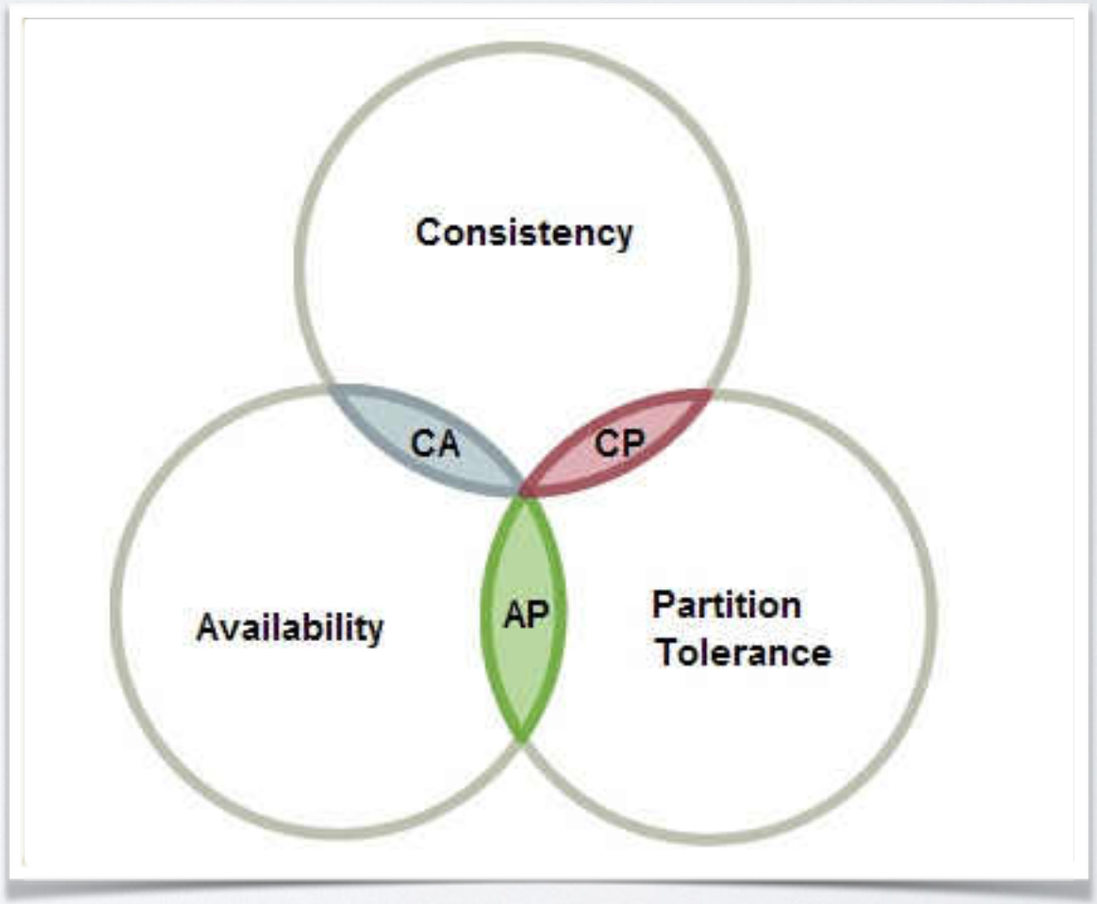
ZEPPELIN

High Available KV Storage Service

OVERVIEW

- CAP High Available
- Distributed KV storage



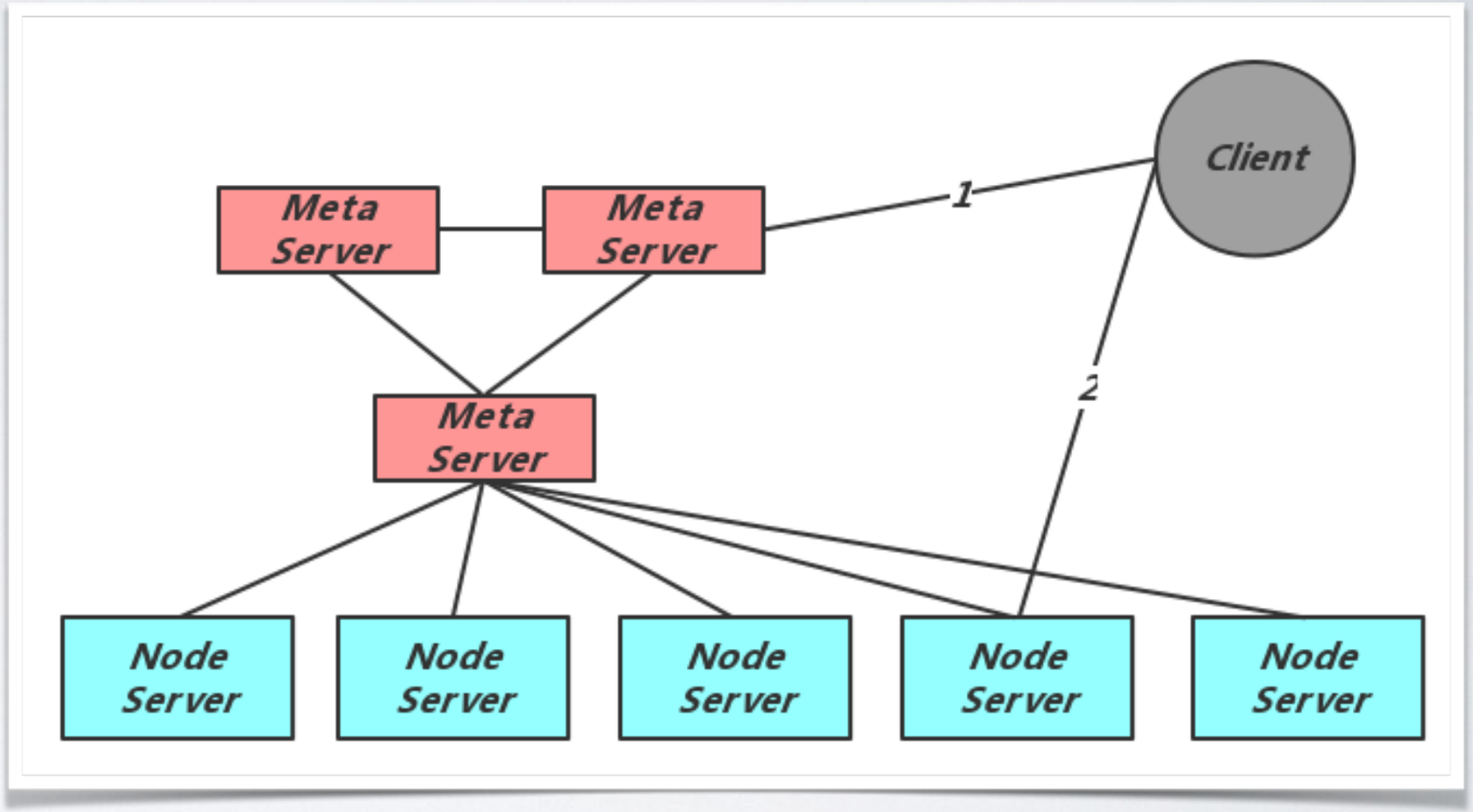
- Online Searching 600,000 QPS
  - 24 Physical Machine
  - 3 meta server 96 node server
  - 40+ tables
  - Highest Table Total query 500,000,000,000 times

OVERVIEW

- Interface Supported: SET, GET, DEL, MSET, MGET, INC
- TTL Supported
- Hashtag Supported

OVERVIEW

- Client pull meta Info
- Calculate partition
- Find node ip in meta info
- Send request to corresponding node server



