Cloud Tracing

From high to low level

Yves J. BATIONO

Msc. candidate

Supervised by: Michel Dagenais

POLYTECHNIQUE MONTRÉAL

LE GÉNIE EN PREMIÈRE CLASSE

yves-junior.bationo@polymtl.ca

Outline

Introduction

Research objectives

Analyse Model

Openstack nova

Virtualization layer

Kernel layer

VM Live Migration case

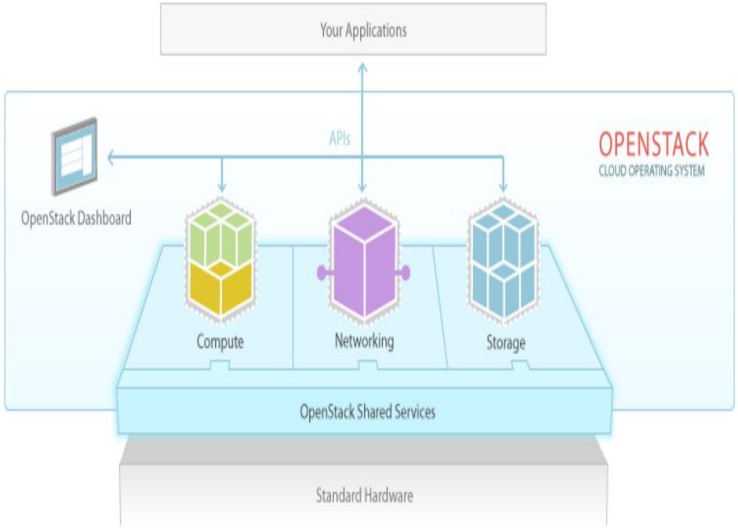
Future work

Introduction

- Complexity of cloud services
- Consumers experiment some services latencies
- Where to start troubleshooting ?
- Complete view of the cloud environment
- Correlate informations from different nodes

Research objectives

- Focused on Openstack platform
  - Infrastructure as a service
  - get system performance
- Analyse Openstack Services
  - Nova: Compute resources
  - Neutron: Networking as a service

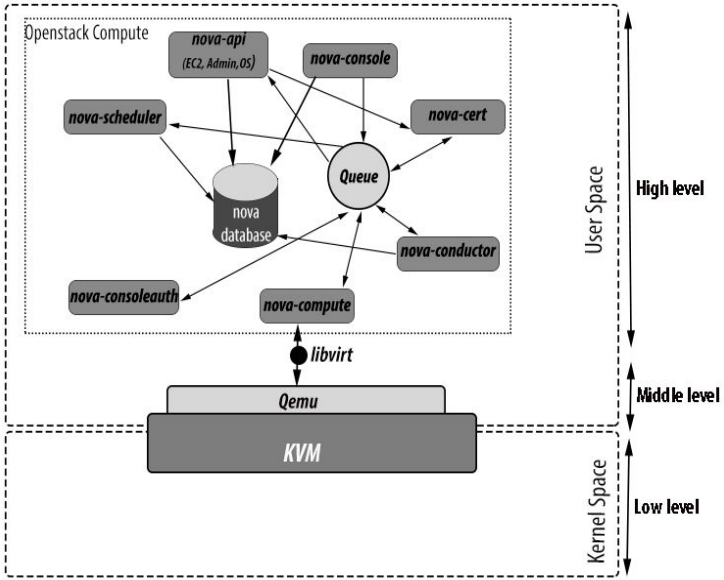


Cloud system architecture
https://www.openstack.org/software/

Research objectives

- Analyse cloud infrastructure
  - services, virtual resources ...
- Show openstack service efficiency
  - show interaction and service bottleneck
- Correlate cloud nodes information
- Understand execution failures
  - find weak link

