Lab OSTK - OpenStack Heat

Introduction and Prerequisites

This laboratory is to:

- Explore various capabilities of OpenStack's orchestration engine Heat
- You will recreate several virtual resources similar to what was done in CC-IAAS lab but using various Heat resource supported by the engine
- You will orchestrate a virtual application by utilizing various resources supported by the HOT specification and the orchestration engine.
- You will study the automation of scaling of your virtual machine based on specific rules governing the load on the VM

The following resources and tools are required for this laboratory session:

- ∉ Any modern web browser
- ∉ OpenStack official documentation (pointers see Appendix)
- ∉ Modern text editor such as Sublime Text that are capable of editing YAML files
  - http://docs.ansible.com/ansible/latest/YAMLSyntax.html
- ∉ Any modern SSH client application
- ∉ OpenStack Horizon dashboard: https://ned.cloudlab.zhaw.ch
- ∉ OpenStack account details. Please contact the lab assistant in case you already have not received your access credentials.
- ∉ The code for this lab is provided in a single zip file on Moodle.

Task 1 – Setting-up and Launching an Instance

A VM with the OpenStack command line tool preinstalled is provided as a working environment for this lab. You will further create simple OpenStack HEAT template files and execute these templates using the command line tool.

Subtask – Prepare Your Lab Environment

- Launch a VM instance
  - Image: CCP1-EN-OSTK-Lab_ubuntu-16.04-openstackclient
  - Use m1.small flavor while launching the VM
  - Assign a floating IP
  - SSH into this VM
- Upload your OpenStack (NED) credentials file into your Instance
  - Download your openstack credentials file
    - Log into your account in the horizon dashboard
    - Navigate to: Project → API Access → Download OpenStack RC
  - Use, for instance, sftp or scp commands to transfer the file to your Instance
- Upload the provided lab files into the Instance
  - Download the file from Moodle
  - Upload the file to your Instance (e.g. via sftp, scp, etc)
  - Decompress the file
- SSh into your Instance, configure the OpenStack CLI environment and test the CLI tool
  - Source the credentials file: $ source <file-name>-openrc.sh
    - Replace *<file-name>-openrc.sh* with the actual filename you saved
    - This has to be done for every ssh session.
  - Run this command
    - $ openstack server list
    - $ openstack help

Subtask – Creating an Instance with the CLI

The command template below creates an Instance using the “openstack” command line tool. Adapt the example and launch an Instance on NED.

```
$ openstack server create --flavor FLAVOR_ID --image IMAGE_ID --key-name
KEY_NAME --security-group SEC_GROUP_NAME --network NETWORK_NAME
INSTANCE_NAME
```

Subtask – Elasticity, Scale an Instance Vertically

Resize the instance to a different flavor using the command line tool (CLI).

- Does the status get active? Check the help description for the resize command.

Subtask – Exploit Pay-as-you-Go

Check which options exist to save money if your instance is a.) temporarily not needed and b.) not needed anylonger.

Subtask – Reboot an Instance

Reboot your Instance using the CLI with the possible modes and discuss the effect of both.

Subtask – Recapitulate the OpenStack Architecture

Based on your experience so far, recapitulate, analyze and discuss the OpenStack architecture. Some hints (list not exhaustive nor complete) below:

- Which components are involved?
- What is the sequence of interactions?
- What is required and what is provided by which component?
- How are resources identified in OpenStack
- What are the states of resources?
- What kind of operations can be applied to which resources?

Subtask – Infrastructre as Code - Creating a Simple VM via Heat

In this task you will create a basic stack containing just a single VM and discover the associated floating IP allocated at runtime.