NODE SERVER

- Data Distribution & Replication
- Thread Model
- Synchronization

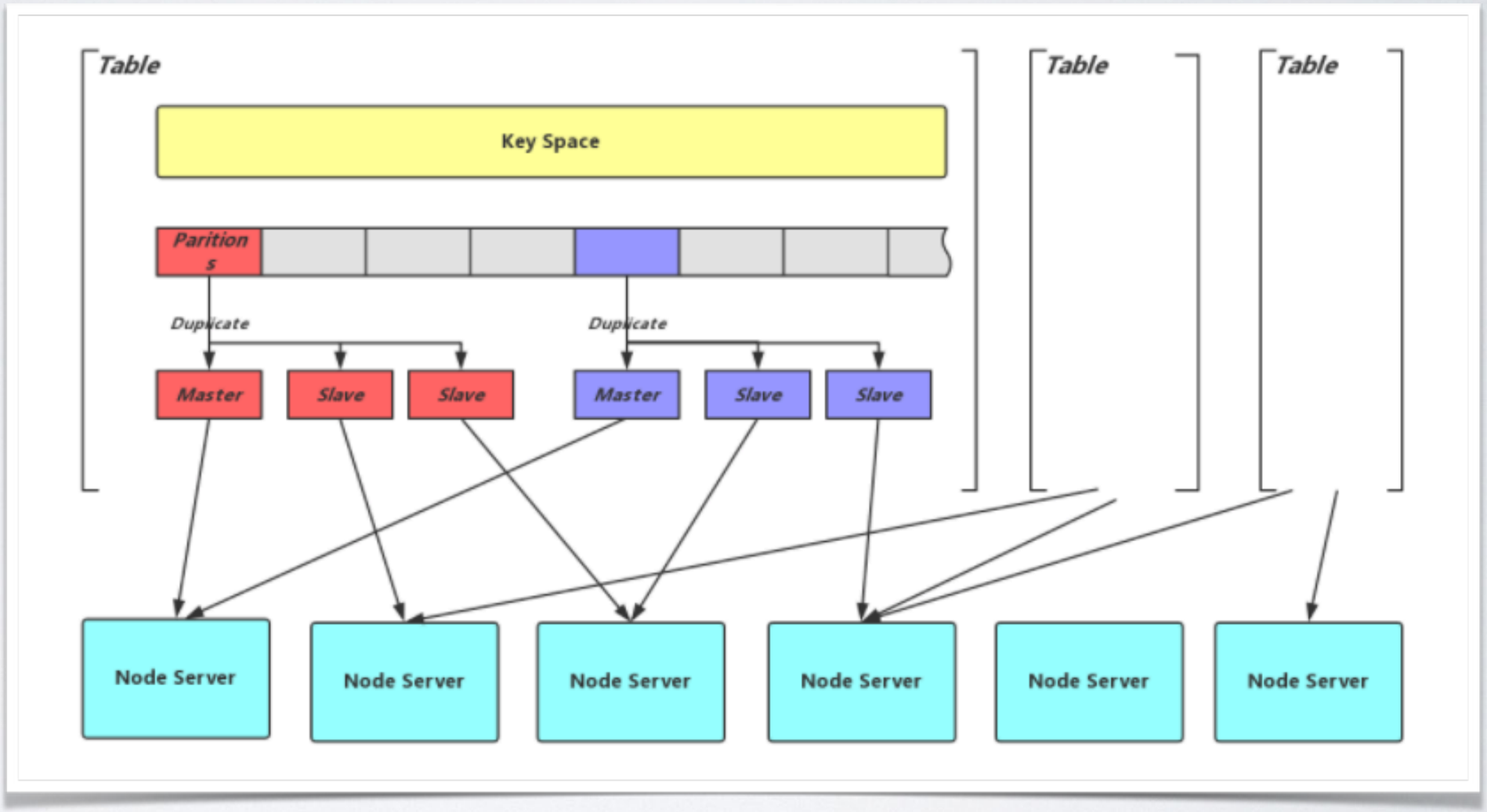
NODE SERVER

- Data Distribution & Replication

NODE SERVER

Data Distribution & Replication

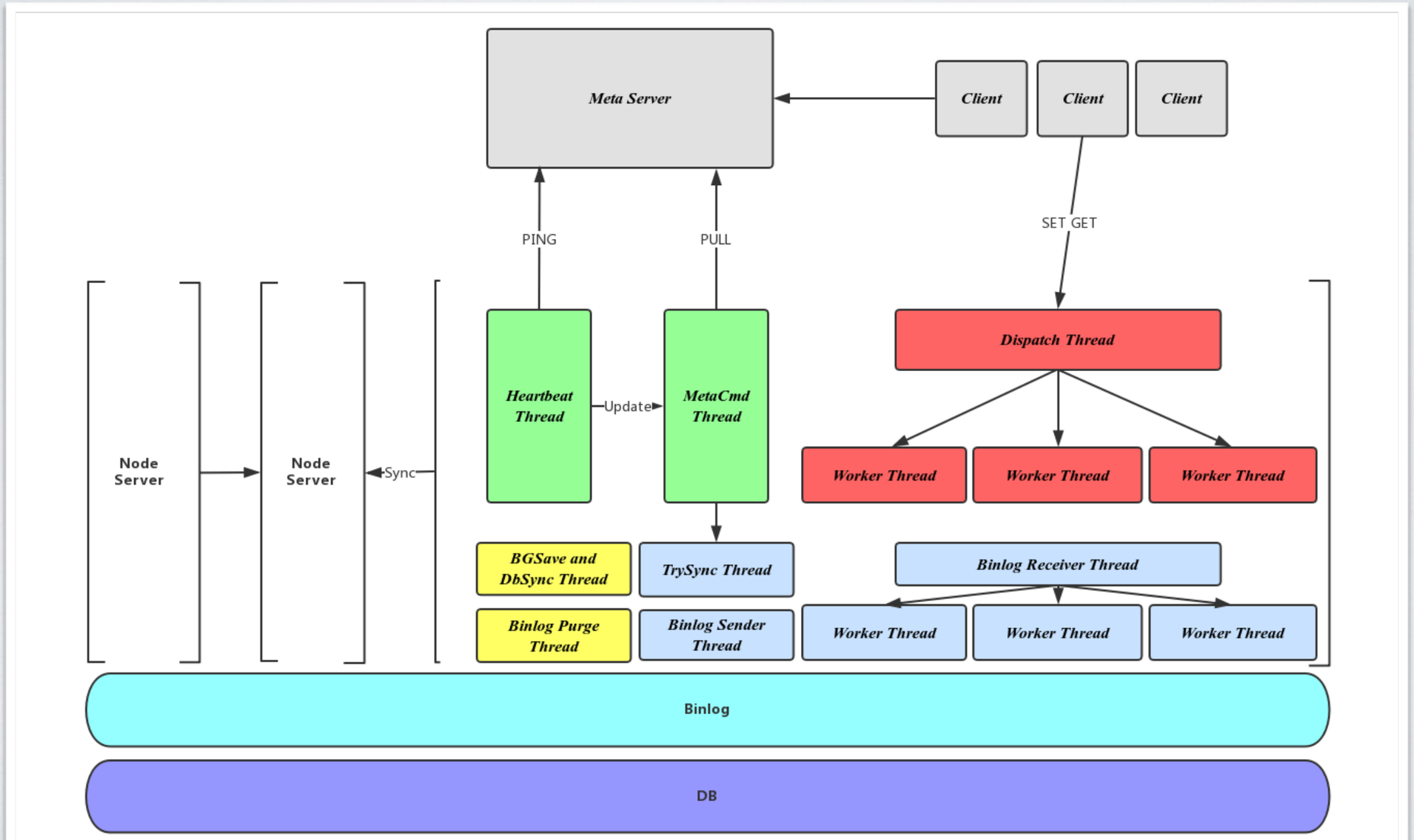
- Partition
- Master
- Slave



NODE SERVER

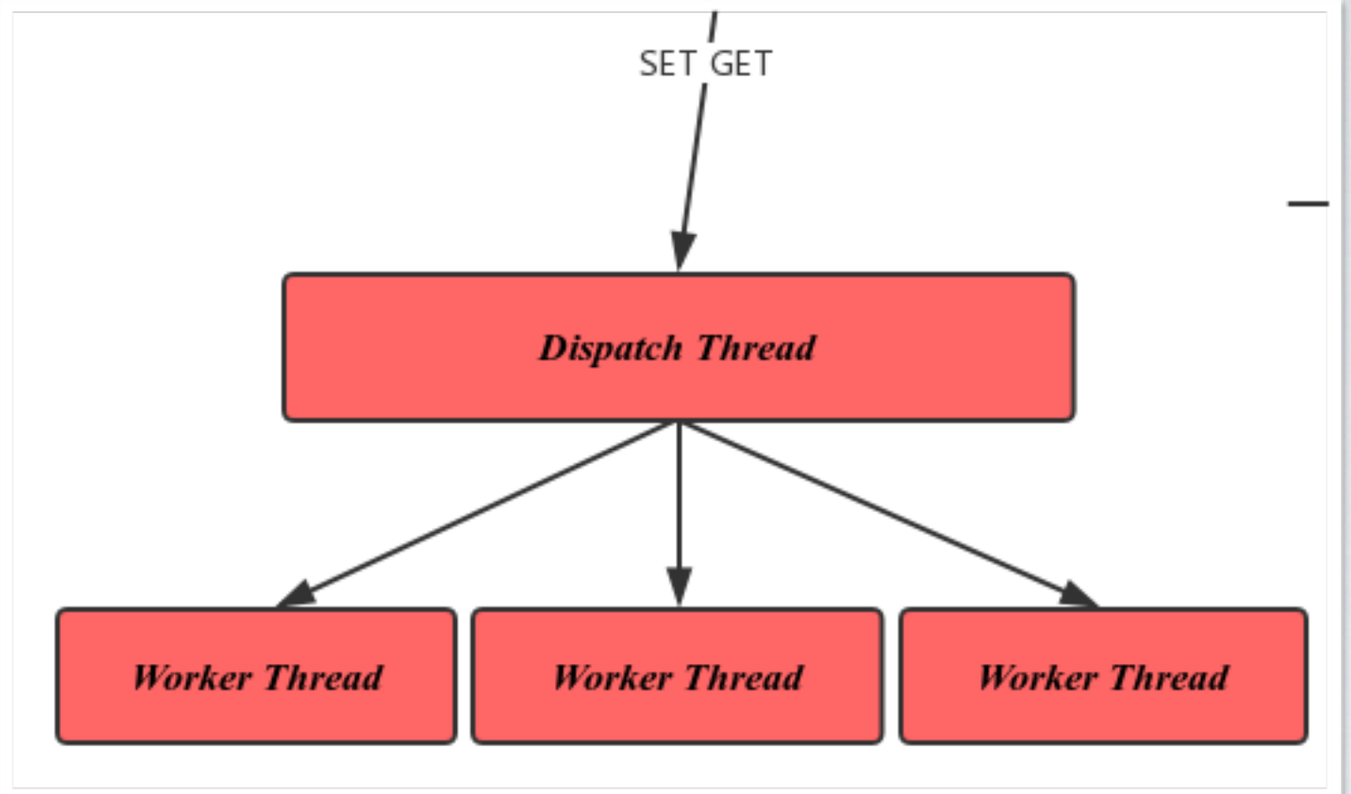
- Thread Model

NODE SERVER

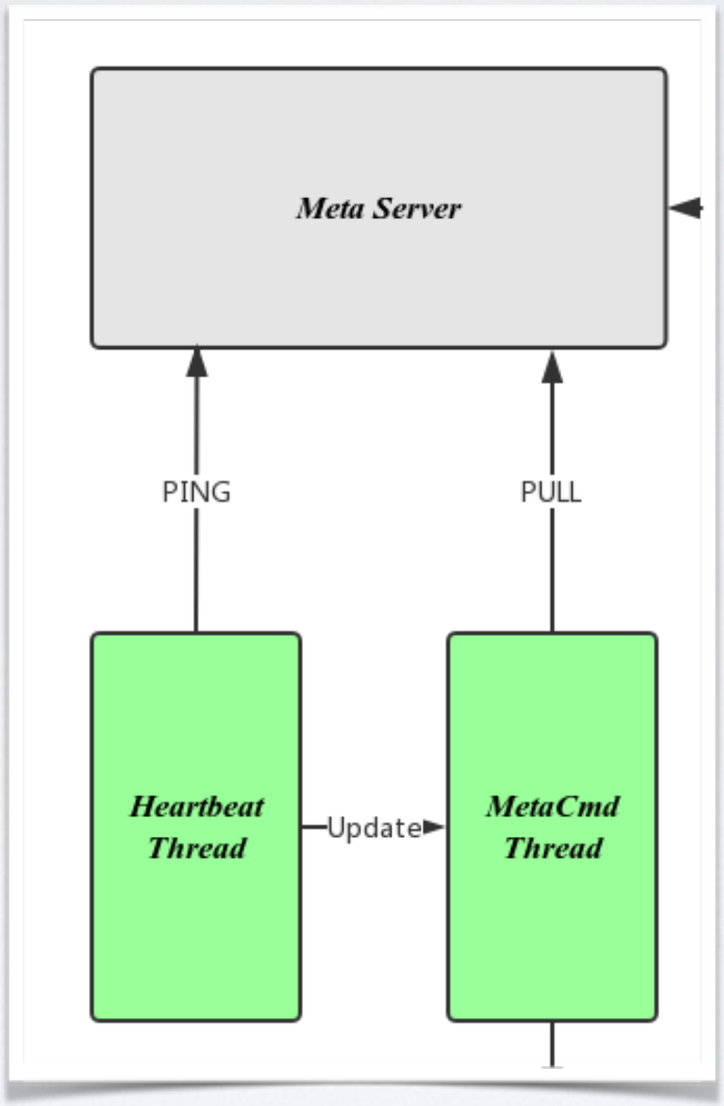

Meta Server
Client
Client
Client
PING
PULL
SET GET
Node Server
Node Server
Sync
Heartbeat Thread
Update
MetaCmd Thread
Dispatch Thread
Worker Thread
Worker Thread
Worker Thread
BGSave and DbSync Thread
TrySync Thread
Binlog Purge Thread
Binlog Sender Thread
Binlog Receiver Thread
Worker Thread
Worker Thread
Worker Thread
Binlog
DB

