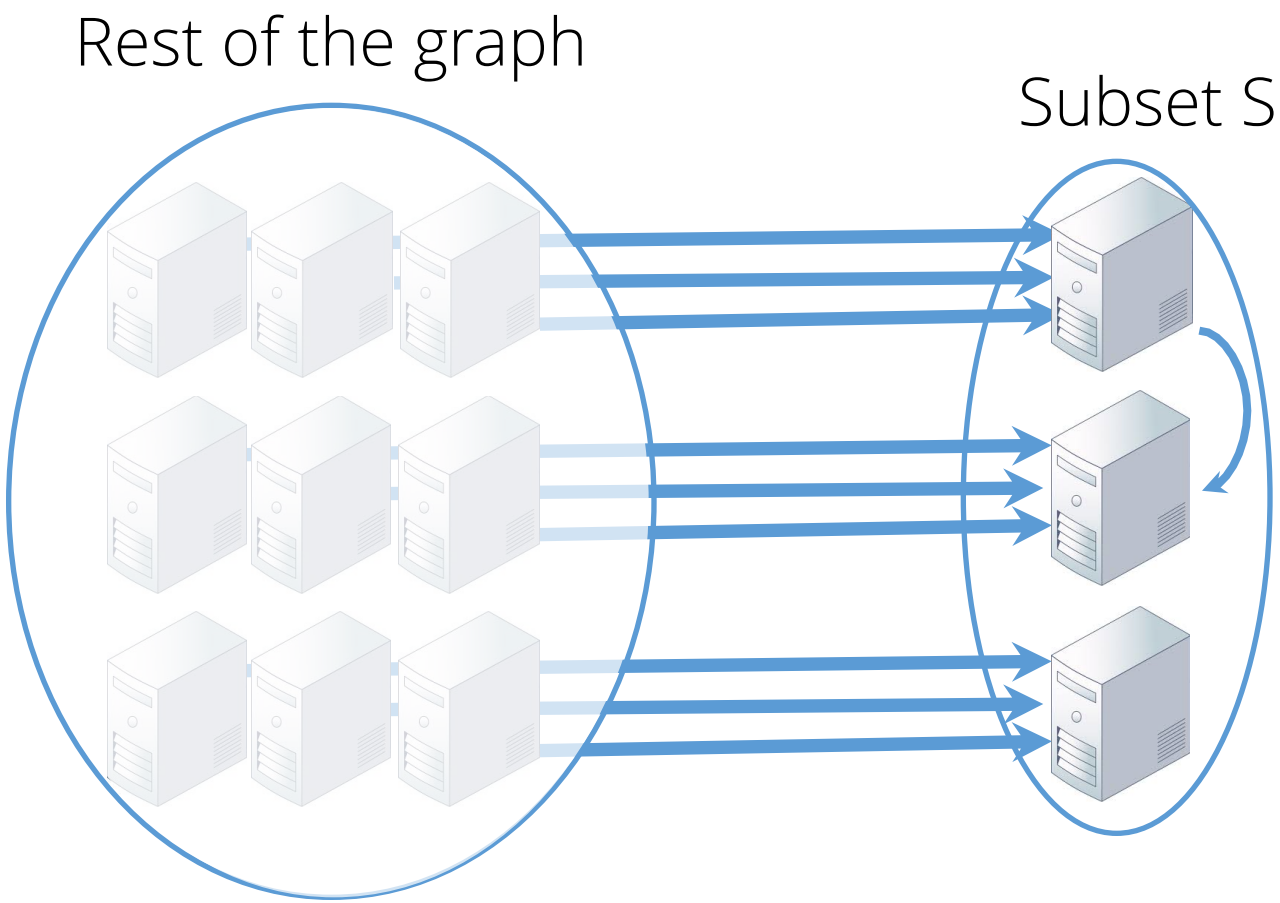
Multi-process cut detection



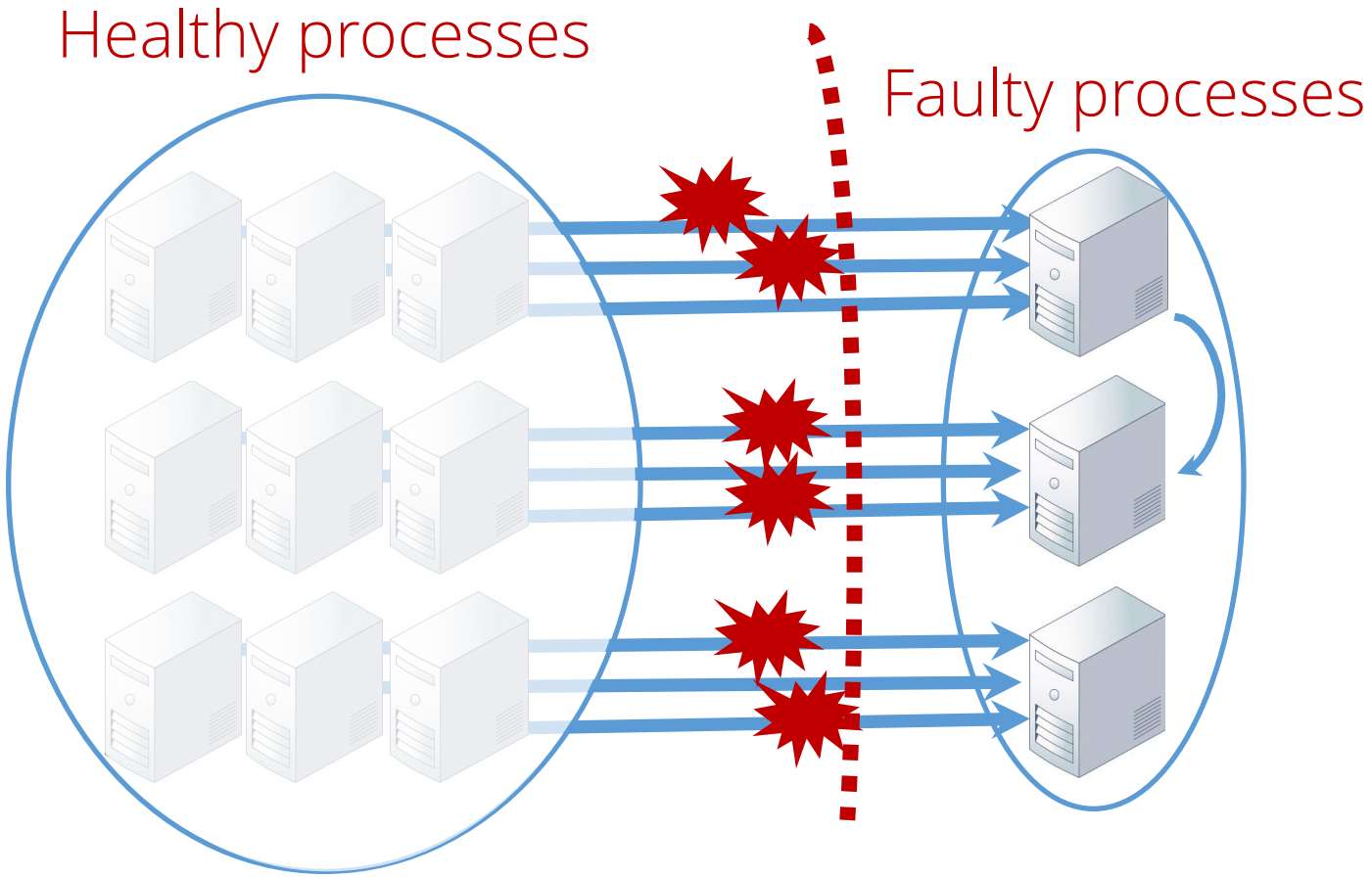
Subset S



Multi-process cut detection



Multi-process cut detection

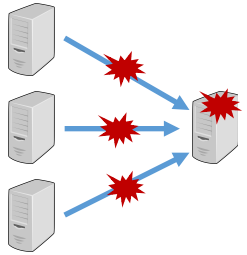


