How to Understand Galera

Alexey Yurchenko
Codership Oy

codership

Agenda

- The difference between traditional (e.g. MySQL) replication and Galera.
- General Galera principles.

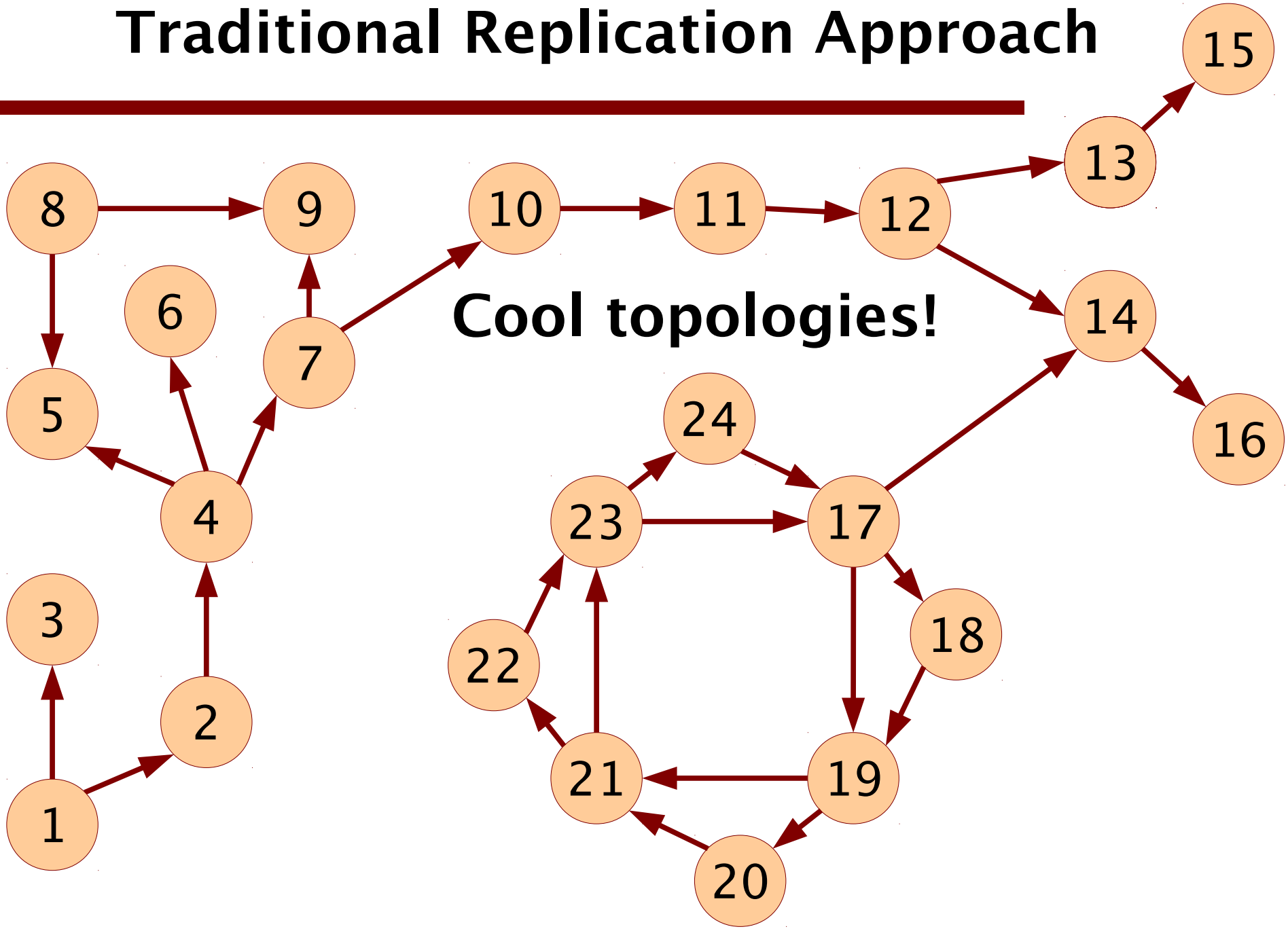
Galera Difference

Traditional Replication Approach

Server-centric:

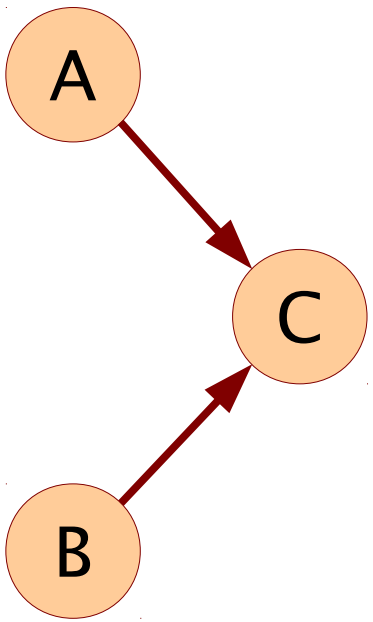
"One server streams data to another"

Traditional Replication Approach



Cool topologies!

Traditional Replication Approach

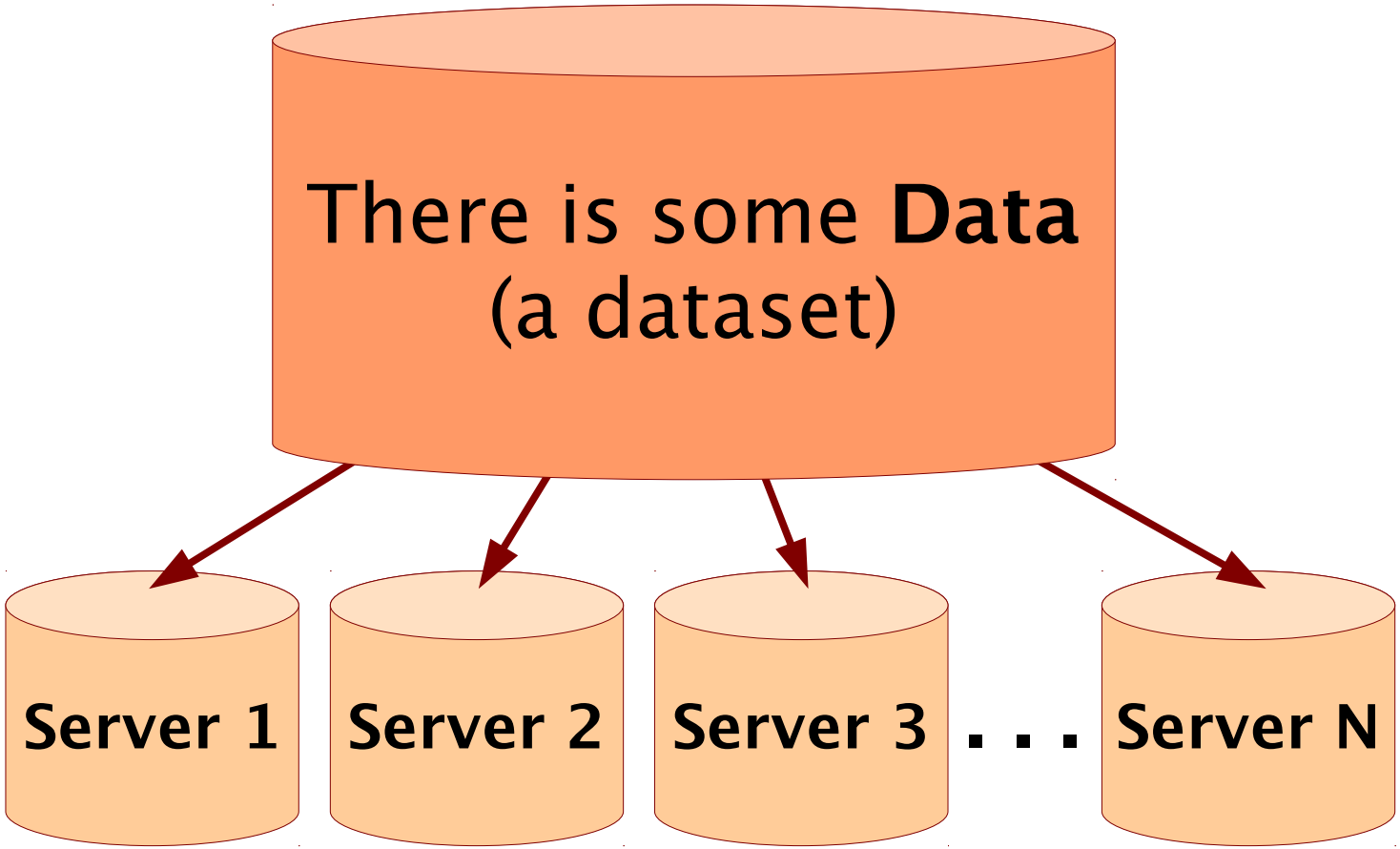


But there are questions:

- If node C crashes, do we still have a cluster?
- If node B crashes and clients failover to C, how B joins back?
- Which node has data X?
- How do we backup the cluster?

Galera (wsrep API) Approach

Data-centric:



- it is synchronized between one or more servers

Galera (wsrep API) Approach

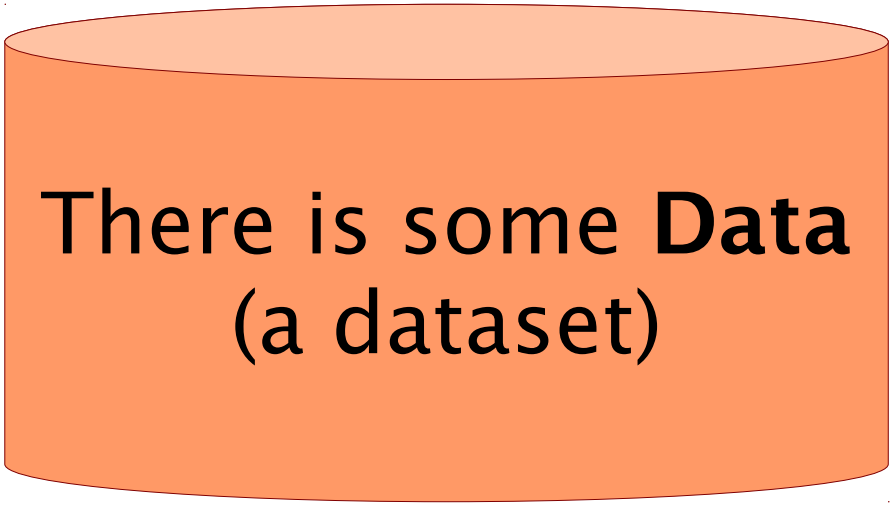
Data-centric:

Data does not belong to a Node –

Node belongs to Data

Galera (wsrep API) Approach

Data-centric:



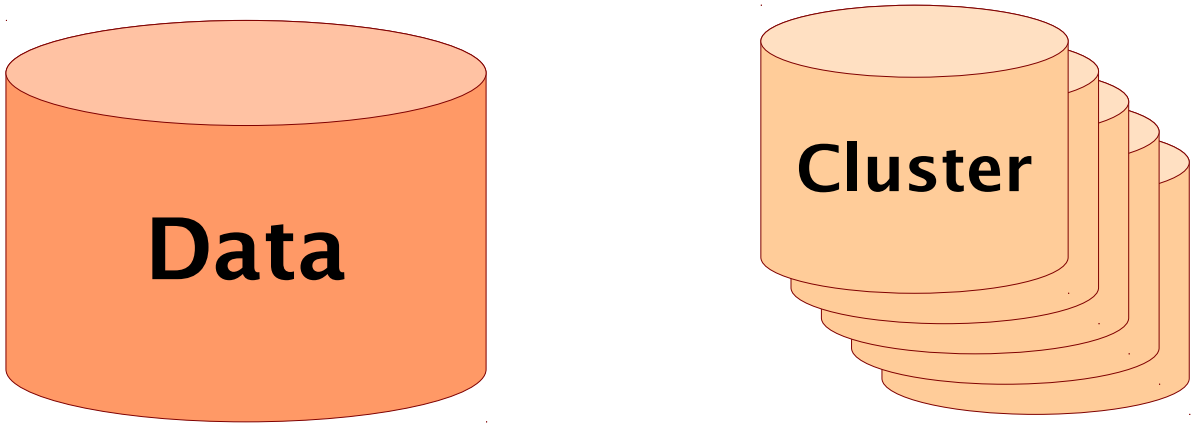
The dataset needs an ID:

`00295a79-9c48-11e2-bdf0-9a916cbb9294`

Galera (wsrep API) Approach

Data-centric:

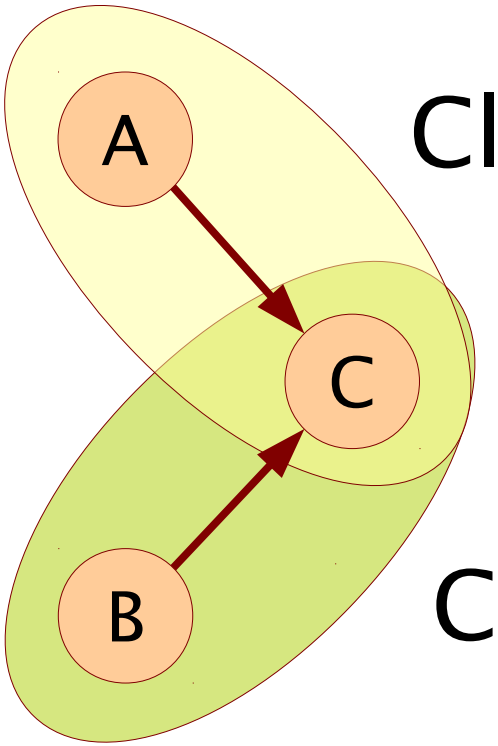
Dataset ID == Cluster ID



00295a79-9c48-11e2-bdf0-9a916cbb9294

Galera (wsrep API) Approach

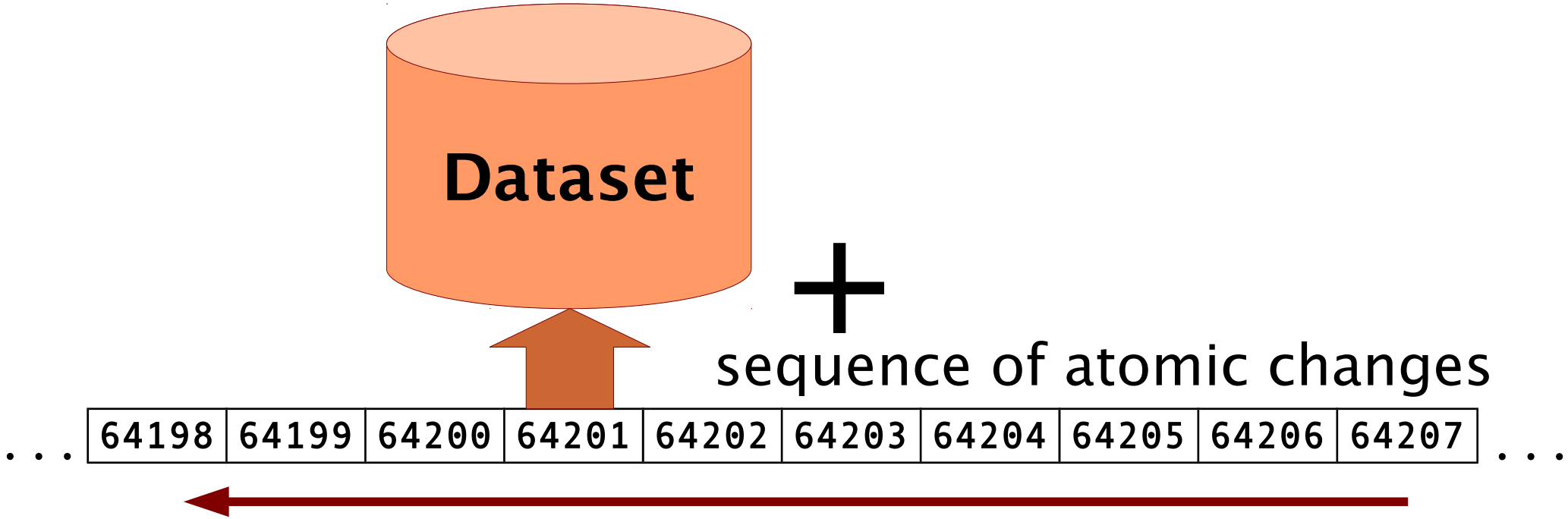
Data-centric:



Cluster A (9b0e4ad8-a42b-11e2-bdf0-9851bb738edb)

Cluster B (e1eedd79-a5d6-11e2-a801-aae16e1f5226)

Global Transaction ID (GTID)

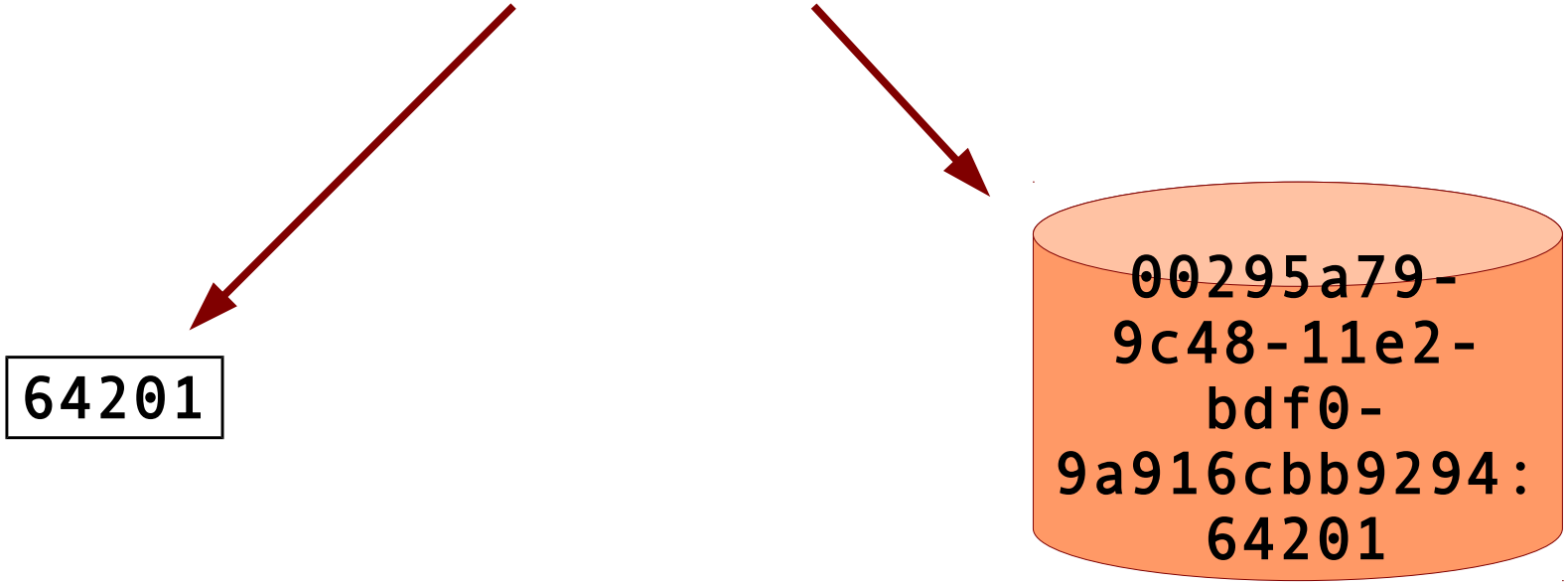


=

00295a79-9c48-11e2-bdf0-9a916cbb9294:64201

Global Transaction ID (GTID)

00295a79-9c48-11e2-bdf0-9a916cbb9294:64201



Global Transaction ID

Dataset State ID

Global Transaction ID (GTID)

`00295a79-9c48-11e2-bdf0-9a916cbb9294:0`

– initial data

`00295a79-9c48-11e2-bdf0-9a916cbb9294:1`

– first change/transaction

`00000000-0000-0000-0000-000000000000:-1`

– undefined GTID

Global Transaction ID (GTID)

Galera GTID:

`00295a79-9c48-11e2-bdf0-9a916cbb9294:64201`

MySQL 5.6 GTID:

`8182213e-7c1e-11e2-a6e2-080027635ef5:12345`

Global Transaction ID (GTID)

Galera GTID:

`00295a79-9c48-11e2-bdf0-9a916cbb9294`:64201

Cluster ID

MySQL 5.6 GTID:

`8182213e-7c1e-11e2-a6e2-080027635ef5`:12345

Server ID

Global Transaction ID (GTID)

Galera GTID:

`00295a79-9c48-11e2-bdf0-9a916cbb9294`:`64201`

Cluster ID

data change in the cluster

MySQL 5.6 GTID:

`8182213e-7c1e-11e2-a6e2-080027635ef5`:`12345`

Server ID

transaction processed by the server

Global Transaction ID (GTID)

What we see in MySQL 5.6:

```
8182213e-7c1e-11e2-a6e2-080027635ef5:12345
8182213e-7c1e-11e2-a6e2-080027635ef5:12346
8182213e-7c1e-11e2-a6e2-080027635ef5:12347
← new master promoted →
f4e3bf7a-a91f-11e2-4e02-3f8dbcffaed8:1
f4e3bf7a-a91f-11e2-4e02-3f8dbcffaed8:2
f4e3bf7a-a91f-11e2-4e02-3f8dbcffaed8:3
```

Global Transaction ID (GTID)

What we see in Galera:

```
00295a79-9c48-11e2-bdf0-9a916cbb9294:64201
00295a79-9c48-11e2-bdf0-9a916cbb9294:64202
00295a79-9c48-11e2-bdf0-9a916cbb9294:64203
← new master promoted →
00295a79-9c48-11e2-bdf0-9a916cbb9294:64204
00295a79-9c48-11e2-bdf0-9a916cbb9294:64205
00295a79-9c48-11e2-bdf0-9a916cbb9294:64206
```

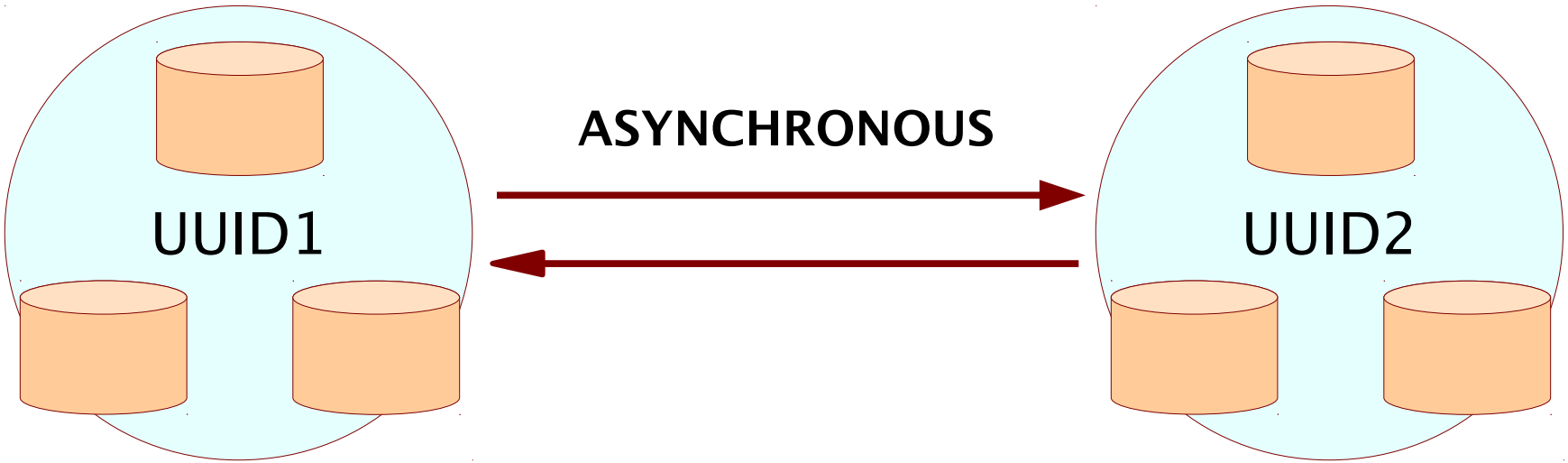
Global Transaction ID (GTID)

1) Galera nodes are **ANONYMOUS** => all equal.

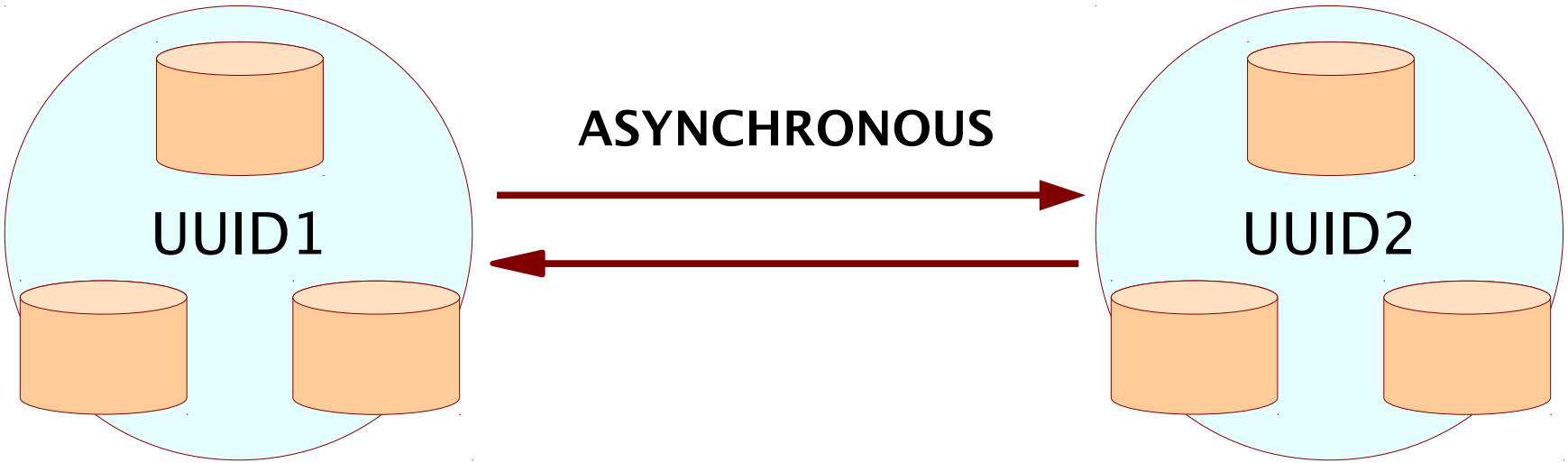
2) Galera cluster == one big distributed "master".

3) Asynchronous replication to/from Galera cluster is now piece of cake.

