Building Centralized Caching Infrastructure at Scale

James Won
Staff Site Reliability Engineer

June 14, 2019

@jwon_me
Question #1
Join at
slido.com
#CACHE
$ whoami
> jwon

- Joined LinkedIn in 2013
- Site Reliability Engineer for multiple teams, but most recently the Caching-as-a-Service team.
- Python Developer
- Vim user

@jwon_me
Question #2
Join at slido.com #CACHE
🚫 This is NOT...

a talk about **Couchbase**, but a talk about **techniques** that can be applied anywhere.

Though if you are interested in learning more about how we’re using Couchbase, check out our blog post: 
[https://engineering.linkedin.com/blog/2018/05/evolution-of-couchbase-at-linkedin](https://engineering.linkedin.com/blog/2018/05/evolution-of-couchbase-at-linkedin)

@jwon_me
Agenda

1. History of Caching @ LinkedIn
2. Offering Caching as a Service
3. Challenges/Takeaways
4. Future work
Teams were frustrated with operating Memcached

- Losing the cache when nodes died
- Resizing & Replacing hosts was difficult
- Cache copying was difficult
Enter Couchbase

- Drop-in replacement
- Fast
- Persistent
- Replicas
- Resizing
Couchbase adoption exploded at LinkedIn 🧵

- Creation of a virtual team/working group to share learnings
- Over 2000 hosts in production
- Over 300 unique clusters
Growing Too Fast?

- Lack of Operations Interest
- Custom Deployment
- Runaway Hardware Growth
Question #3
Join at slido.com
#CACHE
Agenda

1. History of Caching @ LinkedIn
2. Offering Caching as a Service
3. Challenges/Takeaways
4. Future work
Provide Caching as a Service (CaaS)

Centralize caching infrastructure to a team that is passionate about caching and wants to solve caching at scale
Couchbase Terminology

User/application data

Logical key spaces

Dynamically scalable

Source: https://docs.couchbase.com/server/4.1/concepts/buckets-vbuckets.html
Caching as a Service

Build & Manage @ scale  Improve hardware efficiency  Improve security

@jwon_me
CaaS Mission

“Provide secure, high performance, and cost effective caching to all teams within LinkedIn”
Question #4
Join at slido.com #CACHE
Question #5
Join at slido.com #CACHE
What We Offer

- 0-1ms 95Pct latency for GETs/SETs
- 10ms SLO
- New!
  - Replication
  - Indexing
What We Offer

- SSD clusters, HDD clusters, Pure Memory clusters
- Informational dashboards for free
- Alerts on health and availability of cluster
- Handle hardware failures (host/switch/etc)
- Software & OS upgrades
Informational dashboard auto-generated
What We DON’T Offer

- Ownership of your data
- Backups
How much are we caching?

- >200 unique use cases in PROD of varying sizes
- >2000 hosts
- >10M qps across multiple clusters
Agenda

1. History of Caching @ LinkedIn
2. Offering Caching as a Service
3. Challenges/Takeaways
4. Future work
Question #6
Join at slido.com
#CACHE
Challenge #1

GDPR

• Needed to ensure GDPR compliance for over 200 unique caching use cases

• Win: The creation of a dedicated caching team allowed us to drive this initiative forward

• Most use cases did a data migration to our managed platform; built tooling around sizing and actual migration
Challenge #2
Reconceptualize deployments

The old way

1. Hand-edit configuration files
2. Wait for them to sync (30m – 1h).
3. Run non-standard Salt runner to build cluster and hope it works.
4. If you made a mistake, go to step 1.
Challenge #2

Reconceptualize deployments

Solution

• Create wrapper around Couchbase Server to use supported standard deployment infrastructure

• Allow us to use standard deployment tooling for actions like upgrades, cluster expansions, etc.
Challenge #3
Getting off running as the root user

Why didn’t we just run as non-root from the beginning?

- Couchbase didn’t officially support it
- Small team, tight on resources. Ship something quickly as first iteration.
- RPM-based deployments was the path of least resistance to get started. Start with vendor-supported path.
Challenge #3

Getting off running as the root user

Changing an engine of a car going 100 MPH

- Did in-place conversions to "convert" a root node to a non-root node.
- Lot of code regarding file permissions as well as symlinking
- Switched to tarball-based deployments
- Removed conversion code and root code; removed almost 1000 lines of code!
Takeaways
Treat your clusters as cattle 🐮, not pets 🐱

This affects the way you think about automation and scale
Start with a core offering and iterate

Intentionally started with a simple KV offering.

Expanded feature set (replication, indexing) as business needs grew.
Codify checklists into automation

Runbooks/checklists are OK at first. Once process is documented, strive towards automation.
Adding a bucket manually

You have no data buckets. Go to Buckets to add one, or load a sample bucket with data & indexes.
Bad Design Decision

Bucket configuration in configs

- Originally put bucket configuration in application configs

```xml
<entry key="tscp-tracking-fast-dedupe">
    <map>
        <entry key="bucket-ramsize" value="1000"/>
        <entry key="bucket-replica" value="1"/>
        <entry key="bucket-type" value="ephemeral"/>
        <entry key="eviction-policy" value="nruEviction"/>
        <entry key="authorized-users" value="tscp-tracking"/>
    </map>
</entry>
```
Bad Design Decision

Bucket configuration in configs

- Need to deploy configs every time we needed to add/remove buckets
- Lot of unnecessary repetition -- change only needed to be applied on one host of the cluster
Add bucket using caas-tools

<table>
<thead>
<tr>
<th>Cluster</th>
<th>Fabric</th>
<th>Bucket</th>
<th>Type</th>
<th>Replicas</th>
<th>Quota(MB)</th>
<th>QuotaUsed(%)</th>
<th>Purge Interval(h)</th>
<th>MaxTTL(s)</th>
<th>Item</th>
</tr>
</thead>
<tbody>
<tr>
<td>in-couchbase.Test-HDD-4</td>
<td>test_bucket_add</td>
<td>membase</td>
<td>1</td>
<td>2000</td>
<td>1</td>
<td>default</td>
<td>2592000</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>in-couchbase.Test-HDD-4</td>
<td>test_bucket_add</td>
<td>membase</td>
<td>1</td>
<td>2000</td>
<td>1</td>
<td>default</td>
<td>2592000</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>in-couchbase.Test-HDD-4</td>
<td>test_xdcr</td>
<td>membase</td>
<td>1</td>
<td>1000</td>
<td>1</td>
<td>default</td>
<td>2592000</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>
Build platforms, not tools

Rather than building tools for specific issues, build general APIs that can be pieced together to solve specific problems.
Trust your automation

It can be tempting to fall back to doing things the manual way when your automation/tooling fails. Don’t. Instead use it as an opportunity to figure out root cause and improve it!
1 History of Caching @ LinkedIn
2 Offering Caching as a Service
3 Challenges/Takeaways
4 Future work
Future work

- Self-Service provisioning
- Transparent migration of buckets across clusters
- Ease of shuffling nodes across clusters
- Better resource/hardware utilization
Thank you