Using MySQL for Distributed Database Architectures

Peter Zaitsev
CEO, Percona
PingCAP Infra Meetup,
Shanghai, China, May 26, 2018

PERCONA

About Percona

- Solutions for your success with MySQL,MariaDB, MongoDB and PostgreSQL
- Support, Managed Services, Consulting, Training, Software
- Our Software is 100% Free and Open Source
- Support Broad Ecosystem – MySQL, MariaDB, Amazon RDS, Google CloudSQL
- In Business for 11 years
- More than 3000 customers, including top Internet companies and enterprises

Presentation

Cover Basics

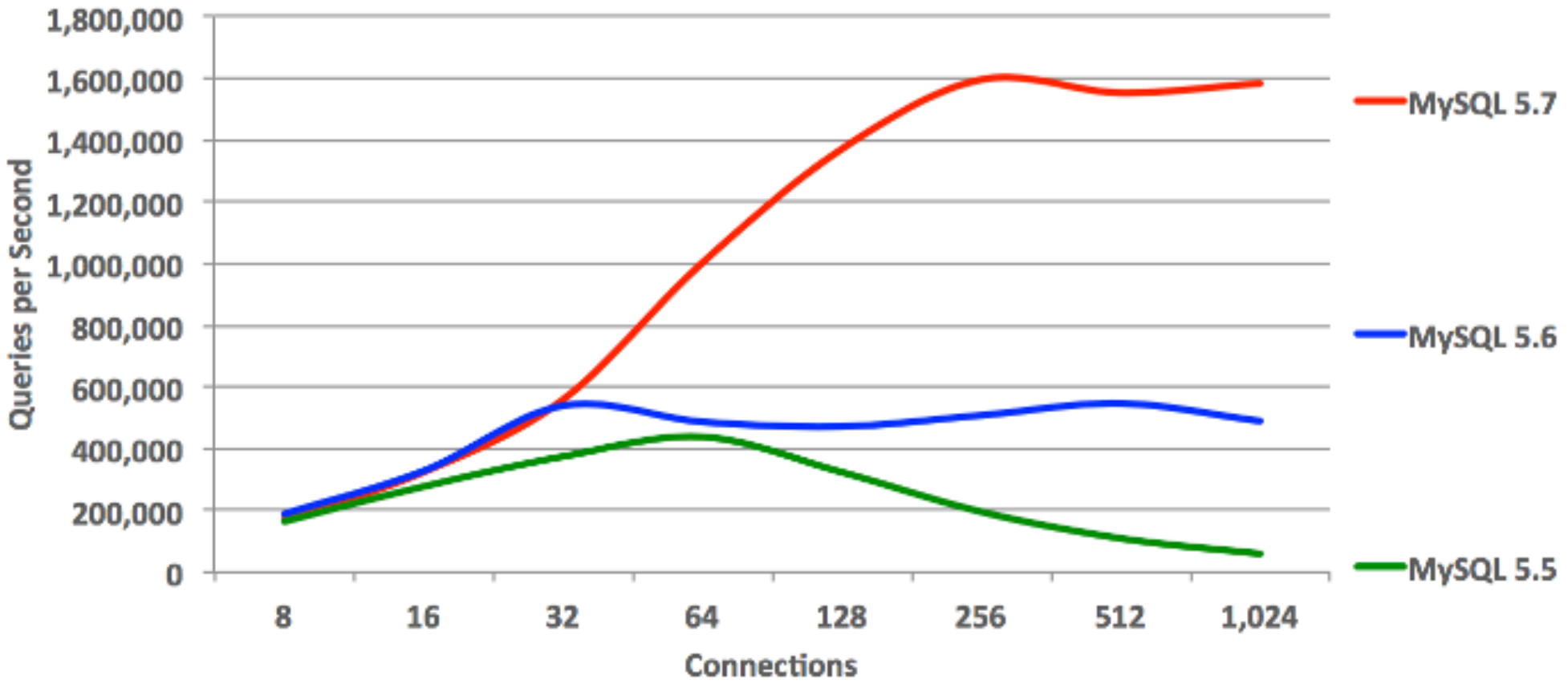
Why Going Distributed

How to do it

Distributed ?

MySQL Deployment on More than one System

Modern MySQL Scalability



PERCONA

Single MySQL Instance Can Do

Hundreds of Thousands of Queries/Sec

Tends of Thousands of Updates/Sec

Traverse Tens of Millions of Rows/Sec

Comfortably Handle Several TB Database size

Lets Do Some Math

100.000 QPS

10 Queries per User Interaction

10.000 User Interactions/sec

864.000.000 User Interactions/Day

30 User Interactions/User Avg

28.000.000 Daily Active Users Possible

15M of Daily Active Users counting time of day skew

Distributed Systems Tend To be

More Complicated to Develop Against

More Complicated to Operate

Have Additional Performance Bottlenecks

Have Complicated Failure Modes

With All of this ?

Why “Go Distributed” ?

Reasons to “Go Distributed”

High Availability

Scalability

Data Distribution

High Availability with MySQL

Cold Standby (ie DRBD)

Failover (Classical Replication)

Active-Active Clustering (PXC, MySQL Group Replication)

Q1:What Failure Modes Do you Consider ?

Server Crash/Server Hardware Failure

Software Bugs and Storage Corruption

Network Failure

“Datacenter” Failure

Developer Mistakes/Intruder Actions

Q2:What Data Am I allowed to lose ?

Limited Data Loss Allowed

No Data Loss

No Transactions In Flight Loss

Q3:How Quickly do you need to Recover ?

“Immediate?”

“Seconds?”

“Minutes?”

“Hours?”

Speed of Light Realities

Data Propagation Can Be

- Synchronous – Slow
- Asynchronous – Data Loss Can Occur

Scalability

Scaling Reads

Scaling Writes

Scaling Data Size

Data Distribution

Some Data Must be in Specific Geographic Location

- User Latency Reasons
- Legal and Compliance Reasons

Distributed Architectures with MySQL

Main Concepts

Replication

Sharding

Failover Management

Traffic Management

Replication

Having Multiple Copies of the data, updated with changes

Availability

Service Stays up when component fails

Availability via Redundancy

Have more than one system

Works well for stateless systems

Is not enough for databases

Availability via Replication

Redundant Computing Resource

Paired with Replicated Data

Where Replication Happens

Storage Level

Database Level

Application Level

Storage Level Replication

Replication in SAN/NAS, DRBD

Typically provides cold standby

Simple choice which works with many systems

Amazon Aurora – Smart Storage

Database level

Most Flexible

Most Common

Hot/Warm Spare

Some can do Active-Active

Application Level

Hard to get right

Rarely used, especially “used right”

Partial Replication/Syncronization

Smart conflict resolution

Cross Vendor Redundancy

Sharding

Split data set by certain criteria and store such “shards” on separate “clusters”

Typical Sharding

By User

By Customer Account/Company

Sharding and Replication

Sharding Almost Always used with some kind of Replication

Failover Management

Many Replication Modes Require “Failover”

