www.ijecs.in
International Journal Of Engineering And Computer Science ISSN:2319-7242
Volume 4 Issue 5 May 2015, Page No. 11778-11782

Heat Template Generator in Openstack

Prashant Sangale, Bharati Ainapure Kedar V. Kulkarni, Shrikant Gangade, Sujay Bothe
Computer Department
MIT College of Engineering (MITCOE)
Pune, India.
Email: sangaleprashant2812@gmail.com
Assistant Professor
Computer Department
MIT College of Engineering (MITCOE)
Pune, India.
Email: bharati.ainapure@mitcoe.edu.in

***Abstract*—Cloud computing seems to be recent technology, but the term's origin is quite unclear. In this paper, we present an overview of a project namely Heat Template Generator in Openstack. Our aim is to equip our readers with the overall functioning of the Heat component of Openstack and help them to understand the current features that are supported by the same. We then also briefly discuss about what is the current scenario about template writing and how this project will improves the productivity. We conclude by providing brief account of how this Template Generator is future of Orchestration in cloud computing.**

***Keywords*—Openstack, Open source, Cloud Computing, Templates, Heat, Template Generator, Orchestration, Horizon**

I. INT RODUCTION

Cloud computing using Openstack is a relatively young field of endeavor in Computer science which deals with all major aspects of computers and relevant technologies such as virtualization, networking, cloud storage, spawning virtual servers, parallel computing etc. Cloud computing can be seen as a technology which started a drift in a way of utilizing computing resources.

In today's date, there are number of tech giants who provide cloud solutions. The names like Microsoft Azure [12], IBM Cloud [13], Amazon EC2 [14], etc. top the list. These all are mainly proprietary, meaning that they are developed and maintained by respective organizations and there is little or no access to source codes of them.

Besides aforementioned names, which are big players in cloud computing, there are open source cloud computing platforms which are under rapid development, mainly because they are open source. For the record, open source software is the one of which, the source code and blueprints are open for public to access and modify. Openstack, OpenNebula [11], Puppet, Eucalyptus are the paragon of open source cloud computing.

Openstack is array of software tools which are intended for managing cloud computation platforms for different kinds of clouds such as public or private cloud. And, it's free . It is primarily deployed as Infrastructure as a Service i.e. IaaS. The technology consists of a progression of interconnected projects that control pools of computing, storage, and network related resources throughout a data center—which one can manage through a web-based interface i.e. Horizon dashboard, command-line tools(CLI), or a RESTful API. OpenStack [1] released it under the Apache License v2.0.

Openstack project was a joint initiative taken by NASA and Rackspace Hosting in 2010.Today, it is managed by Openstack foundation, which is non-profit corporate entity established in September 2012.More than 200+ Companies like IBM, Intel, Cisco, Go Daddy, HP, Dell, Avaya, AMD, VMware, Canonical, Oracle, SUSE Linux has joined this project. Openstack community gathers every six months at a predefined location, somewhere around globe, to make a new release. During the time between two release cycles, there is series of frequent milestones as well. In biannual summits, developers determine the future direct ions as well. Most recently Atlanta summit attracted 4500+ attendees, which is a 50% increase compared to six-month earlier summit.

Openstack community's first official release, namely Austin, appeared in October 2010. Then Bexar, Cactus, Diab lo, Essex, Folsom, Grizzly, Havana, Icehouse and Juno were released in succession, approximately after every six months. Currently, Kilo is the one which is due.

Out of many sub-projects of Openstack, Heat and Horizon are the two on which the template generator is focused. As mentioned earlier, Horizon is Web -based interface of Openstack built upon Django Framework, while HEAT is main project in Openstack Orchestration program. [1]

Orchestration describes the automated arrangement, coordination, and management of complex computer systems, middleware, and services. [2]

The remaining paper is organized as follows. First, we will see the overview of Heat & Horizon components of Openstack. Then, we will see Current Scenario of Template writing. Next , we will p resent Template Generator with its implementation details and architecture. Finally, we conclude with benefits and future scope of the project.

II. ARCHITECTURAL OVERVIEW OF OPENSTACK

Openstack is open source solution to the massively scalable, reliable and easily deployable clouds, where resources are ideally infinite. All components are designed to work together in order to provide infrastructure as a service (IaaS). It consists of many components, which provide different services through Application programming interface. Any service can make use other service using API's. Hence in Openstack integration of different service is achieved through API's.

