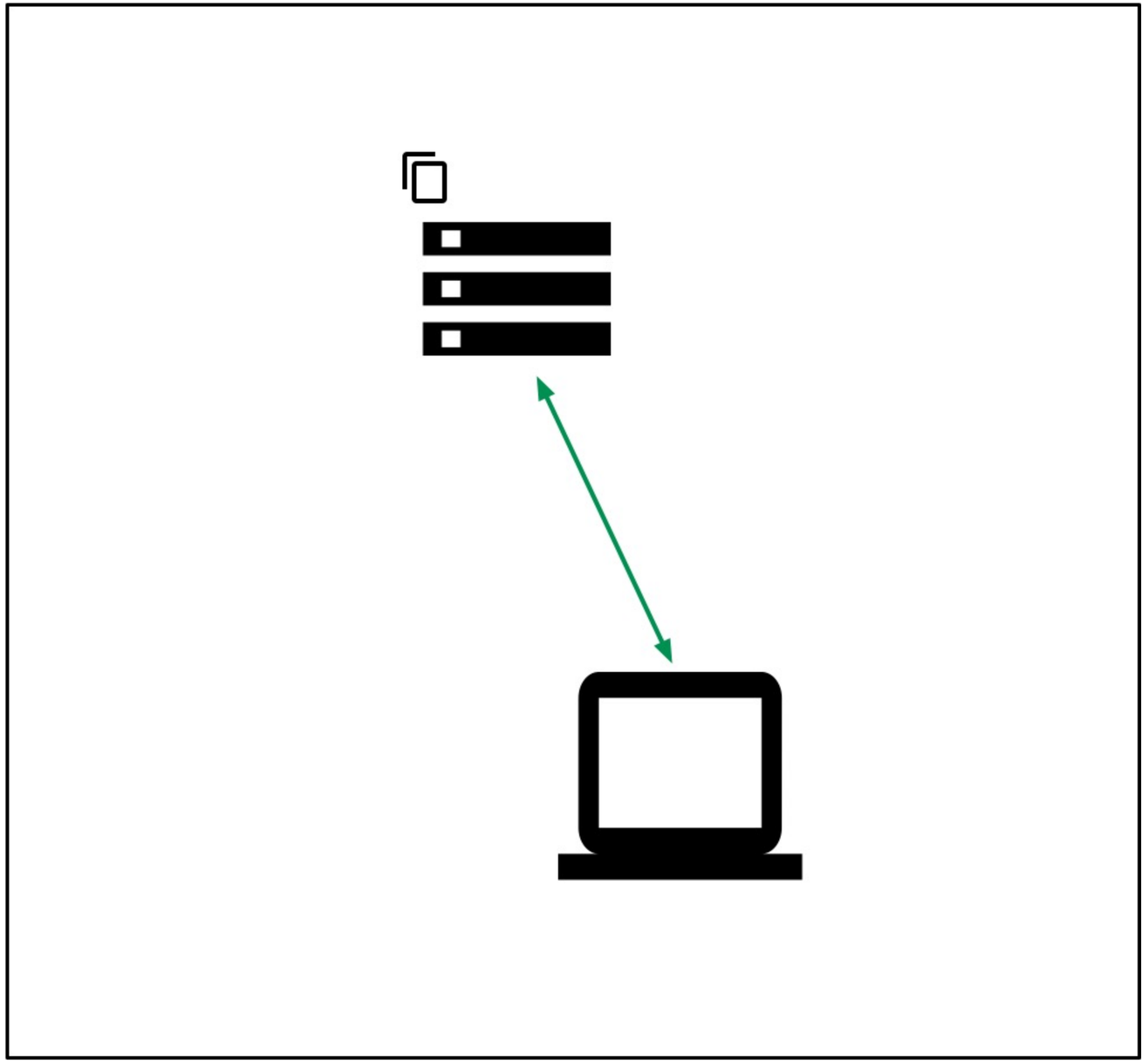
MongoDB
Horizontal scaling with Sharding

