

# MySQL High Availability Solutions

Lenz Grimmer

<lenz@grimmer.com>

2009-02-08

FOSDEM 2009, Brussels, Belgium

# MySQL DevRoom in AW1.126

- MySQL Cluster session in the DevRoom NOW (sorry)
- Updates to the MySQL DevRoom Schedule
- 13:15: Q&A with Kaj Arnö

[http://forge.mysql.com/wiki/FOSDEM\\_2009](http://forge.mysql.com/wiki/FOSDEM_2009)

# Agenda

- High Availability: Concepts & Considerations
- MySQL Replication
- DRBD / Heartbeat
- MySQL Cluster
- Other HA tools/applications
- Links

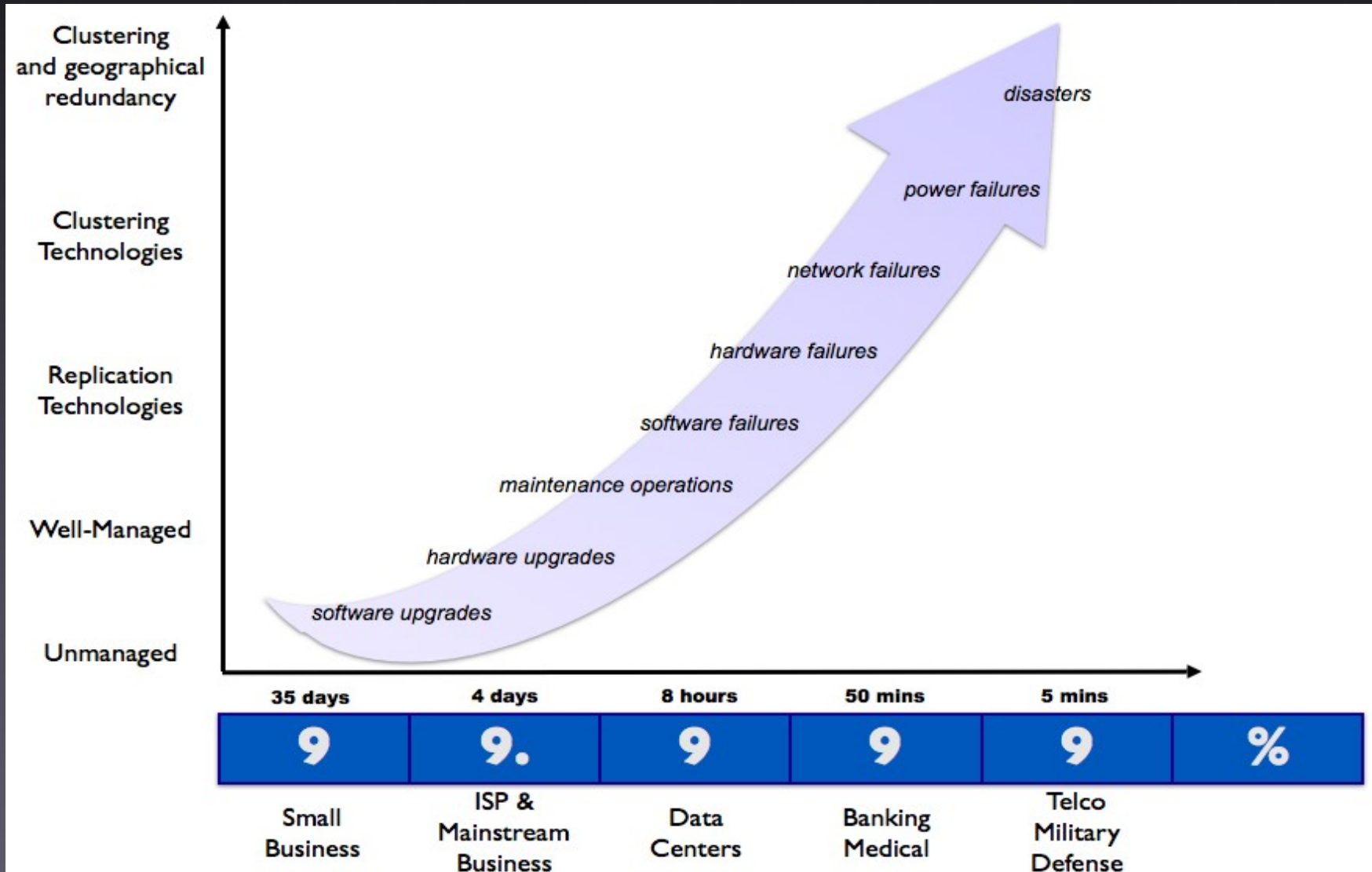
# Why High Availability Matters

- Something can break and usually will
- Maintenance requirements
- Downtime is expensive
- You miss \$\$\$
- Your Boss complains
- New Site visitors won't come back
- Adding HA to an existing system is tricky

# What Is HA Clustering?

- Redundancy
- One service goes down → others take over
- IP address takeover, service takeover
- Failover vs. failback vs. switchover
- **Not designed** for high-performance
- **Not designed** for high throughput (load balancing)

# High Availability Levels



# Eliminating the SPOF

- Identify what will fail
  - Disks
- Find out what can fail
  - Network cables
  - OOM