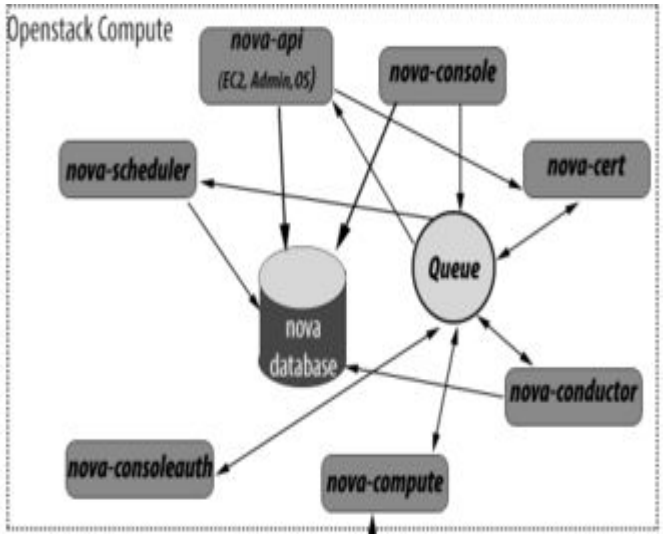
Analyse Model



- Multilevel tracing:
  - High level: **Nova**
  - Middle level: **Qemu**
  - Low level: **Kernel**
- Complete view of the Cloud environment
  - gather multi layer trace from all nodes
- High level
  - users actions
  - services interactions
  - Resources usage per tenants
  - Not enough in somes cases
  - look for more details in low level (qemu/kvm, kernel)

Openstack nova

Nova architecture



Collection of services

- select host for VM creation (scheduler)
- database access (conductor)
- handle VM lifecycle (compute)
- Hub for communication (RabbitMQ)

Openstack nova

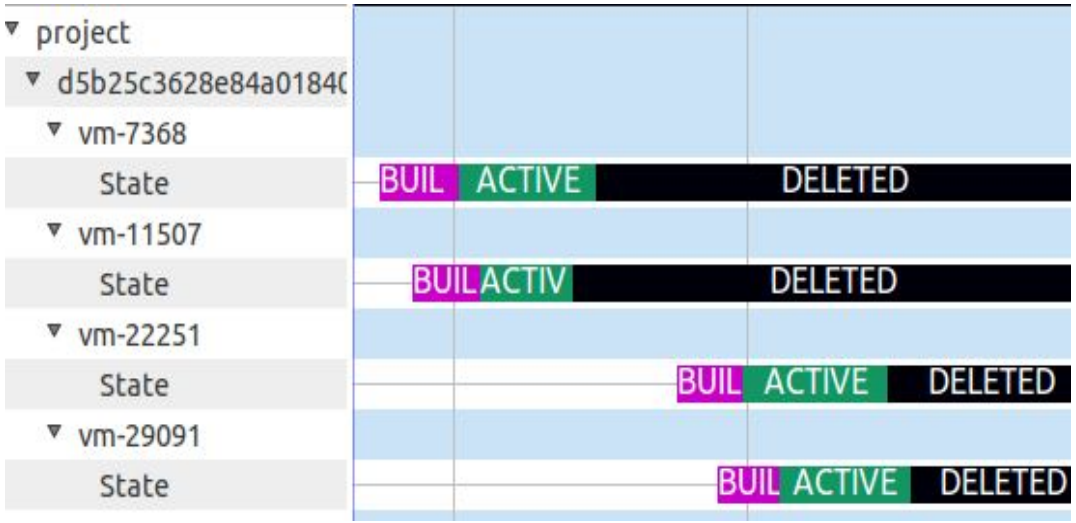
How to trace nova ?

- Based on nova logging activity
  - Lttng python gets logs ouputs
- Write the log in a usefull format
  - we use JSON format to provide: event_type, instance, context, message

Openstack nova

Nova tracing purpose

- VM state
  - VM lifecycle
  - VM network setup
  - VM migration execution

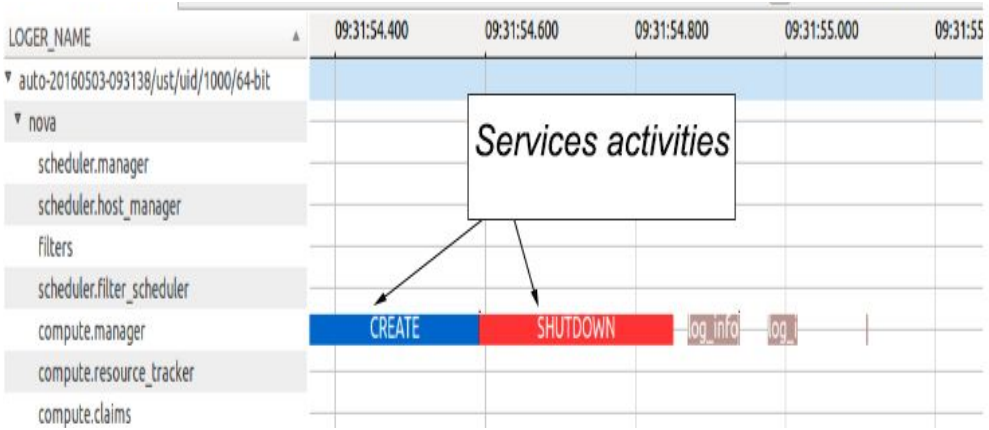


VM states View

Openstack nova

Nova tracing purpose

- Services performances analysis



- Time to perform or execute requests
- scheduling algorithm
- load balancing among services

