SCALABLE DATA SERVICES

2110414 Large Scale Computing Systems
Natawut Nupairoj, Ph.D.

Outline

- □ Overview
- □ MySQL Database Clustering
- □ GlusterFS
- □ Memcached

Overview

Problems of Data Services

- Data retrieval is usually the bottleneck
  - Searching
  - Transferring
- Basic performance improvement schemes
  - Data partitioning
  - Data replication – need to maintain consistency
- Other techniques
  - Database Clustering
  - High-performance File Systems
  - In-memory caching

MySQL Database Clustering

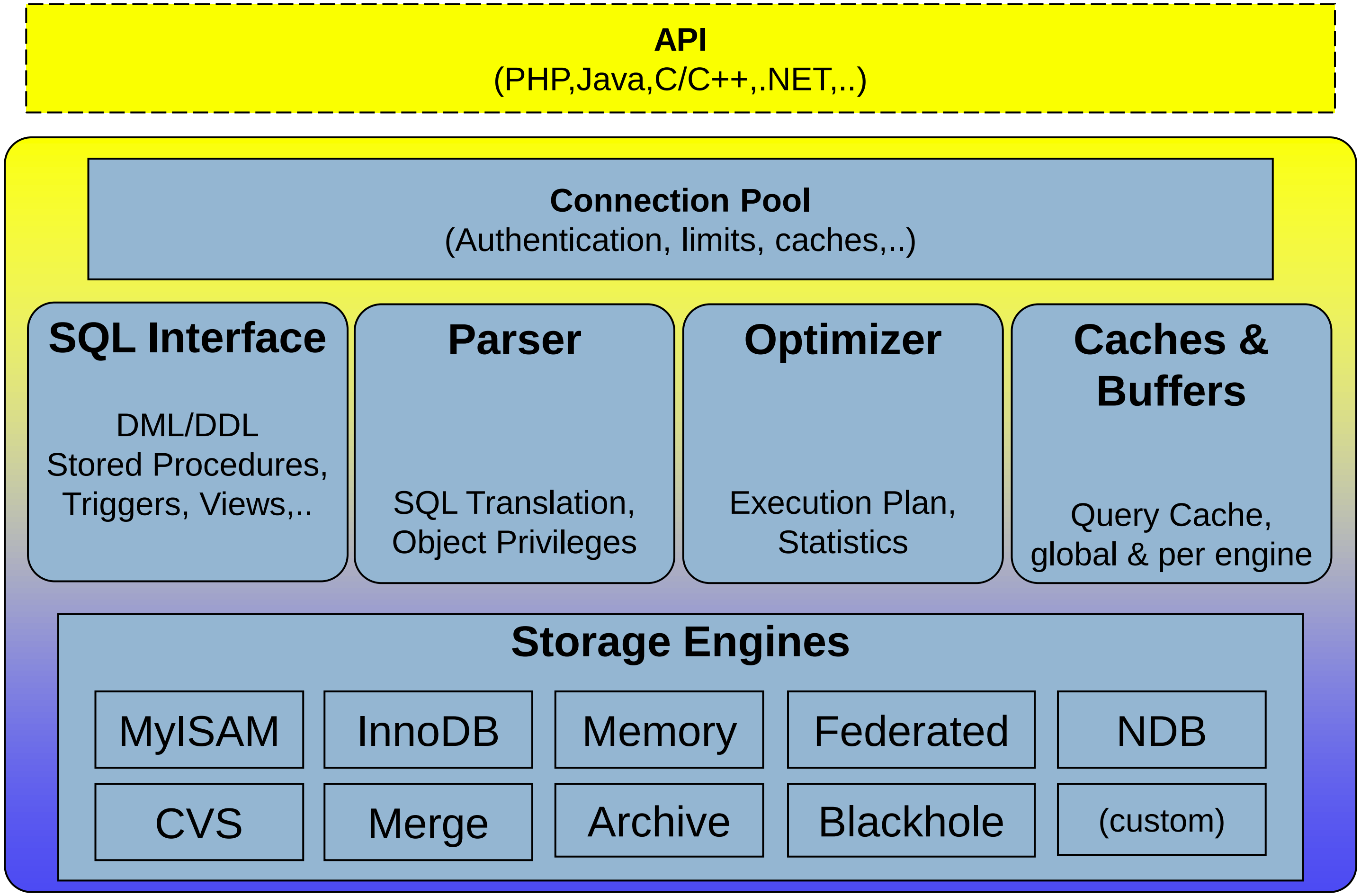
Adapted from G. Vanderkelen,

“MySQL Cluster: An introduction”, 2006

Quick intro to MySQL

- MySQL is a DBMS running on most OS
- Reputation for speed, quality, reliability and easy to use
- Storage Engines (MyISAM, InnoDB, ..)
- Support standard SQL and other features
  - Stored procedures
  - Triggers, Updatable Views, Cursors
  - Precision math
  - Data dictionary (INFORMATION_SCHEMA database)
  - and more..
- Lots of Connectors and API available

MySQL Architecture



API
(PHP,Java,C/C++,.NET,..)

Connection Pool
(Authentication, limits, caches,..)

SQL Interface
DML/DDL
Stored Procedures,
Triggers, Views,..

