Fault Domains in Mesos

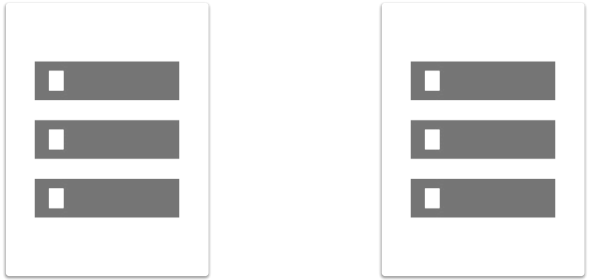
Vinod Kone (vinodkone@apache.org)

About me

- Apache Mesos PMC and Committer
- Engineering Manager for Mesos team @ Mesosphere
- Previously Tech Lead for Mesos team @ Twitter
- PhD in Computer Science @ University of California Santa Barbara

Fault Domain

- A set of nodes that share similar failure (and latency) characteristics



Rack 1

Fault Domain

Rack 2

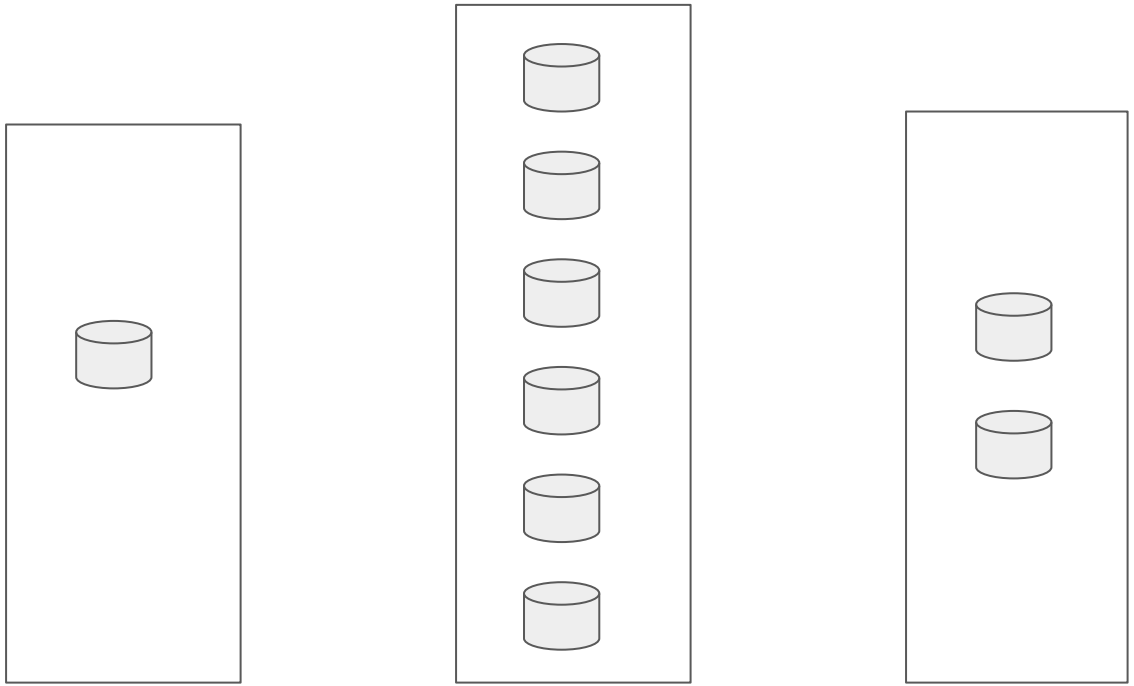
Fault Domain

Use case #1: Fault tolerant scheduling

- Launch highly available applications
  - Stateless and Stateful