COSC 061 - Winter 2025 - Dartmouth College

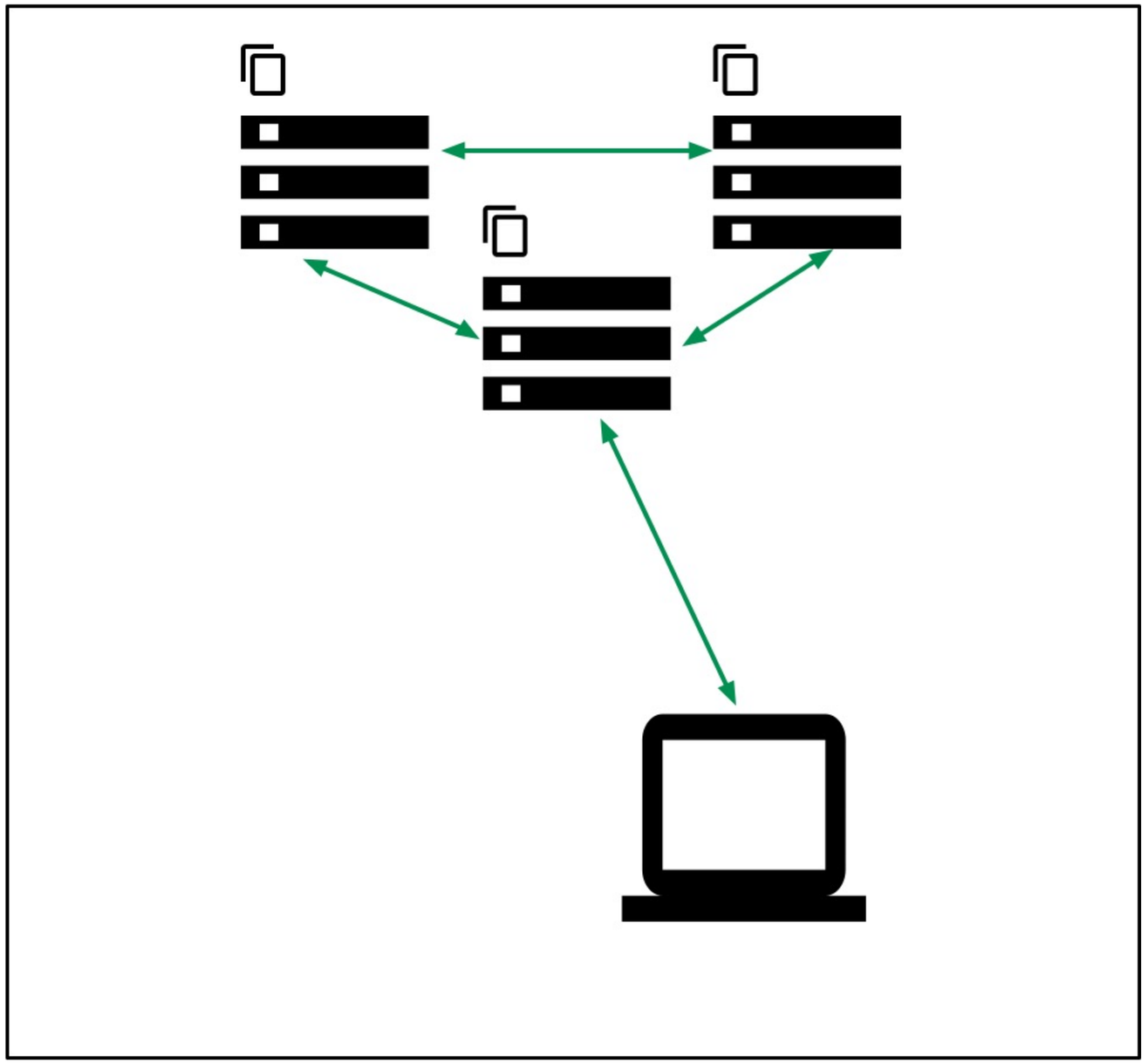
Simple Doc Store

Not even Mongo, just a file system



Simple MongoDB

With replication, the database becomes more resilient to hardware problems, network outages, improved READ performance, and even geopolitical issues.

