MongoDb Databases

Mongo - WiredTiger Storage

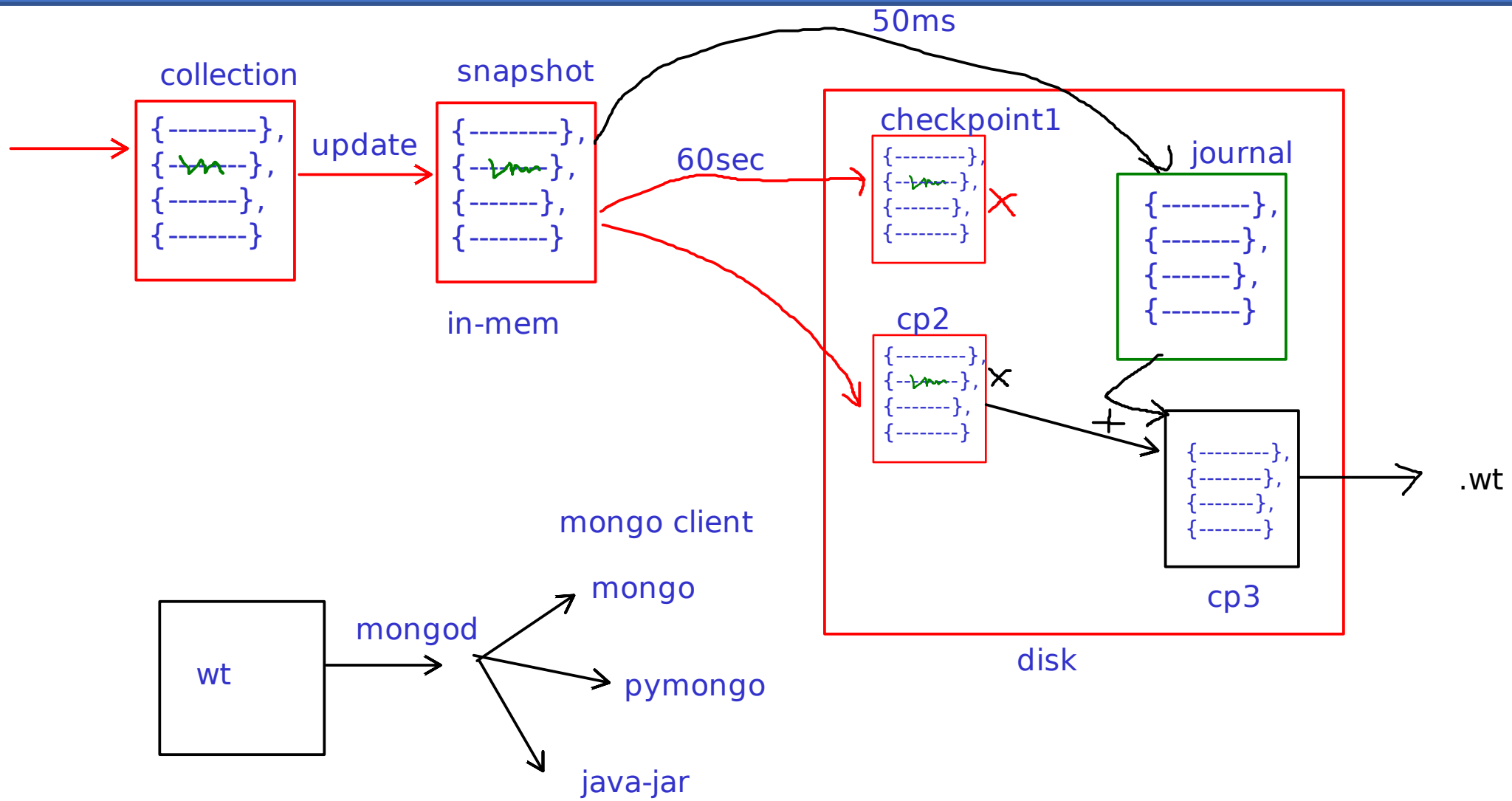
Storage engine is managing data in memory and on disk.