  - Power supplies

# HA Components

- Heartbeat
  - Checks that services that are being failed over, are alive.
  - Can check individual servers, software services, networking etc.
- HA Monitor
  - Configuration of the services
  - Ensures proper shutdown and startup
  - Allows manual control
- Shared storage / Replication

# Replication vs. Shared Storage

- Shared storage resource can become SPOF
- Split brain situations can lead to mayhem (e.g. mounting file systems concurrently)
- SAN/NAS read I/O overhead
- Consistency of replicated data
- Synchronous vs. asynchronous replication

# Split-Brain

- Communications failures can lead to separated partitions of the cluster
- If those partitions each try and take control of the cluster, then it's called a split-brain condition
- If this happens, then bad things will happen  
<http://linux-ha.org/BadThingsWillHappen>
- Use Fencing or Moderation/Arbitration to avoid it

# Rules of High Availability

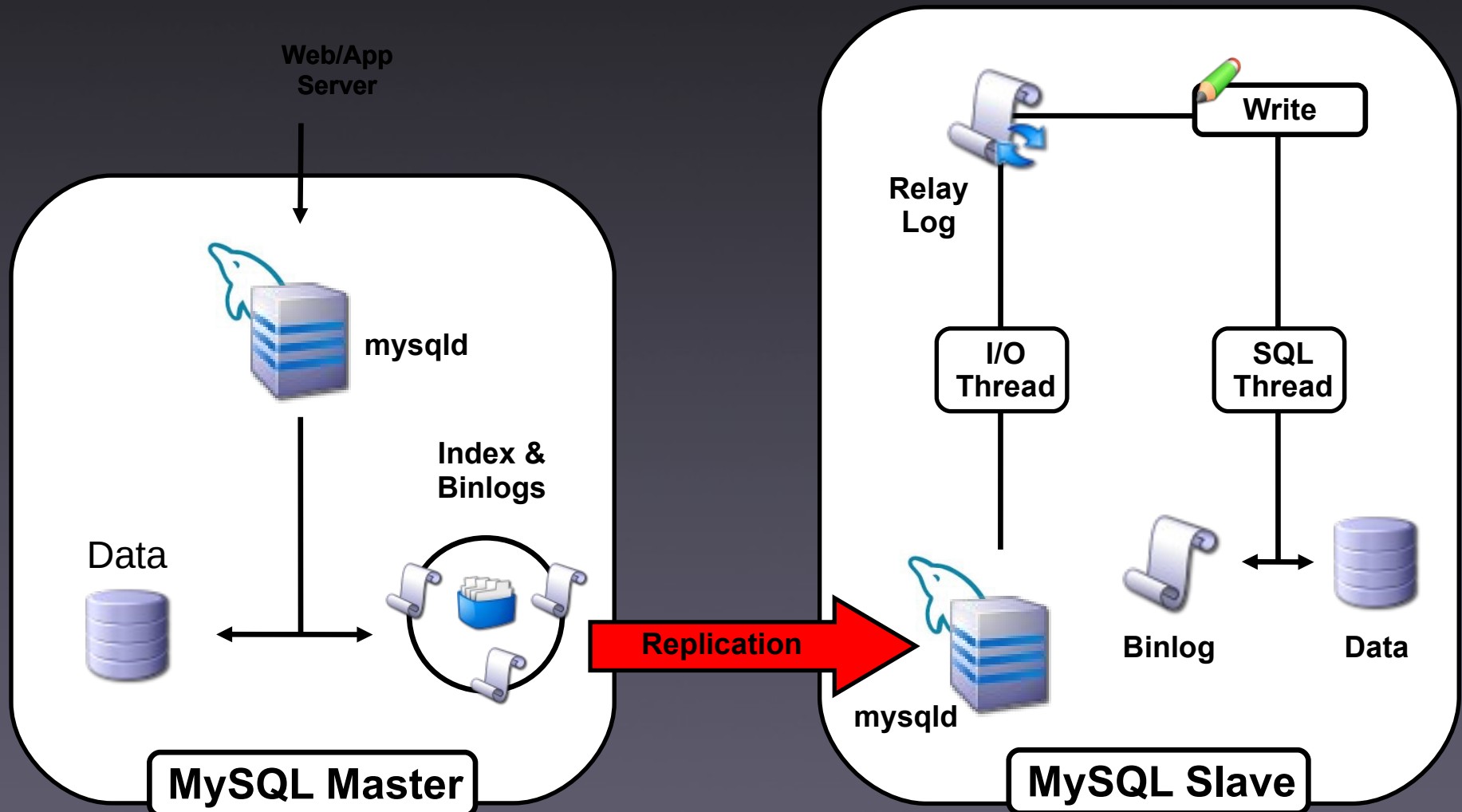
- Prepare for failure
- Aim to ensure that no important data is lost
- Keep it simple, stupid (KISS)
- Complexity is the enemy of reliability
- Automate it
- Test your setup frequently!