Global Transaction ID (GTID)



Global Transaction ID (GTID)



UUID1:123
UUID2:539

UUID1:123
UUID2:543

Global Transaction ID (GTID)

SYNCHRONOUS / ASYNCHRONOUS

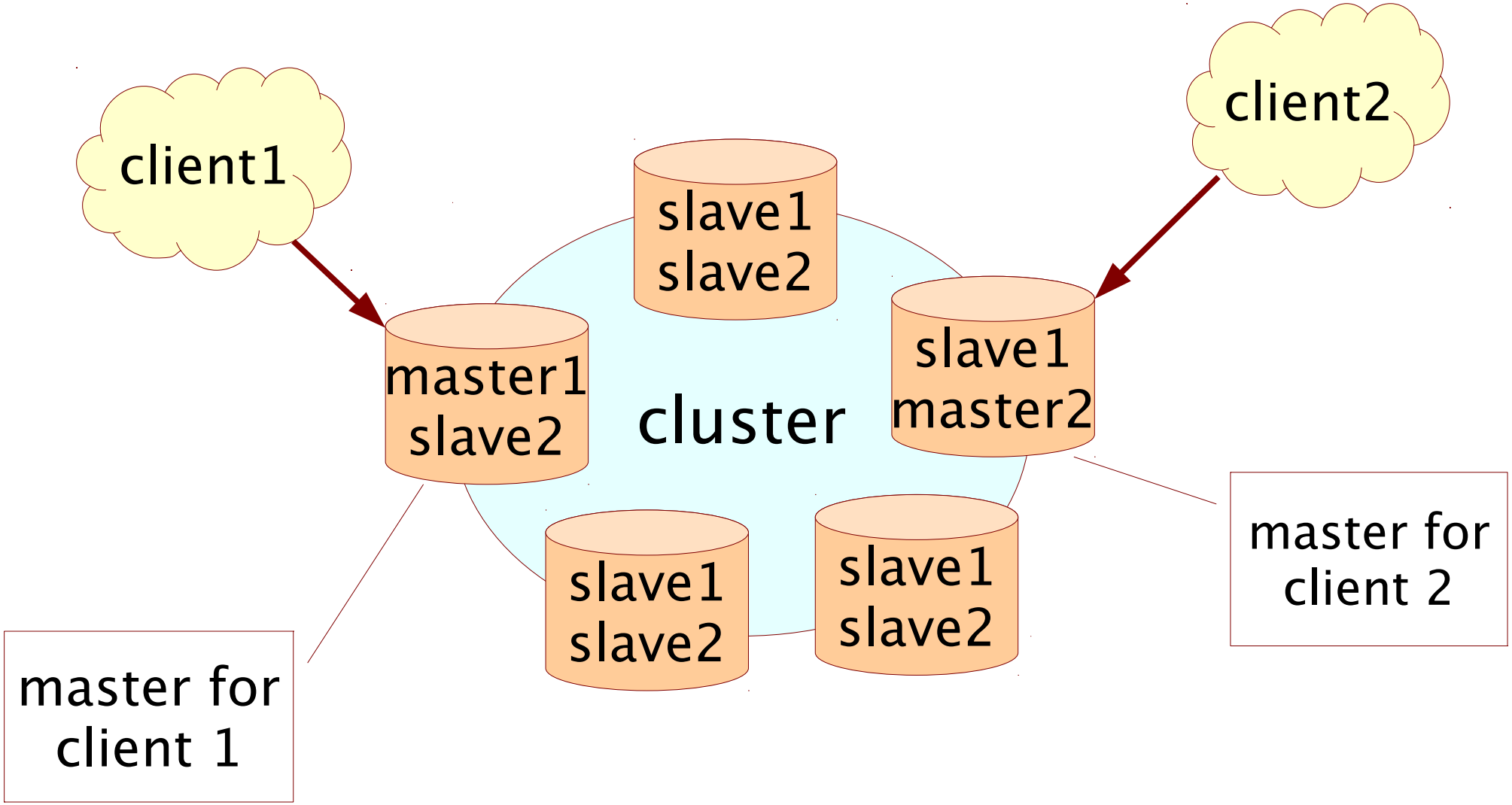
<==>

SINGLE DATABASE / INDEPENDENT DATABASES

<==>

CONSISTENCY / INCONSISTENCY

Master / Slave ?

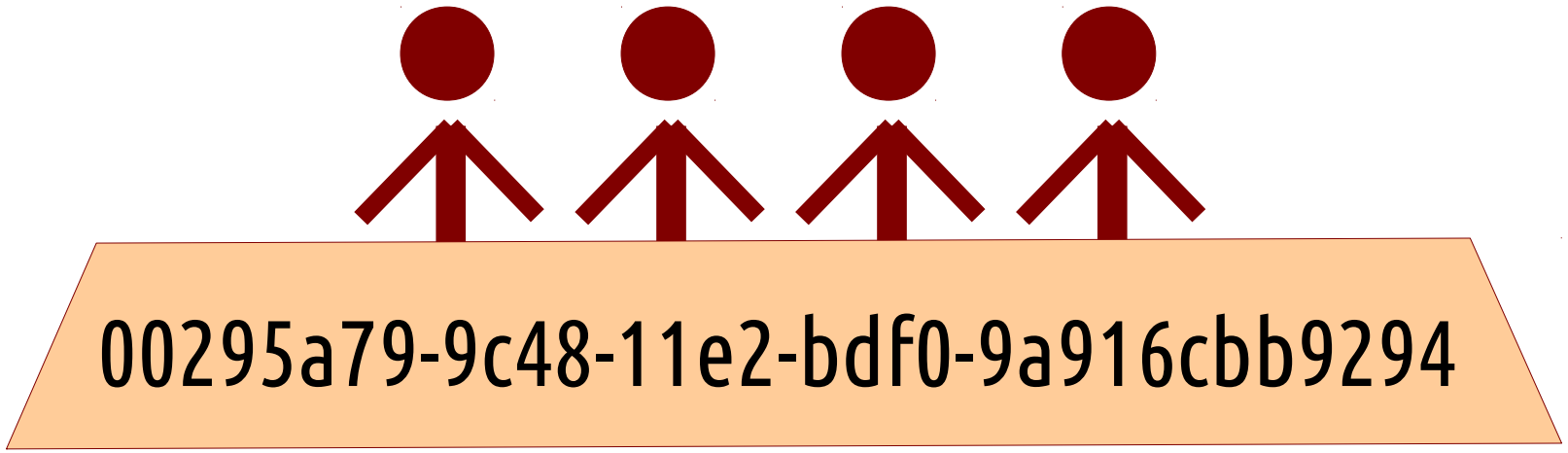


Master / Slave ?

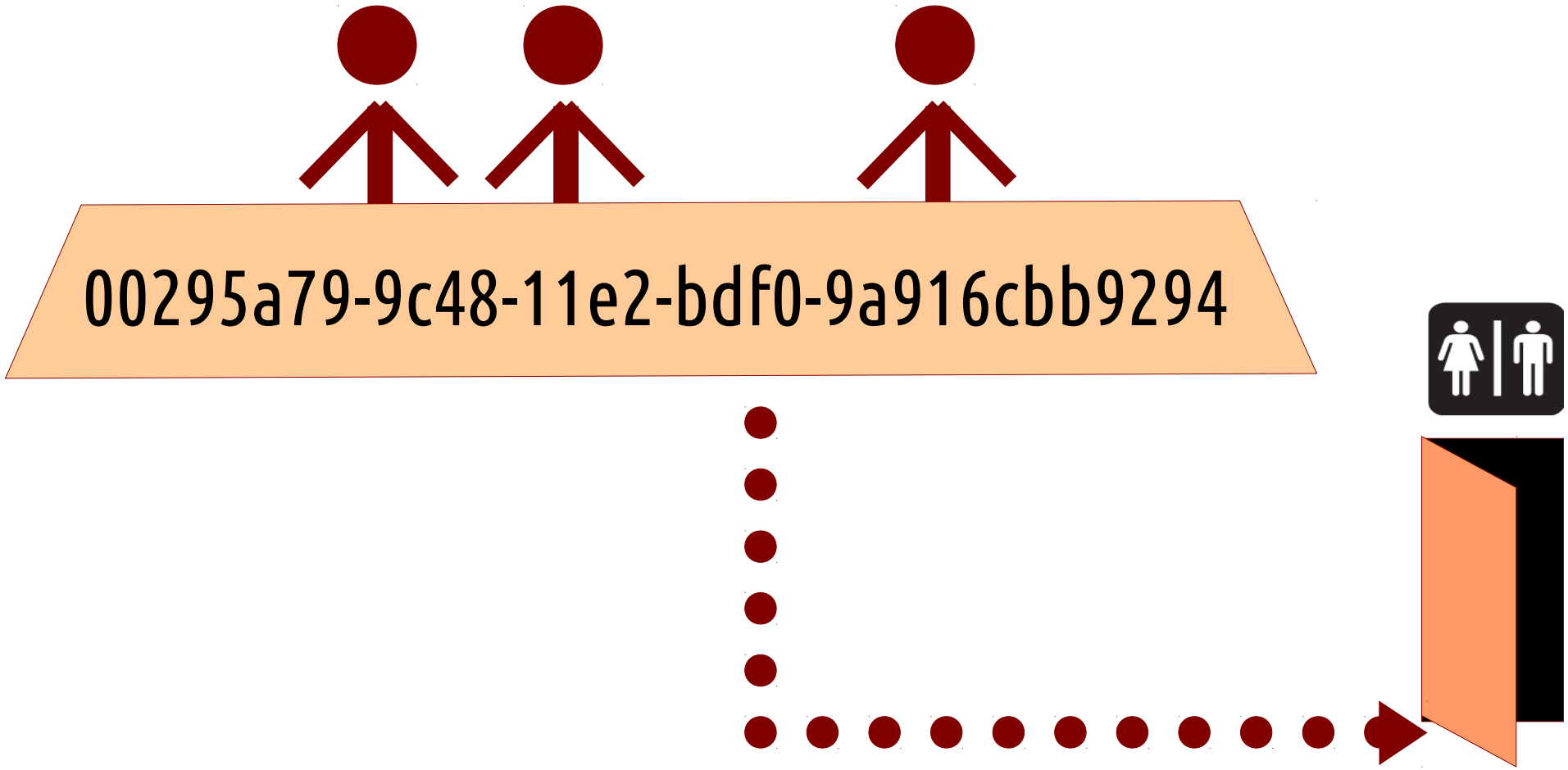
1. Not a node role/function.