Multi-process cut detection

K

H

L



Multi-process cut detection

K

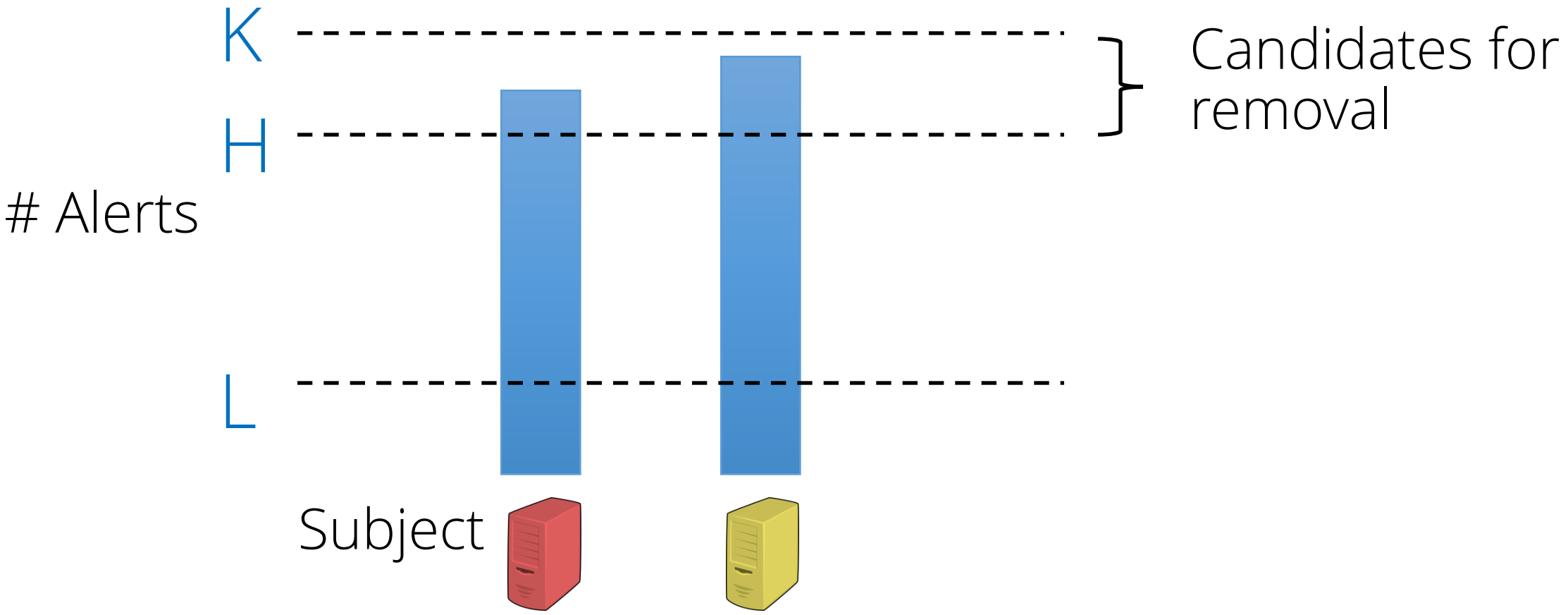
H

Alerts

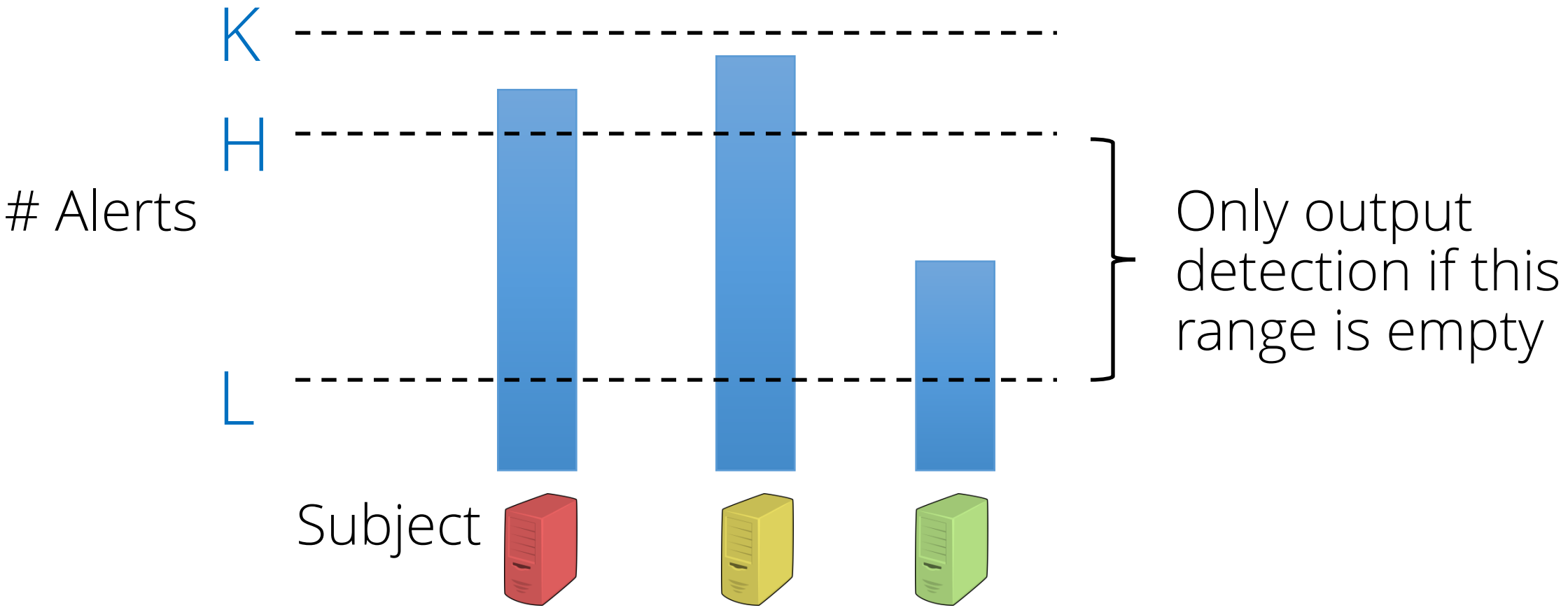
L

Subject

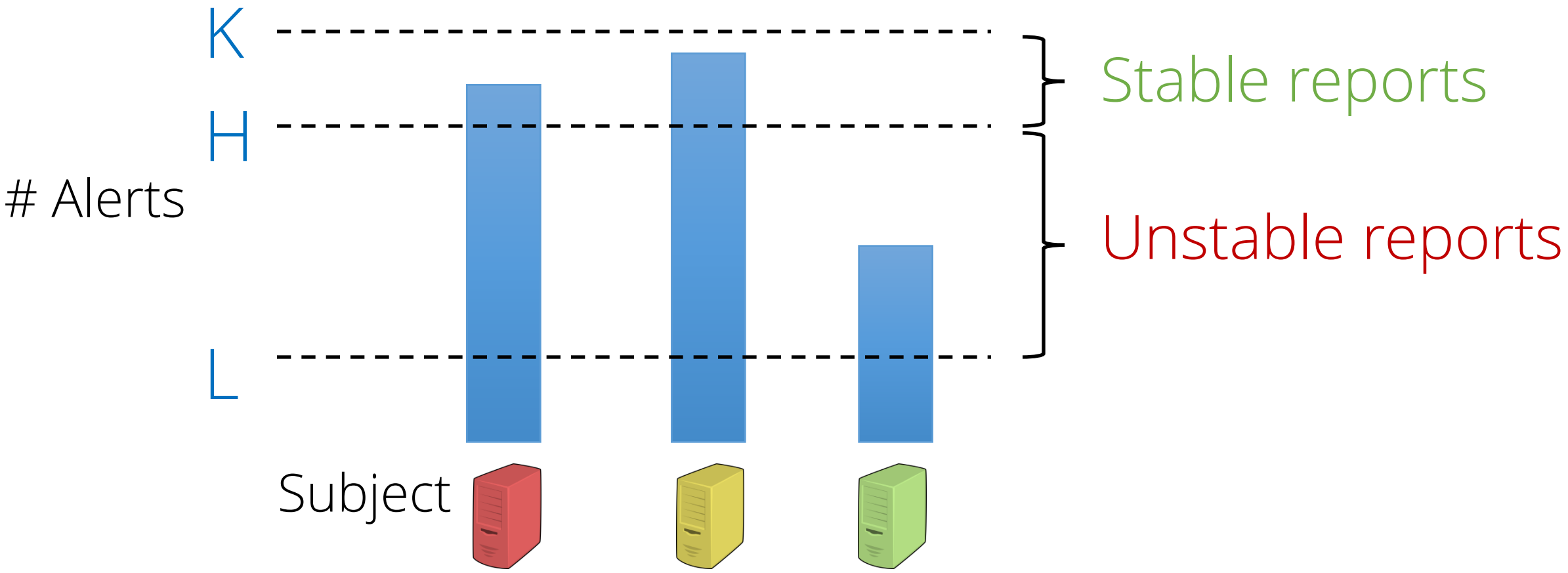
Multi-process cut detection



