Running MySQL on Three Different Platforms at Booking.com

Mohammed Gaafar
Senior Site Reliability Engineer

Martin Alderete
Principal Site Reliability Engineer

Booking.com

Agenda

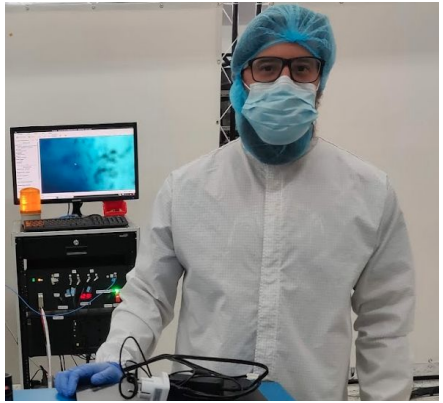
1. About us
2. MySQL Architecture at Booking.com
3. The evolution of Bare-Metal
4. Problems with Bare-Metal
5. The Rise of Virtualization
6. Storage Testing
7. Extending to yet another platform (AWS EC2)

Mohammed Gaafar

1. Senior Site Reliability Engineer
2. 15 years of experience in Linux System Administration, MySQL Administration and Site Reliability Engineering.
3. Computer Science and Bioinformatics Background
4. At Booking.com since Feb 2018
5. Application Data Services Department - DB-Engineering Team

Martin Alderete

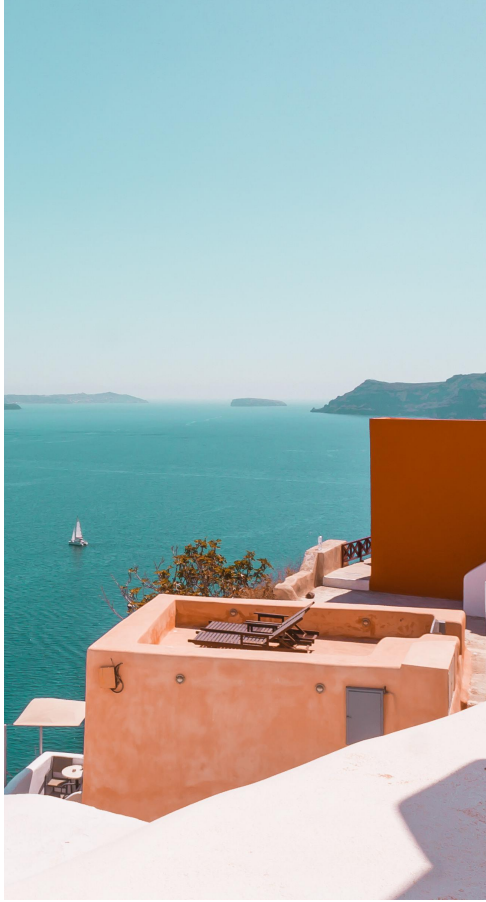
1. Principal Site Reliability Engineer
2. Born in Patagonia Argentina :)
3. Background in System engineering and distributed systems. Worked in multiple industries / academia
4. At Booking.com since Jan 2023
5. Application Data Services Department