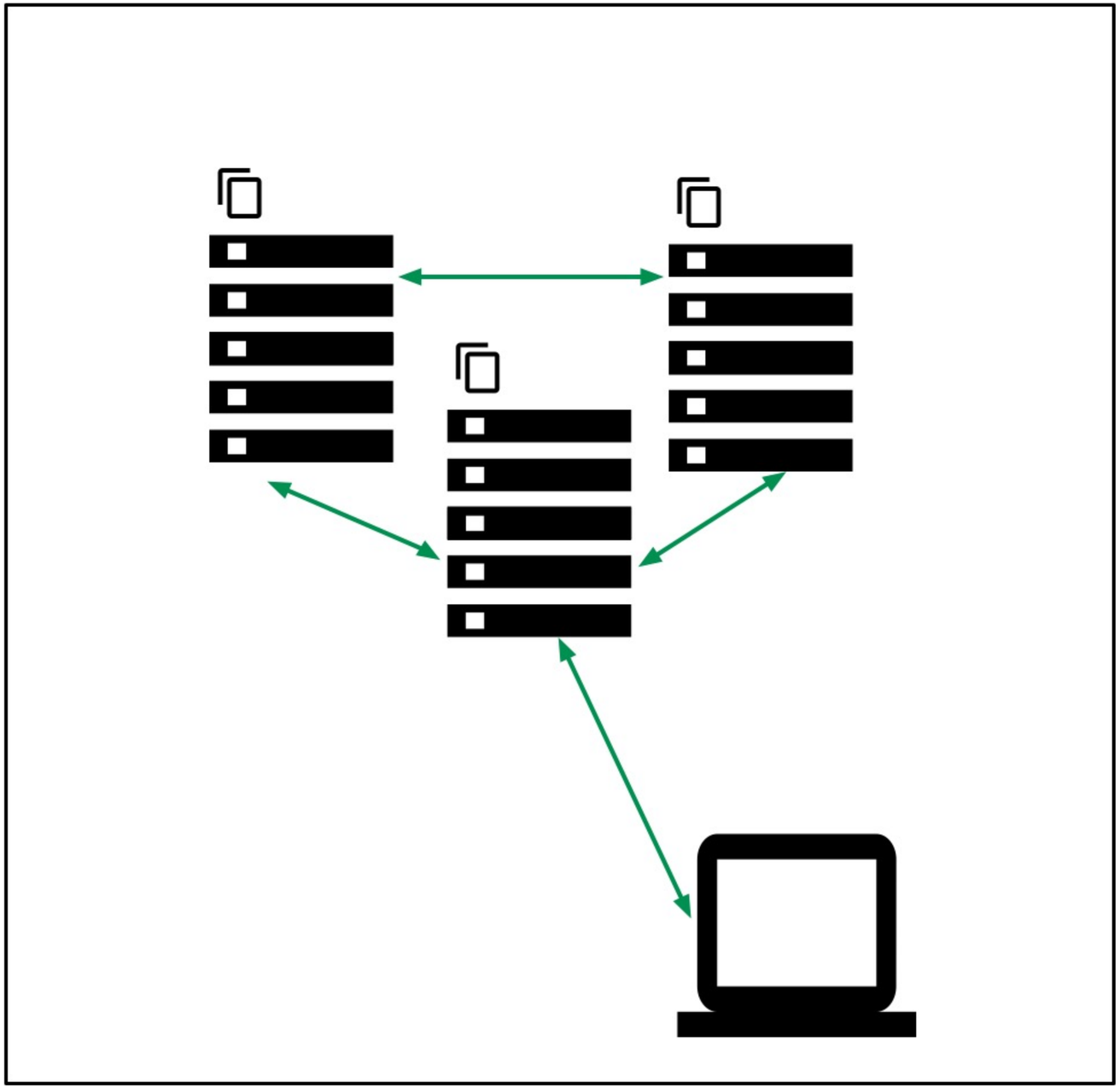


Vertical Scaling

Add more …

- storage
- memory
- CPU power

- **= $$$$**
- **Cloud providers only offer certain configurations**

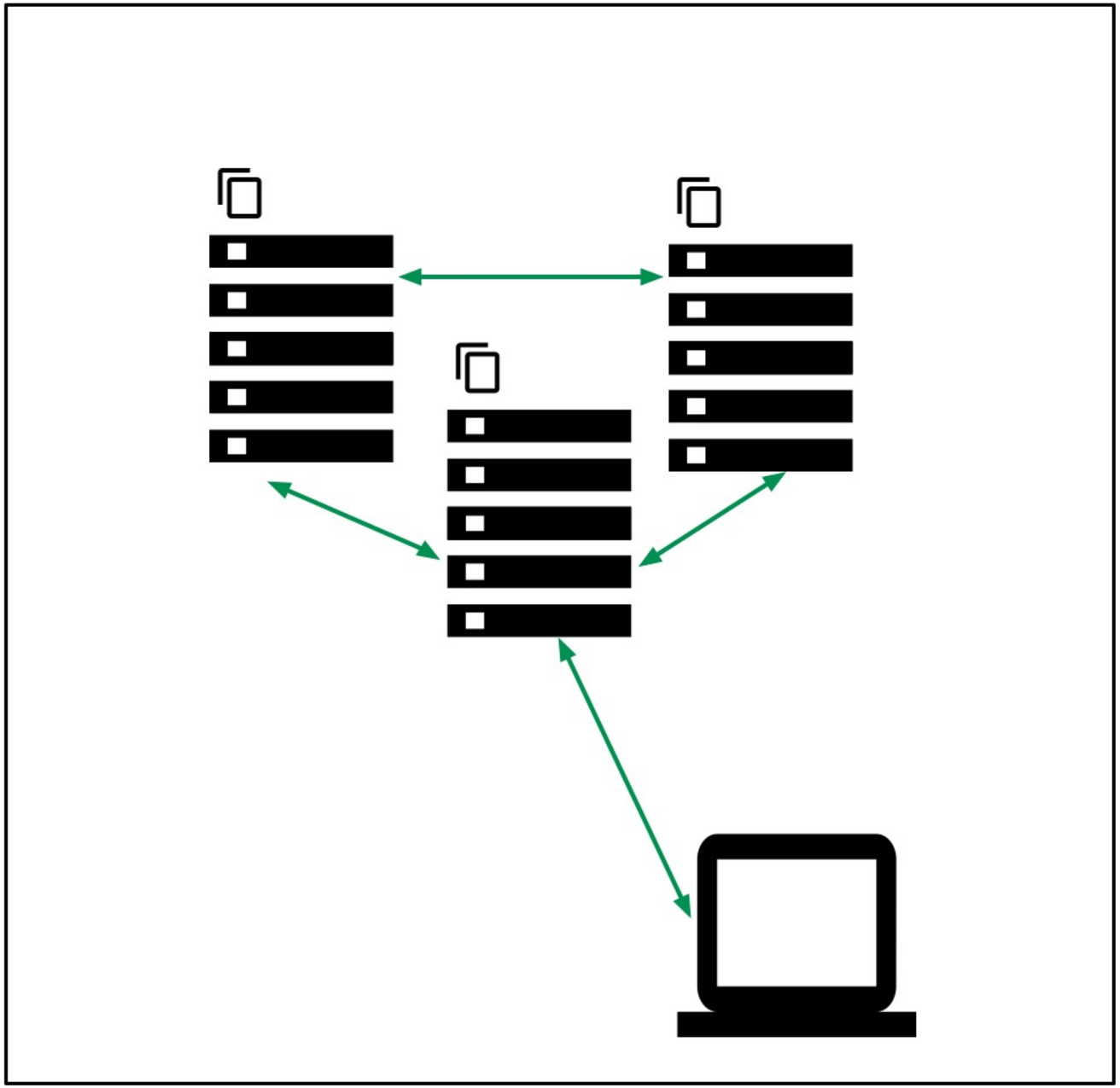


Horizontal scaling

Partition the database into **chunks** stored on different servers

Add additional servers to increase capacity as required.

The overall speed or capacity of a single machine may not be high, but focusing groups of servers on each part improves performance.



