Synergy

a service for optimizing the resource allocation in cloud based environments

Lisa Zangrando
INFN Padova
Overview

**Synergy**

cloud service developed in the context of the INDIGO-DataCloud European project which aims to develop a new cloud software platform for the scientific community

- https://www.indigo-datacloud.eu/

**Main objective**

enable a more effective and flexible resource allocation and utilization in open Clouds such as OpenStack
The issue

- **In the current OpenStack model:**
  - the user request fails (and is lost) if no resource can satisfy it
  - static partitioning: the resource allocation to the user projects can be done only by granting fixed quotas
  - one project cannot exceed its own quota even if there are unused resources allocated to other projects
  - very low global efficiency and an increased cost
  - 20 years old problem solved by batch systems
  - **INDIGO addresses this issue through Synergy**
Synergy

- Synergy is the extensible general purpose management service designed by the National Institute for Nuclear Physics (INFN) for executing tasks in OpenStack
- It is composed by a collection of pluggable managers
The managers

- Managers provide specific and independent pluggable functionality (task) executed periodically or interactively through a RESTful API
- They can interact with each other or with different OpenStack services in a loosely coupled way
The manager interface

The following piece of code shows the Python abstract base class that every manager has to extend:

```python
class Manager(Thread):
    def getName(self): # returns the manager name
    def getStatus(self): # returns the manager status
    def isAutoStart(self): # is AutoStart enabled or disabled?
    def setup(self): # allows custom initialization
    def destroy(self): # invoked before destroying
    def execute(self, cmd): # executes user command synchronously
    def task(self): # executed periodically at fixed rate
```
Advanced scheduling

- Functionality implemented as collection of specific managers
  - different managers can coexist

- Synergy adopts a resources provisioning model based on a fair-share algorithm to maximize the resources usage in OpenStack
  - it guarantees that resources are equally distributed among users

- It provides a persistent priority queuing mechanism for handling user requests that can not be immediately fulfilled
Resource allocation

- Synergy allows the IaaS administrators to allocate a subset of resources (dynamic resources) to be shared among different projects, besides the ones statically partitioned.

- Static resources consumed according the standard OpenStack model.

- Dynamic resources are handled by Synergy which allows the definition of fair-share policies:
  - list of projects allowed to access to the dynamic resources
  - shares on resource usages for the relevant projects
  - max lifetime for Virtual Machines (this is needed to enforce the fair-sharing)
The scheduler managers

Five managers implement the fair-share based scheduling model

- **FairShare-Manager**: implements the main fair-share scheduling logic
  - It dynamically assigns the proper priority value to every user request
  - fair-share algorithm based on the SLURM Priority MultiFactor strategy
- **Queue-Manager**: provides a persistent priority queue service
- **Quota-Manager**: it is in charge of handling the quota of all projects
- **Nova-Manager**: it interacts with Nova components
- **Keystone-Manager**: it interacts with the Keystone service
High level architecture

- Synergy
- Quota M.
- Keystone
- Keysyone M.
- Nova M.
- Queue M.
- FairShare M.
- database
Low level architecture
Remark

- Synergy will not replace any existing OpenStack service (e.g. Nova)
  - it may complement their functionality as an independent service
- No changes in the existing OpenStack components are required
- Synergy allows the coexistence of the new advanced resource allocation and the standard one of OpenStack (i.e. FCFS)
Test (1/2)

- First Synergy prototype tested at INFN OpenStack/Juno production site of EGI Federated Cloud
- Two testing projects set up in fair-share mode: prj_A (70%) and prj_B (30%)
  - it is assumed that all users have the same share
- Dynamic resources: 20% of total

**Total Resources**
- Total resources: 80
- Static: 80
- Dynamic: 20

**Dynamic Resources**
- Shares:
  - prj_A: 70
  - prj_B: 30
Test (2/2)

- Automatic robot instantiates VMs at the same constant rate on both projects by using different users.
- > 20,000 VMs executed over two days, Cirros images with different flavors, VM lifetime limited to 5 min to speed up testing.
- Project resource usage accounted at the end of each period measured to be as expected (70% and 30%) within 1%.
- Tests coexisted and did not interfere/degrade normal operations of other production projects/VOs (not involved in fair-share computation).
The development status

- Synergy will be part of the first Indigo release
  - July 2016
- Code in launchpad
  - https://launchpad.net/synergy-service
  - https://launchpad.net/synergy-scheduler-manager
- On-going integration with the OpenStack Continuous Integration system
Next steps

- Implement a complete test suite
- Test Synergy in the bigger CNRS's production site
- Update Synergy for supporting the latest OpenStack versions
- Improve the fair-share algorithm by implementing the SLURM Fair Tree
- Improves the resource usage calculation by considering even CPU performance measured with HEPSPEC 2006 (HS06) benchmark (not only the CPU wall-clock time)

- The ultimate goal is to have it integrated in the Official OpenStack distribution
Questions?