Need some Tool To Decide on Failover

Need some Tool to Perform Failover

Traffic Management

- Scaling with Large Number of Connections
- Routing Traffic to Right “Shard”
- Read-Write Splitting
- Load Management
- Avoiding “Dead” Nodes

Options in MySQL

Replication Options in MySQL

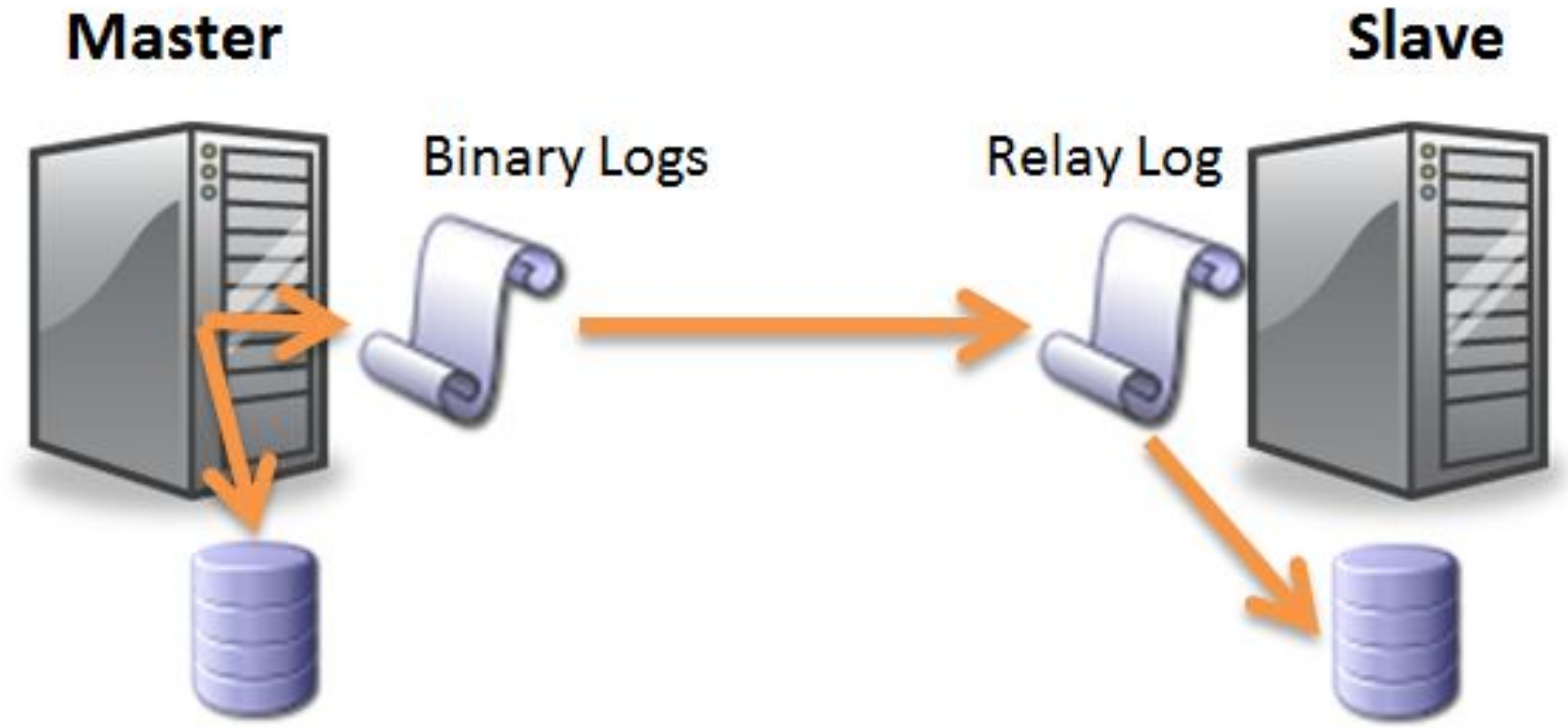
Classical MySQL Replication

MySQL Group Replication

Percona XtraDB Cluster and Galera

MySQL (NDB) Cluster

Classical MySQL Replication



Classical MySQL Replication Properties

Asynchronous or Semi-Synchronous

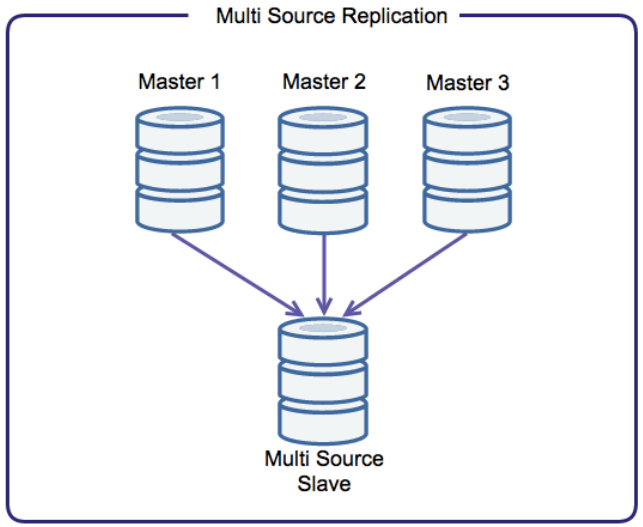
Parallel (since MySQL 5.7)

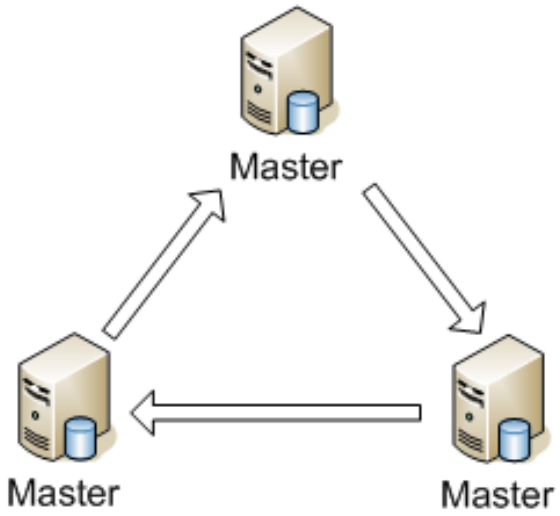
Many Masters to Many Slaves (MySQL 5.7)