Parser
SQL Translation,
Object Privileges

Optimizer
Execution Plan,
Statistics

Caches & Buffers
Query Cache,
global & per engine

Storage Engines

MyISAM	InnoDB	Memory	Federated	NDB
CVS	Merge	Archive	Blackhole	(custom)

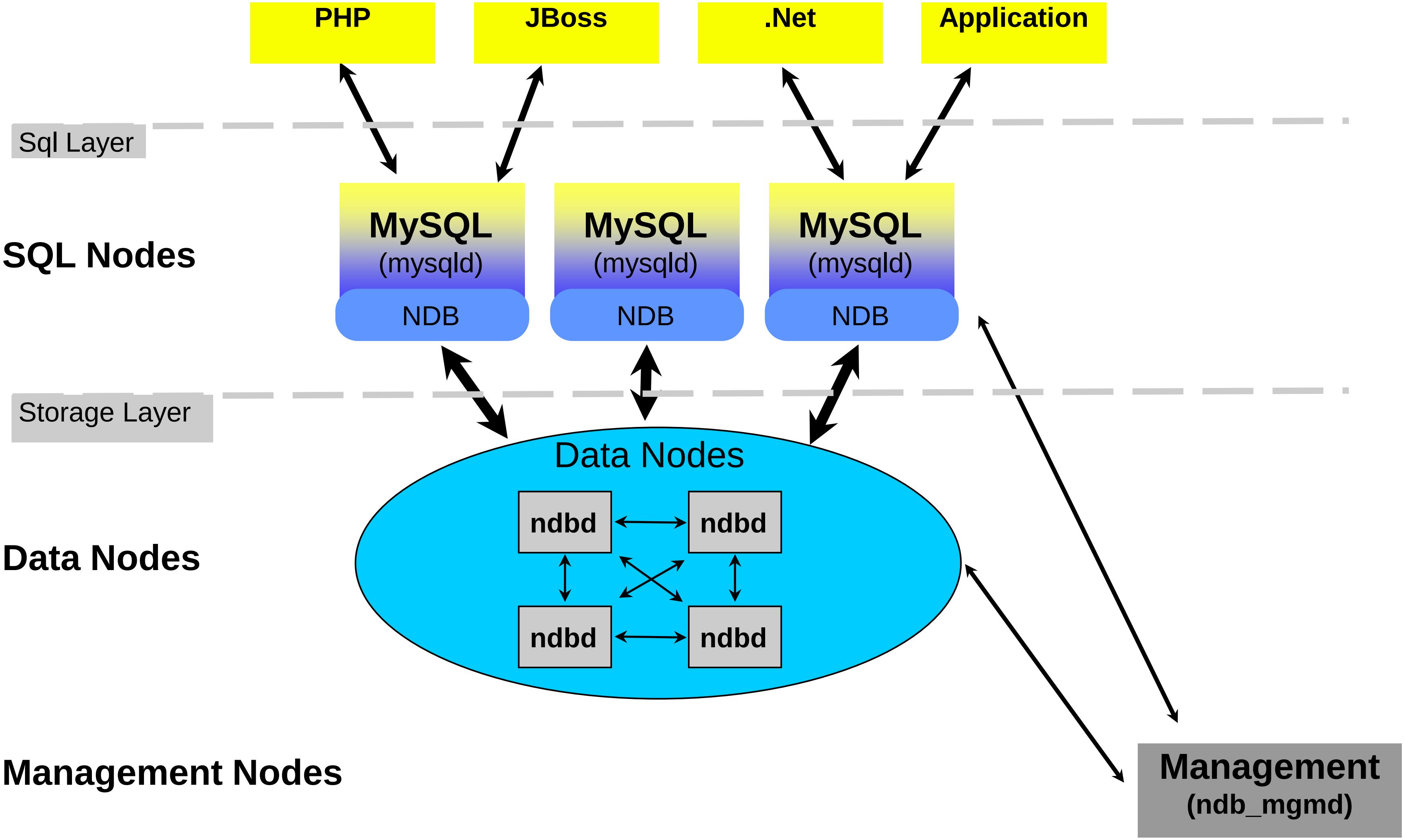
What is MySQL Cluster?

- In-memory storage
  - data and indices in-memory
  - check-pointed to disk
- Shared-Nothing architecture
- No single point of failure
  - Synchronous replication between nodes
  - Fail-over in case of node failure
- Row level locking
- Hot backups

Cluster Nodes

- Participating processes are called ‘nodes’
  - Nodes can be on same computers
- Two tiers in Cluster:
  - SQL layer
    - SQL nodes (also called API nodes)
  - Storage layer
    - Data nodes, Management nodes

Components of a Cluster



Data nodes

- Contain data and index
- Used for transaction coordination
- Each data node is connect to the others
- Shared-nothing architecture
- Up to 48 data nodes