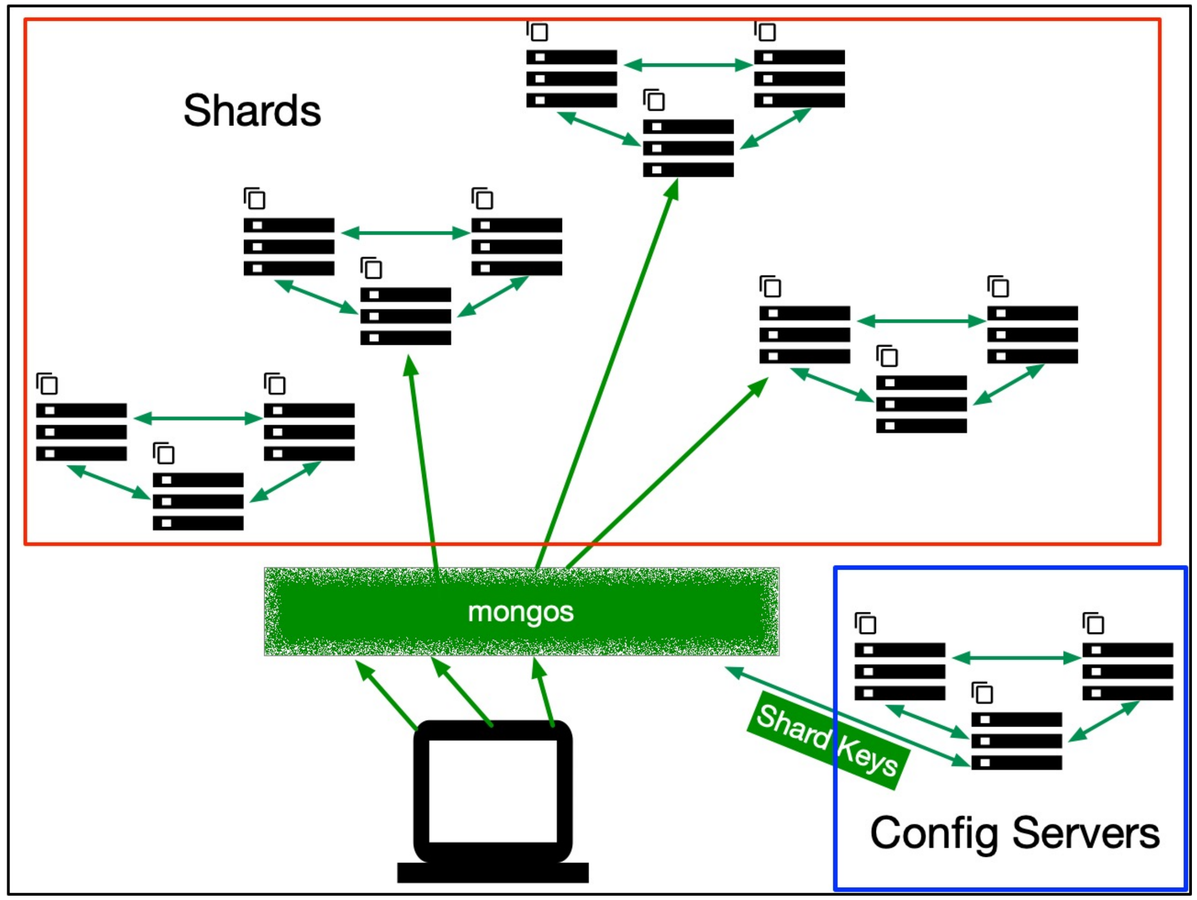
Horizontal scaling: mongoDB sharding

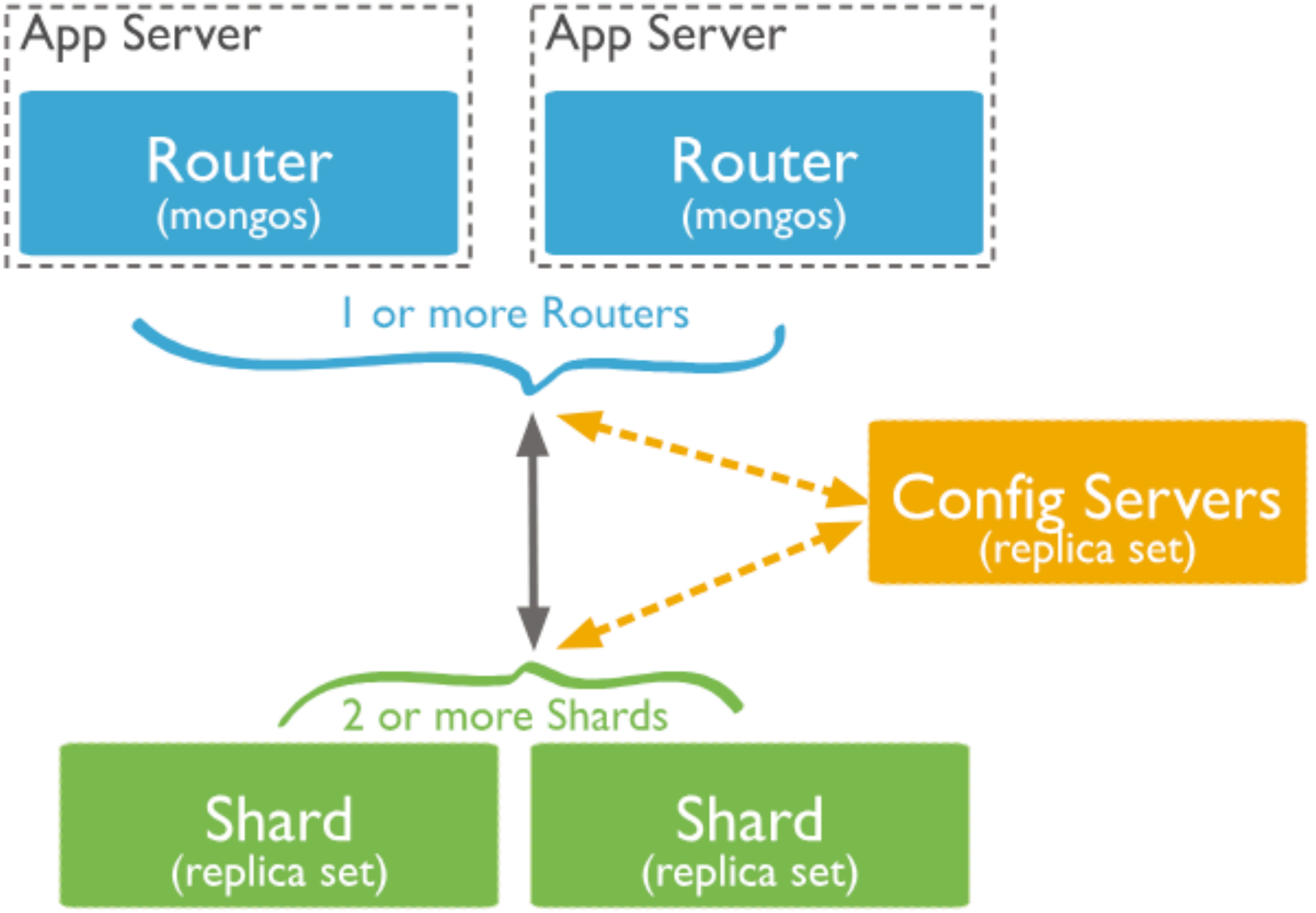
Expanding the capacity of the deployment only requires adding additional servers

Not necessarily faster or bigger

No special hardware

The trade off is increased **complexity** in infrastructure and maintenance for the deployment.



