Couchbase

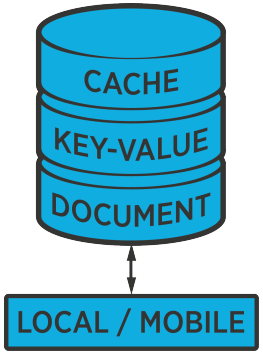
Couchbase Architecture and Administration Basics

Workshop Day 1

Introduction and Use Cases

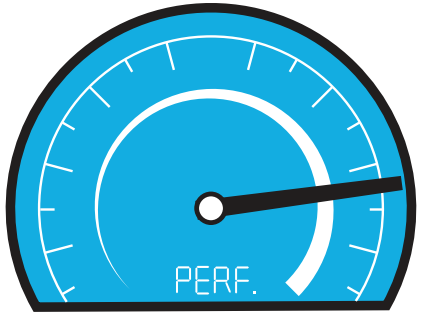
What makes Couchbase unique?

Enterprises choose Couchbase for several key advantages



Multi-purpose

Cache, key value store, document database, and local/mobile database in single platform



Performance & scalability leader

Sub millisecond latency with high throughput; memory-centric architecture

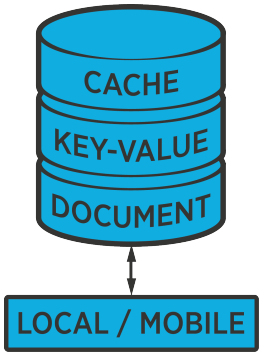
Always-on availability

Data replication across nodes, clusters, and data centers

Simplified administration

Easy to deploy & manage; integrated Admin Console, single-click cluster expansion & rebalance

Multi-purpose database supports many uses



Tunable built-in cache

- Consolidated cache and database
- Tune memory required based on application requirements

Flexible schemas with JSON

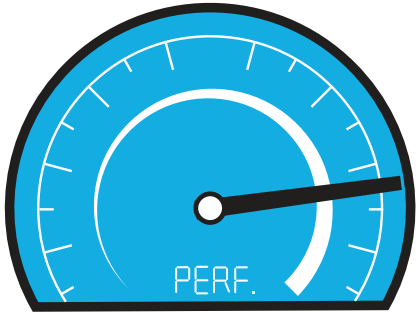
- Represent data with varying schemas using JSON on the server or on the device
- Index and query data with Javascript views

Couchbase Lite

- Light weight embedded DB for always available apps
- Sync Gateway syncs data seamlessly with Couchbase Server

4

Couchbase leads in performance and scalability



Auto Sharding

- No manual sharding
- Database manages data movement to scale out – not the user

Memory-memory XDCR

- Market's only memory-to-memory database replication across clusters and geos
- Provides disaster recover / data locality

Single Node Type

- Hugely simplifies management of clusters
- Easy to scale clusters by adding any number of nodes

Couchbase delivers always-on availability

High Availability

- In-memory replication with manual or automatic fail over
- Rack-zone awareness to minimize data unavailability

Disaster Recovery

- Memory-to-memory cross cluster replication across data centers or geos
- Active-active topology with bi-directional setup

Backup & Restore

- Full backup or Incremental backup with online restore
- Delta node catch-ups for faster recovery after failures

Simplified administration for exceptional ease of use

Online upgrades and operations

- Online software, hardware and DB upgrades
- Indexing, compaction, rebalance, backup & restore

Built-in enterprise class admin console

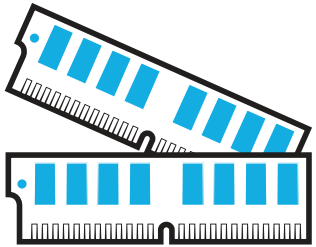
- Perform all administrative tasks with the click of a button
- Monitor status of the system visual at cluster level, database level, server level

Restful APIs

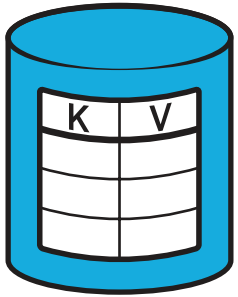
- All admin operations available via UI, REST APIs or CLI commands
- Integrate third party monitoring tools easily using REST

Couchbase provides a complete Data Management solution

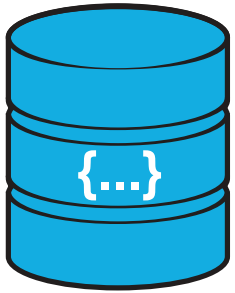
Multi-purpose capabilities support a broad range of apps and use cases



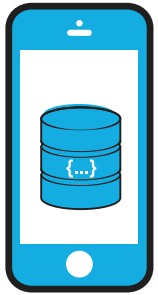
High availability cache



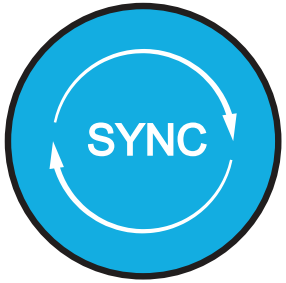
Key-value store



Document database



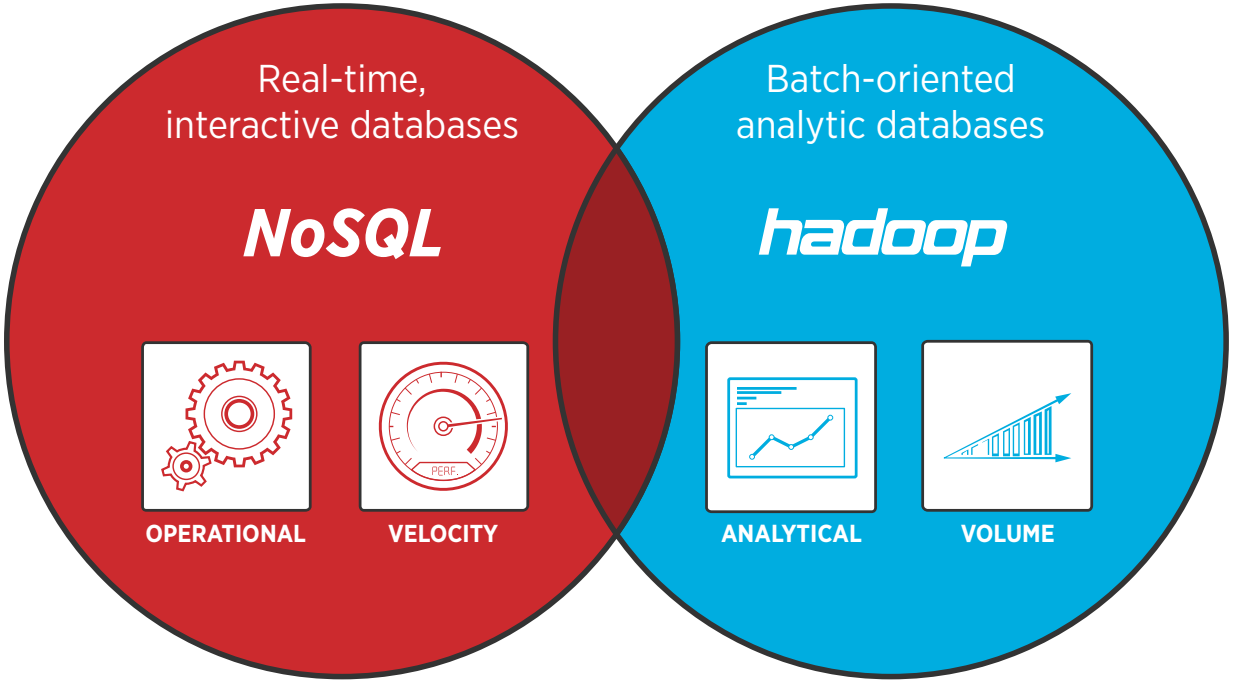
Embedded database



Sync management

Enterprises often start with cache, then broaden usage to other apps and use cases

Big Data = Operational + Analytic (NoSQL + Hadoop)

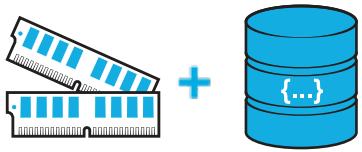


- Online
- Web/Mobile/IoT apps
- Millions of customers/consumers

- Offline
- Analytics apps
- Hundreds of business analysts

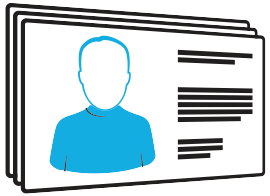
Use case Spotlights

HA Caching



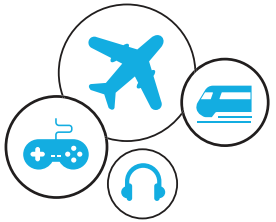
Consistently low latency and high throughput access to any type of data, alleviating load on backend systems and maintaining snappy user experience

Profile Management



Store, access, and update profile data to support user interactions, such as authentication, customer lookup, and preference tracking

Internet of Things

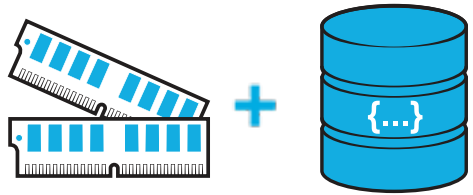


Ingest data from endpoint devices, such as sensors, smart meters, and other network devices, and scale to support massive datasets

HA Caching

The problem

Provide low latency and high throughput access to a variety of data types. Alleviate load on backend systems.



Application Requirements

- Lots of users accessing different datasets
- Data in varying formats: HTML, JSON, protobuf
- High read performance
- Uptime critical

Challenges with other caching technologies

- Complicated to setup and monitor
- Not persistent
- Restriction of supported data types
- Not truly distributed or clustered (i.e. ehcache)
- Poor performance under load

The Couchbase Solution

- Based on memcached = fast!
- Replicated and persistent with auto-failover
- Fully distributed and clustered with "push button" scaling: easy, inexpensive
- Support for binary and JSON data types

Caching @ Walmart

- Walmart.com
- Provide fast access to pricing, inventory, shipping information
- Black Friday success!

The Couchbase Advantage

Massive speed and scale that's easy to manage

The problem

Massive and spikey user traffic to small bits of data supporting web experience

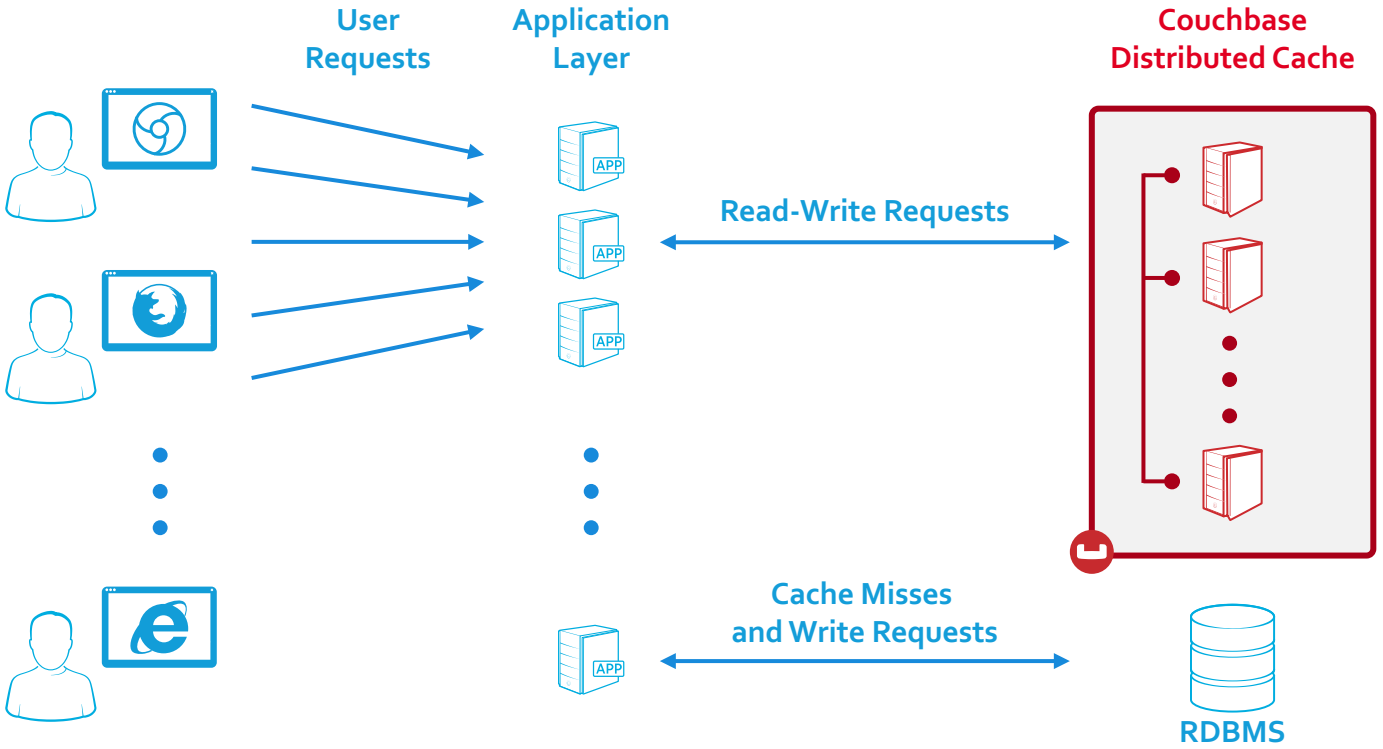
- Many different caching technologies
- New applications constantly coming online
- Huge catalog of products combined with regional pricing and shipping information

The solution

Deploy Couchbase Server as standardized distributed caching layer

- Compatible with memcached, highly optimized for latency and throughput
- Shared nothing, replicated and persistent for reliability
- Support for JSON as well as any binary data type
- Shared-nothing, replicated and persistent architecture

