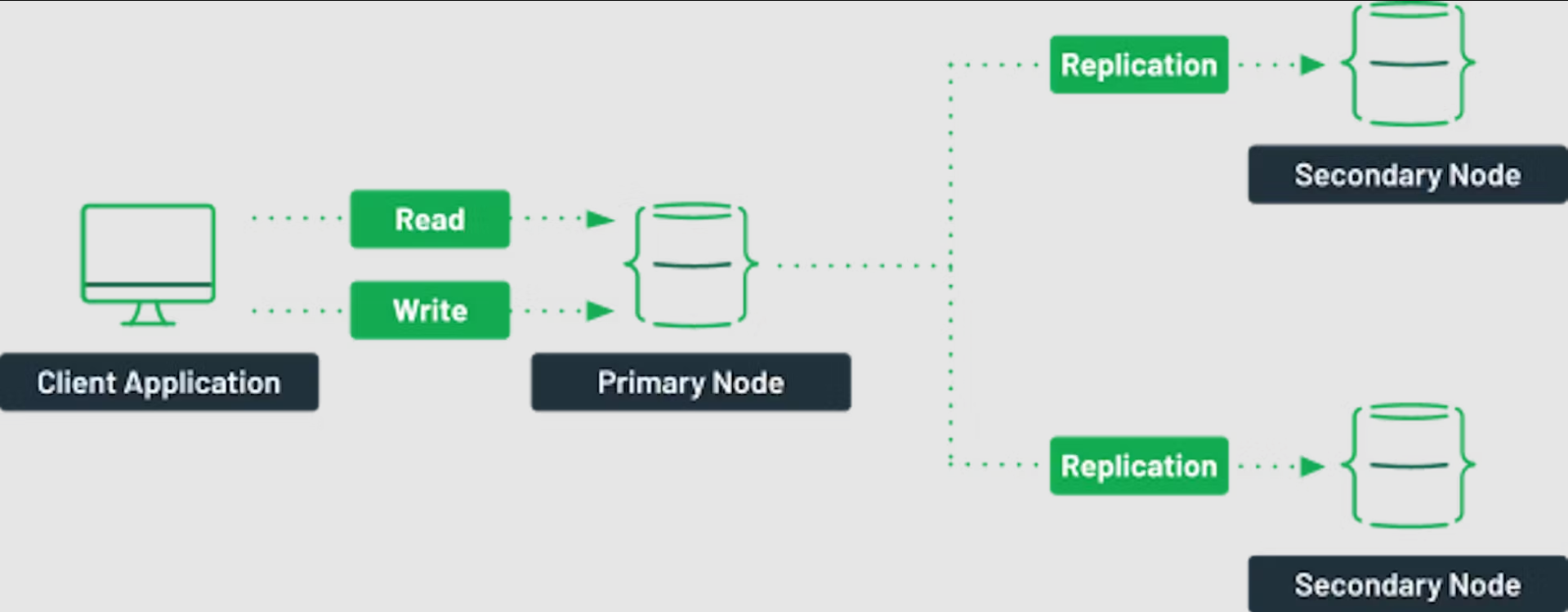
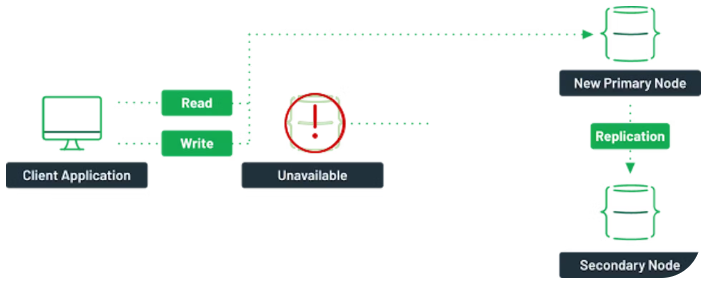
**Replica Sets**

A MongoDB replica set is a group of one or more servers containing the exact copy of the data. While it’s technically possible to have one or two nodes, the recommended minimum is three. A primary node is responsible for providing your application’s read and write operations, while two secondary nodes contain a replica of the data.

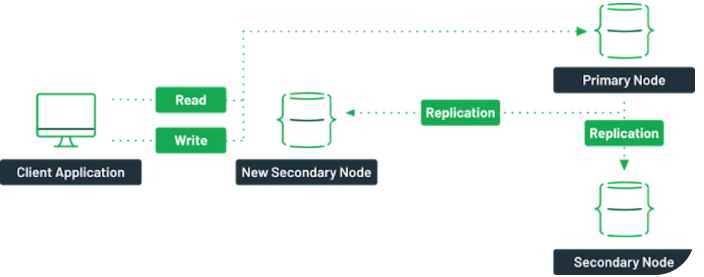


Should the primary node become unavailable for some reason, a new primary node would be picked by an “election process”. This new primary node is now responsible for the read and write operations.



*If a primary node is unavailable, the traffic from the client application is redirected to a new primary node.*

Once the faulty server comes back online, it will sync up with the primary node and become a new secondary node in the cluster.



*When the previous primary node comes back online, it comes back as a secondary node.*

The goal is to provide your application with high availability over your data. Even in a server failure, your client application can still connect to the cluster and access the data, reducing the overall potential downtime.