2. Is a relation between a node and a client.

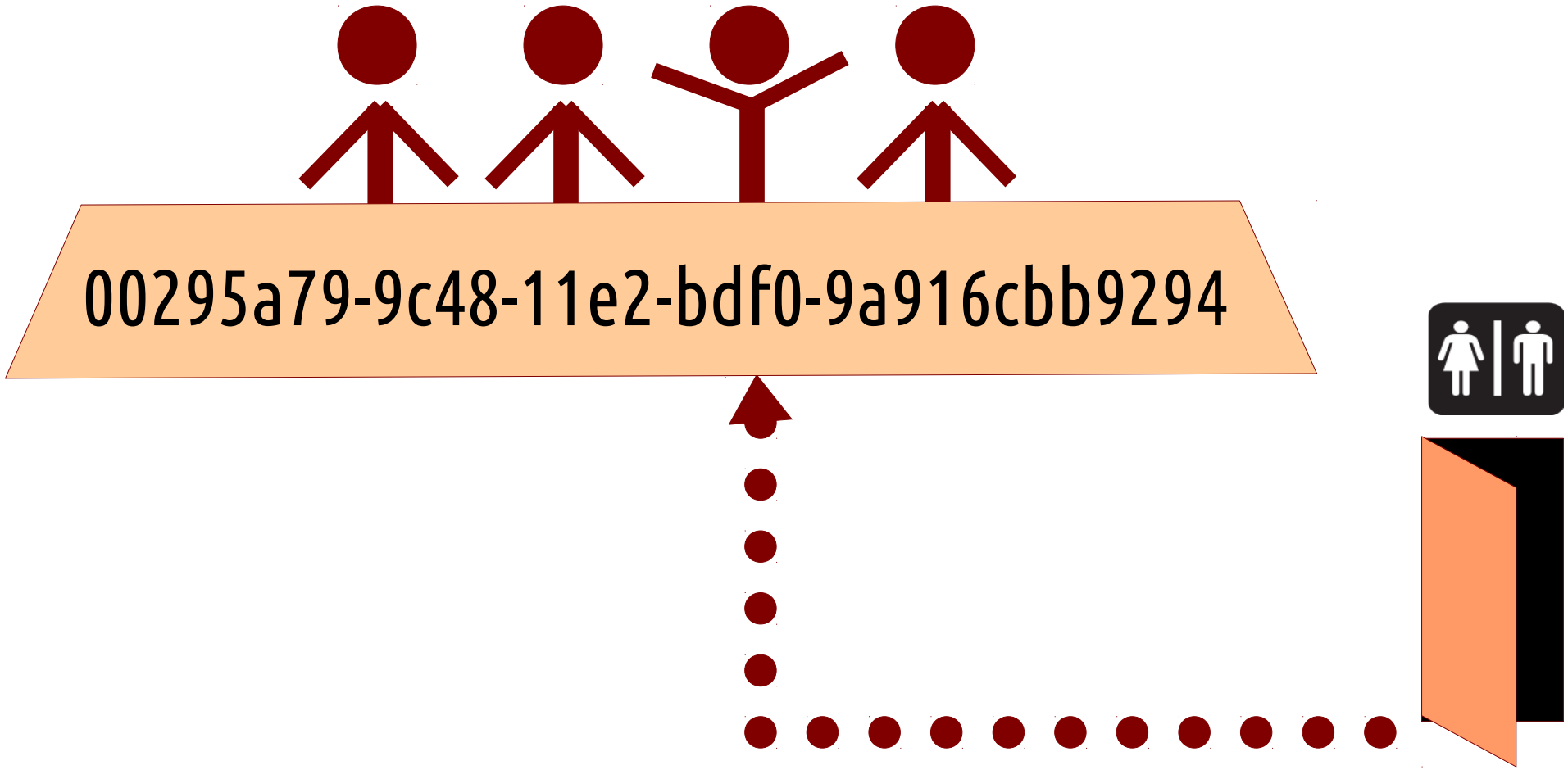
Galera Cluster as a Meeting



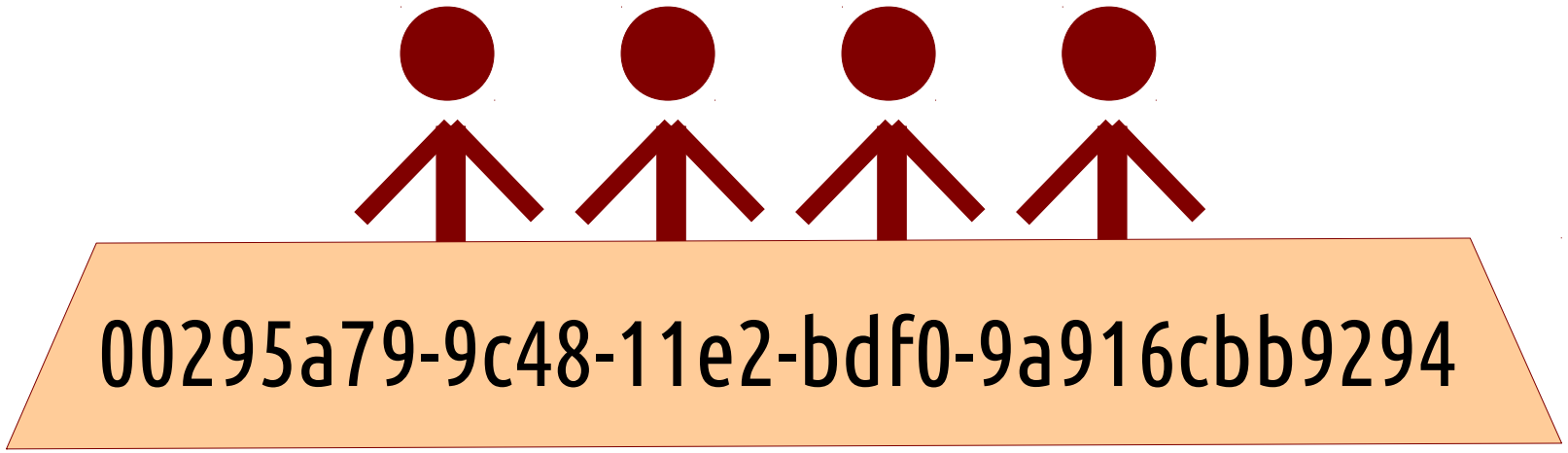
Galera Cluster as a Meeting



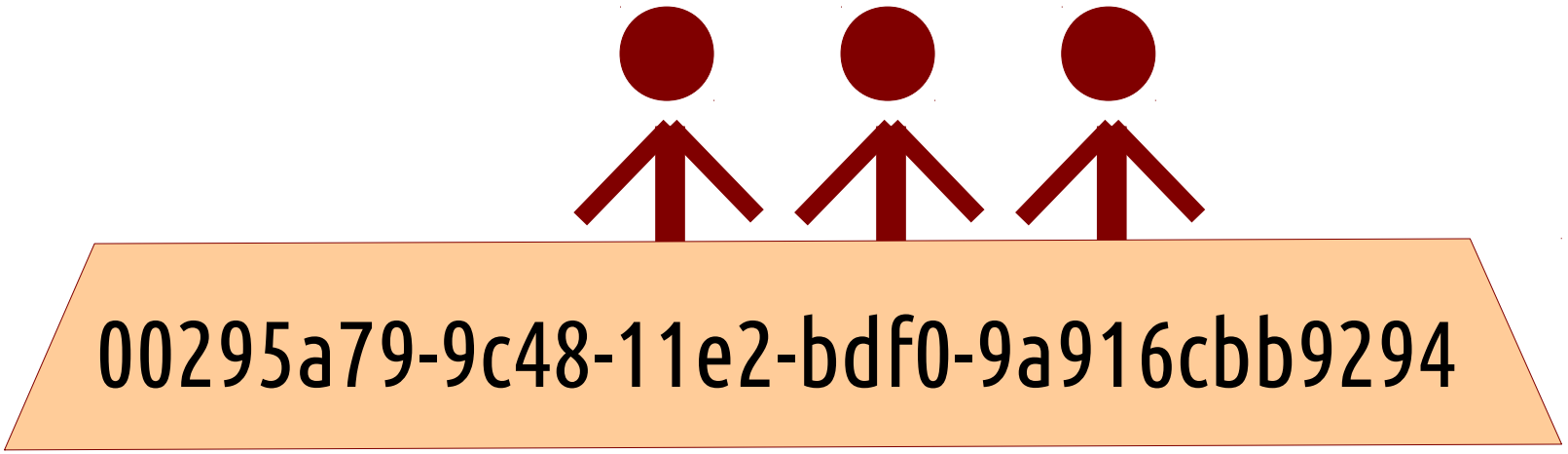
Galera Cluster as a Meeting



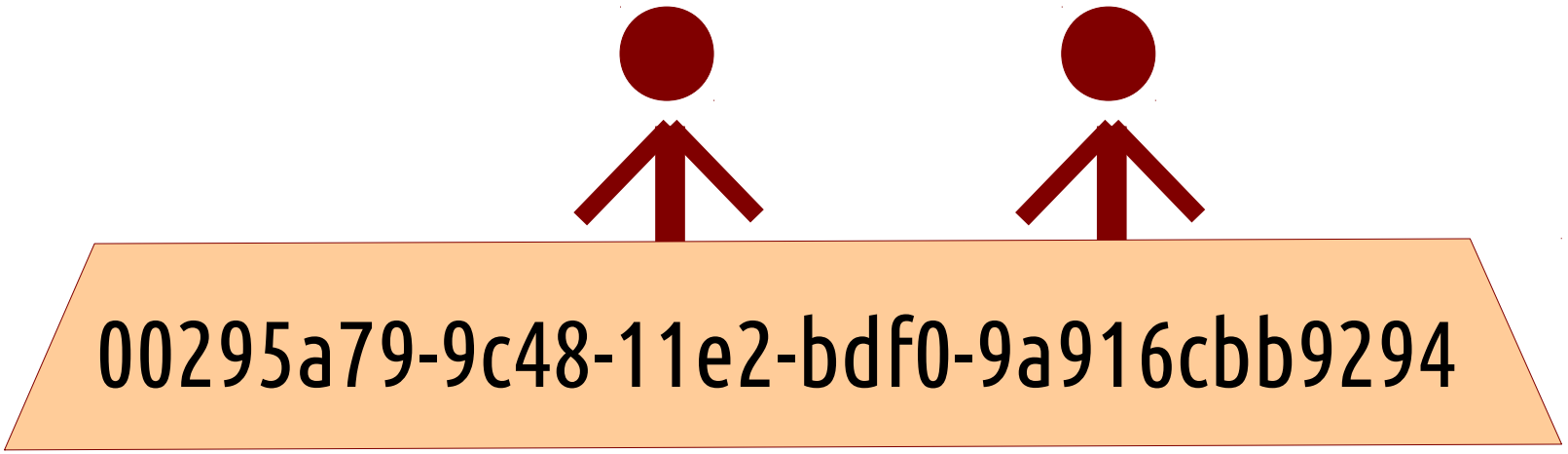
Galera Cluster as a Meeting



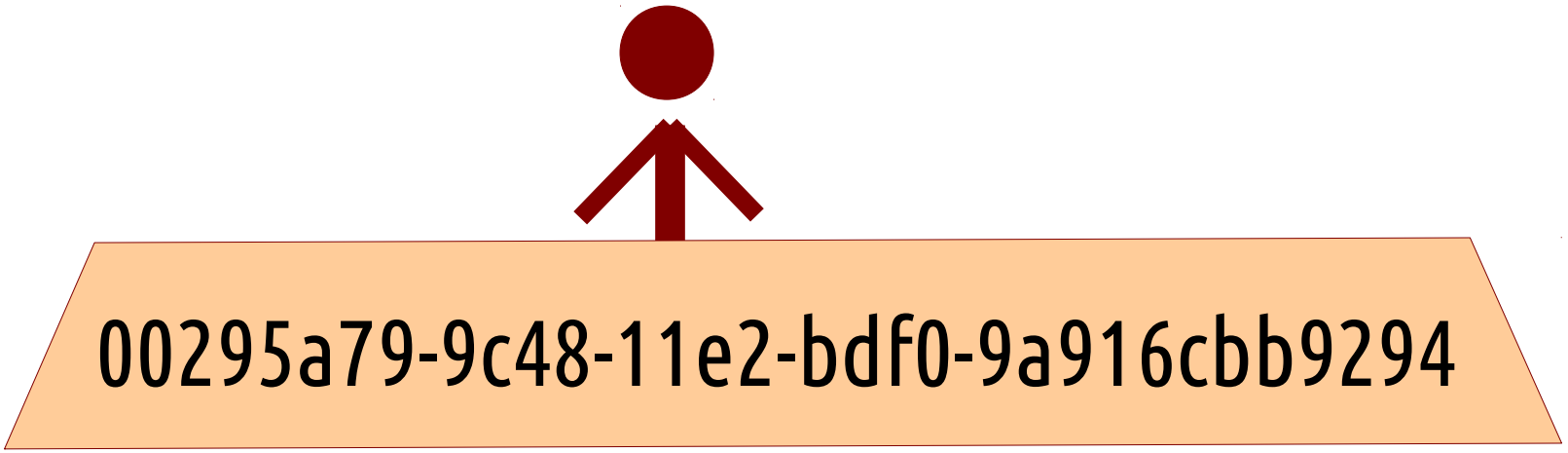
Galera Cluster as a Meeting



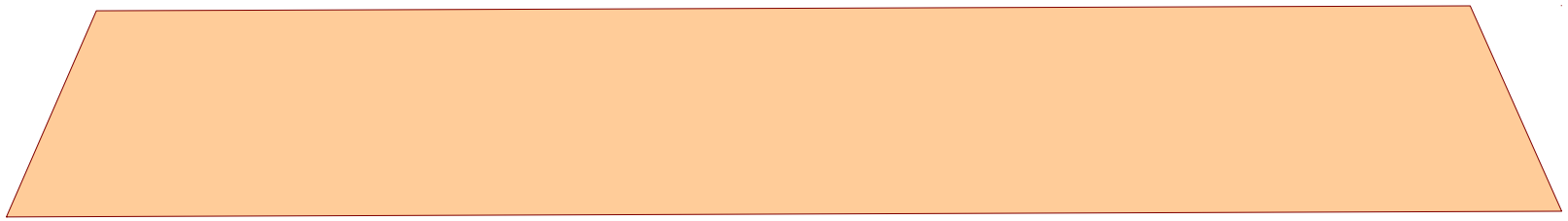
Galera Cluster as a Meeting



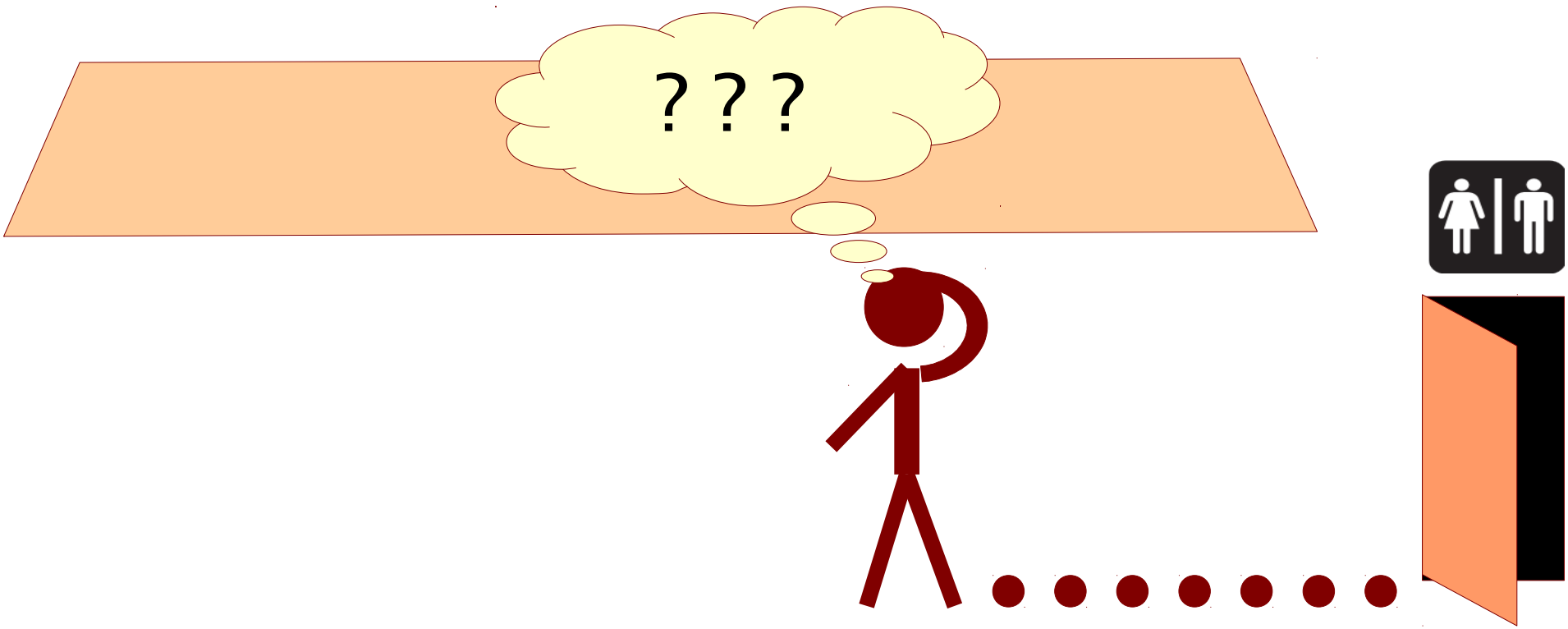