Multi-process cut detection

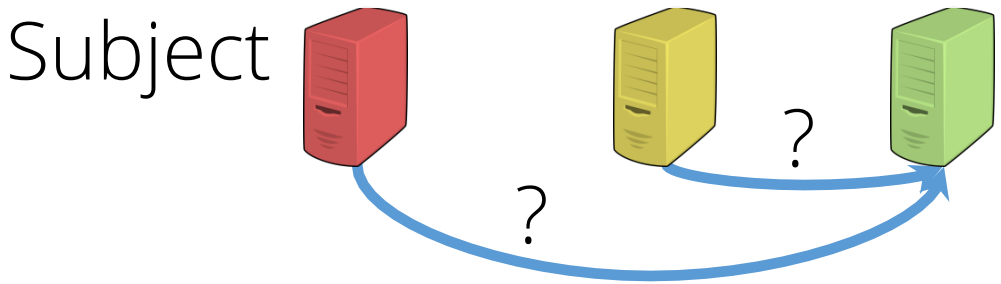


Multi-process cut detection

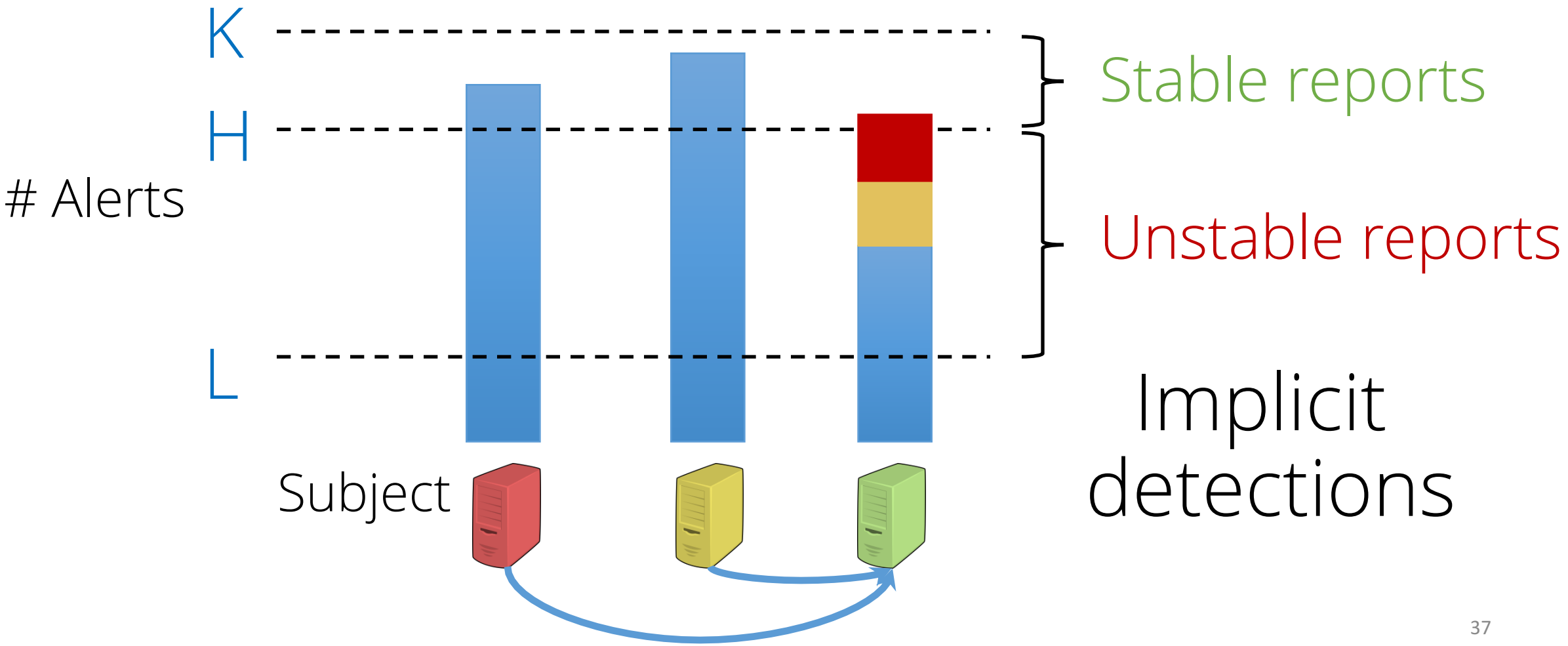


Multi-process cut detection

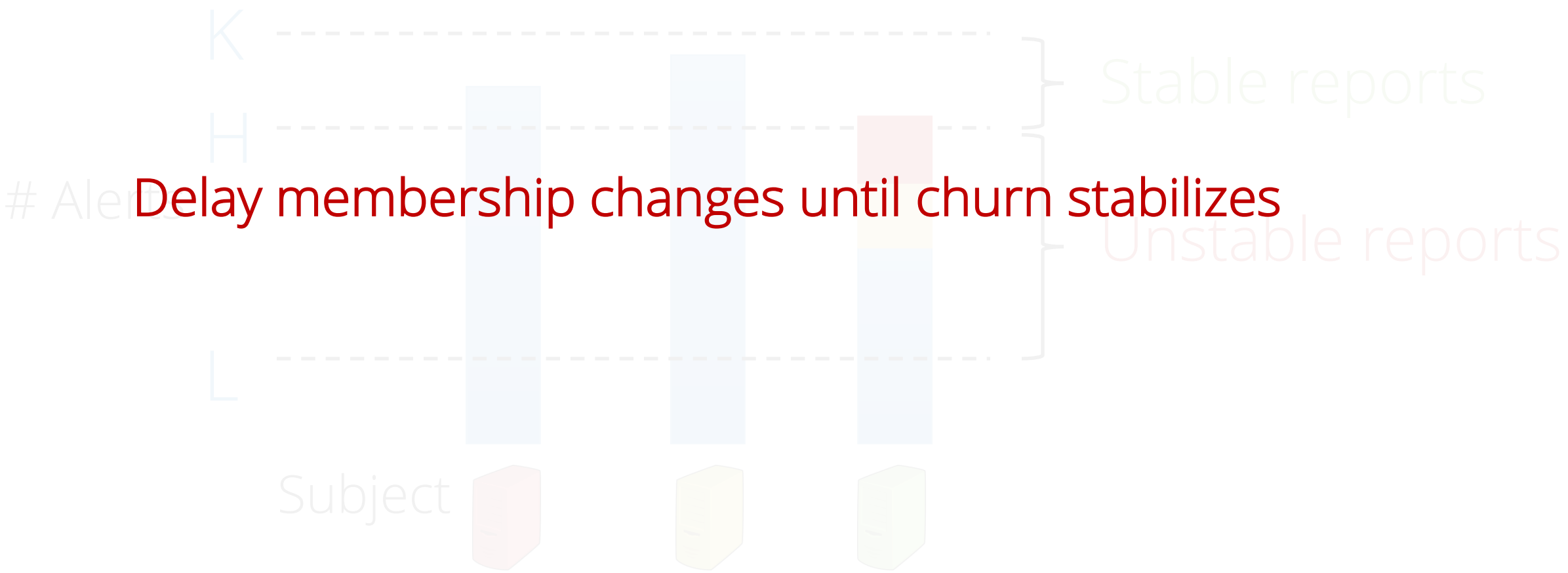
Observer-subject failures?