# MySQL Replication

- One-way, statement- or row-based
- One Master, many Slaves
- Asynchronous – Slaves can lag
- Master maintains binary logs & index
- Easy to use and set up
- Built into MySQL
- Replication is single-threaded
- No automated fail-over
- No heartbeat, no monitoring

# MySQL Replication Overview

 Read & Write



# Statement-based replication

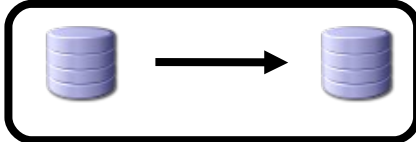
- Pro
  - Proven (around since MySQL 3.23)
  - Smaller log files
  - Auditing of actual SQL statements
  - No primary key requirement for replicated tables
- Con
  - Non-deterministic functions and UDFs
  - `LOAD_FILE()`, `UUID()`, `USER()`, `FOUND_ROWS()`  
(but `RAND()` and `NOW()` work)

# Row-based replication

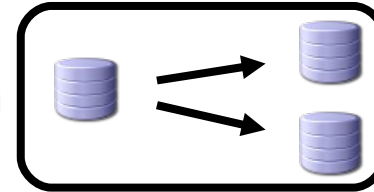
- Pro
  - All changes can be replicated
  - Similar technology used by other RDBMSes
  - Fewer locks required for some INSERT, UPDATE or DELETE statements
- Con
  - More data to be logged
  - Log file size increases (backup/restore implications)
  - Replicated tables require explicit primary keys
  - Possible different result sets on bulk INSERTs