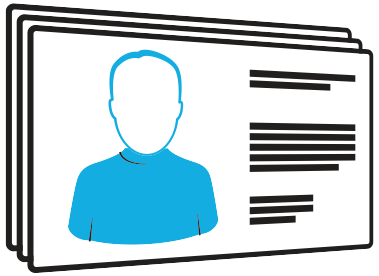
High Availability Caching at Walmart



Profile Management

Objective

Drive customer satisfaction and revenue by delighting customers with highly responsive experiences, based on up-to-date profile data



Business requirements

- Serve profile data rapidly at the moment of interaction
- Accommodate fast changing data types and attributes
- Support growing customer base – millions to billions

Technical requirements

- Low latency
- Data model flexibility
- Fast, easy, affordable scalability

The Couchbase Solution

- **Integrated cache** – provides fast performance, highly responsive interaction
- **Flexible JSON data model** – easily adapts new data types and attributes on the fly
- **Push-button scalability** on commodity hardware – fast, easy and inexpensive

Profile Management @ Apple

Background and Business Context

- Apple global ID store – iTunes, App Store, iCloud, Apple.com
- #1 mission-critical centralized service: store, update, access user profile data

Challenges & Requirements

- Previous system relied on relational DBMS (Oracle)
- Profile lookups were too slow
- Replication across data centers was slow and complex
- Scale to support rapid growth from 800M+ to 2B+ users globally

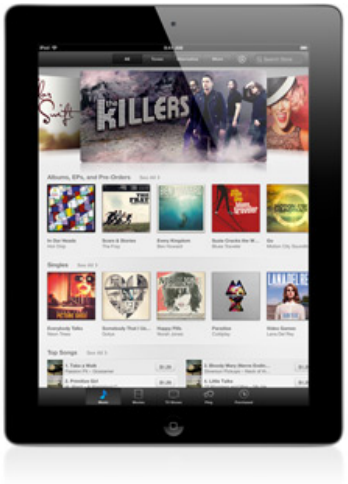
Objectives

- Improve read latency to enable highly responsive customer experience
- Simplify data replication across data centers globally
- Reduce reliance on Oracle over time

Profile Management with Couchbase @ Apple

Couchbase Solution

- Couchbase Server deployed as primary data store for user profile data
- Multiple Couchbase clusters are deployed at multiple data centers
- Each cluster has 2 replicas plus active copy of data
- Couchbase selected over Cassandra and MongoDB for:
  - Consistent high performance under very high read/write loads
  - Scale-out architecture
  - XDCR capabilities



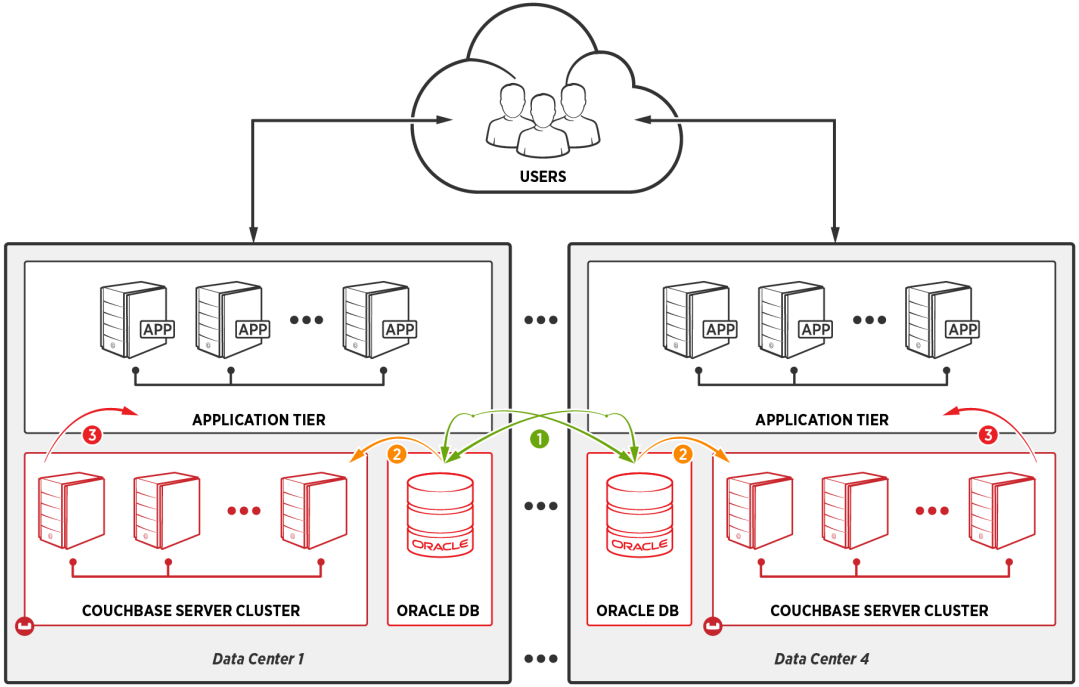
Results and Outlook

- Leverages integrated cache to store data in RAM for fast performance
- Delivers sustained high throughput: 200K reads and 20K writes/sec
- Phase 2 will deploy built-in XDCR to speed and simplify replication across multiple data centers

Profile Management with Couchbase @ Apple

Couchbase deployment at Apple – Phase 1

(live since Sept 2013)



1. All incoming data written to Couchbase and Oracle based on data silos
2. All application reads hit Couchbase
3. 3 copies of data maintained within each Couchbase cluster for high availability

In future phases, data will be replicated across data centers using Couchbase XDCR, and Couchbase will replace instances of Oracle

Internet of Things

Objective

Deliver new products and services, create new revenue opportunities, and drive agility by connecting with and harnessing data from millions of devices

Business requirements

- Manage massive datasets (e.g. billions of data points)
- Interact with numerous devices, sometimes unconnected
- Capture new and evolving data types at high speed

Technical requirements

- Scale to millions of devices, billions of data points
- High throughput
- Synchronize data between device and cloud
- Data model flexibility

The Couchbase Solution

- **Push-button scalability** – easily scales to support massive data volumes
- **Integrated cache** – enables high throughput
- **Embedded JSON database with automated sync –** supports connected and unconnected devices
- **Flexible JSON data model** – easily adapts new data types and attributes on the fly

Internet of Things @ Verizon

Background and Business Context

- Enterprises are seeing significant growth in the number and types of devices running on their corporate networks
- Enterprise customers can take advantage of data to monitor and better manage network devices

Objective

- Enable new service offering for Verizon enterprise customers to manage devices connect to their company's network

Challenges & Requirements

- Collect and store data in real time from 10K's-100K's of devices on a single customer's network
- Analyze data for usage statistics and patterns
- Provide near real-time insights and reports into device usage

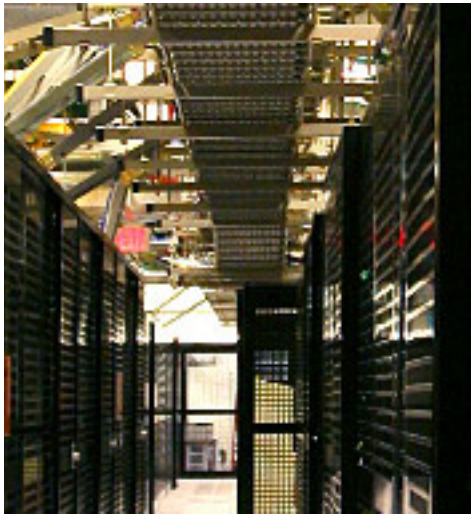
Internet of Things with Couchbase @ Verizon

Couchbase Solution

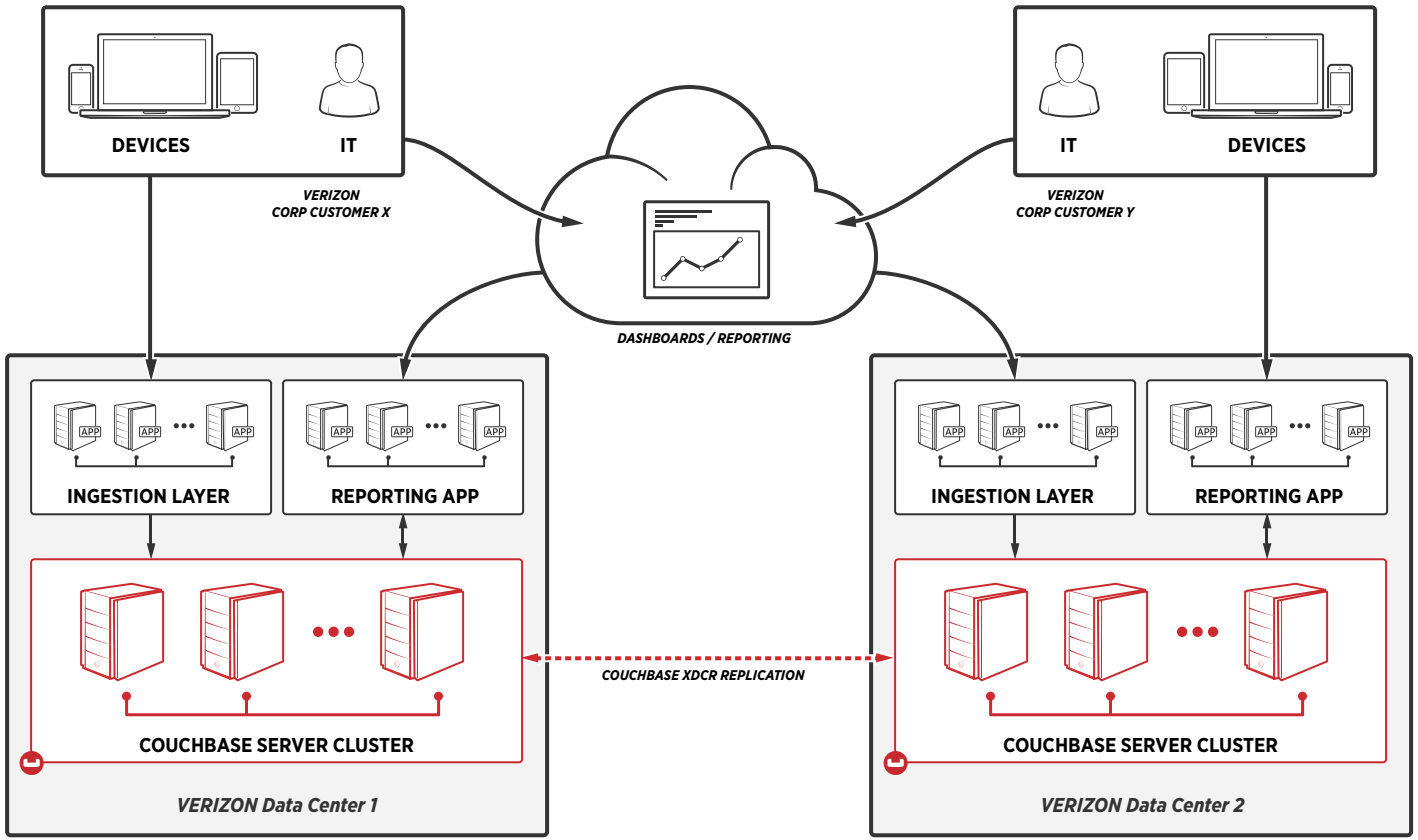
- Deploy Couchbase Server to store data and serve reports on network devices
- Couchbase Server ingests data at high speed, from any kind of connected device: alarms, locking systems, modems, solar panels, cash registers, etc.

Results and Outlook

- Stream-based indexing enables fast views and reports
- JSON data model easily handles any data type, new data types

