Backup, Restore and Disaster Recovery

mongoDB

Agenda

- DR Overview
- Backup
- Restore
- Replication
- MongoDB Backup Service

Disaster Recovery Overview

Disasters do happen

Sometimes they are our fault

Recovery Point Objective

- How much data can you afford to lose?

Recovery Time Objective

- How long can you afford to be off-line?

DR vs. HA

- Don’t confuse the two
- Distinctly different business requirements
- Technical solutions may converge

DR Solution Tradeoffs

- Strict RPO = more $
- Strict RTO = more $

Backup

What’s the most important thing about creating backups?

Restoring Them

If you don't ensure that your backups can be restored, there's no point in doing backups

mongoDB

Backup Options

- mongodump
- Copy Files
- Snapshot disk

mongodump

- Dumps collections to *.bson files
- Mirrors your structure
- Can be run in live or offline mode
- --*dbpath* for direct file access
- --*oplog* to record oplog while backing up
- --*query/filter* selective dump

mongodump

```
$ mongodump --help
Export MongoDB data to BSON files.

options:
  --help                      produce help message
  -v [ --verbose ]            be more verbose (include multiple times
for more
                                             verbosity e.g. -vvvvv)
  --version                   print the program's version and exit
  -h [ --host ] arg           mongo host to connect to ( /s1,s2 for
  --port arg                  server port. Can also use --host hostname
  -u [ --username ] arg       username
  -p [ --password ] arg       password
  --dbpath arg                directly access mongod database files in
                              path, instead of connecting to a mongod
                              needs to lock the data directory, so can
                              if a mongod is currently accessing the s
  -d [ --db ] arg             database to use
  -c [ --collection ] arg collection to use (some commands)
  -o [ --out ] arg (=dump)output directory or "-" for stdout
  -q [ --query ] arg          json query
  --oplog                     Use oplog for point-in-time snapshotting
```

File System Backups

- **Must use journaling**
- Copy */data/db* files
- Snapshot
- Seriously, always use journaling

Ensure Consistency

- *fsyncLock* - flush and stop accepting writes
- Don't forget to *fsyncUnlock*

File System Backups: Pros and Cons

- Entire database
- Backup files will be large
- Fastest way to create a backup
- Fastest way to restore a backup

Restore

mongorestore

- *mongorestore*
- *--oplogReplay* replay oplog to point-in-time

File System Restores

- All database files
- Selected databases or collections
- Replay Oplog

Backup and Restore Examples

Backup Example: Sharded Cluser

1. Stop Balancer (and wait)
2. or no balancing window
3. Stop one config server (data R/O)
4. Backup Data (shards, config)
5. Restart config server
6. Resume Balancer

