Schedule

• Overview  silvia
• History  silvia
• Data Model  silvia
• Architecture  sunil
• Transaction support  sunil
• Case studies  silvia
Overview, history and data model
Overview: what is Membase?

- A key-value distributed database optimized for storing data behind web applications

- Simple - Fast - Elastic (by design)
Overview: before

Application Scales Out
Just add more commodity web servers
Overview: with Membase

Application user

Web application server

Membase Servers

DATA CENTER

ADMINISTRATOR CONSOLE
Overview: after

Application Scales Out
Just add more commodity web servers

Database Scales Out
Just add more commodity data servers
History

- Membase was developed by NorthScale, founded by several leaders of the memcached project
- June 2010: NorthScale, and project co-sponsors Zynga and NHN create a new project (membase.org).
- February 8, 2011, Membase merged with CouchOne. The merged project will be known as Couchbase
Today is, without question, one of the most exciting days of my career – in the top 3 for sure. It is difficult to imagine a marriage of technologies and cultures more right than the merger of Membase and CouchOne. Couchbase, the resulting company and product family, combines the industry’s leading caching and clustering technologies (powering 18 of the world’s top 20 websites and tens of thousands of others) with the most reliable and full-featured document database (with millions of installations worldwide). The result is, by a very wide margin, the fastest, safest and most comprehensive NoSQL database available.

James Phillips, senior Vice President
History

- Initial release March 2010
- Stable release 1.6.4.1 28 Dec 2010

<table>
<thead>
<tr>
<th>Databases and Caching</th>
<th>Enterprise Edition</th>
<th>Community Edition</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Membase Server</strong></td>
<td>Free Version</td>
<td>Free Download</td>
</tr>
<tr>
<td>RHEL, Ubuntu, Windows</td>
<td>Download</td>
<td>Download</td>
</tr>
<tr>
<td><strong>Memcached Server</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RHEL, Ubuntu, Windows</td>
<td>Download</td>
<td>Download</td>
</tr>
<tr>
<td><strong>Couchbase Server</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RHEL, Ubuntu, Windows, MacOS (community only)</td>
<td>Download</td>
<td>Download</td>
</tr>
<tr>
<td><strong>Proxies</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Moxi Server</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RHEL, Ubuntu, Windows</td>
<td></td>
<td>Download</td>
</tr>
</tbody>
</table>
Data Model

• Key-value

• Motivation: applications with natural keys to access data (e.g.: username.birthday)
Key-value

Key
Value

Data types:
Byte[]

Google protobuf
Thrift
Avro

“Any customer can have a car painted any colour that he wants so long as it is black.”
Operators and Programming Languages

- GET/SET
  - getl: get with an expiration time
- Increment/Decrement
- Append/Prepend

- Practically every language and application framework is supported (“memcapable”)

- Data manager: written in C, C++
- Cluster manager: Erlang/OTP
Transactions

- Based on CAS operations
- Compare and Swap
- special instruction that atomically compares the content of a memory location

```c
bool compare_and_swap (int *accum, int *dest, int newval)
{
    if ( *accum == *dest ) {
        *dest = newval;
        return true;
    } else {
        *accum = *dest;
        return false;
    }
}
```
Architecture and transaction support
What is the problem being solved?

- Highly interactive web apps
- Small amount of data
- Why doesn’t the traditional architecture work?
- Is nosql “DB” really a DB?
- Can a Database do what a nosql-db does?
  - If yes? Why not use a database
  - What is it that is really different?
    - De Normalized data
Membase - A practical path to “NoSQL” adoption
Physical Structures

• CA type system: scale linearly and always maintain consistency

• Clustering based on Erlang OTP

• Things are persistent, Data is written to Disk.
Elasticity
Elasticity
Elasticity
Architecture

memcached protocol listener/sender

moxi

membase storage engine

REST management API/Web UI

Heartbeat, Process monitor, Configuration manager, Global singleton supervisor on each node

Node health monitor

vBucket state and replication manager

Rebalance orchestrator

Erlang/OTP

HTTP, erlang port mapper, distributed erlang
vBuckets

Any given vbucket will be in one of the following states on any given server:

- **A**: Active, fully operational
- **D**: Dead, fully non-operational
- **R**: Replica, dead to clients, receives replicas
- **P**: Pending, blocks clients, receives replicas

http://blog.membase.com/scaling-memcached-vbuckets
vBucket mappings

All possible membase keys

Key → vBucket (hash function)

vBucket → Servers (table lookup)

vBucket-Server Map - Example

<table>
<thead>
<tr>
<th>vBucket</th>
<th>HostServer/ReplicaServers</th>
</tr>
</thead>
<tbody>
<tr>
<td>vBucket₁</td>
<td>Serverₐ / Server₇, Server₈</td>
</tr>
<tr>
<td>vBucket₂</td>
<td>Server₈ / Server₉, Server₁₀</td>
</tr>
<tr>
<td>vBucket₃</td>
<td>Server₉ / Server₅, Server₆</td>
</tr>
<tr>
<td>vBucket₄</td>
<td>Server₅ / Server₆, Server₇</td>
</tr>
<tr>
<td>vBucket₅</td>
<td>Server₆ / Server₈, Server₉</td>
</tr>
<tr>
<td>vBucket₆</td>
<td>Server₇ / Server₉, Server₈</td>
</tr>
</tbody>
</table>

Key₁, Key₂, Key₃, Key₄, Key₅, Key₆, Key₇, Key₈, Key₉, Key₁₀
TAP

• A generic, scalable method of streaming mutations from a given server
  – As data operations arrive, they can be sent to arbitrary TAP receivers
• Leverages the existing memcached engine interface, and the non-blocking IO interfaces to send data
• Three modes of operation
Replication & Failover

• Multi-model replication support
  • Peer-to-peer replication support with underlying architecture supporting master-slave replication

• Configurable replication count
  • Balance resource utilization with availability requirements

• High-speed failover
  Fast failover to replicated items based upon request
Case studies
Where does Membase fit?

- Online applications with a lot of users
- Applications with growing datasets which need quick access
• Who uses Membase?
Users: zynga

- Social game leader – FarmVille, Mafia Wars, Café World
- Over 230 million monthly users
  - **Membase Server**
    is the 500,000 ops-per-second database behind FarmVille and Café World
Case Study: Ad targeting

Targeting

Target users based on what they have bought and the sites they have visited

Behavioral targeting
Target users based on their displayed behaviors online.
- **Audience Behaviors**: Target one of over 160 behavioral segments (e.g., Auto Intenders, Apparel Shoppers, Family Planners, Travelers, Investors, Health Seekers, Trendy Homemakers, Moviegoers and more).
- **Custom Audience Behaviors**: Target a custom segment of users who have displayed relevant, discrete behaviors such as site visitation and buying habits across various content categories.

Target users based on registration information

Demographic
Age, gender, income, kids – it's the meat and potatoes of targeting.
- **User/Household**: Target users based on attributes from user registration or third-party data (e.g. age, gender, income, kids).
- **Site**: Place your ads on the sites that are visited most frequently by your desired audience. We aggregate our inventory by demographic and psychographic attributes – based on comScore data. You can, for example, place your ad on sites that are visited by users who attended college, users who applied offline for a credit card in the last six months, user who traveled domestically over six times in the last six months, and many more.
Case study: sharing network
Case study: sharing network

450/mo

million

consumers

50+

social channels

~850

thousand sites
Case study: targeting

Sharing Behavior
Search Keywords
Page Views

Log Files
HDFS
Membase

Map/Reduce
Content Analysis Taxonomy

User
Ad Server
Case Study: Ad targeting

Aol Advertising.

• Data management challenges:
  • to analyze billions of user-related events, presented as a mix of structured and unstructured data, to infer demographic, psychographic and behavioral characteristics ("cookie profiles")
  • make hundreds of millions of cookie profiles available to their AD targeting platform fast
  • to keep the user profiles updated
Case Study: Ad targeting

1. Events
2. Profiles, campaigns
3. Profiles, real-time campaign statistics

5 milliseconds to deliver an answer.
Thanks