Multi-process cut detection



Multi-process cut detection



Delay membership changes until churn stabilizes

Almost-everywhere agreement

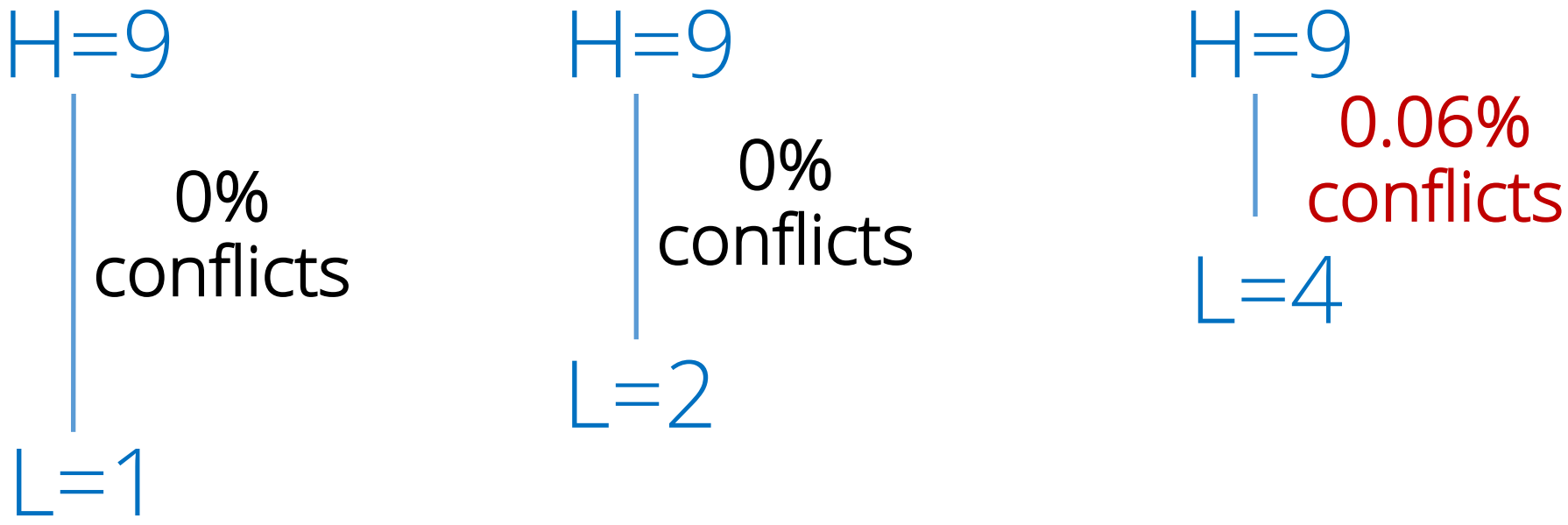
All processes output the same cut

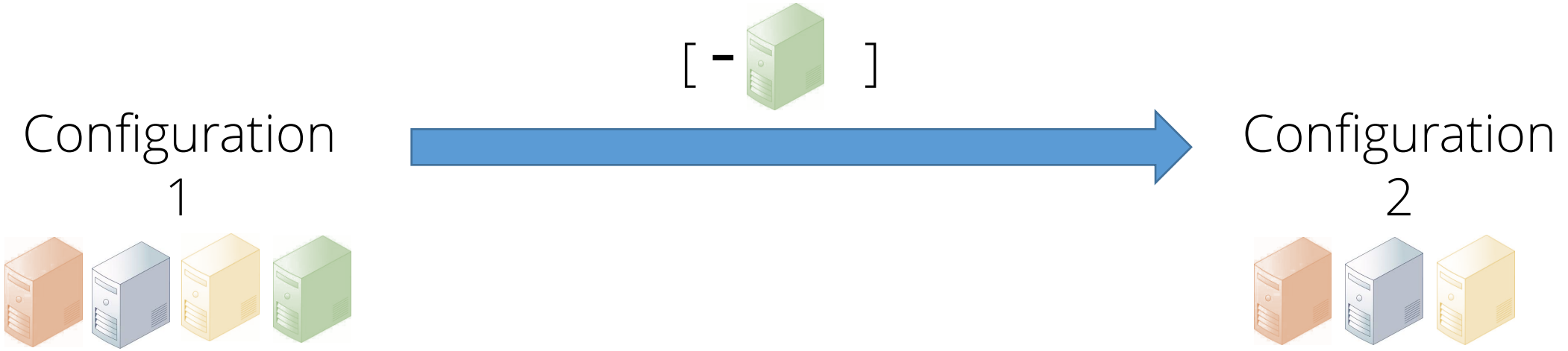
[-  , -  ,- ]

