

Galera Replication

codership

Seppo Jaakola, CEO

seppo.jaakola@codership.com

<http://www.codership.com>

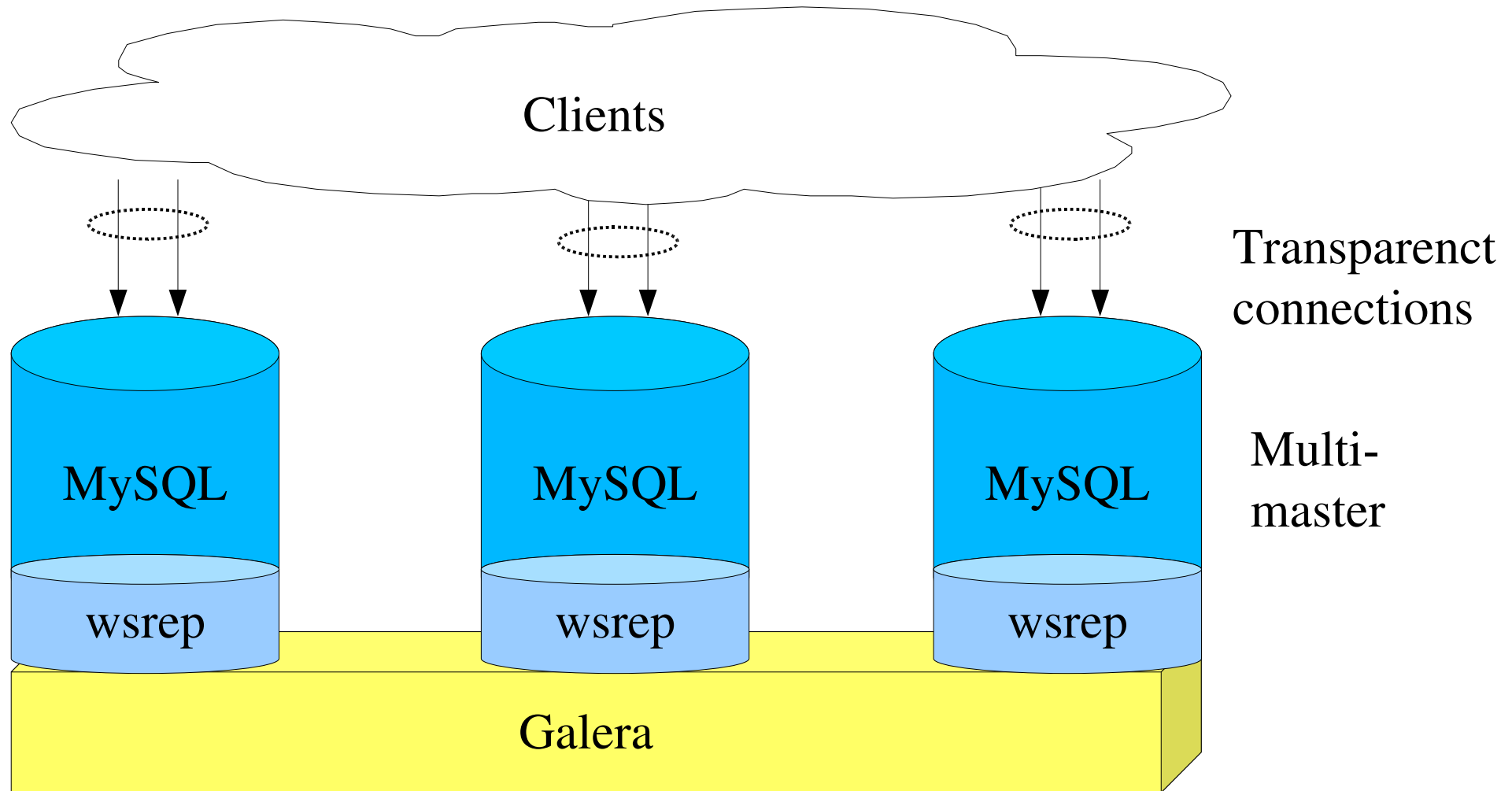
Contents

1. Galera Overview
 - Cluster architecture
 - Wsrep API & MySQL integration
 - Advanced Features
2. Benchmarking
 - Sysbench
 - Dbt2 results
3. Drupal Benchmark
 - EC2 Installation
 - Jmeter test
 - Issues
 - Results
4. Optimal Drupal Cluster
5. Caching Issues
6. Summary

