MySQL at Scale at Square

Bill Karwin, Square Inc.
Square

An honest financial network for everyone
Global: USA, Canada, UK, Japan, Australia
Payment transaction data stored in MySQL
We are hiring: https://square.com/careers
Online DataStores team (ODS)

Eight database admin / developers

ODS does:

- Provisioning and operations for MySQL & Redis
- Monitoring database instances
- Oncall rotation
- Design and develop automation
- Supporting developers
Online DataStores team (ODS)

ODS does not do:

- App development
- Data warehouses
- Cloud databases
How Many Databases?
Question: what is your app:database server ratio?

9:53 AM - 6 Oct 2018
Depends...

How do you count apps?
- Request controllers
- Sidecars
- Cron jobs
- Other...

How do you count databases?
- Schemas
- Shards
- Replicas
Microservice-Oriented Architecture
No Multi-Tenancy

Every MySQL instance serves one app
Enforced through MySQL authentication
MySQL Scale in Numbers?
MySQL Instances

Numbers are approximate.
MySQL Metaclusters

Numbers are approximate
Applications

Apps using MySQL: 350+

Microservices architecture is popular
MySQL Metaclusters per App

Numbers are approximate
Volume of MySQL Data
(not including Archives, Data Warehouses, etc.)

Numbers are approximate
Queries per Second

Numbers are approximate
MySQL Version?
Percona Server for MySQL 5.6.41

- Custom build with a few bugfixes, create our own RPM
- Install to base OS image when new host is provisioned

Updates to latest Percona Server 5.6.x

- Homegrown service “ODS-Patching” automates rolling upgrades
Why Not ________?

No Percona XtraDB Cluster
- Solutions like Galera optimize for a workload different from ours

No TokuDB, RocksDB, other storage engines
- Need to maintain tool compatibility, e.g. Percona XtraBackup

No newer versions (yet)
- 2019: upgrading to Percona Server 5.7
- Evaluating Percona Server 8.0 when it's available
## Community Tools

<table>
<thead>
<tr>
<th>Tool</th>
<th>Description</th>
<th>Link</th>
</tr>
</thead>
<tbody>
<tr>
<td>Innotop</td>
<td>Ad hoc monitoring</td>
<td><a href="https://github.com/innotop/innotop">https://github.com/innotop/innotop</a></td>
</tr>
<tr>
<td>Inspect</td>
<td>Metrics for monitoring and alerting</td>
<td><a href="https://github.com/square/inspect">https://github.com/square/inspect</a></td>
</tr>
<tr>
<td>Orchestrator</td>
<td>Replication topology management and visualization</td>
<td><a href="https://github.com/github/orchestrator">https://github.com/github/orchestrator</a></td>
</tr>
<tr>
<td></td>
<td>- pt-online-schema-change</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- pt-table-checksum</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- pt-archiver</td>
<td></td>
</tr>
<tr>
<td>Shift</td>
<td>GUI and service for running online schema changes</td>
<td><a href="https://github.com/square/shift">https://github.com/square/shift</a></td>
</tr>
</tbody>
</table>
Cloud Hosting?
"Cloudless"

All ODS-managed instances are on physical hosts in data centers:

- US West coast
- US East coast
- Japan
Storage Hardware?
Storage Hardware

Local SSD storage for all MySQL datadirs
RAID-0 (striping) with 2 or 3 storage devices
No RAID-1 or RAID-10
No SAN storage
No DRBD
No CEPH
Storage Space

3.5 TB storage per host
Need free space for:
- Data growth
- MySQL binary logs
- ALTER TABLE

Alert when a database is > 1.0 TB
Multiple MySQL instances
Replication?
Replication Topology

“Cluster”
- A pair of instances
- One cluster in each datacenter

“Metacluster”
- A set of clusters
Replication Topology

appname-001-01.west

appname-001-02.west

appname-001-03.east

appname-001-04.east
One Writeable Primary per Metacluster

appname-001-01.west
appname-001-02.west
appname-001-03.east
appname-001-04.east
MySQL Replication Configuration

ROW-based replication
GTID used for 7 apps' metaclusters (so far)
No semi-synchronous replication
Crash-safe replication:
  relay_log_info_repository = TABLE
  master_info_repository = TABLE
  relay_log_recovery = ON
  sync_relay_log = 0
  sync_relay_log_info = 0
  sync_master_info = 0
Running Multiple Instances per Host?
Containers

One MySQL instance per container

- CentOS 6: LXC
- CentOS 7: Docker pods

Sharing system resources:

- CPU cores
- RAM
- Storage volume
- Network
“ODSgroups” — Sets of Physical Hosts

Multiple metaclusters per odsgroup.