Step1: Setup mongodb

<https://www.mongodb.com/docs/manual/administration/install-community/>

Step2: With [keyfile](https://www.mongodb.com/docs/manual/core/security-internal-authentication/#std-label-internal-auth-keyfile) authentication, each [mongod](https://www.mongodb.com/docs/manual/reference/program/mongod/#mongodb-binary-bin.mongod) instances in the replica set uses the contents of the keyfile as the shared password for authenticating other members in the deployment. Only [mongod](https://www.mongodb.com/docs/manual/reference/program/mongod/#mongodb-binary-bin.mongod) instances with the correct keyfile can join the replica set.

* Create keyfile

|  |
| --- |
| openssl rand -base64 756 > <path-to-keyfile> |
| chmod 400 <path-to-keyfile> |

* Copy the keyfile to each replica set member.
  + Example:
    - (Windows) C:\Program Files\MongoDB\Server\7.0\keyfile.txt
    - (Linux) /mongokeys/keyfile.txt

Step3: Config mongodb.

# mongod.conf

# for documentation of all options, see:

# http://docs.mongodb.org/manual/reference/configuration-options/

# Where and how to store data.

storage:

dbPath: C:\Program Files\MongoDB\Server\7.0\data

# where to write logging data.

systemLog:

destination: file

logAppend: true

path: C:\Program Files\MongoDB\Server\7.0\log\mongod.log

# network interfaces

net:

port: 27017

bindIp: 0.0.0.0

#processManagement:

security:

authorization: enabled

keyFile: C:\Program Files\MongoDB\Server\7.0\keyfile.txt

#operationProfiling:

replication:

replSetName: rs0

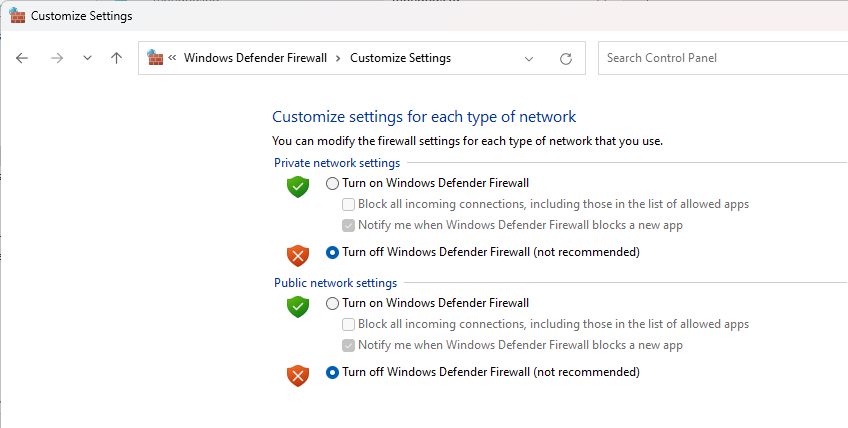
#sharding:

## Enterprise-Only Options:

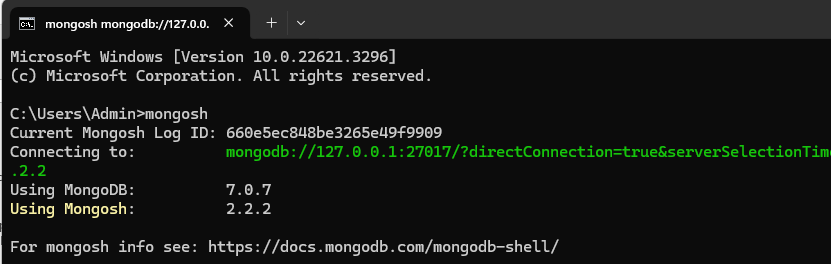
#auditLog:

Step4: Restart mongodb after change config in step3.

Step5: change firewall setting.



Step6: Access vào mongo để config replica bằng lệnh mongosh



Step6.1: Create admin user

use admin

db.createUser(

{

user: "tna2",

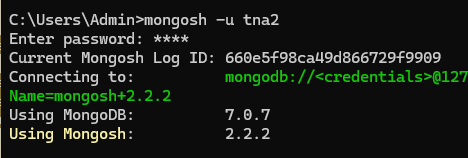
pwd: "tna2",

roles: [ { role: "root", db: "admin" } ]

}

)

Step6.2: Login using created user



Step9: run rs.initiate()

rs.add(“<IPAdress>:27017”)

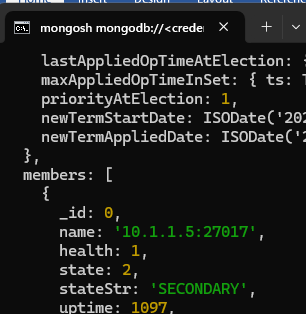
rs.status()





End: now you can test using

spring.data.mongodb.uri=mongodb://tna2:tna2@10.1.1.230:27017,10.1.1.5:27017,10.1.1.235:27017/?replicaSet=rs0



cfg = rs.conf()

cfg.members[0].host = "192.0.2.221"

rs.reconfig(cfg)