Openstack nova

Nova tracing purpose

- Request flow analysis
  - Communication performance through RABBITMQ

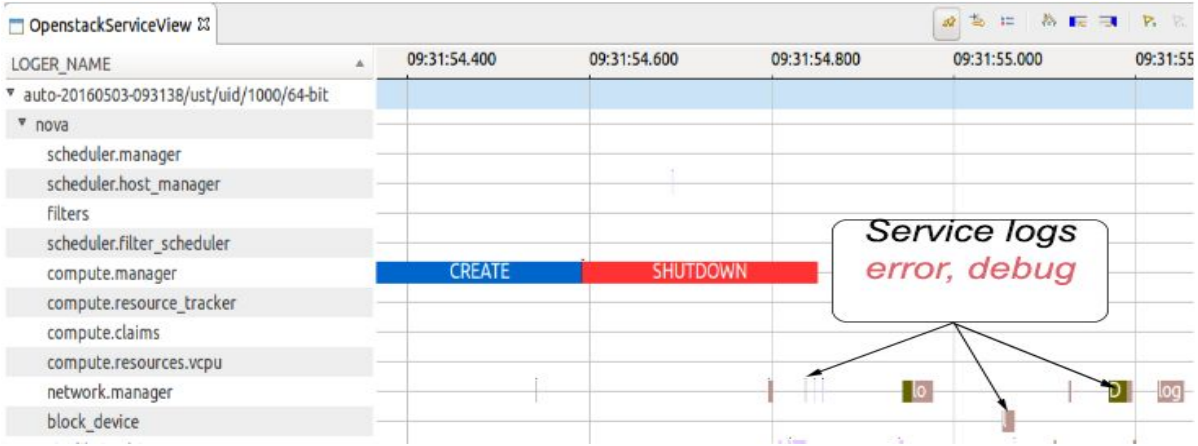
    bottleneck in the messaging hub
  - Activity process

    check that no service is waiting unnecessarily for another.
    request is handled as expected by the services.

Openstack nova

Nova tracing purpose

- Troubleshooting
  - analyse log to pinpoint error cause
  - find critical service in the cloud system