# Replication Topologies

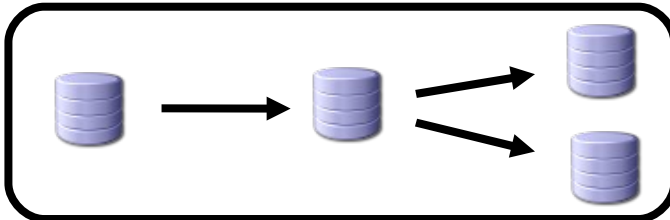
Master > Slave



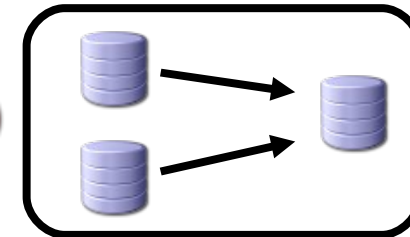
Master > Slaves



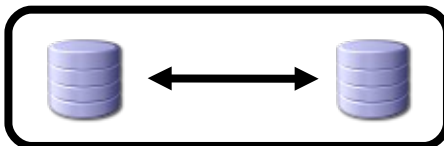
Master > Slave > Slaves



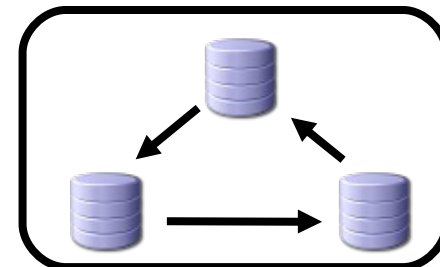
Masters > Slave (Multi-Source)



Master < > Master (Multi-Master)



Ring (Multi-Master)



# Master-Master Replication

- Useful for easier failover
- Not suitable for load-balancing
  - Writes still end up on both machines
  - Neither machine has the authoritative data
- Don't write to both masters!
- Use Sharding or Partitioning instead (e.g. MySQL Proxy)

# MySQL Replication as an HA Solution

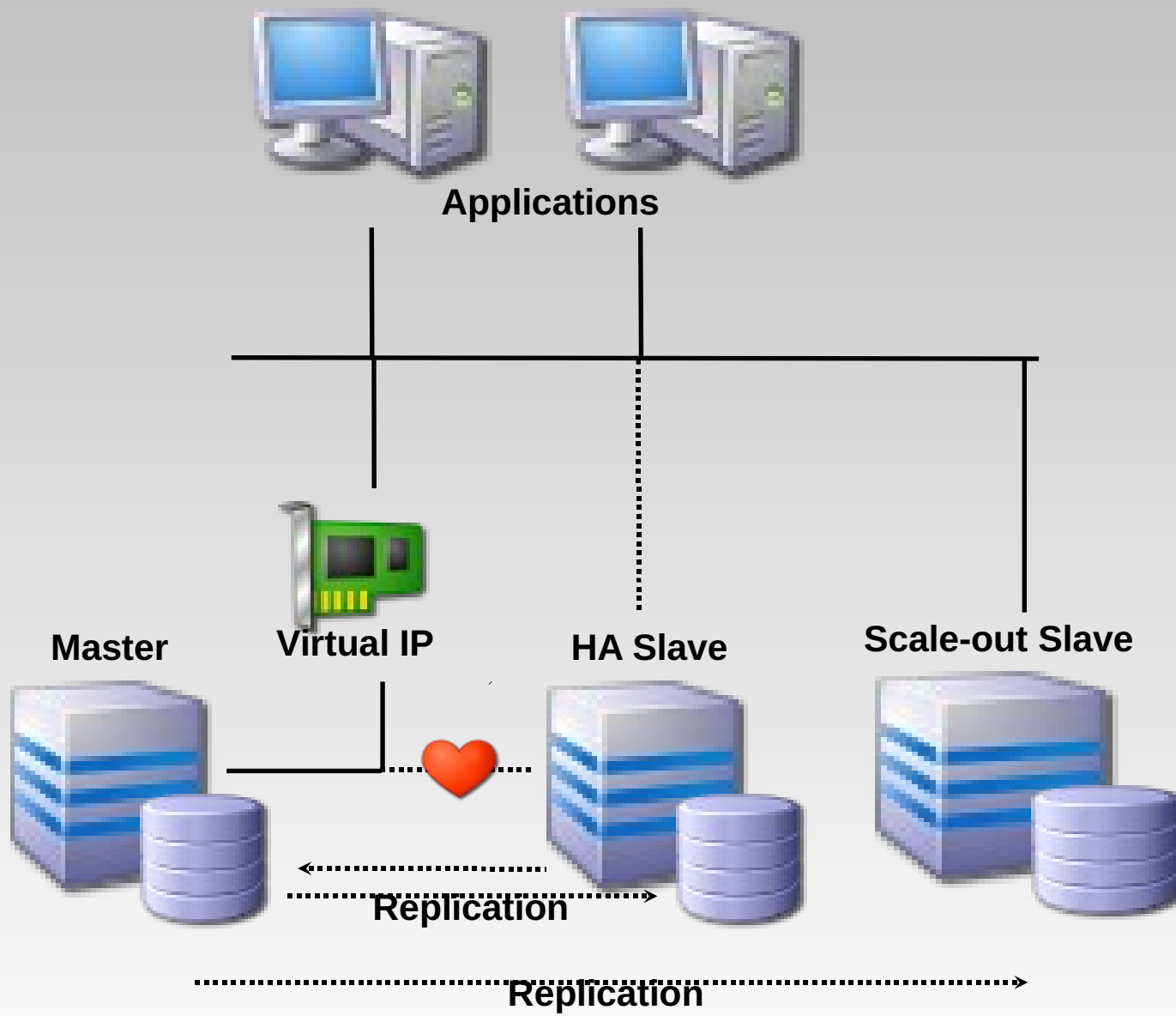
- What happens if the Master fails?
  - Nothing really, except the application will not work and the Slave will not have anything to replicate from
- What happens if the Slave fails?
  - Nothing really (except data will no longer be replicated)
- This doesn't sound like High Availability?
  - Correct – MySQL Replication is a component of an HA setup, but it doesn't implement all parts of an HA Solution!