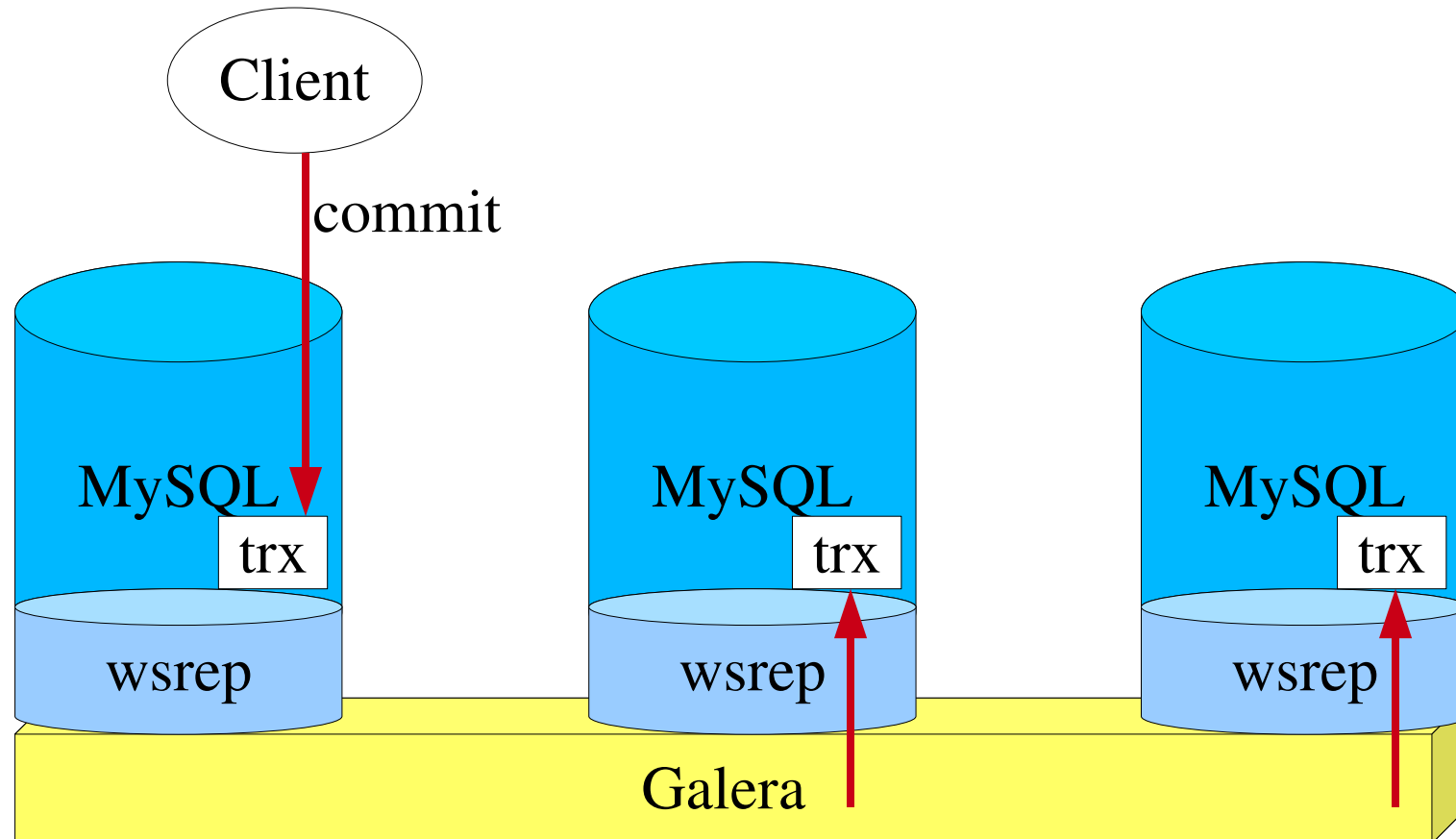
Galera Replication

- Multi-master synchronous replication system
- Certification based replication model (based on academic research by F. Pedone et al)
- Avoids using middle-ware, connections go directly to DBMS -> transparency
- Row level locking -> write scalability
- Generic replication system to make a cluster from any transactional DBMS
- First implementation MySQL/InnoDB cluster