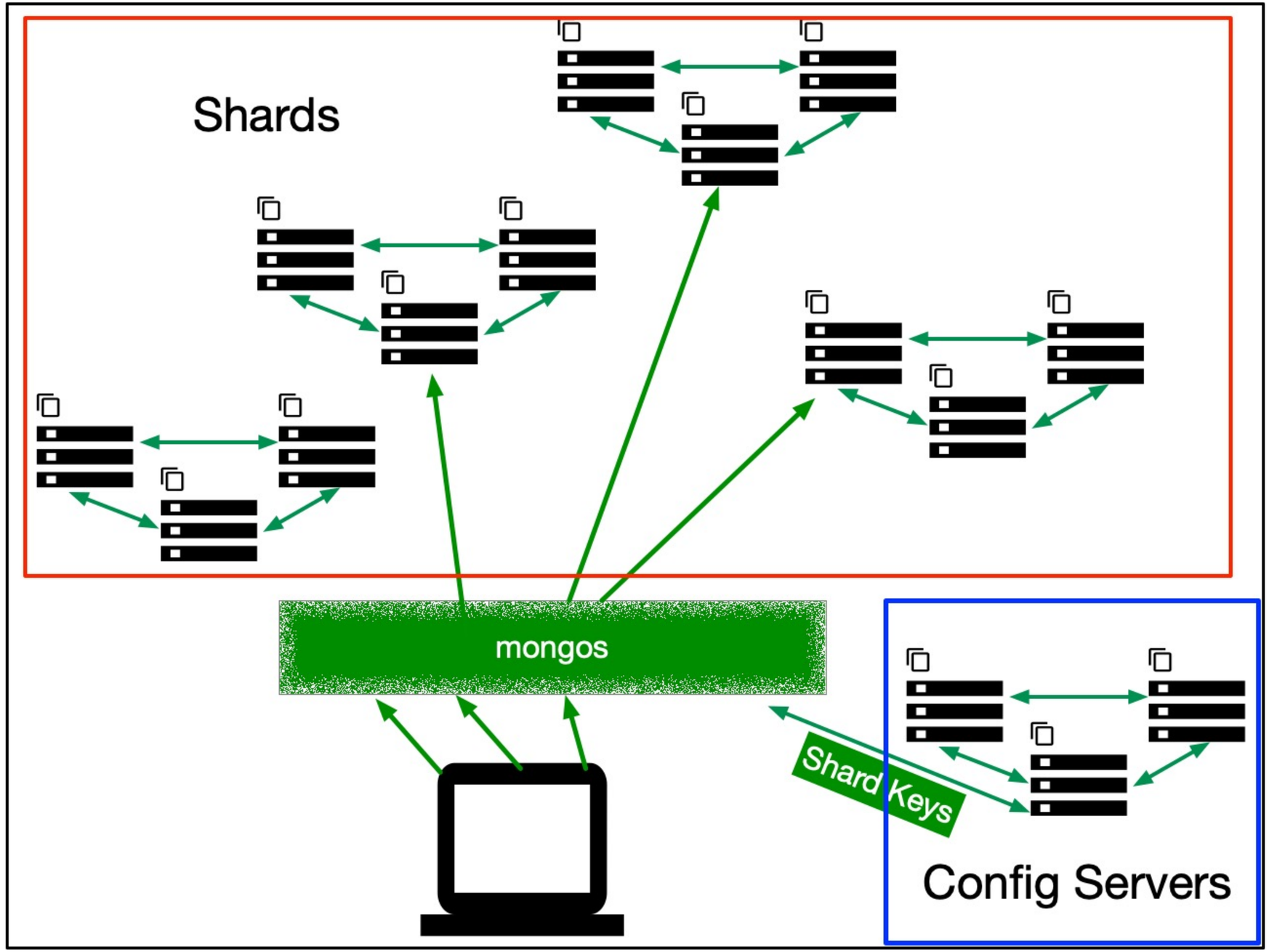

App Server
Router
(mongos)
App Server
Router
(mongos)
1 or more Routers
Config Servers
(replica set)
2 or more Shards
Shard
(replica set)
Shard
(replica set)

Sharding

The shard key is used to distribute the collection's documents across shards.

The shard key consists of a field or multiple fields in the documents.