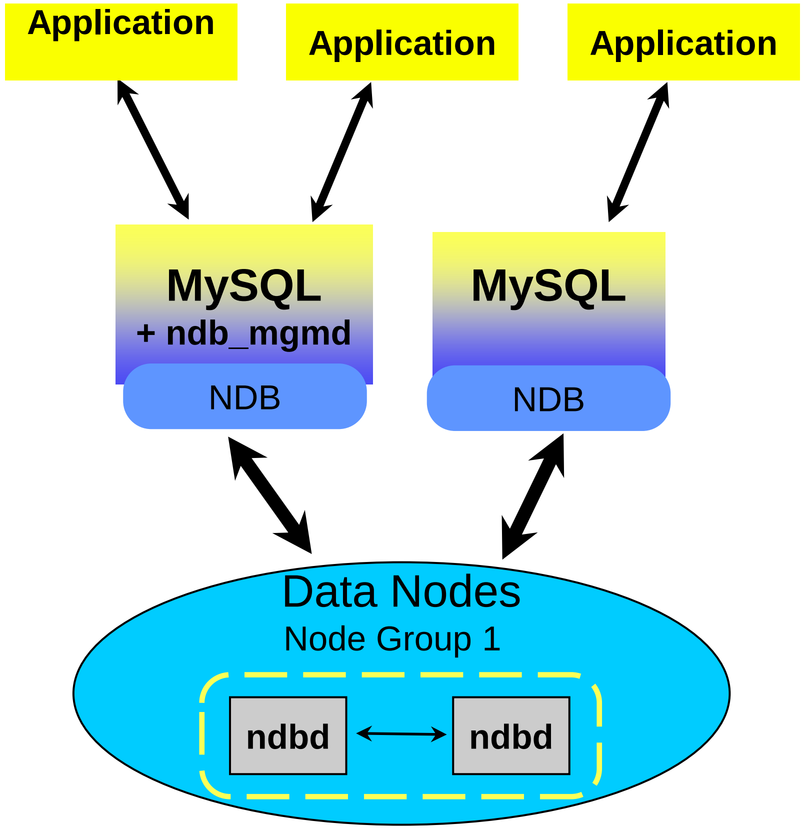
SQL nodes

- Usually MySQL servers
- Also called API nodes
- Each SQL node is connected to all data nodes
- Applications access data using SQL
- Native NDB application (e.g. ndb_restore)
- Client application written using NDB API

Management nodes

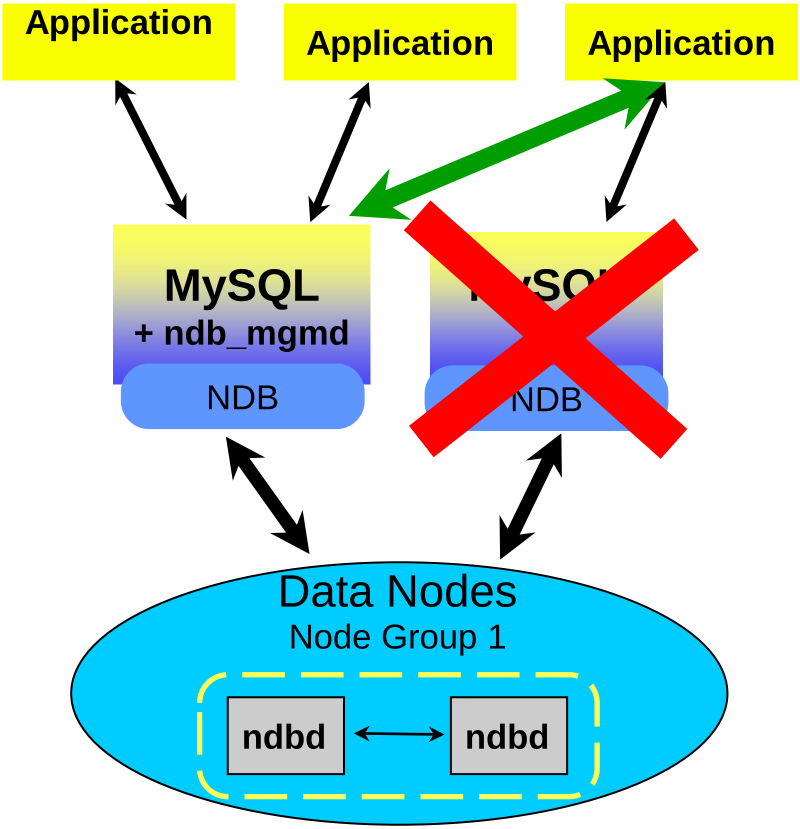
- Controls setup and configuration
- Needed on startup for other nodes
- Cluster can run without
- Can act as arbitrator during network partitioning
- Cluster logs
- Accessible using ndb_mgm CLI