Galera Cluster

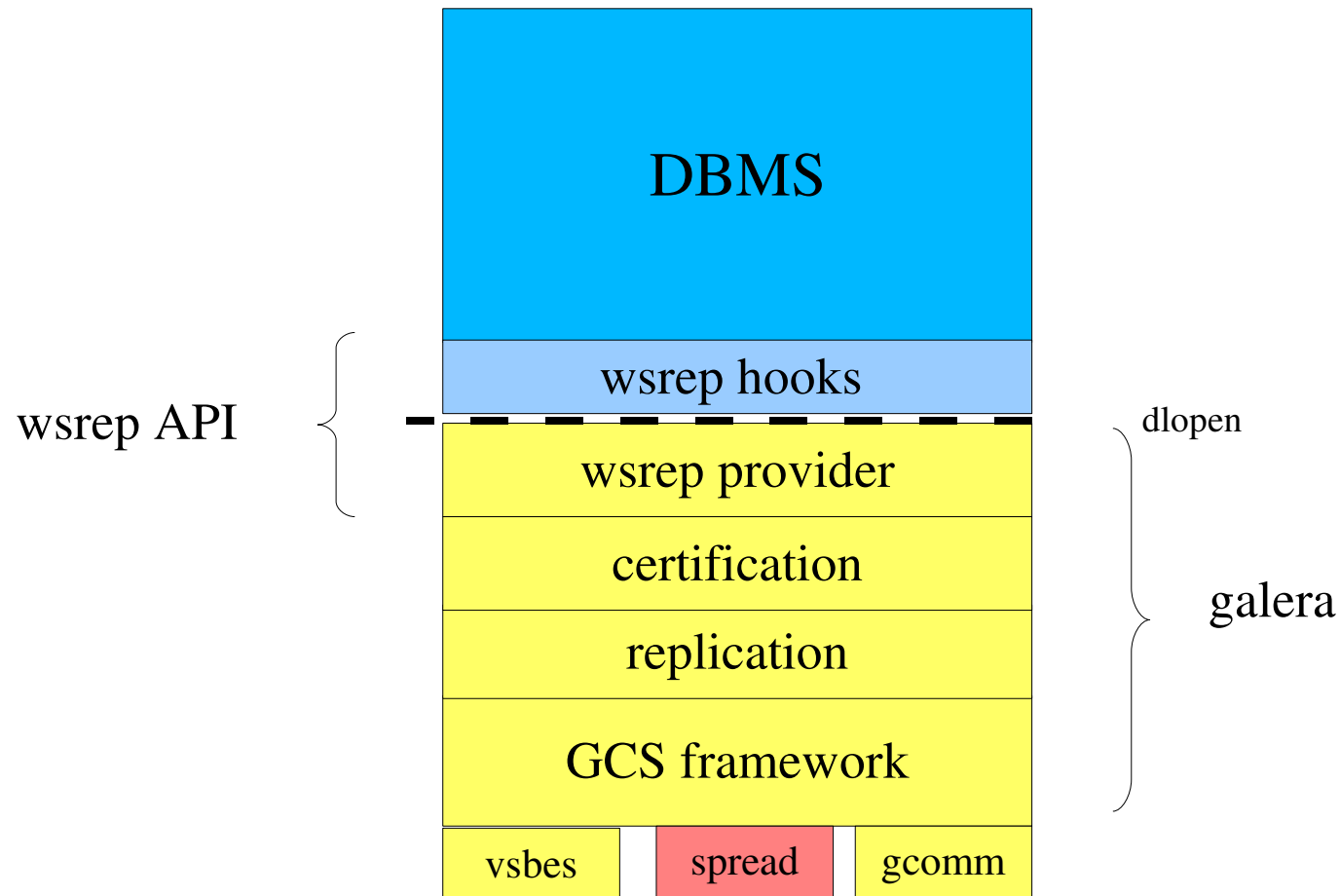


Synchronous Replication



Transaction is committed in all nodes => HA

Generic Replication





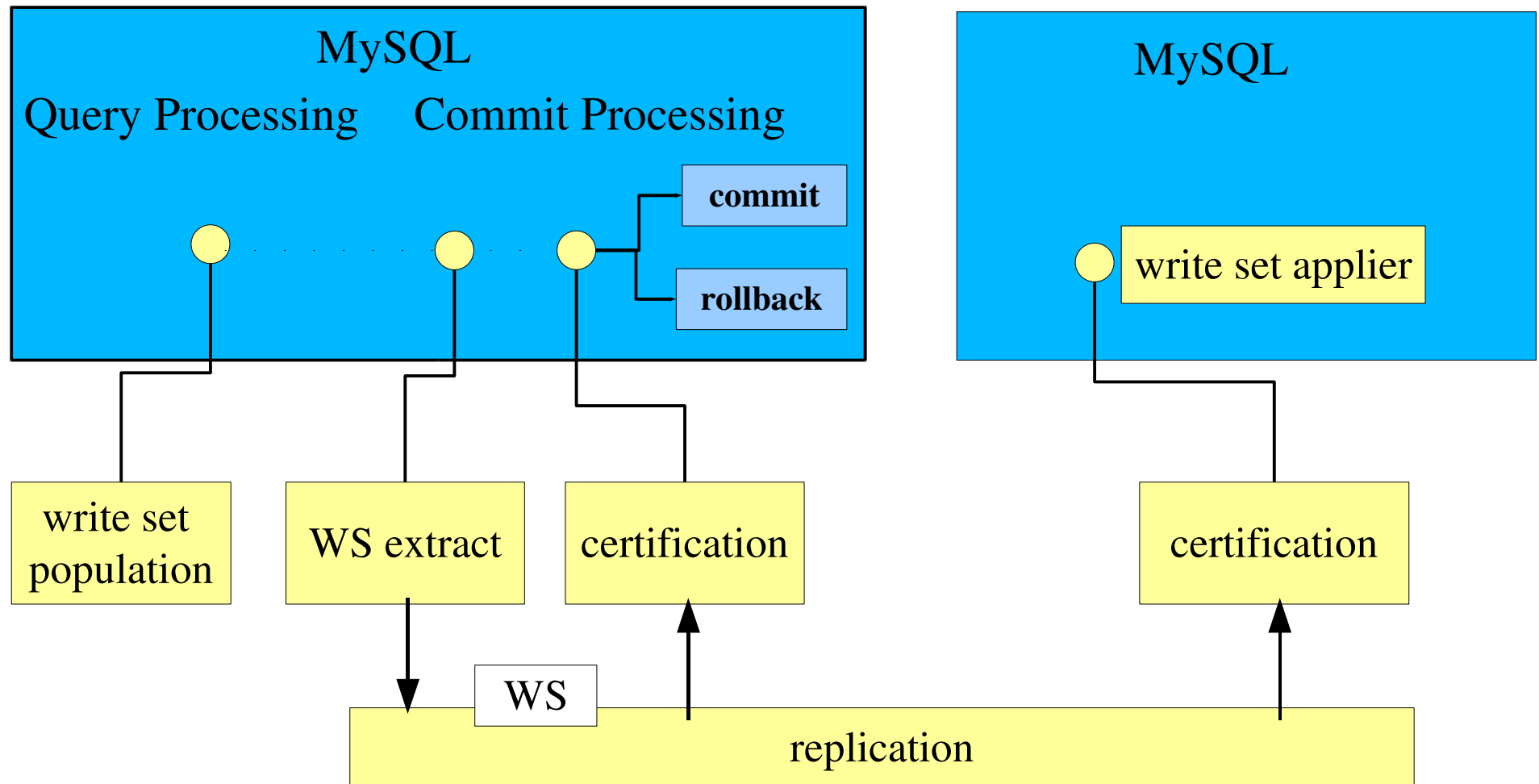
wsrep API

- Defines a generic interface for DBMS and replication system
- Write set replication API for transactions
- DDL replication using TO isolation
- Launchpad project: <https://launchpad.net/wsrep>

wsrep integration in MySQL

- Launchpad project:
<https://launchpad.net/codership-mysql>
- Calls to wsrep provider:
 - Ws populating, replication...
- Handlers for various wsrep callbacks:
 - ws applying, DDL applying ...
- Changes in innodb code to provide **prioritized transactions**

Certification Based Replication