Each host has one container from each metacluster.
How Many Containers per Host?
Access and Authentication?
ENCRYPT ALL THE THINGS

USE TLS
Password Policy? None!

No passwords for MySQL users!

Require X.509 certificates for authentication

GRANT ALL PRIVILEGES ON test.* TO 'ramesh'@'localhost' REQUIRE SUBJECT '/C=US/ST=Massachusetts/L=Burlington/O=Sun Microsystems/CN=Ramesh Nagappan' AND ISSUER '/C=US/ST=Massachusetts/L=Burlington/O=Sun Microsystems/CN=SunTest CA';

Square generates certs internally, all instances enforce the Square CA
Internal microservice deploys new certs, issues GRANT statements

High Availability?
Uptime Goal: 99.50%
Actual Average: 99.95%

Sometimes metaclusters have problems during provisioning...
but they are fixed before being employed in production.
Apps Not Allowed to Use Passive Side

Appname-001-01.west

Appname-001-02.west

Appname-001-03.east

Appname-001-04.east
Passive Side

Passive side is for maintenance, not for apps

- Software upgrades
- Software reconfiguration
- Hardware repair
- Ad hoc testing queries
“Successover” == Failover (with Better PR)

Procedure:
1. Make old primary read-only
2. Down RW SIP on old primary
3. Kill client connections on old primary
4. Wait for replication to catch up on all instances
5. Up RW SIP on new primary
6. Make new primary read-write

This procedure usually takes less than 1 second
CNAME: appname-001-rw.global

SIP: appname-001-rw.west

appname-001-01.west

appname-001-02.west

appname-001-03.east

appname-001-04.east
Successover Between Datacenters

Procedure:
1. Make old primary read-only
2. Down RW SIP on old primary
3. Kill client connections on old primary
4. Wait for replication to catch up on all instances
5. Change DNS resolution for CNAME
6. Reconfigure replication
7. Up RW SIP on new primary
8. Make new primary read-write

This procedure usually takes about 1 second
Provisioning?
SpinCycle

Job scheduler for executing distributed automation tasks
Like Chef, Ansible, Celery, etc.

SpinCycle uniquely satisfies design requirements:
  ● Requests are API-driven
  ● Specify requests based on future state of the world
  ● First-class end-to-end feedback loop/status (observable, auditable)

Framework is open-source, Square-specific code is not
  ● https://github.com/square/spincycle
MySQL Instance Inventory?
Database of Databases

Etre (https://github.com/square/etre)
- Basically: JSON-as-a-service
- Every MySQL instance, host, app, and SIP is an entity in Etre

Other tools use Etre data as the “source of truth”
- Provisioning
- Successover
- Monitoring
Example Etre Entity

```
{
  "_type": "node",
  "_id": "5afcbe6496...",
  "container_type": "docker",
  "dbtype": "mysql",
  "env": "production",
  "hostname": "vam-mysql-ods-bkarwintest-001-01.west",
  "metacluster": "am-mysql-ods-bkarwintest-001",
  "app": "ods-bkarwintest",
  "backup_schedule": "5 8 * * *",
  "mysql_innodb_buffer_pool_size": "1",
  "mysql_binlog_format": "ROW",
  ...other attributes...
}
```
Monitoring?
Metrics

Inspect

- [https://github.com/square/inspect](https://github.com/square/inspect)
- Metrics agent runs in every MySQL container
- Uploads metrics to SignalFX.com
SignalFX MySQL Dashboard
## Container Viz (home-grown dashboard)

![Container Viz Dashboard](image)