A Configuration



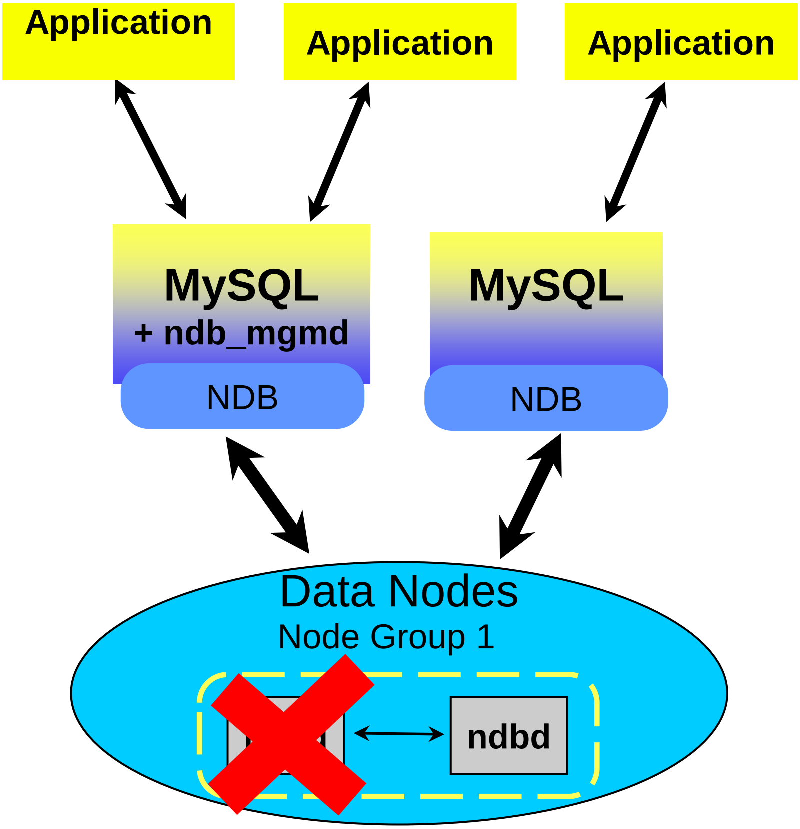
Failure: MySQL server

- Applications can use other
- mysqld reconnects

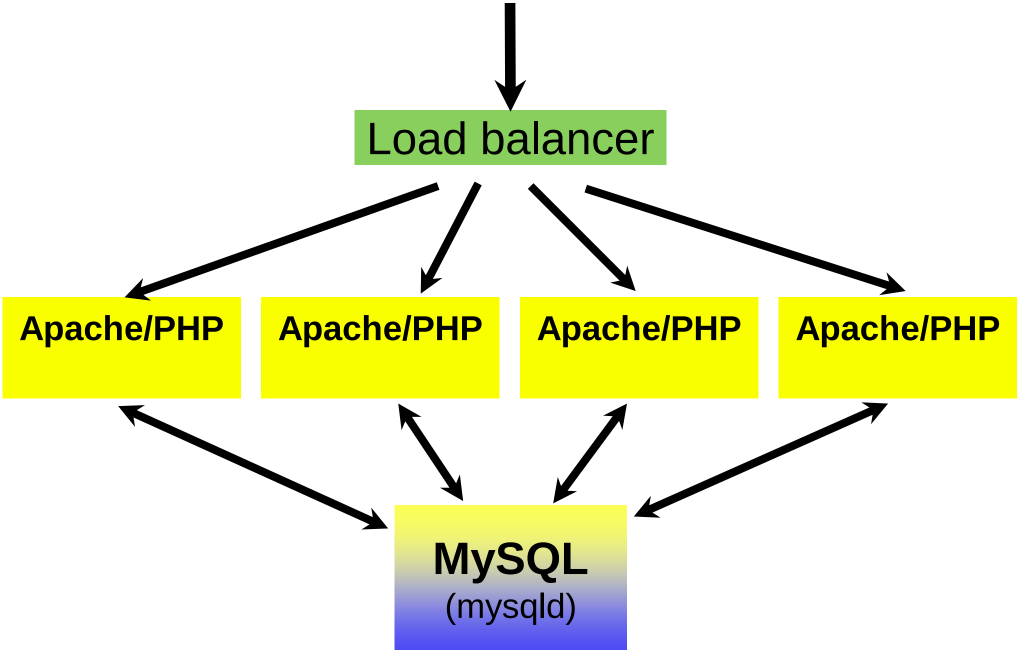


Failure: Data Node

- Other data nodes know
- Transaction aborted
- Min. 1 node per group needed
- 0 nodes in group = shutdown