NODE SERVER

```
zp_data_partiton.cc
if (!cmd->is_suspend()) {
  // read lock
  pthread_rwlock_rdlock(&suspend_rw_);
}
if (cmd->is_write) {
  // lock this key
  mutex_record_.Lock(key);
}
cmd->Do(req, &res);
if (cmd->is_write) {
  if (res->code() == client::StatusCode::kOk) {
    logger_->Put(raw);
  }
  mutex_record_.Unlock(key);
}
if (!cmd->is_suspend()) {
  pthread_rwlock_unlock(&suspend_rw_);
}
```



NODE SERVER

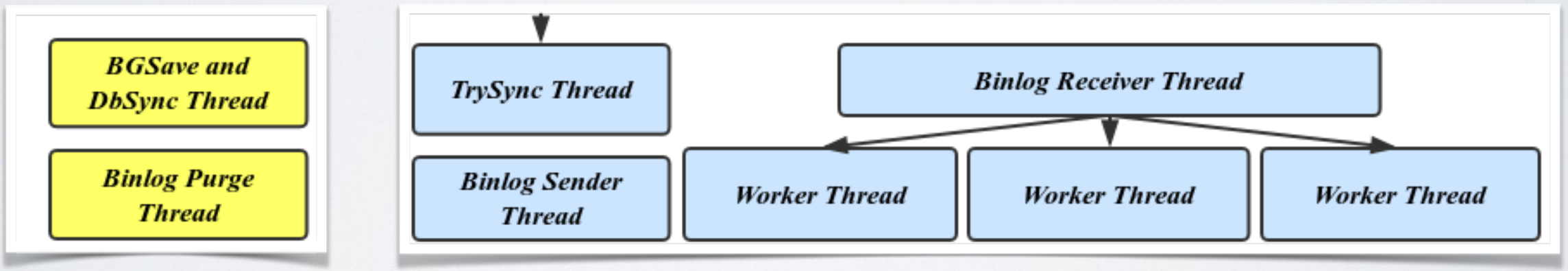

Meta Server
PING
PULL
Heartbeat Thread
Update
MetaCmd Thread

NODE SERVER

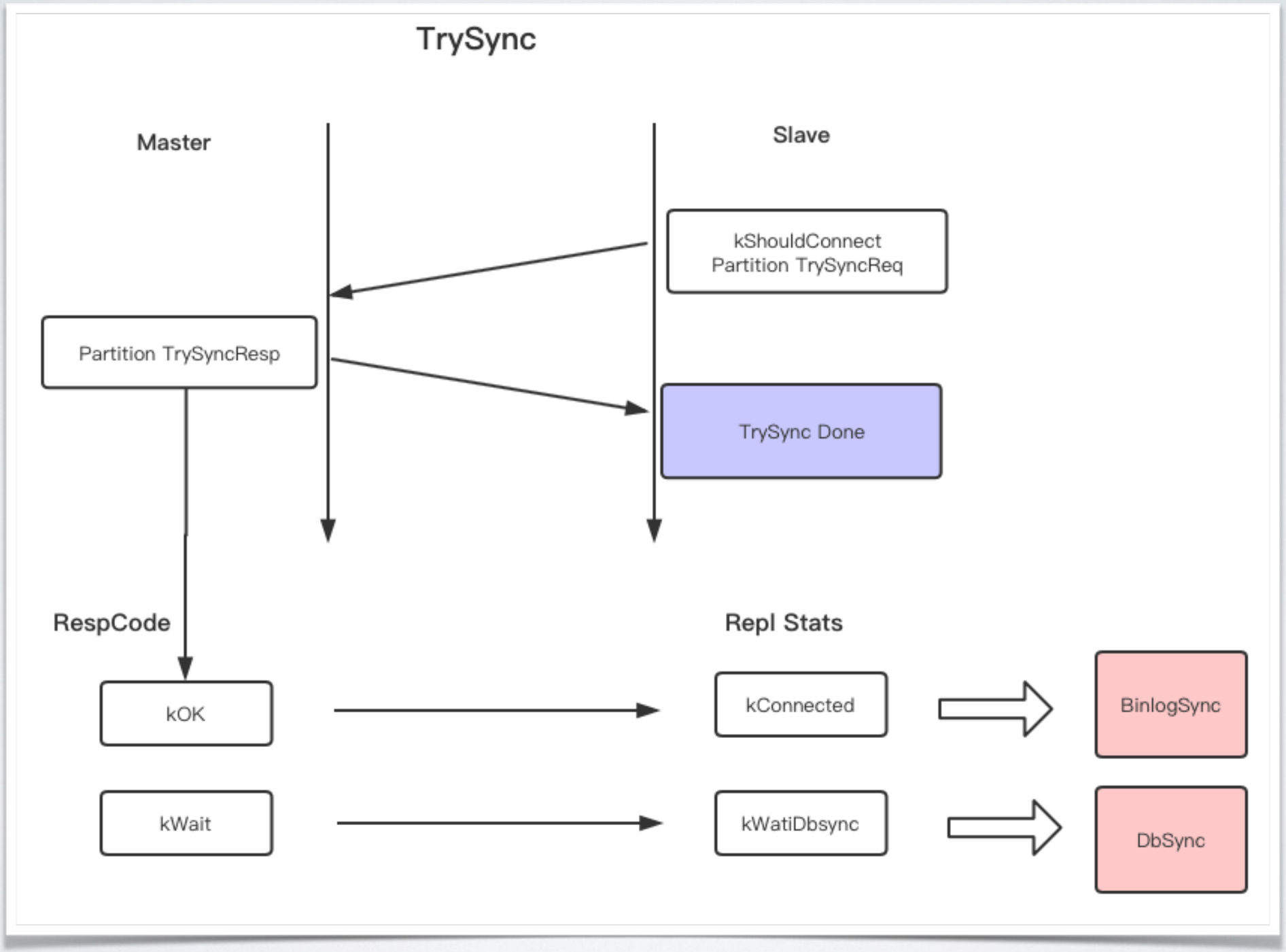
- Synchronization

NODE SERVER

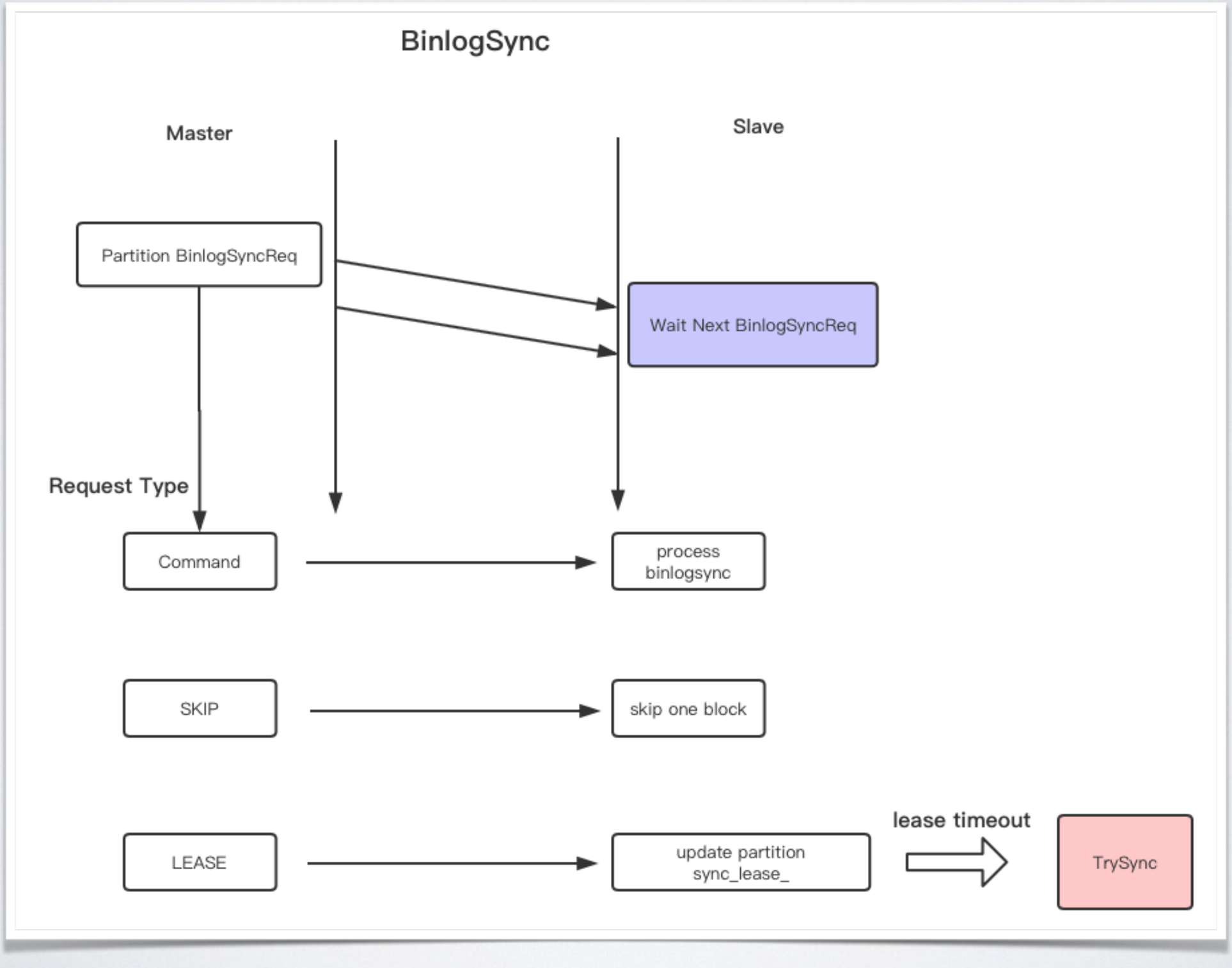
- Binlog
- DBSync & Binlog Sync



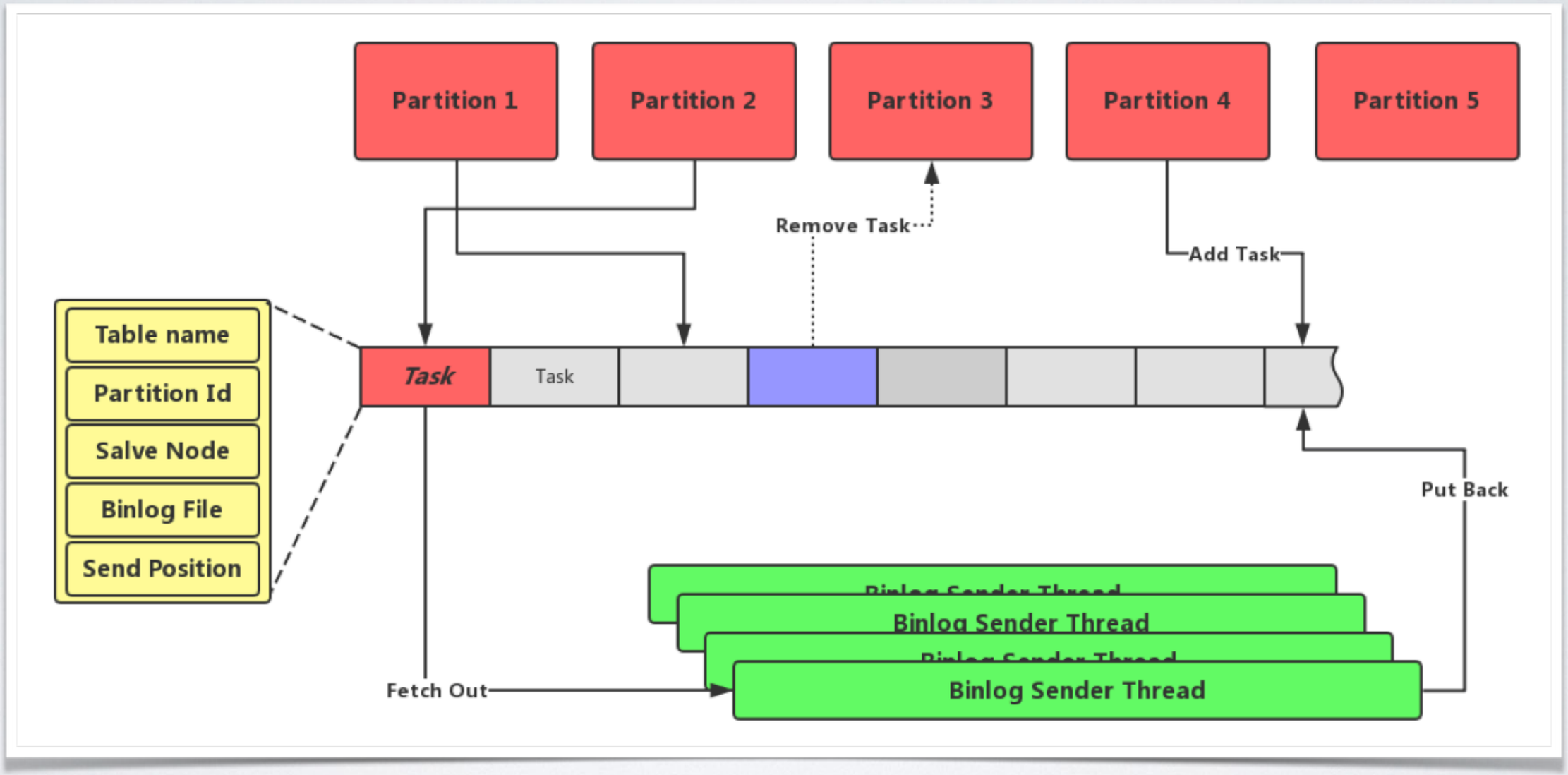
NODE SERVER


TrySync
Master
Slave
kShouldConnect
Partition TrySyncReq
Partition TrySyncResp
TrySync Done
RespCode
Repl Stats
kOK
kConnected
BinlogSync
kWait
kWatiDbsync
