MySQL In the Cloud
Migration, Best Practices, High Availability, Scaling

Dimitri Vanoverbeke (credits to Peter Zaitsev)
Solution Engineer
Percona University Ghent
June 22th of 2017
Who am I

Solution engineer within Percona!

www.twitter.com/dim0 (Personal views and potentially NSFW)
Let me start....

With some Questions!
Question One

How Many of you are running MySQL In the Cloud?
Question Two

Are you running it in Public Cloud? Private Cloud? Both?
Question Three?

Are you using DBaaS such as Amazon RDS or Google CloudSQL?
Question Four

Are you using Containers?
Let's Cover some Basics
What is “Cloud”

Dynamic Programmable Infrastructure
# Public and Private

<table>
<thead>
<tr>
<th>Public</th>
<th>Private</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Infrastructure Shared with other Users</td>
<td>• Infrastructure Private for company</td>
</tr>
<tr>
<td>• Amazon AWS typical example</td>
<td>• OpenStack installation typical example</td>
</tr>
</tbody>
</table>
All of those XaaS

IaaS (Infrastructure as a service)
- Works in Infrastructure level: “Compute”, “Storage”, “Network”
- Examples: AWS EC2, S3, EBS

DBaaS (Database as a service)
- Provides Database Service (Instances or Clusters) to use
- Examples: Amazon RDS, Google Spanner

PaaS (Platform as a service)
- Provides full platform for your application development
- Examples: Heroku, Amazon Elastic Beanstalk, OpenShift
Regions and Availability Zones

<table>
<thead>
<tr>
<th>Region</th>
<th>Availability Zone</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Specifies Geographic Region</td>
<td>• Is located in the region</td>
</tr>
<tr>
<td>• Hierarchy - Europe – West – Ireland</td>
<td>• Reasonably isolated from each other</td>
</tr>
<tr>
<td>• High Latency between Regions</td>
<td>• Medium Latency between AZ</td>
</tr>
<tr>
<td>• Complete Isolation</td>
<td></td>
</tr>
</tbody>
</table>

- Complete Isolation: Isolated from each other.
Making it Visual

Top Cloud Providers

Google Cloud Platform

Amazon Web Services

Microsoft Azure
Technologies to be aware of

- openstack
- kubernetes
- OPENSHIFT
- docker
Decisions to Make
Should you move to the cloud?

- This is a decision you rarely have.
- Programmable infrastructure is the future.
- Virtualization overhead is going down.
- Some clouds providers support Bare Metal.
Public ? Private ? Hybrid

Public Cloud
- Agility
- Scalability
- Costs
- Small and Medium Businesses

Private Cloud
- Control
- Costs
- Legacy Integration
- Some Enterprise Companies

Hybrid Cloud
- Infrastructure using Both
- Can get benefits of both
- At the cost of extra complexity
Single vendor vs Multi Vendor

**Single Vendor**
- Use all features vendor has to offer
- Danger of Vendor Lock In

**Multi Vendor**
- Have to use “lowest common denominator”
- Avoid Vendor Lock In
DBaaS

DBaaS (ie Amazon RDS)
- Easier
- Takes off some operational pains
- Less Flexible
- More Expensive
- More Lock-In

IaaS (ie EC2+EBS+S3)
- Harder to roll your own
- Operations on your own (or your partner)
- More Flexible
- Less Expensive
- Less Lock-In
Open Source in the Cloud

Open Source Compatible is not same as Open Source
Keep it Simple

Do not try doing upgrade at the same time as migration

Exactly same minor version is optimal

Same major version - must
Moving to IaaS Cloud

- General Practices as in Datacenter Migrations apply
- Easy to use Binary Backups
- Slave_compressed_protocol or compression in VPN
- Support utilities may need to be modified for EBS/S3
Moving to DBaaS

---

**Need to use database dump to copy**
- Mysqldump
- Mysqlpump
- Mydumper

**Can set external slave (Amazon RDS)**
- CALL mysql.rds_set_external_master

**Monitoring Backup may need revision**
- Do not have direct access to physical box
- Do not have root user
New With Amazon RDS Aurora