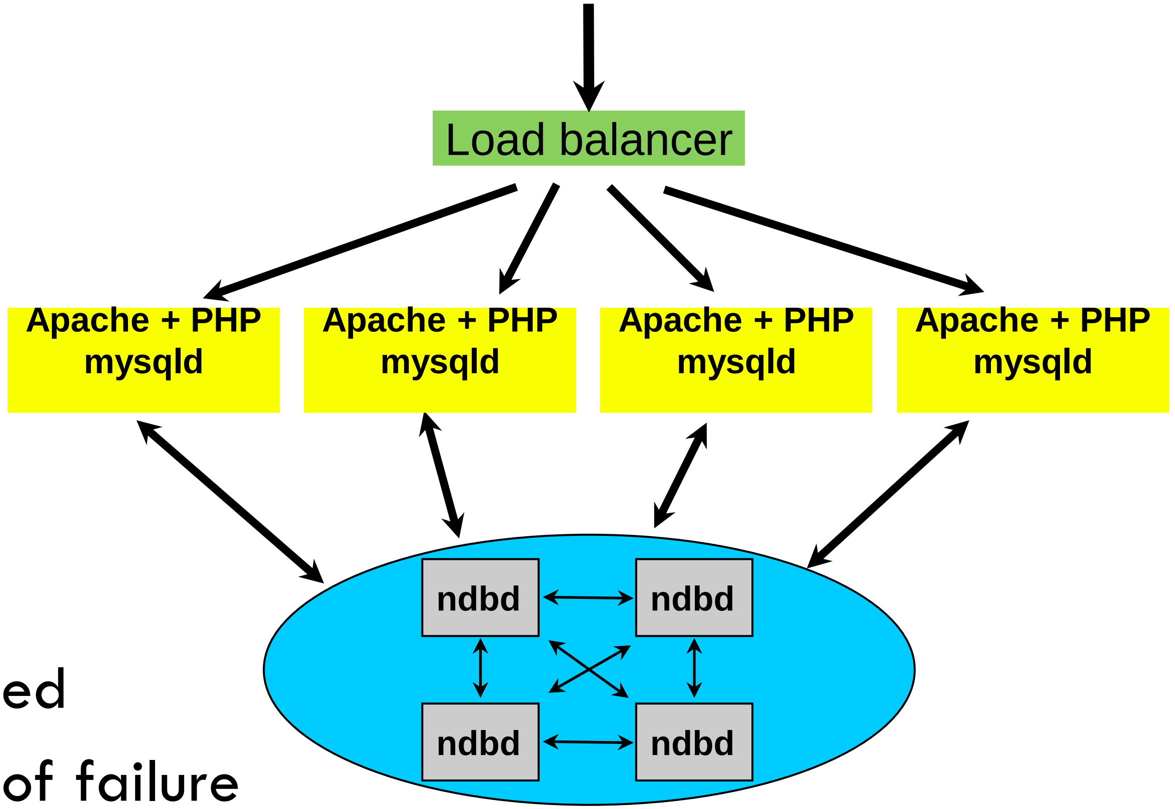


Example: Web Sessions



- Without Cluster
  - One MySQL server holding data
  - Single point of failure

Web Sessions



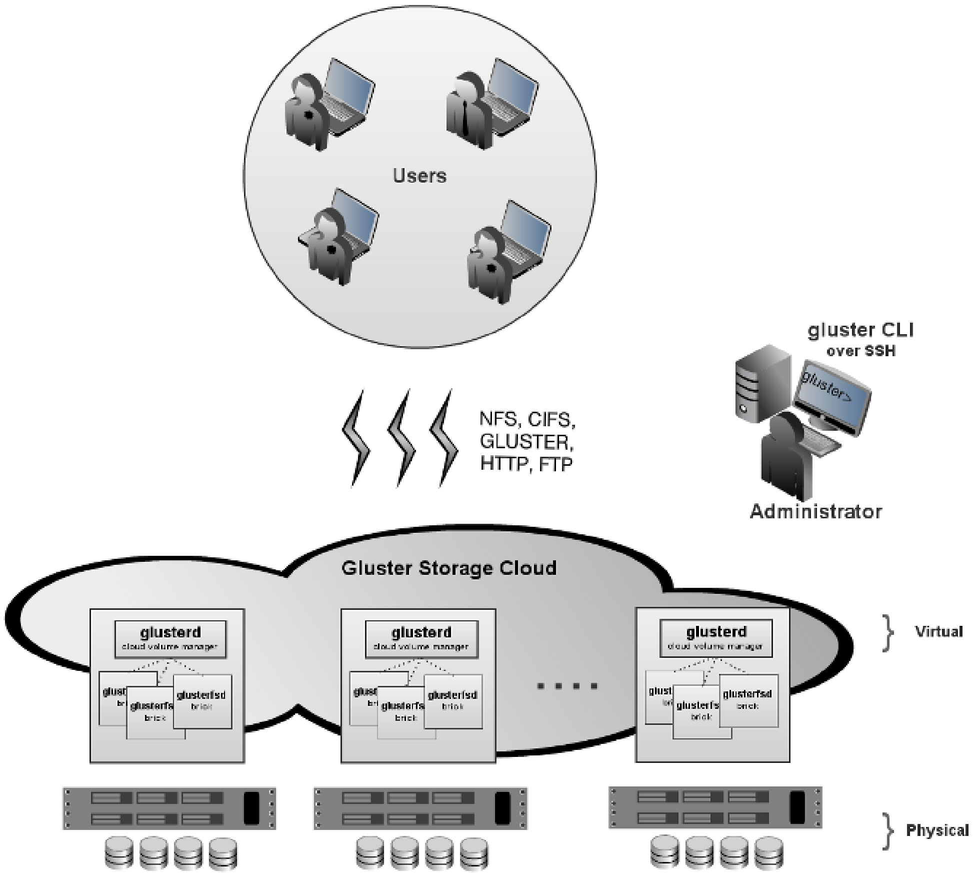