DbSync

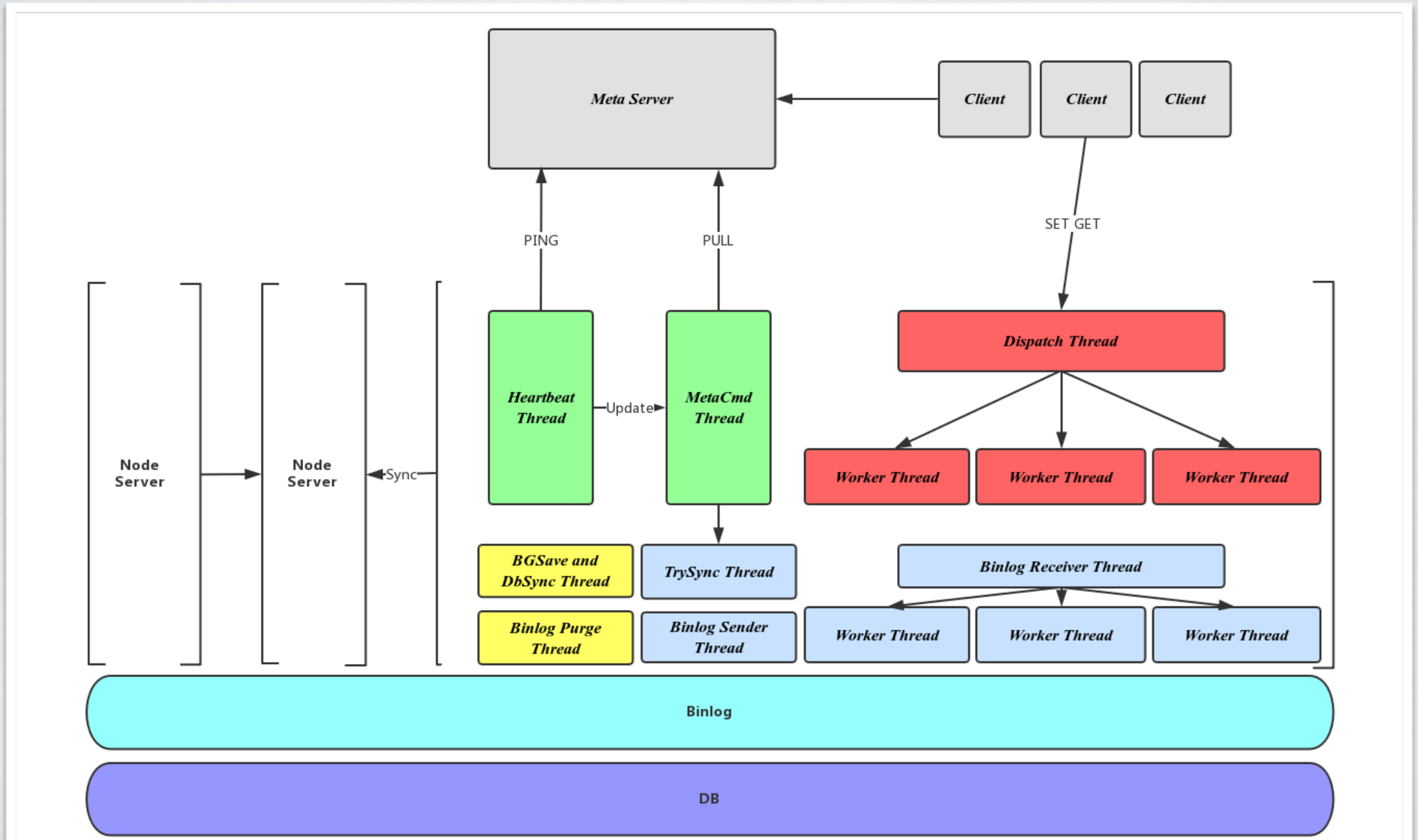
NODE SERVER


BinlogSync
Master
Slave
Partition BinlogSyncReq
Wait Next BinlogSyncReq
Request Type
Command
process binlogsync
SKIP
skip one block
LEASE
update partition sync_lease_
lease timeout
TrySync

NODE SERVER


Partition 1
Partition 2
Partition 3
Partition 4
Partition 5
Remove Task
Add Task
Table name
Partition Id
Salve Node
Binlog File
Send Position
Task
Task
Put Back
Binlog Sender Thread
Binlog Sender Thread
Fetch Out

NODE SERVER


Meta Server
Client
Client
Client
PING
PULL
SET GET
Node Server
Node Server
Sync
Heartbeat Thread
Update
MetaCmd Thread
Dispatch Thread
Worker Thread
Worker Thread
Worker Thread
BGSave and DbSync Thread
TrySync Thread
Binlog Purge Thread
Binlog Sender Thread
Binlog Receiver Thread
Worker Thread
Worker Thread
Worker Thread
Binlog
DB

META SERVER

- Meta Info
- Thread Model
- Cluster Management(Migrate)