with high probability

Almost-everywhere agreement

1000 processes, 8 failures, K=10



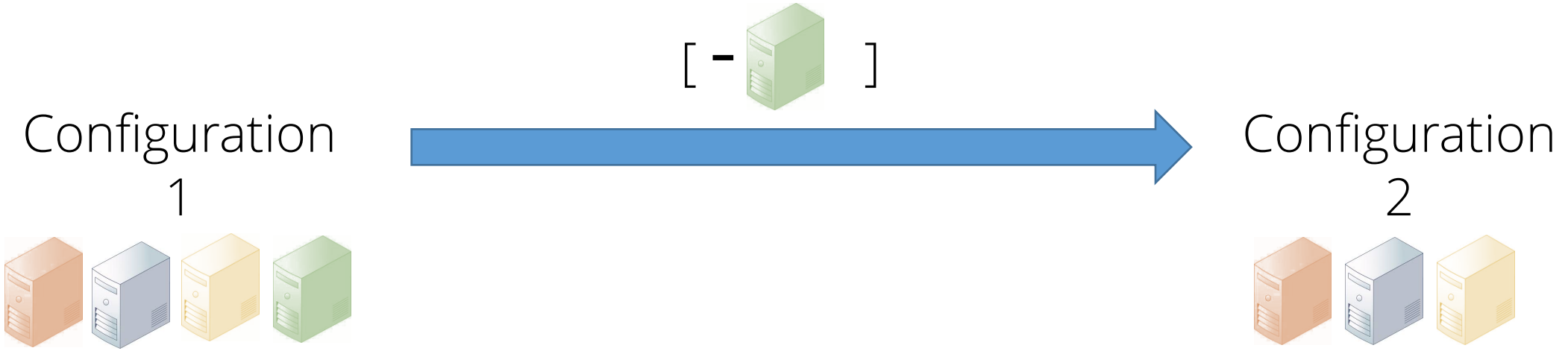


Monitoring Overlay → Membership change proposal → View change with consensus

Expander graph-based monitoring

Multi-process cut detection

Fast path to consensus



Monitoring Overlay → Membership change proposal → View change with consensus

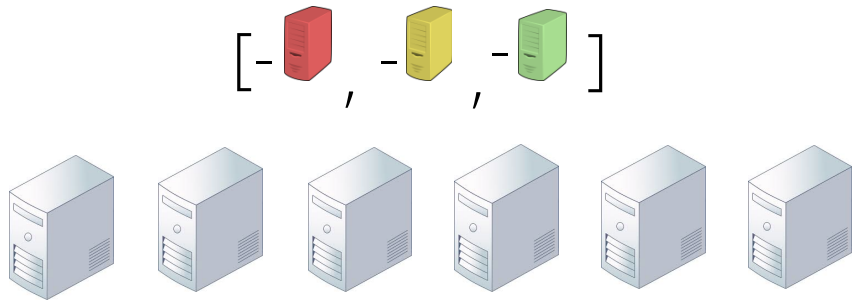
Expander graph-based monitoring

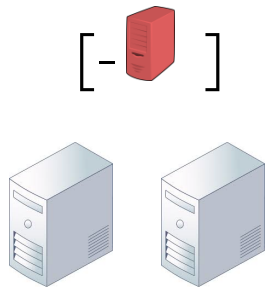
Multi-process cut detection

Fast path to consensus

Almost-everywhere agreement

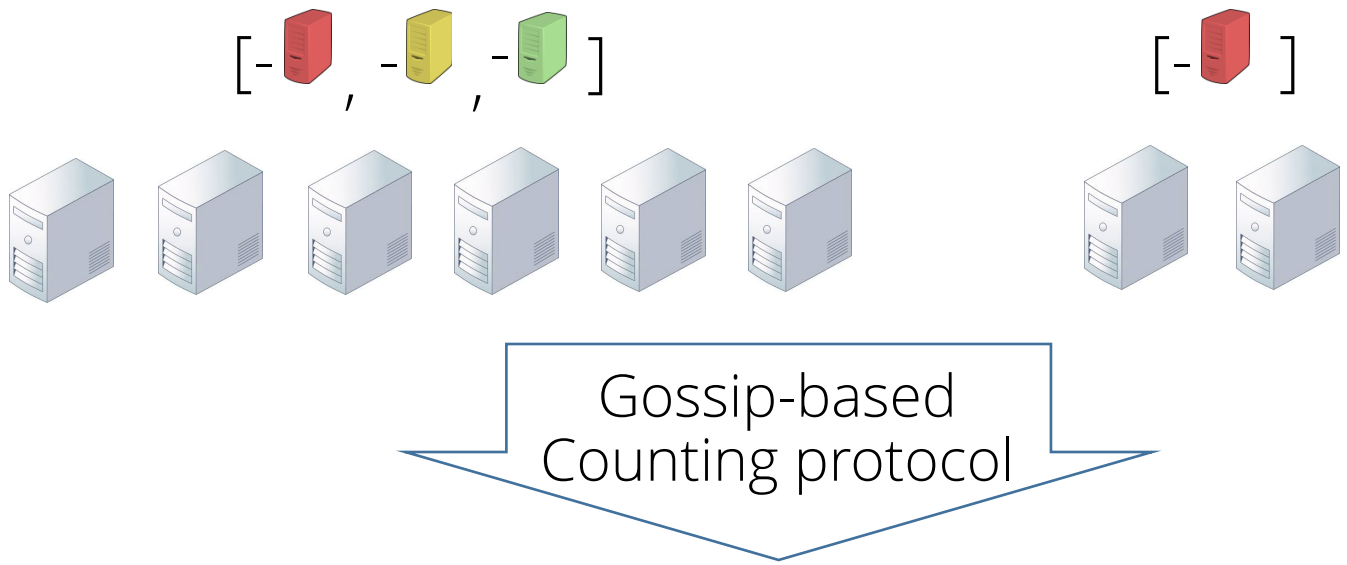
Full agreement





Almost-everywhere agreement

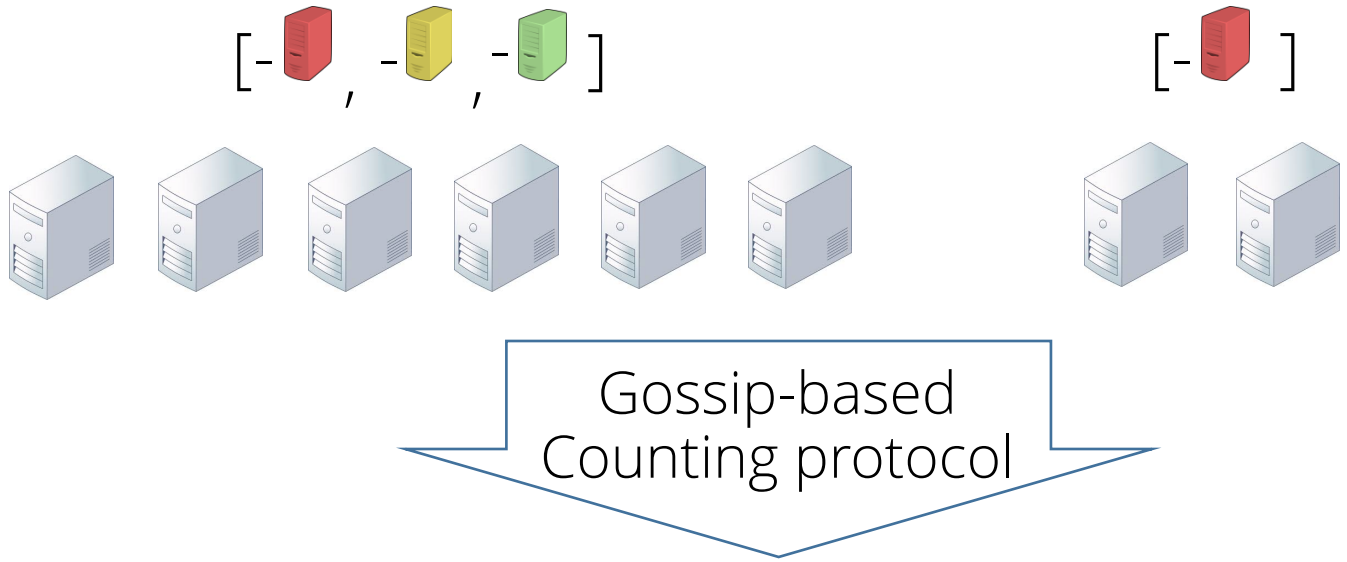
Full agreement



Every node counts #votes per-proposal

Almost-everywhere agreement

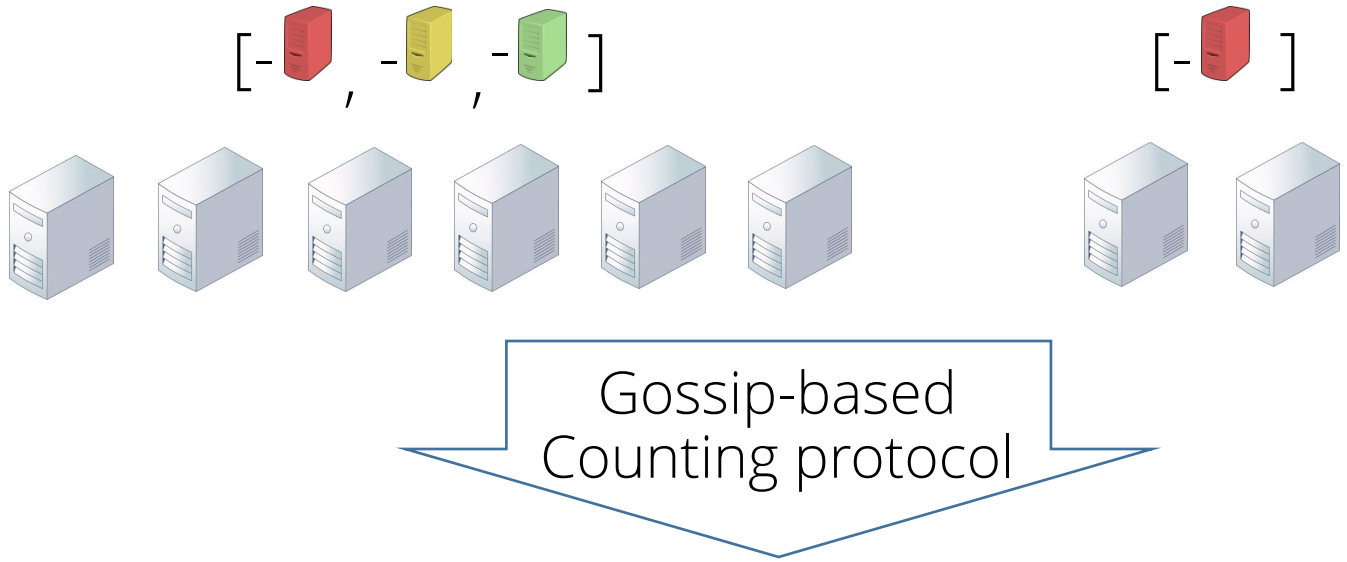
Full agreement



Decide if Fast Paxos quorum (> $\frac{3n}{4}$ nodes) of identical votes

Almost-everywhere agreement

Full agreement



1000 processes, 10 node membership change
~11 KB bandwidth usage per node for 1 second
(Memberlist uses ~8 KB/s)