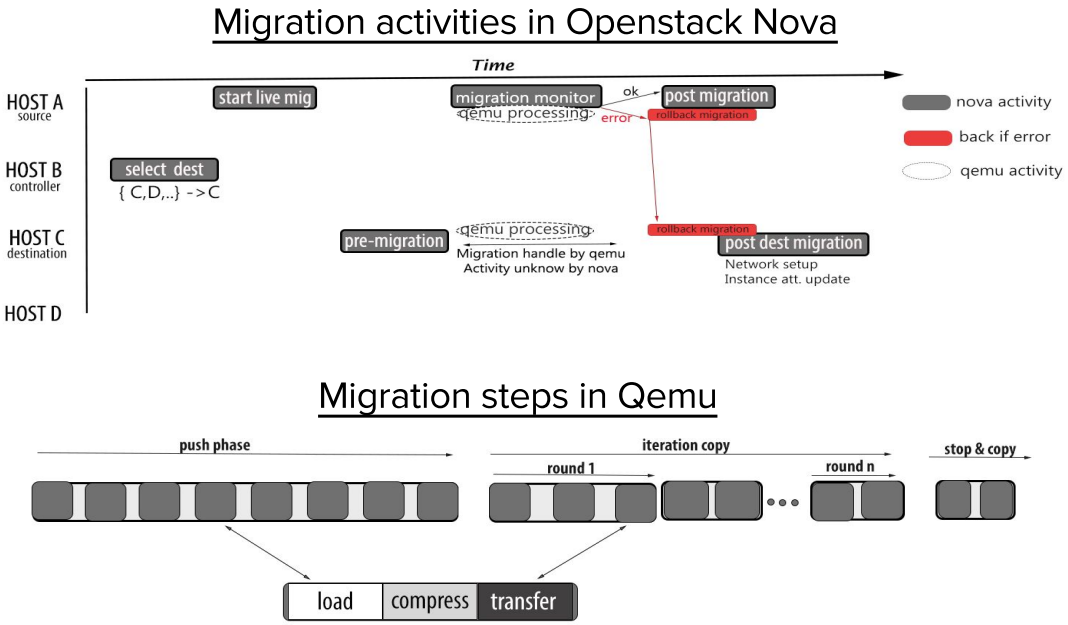
Virtualization layer

- What is Qemu ?
  - emulate hardware device: net, disk
  - Used with KVM, Xen, ...
  - Handle VM request to the hardware
- How to trace Qemu ?
  - already instrumented (support Lttng, DTrace, ...)
- Why tracing Qemu?
  - show VM internal process
  - memory leaks Qemu not freeing memory

Kernel layer

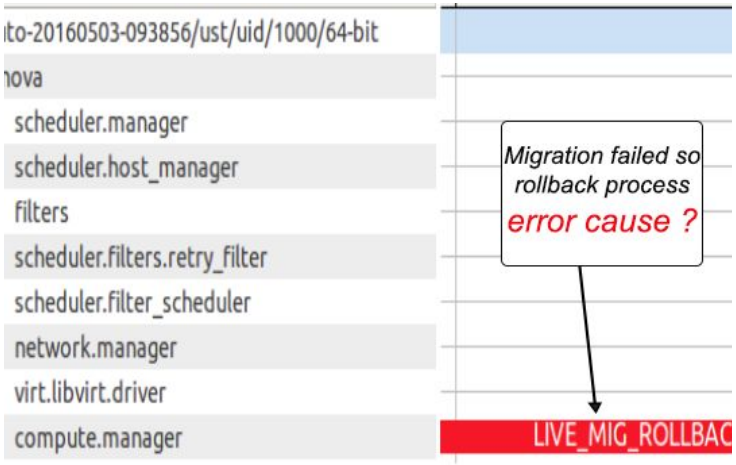
- Virtual machine is a simple process using compute resources
  - Analysis is made like for any process
- A lot of features available in Lttng and Tracecompass
  - Control flow, critical path, resource views
- Provide fine-grained data
  - Resources usage, services latencies
- Resources sharing cause interferences between virtual machines
  - CPU contention, memory and Network Interferences

VM Live Migration Case



- Nova does not have access to migration internal process.
- Only Qemu can report about migration copy step
- high level process not enough to understand some cases

VM Live Migration Case

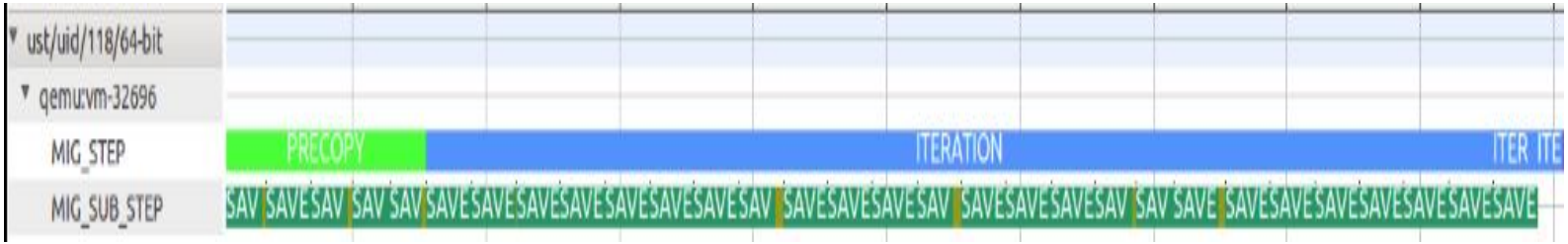


Nova service view

Why migration fail?

- Bug in virtualizaton layer ?
- Migration timeout reached ?
- Network or CPU contention interrupt data copy to the destion host ?
- VM has a high workload: data copy never end ?
- Seek for detail in low level

VM Live Migration Case



Migration step in Qemu

- We only get 2 steps:Precopy, iteration copy
- Stop&copy is missing as migration never complete
- look for dirty pages rate for more details about the migration