META SERVER

- Meta Info

META SERVER

- Meta Info
  - Cluster Status
  - Cluster Topology

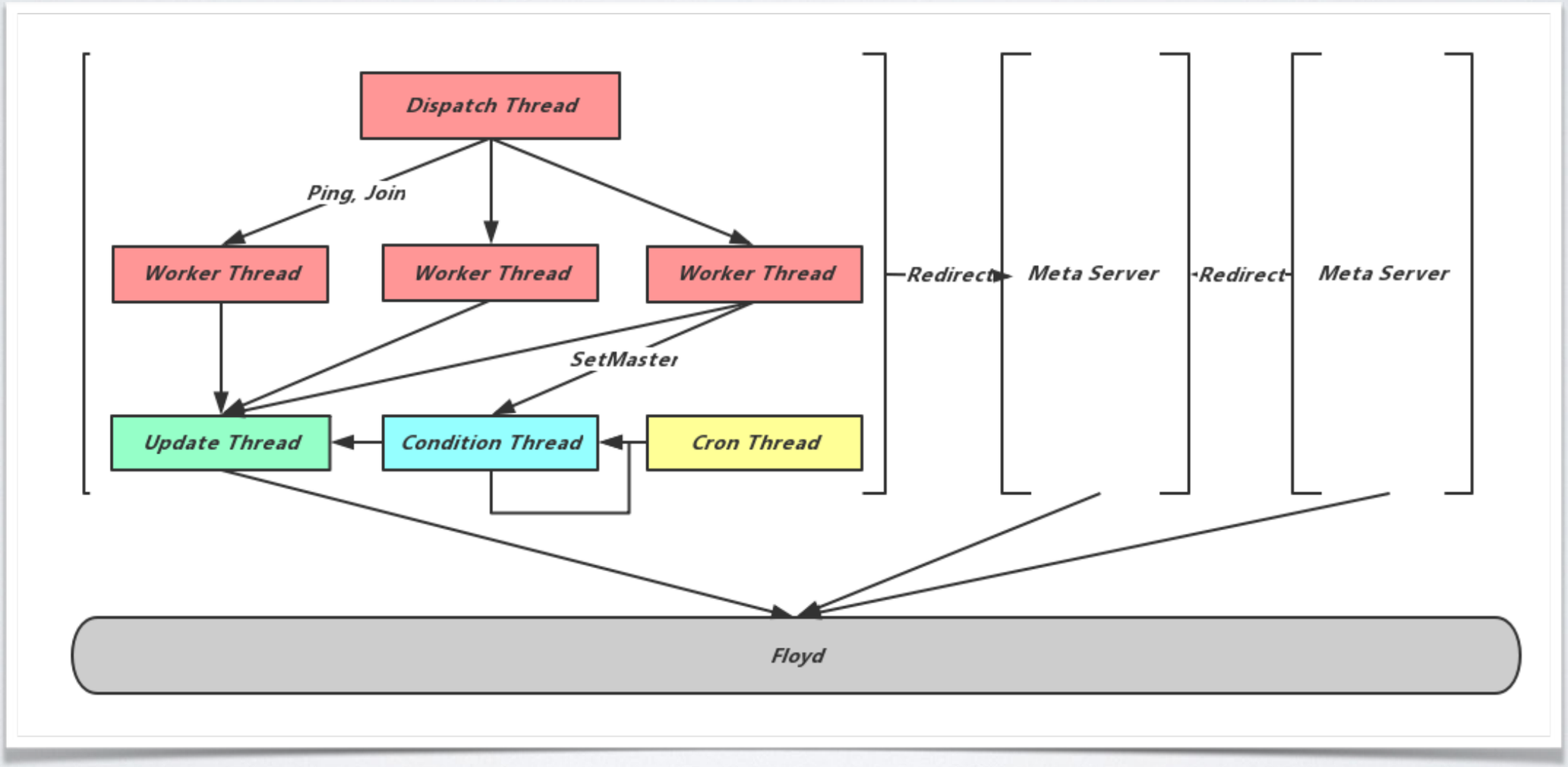
```
class ZPMetaInfoStore {
  // -2 for uninitialed
  // -1 for initialed but no table
  // Otherwise non-negtive integer and monotone increasing
  // epoch records topology info changes
  std::atomic<int> epoch_;
  // table => ZPMeta::Table set
  std::unordered_map<std::string, ZPMeta::Table> table_info_;
  // node => alive time + offset set, 0 means already down node
  // only valid for leader
  std::unordered_map<std::string, NodeInfo> node_infos_;
}

message Partitions {
  required int32 id = 1;
  required PState state = 2;
  required Node master = 3;
  repeated Node slaves = 4;
}

ZPMeta::Table
message Table {
  required string name = 1;
  repeated Partitions partitions = 2;
}

struct NodeInfo {
  uint64_t last_active_time;
  // table_partition -> offset
  std::map<std::string, NodeOffset> offsets;
}
```

META SERVER

- Thread Model

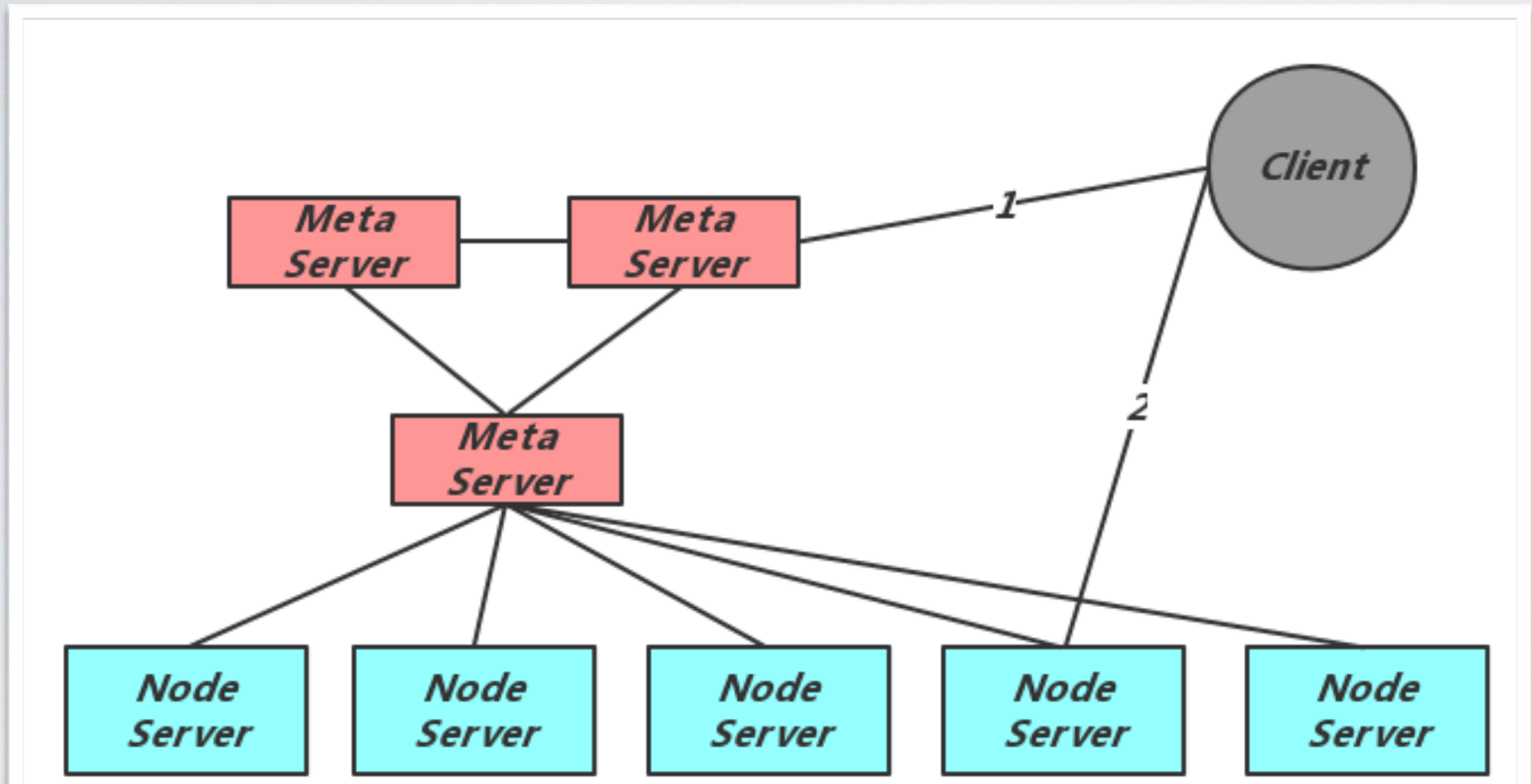
META SERVER

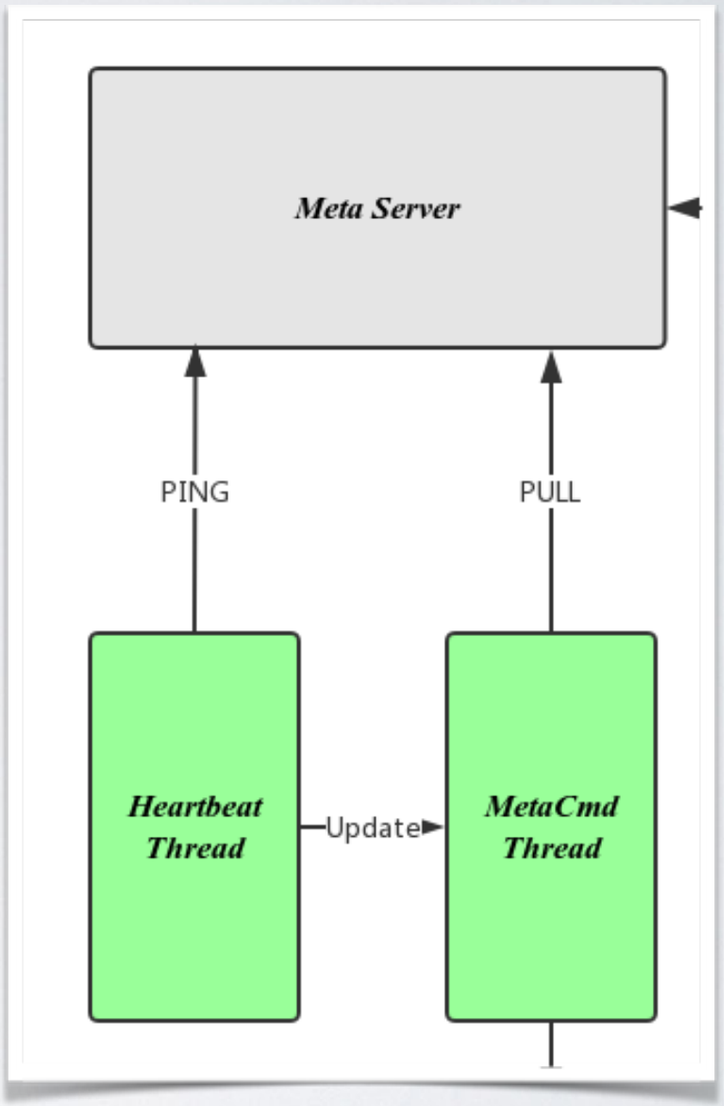
- Thread Model



META SERVER

- How does meta Info apply to node server?