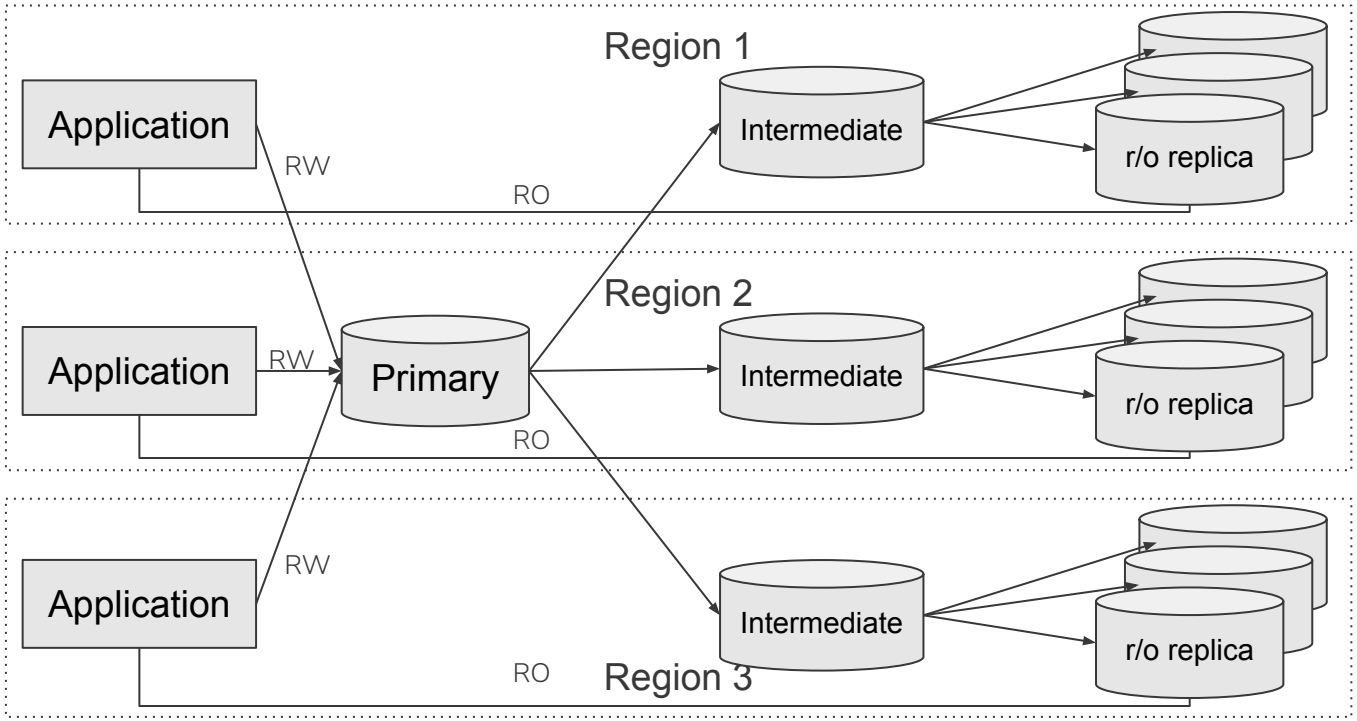
MySQL at Booking.com

The Scale of MySQL at Booking.com

- MySQL is the main relational datastore at Booking.com.
- Thousands of MySQL servers grouped in over 200 MySQL clusters, a.k.a. replication chains across multiple fully isolated network environments.
- Each replication chain consists of a primary for RW queries and reads are scaled out to 10s or 100s of RO replicas.
- Multi-tier replication chains to avoid overloading the primary.
- A typical replication chain spans 3 different regions within Europe.
- Some RDS clusters are being used for production workloads.
- Complete testing environment with periodic copies of data/schema from production.



Replication Chains



The Evolution of Bare-Metal

- Over the years, we have been fully running our MySQL servers on Bare-Metal servers.
- Over 90% of the fleet run on local ephemeral storage. The rest of the servers are systems with relatively large datasets, >7TB, and they use high performance NAS.
- NAS vendors changed over the years. However, currently, we are running on PURE storage and plan to use HPE Aletra in the near future.
- Local storage Evolved from HDDs to SSDs over the years. Only a few systems run on HDDs at the moment and these are mostly small datasets that fit in the InnoDB buffer pool.
- Chains have enough redundancy to tolerate server failures and up to a region failure for enough time to provision new servers.
- Copying data to new servers is mainly done using MySQL Native Cloning.