MongoDB 3.2 onwards default storage engine is *WiredTiger*; while earlier version it was MMAPv1.

WiredTiger storage engine:

- Uses document level optimistic locking for better performance.
- Per operation a snapshot is created from consistent data in memory.
- The snapshot is written on disk, known as checkpoint → for recovery.
- Checkpoints are created per 60 secs or 2GB of journal data.
- Old checkpoint is released, when new checkpoint is written on disk and updated in system tables.
- To recover changes after checkpoint, enable journaling.

collection-9-5033127685441071718.wt


collection
snapshot
50ms
update
60sec
checkpoint1
journal
in-mem
cp2
.wt
cp3
disk
mongo client
mongo
mongod
wt
pymongo
java-jar

Mongo - WiredTiger Storage

WAT

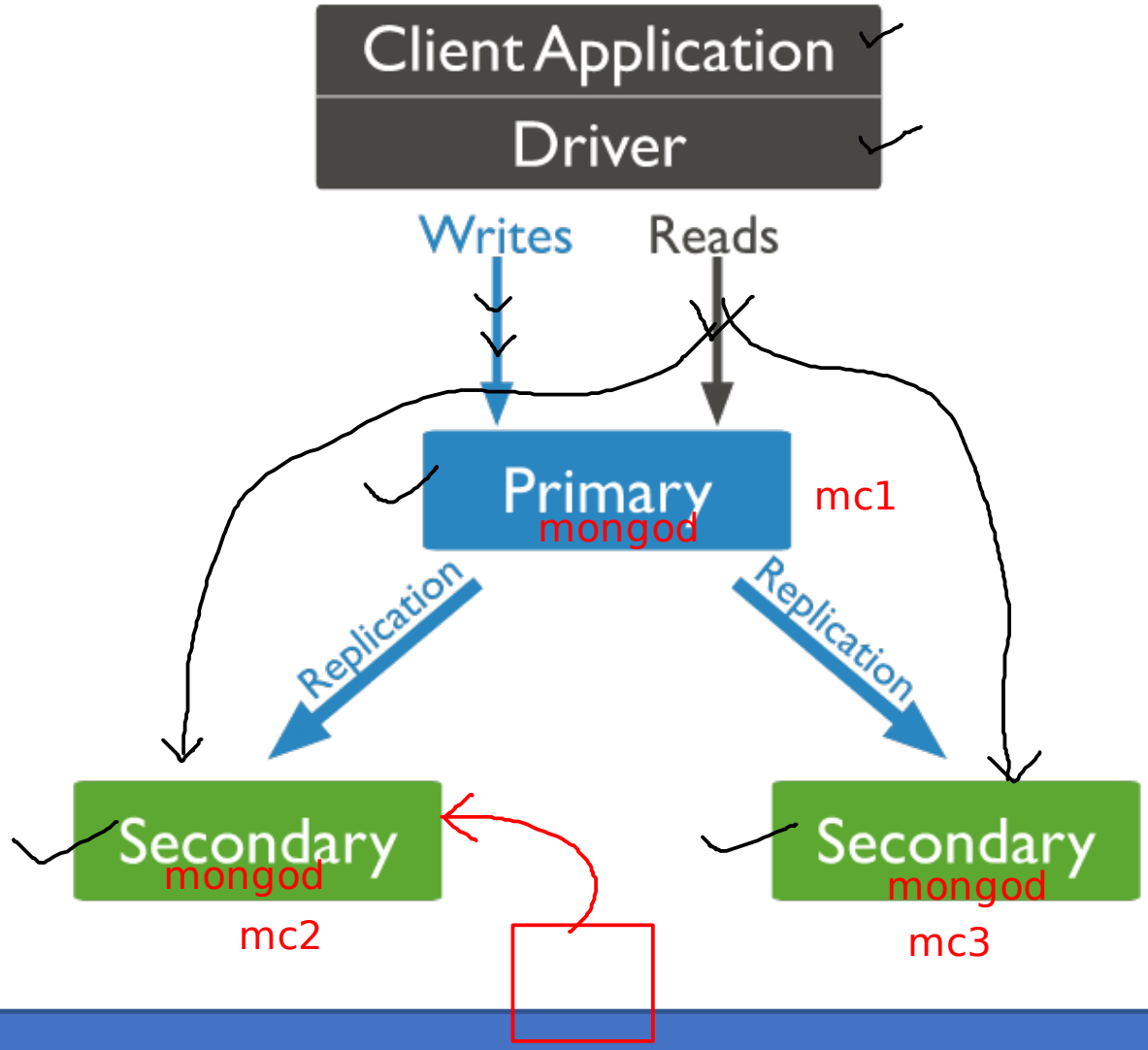
- WT uses write-ahead transaction in journal log to ensure durability.
- It creates one journal record for each client initiated write operation.
- Journal persists all data modifications between checkpoints.
- Journals are in-memory buffers that are created on disk per 50 ms.
- WiredTiger stores all collections & journals in compressed form.
- Recovery process with journaling:
  - Get last checkpoint id from data files.
  - Search in journal file for records matching last checkpoint.
  - Apply operations in journal since last checkpoint.

