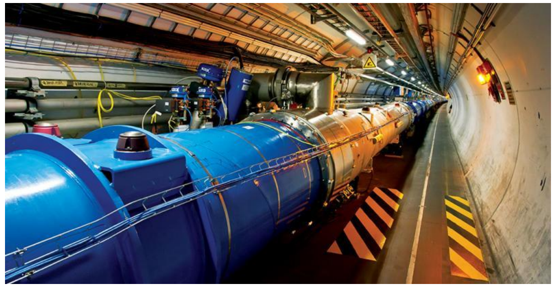
PREEMPTIBLE INSTANCES

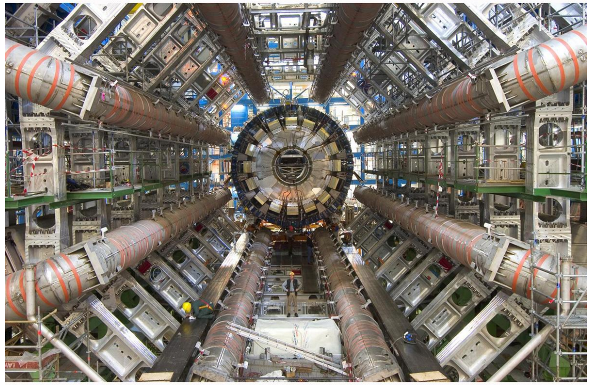
THEODOROS TSIOUTSIAS
theodoros.tsioutsias@cern.ch

17/05/2018

Outline

- Introduction
- Scenarios
- The problem
- The solution
- Openstack Preemptible Instances
- Reaper Implementation





Introduction

Quotas

Cloud Computing gives the illusion of infinite capacity

- Quota System:

1. Sets limits on resources
2. Ensures everyone makes use of their fair share of the resources

- Operators use **quotas** per project to:

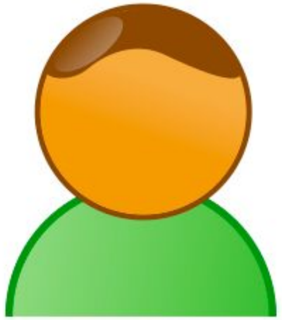
1. Prevent system capabilities from being exhausted without notification
2. Manage the resource allocations
3. Avoid “Over-committing” resources
4. Reserving Resources for operations with higher priority

Scenario 1/3

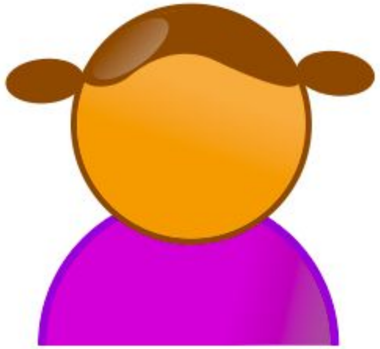
Micro view

- Alice has available resources
- Bob’s quota is exhausted and he needs more computing power

Bob could tell Alice: “Please let me use the your idle resources.”