Openstack consist of following services,

A. Horizon (Dashboard service)

B . Nova (Compute service)

C . Glance (Image store)

D. Keystone (Authentication service)

E. Swift (Object Store)

F . Heat(Orchestration)

As mentioned earlier, we are interested in only two of these components. Let's take a look at them in brief.

A. Horizon

Horizon is a Django application, which provides graphical user interface to the user in order to use the services of Openstack.

Its architecture is akin to any other web application,

1) It is deployed on apache server using mod-wsgi.

2) Database, which relies on the other services for the data.

It is consisting of separate reusable python modules which usually communicate with the other services API's. [3] [4]

B. Heat

Heat is the main project in the OpenStack Orchestration program. It implements an orchestration engine to launch multiple composite cloud applications based on templates in the form o f text files that can be treated like code. A native Heat template format is evolving, but Heat also endeavors to provide compatibility with the AWS CloudFormation template format, so that many existing CloudFormation templates can be launched on OpenStack. Heat provides both an OpenStack-native ReST API and a CloudFormation -compatible Query API.

III. WORKING OF HEAT [5]

Heat template describes the desired cloud infrastructure in terms o f human readable text file.

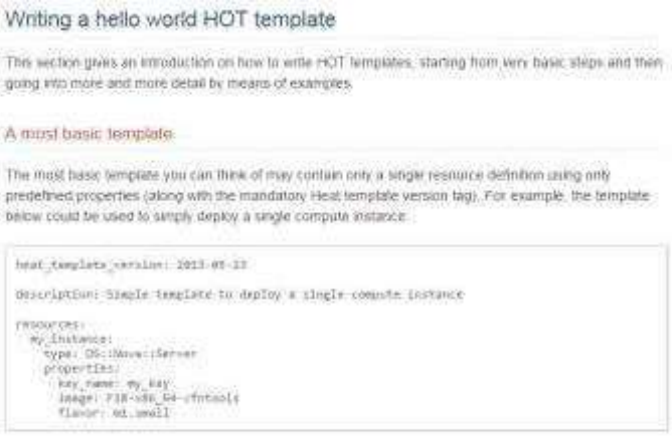
- This infrastructure comprises of number of resources which include but not limited to servers, floating IPs, volumes, security groups, users, etc.
- In a template, you can also specify relationship between resources. E.g. Load Balancer L1 is connected to server S1 and Server S2, etc. This enables heat to call the Openstack APIs to create your entire infrastructure in correct order to launch your application.
- Heat also allows you to update your infrastructure.

  Simply uploading new template for existing stack will change your infrastructure, and heat will take care of the things like adding or removing the resources as per the need.

IV. DRAWBACKS ASSOCIATED WITH TEMPLATES

Writing a template, passing it to heat, in order to launch the stack including desired infrastructure, looks perfect. But, it also has some drawbacks. Let's have a look at them one by one:

- Writing templates is a tedious task; we need to follow some format, such as HOT (Heat Orchestration Temp late) format. [7]
- Writing a simple template to launch a stack with single instance also seems like a lot of work.
- There is no compiler or debugger in horizon which will point out errors, and allow us to correct them. So, if you miss out even a single character, horizon will flag an error. Then, you will need to read your temp late line by line in order to find and rectify the errors.
- Here's a screenshot of temp late, which is used to launch a single instance.



Writing a hello world HOT template

This section gives an introduction on how to write HOT templates, starting from very basic steps and then going into more and more detail by means of examples.

A most basic template

The most basic template you can think of may contain only a single resource definition using only predefined properties (along with the mandatory Heat template version tag). For example, the template below could be used to simply deploy a single compute instance.

```
heat_template_version: 2013-05-23

description: Simple template to deploy a single compute instance

resources:
  my_instance:
    type: OS::Nova::Server
    properties:
      key_name: my_key
      image: F18-x86_64-cfntools
      flavor: m1.small
```

Figure 1 Sample Heat Template (Screenshot) [6, 8]

V . CURRENT ARCHITECTURE OF HEAT

The current architecture to create a stack in Openstack asks to provide a template through either of the following ways: an URL or file input through Horizon or the command line interface. The template provided is then