- WiredTiger use internal cache with size max of 256 MB and 50% RAM - 1GB along with file system cache.

4GB => 1GB
16GB => 7GB

Mongo - Replication

- A replica set is a group of mongod instances that maintain the same data set.
- Only one member is deemed the primary node, while other nodes are deemed secondary nodes.
- The secondaries replicate the primary's oplog.
- If the primary is unavailable, an eligible secondary will become primary.



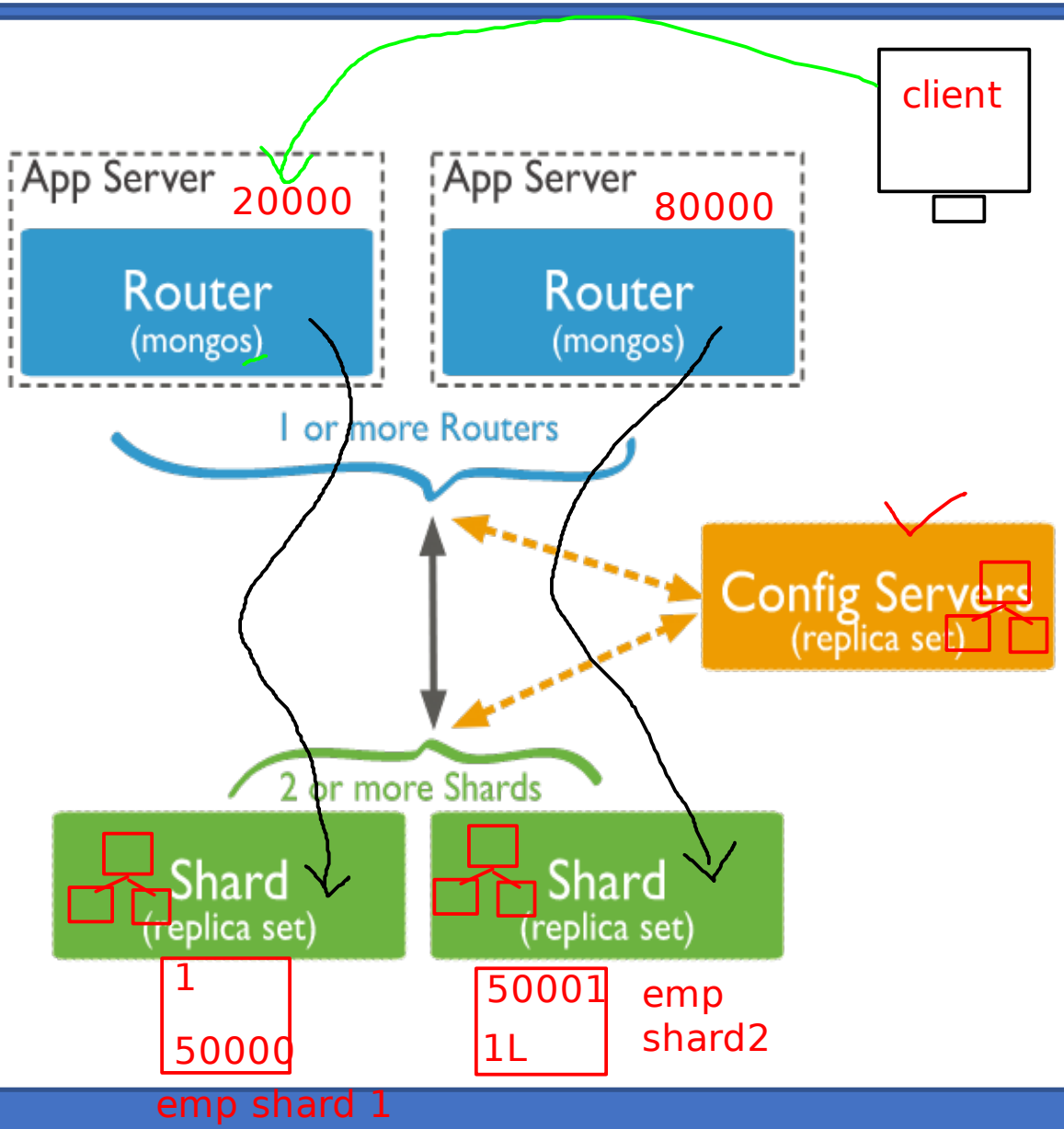
Mongo - Replication

small data

- Secondary servers communicate with each other via heart-beat.
- Secondary applies operations from primary asynchronously.
- When primary cannot communicate a secondary for more than 10 seconds, secondary will hold election to elect itself as new primary. This automatic failover process takes about a minute.
- An arbiter (do not store data) can be added in the system (with even number of secondaries) to maintain quorum in case of election.
- By default client reads from primary, but can set read preference from secondary. Reading from secondary may not reflect state of primary; as read from primary may read before data is durable.

Mongo - Sharding

- Sharding is a method for distributing large data across multiple machines.
- This is mongodb approach for horizontal scaling/scaling out.
- shard: part of collection on each server (replica set).
- mongos: query router between client & cluster.
- config servers: metadata & config settings of cluster.