Bob



Alice

Scenario 2/3

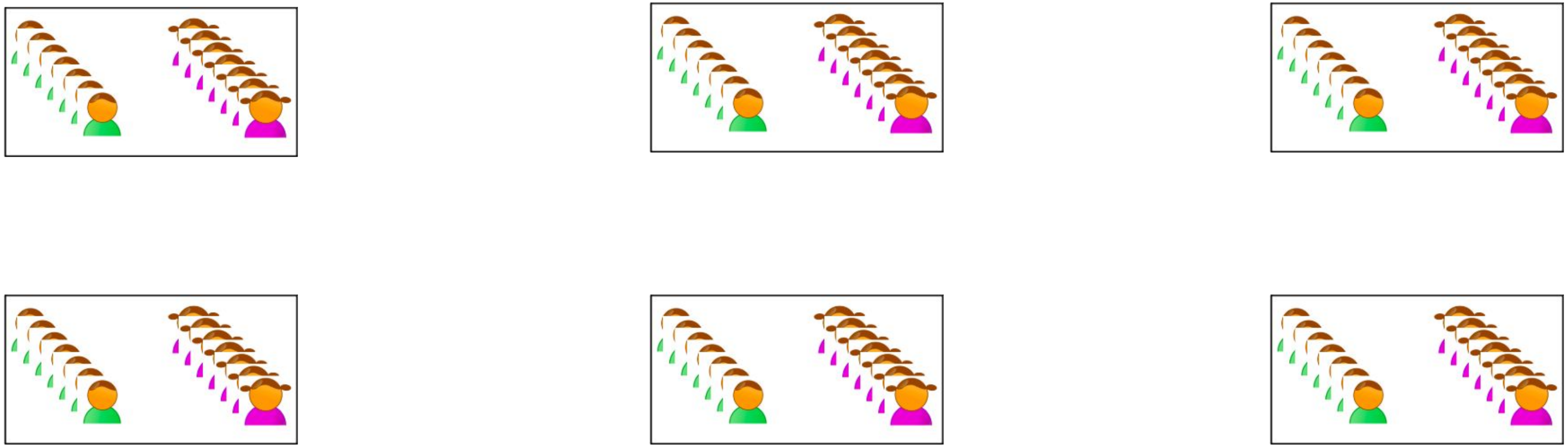
But what happens in a department?

- Imagine a now department in a company with tens of employees having the same issue

Scenario 3/3

Or even worse, in a big organization?

- The same race exists between different departments in an organization



Problem

Keeping track of the available resources

- As the organization grows, the amount of unused/idle resources may increase as well.

- Quotas are hard limits:

> Even if there are free resources, they cannot be allocated to a project whose quota is exceeded

- This leads to a reduction in cloud utilization:

> There are resources in idle state!

Solution

- Introducing the concept of Preemptible Instances:
  - created even after the quota for a project is exceeded
  - use idle resources
  - terminated as soon as the resources are needed for higher priority tasks

- The result of this:
  - handling the demand for extra resources
  - increasing the cloud utilization
  - maintaining the fair sharing of the infrastructure

Openstack Preemptible Instances

Starting things simple

- Use dedicated projects:
  - These projects have unlimited quota
  - Instances in these projects are preemptible

- Introduce a Reaper service:
  - Orchestrator for the preemptible instances
  - Applies strategies to free up the resources