Evaluation

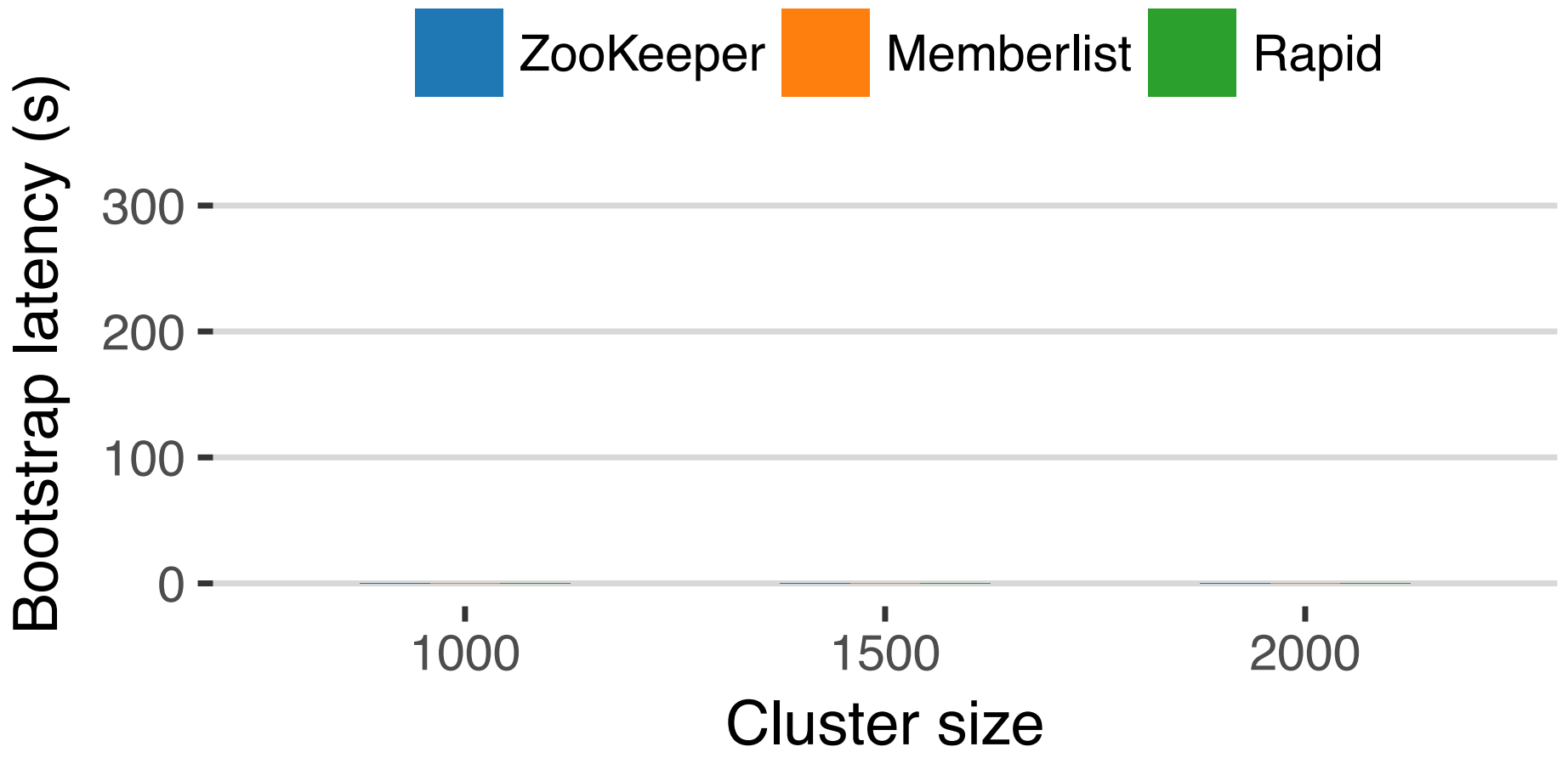
Implementation: ~2700 LOC in Java (~2600 LOC of tests)
github.com/lalithsuresh/rapid

Compared against 3-node Zookeeper cluster and Memberlist.

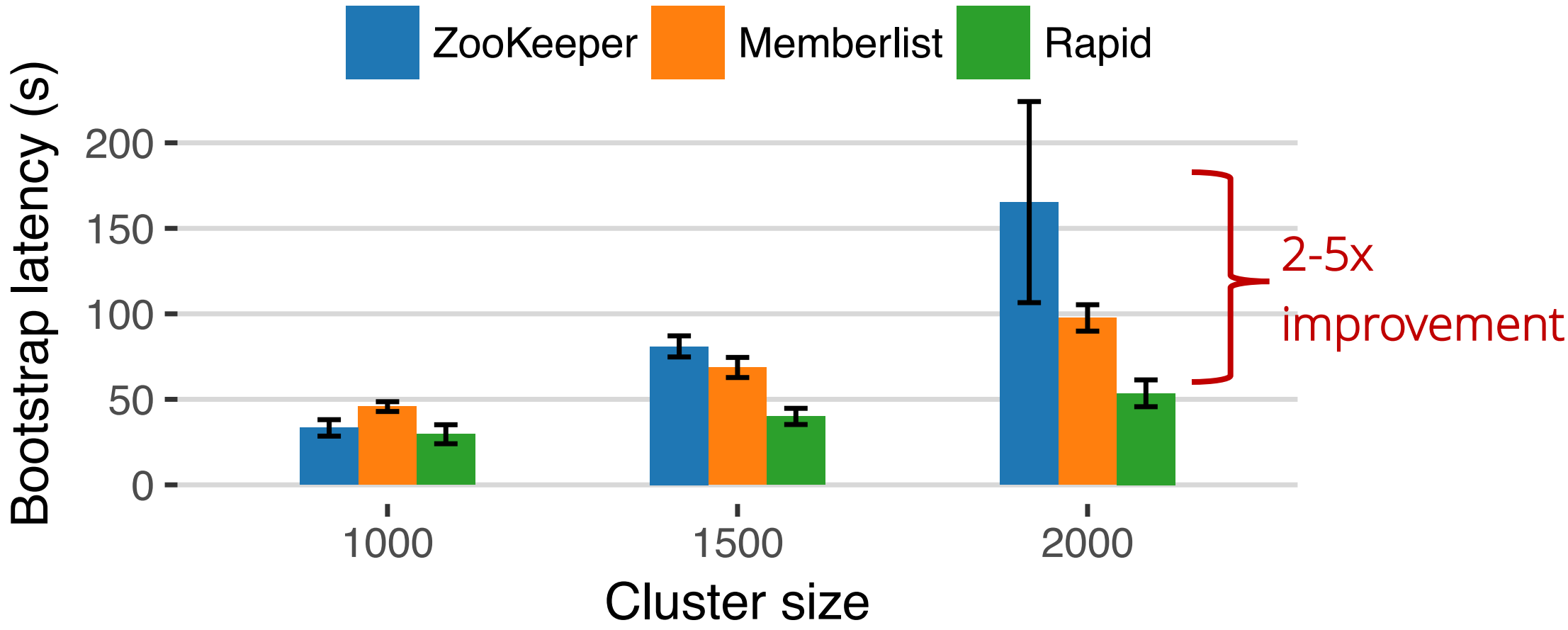
Experiments run on 100 VMs
(2 cores, 4GB RAM each)

Not showing Akka Cluster because it did not scale past 500 nodes.