Galera Cluster as a Meeting



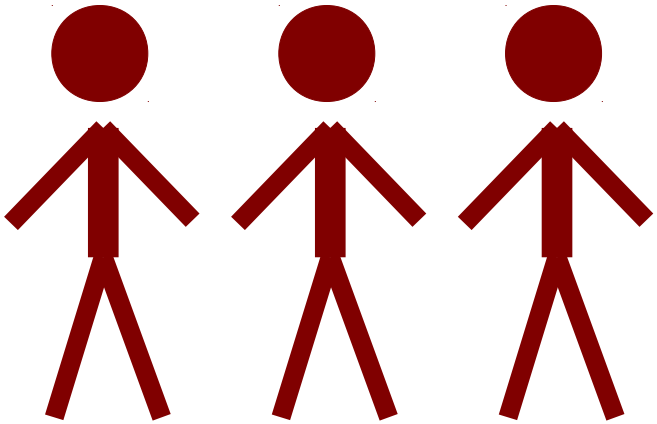
Galera Cluster as a Meeting



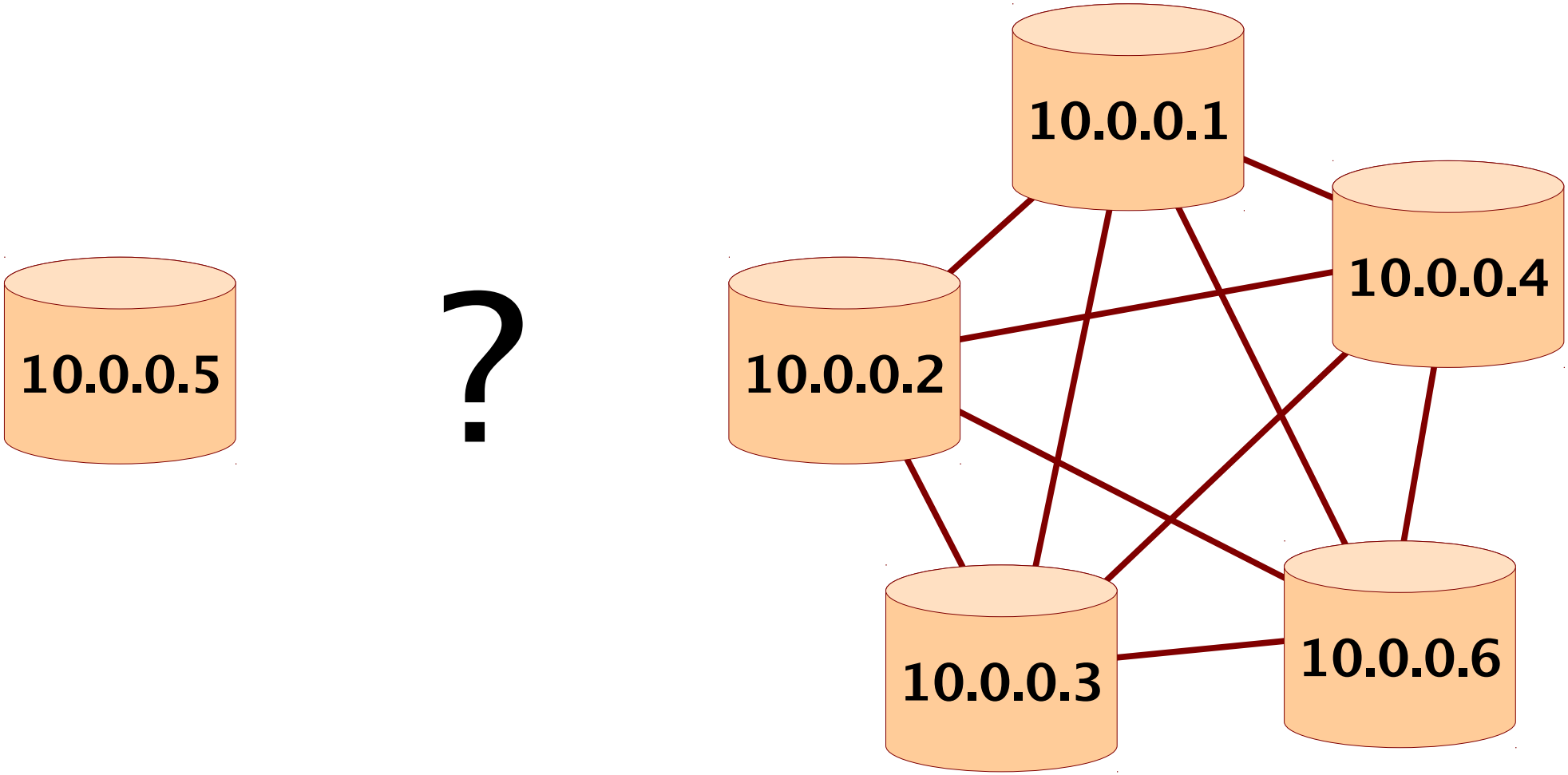
Galera Cluster as a Meeting



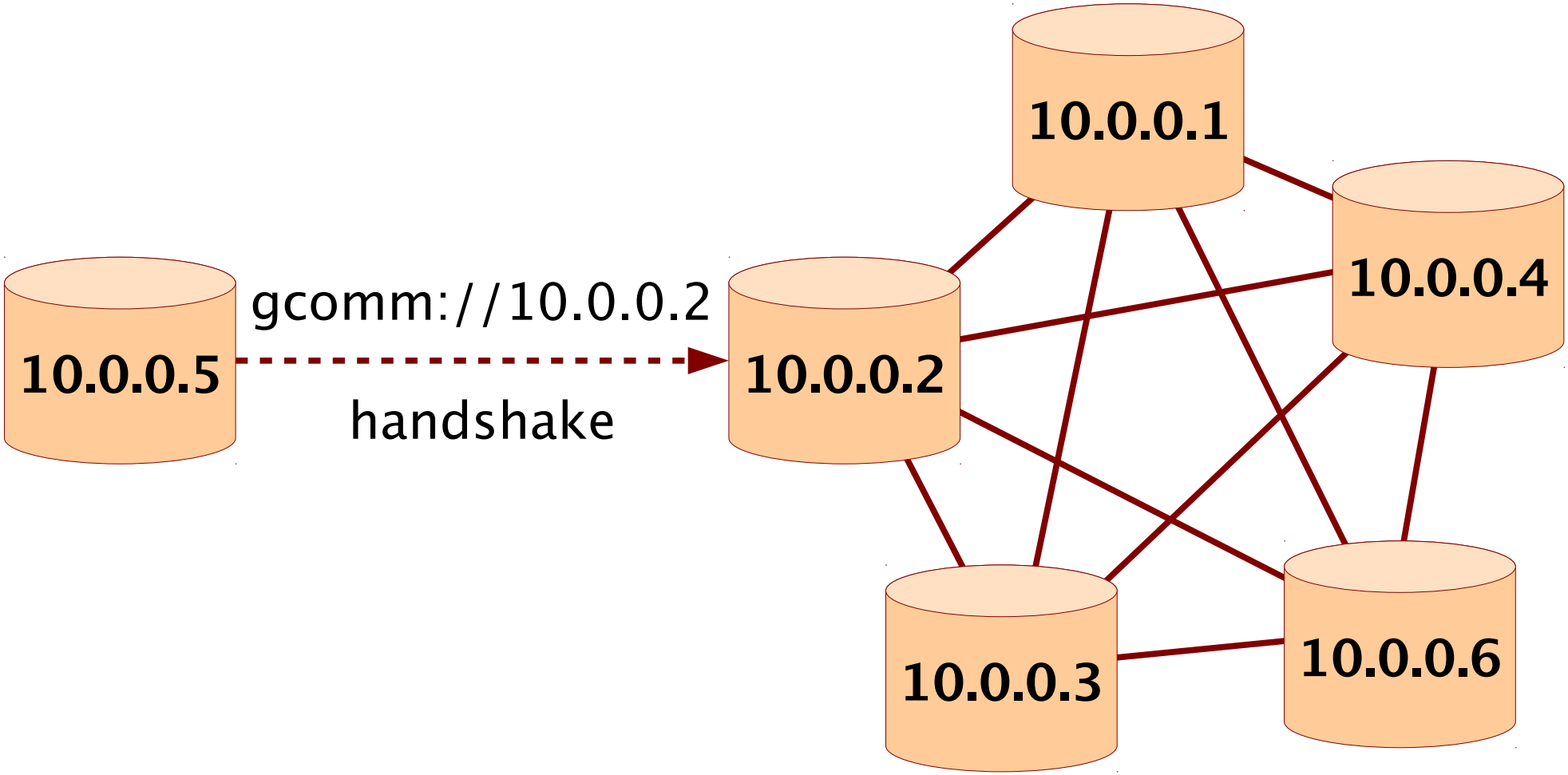
Galera Cluster as a Meeting



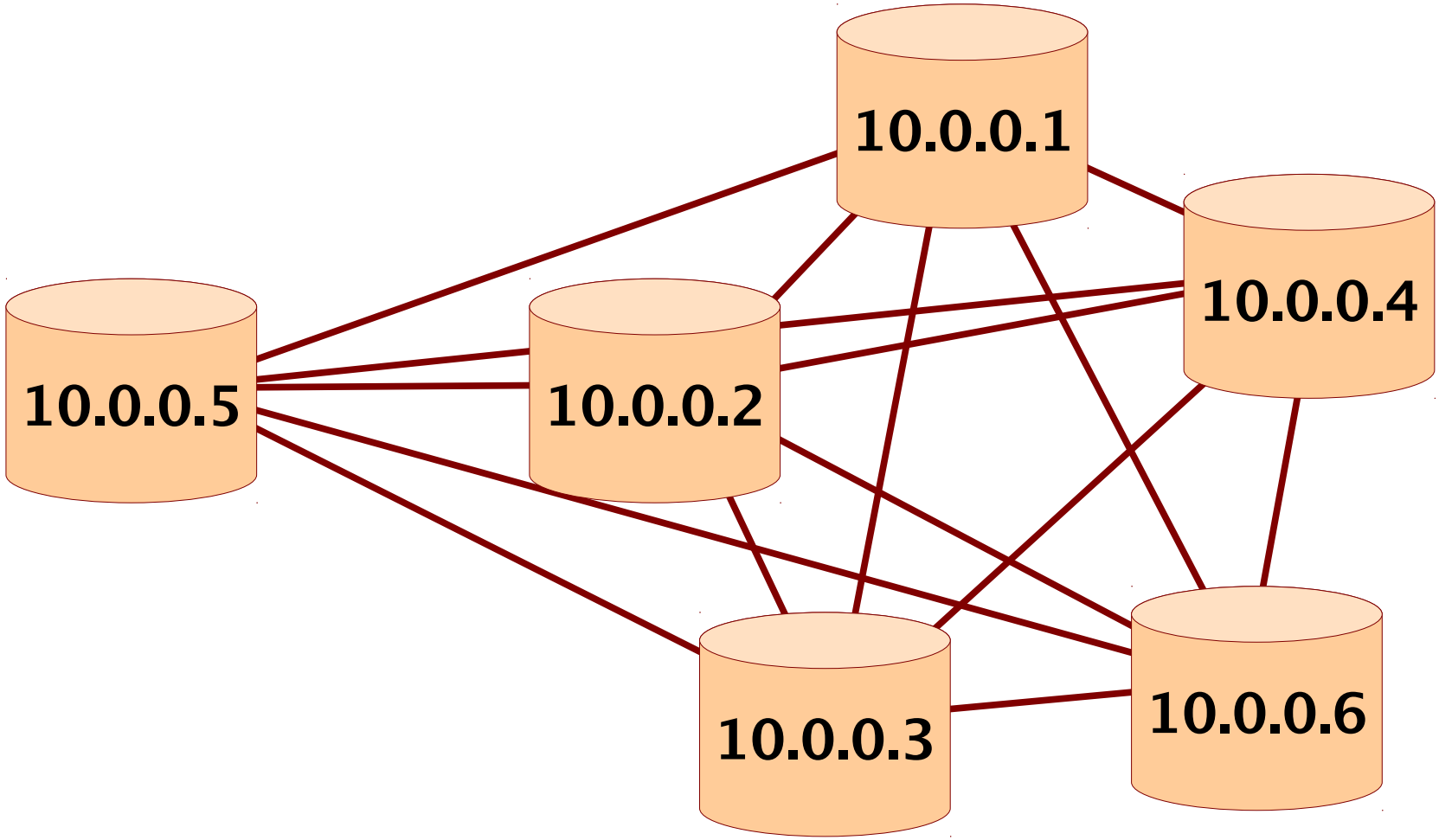
wsrep_cluster_address



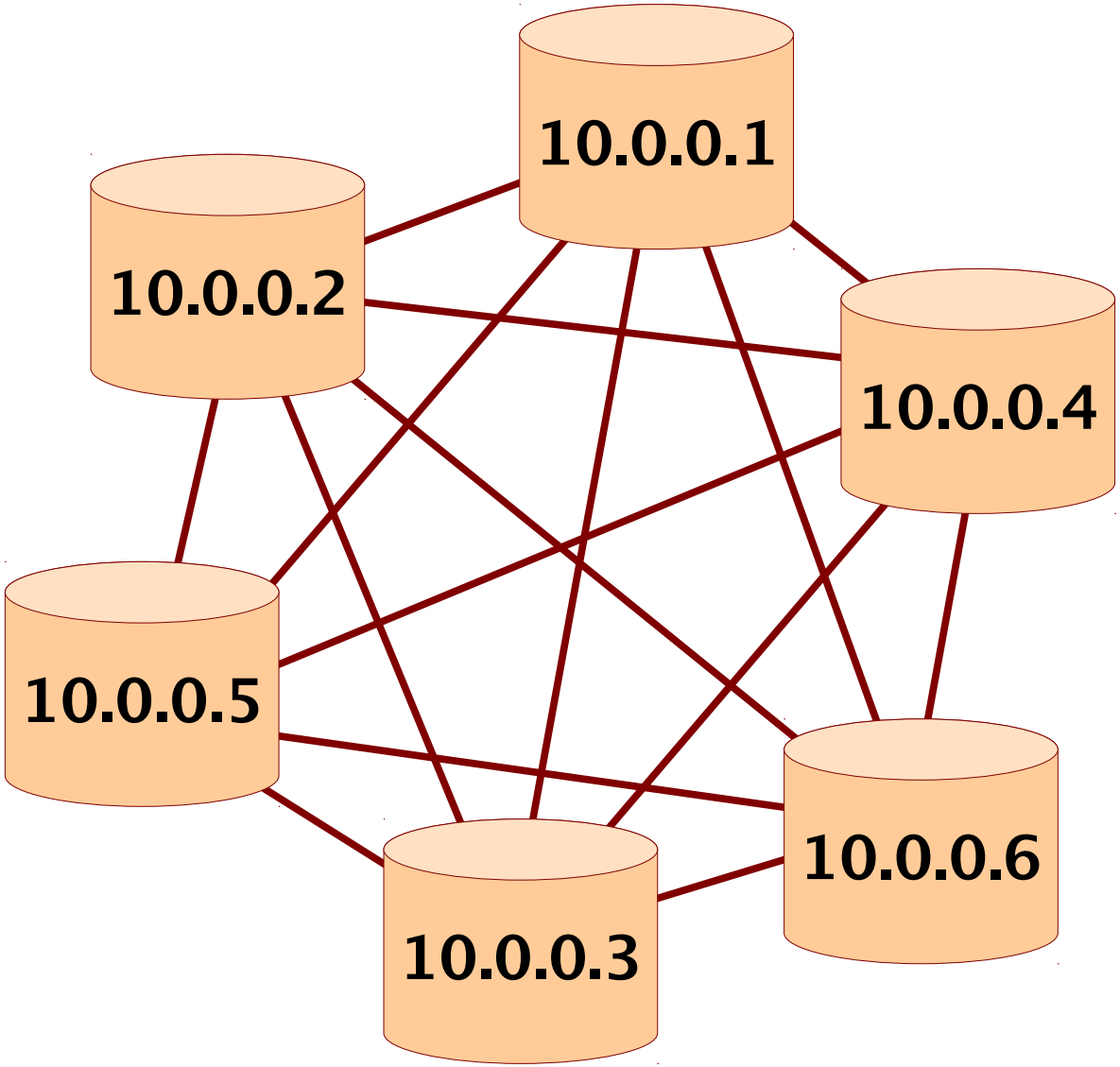
wsrep_cluster_address



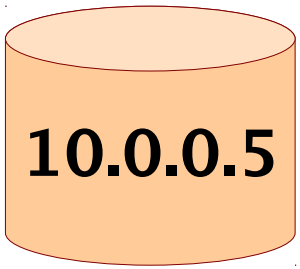
wsrep_cluster_address



wsrep_cluster_address



wsrep_cluster_address



?