VM Live Migration Case

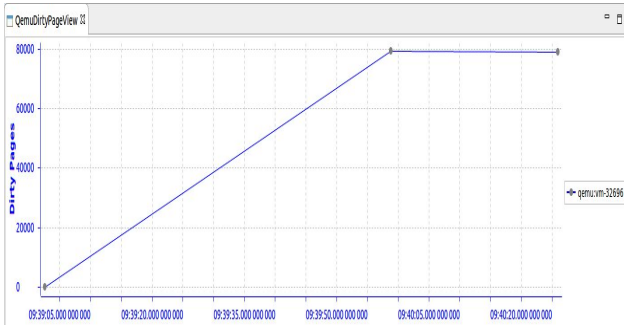


Fig.1 Abnormal Dirty page rate curve

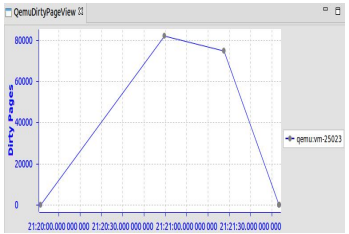


Fig.2 Normal Dirty page rate curve

Fig.1 (our case)

- Curve appearance is different from normal migration one
- The curve does not converge to 0
- VM has a high workload
- page dirtying rate is higher than the data copy to the destination host
- Qemu cannot transfer quickly the memory
- CPU usage from kernel trace shows a high CPU usage for the VM process

VM Live Migration Case

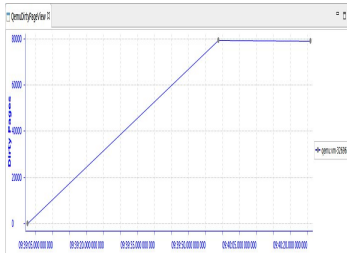


Fig.1 Abnormal Dirty page rate curve

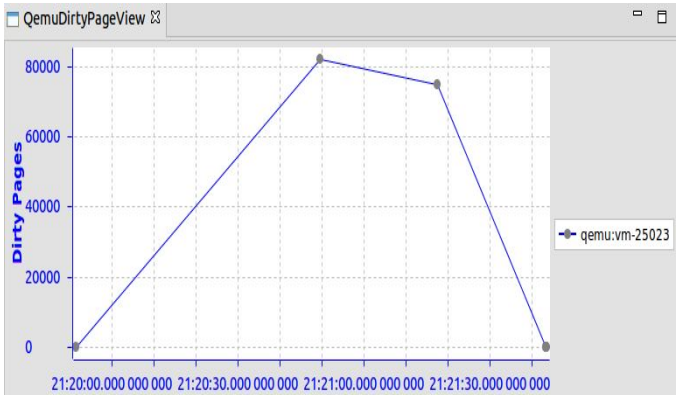


Fig.2 Normal Dirty page rate curve

Fig.2

- Show a normal view of a succeded migration
- curve converging to 0 means that the page dirtying rate decreases over time

VM Live Migration Case

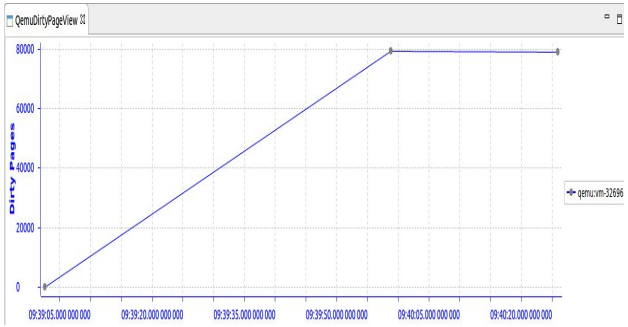


Fig.1 Abnormal Dirty page rate curve

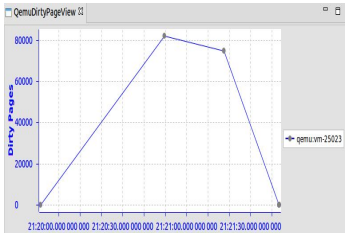


Fig.2 Normal Dirty page rate curve

Possible solutions

- increase the priority of the VM process for resource usage
- decrease if possible other process priority to the computing resource

Future work

- Network functions virtualization
  - DNS, firewall, NAT deployment and managing
- Neutron project tracing
  - project to provide networking as a service
  - provide API abstraction for port, subnet, network.
- Opendaylight services analyses
  - Controller infrastructure for SDN deployement

QUESTION ?

References

https://wiki.openstack.org/wiki/Nova
https://pdfs.semanticscholar.org/2f2c/dd7b0c98b5e43b61272d2ac3ebb5cd29041d.pdf
https://projects.eclipse.org/projects/tools.tracecompass
http://lttng.org/docs/#doc-python-application
https://wiki.openstack.org/wiki/Neutron
https://www.opendaylight.org/
https://www.openstack.org/software/