Distributing OpenStack on top of a Key/Value store

Jonathan Pastor
(Inria, Ascola, Ecole des Mines de Nantes)

PhD candidate, under the supervision of Adrien Lebre and Frédéric Desprez
Discovery Initiative

- The Discovery Initiative targets the delivery of Locality based Utility Computing (LUC).
  [http://beyondtheclouds.github.io](http://beyondtheclouds.github.io)

- Leveraging network backbones: extend any point of presence of network backbones with UC servers.

- Bring UC as close as possible to the end users
  ⇒ more efficient (latency aware application ++).
In short

• Building an infrastructure composed of 100/1000 remote sites (10/100 servers per site).

• Target collaboration without any broker.

• But without reinventing the wheel to limit the development effort, we propose to leverage existing successful mechanisms: **OpenStack**.
OpenStack

• Open source IaaS manager with a large community.

• Composed of several services dedicated to each aspect of a cloud.

• Services collaborate via:
  • Messaging queue (RabbitMQ)
  • Shared database (MySQL)
Distributing OpenStack

- Few proposals to federate/operate distinct OpenStack DCS
  - 'Flat' approach:
    - Leveraging HaProxy and Galera (Active replication)
      ⇒ *Complexity and scalability issues*
  - Hierarchical approaches:
    - Cells based (CERN: 3 Sites / 50K cores)
      ⇒ *the top cell is a SPOF*
    - Cascading OpenStack.
- You know others!? please mail us!
Leveraging Key/value stores

- Alternate solutions exists for storing states over a *highly distributed infrastructure* (more relevant?).

- **Key/value store** in place of the classical relational database (with active replication)?

- We focus on the compute service (Nova) by *studying* the replacement of MySQL by Redis.
Replacing MySQL in Nova?

- Nova services don’t access directly the database: they communicate with nova-conductor, which in turn calls functions located in: `nova/nova/db/api.py`

```python
def instance_get(context, instance_id, columns_to_join=None):
    """Get an instance or raise if it does not exist."""
```

- These functions manipulate databases and return python objects from:

```bash
/Users/jonathan/vm3/stack/nova/nova/objects
__init__.py  external_event.py  instance_fault.py  quotas.py
aggregate.py  fields.py  instance_group.py  security_group.py
base.py  fixed_ip.py  instance_info_cache.py  security_group_rule.py
block_device.py  flavor.py  keypair.py  service.py
compute_node.py  floating_ip.py  migration.py  virtual_interface.py
dns_domain.py  instance.py
ec2.py  instance_action.py
```
Replacing MySQL

- Nova already includes the possibility of using a different DB backend through ORMs (only SQLAlchemy yet).

- `nova/nova/db/api.py` => provides functions that manipulate the DB via an ORM.

- It means that we can add a new backend:
How Nova is linked with MySQL?

- Nova Scheduler
- Nova Network
- Nova Compute
- Nova Conductor
- db.api

Relational

SQLAlchemy

MySQL DB
Rome

• Relational Object Mapping Extension for key/value stores.
  https://github.com/badock/rome

• Enables the query of KVS (key/value store) with the same interface as SQLAlchemy.

• Enables to use KVS in Nova by limiting the break of code.
Integrating Nova with Redis
Proof of concept

• We can use the Key/Value as the cervical spin of a deployment.

• The Key/value store is clustered on controllers

• Compute nodes connect to the Key/value cluster.
Experiments

- Preliminary experiments have been conducted on Grid’5000.

- **mono-site experiments**: to evaluate the overhead of using REDIS and the network impact.

- **multi-site experiments**: To determine the impact of latency.

- Ask for the creation of 500 VMs, fairly distributed on each controller.
Preliminary results

- **Time measured for creating 500 VMs in parallel.**

- Experiments performed on servers with homogeneous hardware.

- For a fair comparison (routing issues can disturb Galera): *use servers on the same site (Rennes).*

- Clusters were simulated by adding latency between nodes with TC.

  - *We followed configuration advised by OpenStack multi-site documentation.*

### 10 ms intersite latency

<table>
<thead>
<tr>
<th></th>
<th>Redis</th>
<th>MySQL (no replication)</th>
<th>Galera</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 cluster (no replication)</td>
<td>298</td>
<td>357</td>
<td>-</td>
</tr>
<tr>
<td>2 clusters</td>
<td>271</td>
<td>209</td>
<td>2199</td>
</tr>
<tr>
<td>3 clusters</td>
<td>280</td>
<td>157</td>
<td>3243</td>
</tr>
<tr>
<td>4 clusters</td>
<td>263</td>
<td>139</td>
<td>2011</td>
</tr>
</tbody>
</table>

### 50 ms intersite latency

<table>
<thead>
<tr>
<th></th>
<th>Redis</th>
<th>MySQL (no replication)</th>
<th>Galera</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 cluster (no replication)</td>
<td>298</td>
<td>357</td>
<td>-</td>
</tr>
<tr>
<td>2 clusters</td>
<td>723</td>
<td>268</td>
<td>1361</td>
</tr>
<tr>
<td>3 clusters</td>
<td>518</td>
<td>210</td>
<td>2202</td>
</tr>
<tr>
<td>4 clusters</td>
<td>427</td>
<td>203</td>
<td>1253</td>
</tr>
</tbody>
</table>
Future work

• Increase the scale of the experiments (more nodes, more sites, more VMs …).

• Make Rome works with other NoSQL DB (*Cassandra*).

• Apply the same strategy with remaining services: *Glance, Neutron*?

• Propose leads to bring locality in an OpenStack deployment (*host aggregates?*)
Thanks for your attention

Questions?

http://beyondtheclouds.github.io/
adrien.lebre@inria.fr / jonathan.pastor@inria.fr
and many others!


Federation?

• Is the LUC-OS a distributed cloud federation broker?
How data are stored?

- Each entity (e.g., VM, Image, SecurityGroup) is modelled by an entity class.
- With ROME, each entity class corresponds to a bucket.
- With current implementation, the data model is quite normalised.
How data are represented?

Python object with relationships

```json
[
  {
    class: "FOO",
    id: 1,
    name: "A",
    bar: {
      class: "link",
      target: "BAR",
      id: 1
    }
  },
  {
    class: "BAR",
    id: 1,
    name: "B",
    foo: {
      class: "link",
      target: "FOO",
      id: "1"
    }
  }
]
```

JSON representation
How links are modelled?
How data are stored?
How data are queried?

- if no "multiget" available, query of objects is sequential.

- if "multiget" available, query of objects can be parallelised.
Host aggregates for locality?

- “This information can be used in the scheduler to enable advanced scheduling, to set up xen hypervisor resources pools or to define logical groups for migration”

http://docs.openstack.org/developer/nova/devref/aggregates.html

- Host aggregates could be set-up dynamically in an automatic way (Vivaldi?).