Higher cardinality is better

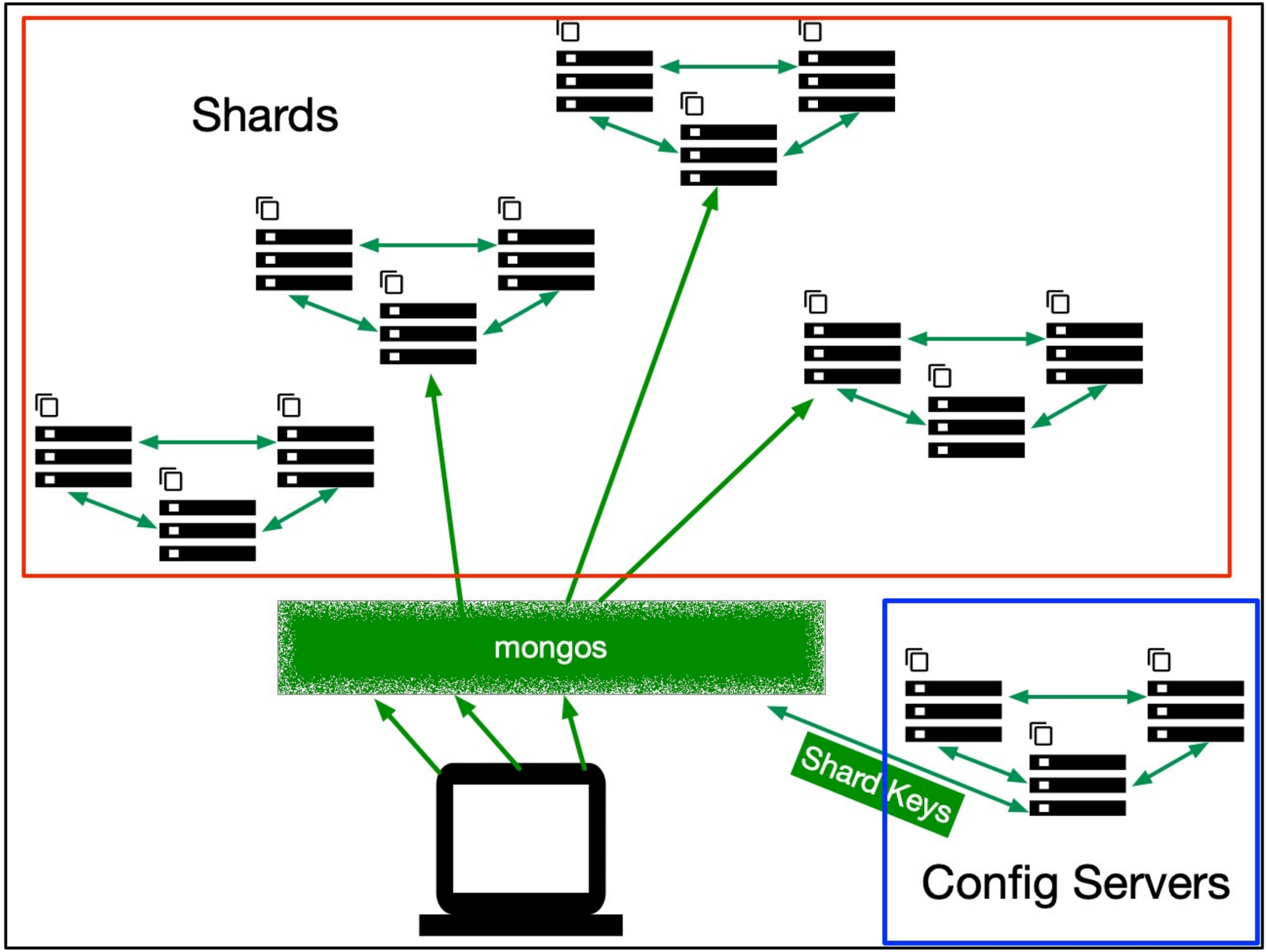


Sharding

Careful selection of the **shard key** is essential

The resulting distribution affects the efficiency and performance of operations

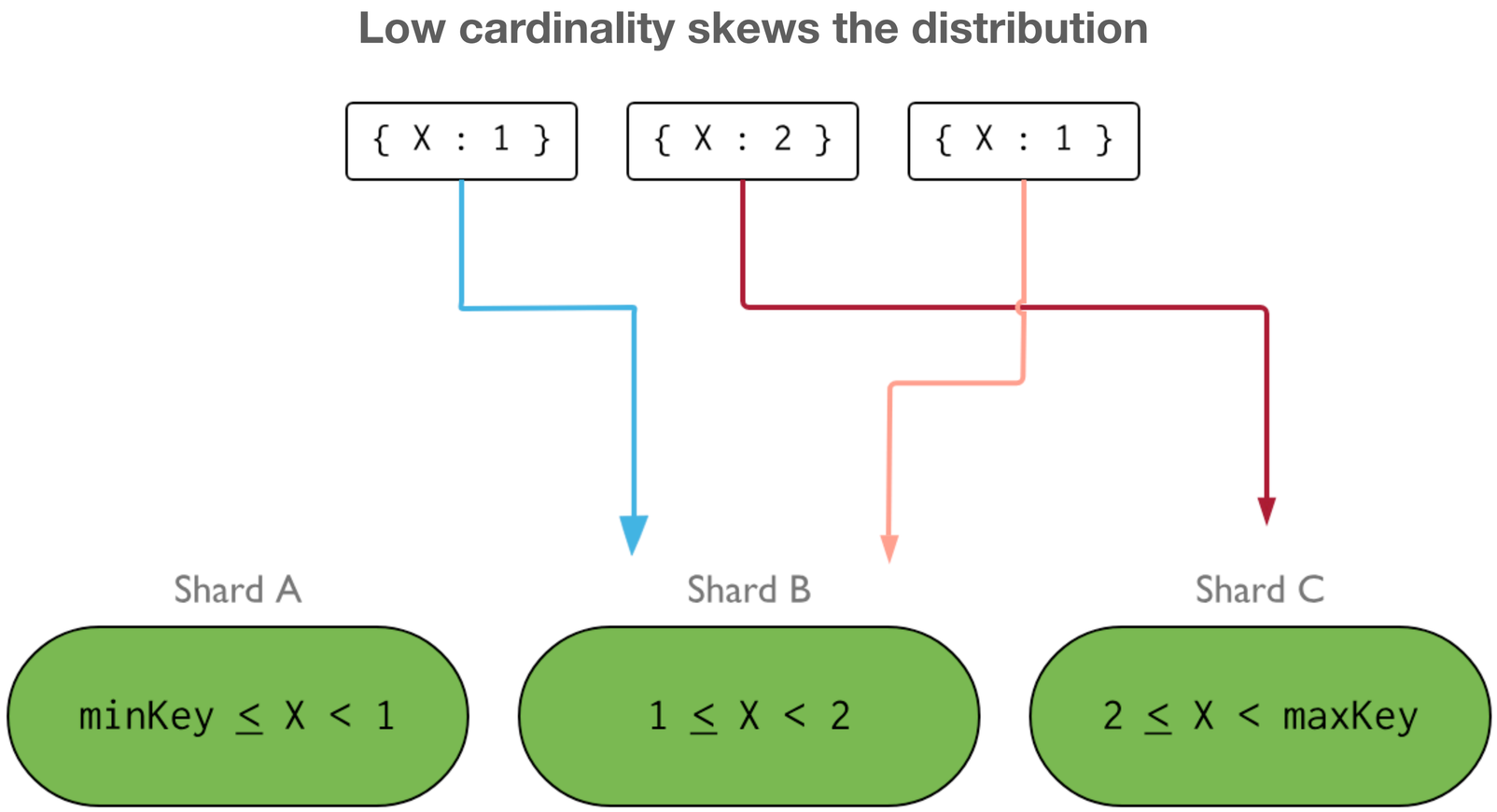
An ideal shard key distributes documents evenly ***while also*** facilitating common query patterns.



Shard key characteristics

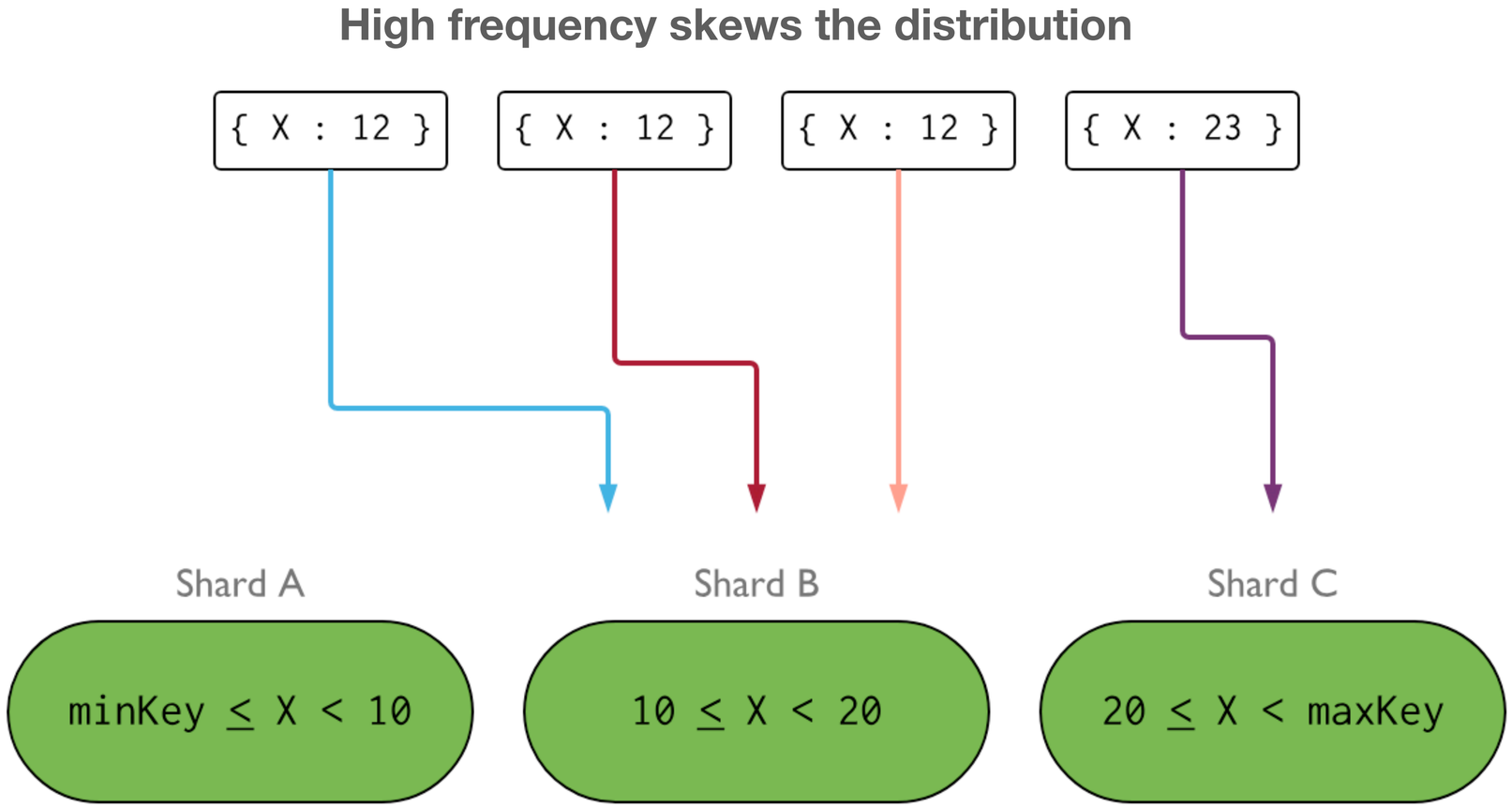
Cardinality is the maximum number of chunks the balancer can create