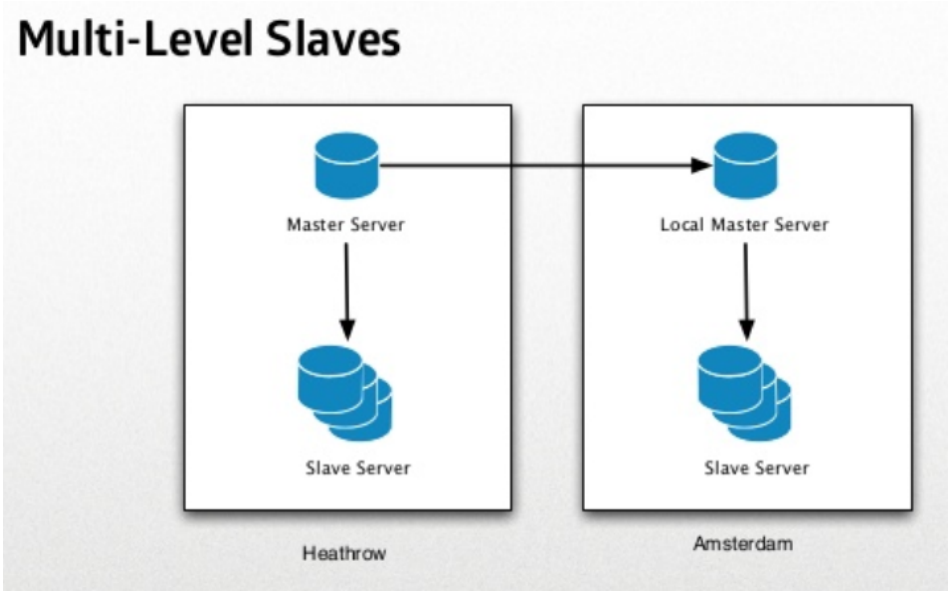
No Conflict Resolution or “Protection”