- With Cluster
  - MySQL distributed
  - No Single point of failure
  - Shared storage, but redundant

GlusterFS

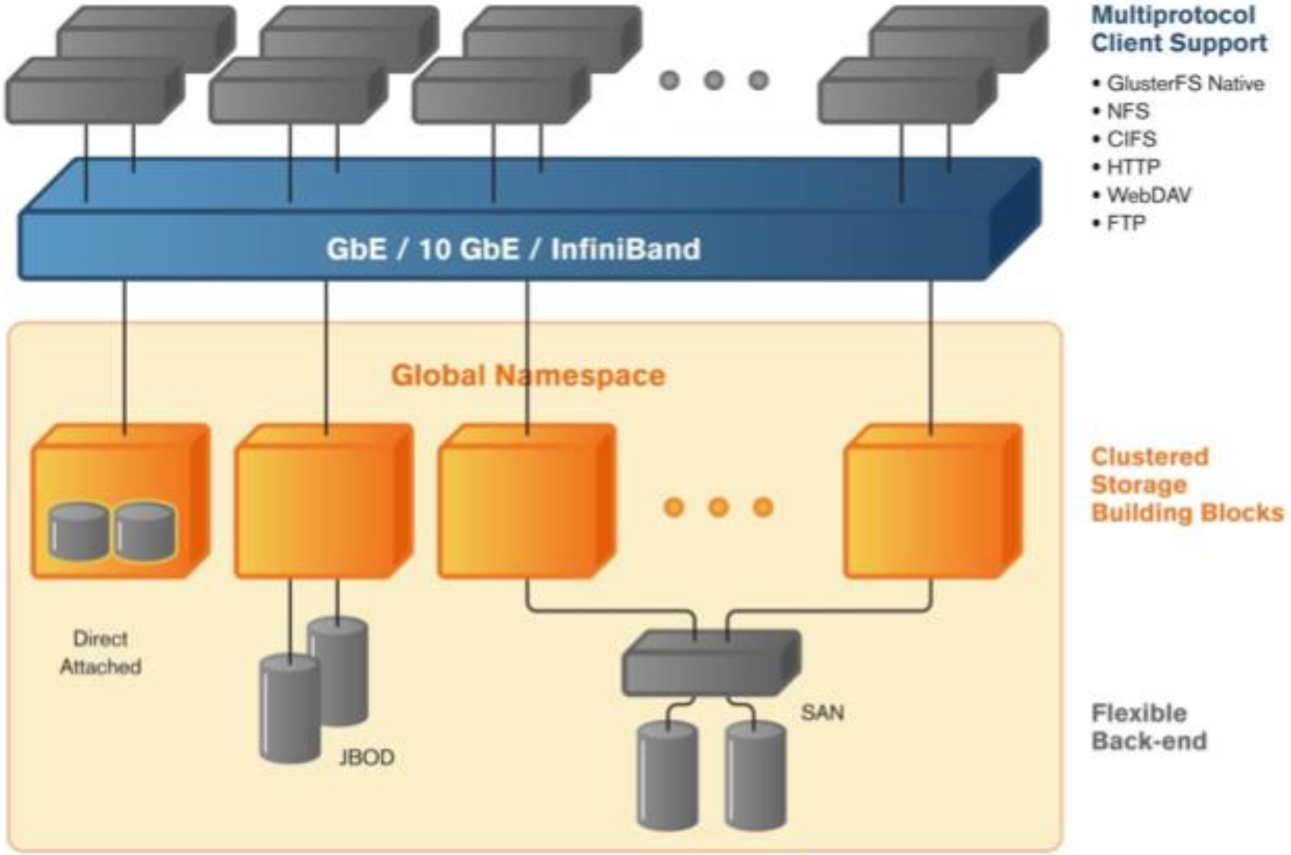
What is GlusterFS?