- Stateful applications are sensitive to rack placements
  - Replication factor

Bad scheduling

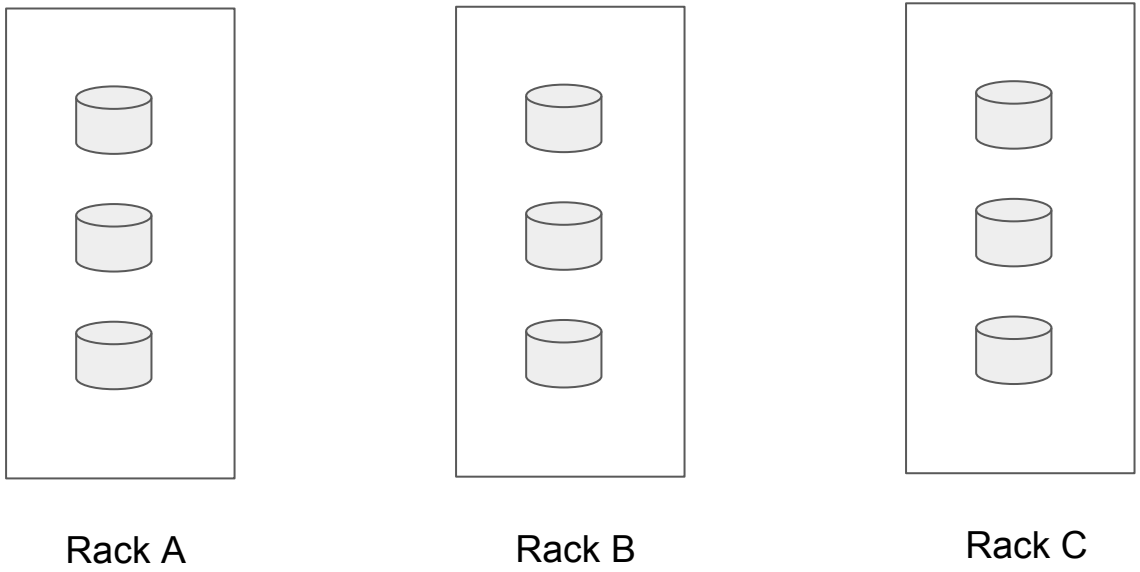


Rack A

Rack B

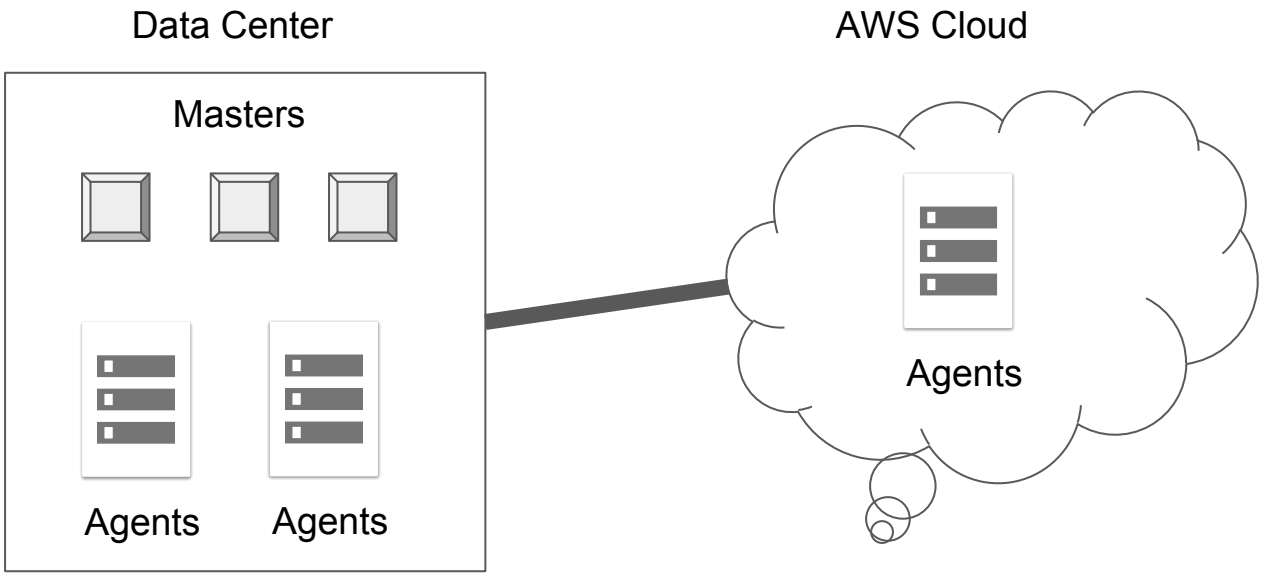
Rack C

Good scheduling



Use case #2: Hybrid Cloud

- Extend on-prem cluster with cloud provider resources on-demand



Hybrid Cloud Scheduling considerations

- Latency
  - Cloud agents have higher latency compared to on-prem agents
- Fault characteristics
  - Cloud providers have their own fault domains (e.g., zones, regions)
- Control
  - Users need to explicitly opt-in to cloud/remote resources