No Built-in Failover

Advanced MySQL Replication Topologies

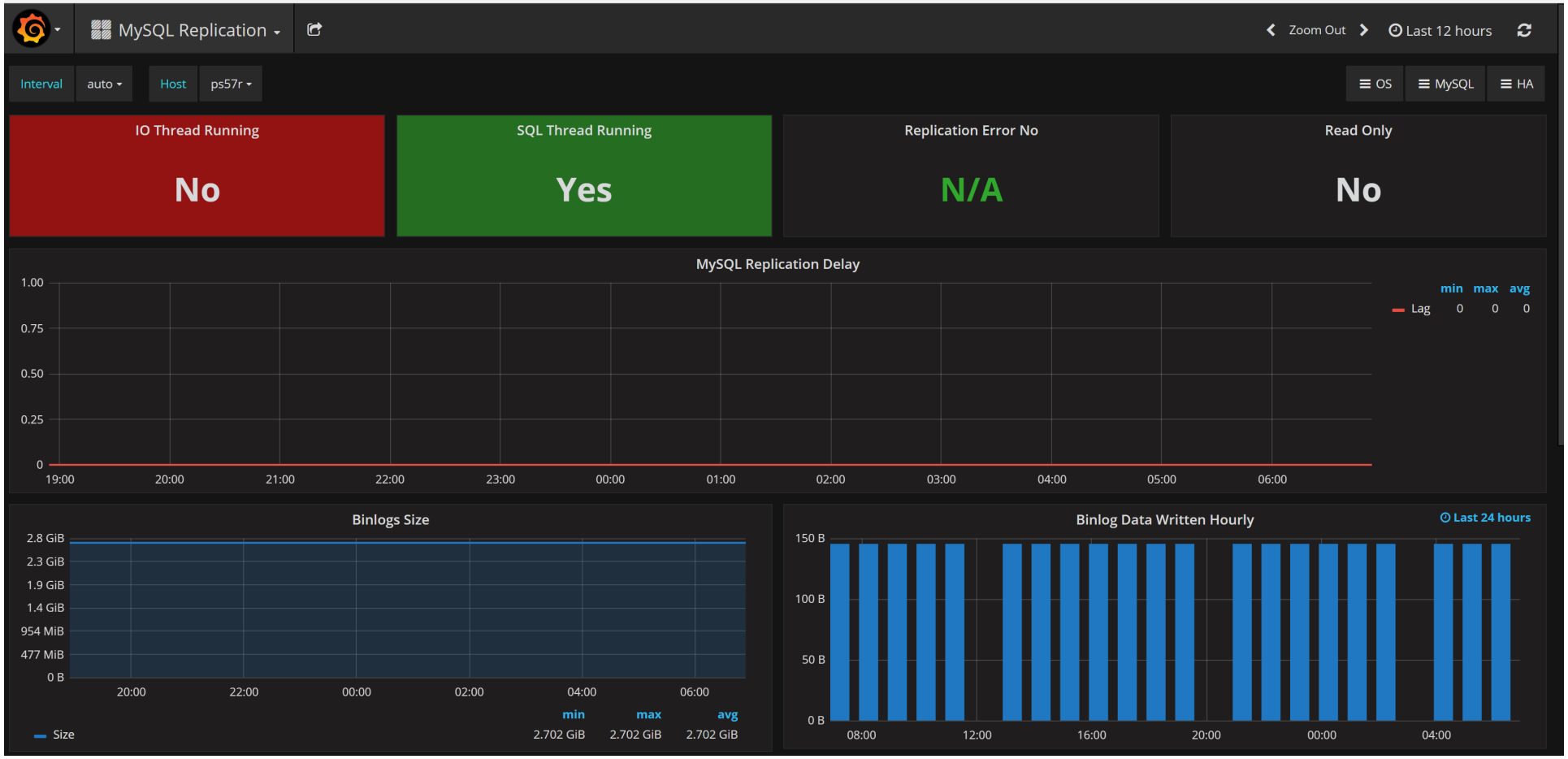






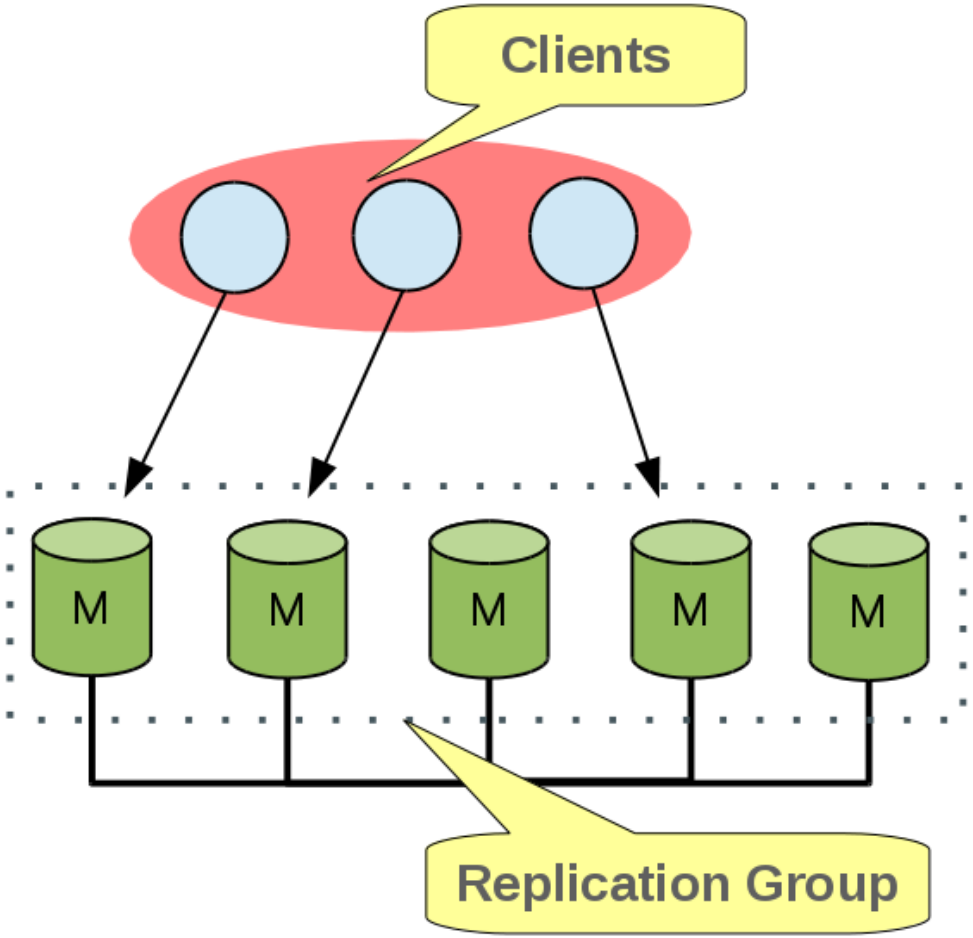
PERCONA

PMM Dashboard for Replication



PERCONA

MySQL Group Replication



MySQL Group Replication (New in 5.7)

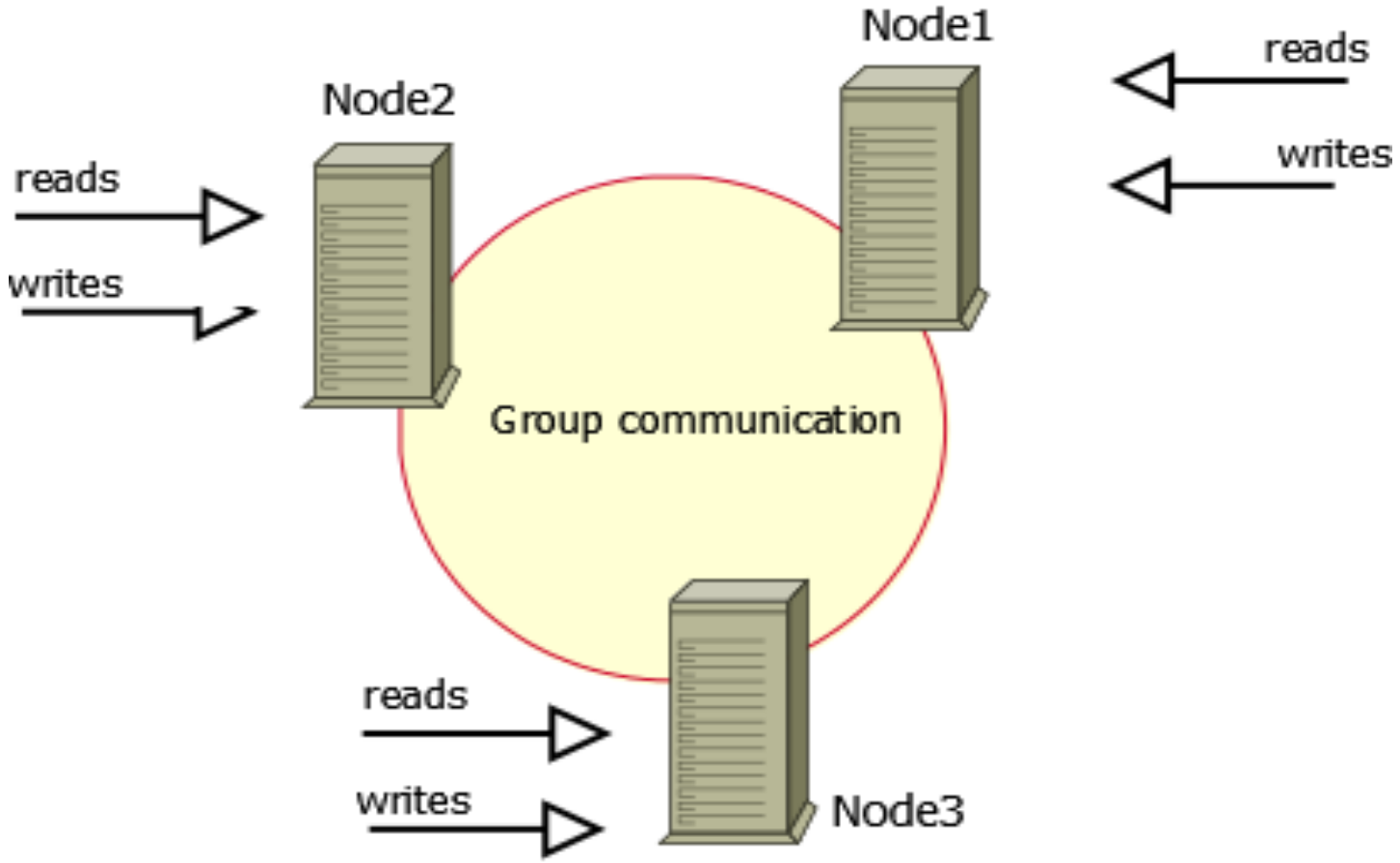
- “Group of Peers”
- Write-Anywhere or Dedicated Writer
- Asynchronous Replication with Flow Control
- Conflicts Prevented through Certification
- Built in Failover
- No Automated Provisioning