- Open source, clustered file system
- Scale up to several petabytes for thousands of clients
- Aggregate disk and memory resources into a single global namespace over network
  - Leverage commodity hardware
  - Lead to storage virtualization
- Allow administrators to dynamically expand, shrink, rebalance, and migrate volumes
- Provide linear scalability, high performance, high availability, and ease of management

GlusterFS Overview

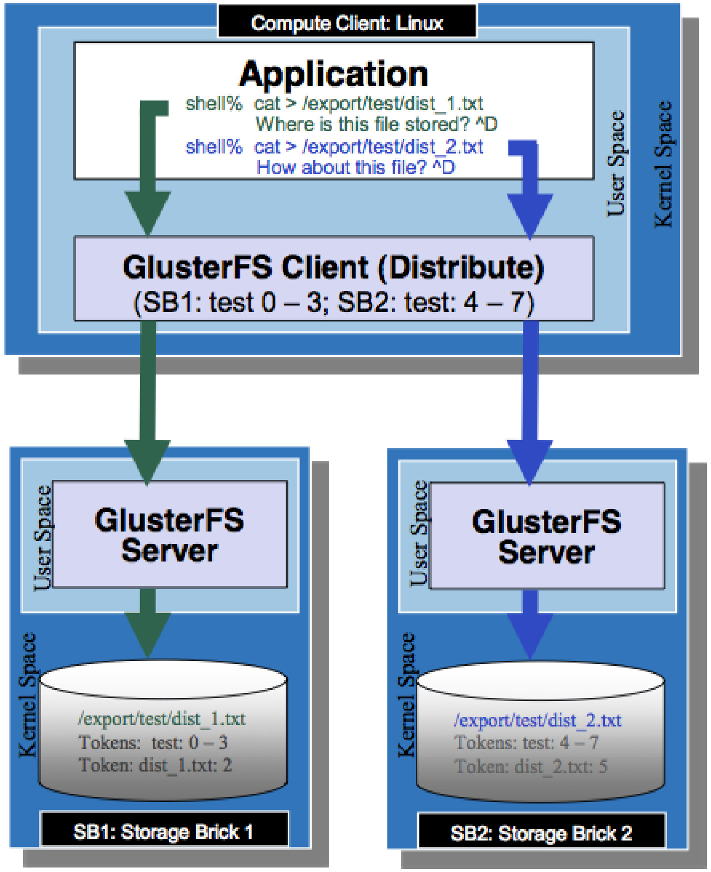


GlusterFS Architecture



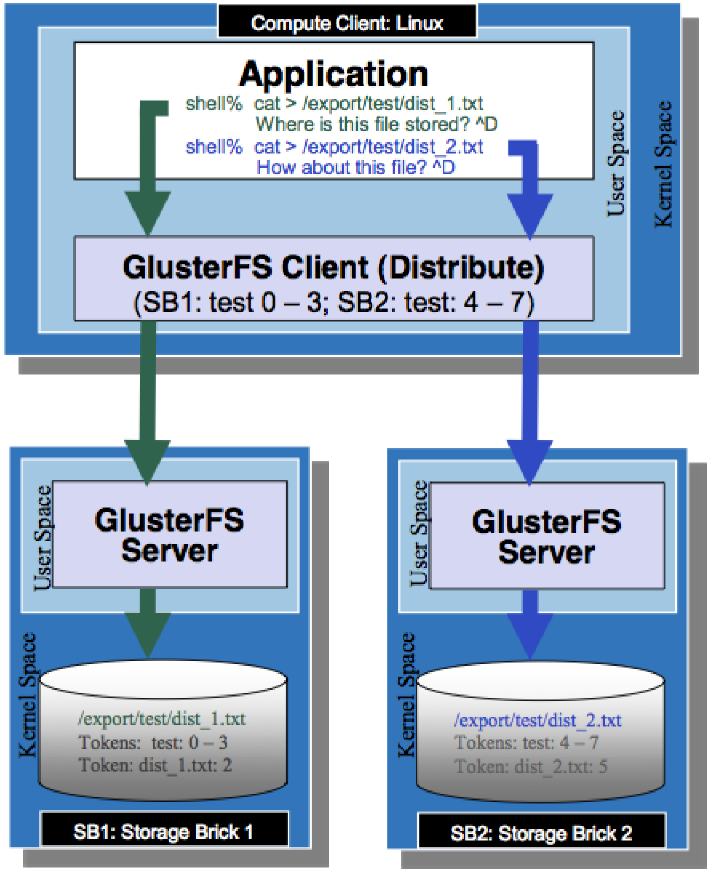
GlusterFS Load Balancing Mode

- Data are stored as files and folders
- Use tokens
  - Extended attributes of a file
  - Identify the location of a file
  - Distributed across directories
  - No need for dedicated metadata server
- Gluster translates the requested file name to a token and access the files directly



GlusterFS Replication Mode

- Support auto replication across multiple storages
- Provide high availability (auto fail-over) and auto self-healing
- Uses load balancing to access replicated instances



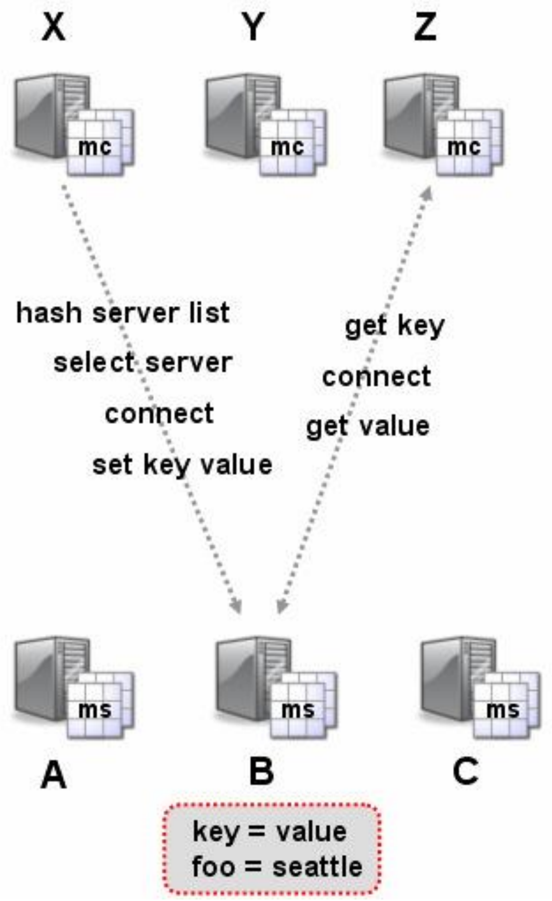
```
shell%  cat > /export/test/dist_1.txt
        Where is this file stored? ^D
shell%  cat > /export/test/dist_2.txt
        How about this file? ^D
```

Memcached

What is Memcached?

- General-purpose high-performance open source distributed memory caching system
  - giant hash table distributed across multiple machines
- Speed up dynamic database-driven websites by caching data and objects in RAM
- Being used by many popular web sites
  - LiveJournal, Wikipedia, Facebook, Flickr, Twitter, Youtube
- API is available in many languages
  - PHP, Java, Python, Perl, C, MySQL API

Basic Memcached Operations



Client X

1) set key “foo” with value “seattle”

2) hashes the key against server list

3) Server B is selected

4) connects to Server B and sets key

Client Z

5) get key “foo”

6) connects to Server B

7) requests “foo” and gets value “seattle”

Memcached with Java

```
MemcachedClient c=new MemcachedClient(
  new InetSocketAddress("127.0.0.1", 11211));

c.set("someKey", 3600, someObject);
Object myObject=c.get("someKey");
c.delete("someKey")
```