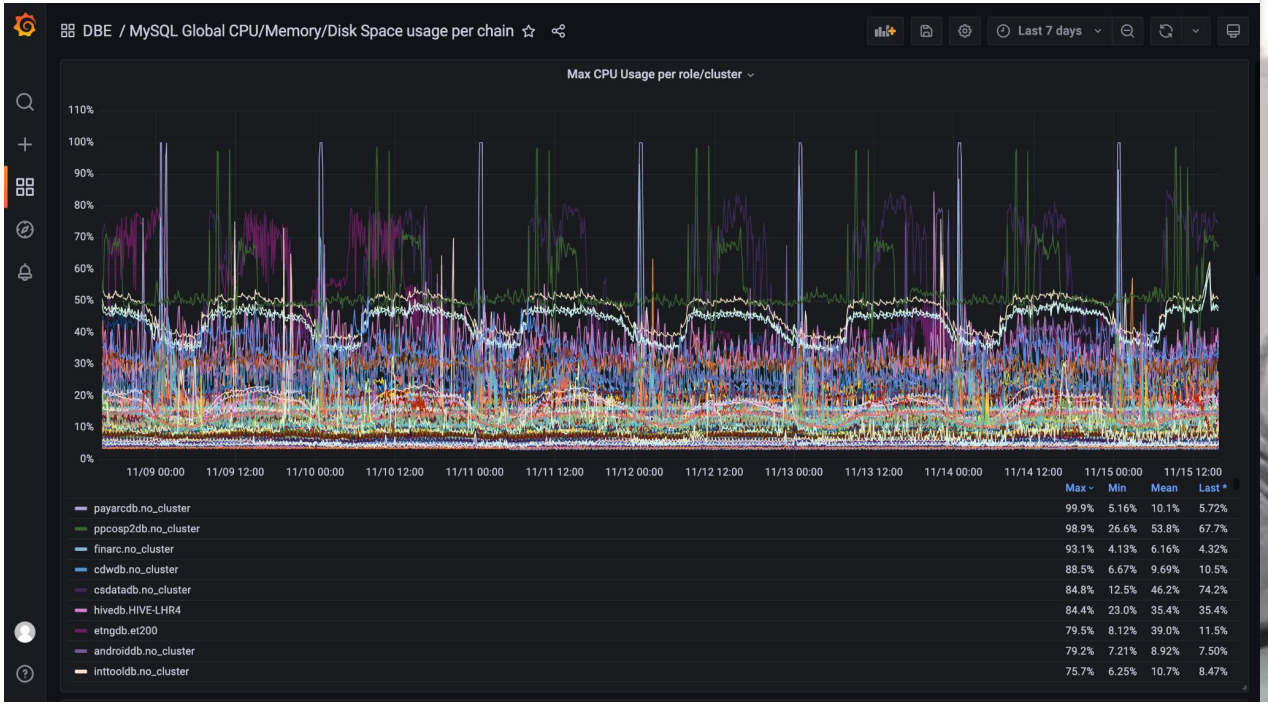
The Evolution of Bare-Metal

- Focusing on Majority of our fleet running on local SSDs.
  - Servers come in 3 main flavours that mainly vary in terms of disk space size.
  - The same memory and CPU capacity on all flavours.
  - Smallest disks are 1.7TB and largest are 8TB.
  - A few systems require different flavours with high CPU or Memory.

Problems with Bare-Metal

- Moving to SOA means 1000s of smaller schemas each is accessed by a few services this translates to either 1000s of underutilized servers or multi-tenant chains.
    - *The Noisy Neighbor headache*.
    - Capacity planning becomes more challenging.
    - Cost attribution becomes harder and harder.
- Horizontal scaling is always at the risk of delayed hardware deliveries and installation.
- Servers are becoming more powerful than we actually need.
- Vertical scaling is not suitable for a lot of systems to be able to guarantee redundancy, reliability and fault tolerance.
- Limited data centers capacity
- Data Centers expansion are slow, complicated and expensive.

Resource Utilization



Resource Utilization

Resource Utilization

The Rise of Virtualization

- Virtualization gives a lot of flexibility when it comes to server sizing.
  - We get to define what flavours we need.
  - Higher resource utilization.
- Vertical and horizontal scaling becomes easier.
- Shorter server provisioning time due to the usage of pre-baked images.
- Better control over servers quotas and cost attribution.

Virtualization Challenges

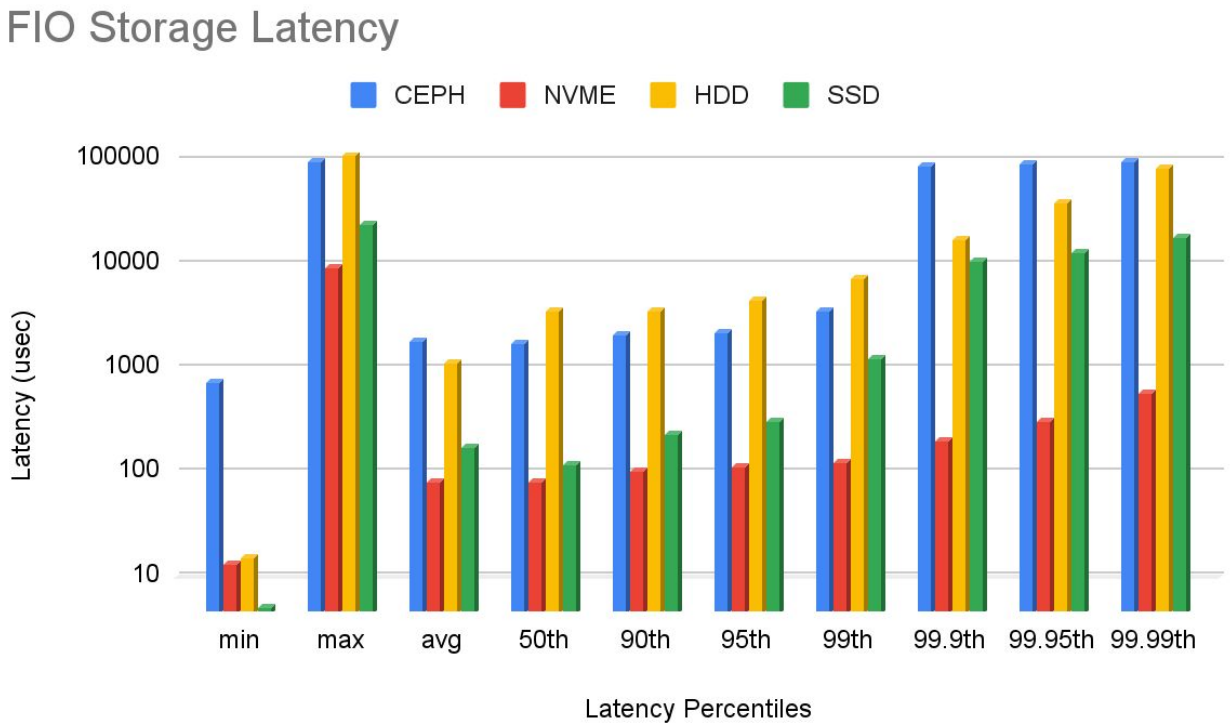
- Automation changes:
  - Changing the assumption that everything runs on the same platform.
  - Supporting new backends and server inventories.
  - Supporting transient migration states.
  - Making the automation extendable to support more platforms in the future.
  - Creating an automated, transparent and reliable migration plan.
- Compliance
  - Compliance of the platform.
  - Server patching.
- Which Storage to use?!