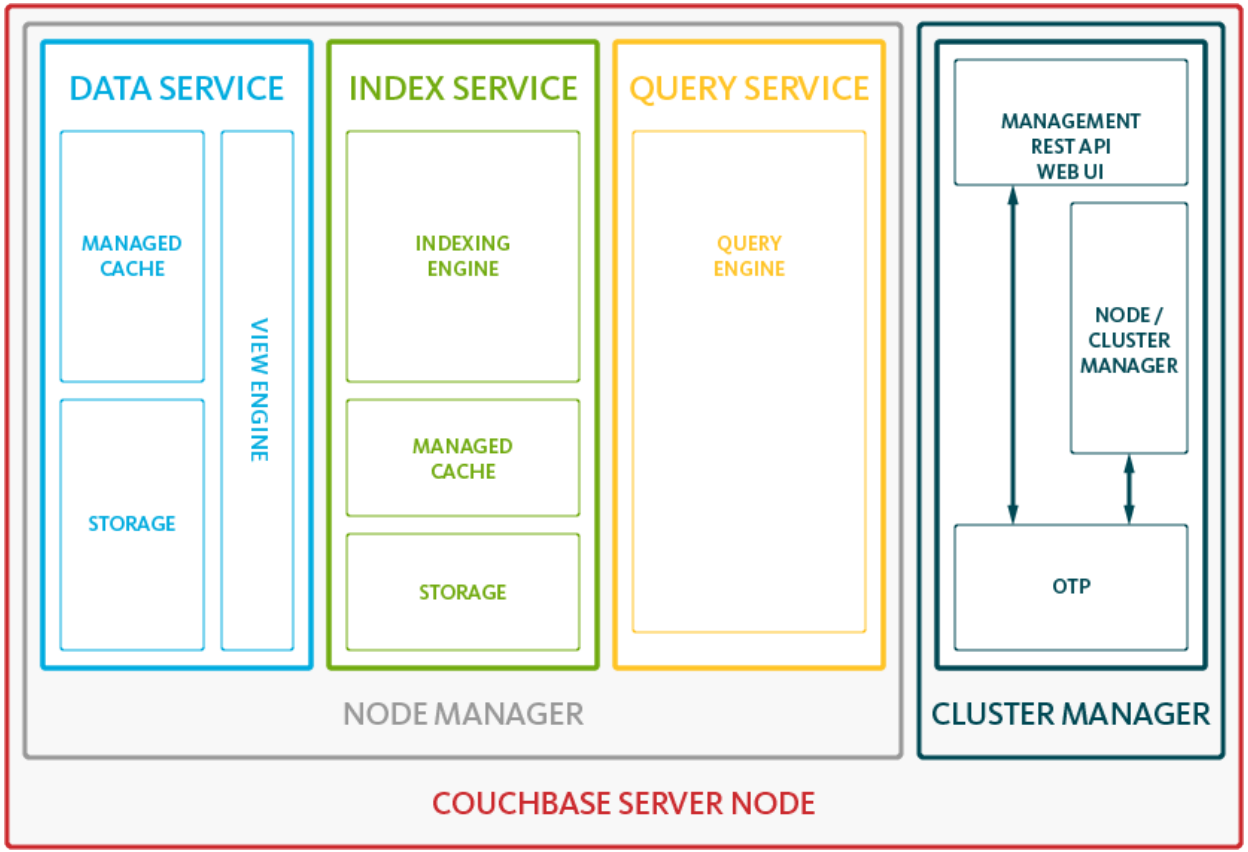


Internet of Things with Couchbase @ Verizon

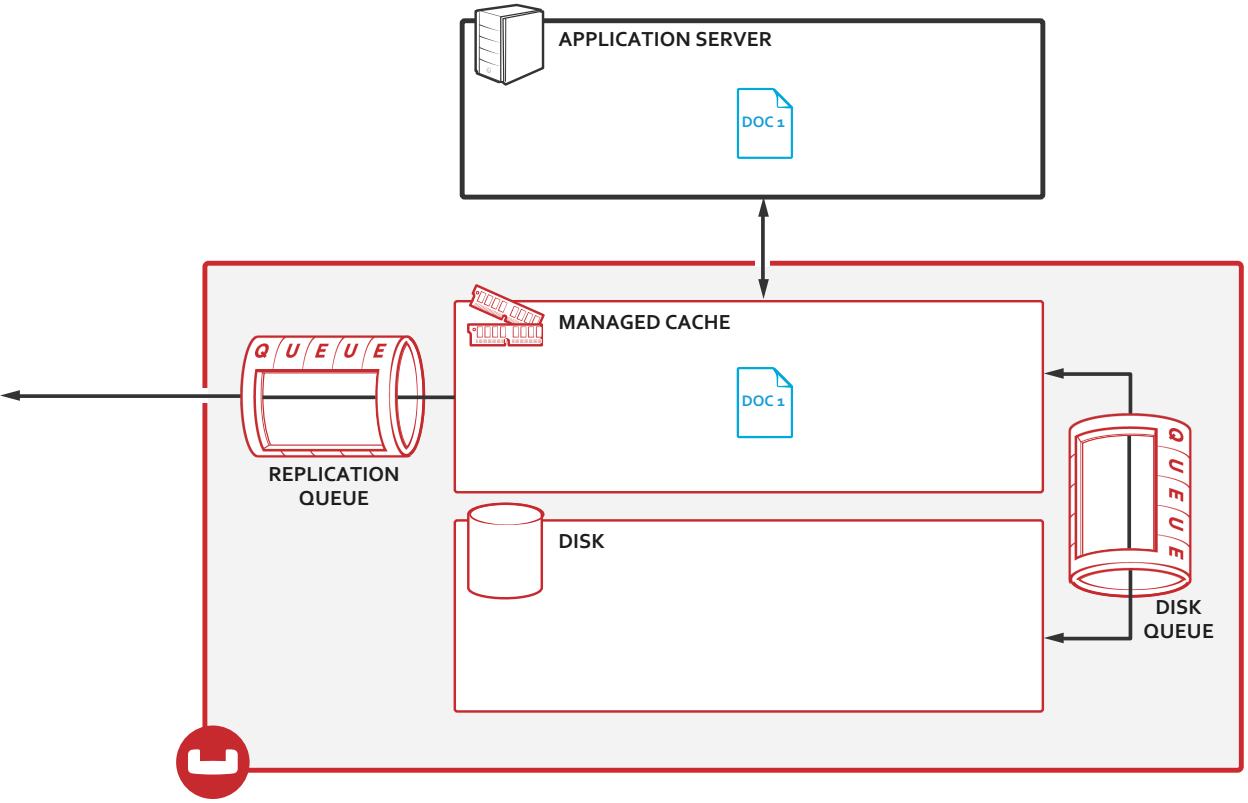


Couchbase Server Technical Architecture

Couchbase Server Architecture



Write Operation

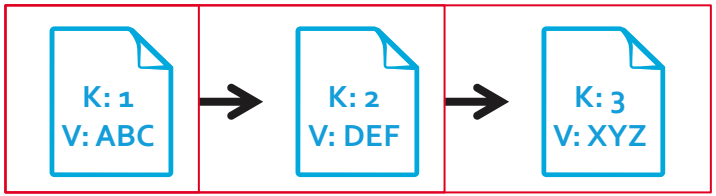


Single-node type means easier administration and scaling

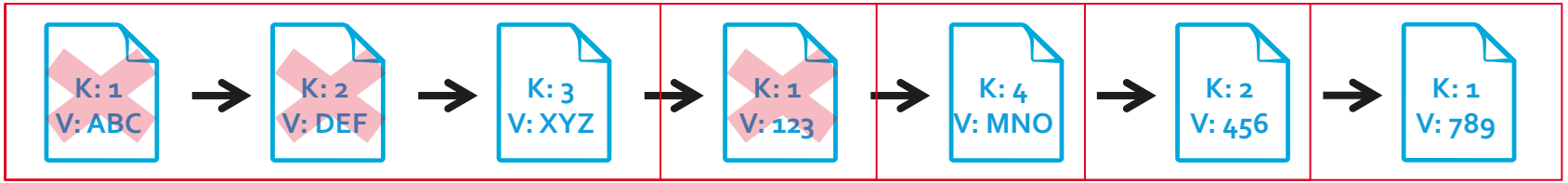
- Writes are async by default
- Application gets acknowledgement when successfully in RAM and can trade-off waiting for replication or persistence per-write
- Replication to 1, 2 or 3 other nodes
- Replication is RAM-based so extremely fast
- Off-node replication is primary level of HA
- Disk written to as fast as possible – no waiting

Append only Storage

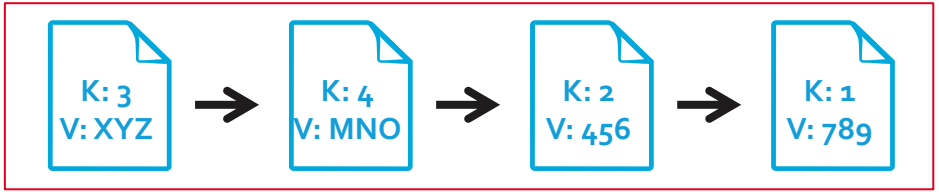
Initial file layout:



Update some data:



After compaction:



Couchbase Server as a Distributed System

Multi Dimensional Scaling

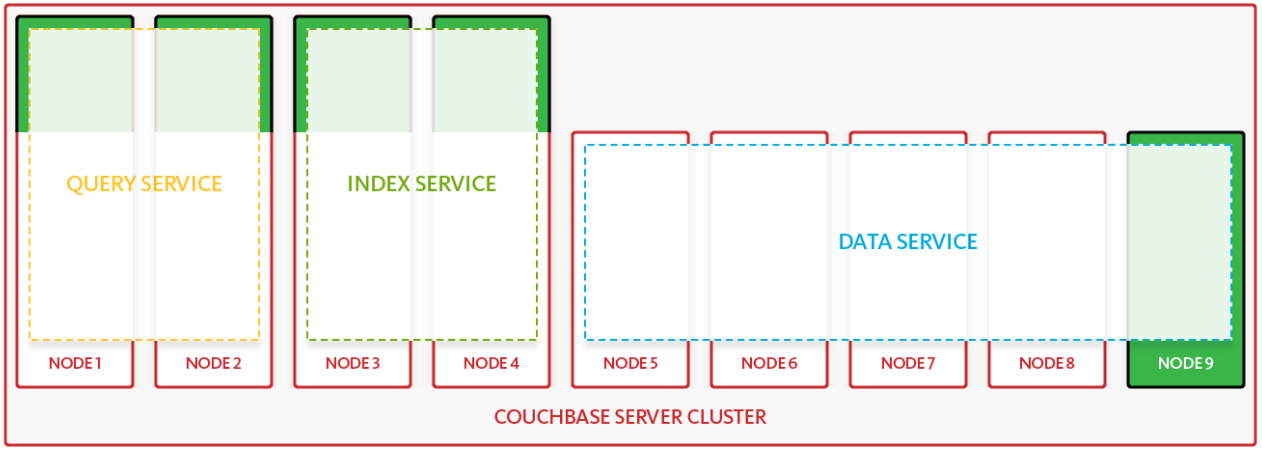
Independent Scalability for Best Computational Capacity — *per Service.*

Heavier Indexing?

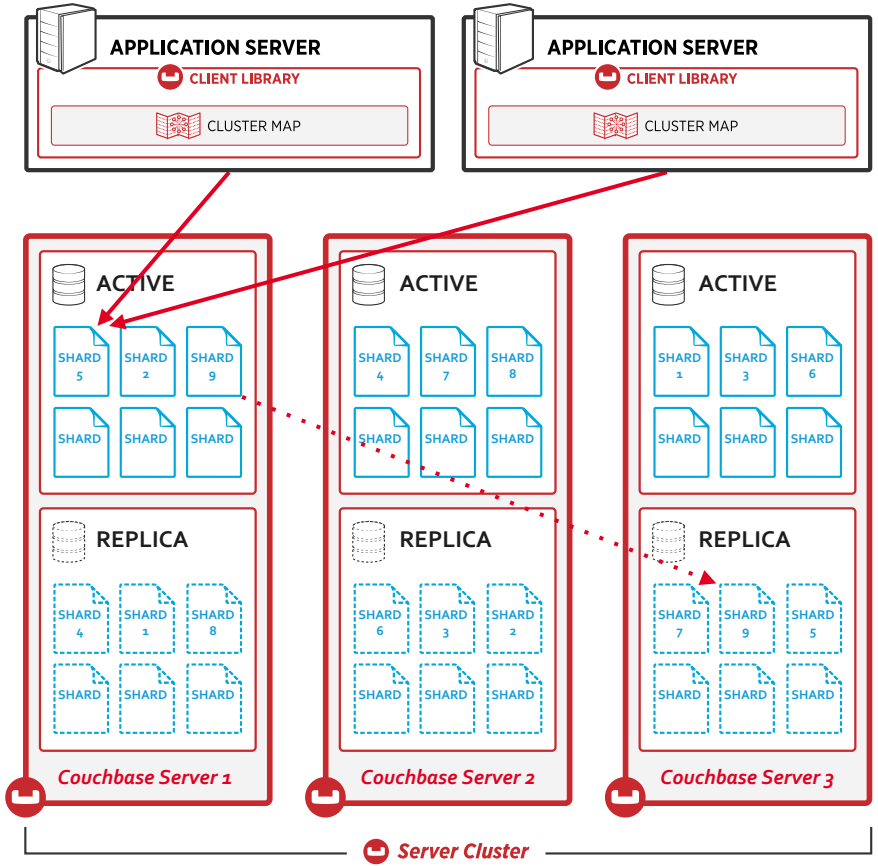
Scale up or out
Index Service Nodes.

More RAM for Query Processing?

Scale up or out
Query Service Nodes.



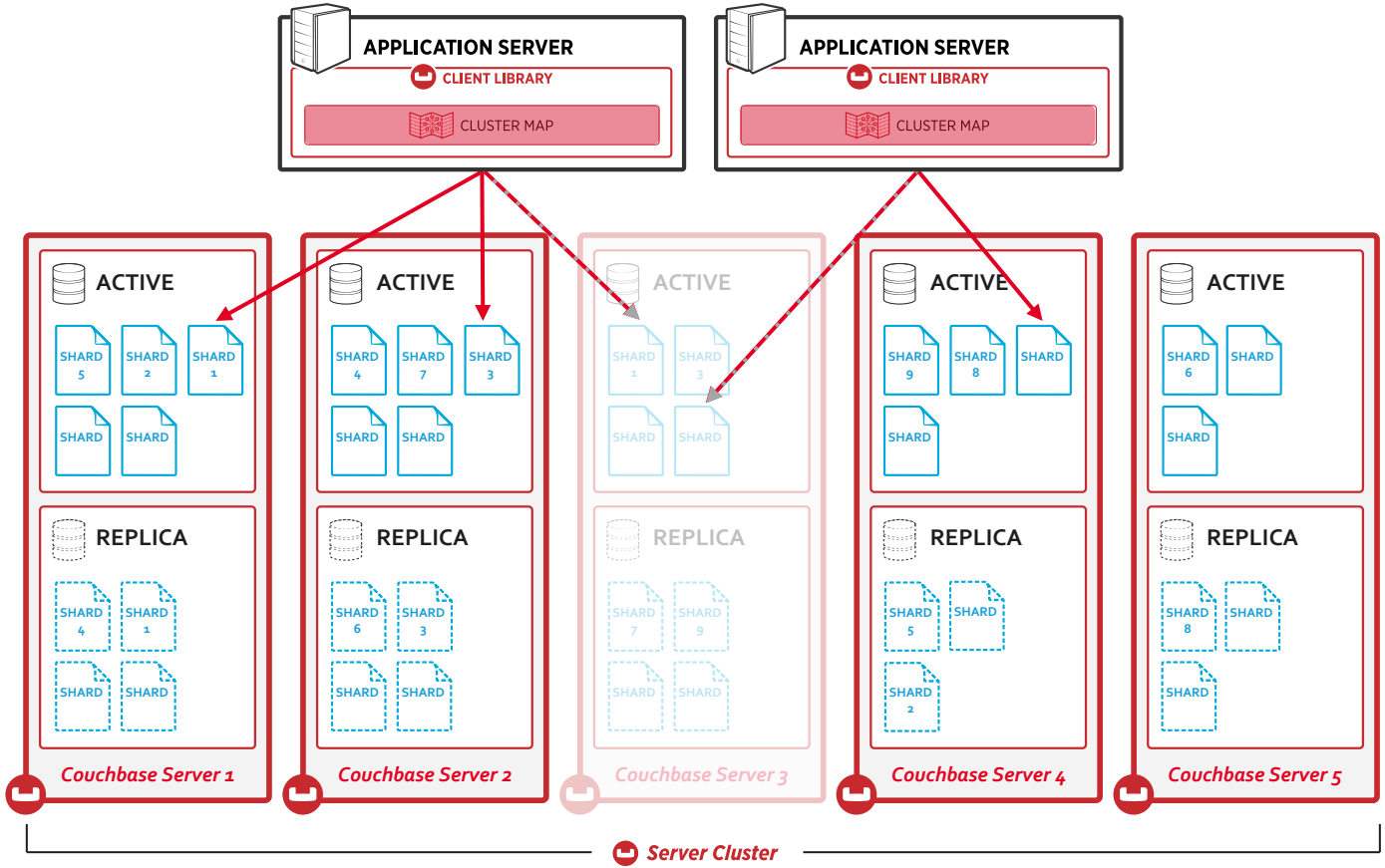
Data Service - Overview



Application has single logical connection to cluster (client object)