Restore Example: Sharded Cluster

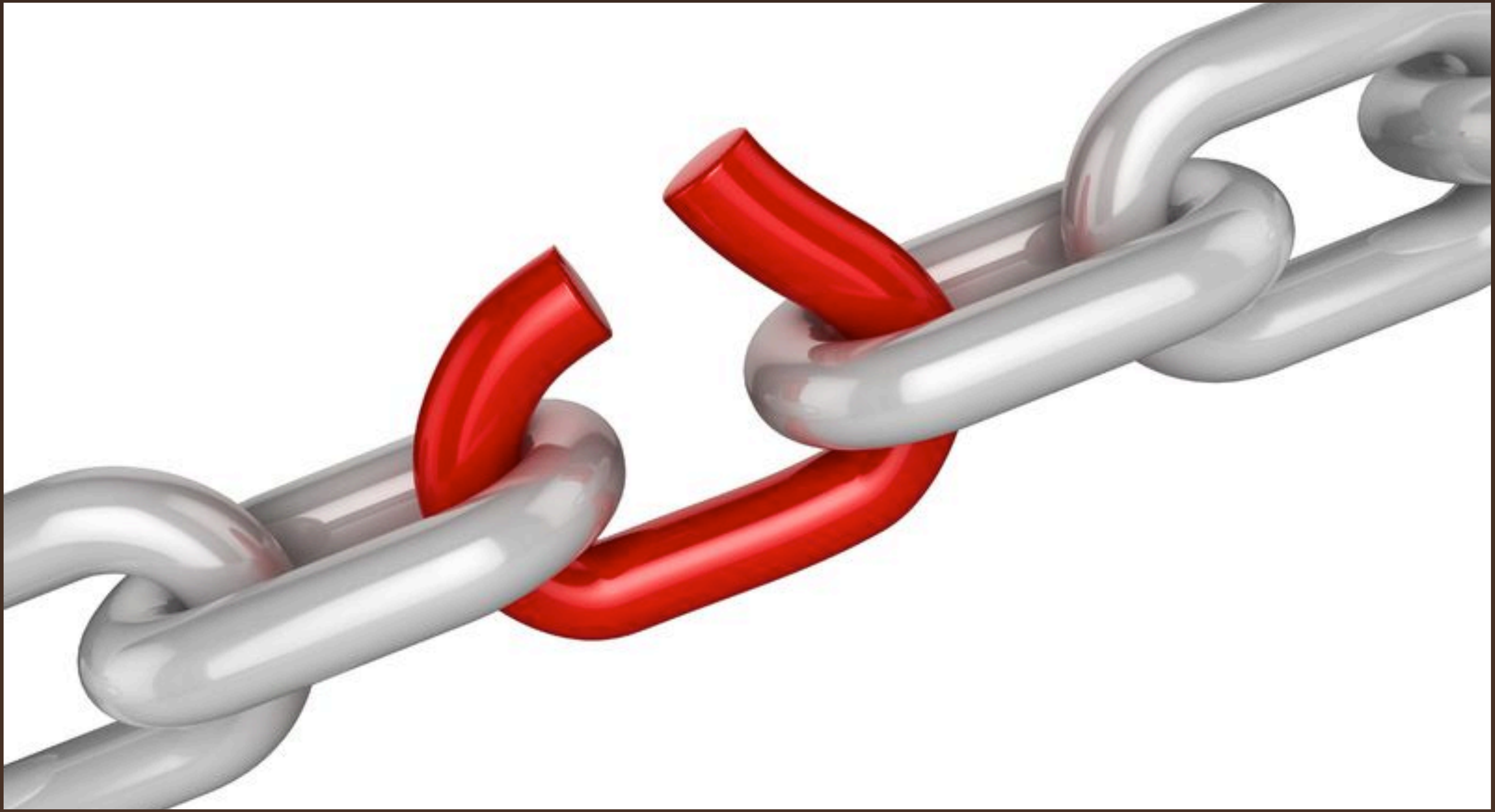
1. Dissimilar #shards to restore to
2. Different shard keys?
3. Selective restores
4. Consolidate shards
5. Changing addresses of config/shards

Tips and Tricks

- mongodump/mongoresore
  - --oplog[Replay]
  - --objcheck/--repair
  - --dbpath
  - --query/--filter
- bsondump
  - inspect data at console
- lvm snapshot time/space trade-off
  - Multi ESB backup
  - clean up snapshots

Replication

Replica Sets
Disaster Avoidance



Avoid a single point of failure

Replica Set Configuration

```
 > rs.conf() {
"_id" : "replSetName",
"version" : 3,
"members" : [
   {
     "_id" : 0,
     "host" : "myhost1.dnsname.com:27017"
   },
   {
     "_id" : 1,
     "host" : "myhost2.dnsname.com:27017"
   },
   {
     "_id" : 2,
     "host" : "myhost3.dnsname.com:27017"
   }
] }
```

Avoid single point of failure in replica sets

Deploy a Resilient Topology

- Redundancy
- Multiple Datacenters
- Multiple Regions
- Can support HA and DR requirements
  - HA by providing intra and inter datacenter failover
  - DR by providing geographically dispersed copies of data