Write Set

- Write set can contain data changes specified in different replication levels:
 - 1.SQL statement**
 - 2.Lex structures (AST) from parser
 - 3.RBR event**
 - 4.Row (as binary image)
- All row changes are identified with keys
- Last_seen_seqno & seqno tracking trx processing state
- Write sets can be saved for future needs

Advanced Replication Features

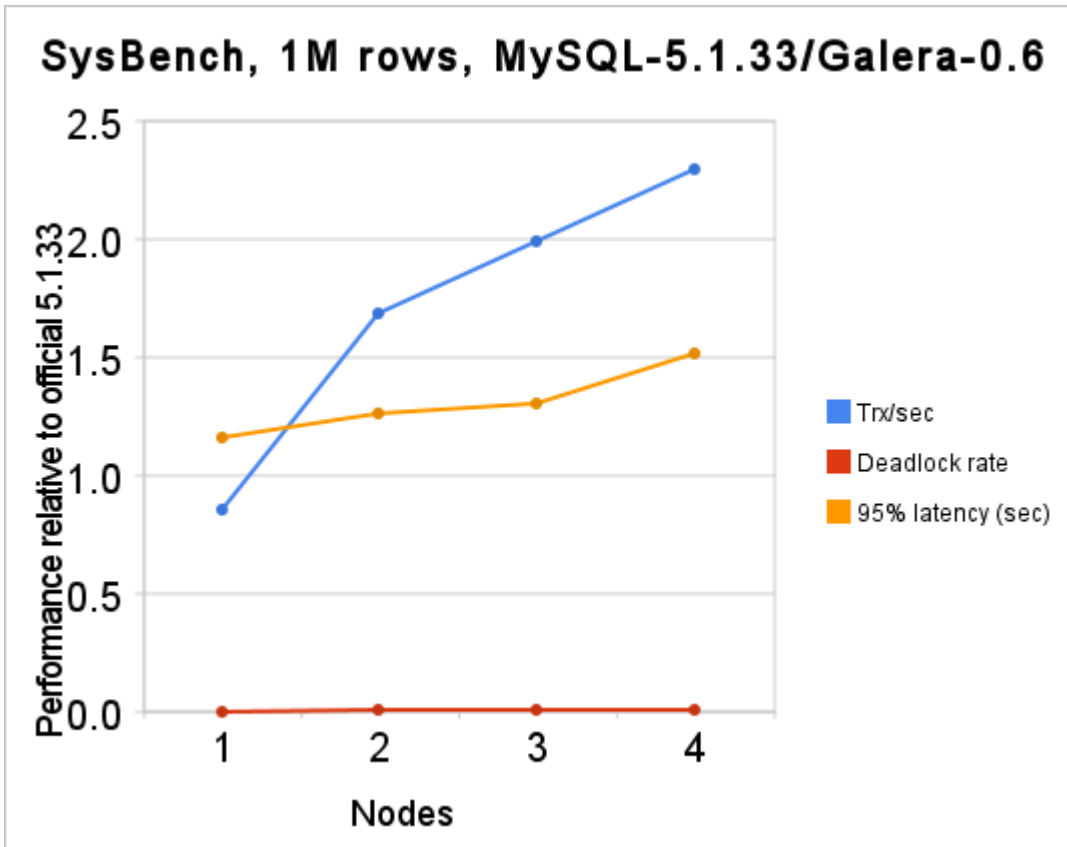
- Replication is optimistic in nature
 - Hot spots cause replication aborts
- Flow control
 - GCS feature to adjust nodes' progress
- Autoincrement management
 - Cluster adjusts increments and offsets on the fly
- Asymmetric lock granularity issue
 - Solved by replaying as slave trx
- Retrying of aborted autocommit trxs