# Replication & HA

- Combined with Heartbeat
- Virtual IP takeover
- Slave gets promoted to Master
- Side benefits: load balancing & backup
- Tricky to fail back
- No automatic conflict resolution
- Proper failover needs to be scripted

# Linux-HA / Heartbeat

- Supports 2 or more nodes
- Resource monitoring
- Active fencing mechanism: STONITH
- Policy-based resource management, dependencies & constraints
- Time-based rules
- Includes support for many applications
- GUI support
- Low dependencies / requirements
- Subsecond node failure detection



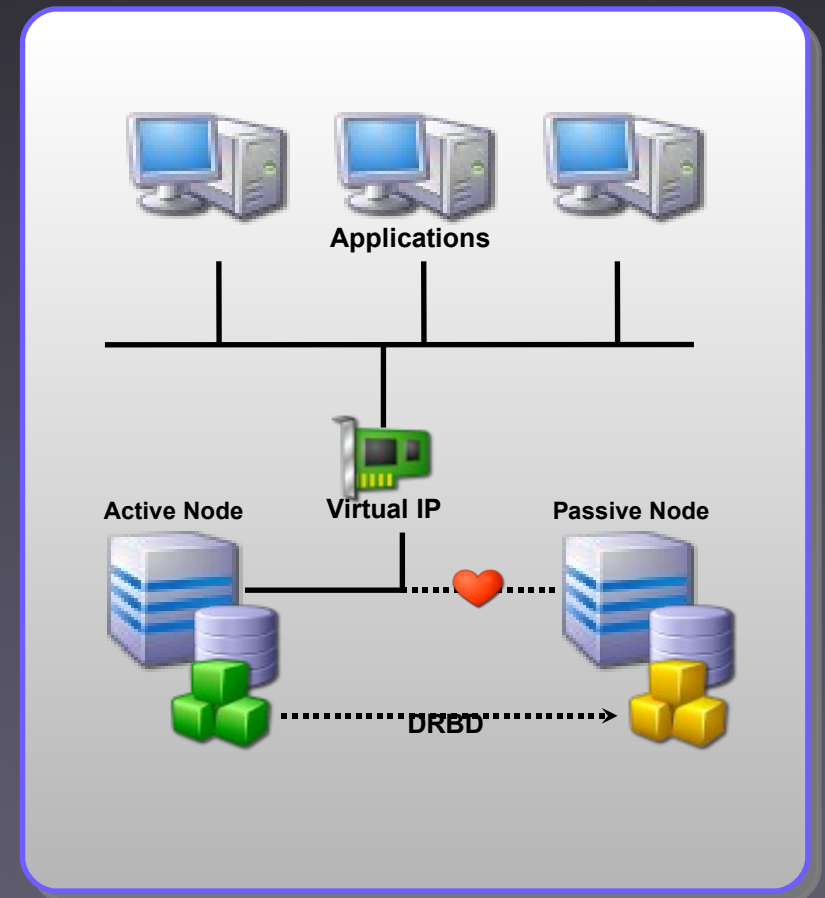
# DRBD



- Distributed Replicated Block Device
- “RAID-1 over network”
- Synchronous/asynchronous block replication
- Automatic resync on recovery
- Application-agnostic
- Can mask local I/O errors
- Active/passive configuration by default
- Dual-primary mode (requires a cluster filesystem like GFS or OCFS2)
- <http://drbd.org/>