Each shard key can only exist on one chunk at any time



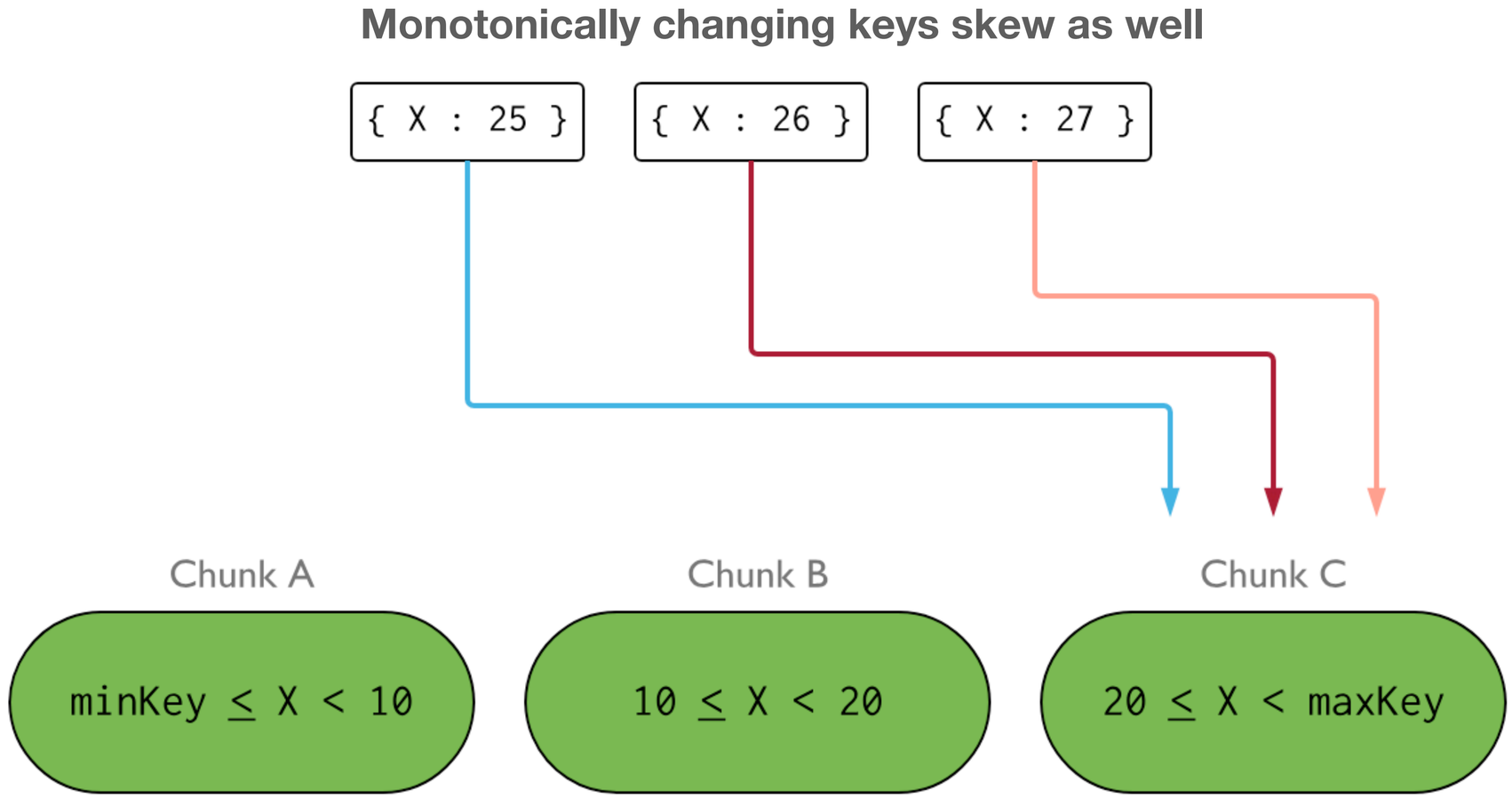
Shard key characteristics

Frequency of the shard key represents how often a given shard key value occurs in the data.



Shard key characteristics

A shard key on a **monotonically changing** value is more likely to distribute inserts to a single chunk within the cluster.



