**1**

**Disable the balancer process.**

To disable the [balancer](https://docs.mongodb.com/manual/core/sharding-balancer-administration/#sharding-internals-balancing), connect the [mongo](https://docs.mongodb.com/manual/reference/program/mongo/#bin.mongo) shell to a [mongos](https://docs.mongodb.com/manual/reference/program/mongos/#bin.mongos) instance and run[sh.stopBalancer()](https://docs.mongodb.com/manual/reference/method/sh.stopBalancer/#sh.stopBalancer) in the config database.

use config

sh.stopBalancer()

For more information, see the [Disable the Balancer](https://docs.mongodb.com/manual/tutorial/manage-sharded-cluster-balancer/#sharding-balancing-disable-temporarily) procedure.

**WARNING**

If you do not stop the balancer, the backup could have duplicate data or omit data as [chunks](https://docs.mongodb.com/manual/reference/glossary/" \l "term-chunk)migrate while recording backups.

**2**

**Lock one secondary member of each replica set.**

Lock a secondary member of each replica set in the sharded cluster, and one secondary of the [config server replica set (CSRS)](https://docs.mongodb.com/manual/core/sharded-cluster-config-servers/" \l "replset-config-servers).

Ensure that the [oplog](https://docs.mongodb.com/manual/reference/glossary/" \l "term-oplog) has sufficient capacity to allow these secondaries to catch up to the state of the primaries after finishing the backup procedure. See [Oplog Size](https://docs.mongodb.com/manual/core/replica-set-oplog/" \l "replica-set-oplog-sizing) for more information.

**Lock shard replica set secondary.**

For each shard replica set in the sharded cluster, connect a [mongo](https://docs.mongodb.com/manual/reference/program/mongo/#bin.mongo) shell to the secondary member’s[mongod](https://docs.mongodb.com/manual/reference/program/mongod/#bin.mongod) instance and run [db.fsyncLock()](https://docs.mongodb.com/manual/reference/method/db.fsyncLock/" \l "db.fsyncLock" \o "db.fsyncLock()).

db.fsyncLock()

When calling [db.fsyncLock()](https://docs.mongodb.com/manual/reference/method/db.fsyncLock/" \l "db.fsyncLock" \o "db.fsyncLock()), ensure that the connection is kept open to allow a subsequent call to[db.fsyncUnlock()](https://docs.mongodb.com/manual/reference/method/db.fsyncUnlock/#db.fsyncUnlock).

**Lock config server replica set secondary.**

If locking a secondary of the CSRS, confirm that the member has replicated data up to some control point. To verify, first connect a [mongo](https://docs.mongodb.com/manual/reference/program/mongo/#bin.mongo) shell to the CSRS primary and perform a write operation with ["majority"](https://docs.mongodb.com/manual/reference/write-concern/#writeconcern.%22majority%22) write concern on a control collection:

use config

db.BackupControl.findAndModify(

{

query: { \_id: 'BackupControlDocument' },

update: { $inc: { counter : 1 } },

**new**: **true**,

upsert: **true**,

writeConcern: { w: 'majority', wtimeout: 15000 }

}

);

The operation should return either the newly inserted document or the updated document:

{ "\_id" : "BackupControlDocument", "counter" : 1 }

Query the CSRS secondary member for the returned control document. Connect a [mongo](https://docs.mongodb.com/manual/reference/program/mongo/#bin.mongo) shell to the CSRS secondary to lock and use [db.collection.find()](https://docs.mongodb.com/manual/reference/method/db.collection.find/" \l "db.collection.find" \o "db.collection.find()) to query for the control document:

rs.slaveOk();

use config;

db.BackupControl.find(

{ "\_id" : "BackupControlDocument", "counter" : 1 }

).readConcern('majority');

If the secondary member contains the latest control document, it is safe to lock the member. Otherwise, wait until the member contains the document or select a different secondary member that contains the latest control document.

To lock the secondary member, run [db.fsyncLock()](https://docs.mongodb.com/manual/reference/method/db.fsyncLock/" \l "db.fsyncLock" \o "db.fsyncLock()) on the member:

db.fsyncLock()

When calling [db.fsyncLock()](https://docs.mongodb.com/manual/reference/method/db.fsyncLock/" \l "db.fsyncLock" \o "db.fsyncLock()), ensure that the connection is kept open to allow a subsequent call to [db.fsyncUnlock()](https://docs.mongodb.com/manual/reference/method/db.fsyncUnlock/" \l "db.fsyncUnlock" \o "db.fsyncUnlock()).

**3**

**Backup one config server.**

Run [mongodump](https://docs.mongodb.com/manual/reference/program/mongodump/" \l "bin.mongodump" \o "bin.mongodump) against a config server [mongod](https://docs.mongodb.com/manual/reference/program/mongod/" \l "bin.mongod" \o "bin.mongod) instance to back up the cluster’s metadata. You only need to back up one config server. Perform this step against the locked config server.

Use [mongodump](https://docs.mongodb.com/manual/reference/program/mongodump/" \l "bin.mongodump" \o "bin.mongodump) with the [--oplog](https://docs.mongodb.com/manual/reference/program/mongodump/#cmdoption-mongodump-oplog) option to backup one of the [config servers](https://docs.mongodb.com/manual/core/sharded-cluster-config-servers/" \l "sharding-config-server).

mongodump --oplog

If your deployment uses CSRS config servers, unlock the config server node before proceeding to the next step. To unlock the CSRS member, use [db.fsyncUnlock()](https://docs.mongodb.com/manual/reference/method/db.fsyncUnlock/" \l "db.fsyncUnlock" \o "db.fsyncUnlock()) method in the [mongo](https://docs.mongodb.com/manual/reference/program/mongo/#bin.mongo) shell used to lock the instance.

db.fsyncUnlock()

**4**

**Back up a replica set member for each shard.**

Back up the locked replica set members of the shards using [mongodump](https://docs.mongodb.com/manual/reference/program/mongodump/" \l "bin.mongodump" \o "bin.mongodump) with the [--oplog](https://docs.mongodb.com/manual/reference/program/mongodump/#cmdoption-mongodump-oplog) option. You may back up the shards in parallel.

mongodump --oplog

**5**

**Unlock replica set members for each shard.**

To unlock the replica set members, use [db.fsyncUnlock()](https://docs.mongodb.com/manual/reference/method/db.fsyncUnlock/" \l "db.fsyncUnlock" \o "db.fsyncUnlock()) method in the [mongo](https://docs.mongodb.com/manual/reference/program/mongo/#bin.mongo) shell. For each locked member, use the same [mongo](https://docs.mongodb.com/manual/reference/program/mongo/#bin.mongo) shell used to lock the instance.

db.fsyncUnlock()

Allow these members to catch up with the state of the primary.

**6**

**Re-enable the balancer process.**

To re-enable to balancer, connect the [mongo](https://docs.mongodb.com/manual/reference/program/mongo/#bin.mongo) shell to a [mongos](https://docs.mongodb.com/manual/reference/program/mongos/#bin.mongos) instance and run[sh.setBalancerState()](https://docs.mongodb.com/manual/reference/method/sh.setBalancerState/#sh.setBalancerState).

use config

sh.setBalancerState(**true**)