Percona XtraDB Cluster/Galera

PERCONA

XtraDB Cluster

Percona XtraDB Cluster Topology

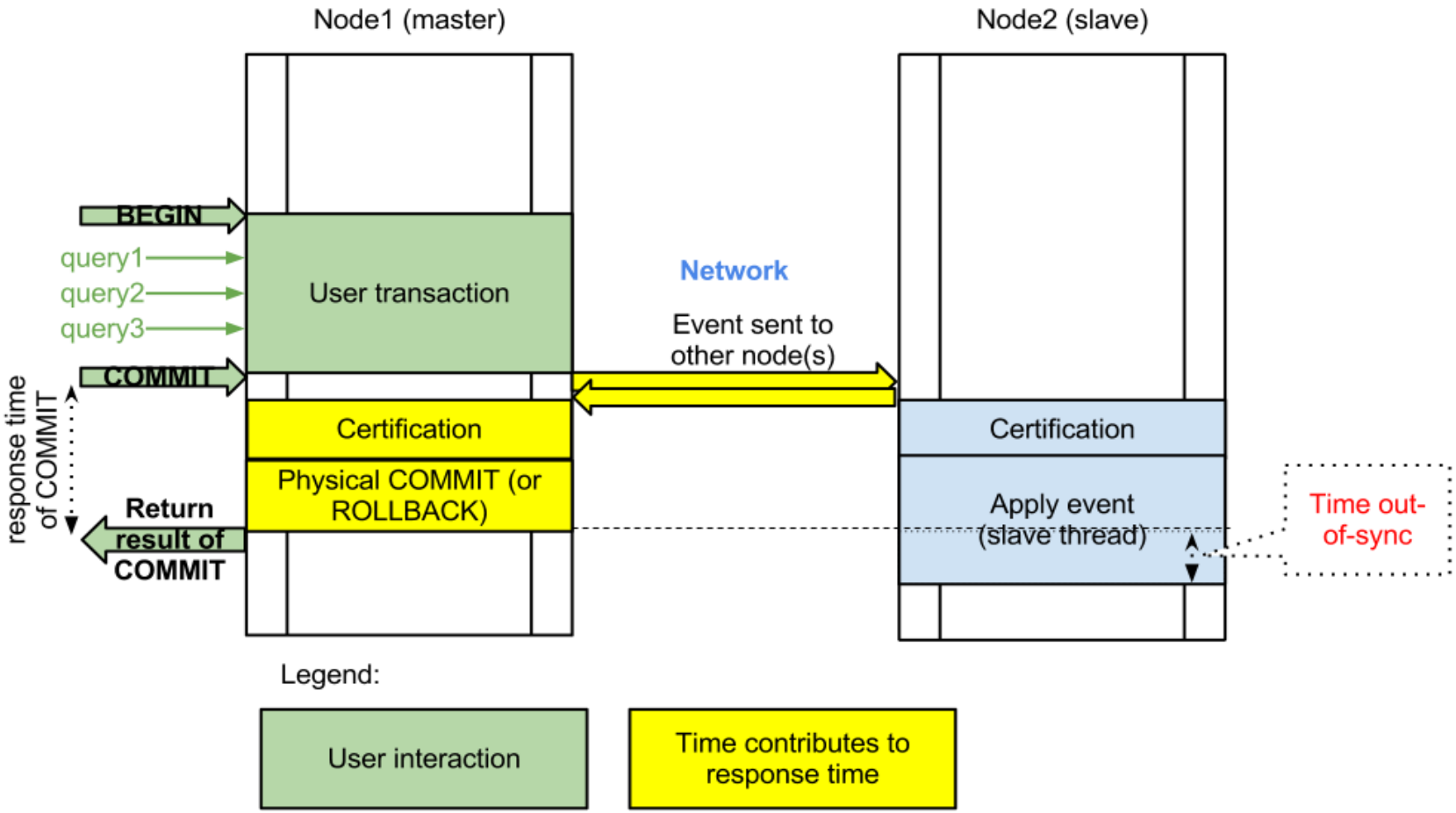


PERCONA

PXC/Galera Properties

- Write to Any Node
- Certification Based Replication
- Virtually Synchronous; Can ensure no stale reads
- Built in Fail-Over
- Built-in Node Provisioning