wsrep_cluster_address

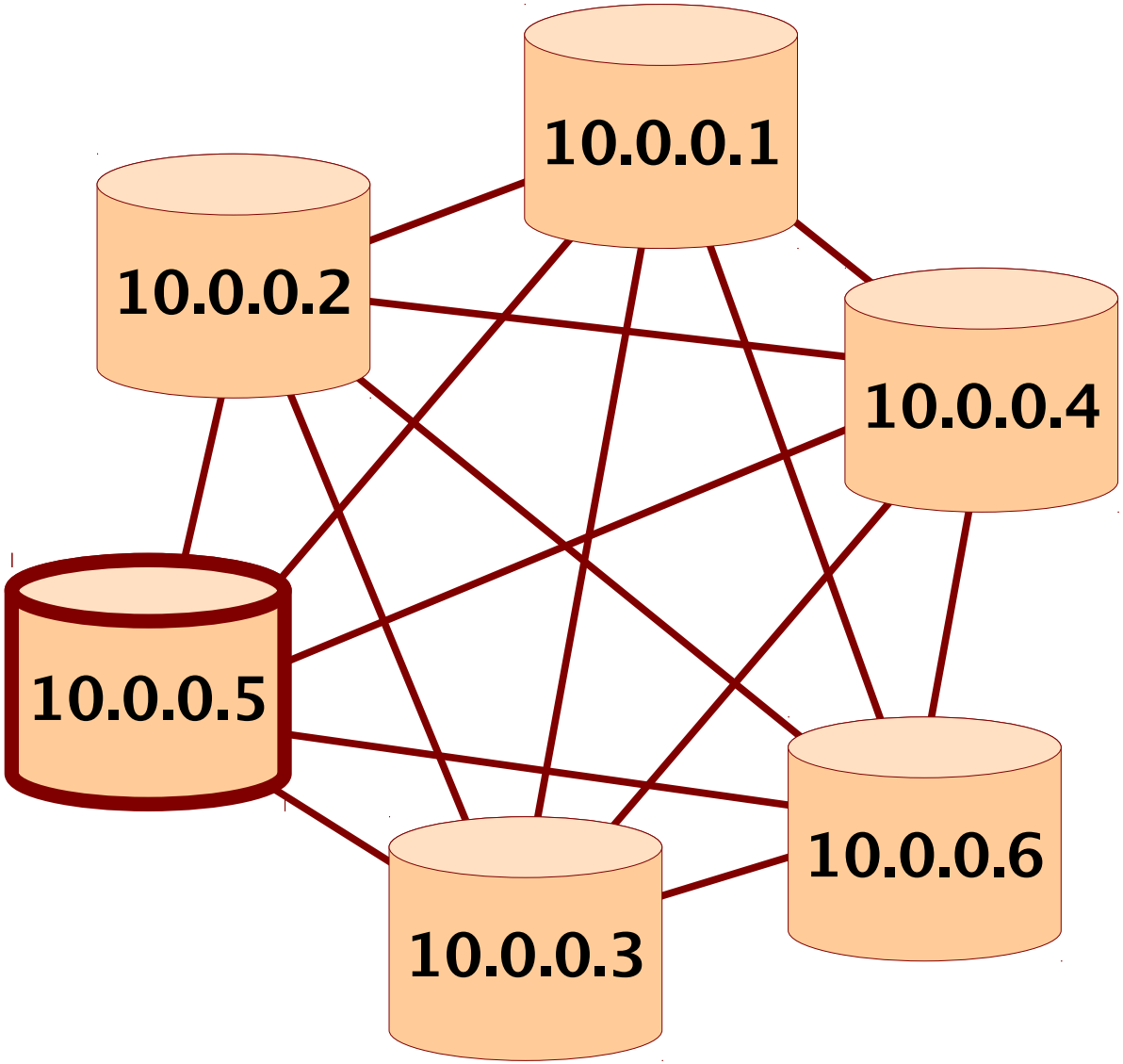
```
wsrep_cluster_address = gcomm://node1,node2
```

=> try to connect to members: node1, node2

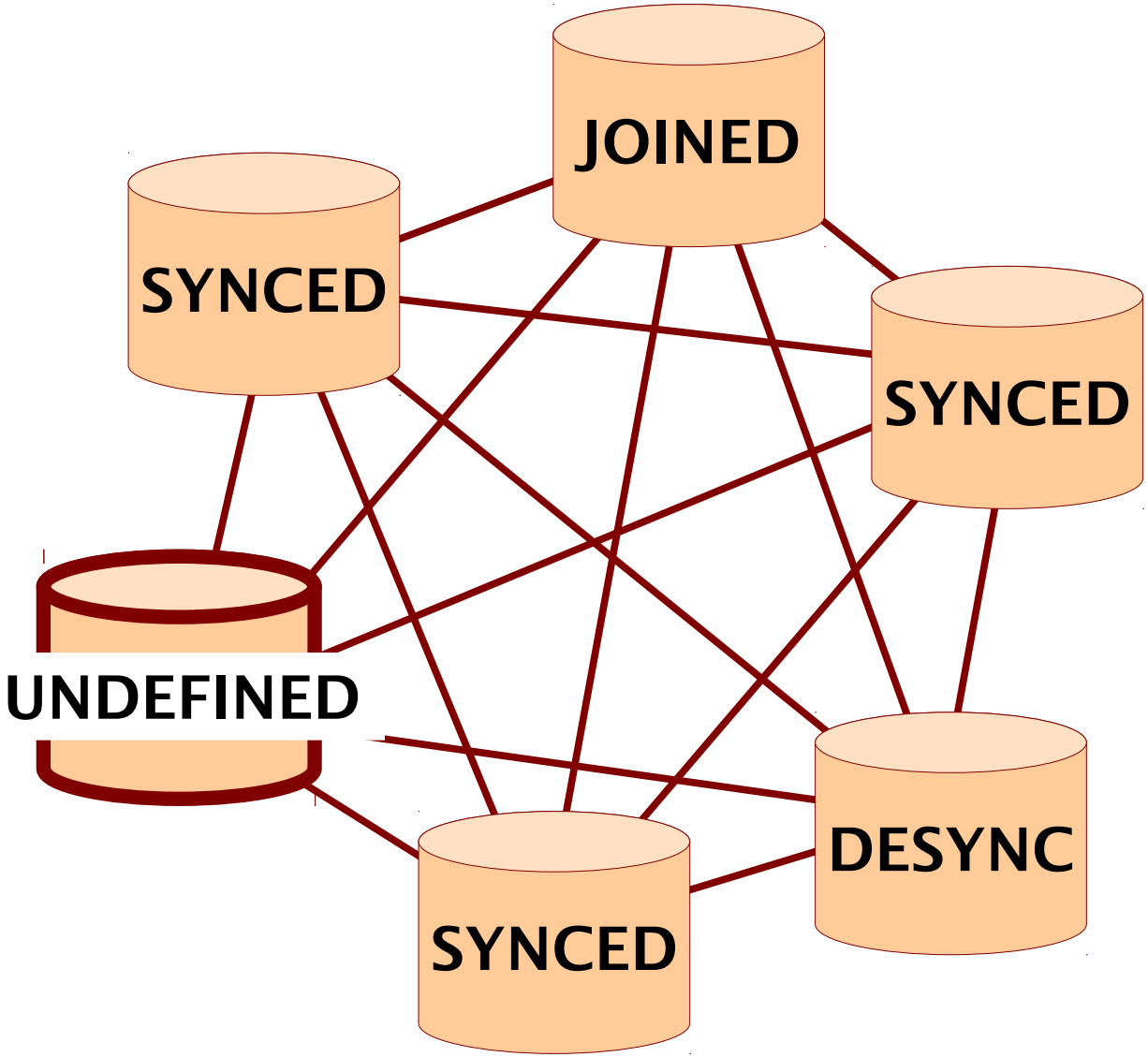
```
wsrep_cluster_address = gcomm://
```

=> no members in the cluster, you are the first one. Start a new cluster.

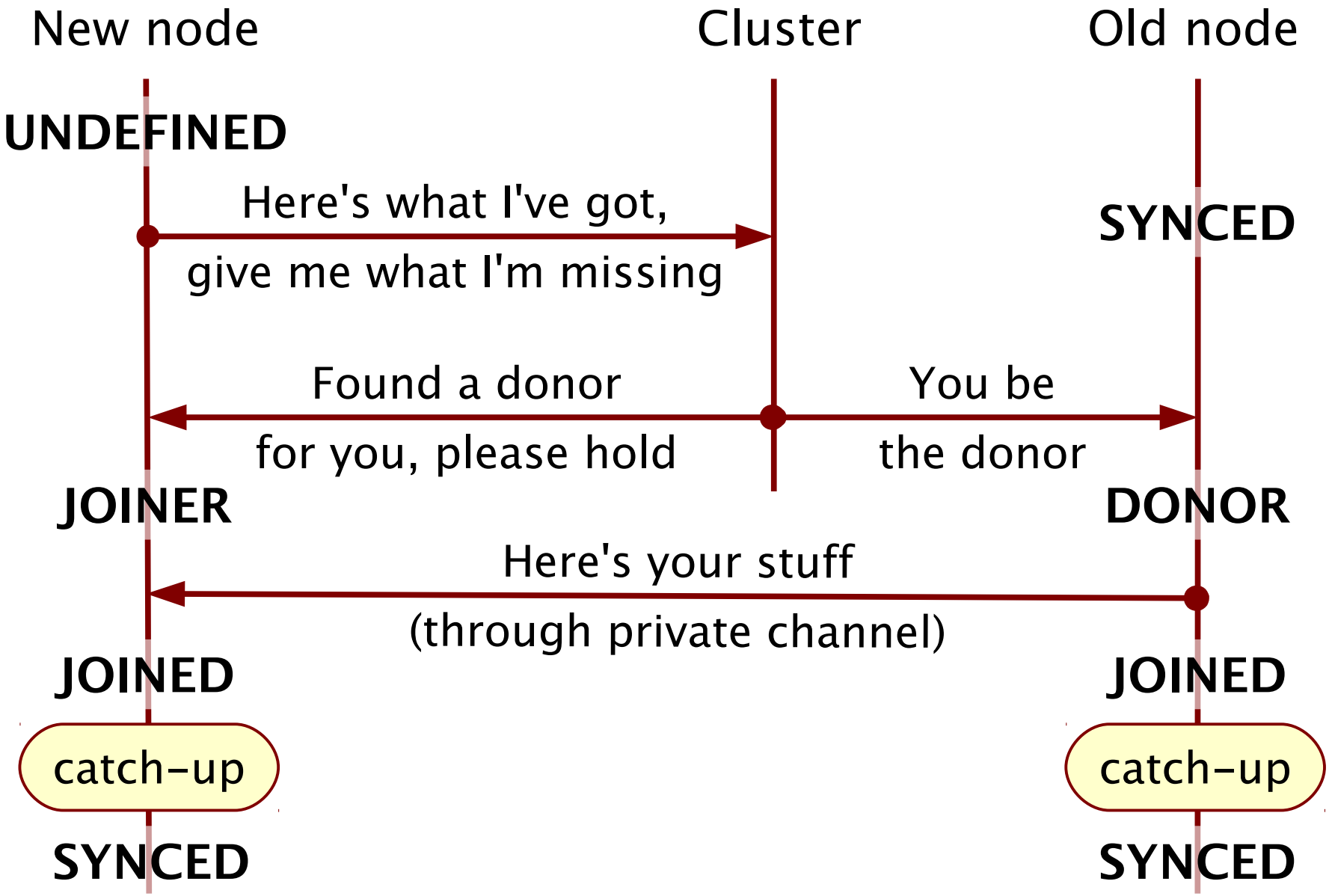
Node Synchronization (State Transfer)



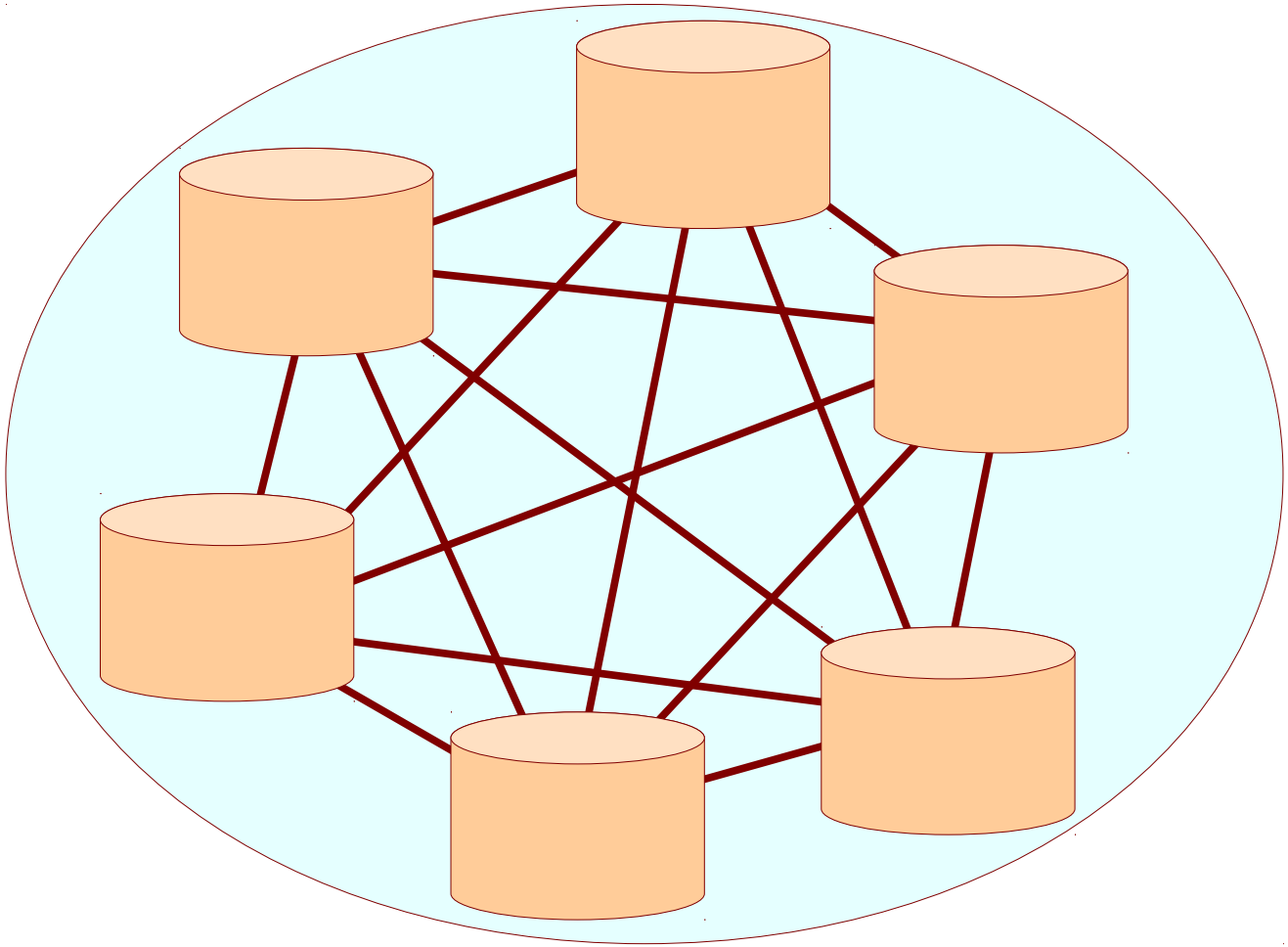
Node Synchronization (State Transfer)