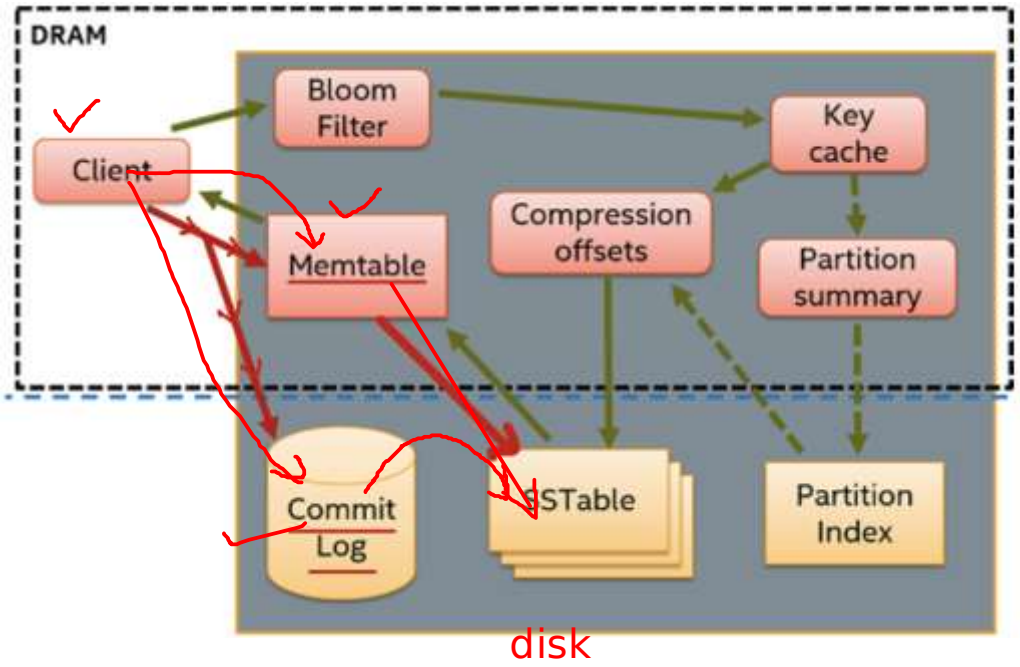
Mongo - Sharding

- Collections can be sharded across the servers based on shard keys.
- Shard keys:
  - Consist of immutable field/fields that are present in each document
  - Only one shard key to be chosen when sharding collection. Cannot change shard key later.
  - Collection must have index starting on shard key.
  - Choice of shard key affect the performance.
- Advantages:
  - Read/Write load sharing
  - High storage capacity
  - High availability

Cassandra

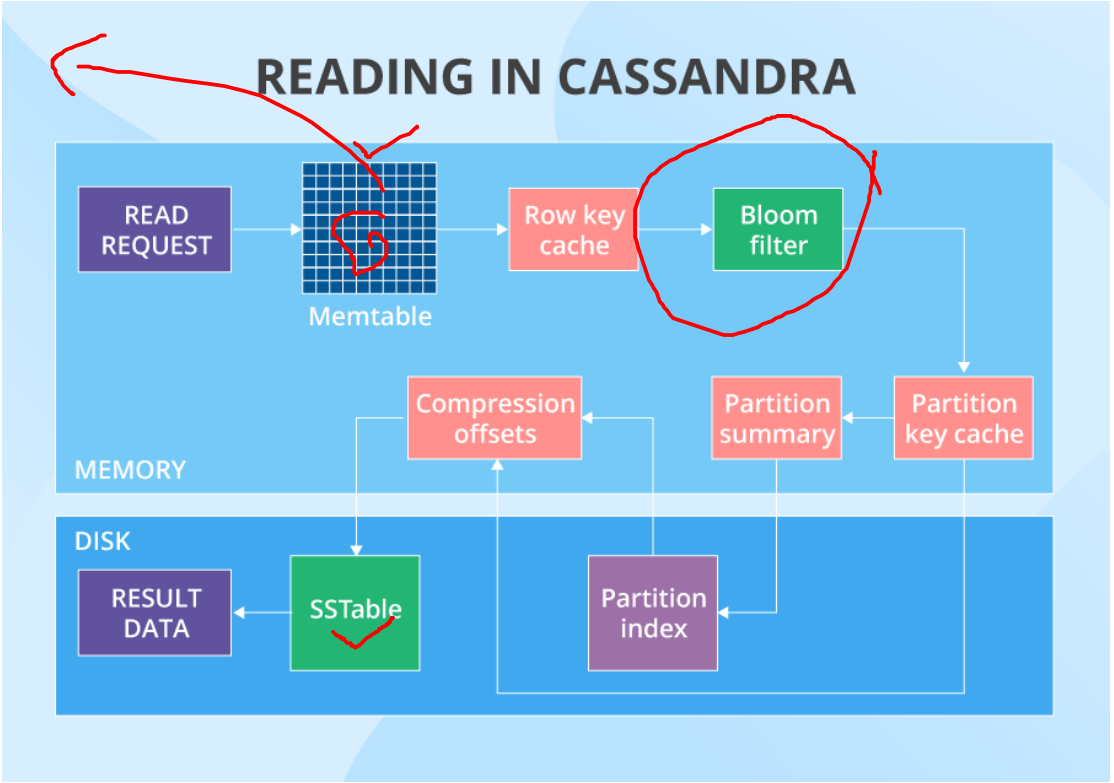
Architecture

- **Commit Log**
  - Append only log of all mutations local to a node.
  - Client data commit log -> memtable.
  - Durability in the case of unexpected shutdown
  - On startup, any changes in log will be applied to tables
- **Memtable**
  - In-memory structures to write Cassandra buffers.
  - One active memtable per table.
- **Sorted String Table**
  - Immutable data files for persisting data on disk.
  - Multiple memtables merged into single SSTable.
- **LSM Tree** =log structure marge tree
  - Disk based data structure to provide low-cost
  - indexing for a file, in which records are to be
  - inserted at very high rate