Benchmarking

Benchmarking

- Tested with several benchmarks
 - Sysbench, dbt2, DOTS, osdb, jmeter, sqlgen...
- Benchmarks testing with 'physical hardware' and with Amazon EC2 small and large instances
- Currently tests only up to 5 cluster nodes
- In general, shows good scalability even with write intensive work loads

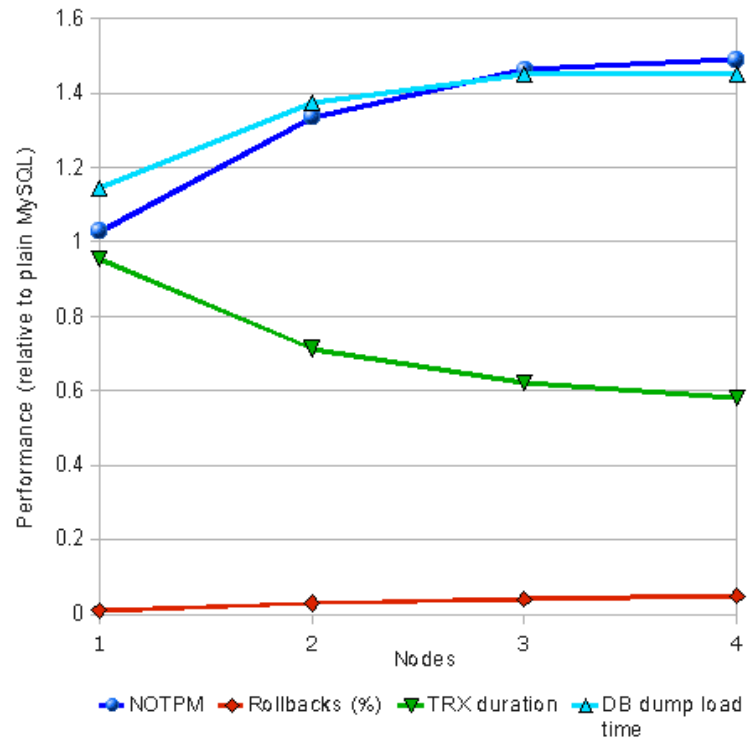
Sysbench Benchmarks



- Sysbench oltp mode test
- EC2 Large instances

nodes	users	trx/s	deadlks	95%lat
1	18	385	0	0.092
2	36	761	2.54	0.100
3	45	900	3.42	0.103
4	60	1034	4.54	0.120
official 5.1.33 binary:				
1	18	451	0	0.079