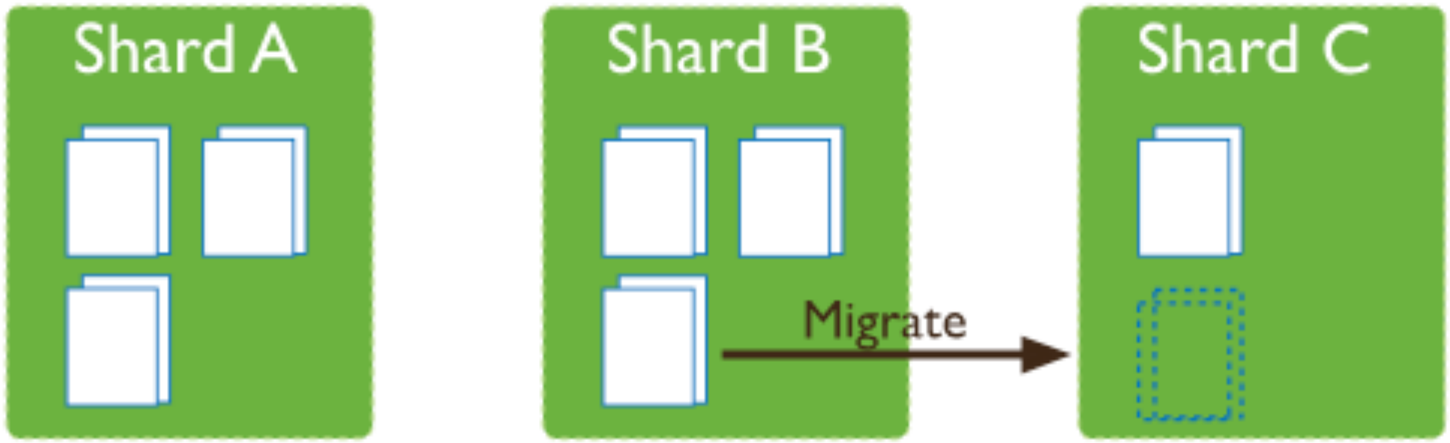
Shard key characteristics

The MongoDB balancer monitors the amount of data on each shard in a collection.

It will attempt to migrate data between shards to keep accesses balanced

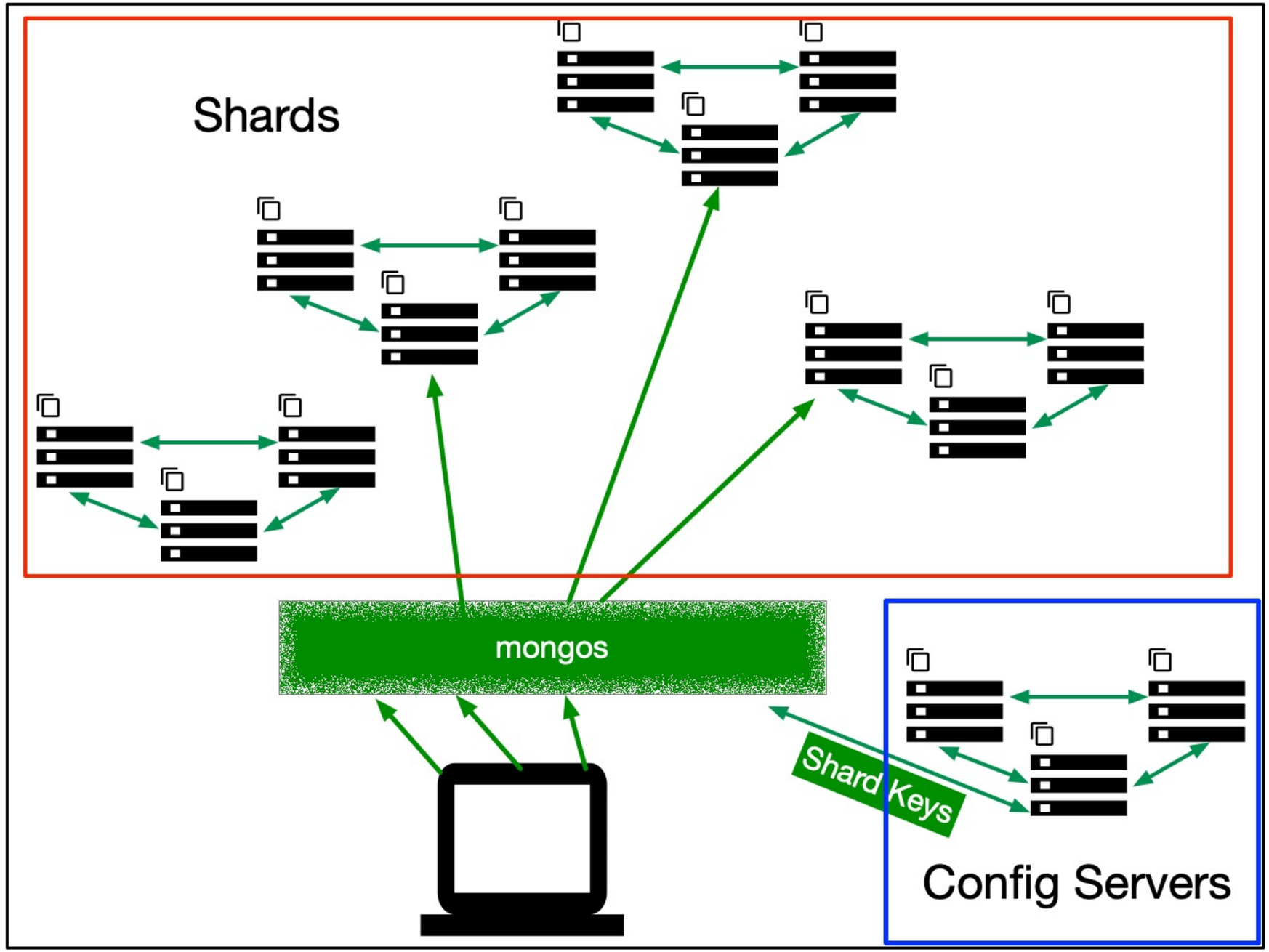
If one chunk is getting too big (>64MB), mongos will split it into multiples

Balancer migrating data between shards



Sharding

- `mongod` runs replicas
- Config servers manage the shards
- `mongos` routes queries to shards



To Shard or not to Shard ?

Sharing is complicated, make sure you really need it

- Running out of disk space and can’t add more?
- MongoDB works best with indexes and most common queries are kept resident in RAM.
    - If you start running out of RAM, your queries will slow down and your memory accesses will begin to thrash

Shard keys are (almost) forever

- Changing a shard key impacts performance while it's happening
- Must have sufficient cardinality - sharding across *n* replicas won’t work if the shard key cardinality is < *n*
- *Unique* attributes should be part of the shard key, since otherwise MongoDB can’t ensure the required uniqueness since *shards are independent*
- *Shards **only** index on _id and the shard key, since using any other field would require inter-shard communication*
- *If distribution of shard keys isn’t uniform (across shards), consider a multi-field composite shard key with the better key first.*