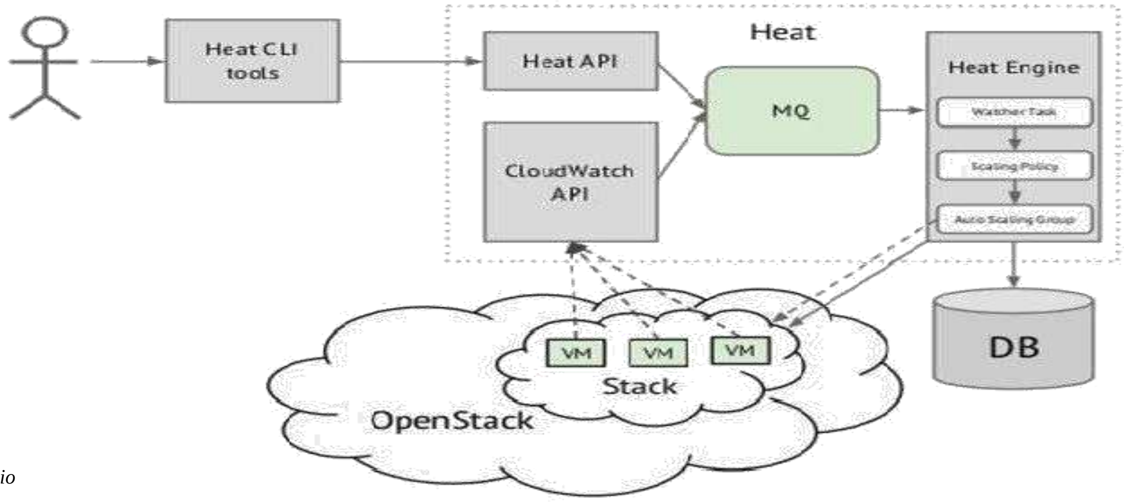


Figure 2 Current Scenario

validated by the Heat Engine after which stack is launched. This template is handwritten. The client can only view the stack topology after deploying the stack.

V I. MODIFIED ARCHITECT URE OF HEAT

We are proposing novel solution to this problem. There is no module or any integration in the dashboard (Horizon), which provides the graphical user interface (GUI) to the user for designing stack. You have to write long templates and provide it to the Heat engine, which in order performs the operations to create the stack. We are easing the stack creation by providing abstraction at interface level.

In proposed architectural change, we are adding a new component '*Stack Designer*' in the Openstack dashboard (Horizon), which will provide a graphical user interface to the user, using which user can design the stack topology. The design is then converted to the template by our module, which in turn is provided to the heat engine via API, which performs tasks to create that stack in cloud infrastructure. This change in architecture will provide a facility to check the template before launching the stack, thereby providing the ability to edit the template before launching the stack. The template used to launch the stack can also be stored, which can be reused later, on same system or somewhere else.

V II . WORKING OF MODIFIED ARCHITECT URE

After integrating the change in dashboard service i.e. Horizon, there will be a panel in panel group for stack designer in the main UI of horizon. After clicking on that panel, canvas for stack design is rendered in page, which has drag and drop facilities, using which we can design the stack graphically and no need to write long and complex heat templates to generate or orchestrate the stacks.