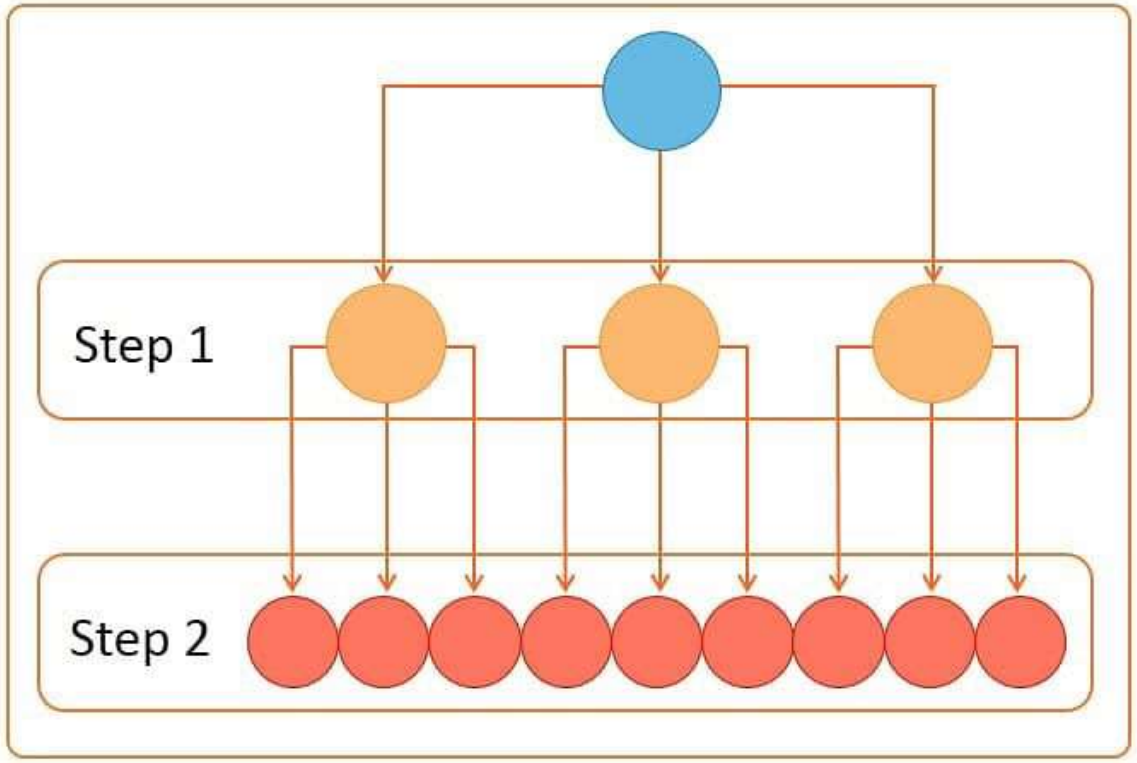
Architecture

- **Bloom filter**
  - In the read path, Cassandra merges data on disk (in SSTables) with data in RAM (in memtables).
  - To avoid checking every SSTable data file for the partition being requested, Cassandra employs a data structure known as a bloom filter.
  - Bloom filters are a probabilistic data structure that allows Cassandra to determine one of two possible states
    - The data definitely does not exist in the given file
    - The data probably exists in the given file



Gossip Protocol

- Cassandra uses a gossip protocol to communicate with nodes in a cluster
- It is an inter-node communication mechanism similar to the heartbeat protocol in Hadoop
- Cassandra uses the gossip protocol to discover the location of other nodes in the cluster and get state information of other nodes in the cluster
- The gossip process runs periodically on each node and exchanges state information with three other nodes in the cluster
- Eventually, information is propagated to all cluster nodes. Even if there are 1000s of nodes, information is propagated to all the nodes within a few seconds