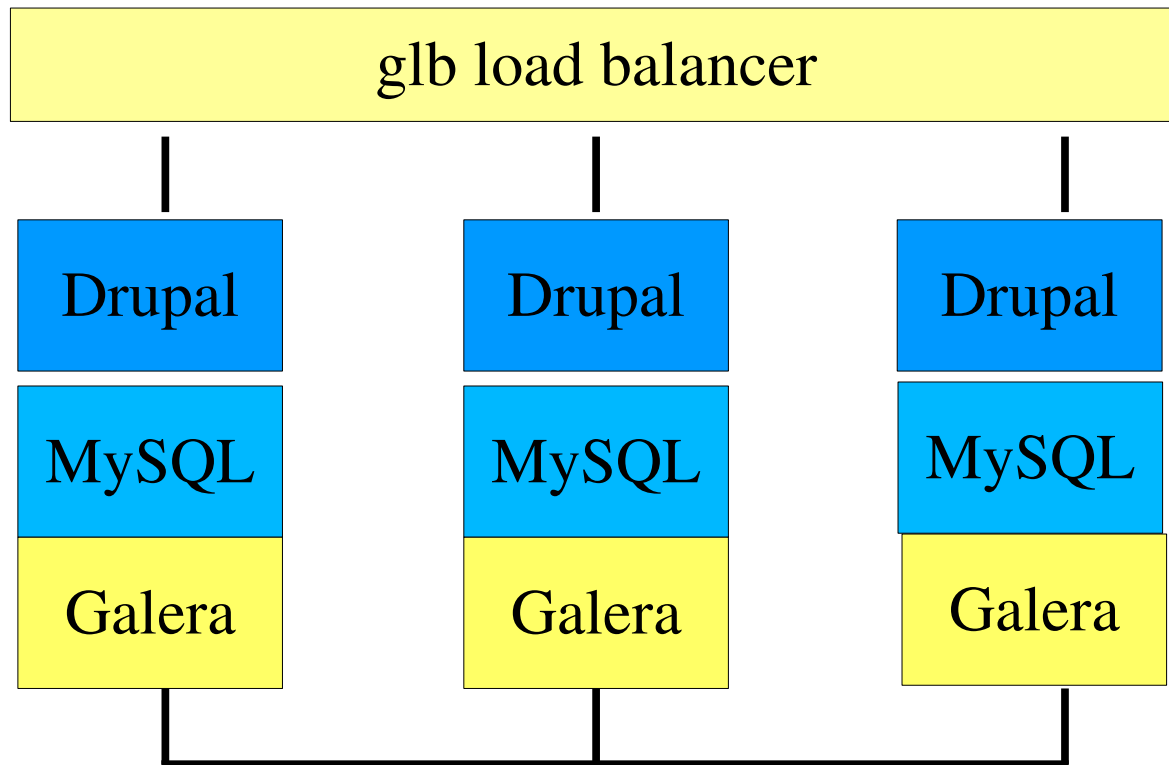
Dbt2 Benchmark



- EC2 large instances
- Dbt2 benchmark
- 60 warehouses

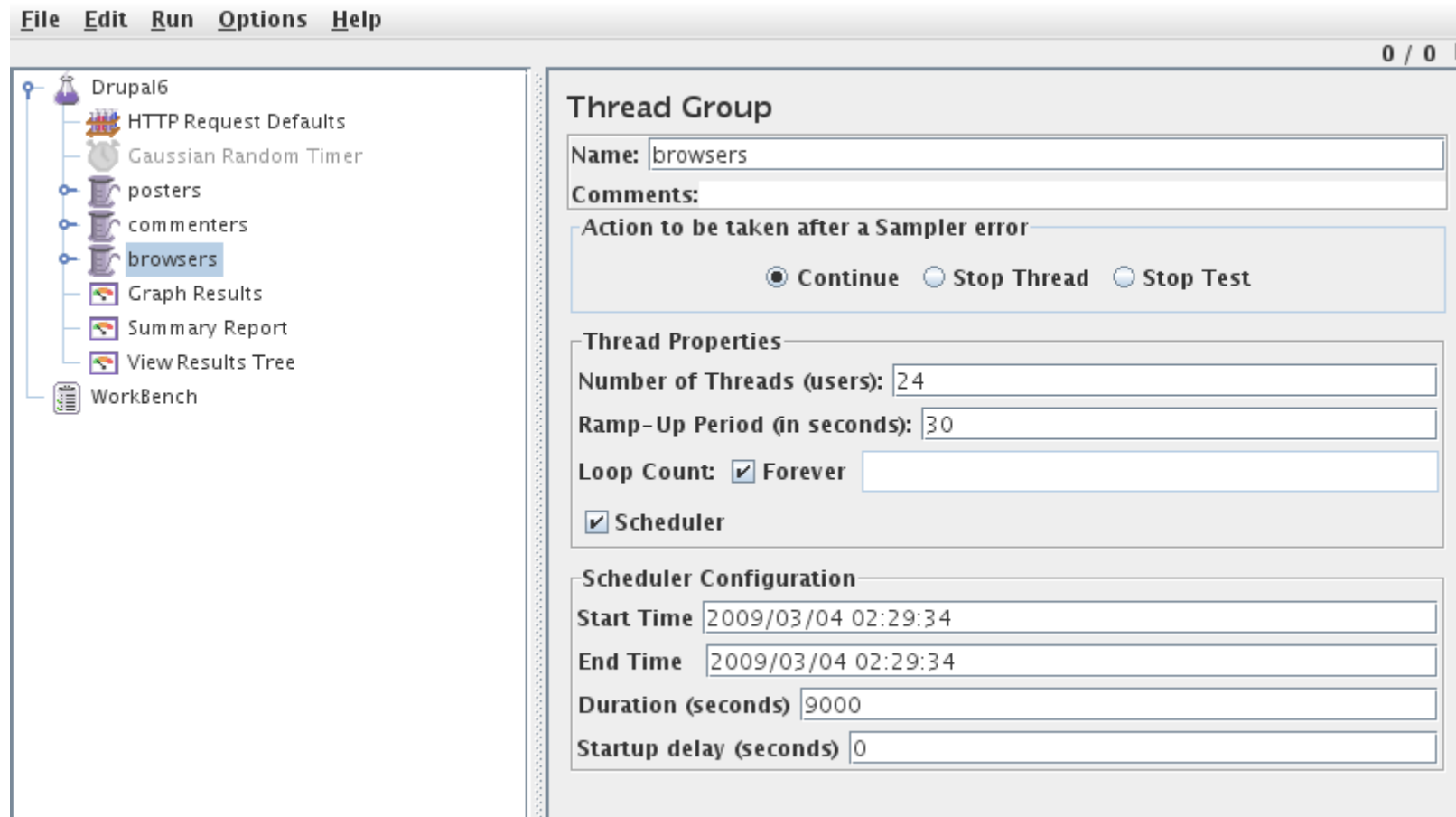
	Conns	NOTPM	Rollbacks(%)	TRX duration(sec)	Dump load(min)
Plain 5.1.30:	20	~7220	1	2.27	26
1 node	: 12	~7420	1	2.17	30
2 nodes	: 24	~9630	3	1.63	36
3 nodes	: 36	~10555	4	1.41	38
4 nodes	: 48	~10753	5	1.32	38

Drupal Scale-Out



- Proof of concept
- Each drupal node has local MySQL
- all nodes identical
- ~10% of CPU for MySQL
- glb load balancer