- Data is automatically sharded resulting in even document data distribution across cluster
- Each vbucket replicated 1, 2 or 3 times ("peer-to-peer" replication)
- Docs are automatically hashed by the client to a shard'
- Cluster map provides location of which server a shard is on
- Every read/write/update/delete goes to same node for a given key
- Strongly consistent data access ("read your own writes")
- A single Couchbase node can achieve 100k's ops/sec so no need to scale reads

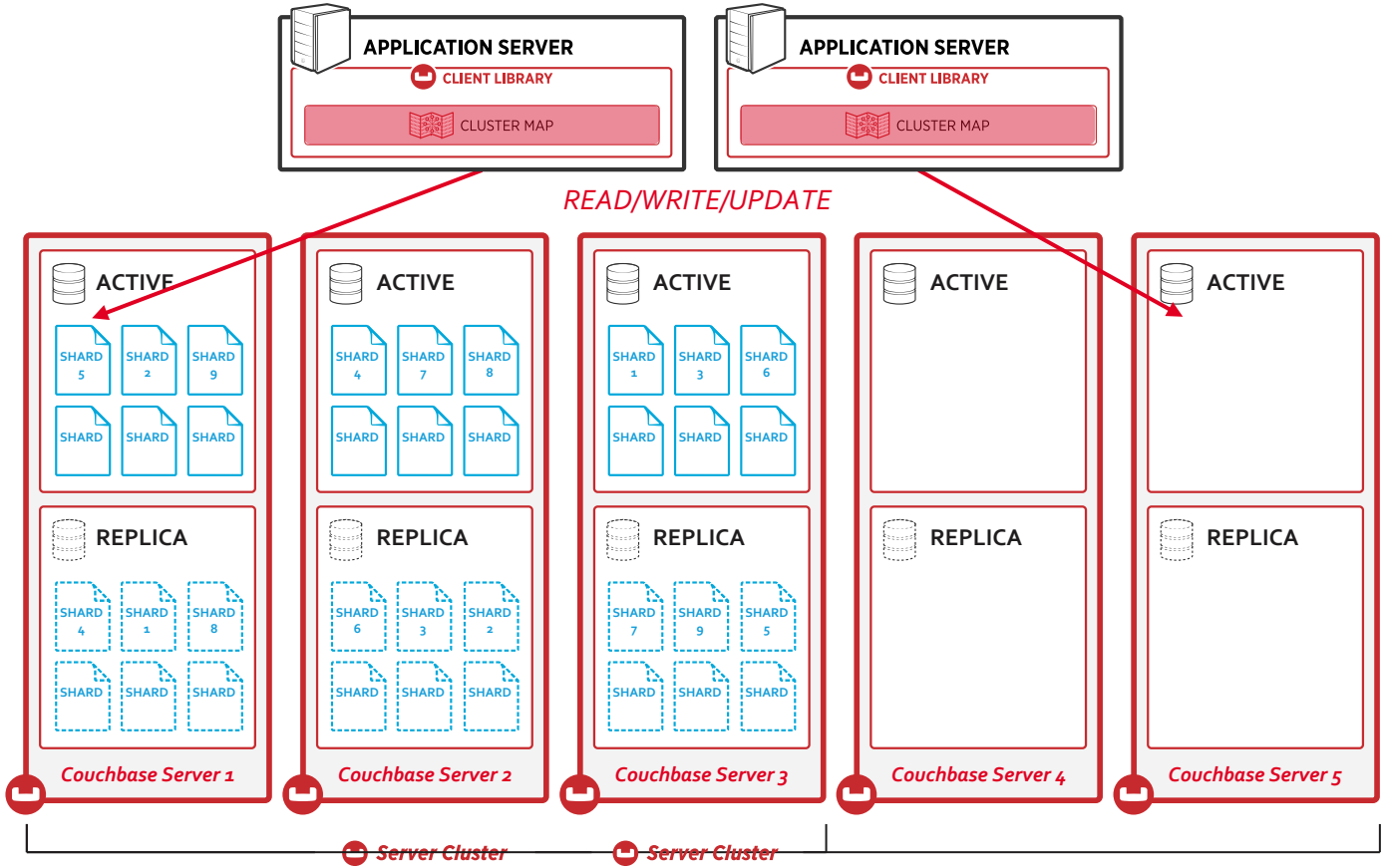
Data Service - Failover



Application has single logical connection to cluster (client object)

- When a node goes down, some requests will fail
- Failover is either automatic or manual
- Client library is automatically updated via cluster map
- Replicas not recreated to preserve stability
- Best practice to replace node and rebalance

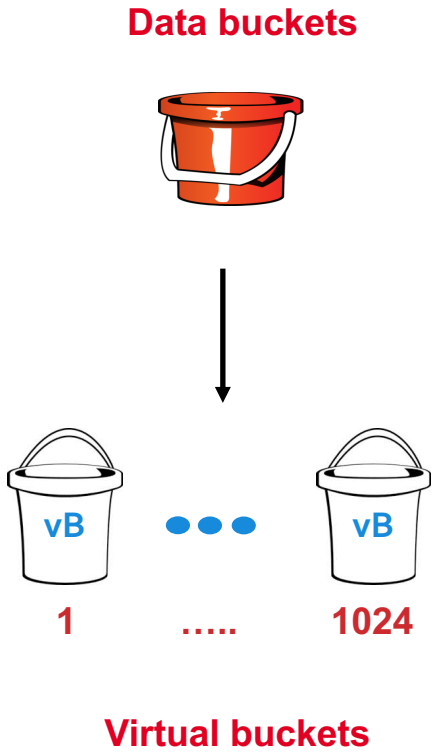
Data Service - Rebalance



Add/Remove of nodes requires a resharding of the data

- Multiple nodes added or removed at once
- One-click operation
- Incremental movement of active and replica vbuckets and data
- Client library updated via cluster map
- Fully online operation, no downtime or loss of performance

Data Service - Auto sharding

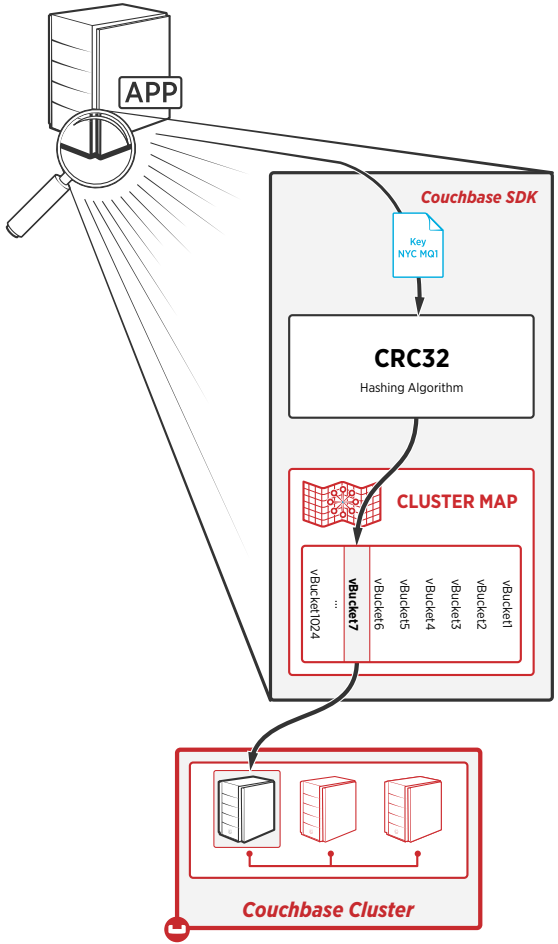


- A bucket is a logical, unique key space
- Multiple buckets can exist within a single cluster of nodes

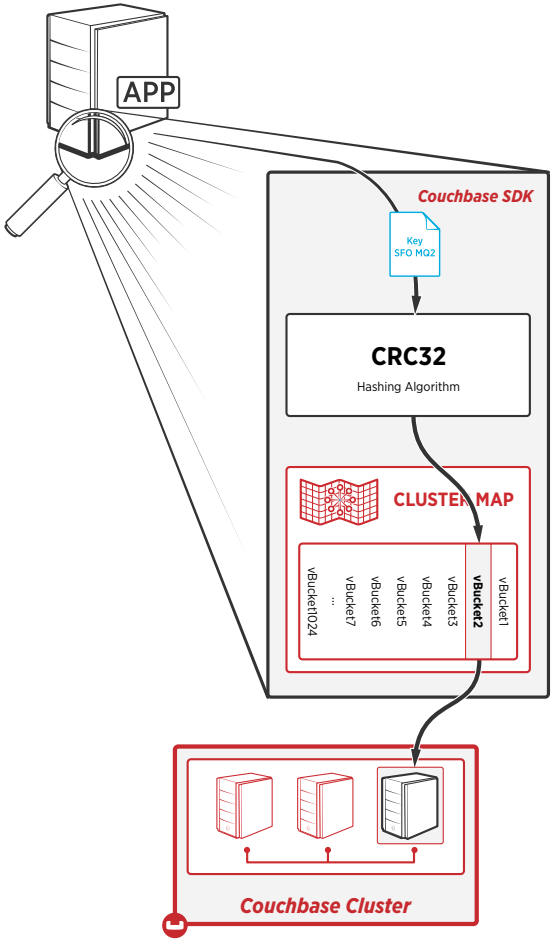
- Each bucket has active and replica data sets (1, 2 or 3 **extra** copies)
- Each data set has 1024 Virtual Buckets (vBuckets)
- Each vBucket contains 1/1024th portion of the data set
- vBuckets do not have a fixed physical server location

- Mapping between the vBuckets and physical servers is called the cluster map
- Document IDs (keys) always get hashed to the same vbucket
- Couchbase SDK's lookup the vbucket -> server mapping

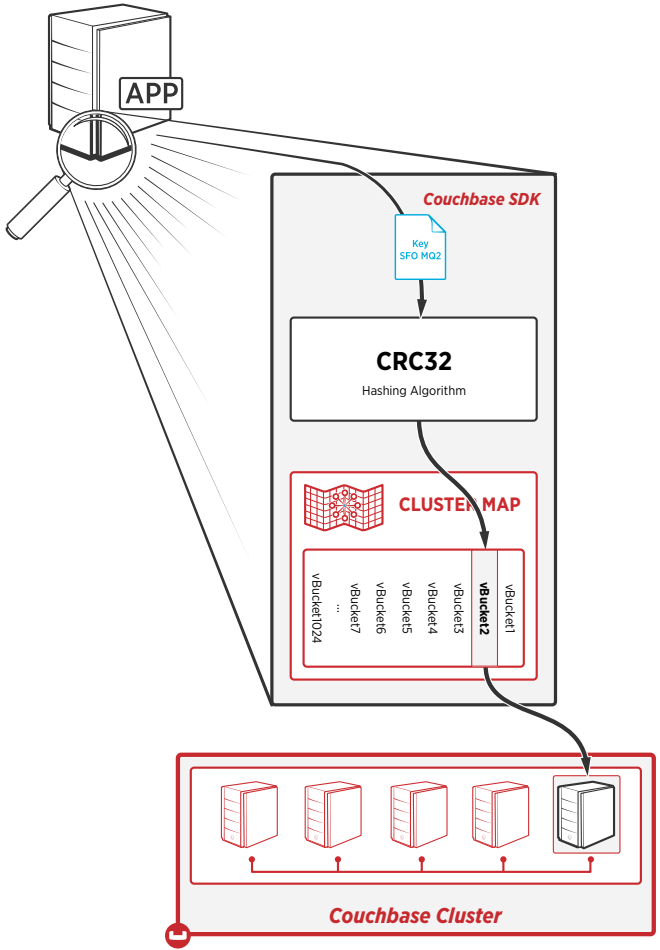
Data Service - Auto sharding



Data Service - Auto sharding



Data Service - Auto sharding, 2 nodes added



Installation and Configuration

Exercise 1

Installation

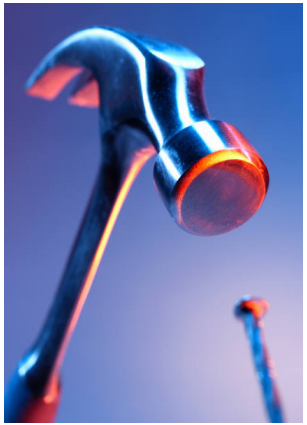
Perform the following steps in order install Couchbase Server on CentOS 6.x

- Disable Swappiness

```
## Run as root
# This defaults to 60
cat /proc/sys/vm/swappiness

# This disables it for the running system
echo 0 > /proc/sys/vm/swappiness

# This disables it permanently
echo '' >> /etc/sysctl.conf
echo '#Set swappiness to 0 to avoid swapping' >> /etc/sysctl.conf
echo 'vm.swappiness = 0' >> /etc/sysctl.conf
reboot
```