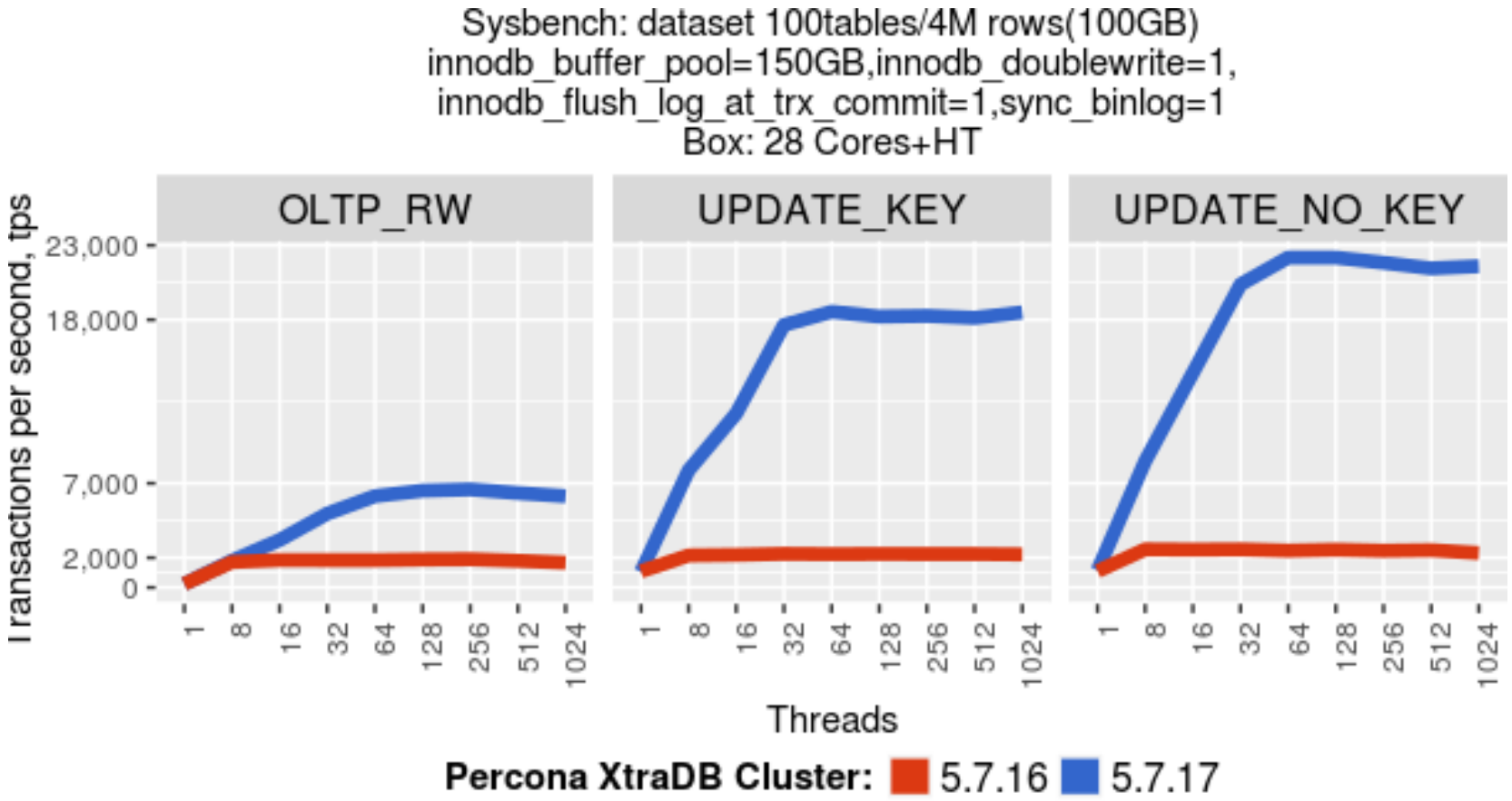
Transaction Commit Flow



PERCONA

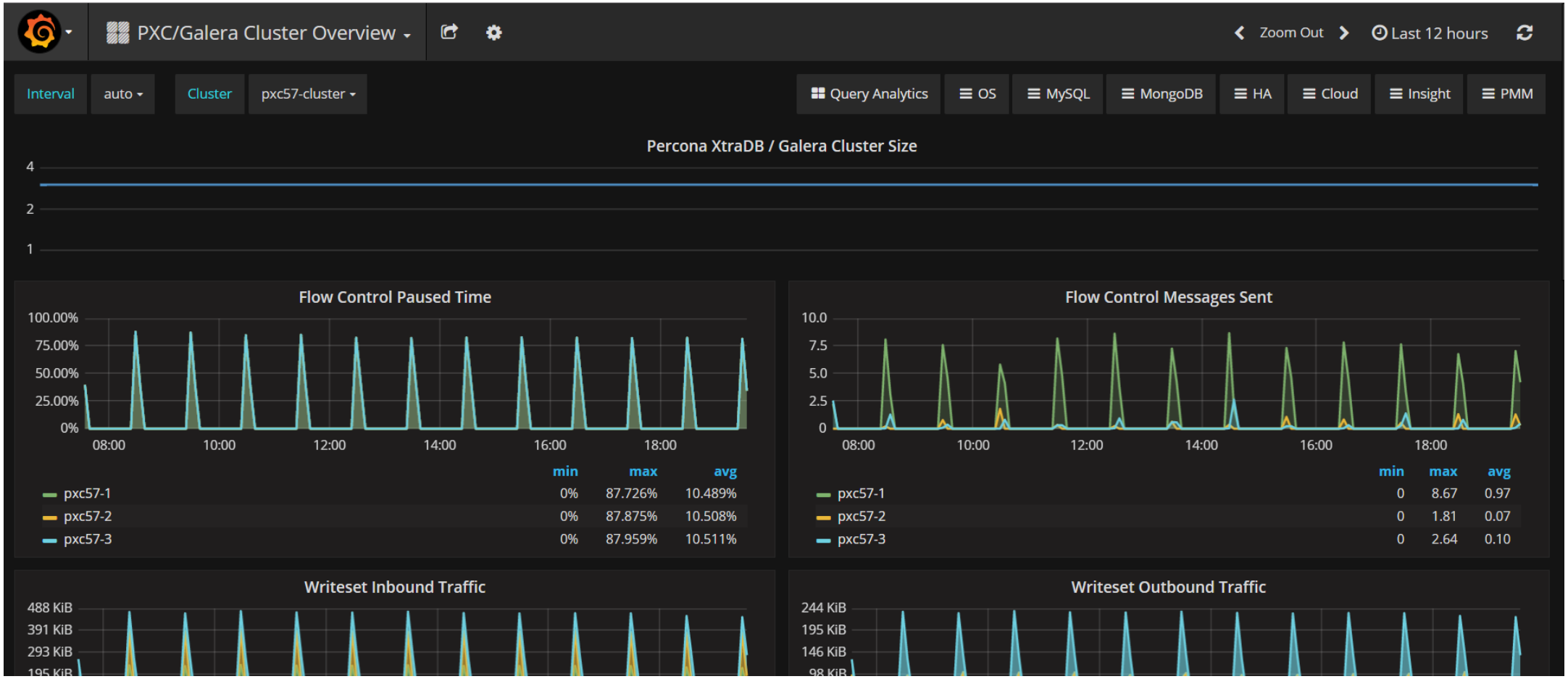
PXC 5.7 Performance Improvements

http://bit.ly/2qGCr0T and http://bit.ly/2pzvAIW



PERCONA

PMM Dashboard for PXC



PERCONA

MySQL Cluster

Mostly In Memory Storage

Synchronous Replication

Pessimistic Locking

Conflict Detection with Async Replication

Niche Use Only

Sharding in MySQL

Typically Painful or Very Painful

No Built-In or Standard Solution

Many Custom App-Layer Implementation Exists

Thinking about Sharding

Query Routing to the right Shard

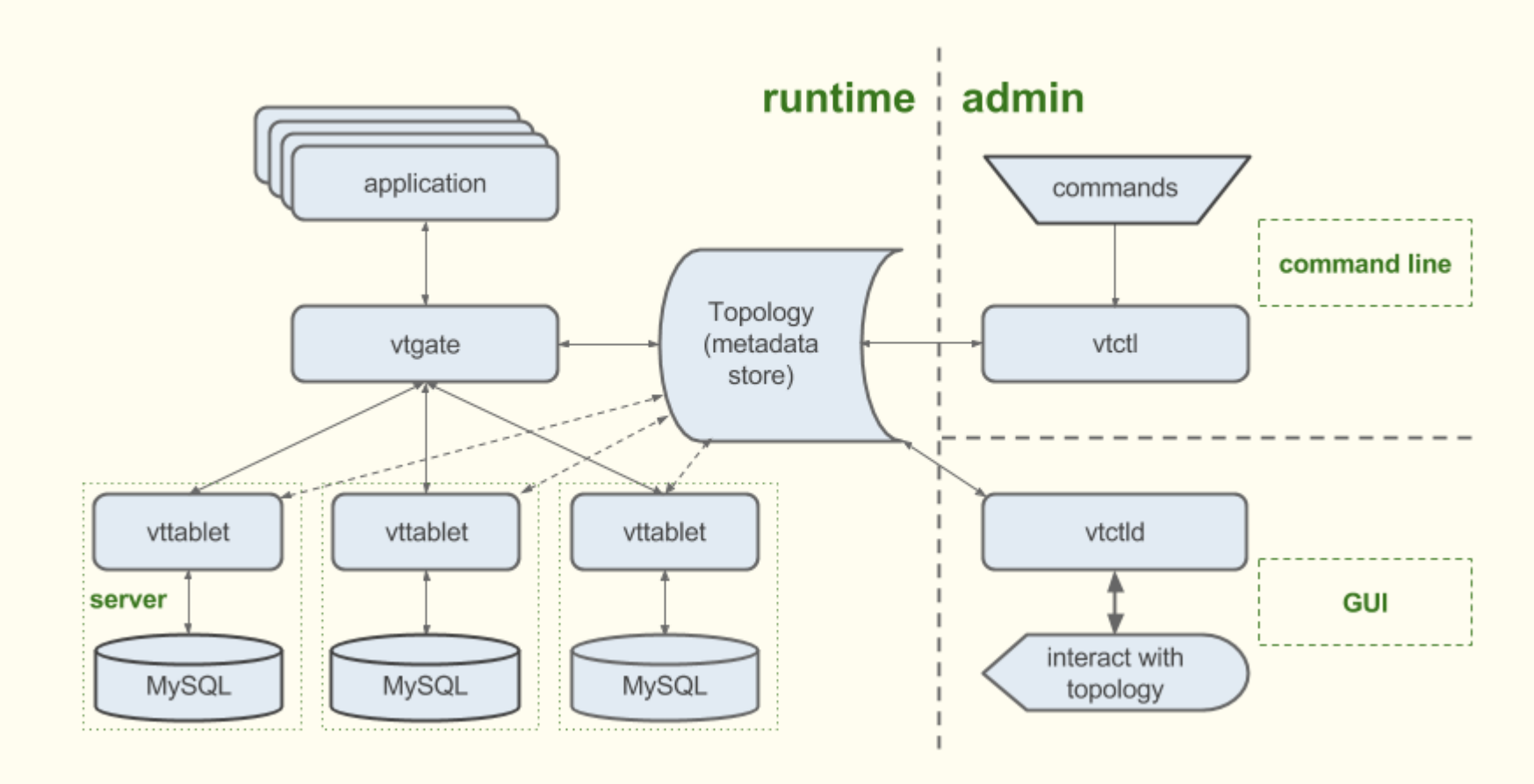
Data Aggregation for Cross-Shard Queries