The Test



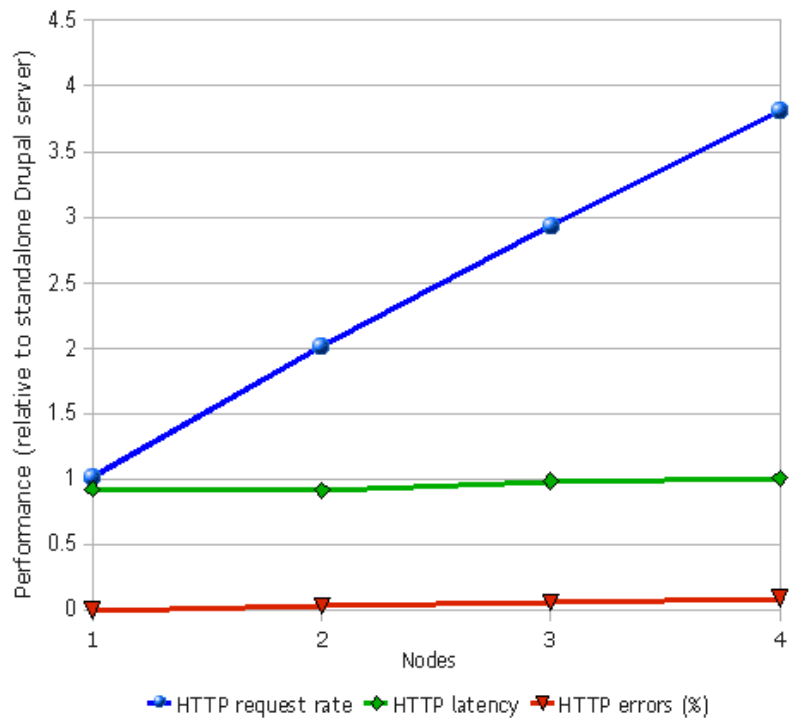
The Test

- Jmeter thread groups:
 - Posters write new pages
 - Commenters read pages and add comments
 - Browsers read pages and comments
- Threads created in proportion: 4/12/24
- Write intensive work load

Issues

- Drupal Cache must be off
- 'files' directory contents
 - use network file system
 - store all files in DB
- Concurrent login failures
- Autoinc insert bug:
<http://drupal.org/node/282555>
 - Sorted out by a workaround to retry failed autoinc insert