Installation

- Disable the Linux Firewall
  - May be configured in a production environment regarding http://developer.couchbase.com/documentation/server/current/install/install-ports.html

RHEL6

```
## Run as root or sudo
# Check the state
service iptables status
chkconfig --list iptables

# Stop it
service iptables stop

# Disable it
chkconfig --level 123456 iptables off

# Check the status again
chkconfig --list iptables
service iptables status
```

RHEL7

```
## Run as root or sudo
# Check the state
systemctl status firewalld

# Stop it
systemctl stop firewalld

# Disable it
systemctl disable firewalld
```

Installation

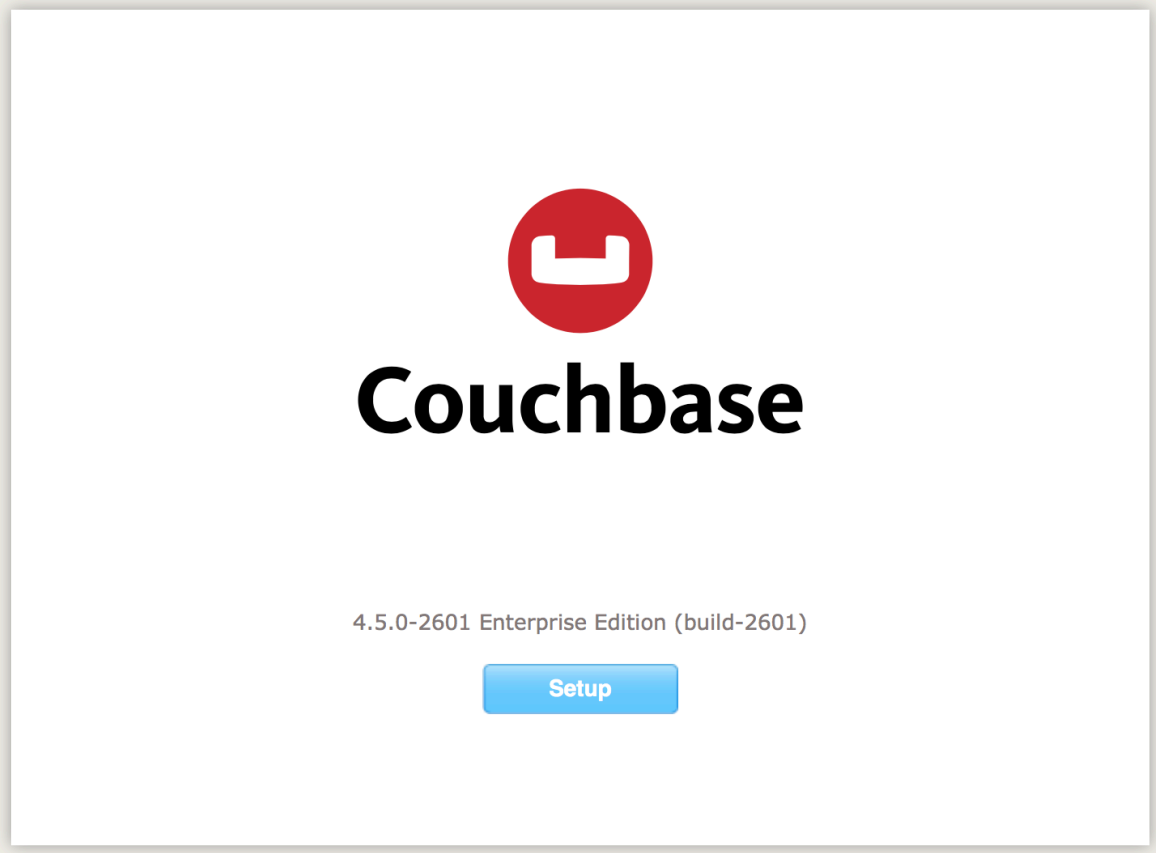
- Download the installation package from http://www.couchbase.com/nosql-databases/downloads
- You may wish to download the .rpm to your local machine and then 'scp' the file to the VMs.
- Perform the installation by using RPM

```
sudo rpm --install ${downloaded package}.rpm
```

- Open the Web UI Wizard

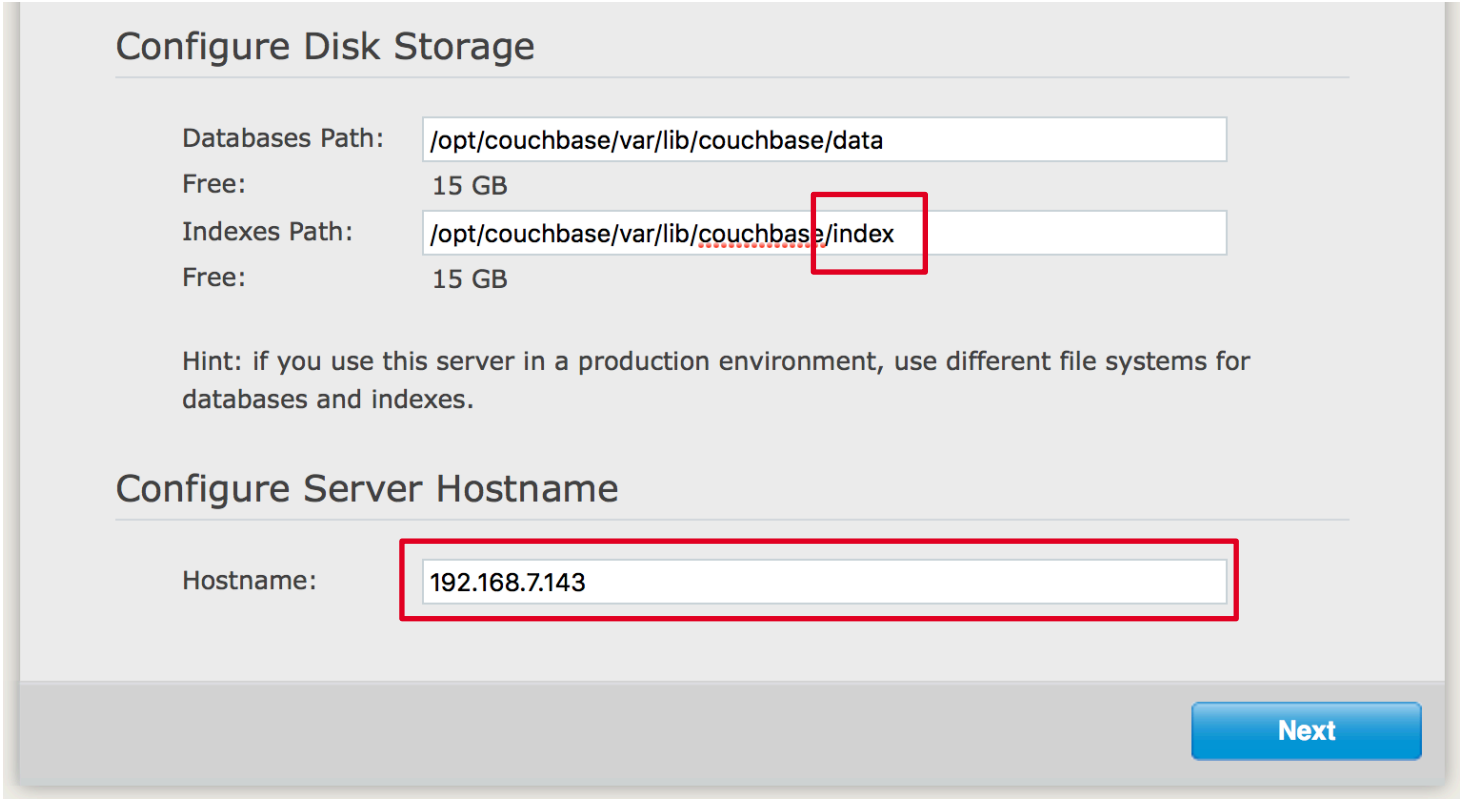
```
http://<public hostname of your VM>:8091
```

Installation



Installation

Installation



Installation

- Perform further steps in the Wizard
  - Select the sample buckets
  - Configure the default bucket by assigning 128MB RAM (per node) and by disabling the Replicas
  - Agree to the T&C
  - Create the Admin user

- Start and stop the Couchbase Server

RHEL6

```
sudo /etc/init.d/couchbase-server stop
sudo /etc/init.d/couchbase-server start

#Wait by double checking the state via the 'status' command
sudo /etc/init.d/couchbase-server status
```

RHEL7

```
sudo systemctl stop couchbase-server
sudo systemctl start couchbase-server

#Wait by double checking the state via the 'status' command
sudo systemctl status couchbase-server
```

Installation

- Set up the command line environment by editing $HOME/.bash_profile

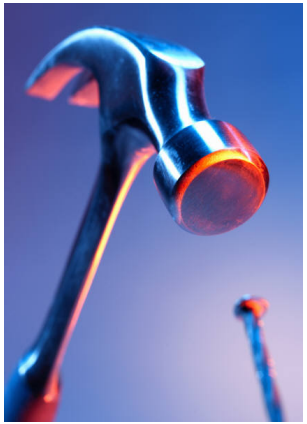
```
PATH=$PATH:$HOME/bin:/opt/couchbase/bin
```

Testing the Installation

Exercise 2

Test the Installation

Perform the following steps in order to test your installation



- List the nodes of your current cluster

```
couchbase-cli server-list --cluster=${host name or ip}:8091 \
--user=${admin user} -- password=${password}
```

- Investigate the data and index directory

```
sudo ls -al /opt/couchbase/var/lib/couchbase/index/@indexes/...
sudo ls -al /opt/couchbase/var/lib/couchbase/index/@2i/...

# You should see approximately 1030 files in this directory. So one file per vBucket + some extra files.
sudo ls -al /opt/couchbase/var/lib/couchbase/data/travel-sample
```

Test the Installation

- Get some data from a vBucket file

```
couch_dbdump /opt/couchbase/var/lib/couchbase/data/travel-sample/0.couch.1
```

- Get some info about a vBucket file

```
couch_dbinfo /opt/couchbase/var/lib/couchbase/data/travel-sample/0.couch.1
```

- Install Telnet

```
sudo yum install telnet
```

Test the Installation

- Connect via Telnet

```
telnet ${host} 11211
```

- Retrieve some statistics of the default bucket

```
stats
```

- Set a value (set $key $flags $exptime $numbytes $value)

```
set test_key 0 300 4
<Enter>
data
```

Test the Installation

- Get a value

```
get test_key
```

- Install Curl

```
sudo yum install curl
```

- Get details about the cluster via the REST API

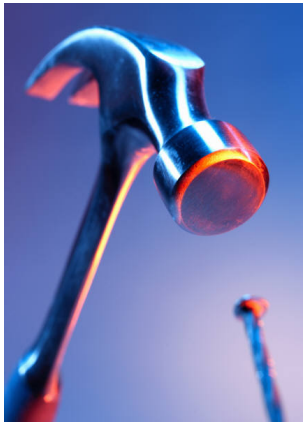
```
curl -u ${admin user}:${password} http://${host}:8091/pools/default/buckets/
```

Working with Buckets

Exercise 3

Create a Bucket

Perform the following steps in order to create a bucket

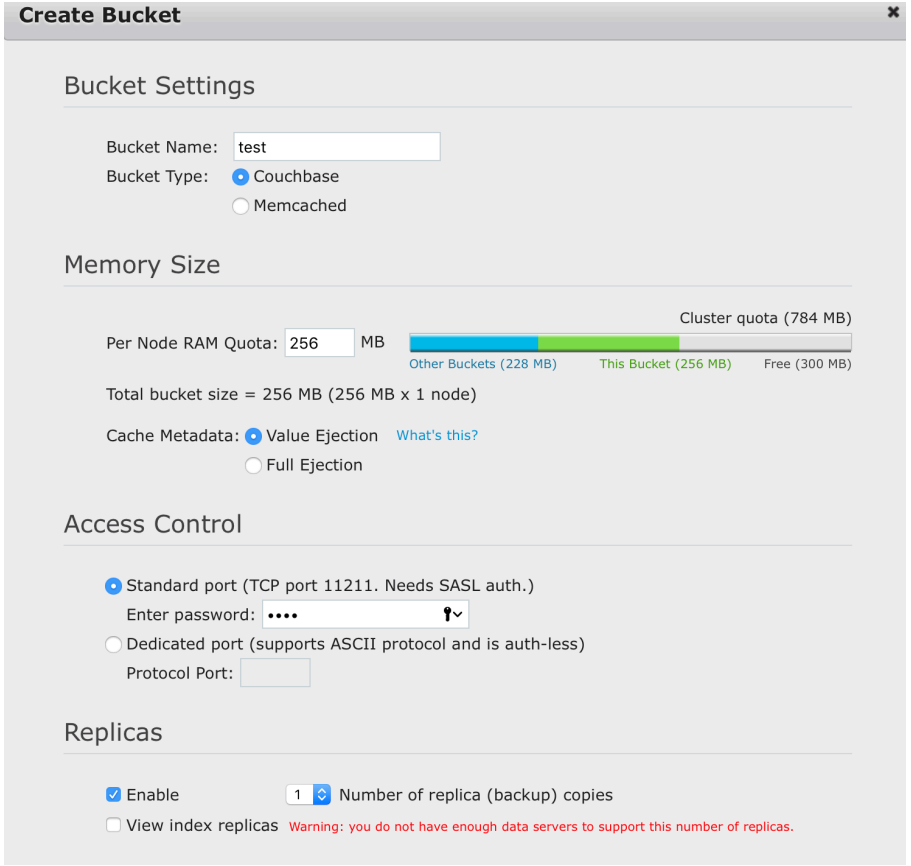


- Open the Web Admin UI and go to the 'Data Buckets' tab

  http://<public hostname of your VM>:8091

- Create a new bucket with the name 'test'
  - Allocate 256 MB RAM to it
  - Use the password 'test'
  - Use one Replica
  - Keep the other default settings