Shard Balancing

Vitess

- Stated by team in YouTube
- Now Open Source in CNCF
- Deployed at Slack
- Supports MySQL Protocol
- Supports Cross Shard Queries
- Supports Re-Sharding

Vitess Architecture



PERCONA

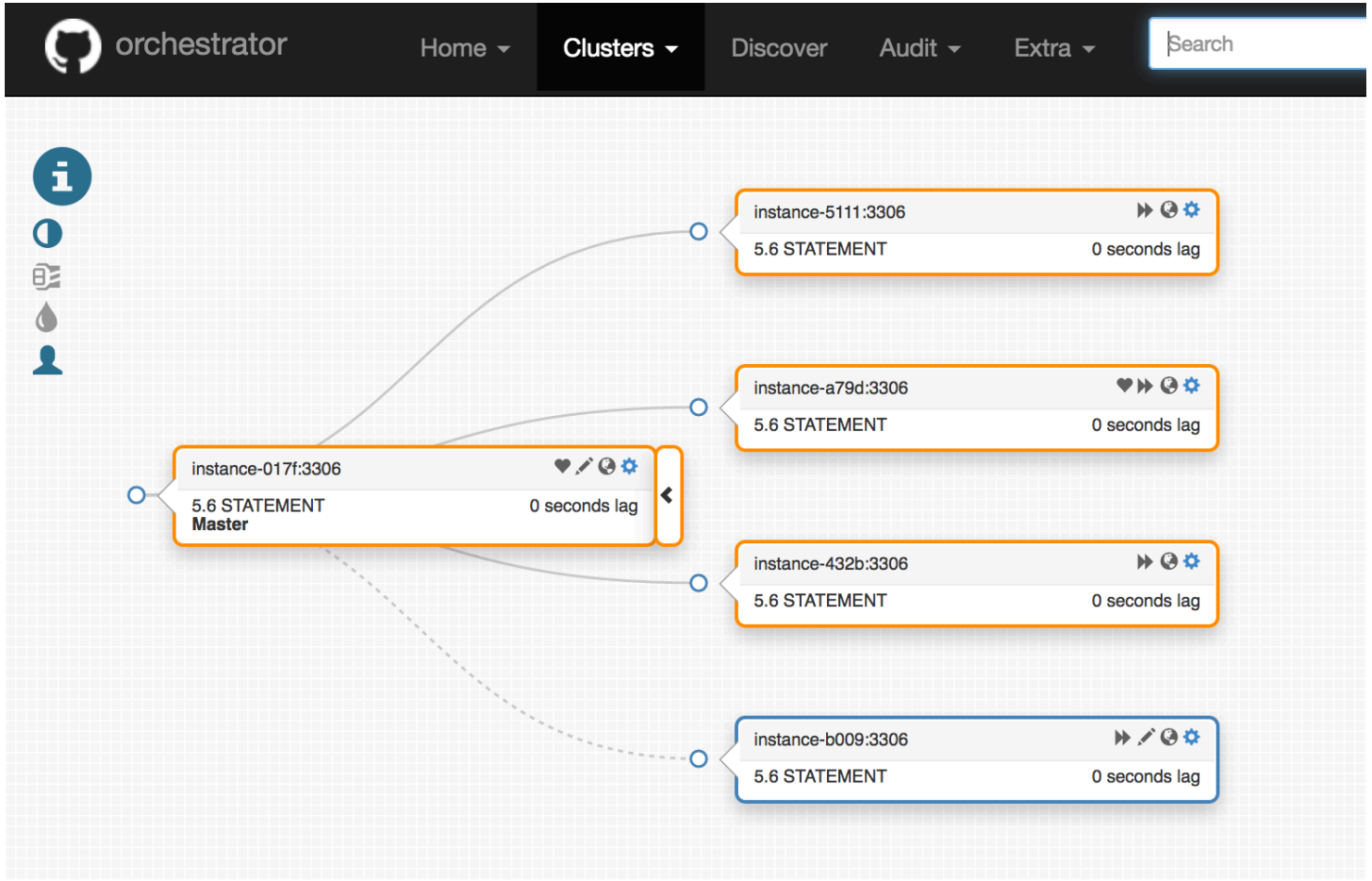
Failover in MySQL

MHA

MySQL Failover

Orchestrator

Orchestrator Screenshot



PERCONA

Traffic Management Solutions

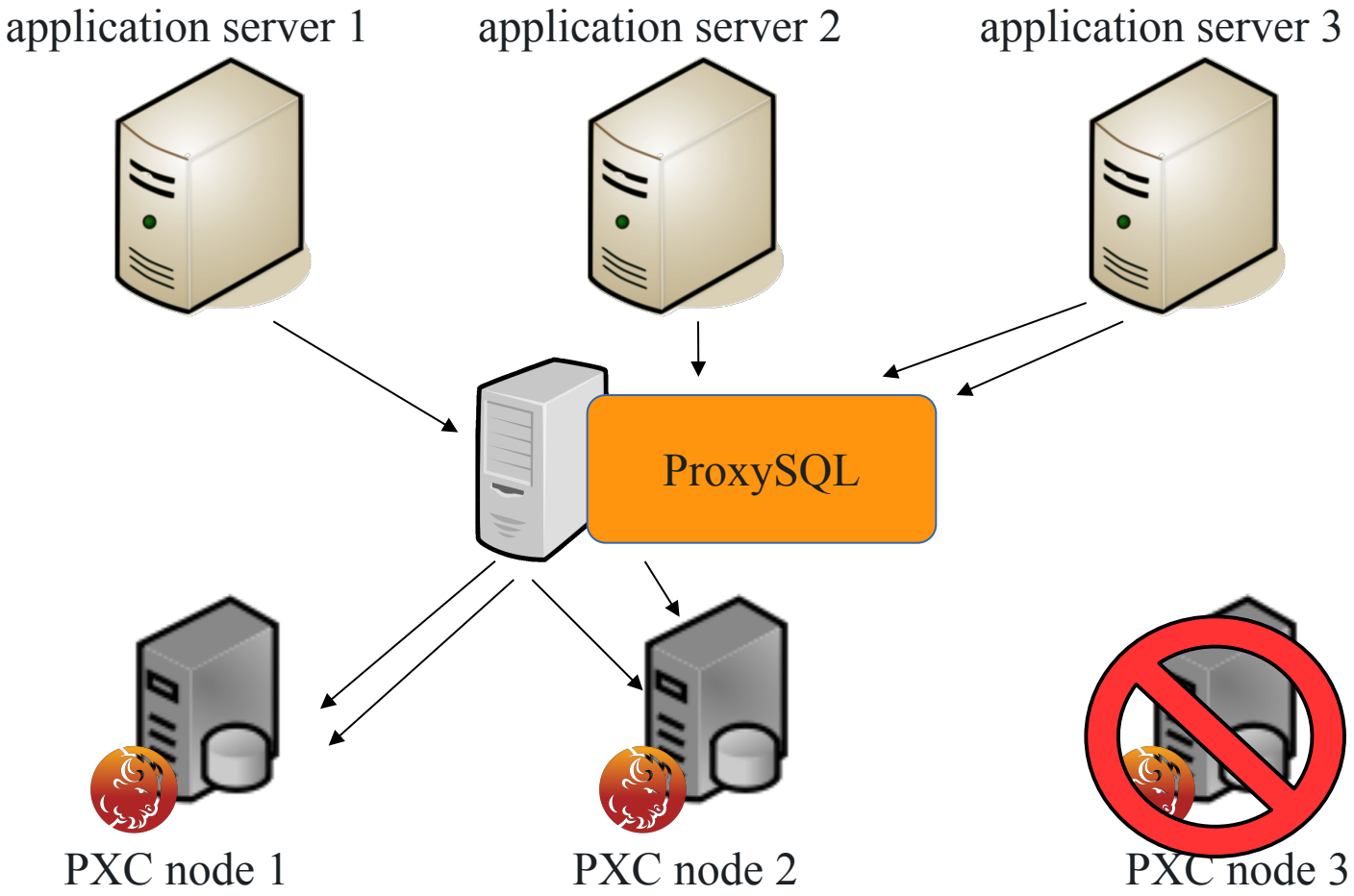
DNS	• Not Immediate, Does not handle existing connections
HAProxy	• TCP/IP Port level. Does not understand MySQL Protocol
Elastic Load Balancer (etc)	• Like HA Proxy in the Cloud
MySQL Router	• Currently very basic
MaxScale, ScaleArc	• Proprietary Solutions
ProxySQL	• MySQL Protocol Aware 100% Open Source Proxy Solution for MySQL