# DRBD in detail

- DRBD Replicates data between two disk partitions
- DRBD integrates nicely with Linux-HA and other HA Solutions
- MySQL runs on the Active node as usual
- MySQL is dormant on the Passive node
- DRBD is Linux only



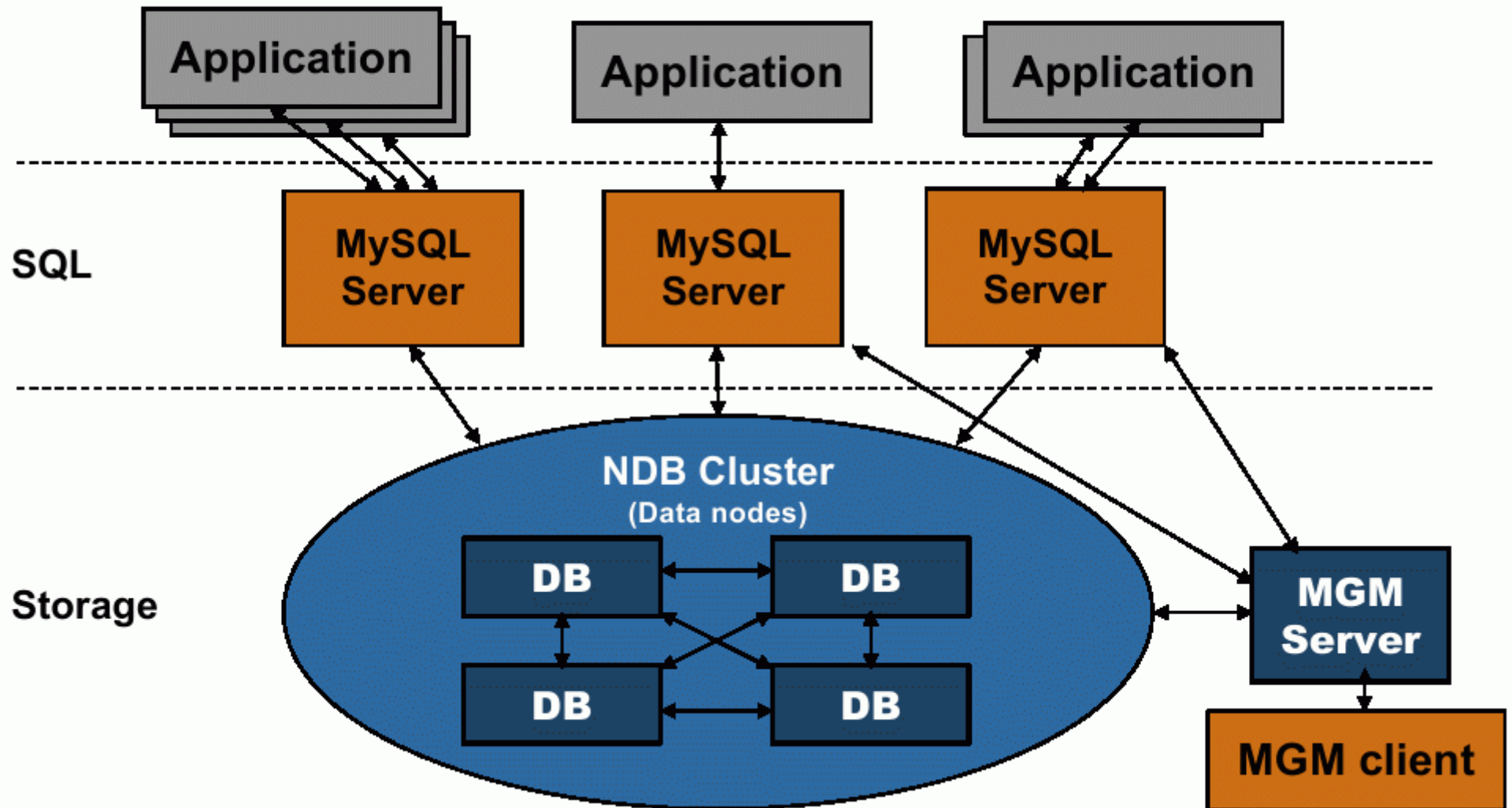
# MySQL Cluster

- Shared-nothing architecture
- Automatic partitioning
- Distributed Fragments
- Synchronous replication
- Fast automatic fail-over of data nodes
- Automatic resynchronization
- Transparent to Application
- Supports Transactions

# MySQL Cluster

- In-memory indexes
- Not suitable for all query patterns (complex JOINS, range scans)
- No support for foreign key constraints
- Not suitable for large datasets/transactions
- Latency matters
- Can be combined with MySQL Replication (RBR)

# Cluster Components



	Requirements	MySQL Replication	MySQL Replication & Heartbeat	MySQL, Heartbeat & DRBD	MySQL Cluster
<b>Availability</b>	Automated IP Failover	No	Yes	Yes	No
	Automated DB Failover	No	No	Yes	Yes
	Typical Failover time	Varies	Varies	< 30s	< 3s
	Auto resync of data	No	No	Yes	Yes
	Geographic redundancy	Yes	Yes	MySQL Replication	MySQL Replication
	<b>Scalability</b>	Built-in load balancing	MySQL Replication	MySQL Replication	MySQL Replication
Read-intensive		Yes	Yes	MySQL Replication	Yes
Write-intensive		No	No	Possible	Yes
# Nodes/Cluster		Master/Slave(s)	Master/Slave(s)	Active/Passive	255

# MMM – MySQL Master-Master Replication Manager

- Collection of scripts to perform monitoring/failover and management
- Master-Master replication configurations (one writable node)
- Failover by moving virtual IP
- <http://code.google.com/p/mysql-master-master/>

# Flipper

- Perl Script that manages pairs of MySQL servers replicating to each other.