### am-odsgroup2 (staging / mysql / am)

<table>
<thead>
<tr>
<th>Container</th>
<th>Memory</th>
<th>Storage</th>
</tr>
</thead>
<tbody>
<tr>
<td>vam-mysql-001-01</td>
<td>1 GB</td>
<td>0.2 GB</td>
</tr>
<tr>
<td>vam-mysql-001-02</td>
<td>1 GB</td>
<td>2.4 GB</td>
</tr>
<tr>
<td>vam-mysql-001-03</td>
<td>1 GB</td>
<td>0.3 GB</td>
</tr>
<tr>
<td>vam-mysql-001-04</td>
<td>1 GB</td>
<td>4 GB</td>
</tr>
<tr>
<td>vam-mysql-001-05</td>
<td>1 GB</td>
<td>0.1 GB</td>
</tr>
<tr>
<td>vam-mysql-001-06</td>
<td>1 GB</td>
<td>0.1 GB</td>
</tr>
<tr>
<td>Total container allocation:</td>
<td>12 GB</td>
<td>19.1 GB</td>
</tr>
<tr>
<td>Other resource usage:</td>
<td>16.4 GB</td>
<td>25.3 GB</td>
</tr>
<tr>
<td>Actual usage:</td>
<td>28.4 GB</td>
<td>44.5 GB</td>
</tr>
<tr>
<td>Capacity: (sandisk_sdb)</td>
<td>88.9 GB</td>
<td>1.7 TB</td>
</tr>
<tr>
<td>Percent used:</td>
<td>31.9%</td>
<td>2.5%</td>
</tr>
</tbody>
</table>

### am-odsgroup2 (staging / mysql / am)

<table>
<thead>
<tr>
<th>Container</th>
<th>Memory</th>
<th>Storage</th>
</tr>
</thead>
<tbody>
<tr>
<td>vam-mysql-001-01</td>
<td>1 GB</td>
<td>0.2 GB</td>
</tr>
<tr>
<td>vam-mysql-001-02</td>
<td>1 GB</td>
<td>2.6 GB</td>
</tr>
<tr>
<td>vam-mysql-001-01</td>
<td>1 GB</td>
<td>0.2 GB</td>
</tr>
<tr>
<td>vam-mysql-001-01</td>
<td>1 GB</td>
<td>0.1 GB</td>
</tr>
<tr>
<td>vam-mysql-001-01</td>
<td>1 GB</td>
<td>0.1 GB</td>
</tr>
<tr>
<td>vam-mysql-001-01</td>
<td>1 GB</td>
<td>0.1 GB</td>
</tr>
<tr>
<td>Total container allocation:</td>
<td>12 GB</td>
<td>19.3 GB</td>
</tr>
<tr>
<td>Other resource usage:</td>
<td>14.3 GB</td>
<td>25.9 GB</td>
</tr>
<tr>
<td>Actual usage:</td>
<td>26.3 GB</td>
<td>45.2 GB</td>
</tr>
<tr>
<td>Capacity: (sandisk_sdb)</td>
<td>85.2 GB</td>
<td>744.3 GB</td>
</tr>
<tr>
<td>Percent used:</td>
<td>30.9%</td>
<td>6.1%</td>
</tr>
</tbody>
</table>
This host died; all containers are down

Replication from offline containers has stopped

Two metaclusters unaffected because their primary instance was on the 04 host
Alerting?
Who Gets Alerts?

Alerts must be *actionable* by the person paged

**App Owners**
- Active long running queries
- Current connections percent
- History list length
- Oldest query
- Queries per second
- Seconds behind master
- Threads running

**ODS team**
- At least one node per metacluster is running MySQL
- CNAME responds
- ReadOnly == SuperReadOnly
- Replication running
- SIPs assigned properly
Recommendations
Cattle, Not Pets

Make instances:

- Generic
- Replaceable
- Without human intervention

Not Like These →
Pebbles, Not Boulders

Make operations:
● Granular
● Incremental
● Non-invasive
Uniformity

Simplify tools and procedures
Divide large into small
Standard type
Standard size
Automation

Fewer manual operations
More scripted operations
Asynchronous
Observable
Q&A