Each resource in the canvas has associated with it a

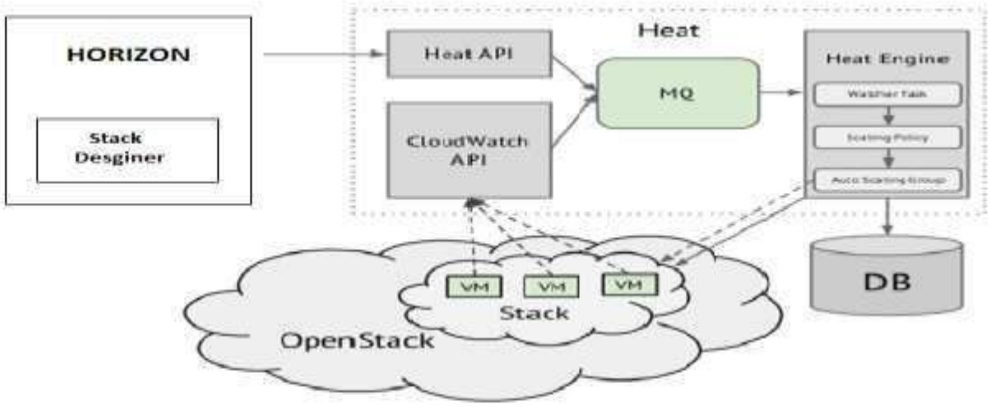


Figure 3 Modified Architecture of Heat

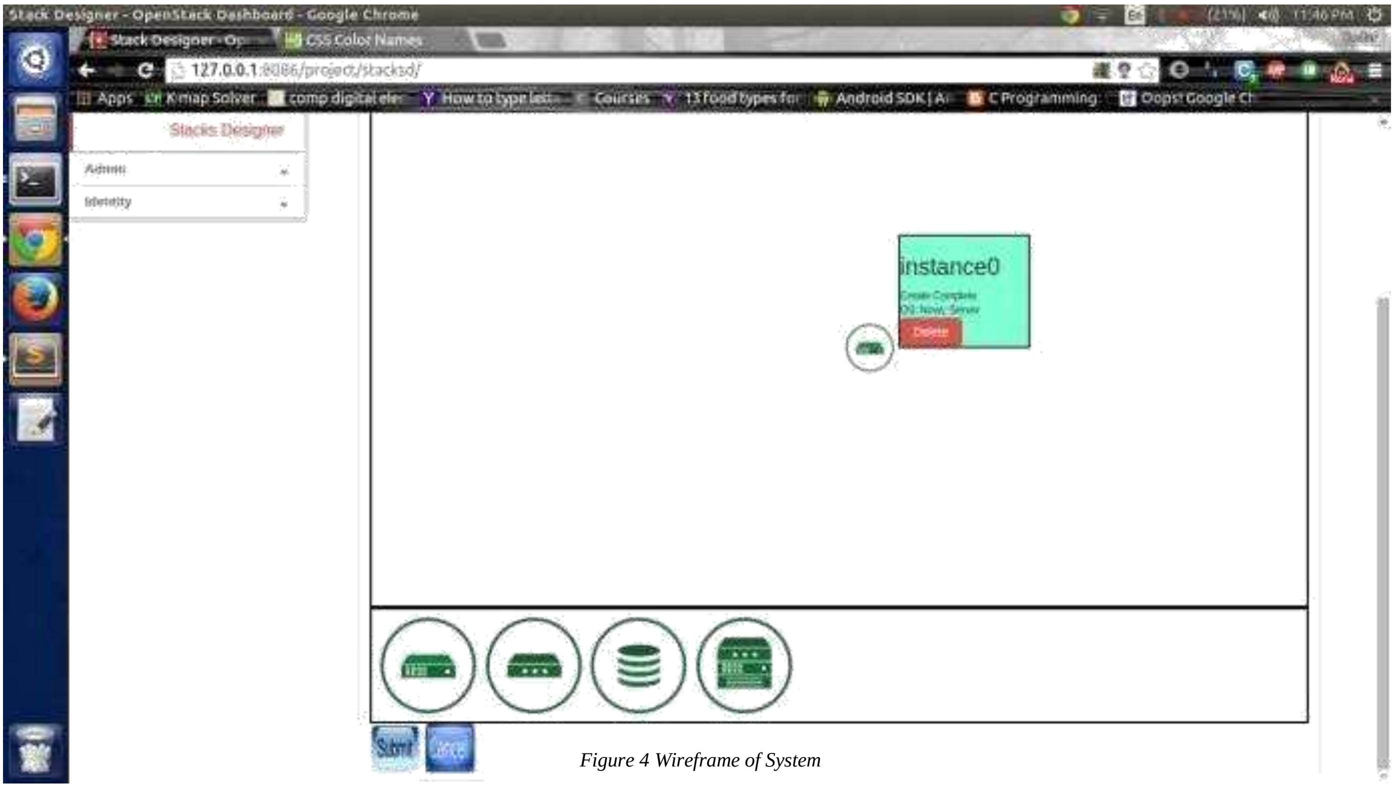

Stack Designer - OpenStack Dashboard - Google Chrome
127.0.0.1:8086/project/stacksd/
Stacks Designer
Admin
Identity
instance0
Delete
Submit

Figure 4 Wireframe of System

form which provides the user to select the properties to be Associated with that resource in the template. (Example, for the instance resource, the user can provide the instance name, image, key-pair, etc.).

On submission of this form data for a resource, the data is stored in the back-end. Prior to creation of the template, this data can be edited number of times.

After forms for all resources have been submitted, a template can be generated, which converts the data associated with the resources forming the stack into a Heat Orchestration Temp late.

This template is then provided to the Openstack Heat API which first does template validation on the generated template, after which stack is launched.