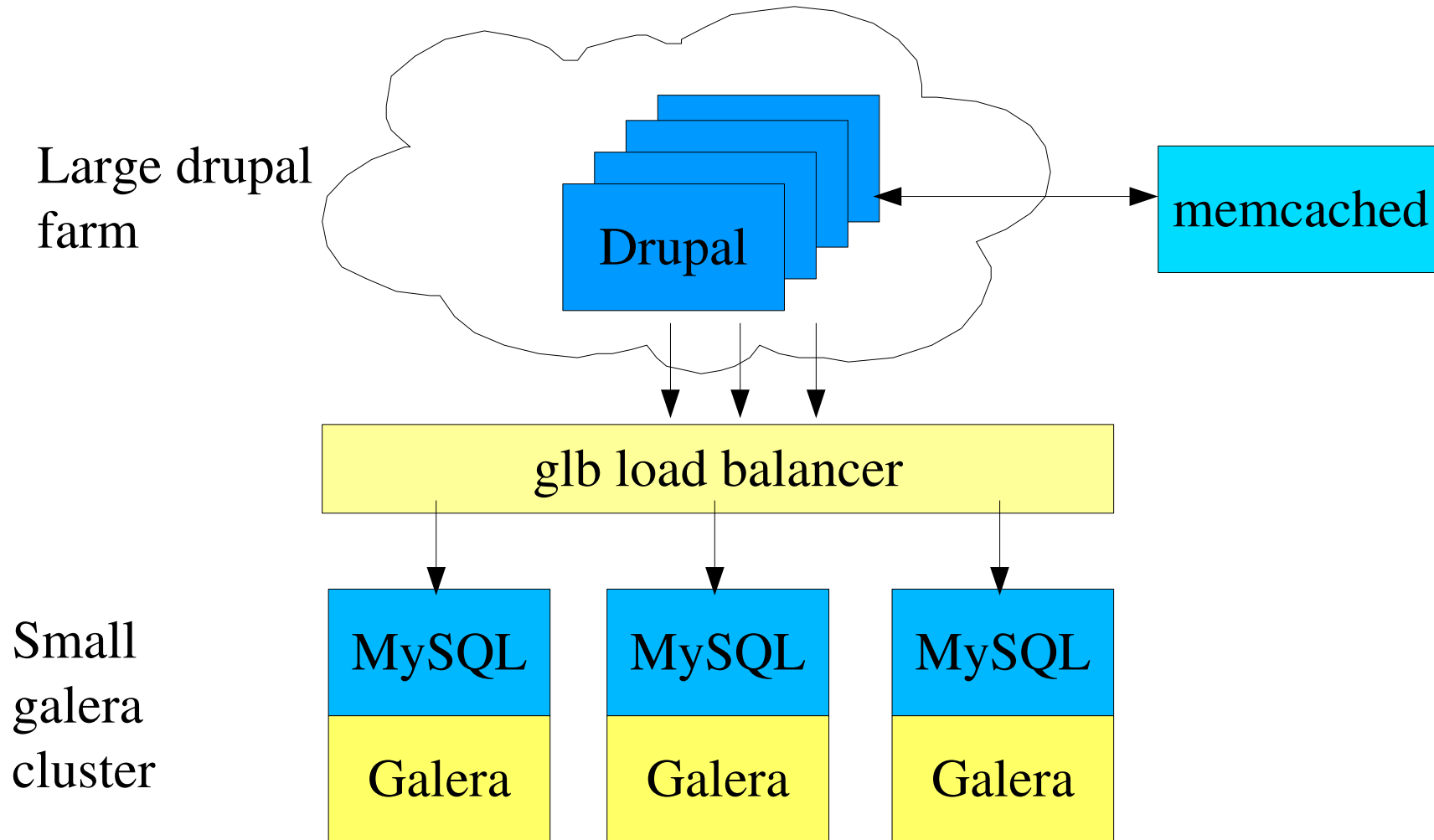
Results



Nodes	Users	Throughput (req/min)	Latency (ms, median)	Latency (ms, average)	Errors (%)
1	180	724	1203	1827	0.00
2	360	1436	1190	1829	0.03
3	540	2091	1280	2150	0.06
4	720	2717	1214	2330	0.12

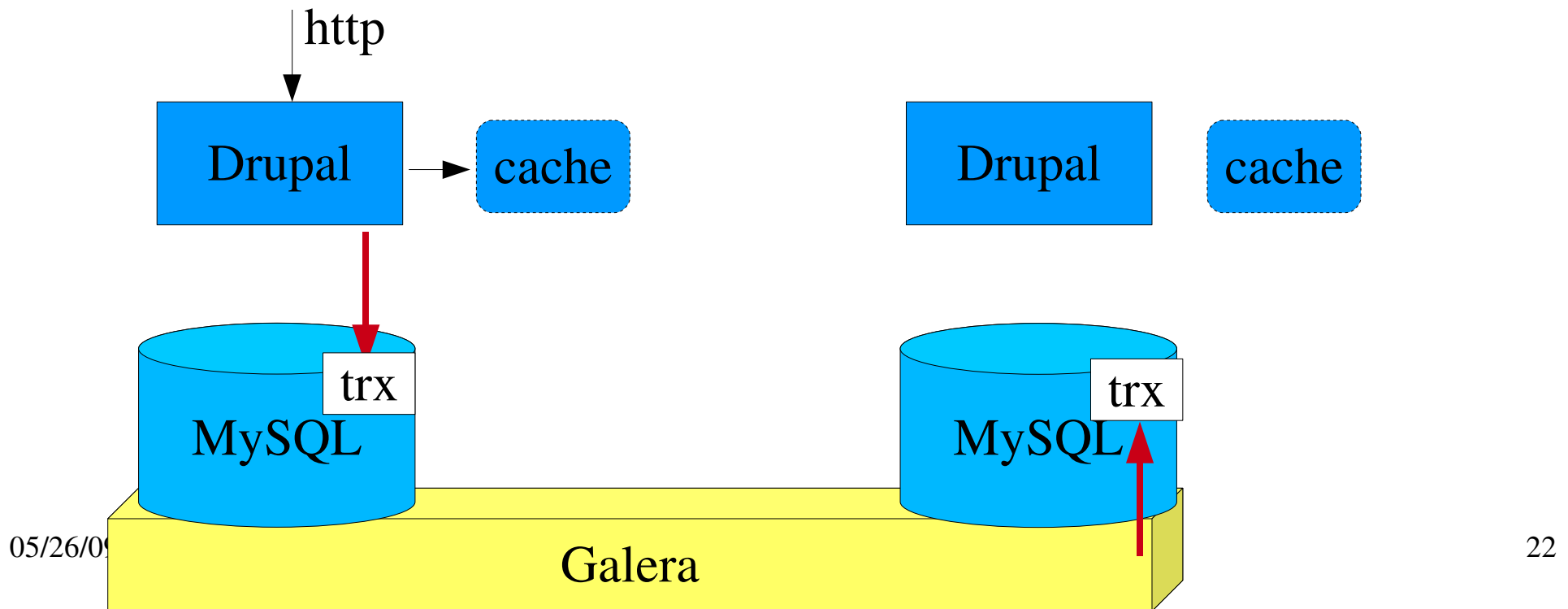
- <http://www.codership.com/content/scaling-drupal-stack-galera-part-2-mystery-failed-login>

The Way to Do It



About Caching

- ✓ MySQL query cache is good to use
- ✓ External cache (memcached) is good also
- ✗ Drupal Cache cannot be used



Summary

- High Availability, transparency
- Good scalability even with high write rates
- Roadmap:
 - Release 0.6 available for download
 - Good for evaluation
 - Release 0.7 scheduled for June
 - Galera Library Open Sourced
 - DB state transfer during node join
 - New group communication

Get in Touch!

codership

- Downloads available: <http://www.codership.com>
- R&D consulting services
- Evaluation support
- info@codership.com