Create a Bucket



Create a Bucket

- List the buckets by using the CLI

```
couchbase-cli bucket-list -c ${host}:8091 -u ${admin} -p ${password}
```

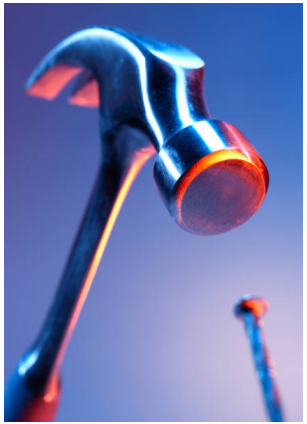
- Create a second test bucket 'test2' by using the CLI

```
couchbase-cli bucket-create -c ${host}:8091 -u ${admin} -p ${password}\
    --bucket=test2\
    --bucket-type=couchbase\
    --bucket-port=11211\
    --bucket-ramsize=256\
    --bucket-replica=1
```

Working with the Cluster

Exercise 4

Add/remove a node



Perform the following steps in order to add/remove a node To/from the cluster

- Stop the local Couchbase instance again
- Start all 3 Docker containers

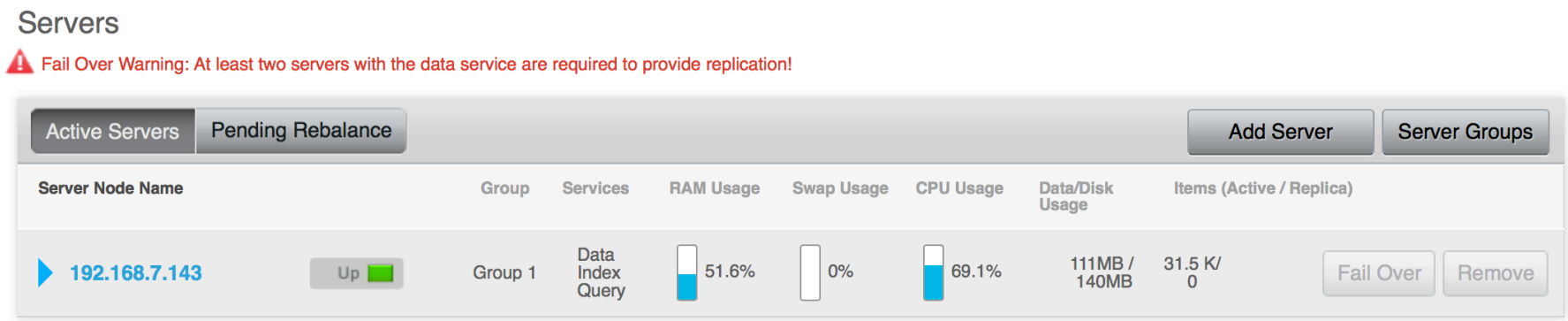
```
docker run -d --name couchbase-1 -p 8091-8094:8091-8094\
-p 11210-11211:11210-11211 couchbase
docker run -d --name couchbase-2 couchbase
docker run -d --name couchbase-3 couchbase
```

- Open the Web Admin UI and set up the first node and then add a the second node

```
http://<public hostname of your VM>:8091
```

- Don't forget to rebalance!

Add/remove a node



Add/remove a node

- On the 3rd node execute the following command

(you can log-in by using docker exec -it couchbase-3 /bin/bash)

```
couchbase-cli server-add --server-add=${node 3 name/ip}\
--server-add-username=${node3 admin} --server-add-password=${node3 pwd}\
--group-name="Group 1" --cluster=${node1 name/ip}:8091\
--user=${node1 admin} --password=${node1 pwd}
```

- Don't forget to rebalance!
  - Perform the Rebalance again via the UI
  - BTW: The CLI command 'couchbase-cli rebalance' can be used to invoke it from the command line
  - Which Service role was enabled on the 3rd node?

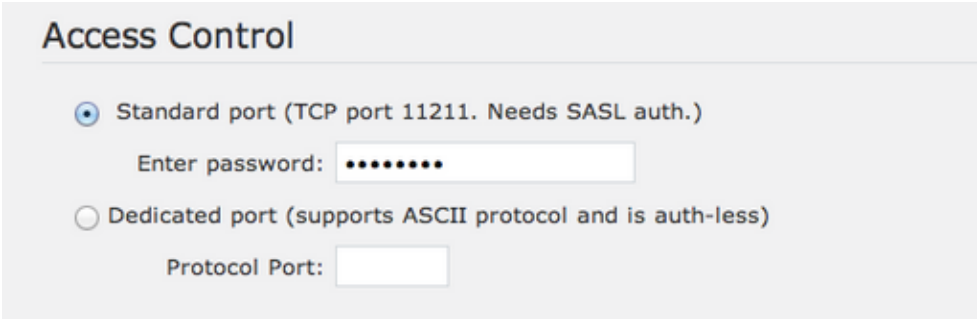
- Remove a cluster node by executing the following command

```
couchbase-cli rebalance -c ${another name/ip}:8091 --server-remove=${to remove name/ip} -
-user=${admin user} --password=${password}
```

Security

Couchbase Authentication

- **Application authentication**
  - Buckets are protected with challenge-response SASL protocol
  - AuthN happens over CRAM-MD5

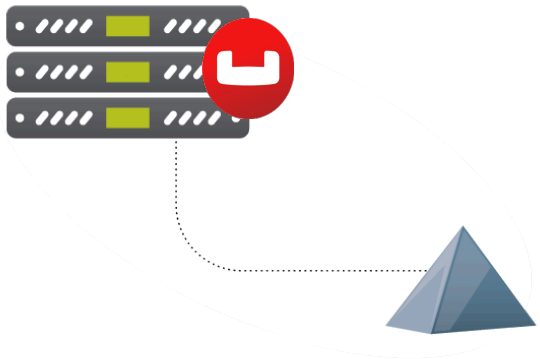


- **Admin authentication**
  - Authentication through admin username and password
  - Authentication through LDAP

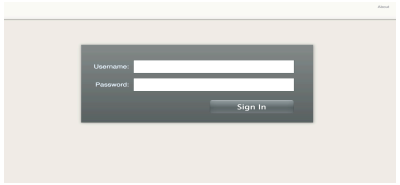
External identity management using LDAP

- **Centralized identity management**
  - Define multiple admins
  - Centralized security policy management for admin accounts for stronger passwords, password rotation, and auto lockouts

- **Individual accountability. Simplified compliance.**
  - Define UIDs in LDAP, and map UIDs to admin roles in Couchbase
  - Comprehensive audit trails with LDAP UIDs in audit records



LDAP architecture in Couchbase



Admin
UID / password

Authentication SUCCESS!

YES

YES

Authentication FAILED!

NO

CHECK IN ADMIN PASSWORD FILE

SASL protocol

UID / password

SASLAUTHD

CHECK IN LDAP ?

OpenLDAP protocol

YES / NO?

UIDs defined in LDAP

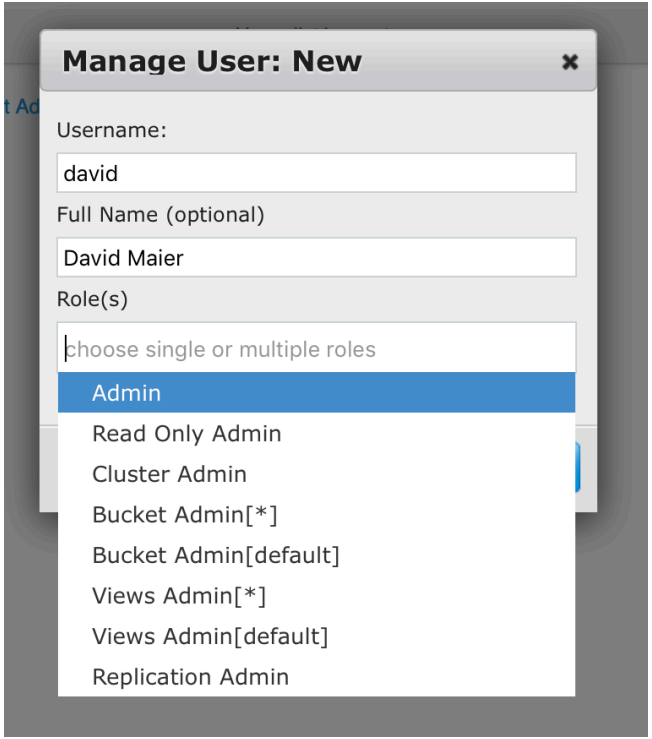
OpenLDAP

Windows Server Active Directory

saslauthd config file

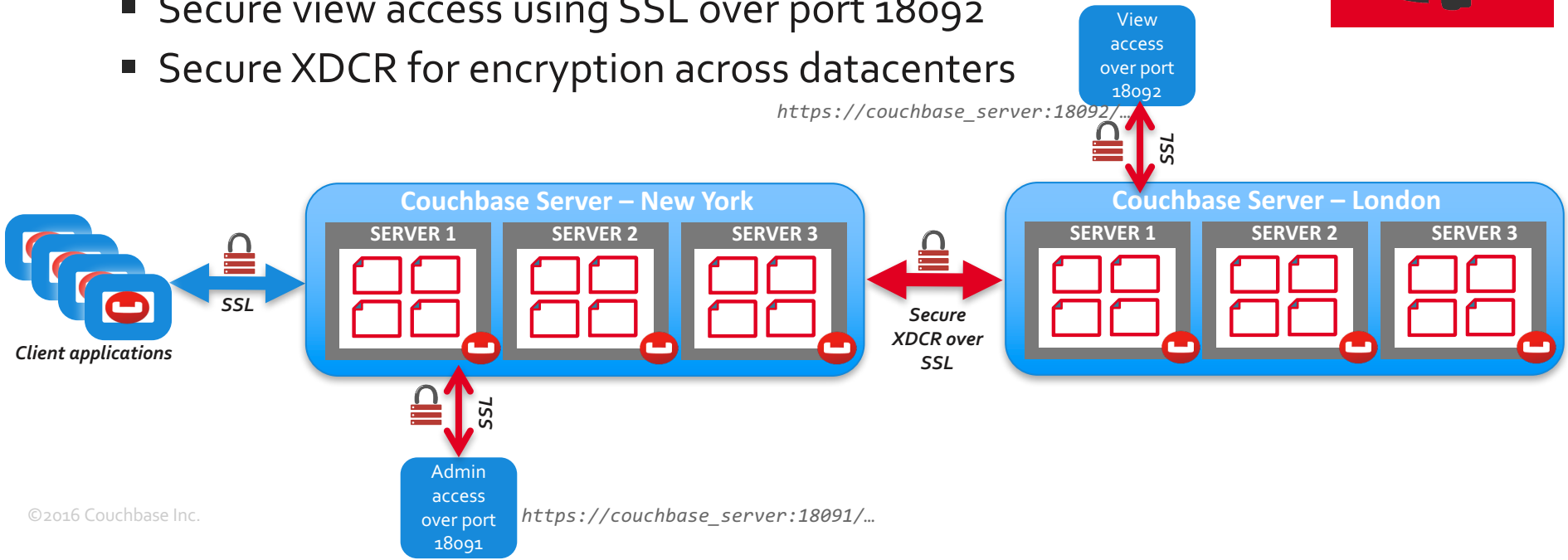
Couchbase authorization

- **Application data access**
  - Full access to specific buckets
- **Admin access**
  - Several roles



Couchbase encryption – in motion

- **Data-in-motion encryption**
  - Client-server communication should be encrypted using SSL
  - Secure admin access using SSL over port 18091
  - Secure view access using SSL over port 18092
  - Secure XDCR for encryption across datacenters



Auditing events

LIST OF ADMIN AUDIT EVENTS
Success/failure login for administrator
Audit configuration changes
Enable/disable audit
Add a node to the cluster
Remove a node from the cluster
Failover a node
Rebalance the cluster
Shutdown/startup of the system by the administrator
Create a bucket
Delete a bucket
Flush a bucket
Modify bucket settings
Change configured disk and index path
Add read-only administrator user
Backup

Remove read-only administrator user
Add admin user
Remove admin user
Setup remote cluster reference
Delete remote cluster reference
Changes to XDCR
Creating/deleting XDCR profile
Pause resume XDCR stream
Changing XDCR filter rules
Add/remove query node
Add/remove index node
Create server group
Add node to server group
Remove node from server group
Delete server group
Admin password changes/resets

Auditing a successful login


```
{
 "timestamp":"2015-02-20T08:48:49.408-08:00",
 "id":8192,
  "name":"login success",
  "description":"Successful login to couchbase cluster",
  "role":"admin",
  "real_userid": {
                   "source":"ns_serv
                   "user":"bjones"
                 },
  "sessionid":"ofd0b5305d1561ca2b10f9d795819b2e
  "remote":{"ip":"172.23.107.165", "port":59383}
}
```