Storage Testing

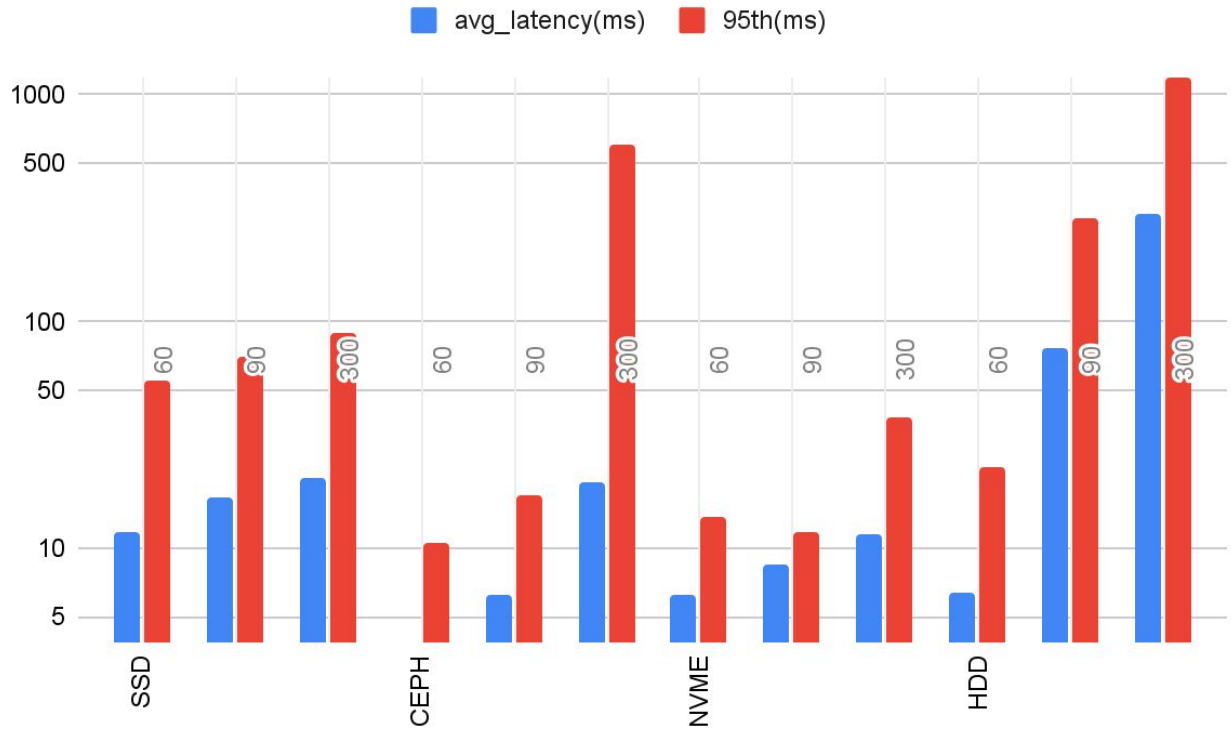
- There are 3 main storage solutions available on OpenStack.
  - CEPH: the default OpenStack storage.
  - NAS: high performance network attached storage.
  - Local storage: NVME drives on OpenStack hypervisors.

Write Latency (µs)	Max IOPs	Notes
1,000	1,000	Ceph baseline
500	2,000	Historical target for shared storage
400	2,500	DBE target for MySQL
250	4,000	iSCSI baseline
100	10,000	Local SATA SSD storage
40	25,000	Local NVMe SSD storage

FIO


FIO Storage Latency
CEPH
NVME
HDD
SSD
100000
10000
1000
100
10
Latency (usec)
min
max
avg
50th
90th
95th
99th
99.9th
99.95th
99.99th
Latency Percentiles

SYSBENCH


avg_latency(ms)
95th(ms)
1000
500
100
50
10
5
60
90
300
SSD
CEPH
NVME
HDD

Other Tests

- Replication load testing
  - How much time it takes a server to catch up with replication.
- Load Capacity testing
  - Push servers to saturation on one of the resources to measure its capacity.

Pain Points

- New platform comes with new challenges/problems.
- Stability of OpenStack and CEPH.
- Increased failure domain; rack vs cloud.
- Lift & Shift Vs Optimization.
- Increased risk of noisy neighbor.
- Automation refactoring.

Extension to AWS EC2 - Why?

- Booking's Cloud strategy to adopt Public Cloud (AWS)
- Our current automation assumes, it has full access to a "compute platform"
- Current responsibility model of MySQL's teams (platform organisation)
- Logical next step to have presence in Public Cloud
- Booking's architectures and shared databases (ownership / responsibility)
- Expertise
- ...

What about managed services like RDS MySQL?