Node Synchronization (State Transfer)

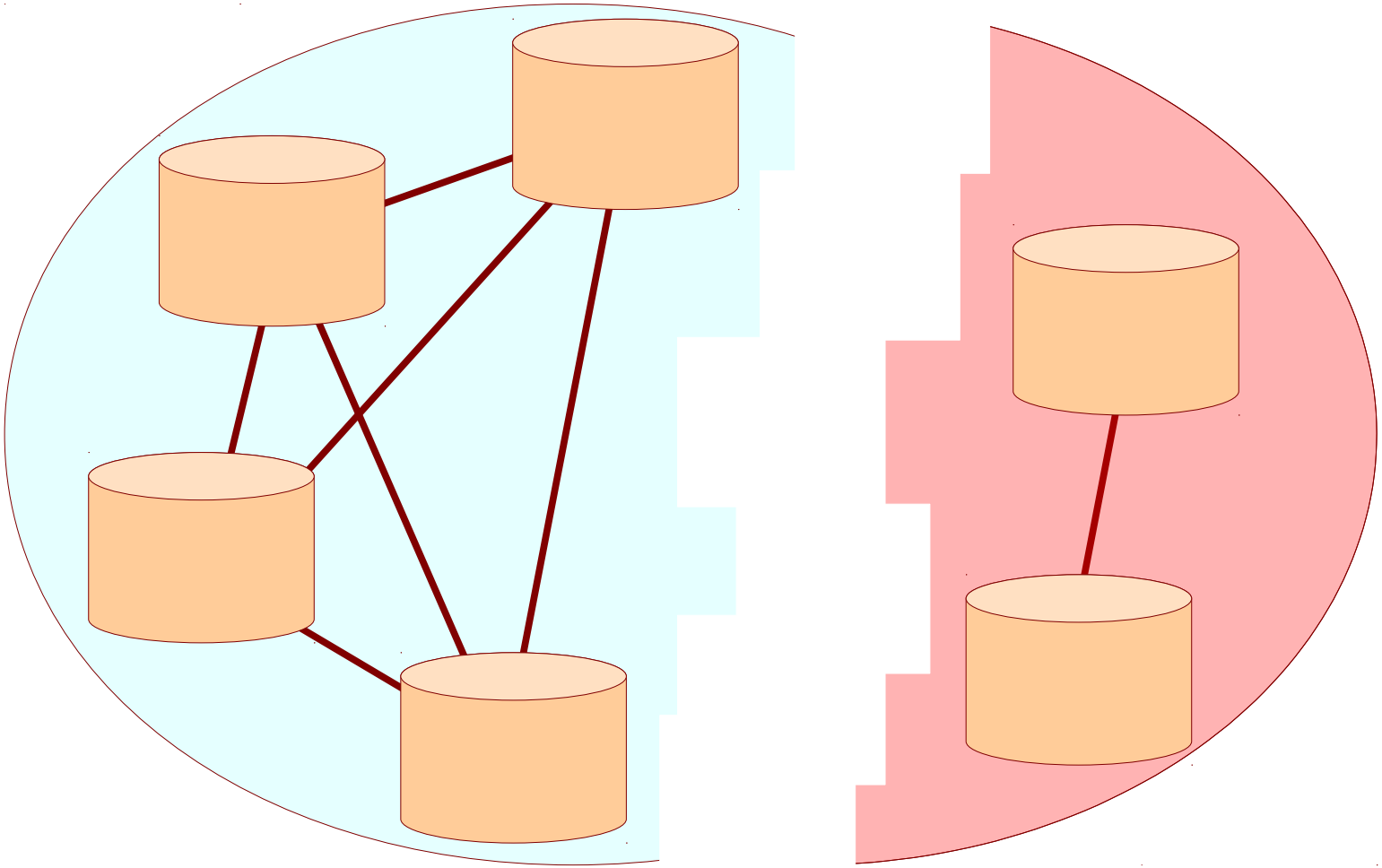


Primary Component

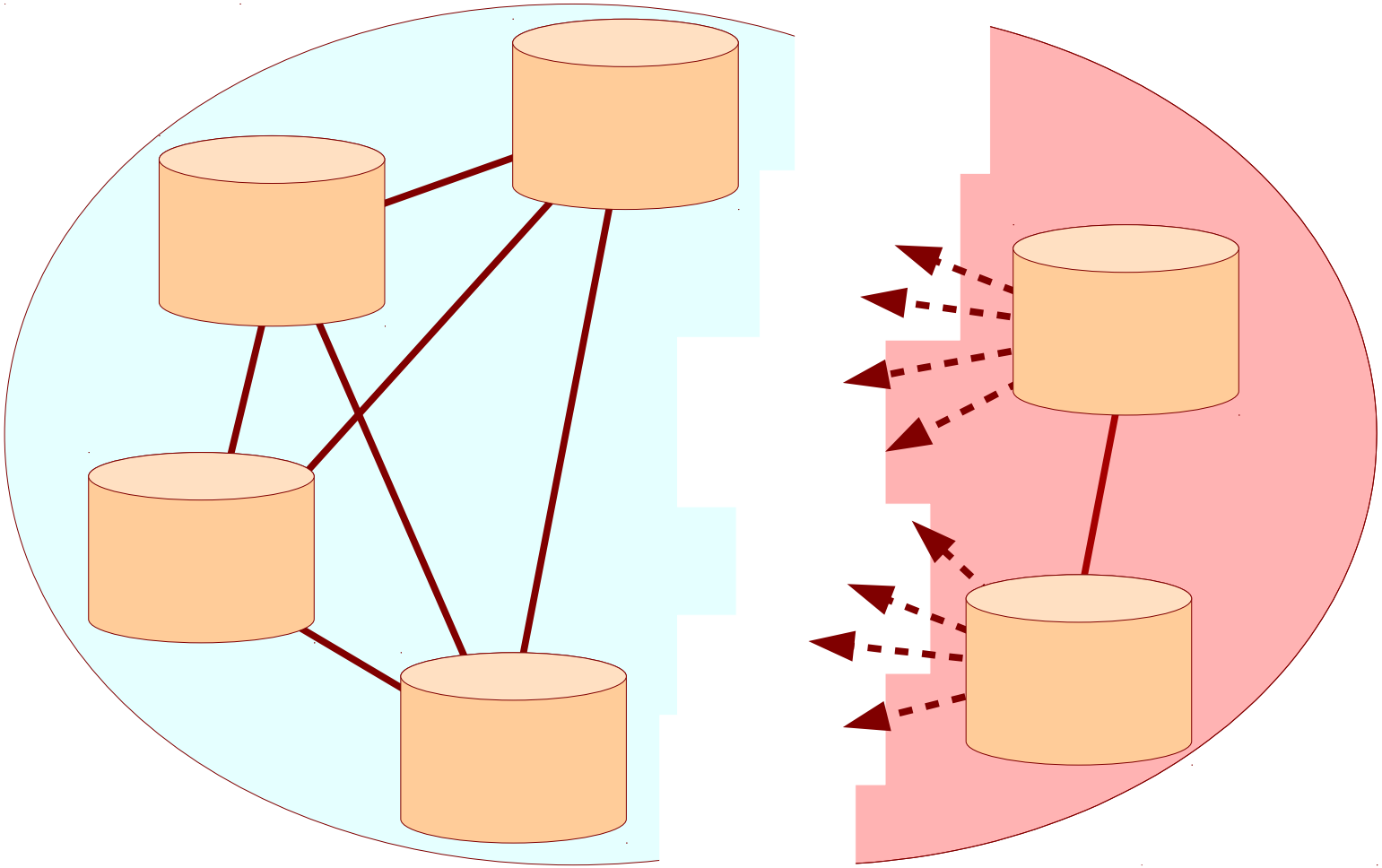


PRIMARY

Primary Component



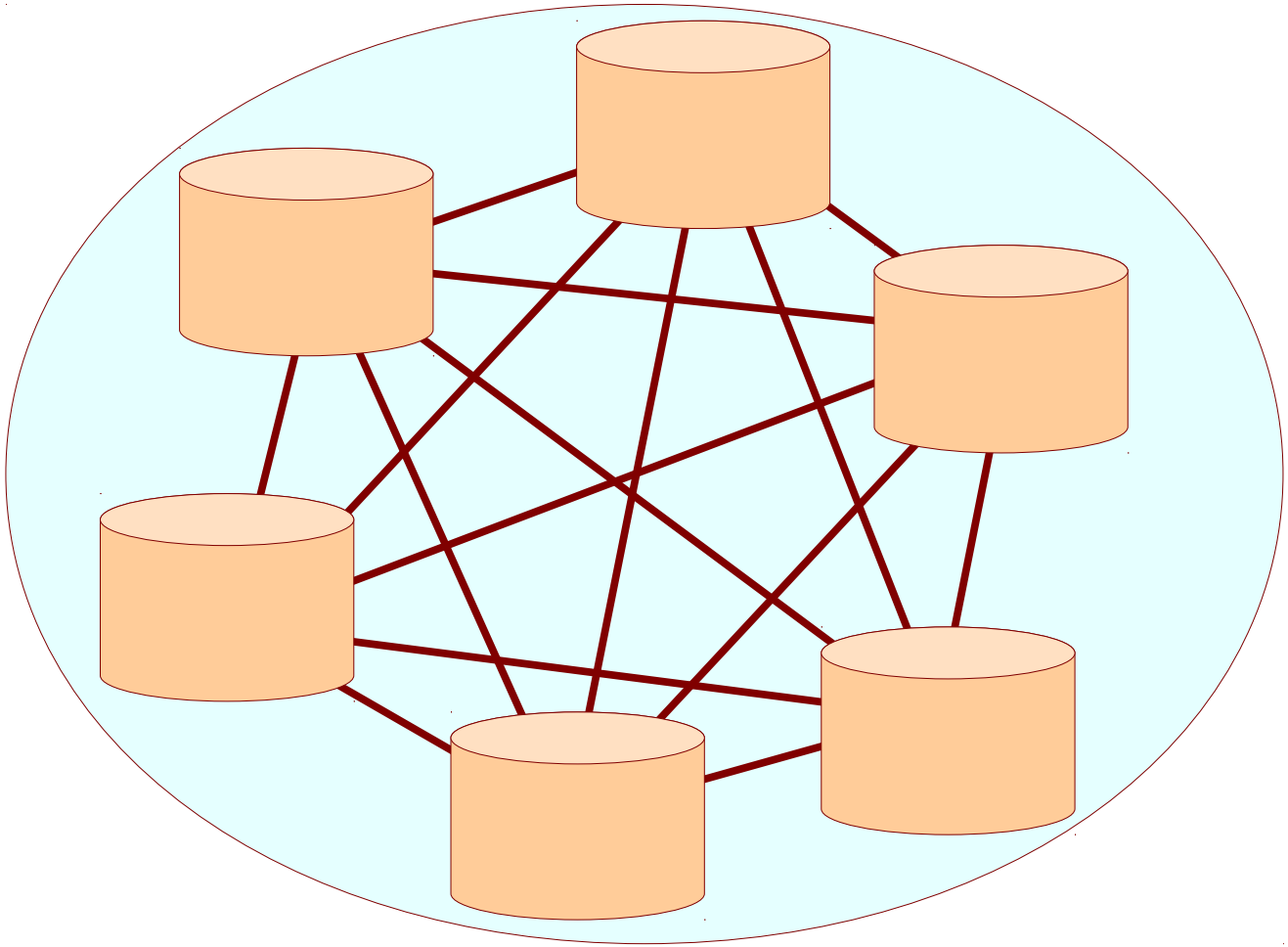
Primary Component



PRIMARY
keeps on working

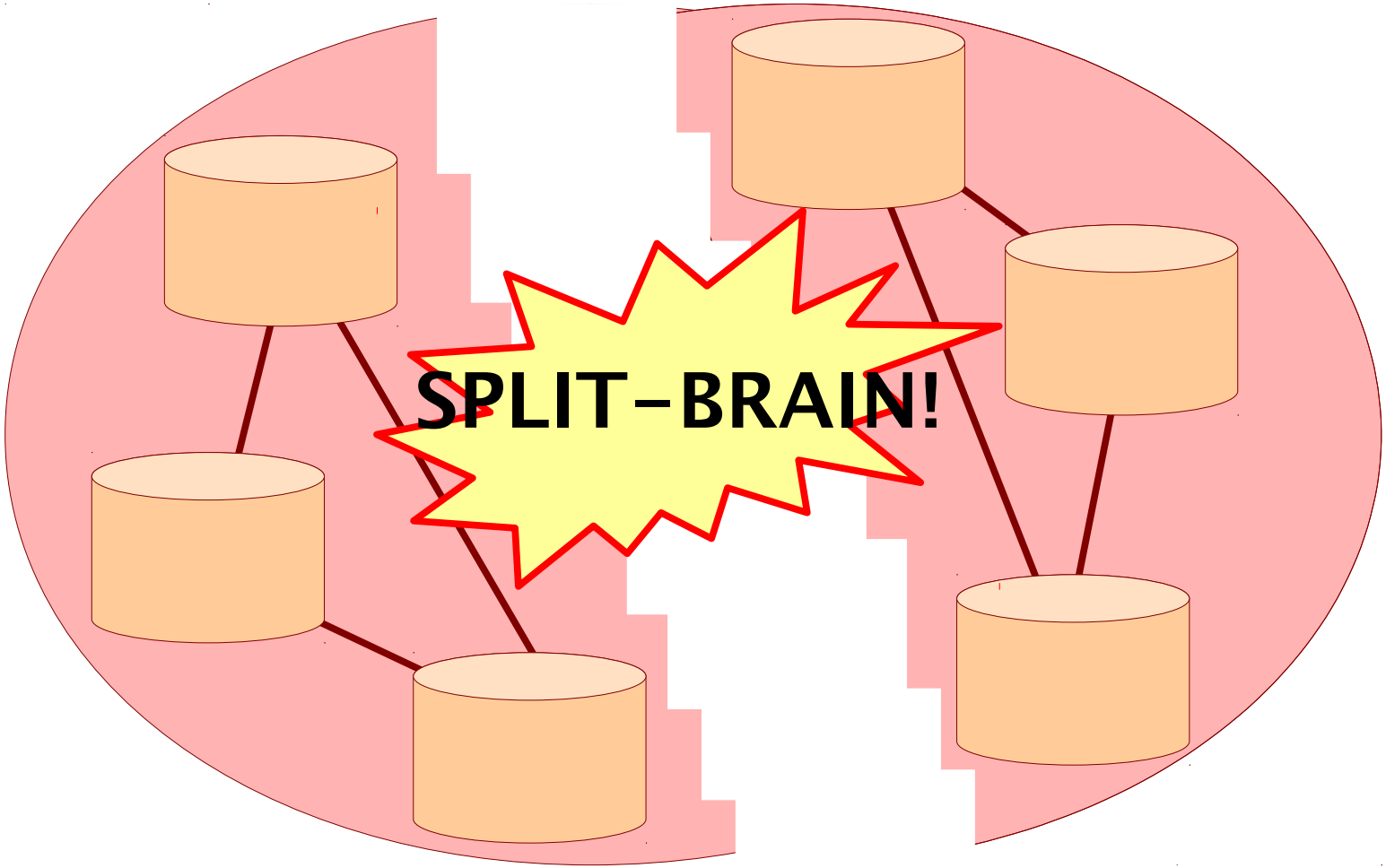
NON-PRIMARY
tries to reconnect

Primary Component



PRIMARY

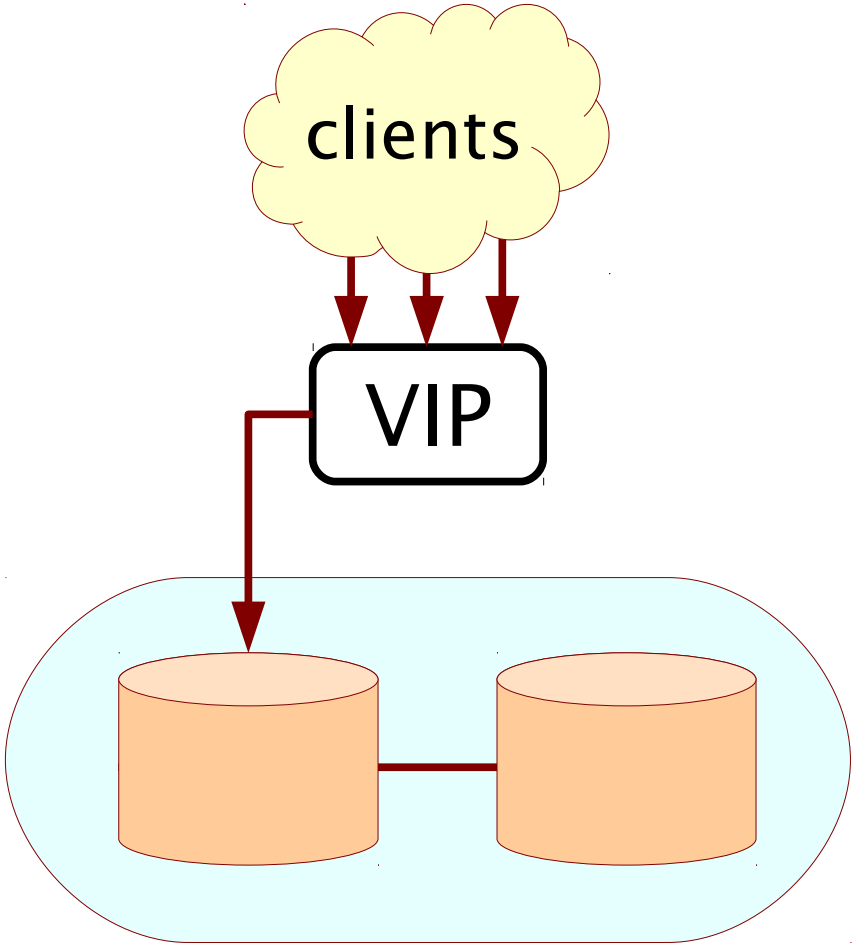
Primary Component



NON-PPRIMARY

NON-PRIMARY

2-node Cluster and Split Brain




```
wsrep_provider_options="pc.ignore_sb"
```

2-node Cluster and Split Brain

Galera replication can be used in every manner traditional asynchronous master-slave replication is.

It implements a SUPERSET

of traditional replication functionality

How Synchronous is Galera?

1. Synchronous penalty.

2. Slave lag.

Galera Synchronous Penalty?

The only thing Galera does synchronously is copying of data buffer to all cluster members on `COMMIT` command from client.

=> ~1 RTT added latency

Galera Synchronous Penalty?

~1 RTT added latency

=>