Can use Percona Xtrabackup’s Backup to seed the cluster

http://amzn.to/2pk6Iq7
Moving from DBaaS

Logical Database dump as well

Replication supported for Migration only

Configure Binary Log Retention

`mysql.rds_set_configuration`
Best Practices
Being Cost Efficient

- Know your cloud vendor pricing policies
- Look beyond “compute” pricing
- Best Price/Performance configuration in the cloud is likely to be different
- AWS: Reserve Instances
- AWS: Spot Instances
Guarantee versus Burst

**Guaranteed**
- Performance resource is “guaranteed” to have in worst case scenario
- This is what you can plan for

**Burst**
- Performance resource can provide
- Typically not guaranteed
- Typically limited in length to prevent abuse
Network

Understand Application-Database Network Latency

Same AZ Optimal; Same Region Must have

10Gb Network

Understand network “jitter”

Latency is critical for most applications

Bandwidth can be important for dumps and batch job
CPU

- Same whenever you’re in the cloud or not
- MySQL uses single thread for single query
- Multi-Core gives good scalability for “Web” workloads
Memory

Use mainly as a cache

Very important for Performance
### Storage

<table>
<thead>
<tr>
<th>Instance Local Storage</th>
<th>Cloud Block Storage</th>
<th>File/Object Storage</th>
</tr>
</thead>
<tbody>
<tr>
<td>• May or may not be available</td>
<td>• Reliable</td>
<td>• Store Files/Objects</td>
</tr>
<tr>
<td>• Not Highly available</td>
<td>• Remote</td>
<td>• No interactive block level access</td>
</tr>
<tr>
<td>• May be inexpensive and high Performance</td>
<td>• Separately Prices</td>
<td>• EBS on AWS</td>
</tr>
<tr>
<td></td>
<td>• EBS on AWS</td>
<td>• S3 on AWS</td>
</tr>
</tbody>
</table>

© 2017 Percona
Things to Note

- You can’t get any combination
- EBS Performance depends on the instance size
- Provisioned IOPs for Optimal Performance
- Glacier storage for old backups
Operating System

Consider Cloud Optimized Linux Versions

At very least use Recent Linux Versions

“Cloud Only” Linux might be inconvenient for development
MySQL Version and Configuration

Use Recent Version

Do not count on good Defaults
There is essentially same systems underneath!

Most of same practices Apply
DBaaS is not always faster

Source: https://twindb.com/rds-vs-aurora-vs-ec2-benchmark/
High Availability
Your Choices

Roll your own

Use DBaaS
Things to Consider

You have less control or visibility into the infrastructure

Things as IP take –over might not work
Load Balancers

Cloud Load Balancer (Elastic Load Balancer at AWS)

HAProxy

ProxySQL
Maintaining copies of Data

MySQL Replication

MySQL Group Replication

Percona XtraDB Replication (PXC) and Galera
Why Percona XtraDB Cluster in the Cloud

- Read/Write to any node works great with simple load balancers
- Automatic Provisioning and Auto Scaling
- Can run with local instance storage
- Can deploy across multiple AZ
Scaling
Scalability in the Cloud

“Better”

- Due to cloud optimized options like Amazon Aurora

“Worse”

- Due to restricted hardware choices
Scaling How

Scale Up
• Vertical Scaling
• Scale with the Hardware Size – CPUs, Memory, Storage

Scale Out
• Horizontal Scaling
• Scale by adding nodes
Bad reputation of Scaling Up... but

Reasonable commodity MySQL Server Can handle

- 3-5TB database size
- 100K+ queries/sec
- 5M+ rows read/sec
- 100K rows modified/sec
Scaling What?

- Reads
- Writes
- Data Size
Scaling Reads

- Replication
- Caching
- Moving some load from MySQL
Scaling Writes

- New MySQL Versions
- Parallel Replication
- TokuDB
- Functional Partitioning
- Sharding
New in Sharding

ProxySQL

Vitess
Scaling Data Size

Functional Partitioning and Sharding

Data Archiving

TokuDB for Compression

Often Operations drive this needs not App Performance
Percona Live Europe Call for Papers & Registration are Open!

Championing Open Source Databases

- MySQL, MongoDB, Open Source Databases
- Time Series Databases, PostgreSQL, RocksDB
- Developers, Business/Case Studies, Operations
- September 25-27th, 2017
- Radisson Blu Royal Hotel, Dublin, Ireland

Submit Your Proposal by July 17th!
www.percona.com/live/e17
Database Performance Matters