Extension to AWS EC2 - Challenges

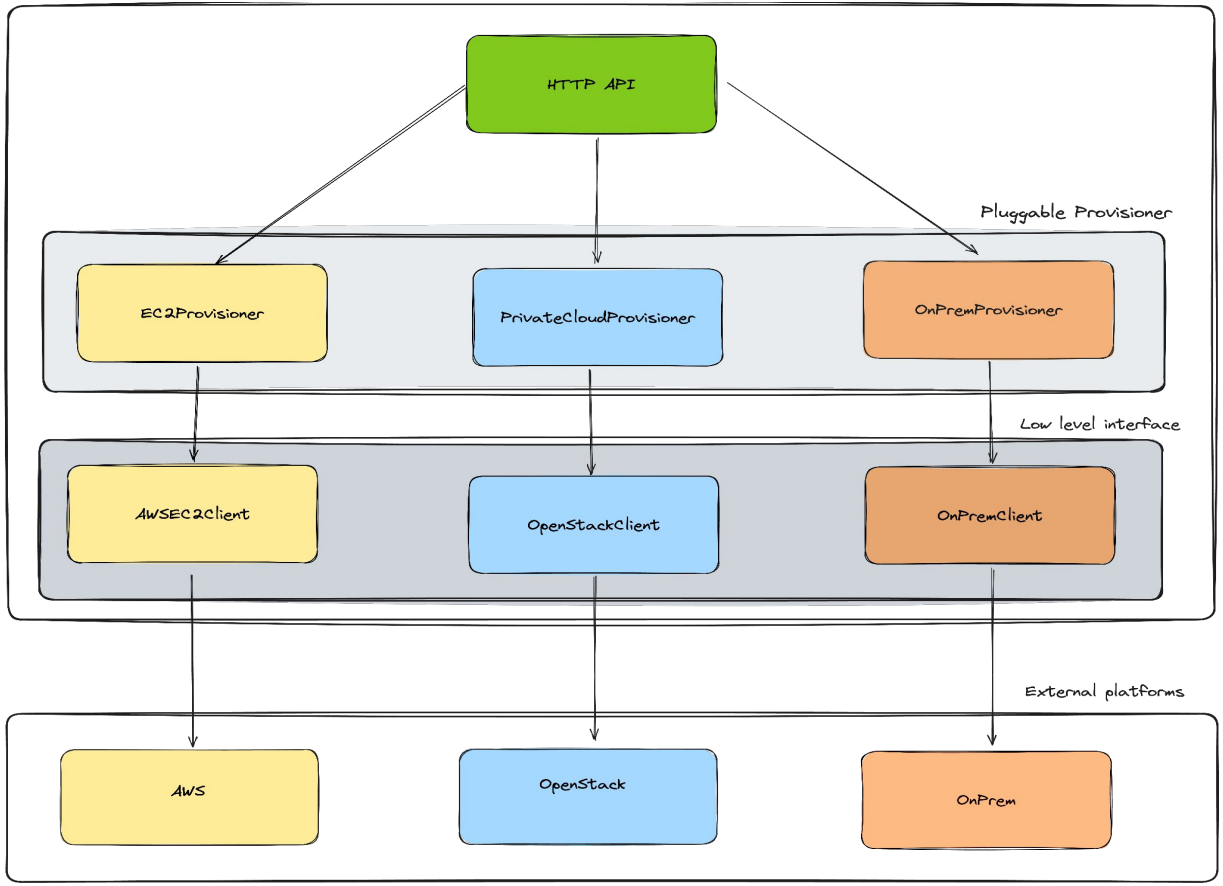
- Learn about the new Platform and all the offering
- Integrate with the new Platform (inventory, metadata,etc)
- Adapt the tooling to handle the new Platform **as transparent as possible**
  - Integrate new concepts... (DC, AZ, Region)
- Images (AMI) management
- Configuration management changes
- Adapt processes to the new Platform (e.g. Patching)
- Ensure new Platform remains compliant