Memcached and MySQL

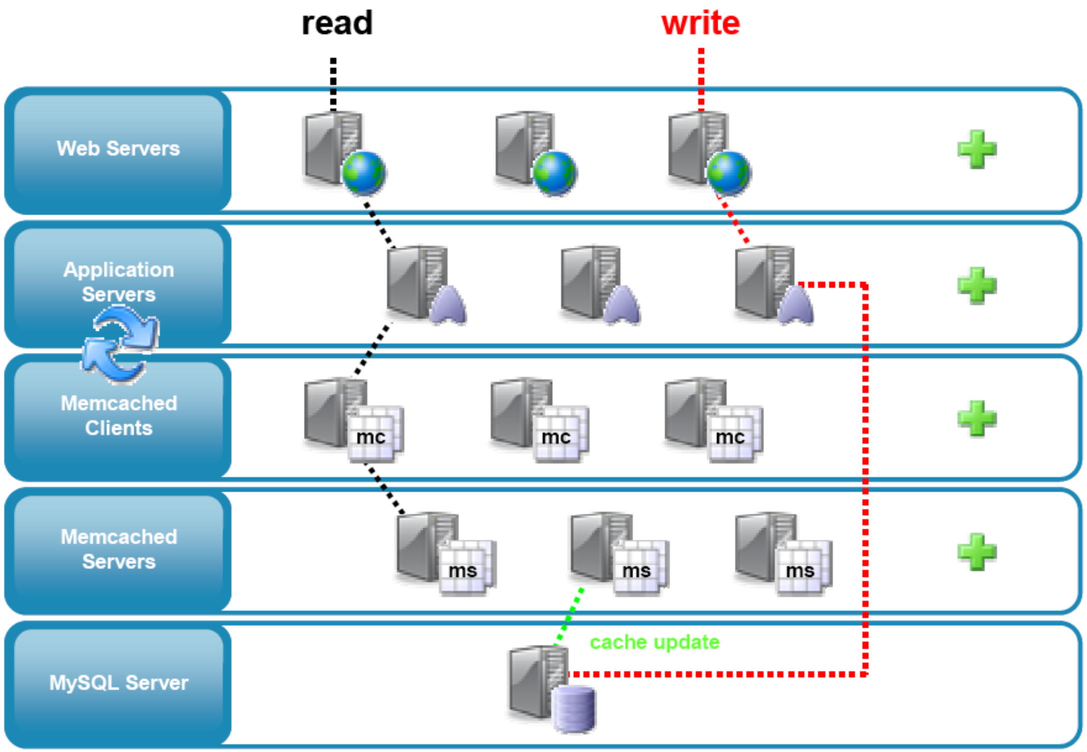
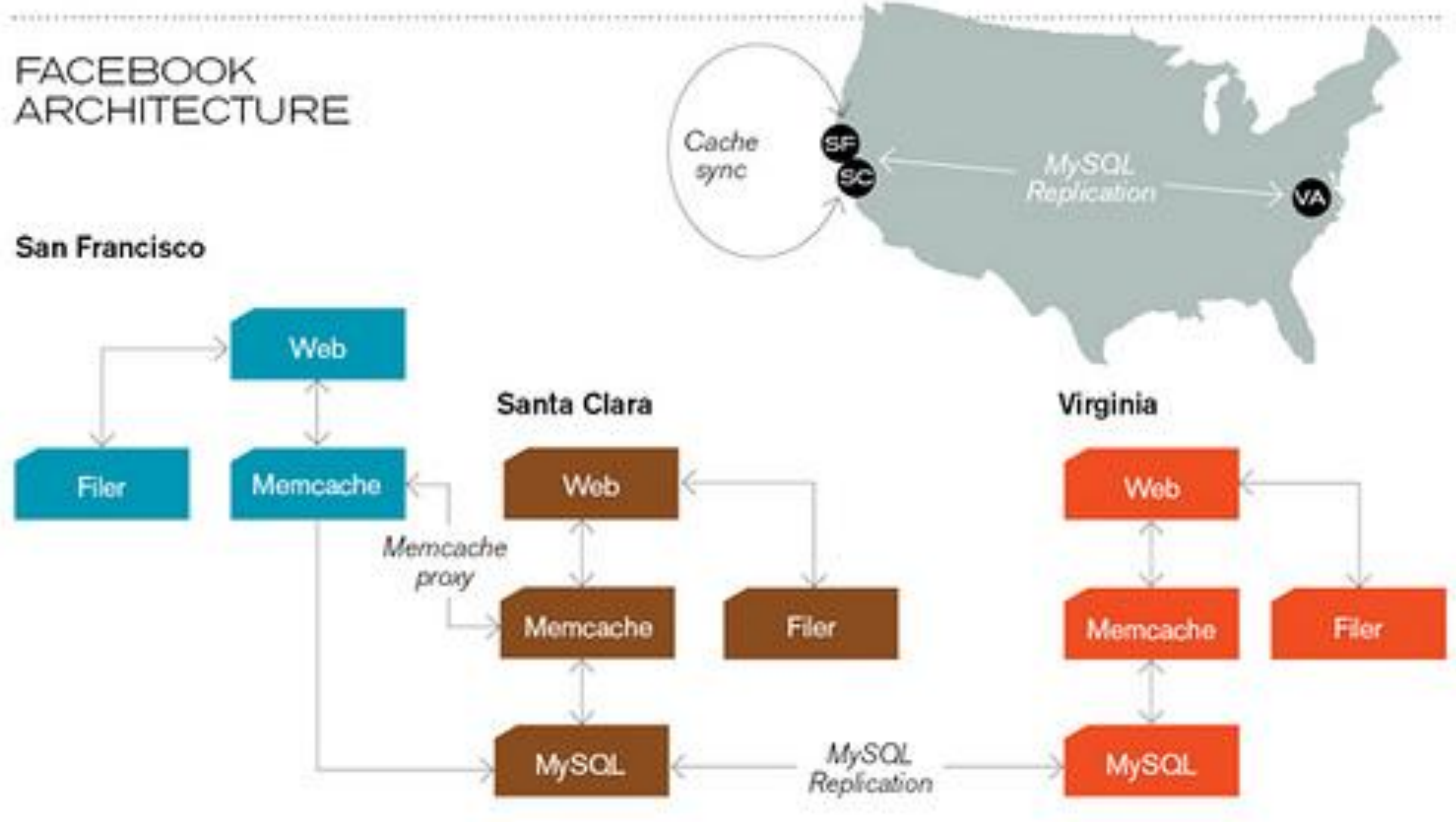


Figure 2: Multiple Memcached Servers and a Stand-Alone MySQL Server

- □ Caching the results of database queries
- □ “SELECT * FROM users WHERE userid = ?” with (userid:user)

Putting It All Together: Facebook Architecture



References

- P. Strassmann, “ Introduction to Virtualization”, http://www.strassmann.com/pubs/gmu/2008-10.pdf, George Mason University, 2008
- Gluster Community, “Gluster 3.1.x Documentation”, http://www.gluster.com/community/documentation/index.php/Main_Page, 2010
- “Designing and Implementing Scalable Applications with Memcached and MySQL”, MySQL White Paper, 2008
- Wikipedia, “Memcached”, http://en.wikipedia.org/wiki/Memcached, 2010