MongoDB Management Service

You can do it yourself…
Or have the people who created MongoDB run your backups

mongoDB

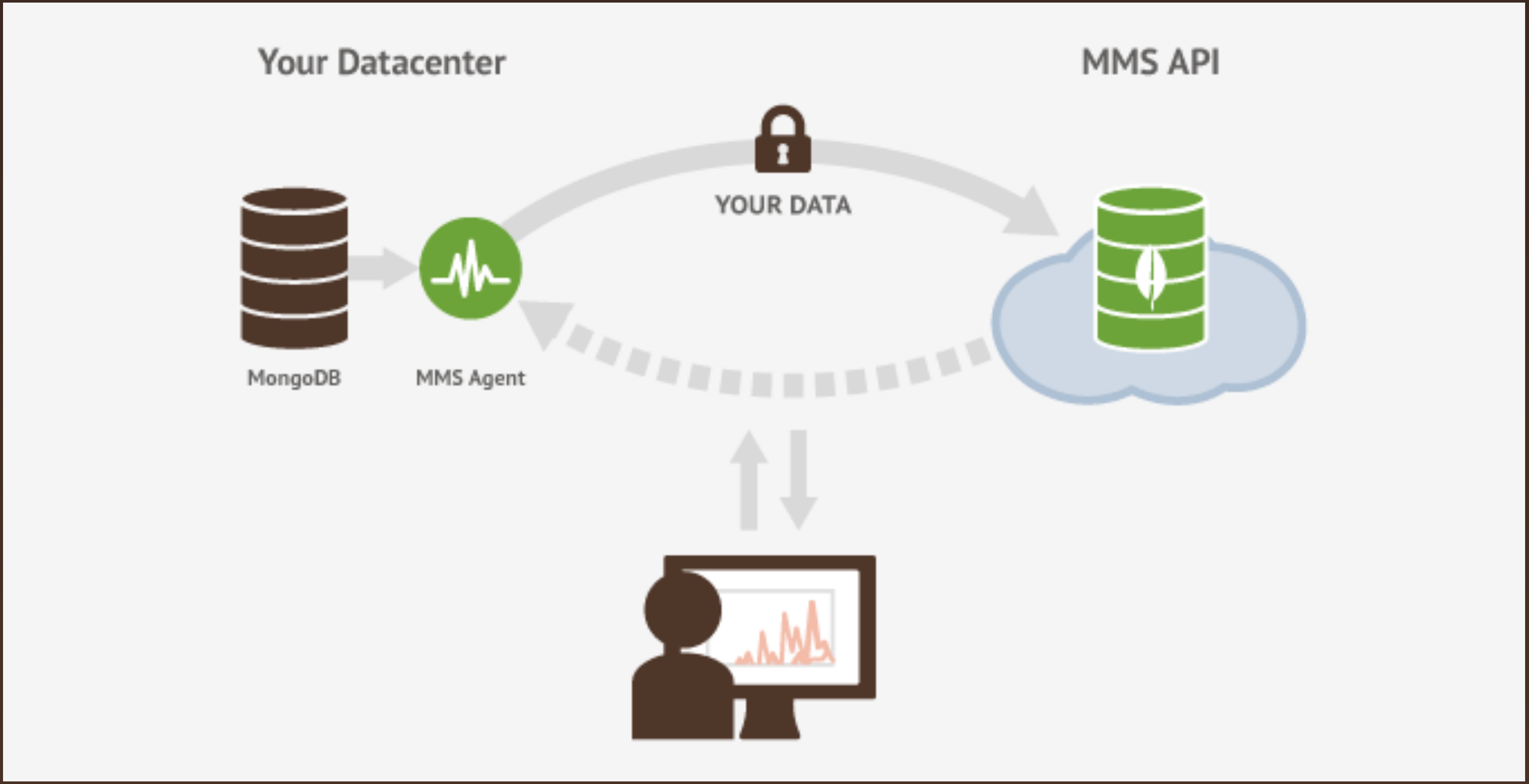
MongoDB Management Service

- Cloud-based backup and restore service
- Developed and monitored by 10gen engineers
- Point-in-time restore of replica sets
- Performance impact similar to adding a secondary
- Supports sharded clusters

Unlimited for restores

To seed new secondary nodes, build dev/QA systems, analytics and send data to new environments without impacting production workloads

Integrated into MMS UI



How it Works

Summary

Choose the Right Tool

- **RPO** on the order of seconds or minutes?
  - Use Replication
- **RPO** on the order of hours?
  - Maybe backups will suffice
- **RTO** on the order of seconds or minutes?
  - Use Replication
- **RTO** on the order of hours or days?
  - Use backups with warm/cold standby
- Need HA and DR?

Use Replica Sets

Design a topology to support both HA and DR

mongoDB

MongoDB Management Service

is a reliable and convenient way to protect your data

mongoDB