Our complex fleet management tool

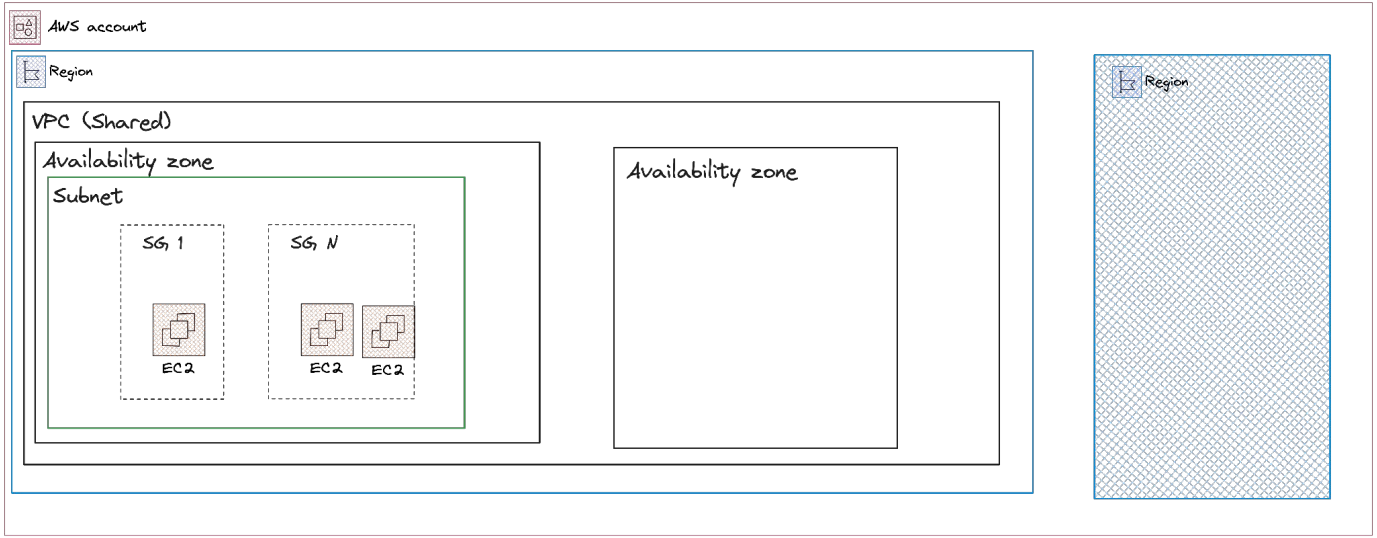
- Chain (cluster) management
- Provisioning / Deprovisioning
- Server “flavours” management
- MySQL Version management
- Topology management
- Service Discovery
- Switchover / Failover
- Monitoring, self-healing and alerting
- Capacity testing
- Backup and Restore testing
- Connection management
- Root password rotation
- Schema changes
- ETC...

Initial commit -> “Date: Thu Sep 6 11:59:32 2012 +0200”

Plugable Provisioner Architecture



AWS Components - Overview



All these components (Account, Region, VPC, AZ, Subnet, Security Group) are new to the automation

Instance types (aka “flavours”)

“Try find the minimum number of vCPU and RAM needed to run our workload”

- Listed all the available types
- Selected **“similar”** (CPU, RAM) to bare-metal “flavours” (**m5.8xlarge**) -> OK
- Tried memory **optimised types,** CPU utilisation is quite low (**r6i.4xlarge**) -> OK
- Tried local-storage: **r6id.4xlarge** (~1TB), i**4i.4xlarge** (~3.7TB) -> OK
- Can we go smaller?
- Tried **i4i.large** -> FAILED (it has only 2vCPU and couldn’t hold replication!)
- We tried **i4i.xlarge** -> OK
- We tried older generation instances too **i3en.2xlarge**-> OK

“In the long run local-storage seems cheaper using Reserved instances”

AWS - EC2 instance types

AWS - EC2 reserved instances

Instance types (aka “flavours”) - EBS

- EBS (Elastic Block Storage) only used for BIG chains (> 7.5TB)
- We don’t really need high IOPS
  - It is cheaper to use **GP3** (up to 16TB) rather than **IO2** (+IOPS and >size)
- > 16TB?
  - Multiple **GP3** (16TB). E.g. 26TB required, 2 x 16TB

AWS - EBS

Instance types (“flavours”) - Heuristic

GB \ Instances	r6id.large	r6id.xlarge	i4i.xlarge	i3en.xlarge	i3en.2xlarge	i3en.3xlarge	r6i.4xlarge
< 118	X						
< 237		X					
< 937			X				
< 2500				X			
< 5000					X		
< 7500						X	
> 7500							X (EBS)