Connection throughput = 1/RTT trx/sec

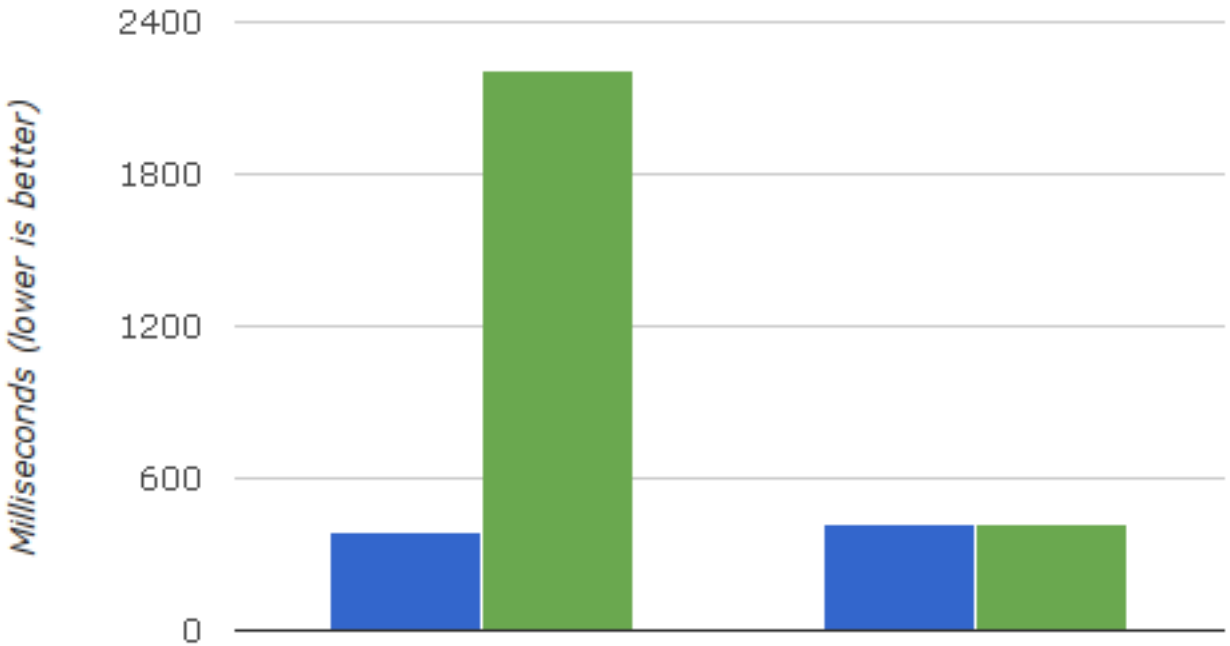
=>

Total throughput = 1/RTT trx/sec × #connections

Galera Synchronous Penalty in WAN (EC2)

- sysbench client at us-east
- sysbench client at eu-west

95% latencies



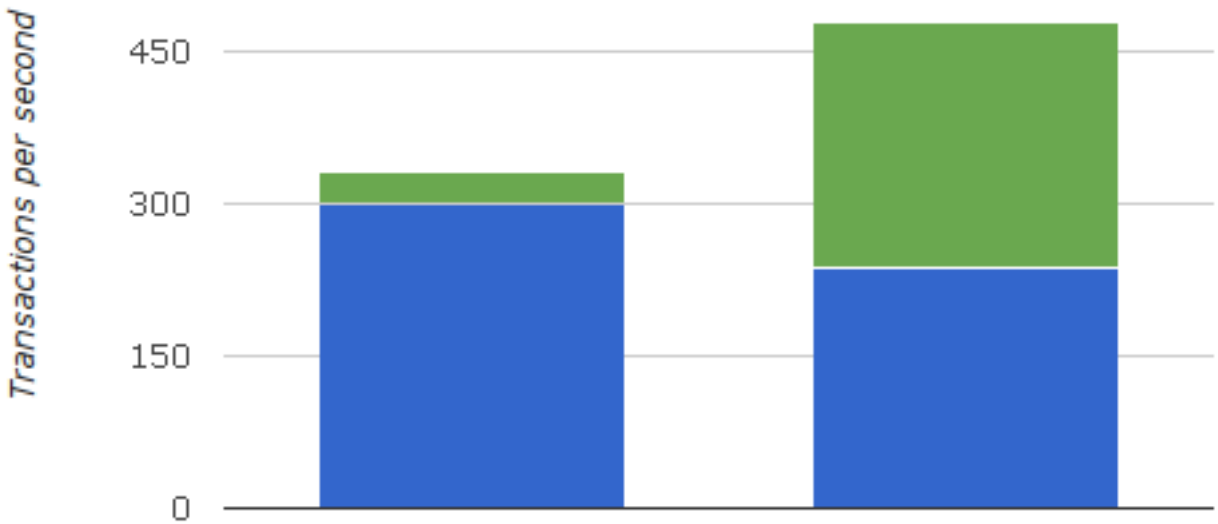
Both clients connect to a standalone server at us-east accessibility zone

2-node Galera cluster: us-east client connects to us-east node, eu-west client connects to eu-west node

Galera Synchronous Penalty in WAN (EC2)

- sysbench client at us-east
- sysbench client at eu-west

Throughput



Both clients connect to a standalone server at us-east accessibility zone

2-node Galera cluster: us-east client connects to us-east node, eu-west client connects to eu-west node

Galera Synchronous Penalty?

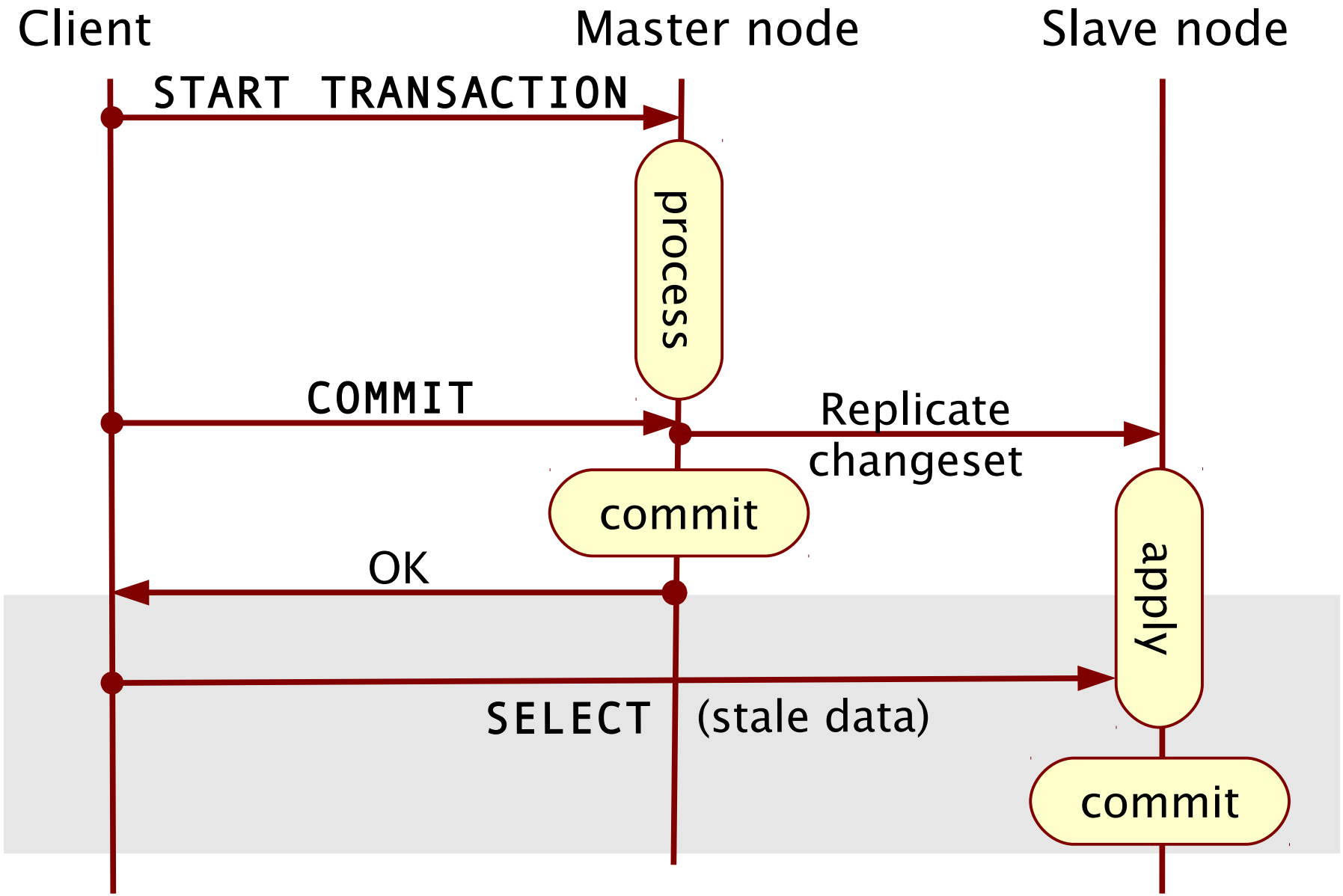
Still:

CALLAGHAN'S LAW:

A given row can't be modified more often than 1/RTT times a second

(discovered by Mark Callaghan)

Slave lag in Galera?



Questions?

Thank you for listening!
Happy Clustering :-)