- Automatic switchover, triggered manually
- Enables one half of the pair to be taken offline for maintenance work while the other half carries on dealing with queries from clients
- Moves IP addresses based on a role ("read-only", "writable") between two nodes in the master pair, to ensure that each role is available
- <http://provencal.com/software/flipper/>

# Red Hat Cluster Suite

- HA and load balancing
- Supports up to 128 nodes
- Shared storage
- Monitors services, file systems and network status
- Fencing (STONITH) or distributed lock manager
- Configuration synchronization
- [http://www.redhat.com/cluster\\_suite/](http://www.redhat.com/cluster_suite/)

# Solaris Cluster / OpenHA Cluster

- Provides failover and scalability services
- Solaris / OpenSolaris
- Kernel-level components plus userland
- Agents monitor applications
- Geo Edition to provide Disaster Recovery using Replication
- Open Source since June 2007
- <http://www.opensolaris.org/os/community/ha-clusters/>

# Related tools / Links

- Linux Heartbeat  
<http://linux-ha.org/>
- Linux Cluster Information Center  
<http://www.lcic.org/ha.html>
- Red Hat Cluster Suite  
[http://www.redhat.com/cluster\\_suite/](http://www.redhat.com/cluster_suite/)
- Sun Open High Availability Cluster  
<http://opensolaris.org/os/project/ha-mysql/>

# Tools/Links

- **MySQL Multi-Master Replication Manager**  
<http://code.google.com/p/mysql-master-master/>
- **Maatkit**  
<http://maatkit.sourceforge.net/>
- **Mon – scheduler and alert management**  
<http://www.kernel.org/software/mon/>
- **Continuent Tungsten Replicator**  
<https://community.continuent.com/community/tungsten-replicator>

Q & A

Questions, Comments?

Thank you!  
Lenz Grimmer <lenz@grimmer.com>