Existing solutions

- User-defined agent attributes + Placement constraints
  - E.g., --attribute={“rack:rack1”, “dc:dc1”}

- Limitations
  - Frameworks and apps are not portable
  - Mesos agnostic

Goals

- Fault domain as a first class primitive
  - Common terminology for frameworks and users
- Support both on-prem and cloud deployments
  - Hybrid as well!
- Sensible default behavior

Solution Overview

- `DomainInfo` protobuf that includes `FaultDomain`

- 2 level hierarchy
  - Regions and Zones

- “REGION_AWARE” framework capability

Fault Domain

```
message DomainInfo {
  message FaultDomain {
    message RegionInfo {
      required string name = 1;
    }

    message ZoneInfo {
      required string name = 1;
    }

    required RegionInfo region = 1;
    required ZoneInfo zone = 2;
  }

  optional FaultDomain fault_domain = 1;
}
```

Fault Domain Hierarchy

- Region
  - Offer the most fault-isolation
  - Inter-region latency is high (50-100ms)
  - Contains one or more zones
  - Maps to “region” in public clouds and “data center” in on-prem

- Zone
  - Inter-zone latency is low (< 10 ms)
  - Moderate degree of fault-isolation
  - Maps to “availability zone” in public clouds and “racks” in on-prem

Terminology

- Default fault domain
  - Fault domain is not configured

- Local Region
  - The region containing masters and *local* agents

- Remote Region
  - Regions other than local region containing *remote* agents

Implementation details

- A new command line flag to configure master and agent with fault domains

```
$ mesos-agent --domain='{
  “fault_domain”: {
    ”region”: {
      ”name”: “region-abc”
    },
    “zone”: {
      “name”: “zone-123”
    }
  }
}’
```

Master changes

- Master’s `DomainInfo` is stored in `MasterInfo`
- Masters are not allowed to span multiple regions
  - Replicated log writes are latency sensitive
- Can span multiple zones within a region
  - Recommended for fault tolerance

Agent changes

- Agent's `DomainInfo` is stored in `AgentInfo`

- Master includes agent's DomainInfo inside `OfferInfo`
  - Allows frameworks to do fault domain aware scheduling

- Configuring an agent with a fault domain requires a drain
  - Will not be required in Mesos 1.5

Framework changes

- Frameworks need to register with REGION_AWARE capability
  - Without this capability offers from remote agents are not sent
  - Guards against legacy frameworks launching tasks in remote regions by accident

- Recommendation: Frameworks should exposed remote region scheduling explicitly to users

Examples with Marathon

- Schedule my app in a remote region
  - Placement constraint: [@region, IS, “aws-east1”]

- Spread my app evenly across zones for HA
  - Placement constraint: [@zone, GROUP_BY, 3]

Upgrades

- Masters can be in “mixed” fault domain mode
  - Some have fault domain configured and some don’t

- Masters must be updated first before agents
  - Fault-domain configured agents are not allowed to register with non-configured Masters
  - Guards against ***remote*** agent accidentally being considered ***local***

Upgrades

	Agent: Domain Set	**Agent: No Domain Set**
Master: Domain Set	If `master.region != agent.region`, only offer to `REGION_AWARE` frameworks	Agent eligible to be offered to all frameworks as normal
Master: No Domain Set	*Configuration error*; agent registration attempt will be ignored	Agent eligible to be offered to all frameworks as normal

State of the feature

- Fault domains are available since Mesos 1.4
  - Experimental

- Agent domain re-configuration without drain will be available in Mesos 1.5
  - Going from default domain to configured domain
  - Going from configured domain to a different configured domain
  - Bonus feature: Changing attributes!

Acknowledgements

- Neil Conway
- Ben Hindman
- Anand Mazumdar
- Joris Van Remoortere

Thank you

Design doc