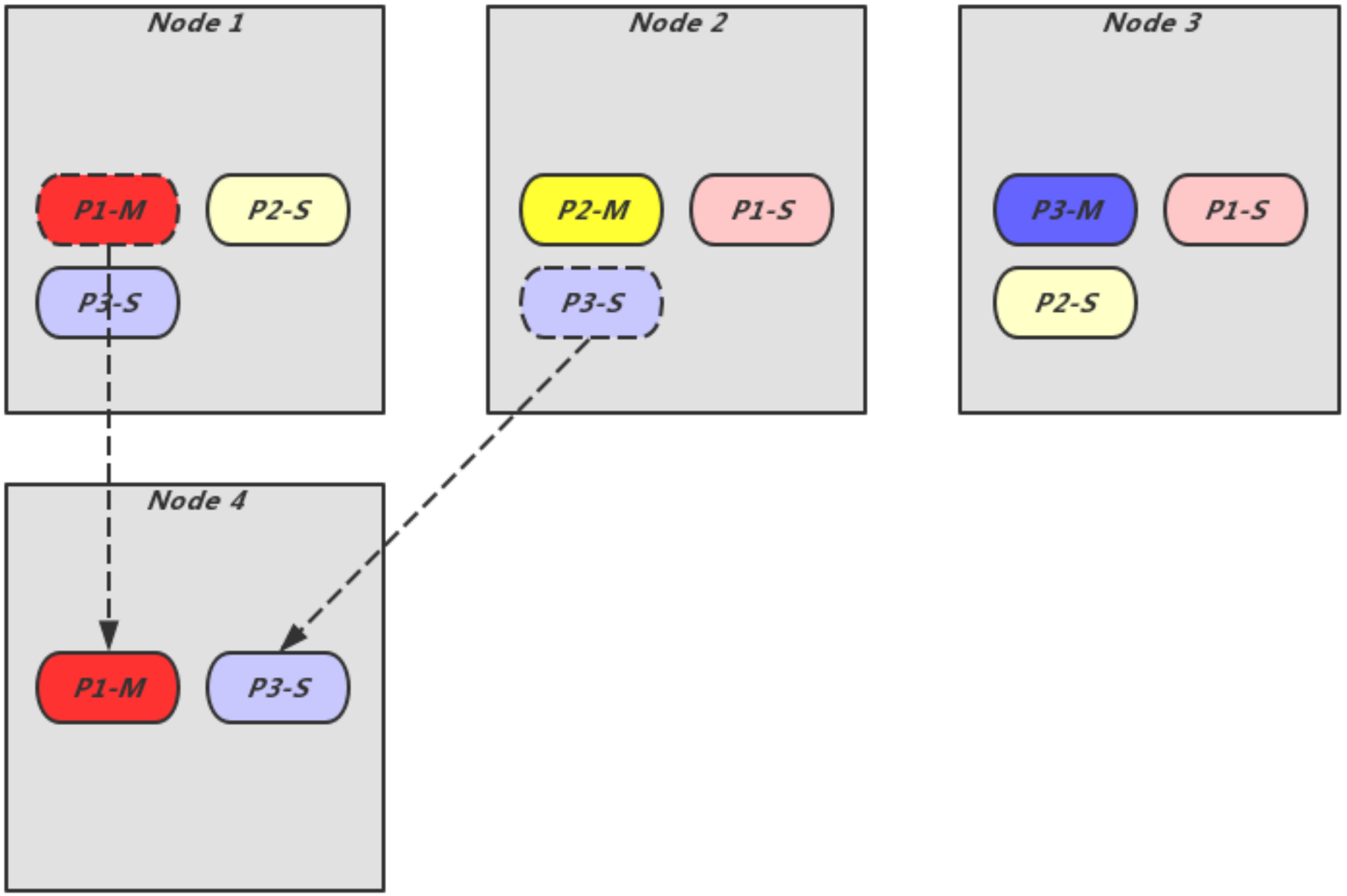


- **1. Client connect to leader Meta Server and modify meta info.**
- **2. As raft exist, follower and leader could reach a consensus about meta info**
- **3. Node server do ping routine. Find epoch differ from Meta Server. Pull newest Meta Info and apply this modification locally.**

META SERVER

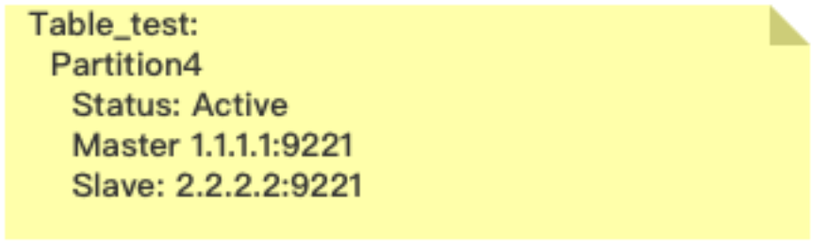
- Migrate

META SERVER

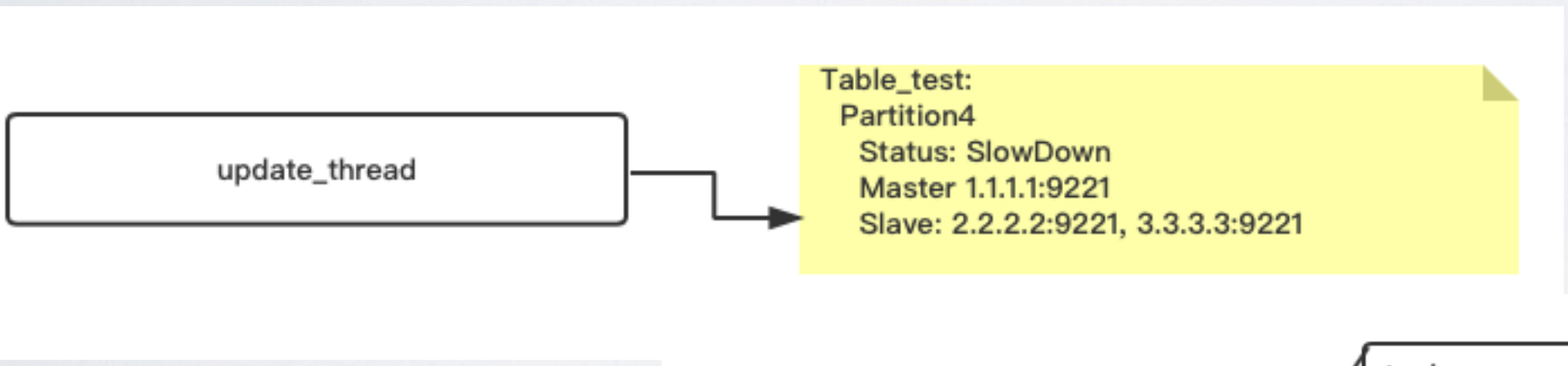

Node 1
P1-M
P2-S
P3-S
Node 2
P2-M
P1-S
P3-S
Node 3
P3-M
P1-S
P2-S
Node 4
P1-M
P3-S