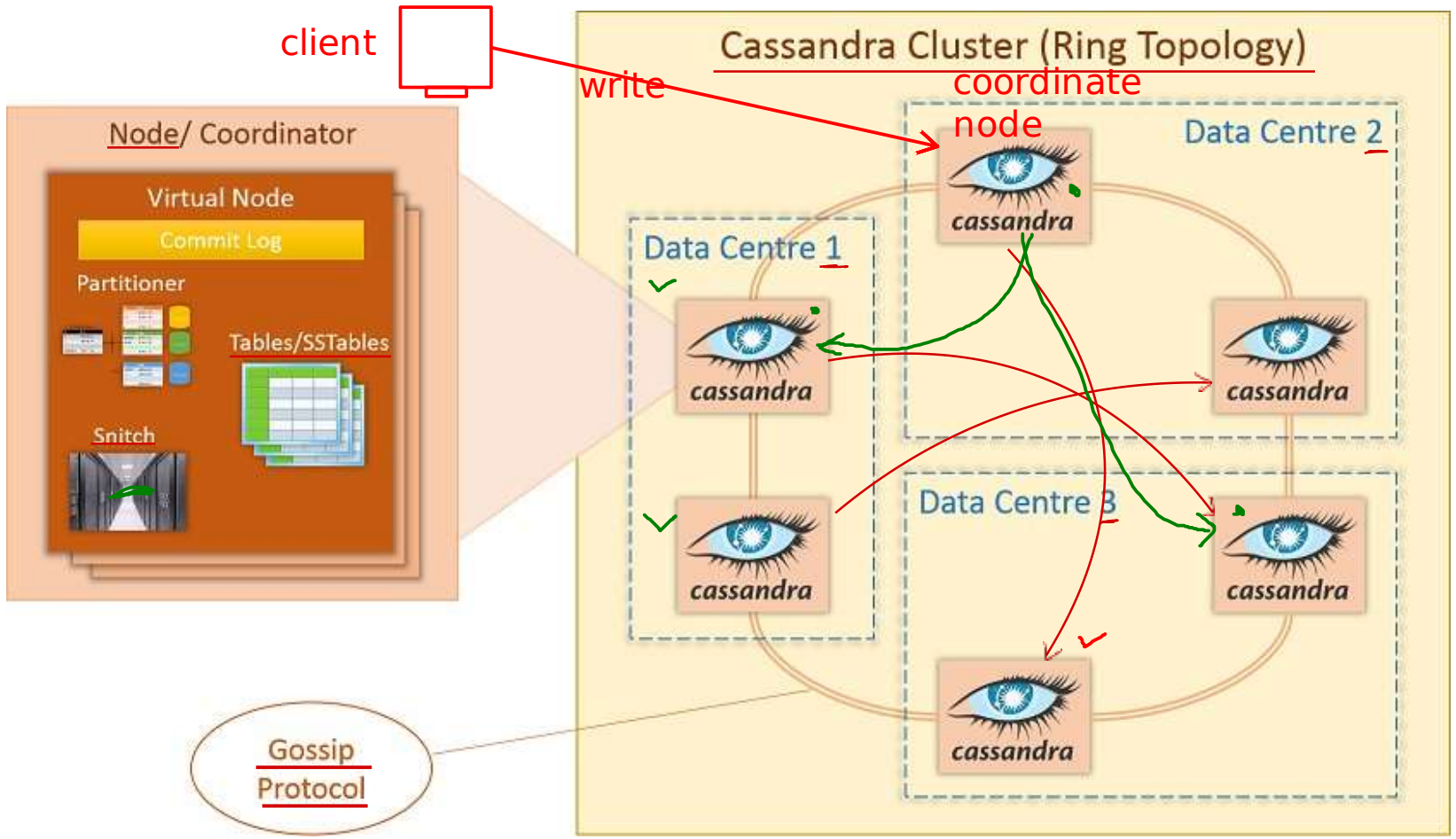
Gossip protocol

Each node learn about cluster topology.
Communicate among nodes.
Detection of faulty nodes.

Snitch

Snitch helps map IPs to racks and data centers.

This info is used for replica location and other tasks.



Thank you!