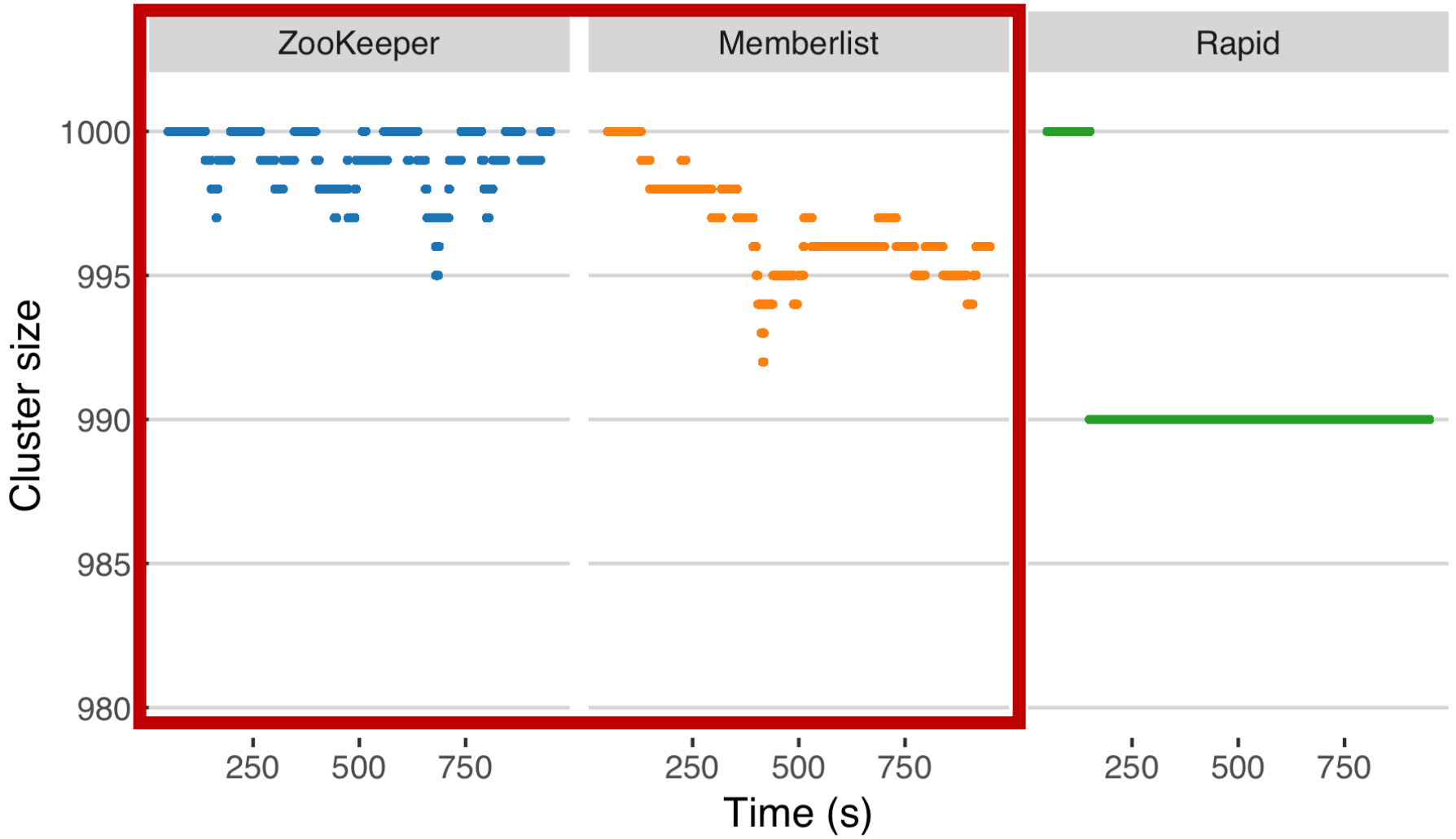
Bootstrap times



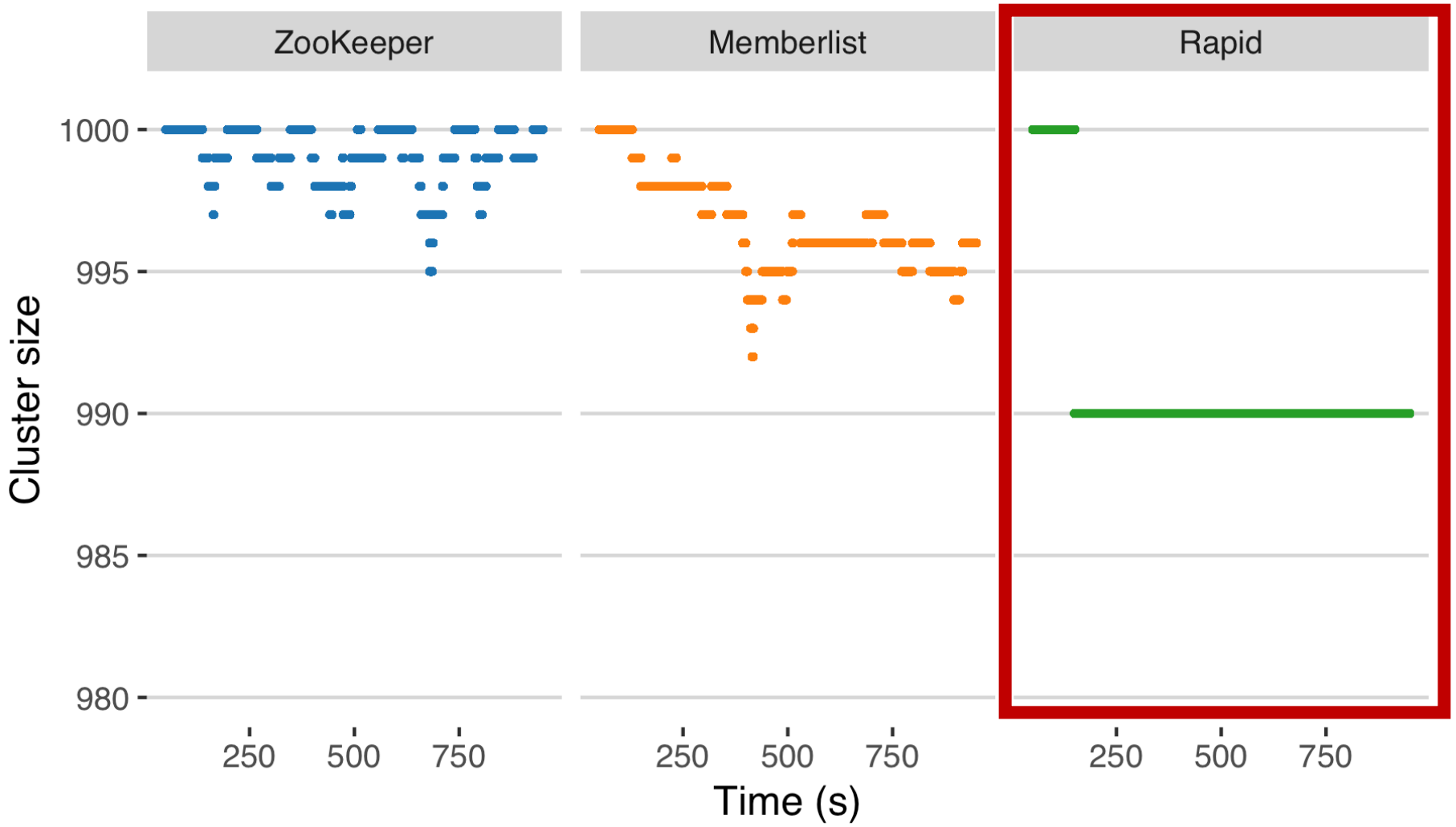
Bootstrap times



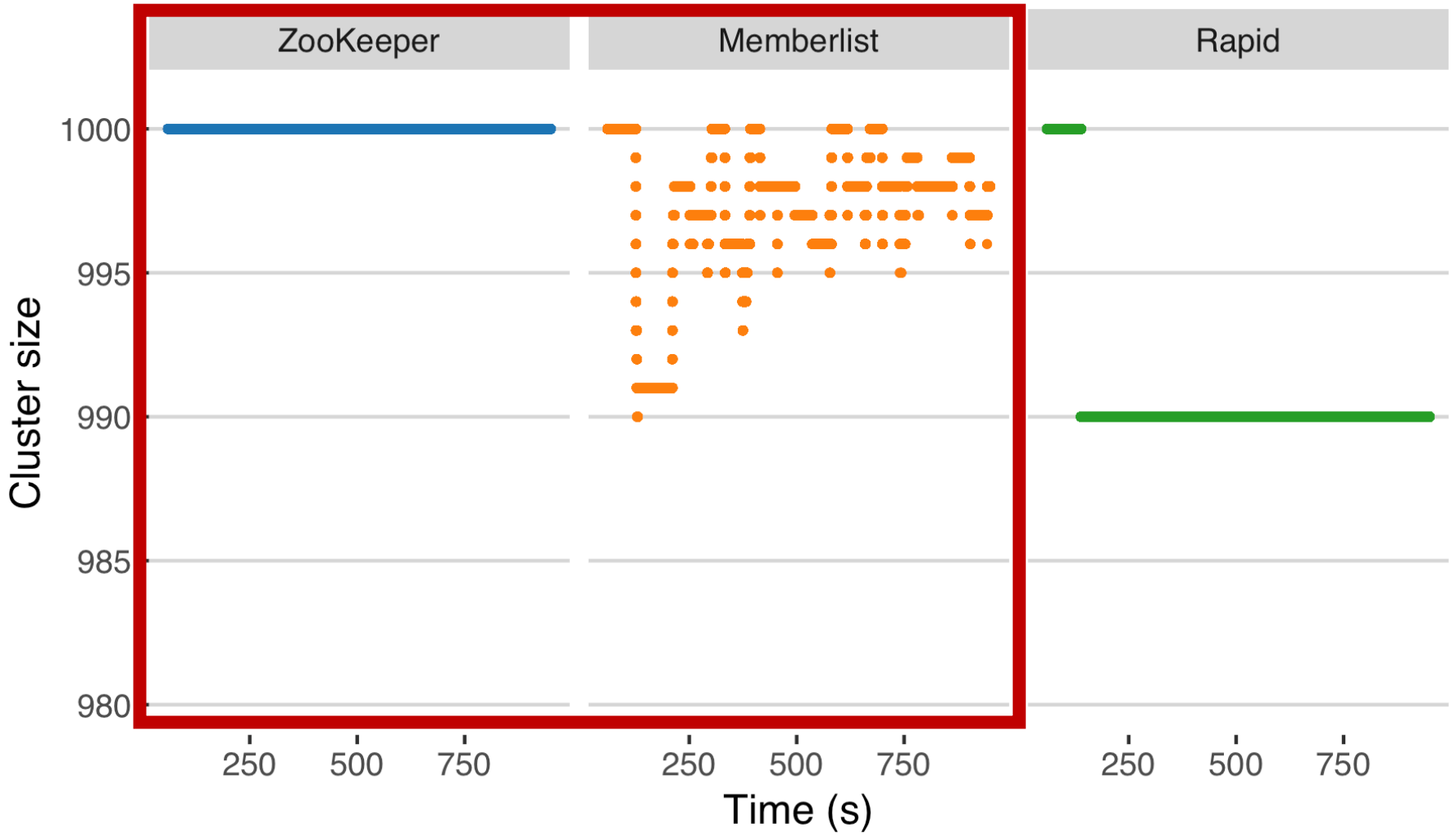
1% of processes experience high packet loss



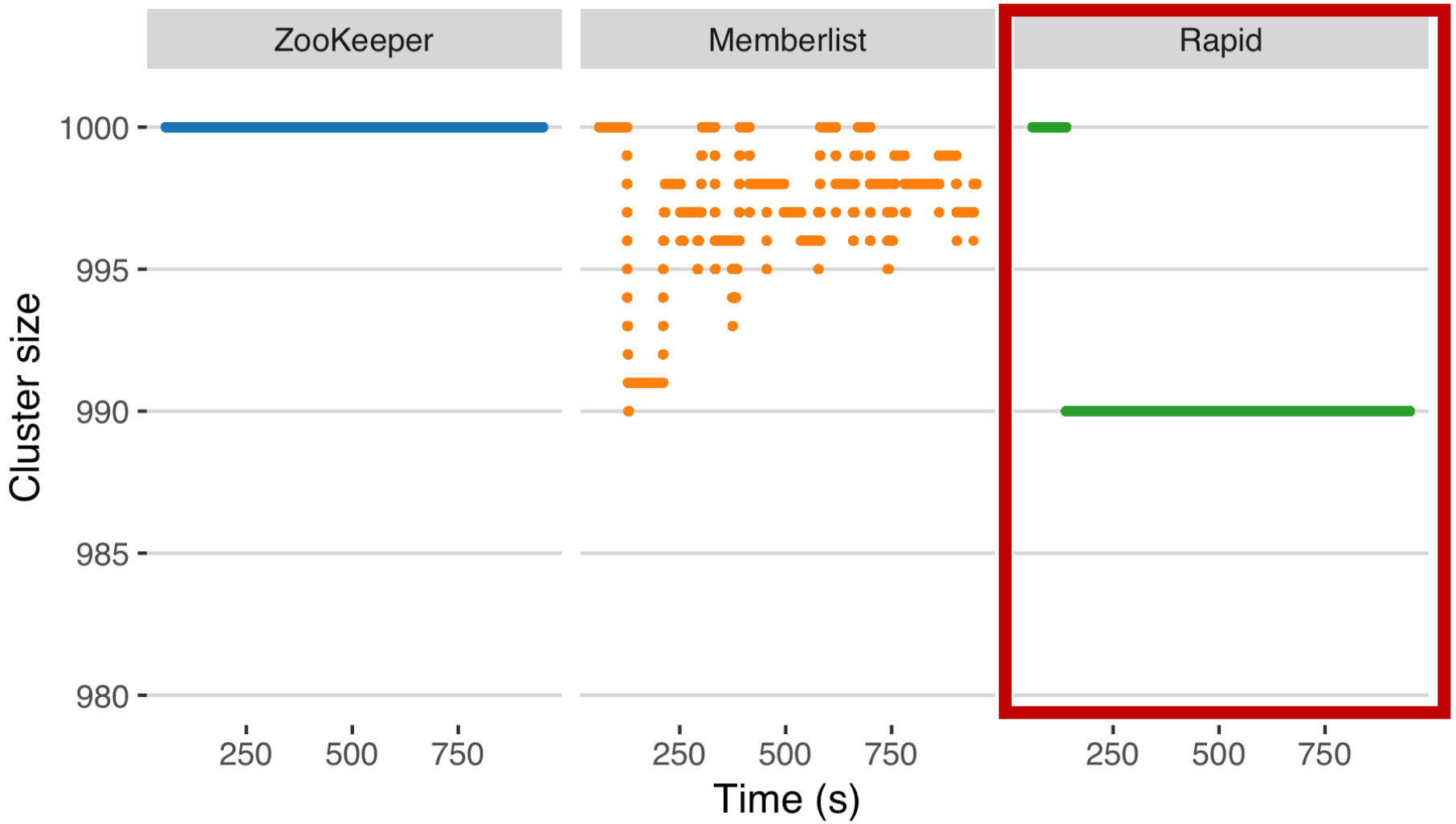
1% of processes experience high packet loss



1% of processes experience one way network partition