VIII. BENEFITS COMPARED TO CURRENT SYSTEM

Current Scenario	**Heat Template Generator**
Topology will be created only after successful launching of` stack with resources.	User friendly UI, with visible topology in the work space, which will help the user to easily create new resources and add it into the topology.
For deletion or updating any resource information you need to change your template file and add or delete the required information manually	As the additions of the resources are so much easier, same applies to deletion of the resources from the topology. We are providing the pop up for each resource to show information of that resource and to delete the resource.
User needs to go in the Images, instances, keystone components to get the availability and use of those resources.	Resource properties can be easily filled with the property window provided on double click of resource image, also the availability of images, flavors, key pair for Instance is shown in drop down.
User needs to manually remember the resources that he wants to link with the load balancer or routers.	Easy linking of instances with the load balancer and also the visual links are provided in the topology.
Including a number of resources in the template makes it complex and increases the possibility of errors.	Number of resources can be included in the topology is sufficient to form a virtual network.
You need to follow the strict format given by the Heat and user needs to learn how to write template and remember each and every part.	No extra knowledge will be required to use our web application.
Need to write whole template without error by user only. No automation for this process.	On a single click template will get generated and stored in text file with extension as .template or .yaml.
Heat template is portable, you can launch stack on any machine with same template.	As our application is part of Devstack which is portable with any system containing Linux, as a result, it makes our application portable, as well.

Template writing is a complex and time consuming	It will save time of user who wants to launch number of instances with other many resources.
As all the fields are user written, there will be high probability of errors in template. And no compiler to find those errors.	User is not a part of template generation process at all, hence the chances of human error are totally void.
Update facility is not provided by Heat, you need to write the updated template and again re launch the stack.	The facility to update is not provided by heat for template, but in our application we provide a partial update for user who wants to make changes in the topology, i.e. add or remove number of resources.
There is no further work going on for simplification of template generation process.	The web application we have developed helps the user in many aspects while creating a template, but still we are looking forward to improving our web applications capability to provide user an error free support with less effort.

IX. LIMITATIONS

The web application we have developed helps the user in many aspects while creating a template, but still we are looking forward to improving our web applications capability to provide user an error free support with less effort.

Nonetheless, this project has a few limitations as follows:

1. At this moment, we are supporting the less number of resources (example Instance, load balancer) to generate a template. In future our web application should support maximum number of resources supported by Heat.
2. Update in generated template is still a challenge for us, in our next releases.
3. We will try to generate topology from existing template so user can easily make changes in it
4. It will be part of the next OpenStack release, so that many more number of user will get chance to use this web application.
5. User can download the OS image through our application just after providing the link and it will get merge in images in nova.
6. Unable to extract template from existing stack. [10]

CONCLUSION

Openstack is one of the open source platform to build private cloud. It is relative young as compared to other open source platform. This paper provides details of our Project Heat Temp late Generator and how it has an edge over traditional template writing. We also briefly discussed how it will benefit the group of users. While it certainly has few limitations, it nonetheless helps us to improve the user experience, as well.

REFERENCES

[1] “OpenStack” [Online] Available: *www.openstack.org*

[2] “Networking concepts” [Online] Available: *http://docs.openstack.org/admin-guide-cloud/content/section_networking-options.html*

[3] “Horizon: The OpenStack Dashboard Project” - http://docs.openstack.org/developer/horizon/

[4] “OpenStack Dashboard ("Horizon")”- *https://wiki.openstack.org/wiki/Horizon*

[5] “Heat ”- *https://wiki.openstack.org/wiki/Heat*

[6] “Heat Orchestration Template (HOT) Guide”- *http://docs.openstack.org/developer/heat/template_guide/hot_guide.html*

[7] *“*Template Guide*”* – *http://docs.openstack.org/developer/heat/template_guide/*
“Heat Orchestration Template (HOT) Specification” - [Online] Available: *http://docs.openstack.org/developer/heat/template_guide/hot_spec.html*

[9] Tom Fifield, Diane Fleming, Anne Gentle, Lorin Hochstein, Jonathan Proulx, Everett Toews, Joe Topjian, *OpenStack Operations Guide,* Openstack Foundation,2014.

[10] Yoji Yamato, Masahito Muroi, Kentaro Tanaka and Mitsutomo Uchimura, *“Development of template management technology for easy deployment of virtual resources on OpenStack”,* Journal of Cloud Computing: Advances, Systems and Applications 2014, 3:7

[11] Milojicic D, Llorente IM, Montero RS (2011), “ *OpenNebula: a cloud management tool*”, IEEE Internet Comput 15(2):11-14

[12] Microsoft Azure [online] Available: http://azure.microsoft.com/en-in/

[13] IBM Cloud [online] Available : http://www.ibm.com/cloud-computing/in/en/

[14] Amazon elastic compute cloud (EC2), [Online] Available:http://aws.amazon.com/ec2/.