WHEN

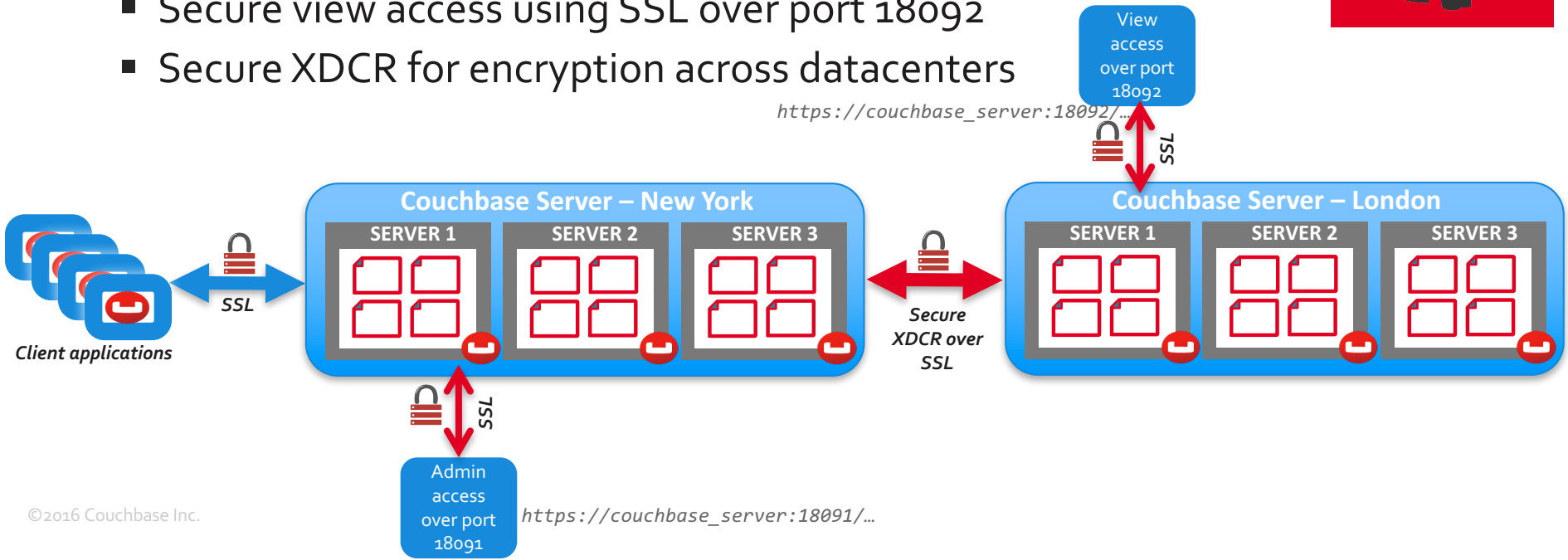
WHAT

WHO

HOW

Couchbase encryption – in motion

- **Data-in-motion encryption**
  - Client-server communication should be encrypted using SSL
  - Secure admin access using SSL over port 18091
  - Secure view access using SSL over port 18092
  - Secure XDCR for encryption across datacenters



Certificate mangement

- **Self-signed or custom (CA-signed) certificates**

Self-Signed Cluster Certificate

The cluster is currently using a **self-signed** SSL/TLS certificate.

```
-----BEGIN CERTIFICATE-----
MIIC/jCCAeigAwIBAgIIFA+g2w6AQAYwCwYJKoZIhvcNAQELMCQxIjAgBgNVBAMT
GUNvdWNoYmFzZSBTZXJ2ZXIgZGJiMmMxYjIwHhcNMTMwMTAxMDAwMDAwWhcNNDkx
MjMxMjM1OTU5WjAkMSIwIAYDVQQDExlDb3VjaGJhc2UgU2VydmVyIGRiYjJjMWIy
MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEArOQbrnZJeJTWyhtUdMyE
Djlk1NJNley5D06leSzS5JDV6Bzd4t3FL6JY3eVFVO/UGoc9qO+SsTo8oWuWhhZo
mN37ZoMC2lnXmHGvziiswT73zRkK3w1//grnYbECa8rLmB16TfTm7jHc9rN7mx+c
U5k6QPYAYam/Yh4ChrY7TVjQl+llT3xD+thQSQY6opb78EVcEenBVVsbaUJry30F
EnagvuLD+KGlcP5voX0uwXFogjsoLfmxVfRx7ZSGJMeCvQ9zxAFubwLkj8ziyZ9h
TEHFjn/lSJf/hvavo1qjTHf/zkgO9xXmP5i+ZeQWm7npWp3kiCD3T+54B5cwhT88
nwIDAQABozgwNjAOBgNVHQ8BAf8EBAMCAKQwEwYDVR0lBAwwCgYIKwYBBQUHAwEw
DwYDVR0TAQH/BAUwAwEB/zALBgkqhkiG9w0BAQsDggEBAG5GT4FwpwdAcp6SCd0L
ozsrPXcXQNUN2N6WvrHKXVGVU2tHDRZP9j4kfkRYX0ftgfu3AwJdfgoU3gCbHWxc
hQrSCVCTfaBTygY9at0fOhzJRxE4ycKrminJAU+WzlEqk2RVF5pr6mRagjSwwdls
+Ui8/axsYo2fyE5DRpmLErDC3vujmOc4JlWHoh5aBnVCE0sW+RHBw/pKKEDvtIFZ
```

Security Outlook

Product Features

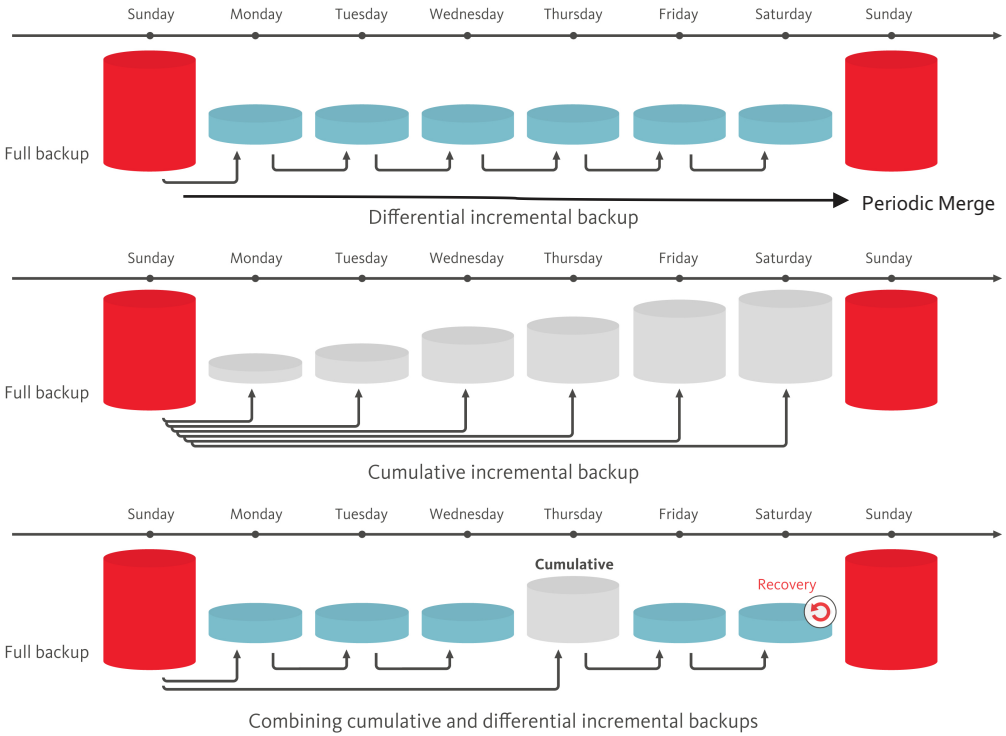
Rich admin security model	Rich application security model	Advanced Compliance
• RBAC for Admins • X.509 server CA certificates for SSL	• Users and RBAC for Applications • Application Auditing	• Kerberos • Native on-disk encryption

Backup and Restore

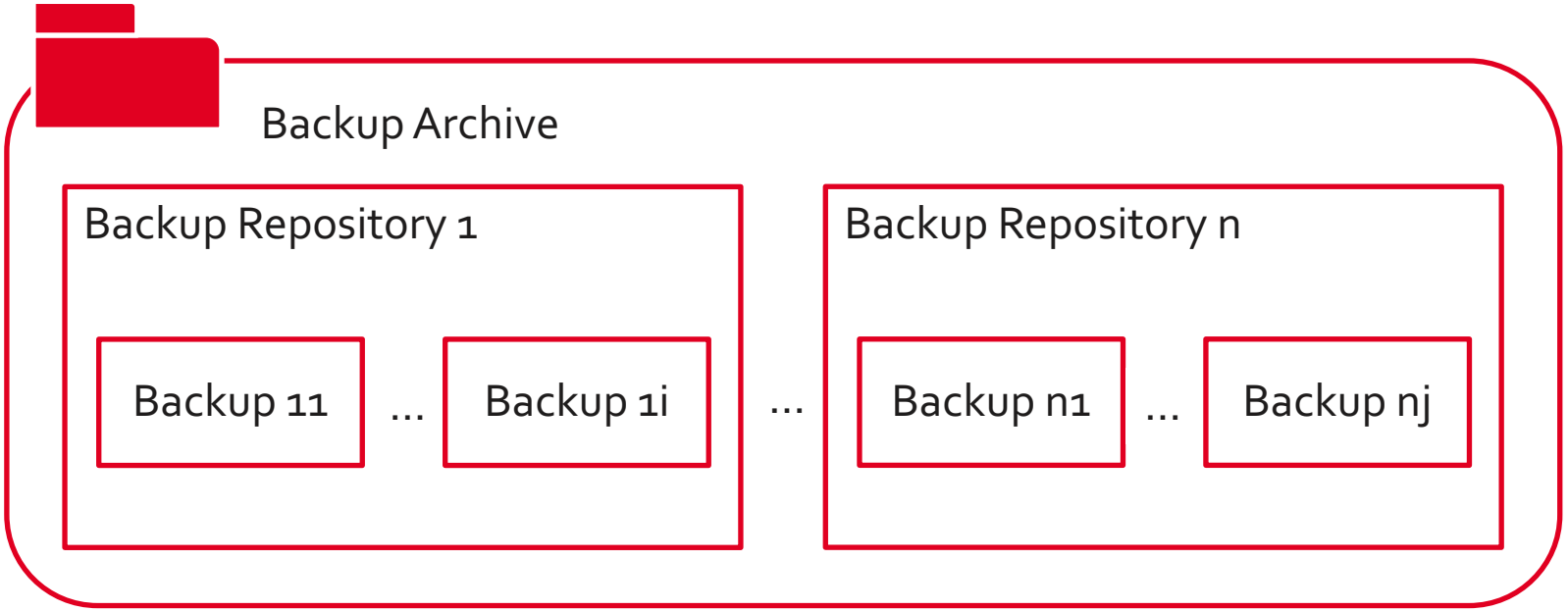
Exercise 5
(Optional)

Backup Strategies

- Full Backup
  - Full Snapshot
  - Per bucket or also per node
- Differential Incremental Backup
  - Only changes since the last full backup
  - Created quicker
  - Takes longer to restore
  - Periodic merge
- Cumulative Incremental Backup
  - All changes since the last full backup
  - Takes longer than differential backups
  - Uses more disk space than differential backups
  - Faster to restore than differential backups
- Combined

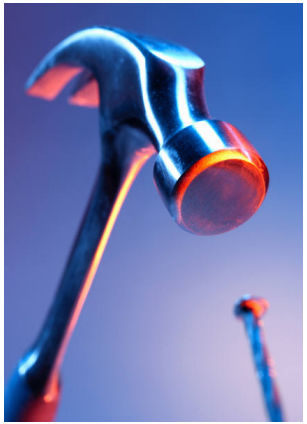


Backup Archive Structure



Backup and Restore

Perform the following steps in order to backup some data



- Create a target folder

```
cd /tmp
mkdir cb-backup
```

- Prepare the backup archive

```
/opt/couchbase/bin/cbbackupmgr config  --archive=/tmp/cb-backup\
 --repo=workshop
```

- Backup the data twice and then use the list command to list the increments!

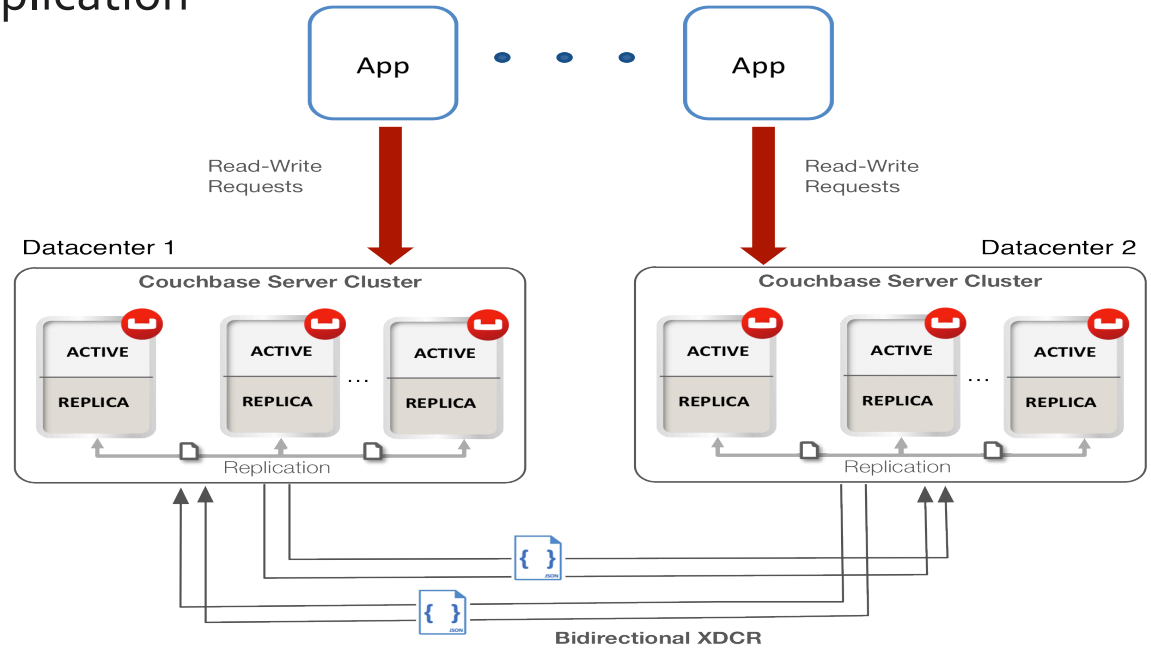
```
cbbackupmgr backup --archive=/tmp/cb-backup --repo=workshop\
--host=http://localhost:8091 --username=${admin}\
--password=${password}
```