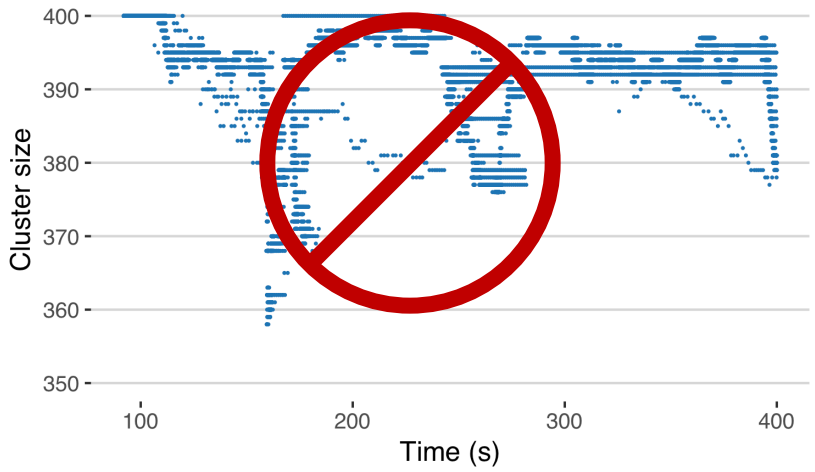


1% of processes experience one way network partition



Rapid

Stable and consistent membership at scale



Monitoring Overlay → Membership change proposal → View change with consensus

Expander graph-based monitoring

Multi-process cut detection

Fast path to consensus