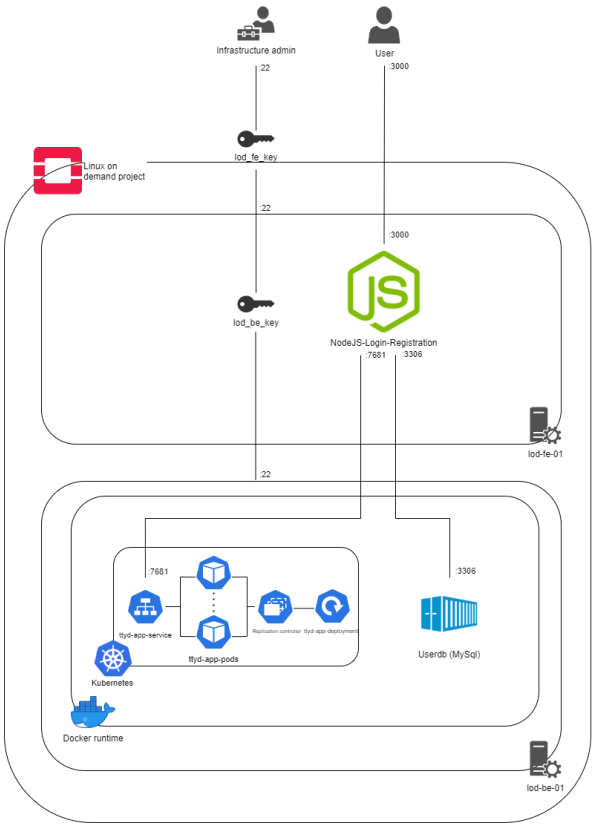
1 Introduction

In this project, we have designed and configure an infrastructure that hosts a Linux virtual sandbox environment. In order to configure the necessary infrastructure assets, we have used OpenStack as our IaaS platform. In addition to this, we have adopted Docker and Kubernetes PaaS solutions to deploy the required containerized application and most importantly to automized all the platform we have used the terraform from hashicorp.

2 Architecture

The architecture of the platform is really straightforward. As you can see in the figure down below. We have explained it step by step.

2.1 Infrastructure Design



After receiving detailed feedback from the professors and according to the available resources, we decided to modify the infrastructure that we have presented before. We eliminated 2 instances from the design architecture in order to avoid any performance issues. We also removed ‘edu-db-01’ and ‘edu-bh-01’, from the infrastructure that was originally providing the database and Bastion Host functionalities. These functionalities are now integrated into the remaining instances: ‘lod-fe-01’, which now acts also as bastion host of the infrastructure, meanwhile ”lod-be-01”hosts a MySQL docker container that acts as the database of the infrastructure.

2.2 Networking

The base OpenStack service that creates the structure of the network subnets that acts like a container for the instances is **Neutron**. It is possible to create a dedicated network for the instances of the project. In this case the network is labeled ”lod-private-network” and any server created will be placed in this network. Different

subnets can be allocated in the network. For this project a single private subnet labeled 'lod-private-network-subnet-1' is allocated. No changes have been performed on the network configuration of the infrastructure.

Subnet name	Network subnet	Public/Private	Gateway IP
lod-private-network-subnet-1	10.0.2.0/24	Private	10.0.2.1

Private subnet is restricted only to be accessible through the dedicated router interface between private project network and public network. The router named "lod-router-01" is connected to the private subnet through an interface. In this way it is possible to reach public internet from the subnet and viceversa when allocating a floating IP.

Router name	Availability zone	Interface	Gateway IP
lod-router-01	Nova	Public $\Longleftrightarrow lod-private-network-subnet-1$	10.0.2.1

3 Asset inventory

This section will describe the servers from the hardware and software point of view. The table below summarises the hardware resources of the servers in place for the project.

Host	Flavor	Number of vCores	RAM	Storage	OS Image	Floating IP
lod-fe-01	m1.small	1	2GB	20GB	Ubuntu 18.04	Yes
lod-be-01	m1.medium	2	4GB	40GB	Ubuntu 18.04	No

3.1 Front-end (lod-fe-01)

The front-end instance is in charge of hosting the WEB Interface for the final user. Originally we wanted to configure the webserver with **Nginx**. However, in order to avoid all the problems related to the complex configuration of Nginx, we adopted a complete **nodejs** web application (nodejs-login-registration repository) that could satisfy our requirements. This nodejs application exposes a login/registration prompt through port 3000. All the users' accounts are saved in the DB located in lod-be-01 as a docker MySQL container reachable from port 3306. Once the user has been authorized, he can access the Linux sandbox which is a Kubernetes service hosted also on lod-be-01 and reachable from port 7681. In addition to the web application, this instance has also the purpose of providing access to a private network from the public network. For this reason, a floating IP is associated with this instance, and system administrators can establish an ssh connection with lod-be-01 through lod-fe-01.

3.2 Back-end (edu-be-01)

The back-end instance is the core of the project as the core web application is executed here. In order to make the Linux sandbox environment available, a Kubernetes deployment and service are configured. Each pod of the deployment is part of a replica set and a service exposes each pod of the deployment through port 7681. All the pods of the deployment are based on the official docker image of the ttyd project (TTYD repository website. Ttyd is a simple web terminal app that lets authorized users access pods. In this way, the user can play around with a Linux sandbox environment directly from the browser. In addition to the core nodejs application ttyd a docker container based on the mysql image is deployed in this instance. This container stores all the users' account information and it has a volume attached to it in order to not lose stored data whenever the container is restarted or removed. The container is exposed through port 3306 and it is exclusevely reached by the front-end.

4 DevOps

In order to automate the development process as we have discussed before. We have used Terraform as our infrastructure as a code software tool. In this way all the infrastructure components are configured and managed in a structured way. Additionally, thanks to this IaaS tool, it is possible to see if any edit can affect the deployed components, before applying those changes. In addition to terraform, all the instances are configured by using bash scripting and cloud-init functionalities in order to install all the necessary packages and create the required assets. All the assets required for the deployment and the configuration of this project will be released in the following GitHub repository: https://github.com/SuperboGiuseppe/education-on-demand.