META SERVER

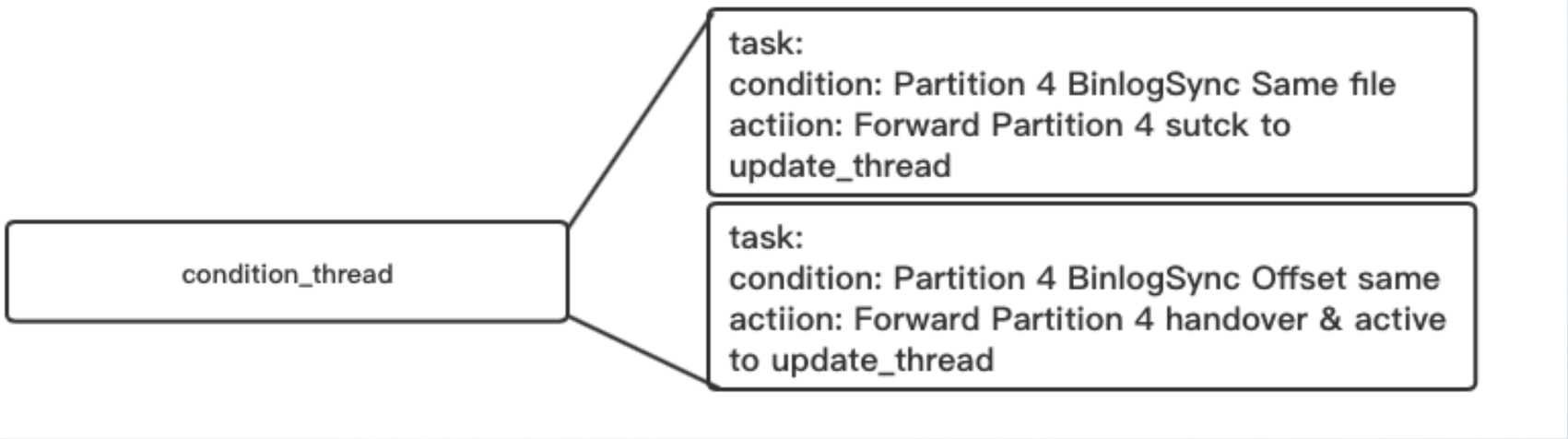
- Original Meta



- Send “migreate table_test 1.1.1.1:9221 4 3.3.3.3:9221” to meta server



-

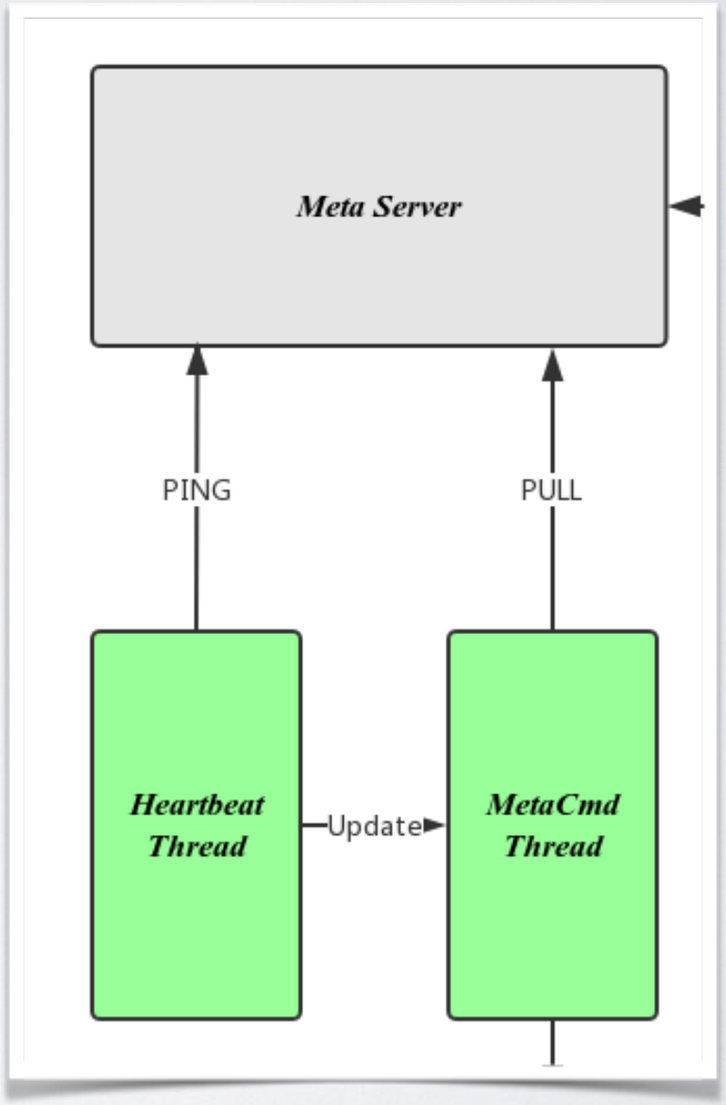


META SERVER

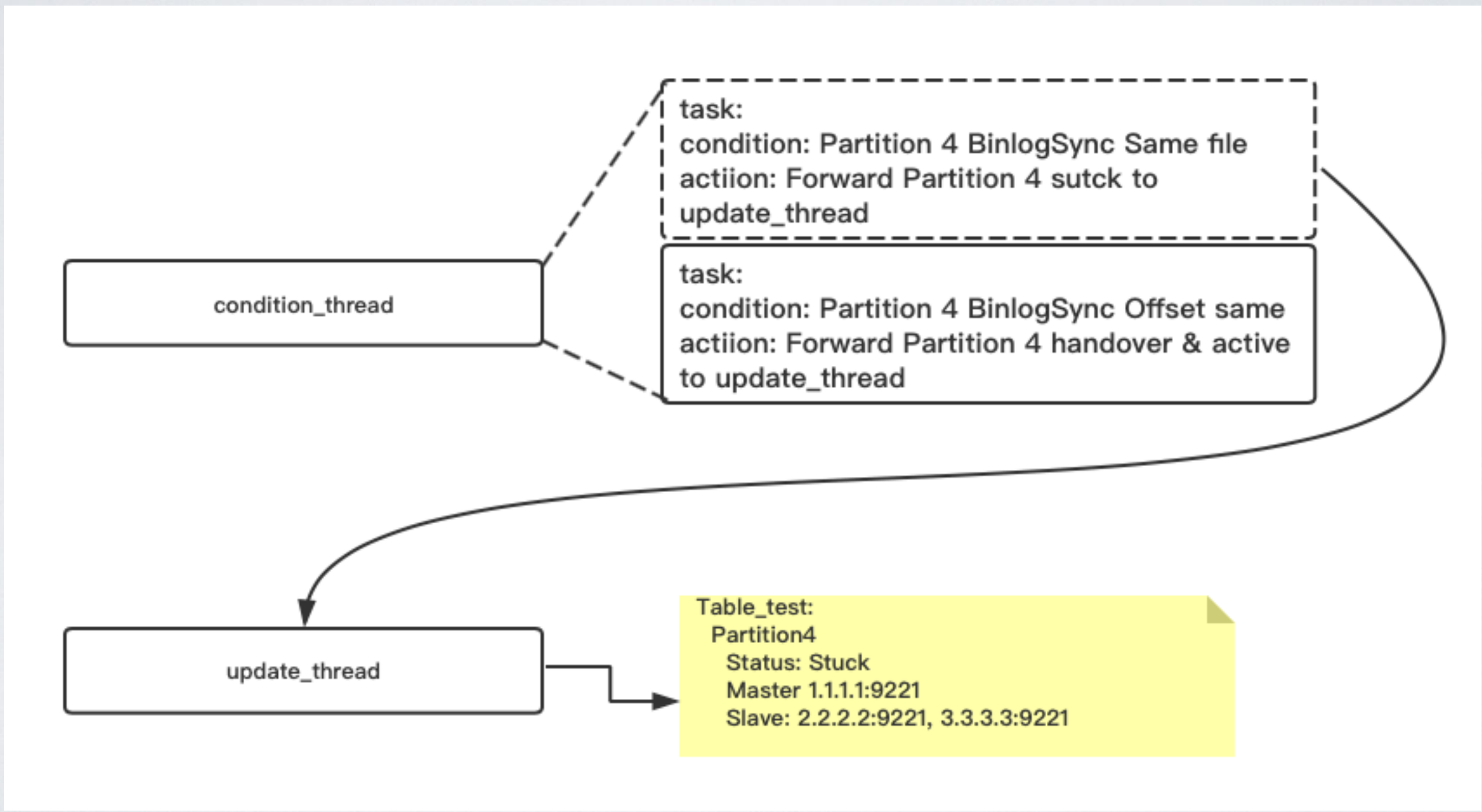
- node server apply meta info changes locally

```
Table_test:
  Partition4
    Status: SlowDown
    Master 1.1.1.1:9221
    Slave: 2.2.2.2:9221, 3.3.3.3:9221
```

- continue ping…



META SERVER

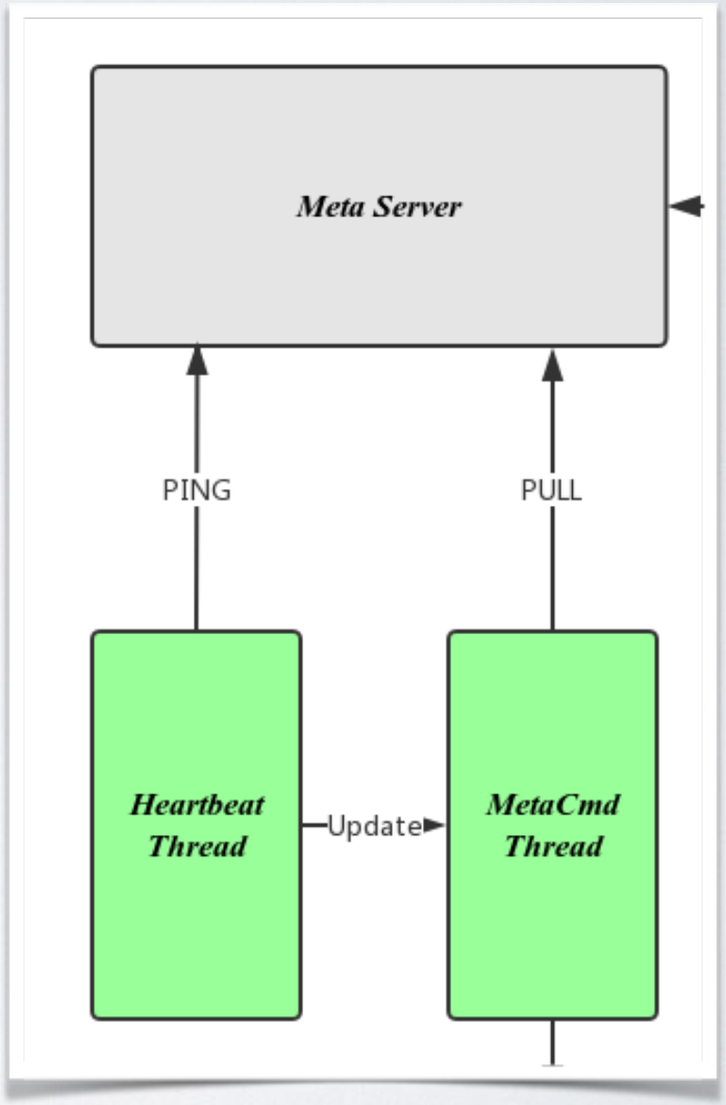

task:
condition: Partition 4 BinlogSync Same file
actiion: Forward Partition 4 sutck to update_thread
condition_thread
task:
condition: Partition 4 BinlogSync Offset same
actiion: Forward Partition 4 handover & active to update_thread
update_thread
Table_test:
Partition4
Status: Stuck
Master 1.1.1.1:9221
Slave: 2.2.2.2:9221, 3.3.3.3:9221