ProxySQL

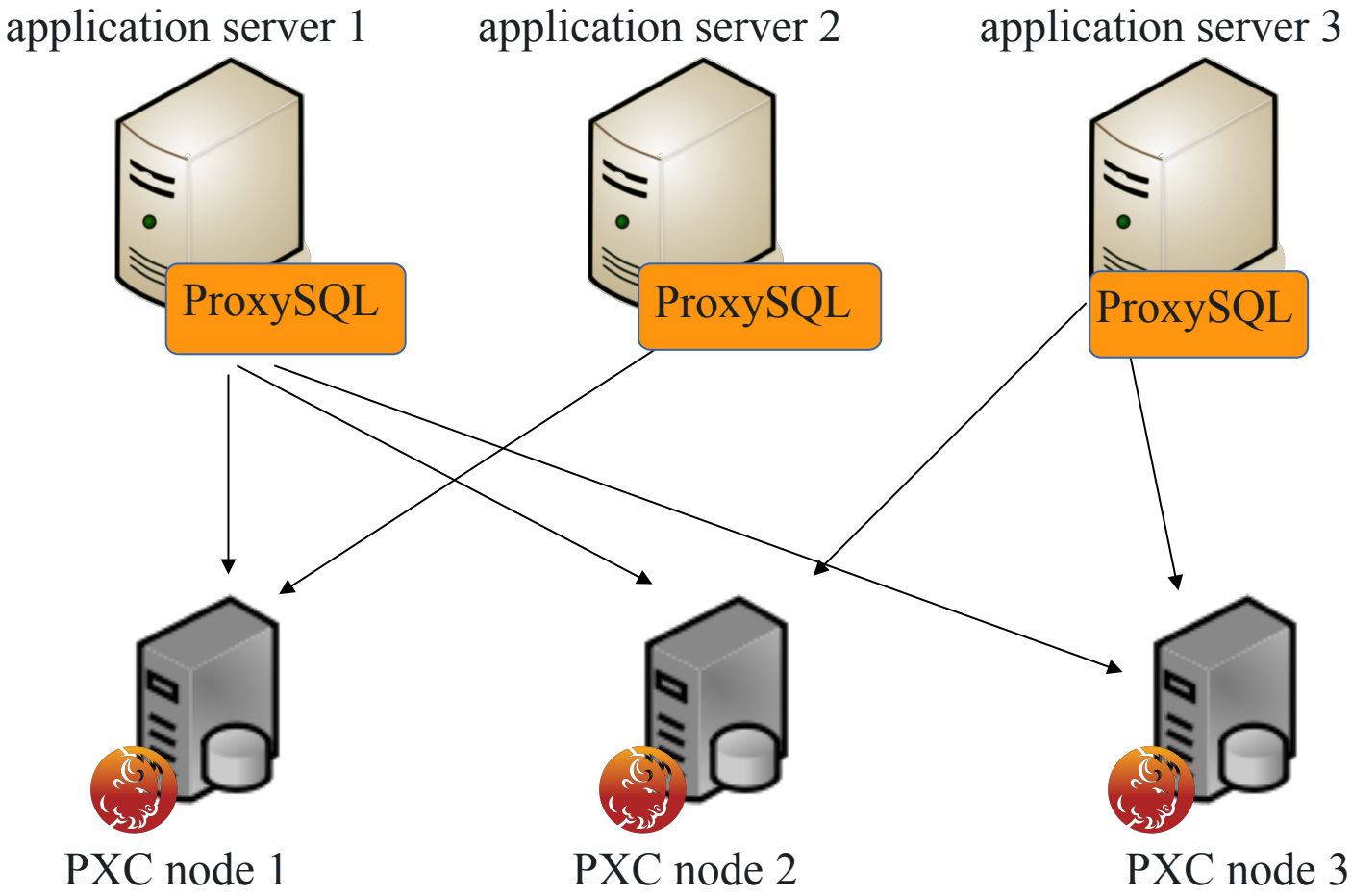
- 100% OpenSource
- Well Supported
- Read-Write Splitting
- Caching
- Query Rewrites
- Load Balancing
- 10k+ Connection Support

PERCONA

Dedicated shared ProxySQL



ProxySQL on application side



ProxySQL at PMM

PERCONA

Cloud Only Options

DBaaS in the cloud comes with built-in HA Options

- Amazon RDS
- Amazon Aurora
- Google Cloud SQL
- Azure Database for MySQL

Beyond MySQL

MySQL is a Part of your Data Infrastructure

MySQL is Good for Many Things but MySQL is not Good for Everything

MySQL can be used together with

Caching	Full Text Search	Queue/Stream Processing	Analytics at Scale
• Redis, Memcache	• Elastic, Solr	• Kafka	• Hadoop, Clickhouse

NewSQL ?

PingCAP TiDB

CockRoachDB

YugaByte

Cross Technology Replication

Trigger Based Replication Tools	• SymmetricDS, Self-Made Solution
Kafka Gateway	• Debezium, Yelp's MySQLStreamer
Tungsten Replicator	• Replication from MySQL to Variety Data Stores
Custom Binary Log Consumers	• MySQL to Tarantool Replication • MySQL to ClickHouse Replication

Summary

Lots of Options Exist for Building Distributed Architectures with MySQL

Lots of Option for Integrating MySQL with other Open Source Data Stores

OPEN SOURCE DATABASE CONFERENCE

PERCONA
LIVE EUROPE
FRANKFURT

5-7 November 2018
Radisson Blu
Frankfurt, Germany

Thank You!