OpenStack Neat – Dynamic Consolidation of Virtual Machines: Blueprint

Anton Beloglazov

26th of July 2012

Contents

1 Summary

OpenStack Neat is a project intended to provide an extension to OpenStack implementing dynamic consolidation of Virtual Machines (VMs) using live migration. The major objective of dynamic VM consolidation is to improve the utilization of physical resources and reduce energy consumption by re-allocating VMs using live migration according to their real-time resource demand and switching idle hosts to the sleep mode. For example, assume that two VMs are placed on two different hosts, but the combined resource capacity required by the VMs to serve the current load can be provided by just one the hosts. Then, one of the VMs can be migrated to the host serving the other VM, and the idle host can be switched to the sleep mode to save energy.

Apart from consolidating VMs, the system should be able to react to increases in the resource demand and deconsolidate VMs when necessary to avoid performance degradation. In general, the problem of dynamic VM consolidation can be split into 4 sub-problems:

1. Deciding when a host is considered to be underloaded, so that all the VMs should be migrated out, and the host should be switched to a low-power mode, such as the sleep mode.
2. Deciding when a host is considered to be overloaded, so that some VMs should be migrated from the host to other hosts to avoid performance degradation.
3. Selecting VMs, which should be migrated from an overloaded host out of the full set of the VMs currently served by the host.
4. Placing VMs selected for migration to other active or re-activated hosts.

This work is a part of PhD research conducted within the Cloud Computing and Distributed Systems (CLOUDS) Laboratory at the University of Melbourne. The problem of dynamic VM consolidation considering Quality of Service (QoS) constraints has been studied from the theoretical perspective and algorithms addressing the sub-problems listed above have been proposed [1], [2]. The algorithms have been evaluated using CloudSim and real-world workload traces collected from more than a thousand PlanetLab VMs hosted on servers located in more than 500 places around the world.

The aim of the OpenStack Neat project is to provide an extensible framework for dynamic consolidation of VMs within OpenStack environments. The framework should provide an overall architecture and abstract interfaces of components implementing the 4 decision-making algorithms listed above. The framework should allow configuration-driven plugging in particular implementations of the decision-making algorithms. The implementation of the framework will include the algorithms proposed in our previous works [1], [2].

2 Release Note

The functionality covered by this project will be implemented in the form of services separate from the core OpenStack services. The services of this project will interact with the core OpenStack services using their public APIs. It will be required to create a new Keystone user within the `service` tenant. The project will also require a new MySQL database for storing information on the host configuration, VM placement, and CPU utilization by the VMs. The project will provide script for automated initialization of the database. The service of the project will need to be run on the management and compute hosts.

3 Rationale

4 User stories

5 Assumptions

Glance ''stores" the server template and metadata map; Nova must ''implement" the server template.

6 Design

This is just one possible design for this feature (keep that in mind). At its simplest, a server template consists of a core image and a ''metadata map". The metadata map defines metadata that must be collected during server creation and a list of files (on the server) that must be modified using the defined metadata.

Here is a simple example: let's assume that the server template has a Linux server with Apache HTTP installed. Apache needs to know the IP address of the server and the directory on the server that contains the HTML files.

The metadata map would look something like this:

```
metadata {
 IP_ADDRESS;
 HTML_ROOT : string(1,255) : "/var/www/";
}
map {
 /etc/httpd/includes/server.inc
}
```

In this case, the {{{metadata}}} section defines the metadata components required; the {{{map}}} section defines the files that must be parsed and have the metadata configured. Within the {{{metadata}}} section, there are two defined items. {{{IP_ADDRESS}}} is a predefined (built-in) value, and {{{HTML_ROOT}}} is the root directory of the web server.

For {{{HTML_ROOT}}}, there are three sub-fields: the name, the data type, and (in this case) the default value. The token {{{required}}} could be used for items that must be supplied by the user.

When the server is created, a (as-yet-undefined) process would look at the files in the {{{map}}} section and replace metadata tokens with the defined values. For example, the file might contain:

```
<VirtualHost {{IP_ADDRESS}}:*>
  DocumentRoot "{{HTML_ROOT}}";
</VirtualHost>
```

7 Implementation

This section should describe a plan of action (the "how") to implement the changes discussed. Could include subsections like:

7.1 UI Changes

Should cover changes required to the UI, or specific UI that is required to implement this

7.2 Code Changes

Code changes should include an overview of what needs to change, and in some cases even the specific details.

7.3 Migration

Include:

- data migration, if any
- redirects from old URLs to new ones, if any
- how users will be pointed to the new way of doing things, if necessary.

8 Test/Demo Plan

This need not be added or completed until the specification is nearing beta.

9 Unresolved issues

This should highlight any issues that should be addressed in further specifications, and not problems with the specification itself; since any specification with problems cannot be approved.

10 BoF agenda and discussion

Use this section to take notes during the BoF; if you keep it in the approved spec, use it for summarising what was discussed and note any options that were rejected.

11 References

[1] A. Beloglazov and R. Buyya, "Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in Cloud data centers," *Concurrency and Computation: Practice and Experience (CCPE)*, 2012.

[2] A. Beloglazov and R. Buyya, "Managing Overloaded Hosts for Dynamic Consolidation of Virtual Machines in Cloud Data Centers Under Quality of Service Constraints," *IEEE Transactions on Parallel and Distributed Systems (TPDS)*, 2012.