XDCR explained

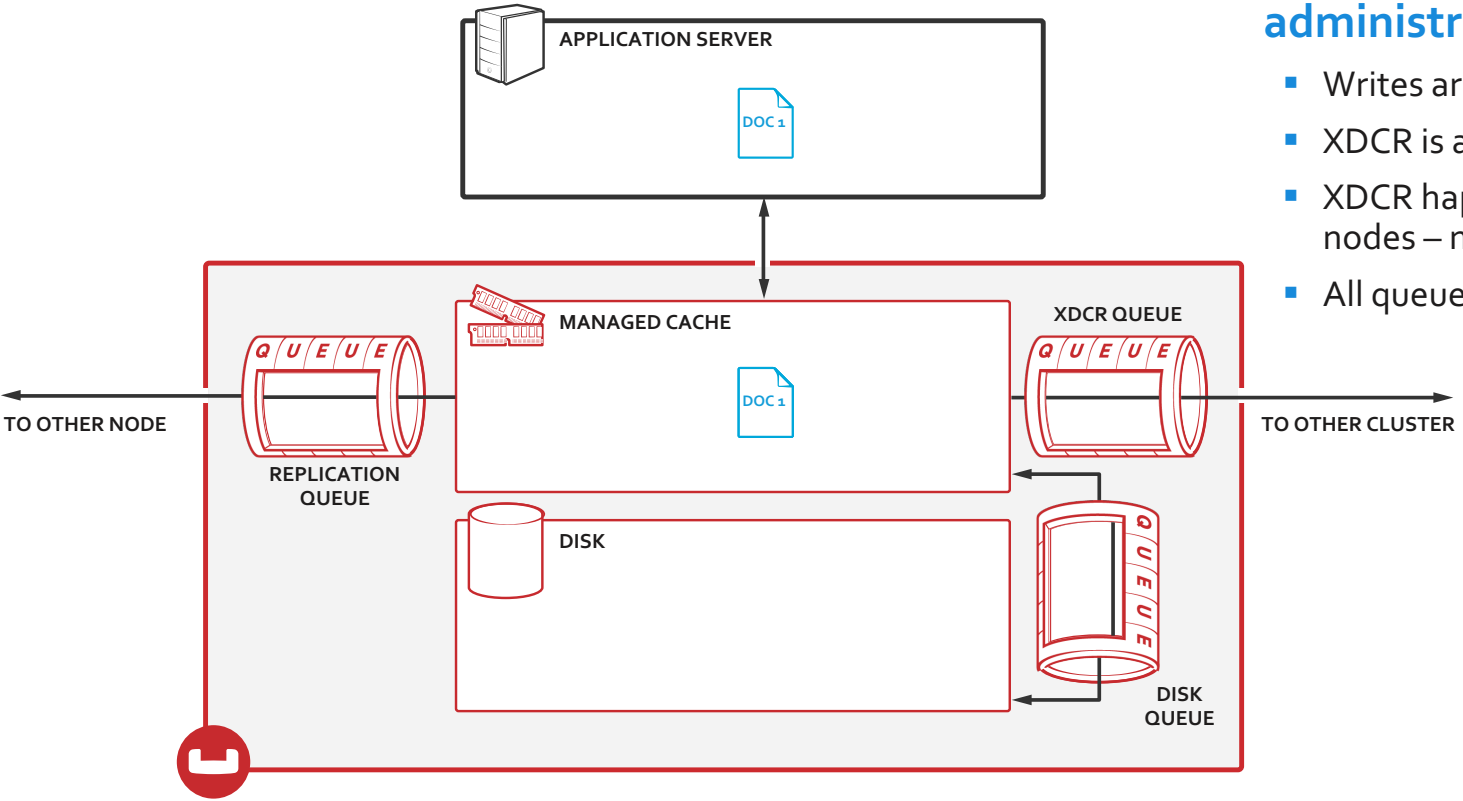
Exercise 6

XDCR: Cross Data Center Replication

- Application can access both clusters (master – master)
- Scales out linearly
- Different from intra-cluster replication ("CP" versus "AP")
- Filtered replication



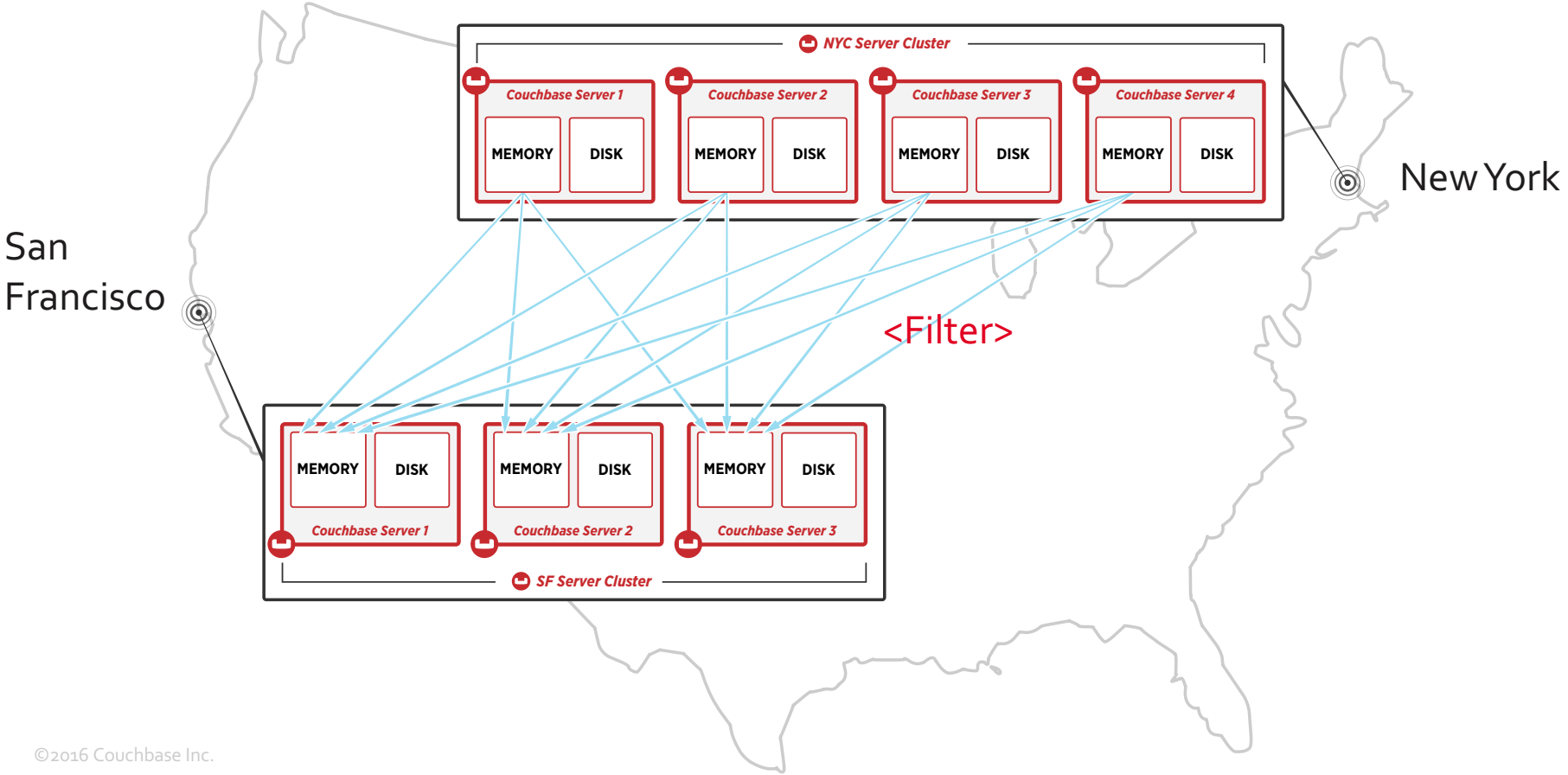
XDCR after Write



Single-node type means easier administration and scaling

- Writes are async by default
- XDCR is also RAM-based
- XDCR happens between individual nodes – no gateway
- All queues are processed in parallel

Market leading memory-to-memory replication



XDCR: Flexible topologies

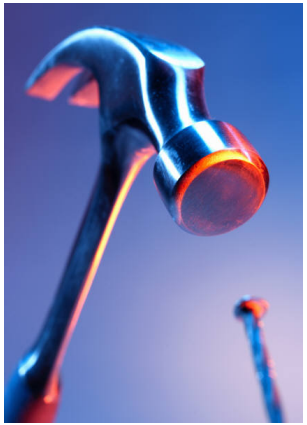
- One-one, one-many, many-one
- Differently sized and resourced clusters supported

XDCR

Perform the following steps in investigate the XDCR part of the Admin UI

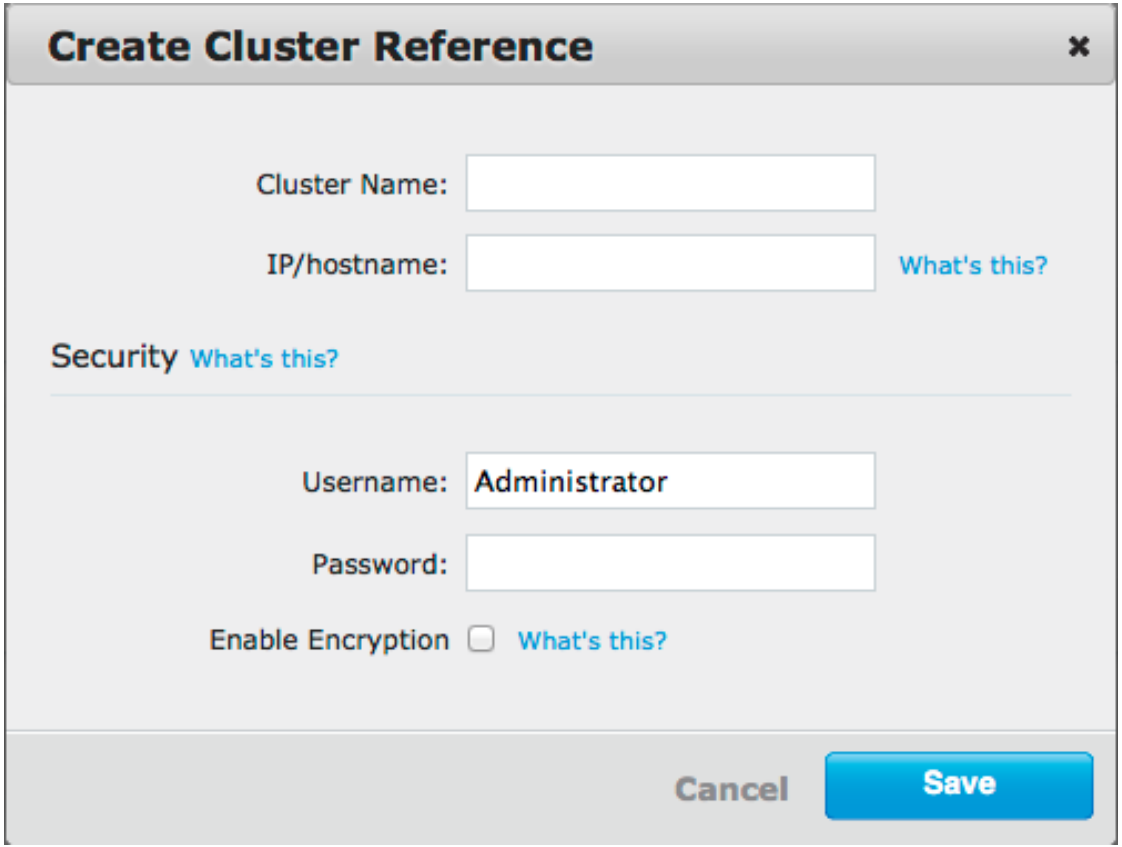
- Open the Web Admin UI and navigate to the XDCR tab

  http://<public hostname of your VM>:8091

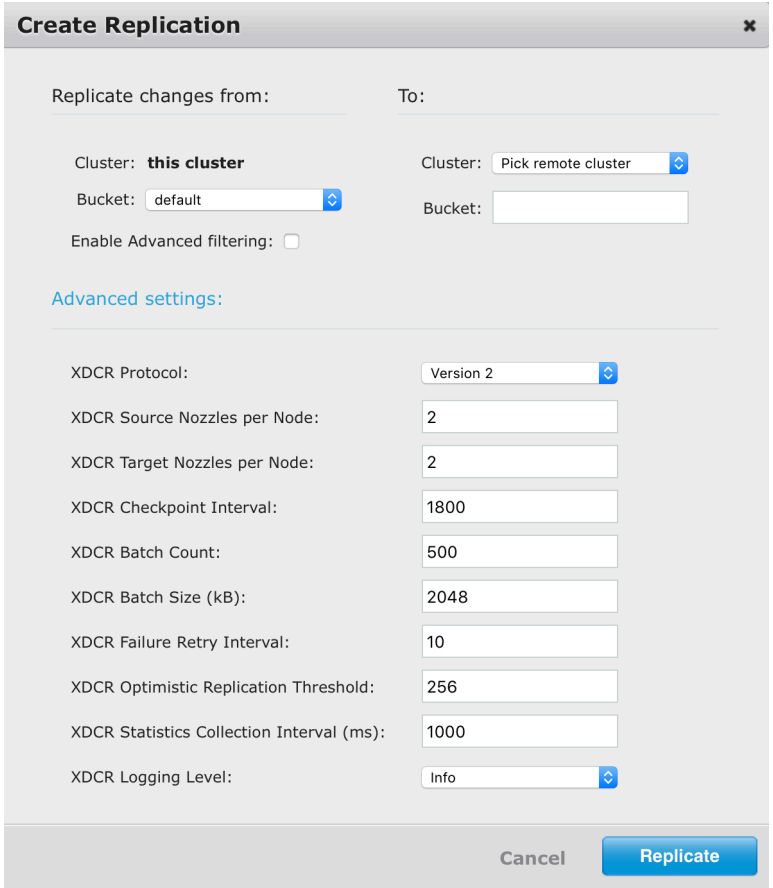


XDCR

XDCR



XDCR



Thank you

Q&A