We consider a “safety factor” when calculating the size

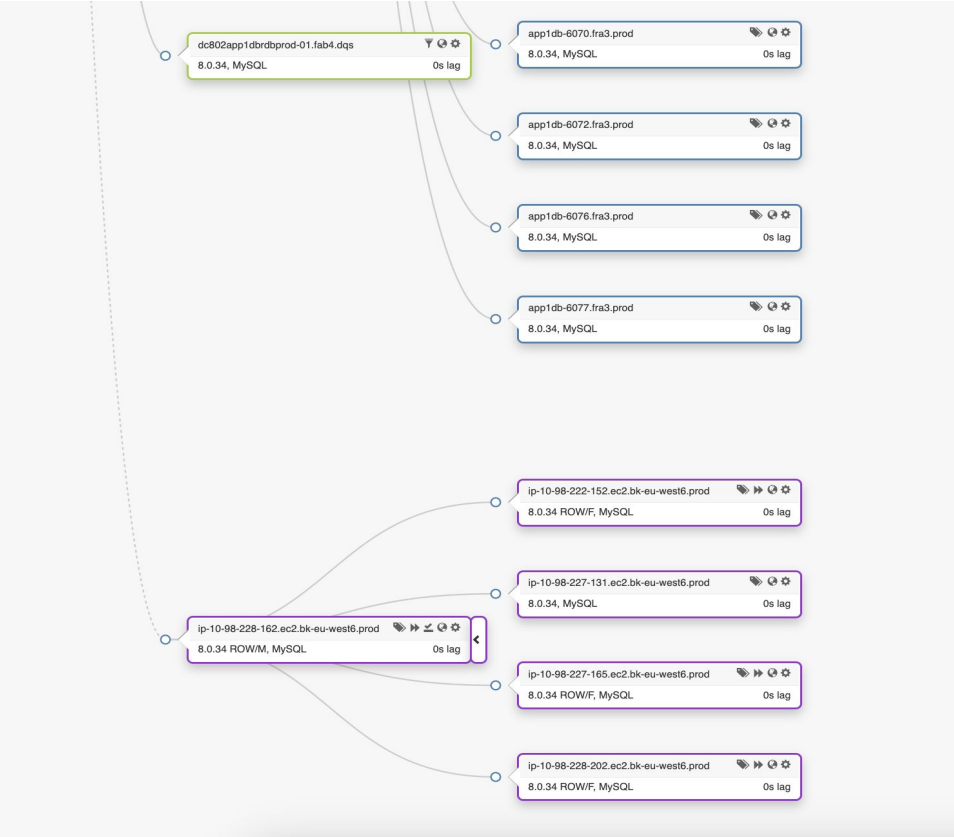
Instance type - Heuristic Automation

```
❯ dba chain onboard_to_ec2_pick_flavor chainX
INFO Gathered disk_used_gb=2227 from Prometheus.
With safety_factor=2.1 this gives us the desired volume_size_gb=4676
INFO The 'set' flag is not used.
It would update flavor to i3en.2xlarge from i4i.4xlarge for chain chainX
-= RETURN =-
i3en.2xlarge
❯
```

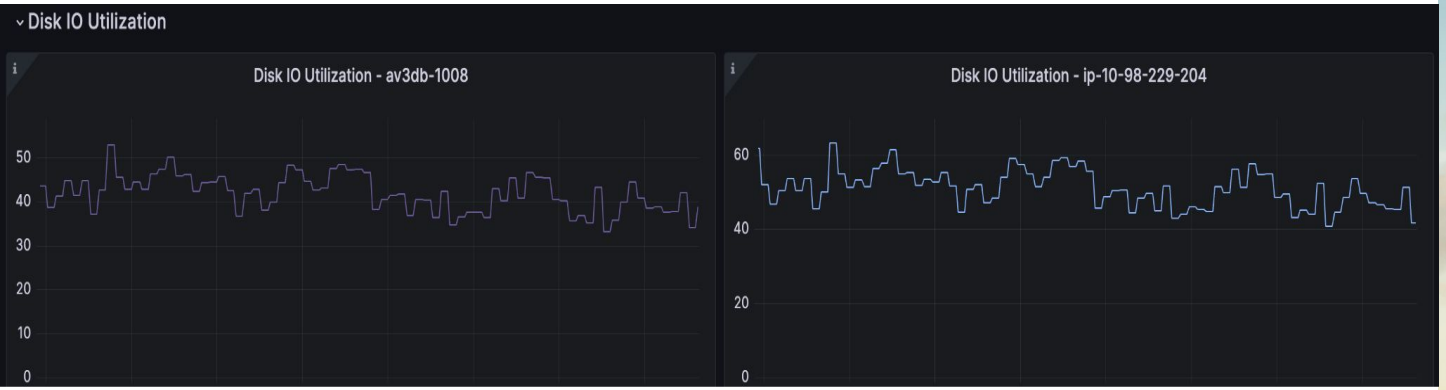
```
{
    "prod": {
        "default_provisioner": "serverdb",
        "flavors": {
            "ec2": "i3en_2xlarge",
            "serverdb": "Blade8_1"
        }
    }
}
```