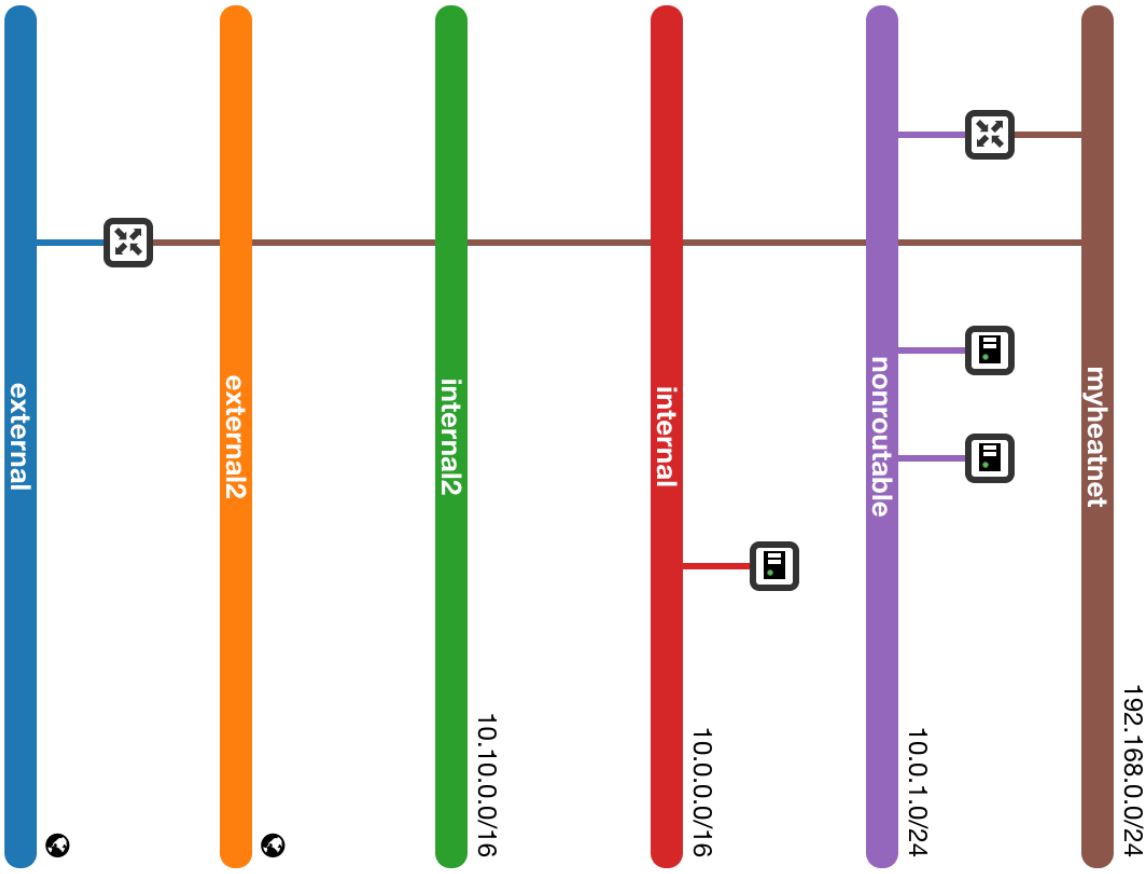
- Make sure your openstack account has a public key registered.
- Analyse the `st-12.yaml` file within the lab templates provided.
- Fill in values based on your OpenStack environment. Use the following image CCP1-EN-OSTK-Lab_ubuntu-16.04-docker-multinic
- Navigate to the OpenStack UI -> Orchestration -> Stacks and check find out how you can create a stack based on the file that you provided.
    - Hint: You can also download the file and edit it on your local PC
    - Hint: Mind that you can work with OSTK resource names and OSTK resource IDs. You can find details about both via the OSTK UI.
    - Hint: Make sure your YAML format is correct. There are YAML validators on the WWW.
- After successful creation via the UI, let's now do the same via the CLI, i.e. the **`openstack stack`** operation, which creates your stack, retrieve details and finally delete your stack
    - Use **`openstack stack --help`** to find more
    - You will be using these commands:
        - `openstack stack create`
        - `openstack stack show`
        - `openstack stack delete`
    - Do not forget to provide all necessary arguments that these above commands require.
- Discuss what you observe with your partner and/or lab assistants.

Subtask - Advanced Virtual Environment Creation

In this task we will create a more realistic environment.

- Start from the base template file '`st-13.yaml`', and analyze the template.
- Discuss how it creates a new key pair, a new network, subnet, router and then a virtual machine.
- Complete the template with data from your OpenStack environment.
- What does this template output?

Use this template (`st-13.yaml`) as a starting point and **extend** it (or create a new HEAT template) to create the same setup as **exemplarily** shown in the figure below using the **correct references** (i.e. names of resources) as found in your actual OpenStack environment.



- You must have one VM called (for instance called frontend) which is publicly reachable (Internet access)
- Two VMs as backend (for instance called backend1 and backend2) nodes that are only accessible through the public VM (e.g. via frontend)
- You can name your virtual resources (i.e. networks, servers) differently as long as the structure resembles the diagram above)
- Similarly you can choose a reasonable subnet range for your networks.
    - All subnets must have an IPv4 private address range
- Ensure that `networks` are used consistently when making declarations about public networking

    - *HINT*: If you need to enforce a particular order in creation of resources, use **depends_on** feature.
- Test whether you can SSH into your frontend VM. Can you ping the internal VMs? If you cannot SSH into it, suggest a solution via Heat.

Task 2 - Studying Auto-Scaling

Heat supports **AutoScalingGroup** which allows trigger based rules to scale out or scale in (Elasticity). This task explores this functionality.

Subtask – Understand a Template File with Built-in Autoscaling

In this task you will study the sample heat template with autoscaling enabled.

- Change into `st-21` directory and open the stack with scaling related templates.
- Analyze and study the two template files, discuss the various elements in detail with your partner and/or lab assistant.
    - Describe how the auto scaling logic is encoded in the template?

Subtask – Understand an Advanced Template for a Wordpress Installation with Built-in Autoscaling

In this task you will study autoscaling rules (Elasticity) by means of a more realistic use case, that is a heat template with autoscaling enabled for a Wordpress deployment.

Task 3 - Cleanup - Stop the Bills!

IMPORTANT: At the end of the lab session:

- **Delete** al - unused - VMs, volumes, security group rules that were created by your team
- **Release** all floating IPs and other resources back to the central pool for others to use

Related Documentation

- OpenStack Command Line
    - https://docs.openstack.org/operations-guide/ops-lay-of-the-land.html#command-line-tools
- Tools: sftp / scp / tar / zip / unzip
    - Linux man pages