META SERVER

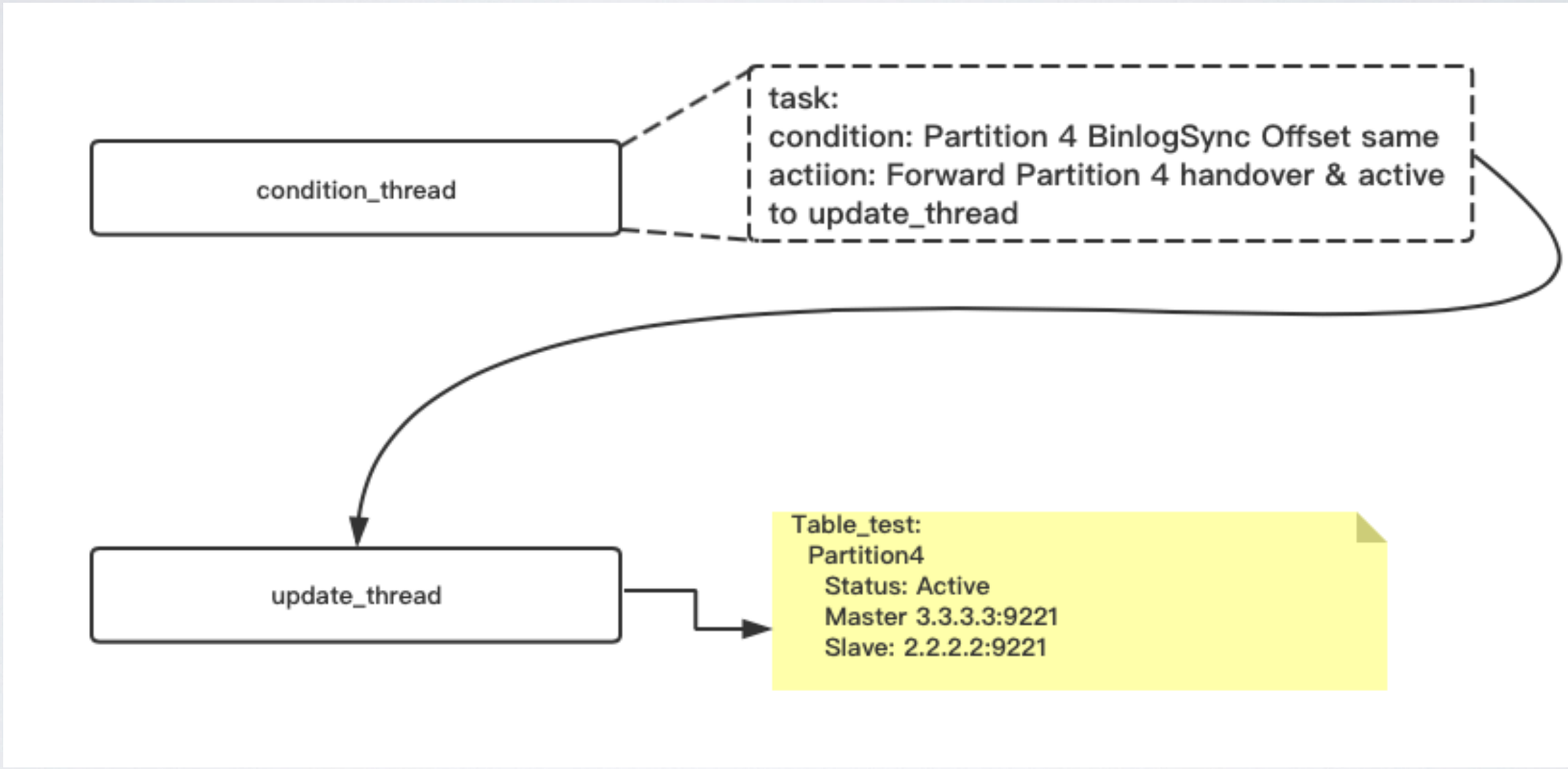
- node server apply meta info changes locally

```
Table_test:
  Partition4
    Status: Stuck
    Master 1.1.1.1:9221
    Slave: 2.2.2.2:9221, 3.3.3.3:9221
```

- continue ping…



META SERVER

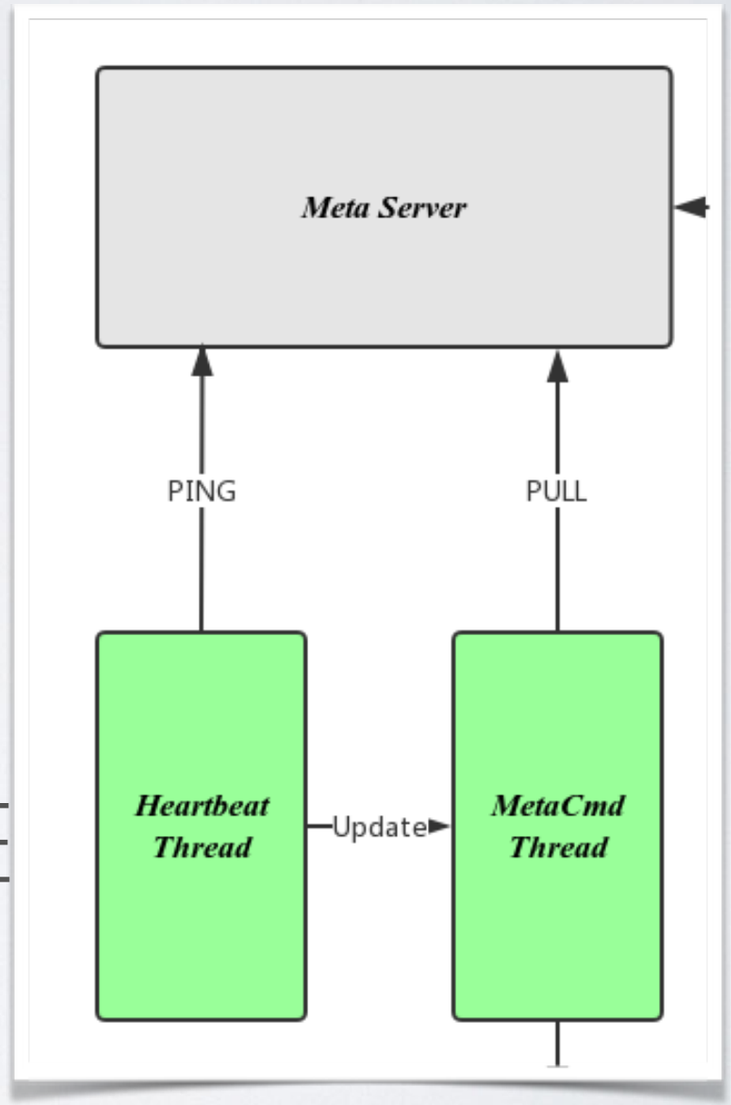

condition_thread
task:
condition: Partition 4 BinlogSync Offset same
actiion: Forward Partition 4 handover & active
to update_thread
update_thread
Table_test:
Partition4
Status: Active
Master 3.3.3.3:9221
Slave: 2.2.2.2:9221

META SERVER

- node server apply meta info changes locally

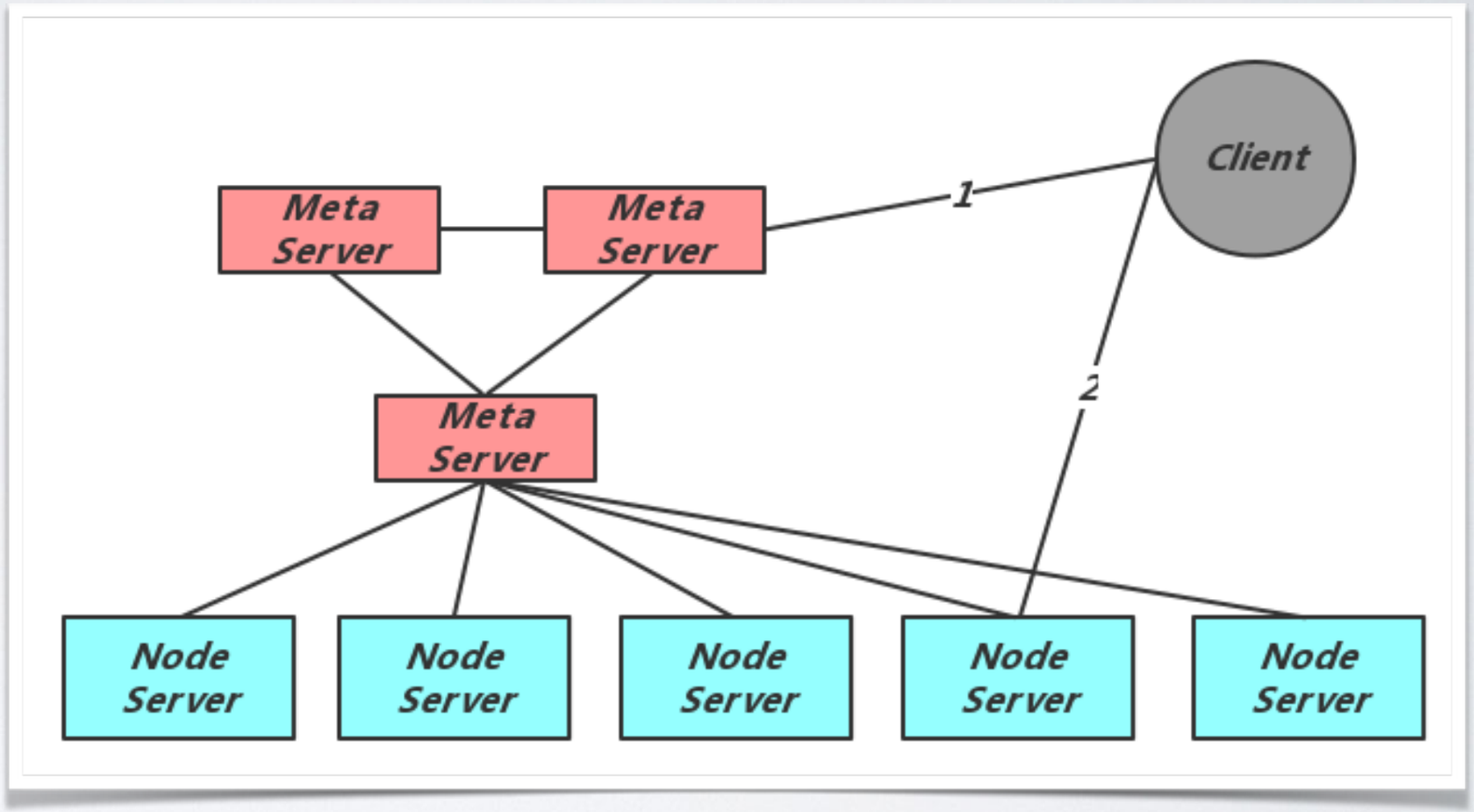
```
Table_test:
  Partition4
    Status: Active
    Master 3.3.3.3:9221
    Slave: 2.2.2.2:9221
```

- New Partition4 Master is ACTIVE



ZEPPELIN OVERVIEW

- Pros & Cons
- Future……



WHAT ABOUT DISTRIBUTED PIKA

- Synchronization Evolution Done

- Next……

THANKS