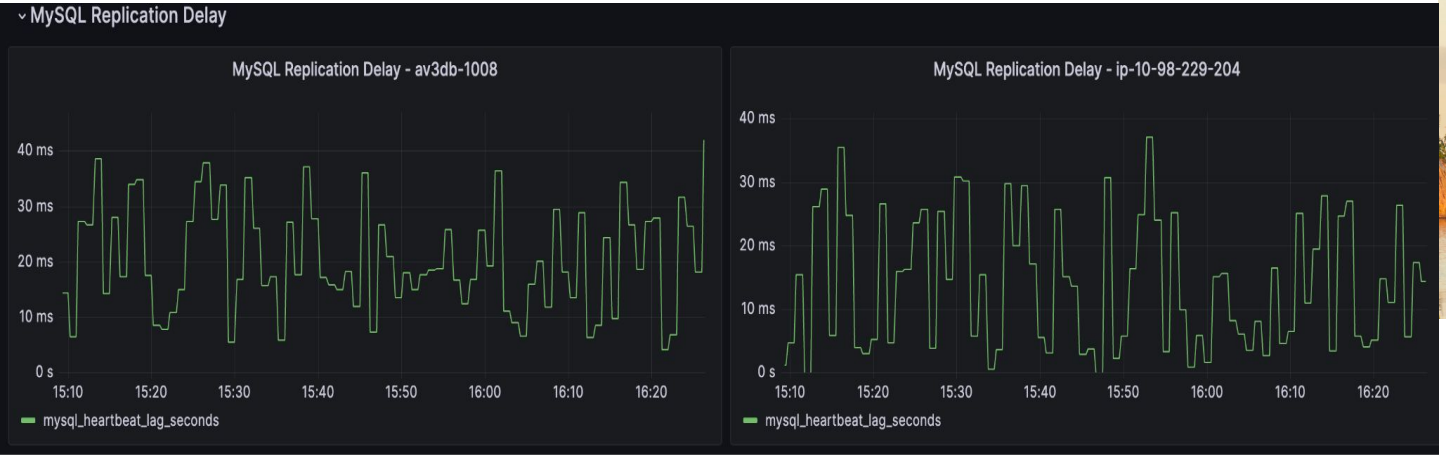
THIS IS HOW CONFIG LOOKS LIKE

Fleet extended to EC2!

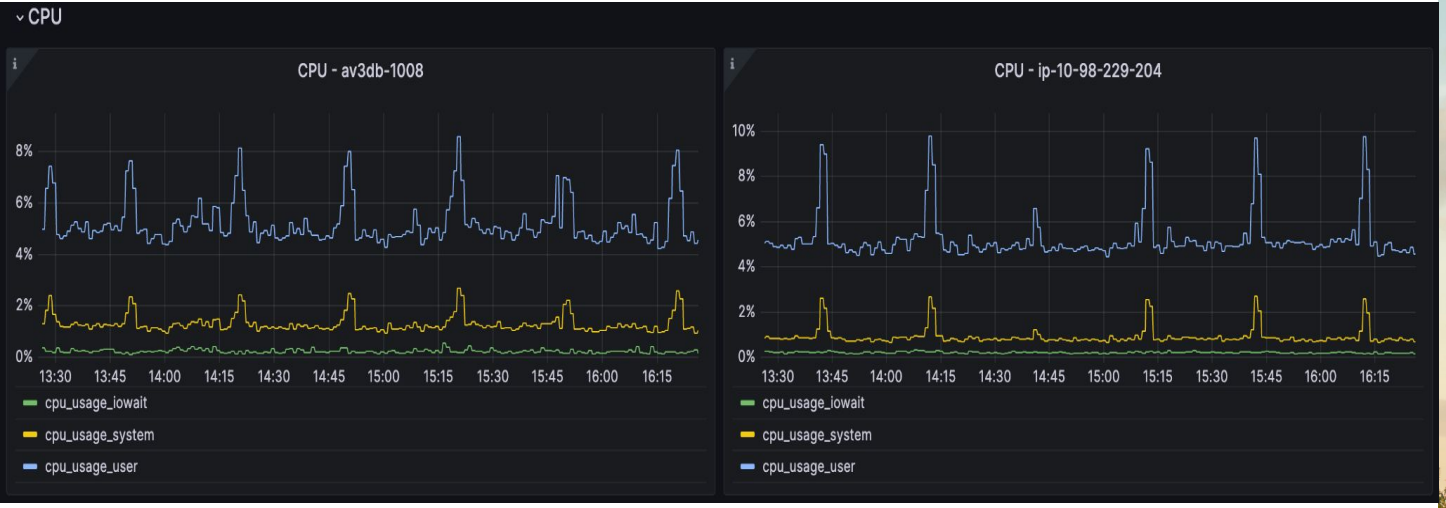


Disk IO utilization / replication lag





CPU utilization



Extending processes - Patching

- Uses an AWS API call ReplaceRrootVolume
- Replaces the AMI of a running instance and keep the data volume
- Takes ~90 seconds

▼ Recent root volume replacement tasks

Filter tasks

Replace root volume

Task ID	Task state	Start time	Completion time	Tags
replacevol-062a5baea817d8d5a	Successful	2024-01-07T10:56:31Z	2024-01-07T10:57:53Z	-
replacevol-062c183f2e5400727	Successful	2023-12-13T17:39:12Z	2023-12-13T17:40:47Z	-

```
import boto3

client = boto3.client('ec2')
inst_id, ami_id = ('i-07c6c0048aa0f7db5', 'ami-0fe3a22e643696344')
resp = client.create_replace_root_volume_task(InstanceId=inst_id, ImageId=ami_id)
```

Pain Points / Future

- New Platform new concepts, new issues, new challenges
- Discover “wrong” and “unknown” thighs in current environment
- Automation refactoring
- Cost AWS billing can skyrocket easily
- Hard to keep updated about new instances types and their performance

- Plan to use “Reserved instances” and then evaluate “Spot instances”
- Define new workload profiles with different reliability targets
- Evaluate “create use destroy” option (if feasible)

Questions?


Booking.com
Empowering people to experience the world
Thanks
mohammed.gaafar@booking.com
martin.alderete@booking.com