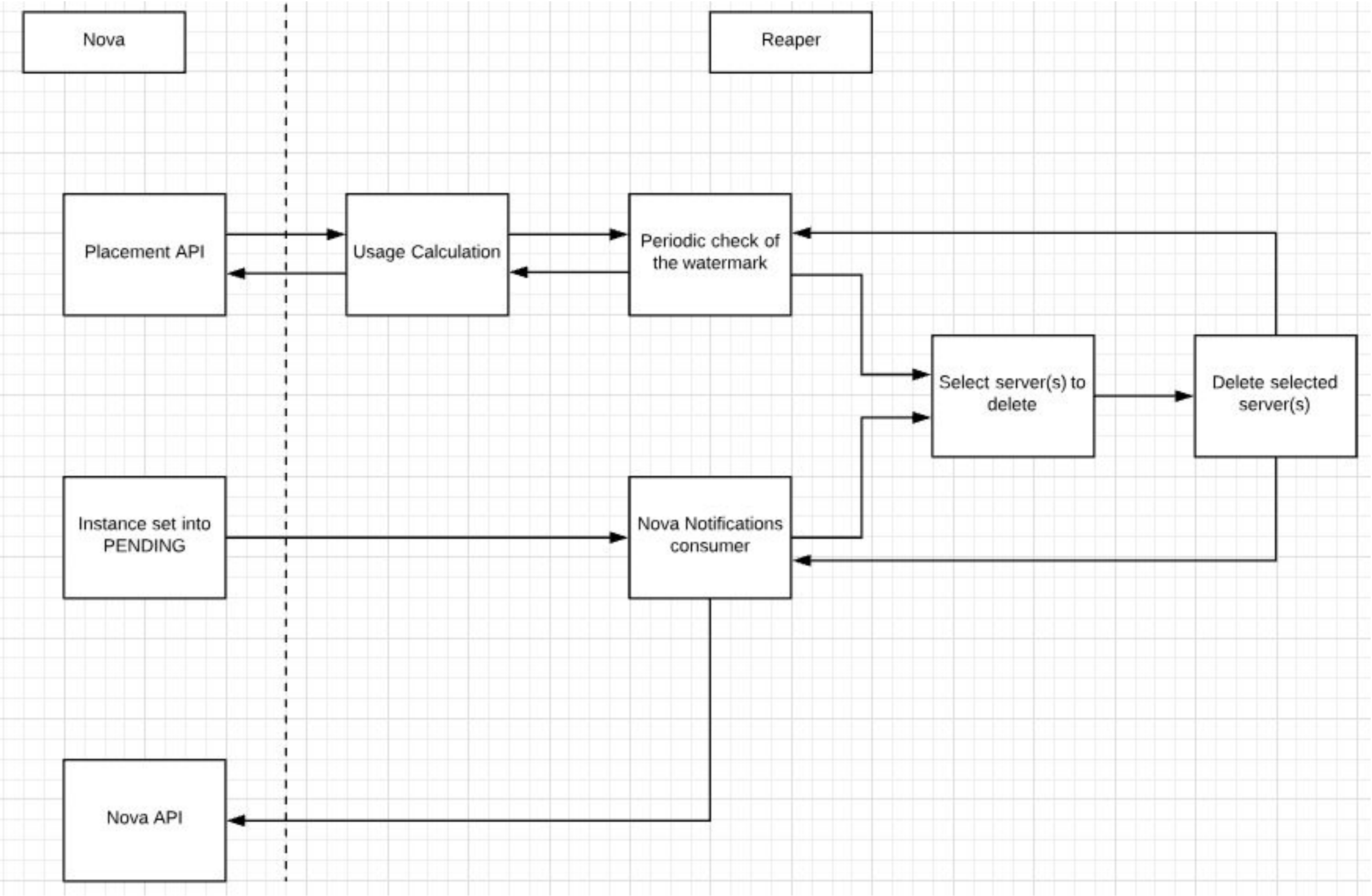
Reaper Implementation 1/3

Modes of operation

- Watermark:
  - The operator defines a margin (e.g. 95% max usage)
  - Periodically checks if we respect the watermark

- Build failure due to lack of resources:
  - Introduce the PENDING instance state
  - Instances that fail for this reason, go to PENDING instead of ERROR
  - The service is notified and it tries to free the requested resources
  - Depending on the outcome:
    - Rebuild to consume the resources that are freed up
    - Reset instance state to ERROR

Reaper Implementation 2/3


Nova
Reaper
Placement API
Usage Calculation
Periodic check of the watermark
Select server(s) to delete
Delete selected server(s)
Instance set into PENDING
Nova Notifications consumer
Nova API

Reaper Implementation 3/3

Proposed changes to Nova

- (spec) Add PENDING instance state:
  - https://review.openstack.org/#/c/554212/
- (spec) Enable rebuild for instances in cell0
  - https://review.openstack.org/#/c/554218/
- Add scheduling notification:
  - https://review.openstack.org/#/c/566470/
- Introduce PENDING instance state
  - https://review.openstack.org/#/c/566473/

Summary

- By providing Preemptible instances:
    - Maximize cloud utilization
    - Better handling of the increased demand for resources
    - For public clouds: Monetize unreserved resources

- Openstack Preemptible Instances
    - Dedicated projects with unlimited quota
    - Preemptible orchestrator

QUESTIONS?

theodoros.tsioutsias@cern.ch