Cloud Computing 2022/23 spring

Introduction to Infrastructure as a Service and OpenStack

In this practice session you learn how to access cloud services that we will be using in the rest of the course. We will be using a private university cloud - which is a Cloud infrastructure running on the hardware of the University of Tartu and is managed by the High Performance Computing Center. In this lab we are working on the OpenStack cloud platform, located at: https://stack.cloud.hpc.ut.ee/

- To access the local university cloud resources your computer has to be inside the Institute network. So you should either use lab computers, Eduroam Wifi (inside the institute building) or set up a VPN connection to university network.
  - VPN (English on the right side column) - https://wiki.ut.ee/pages/viewpage.action?pageId=17105590
  - Eduroam Wifi (English on the right side column) - https://wiki.ut.ee/display/AA/Eduroam

NB! Students have previously reported that using Eduroam in dormitories will not give access to the University cloud. You will have to use VPN in such cases.

Practical session communication!

We run the labs physically in the Delta building. We use Zulip for the running communication between students and lab supervisors.

- You must schedule attending the practice sessions at the correct time (based on your chosen lab group). If you complete the lab tasks outside the scheduled practice sessions, we can not guarantee that you will receive timely support from the lab supervisors.

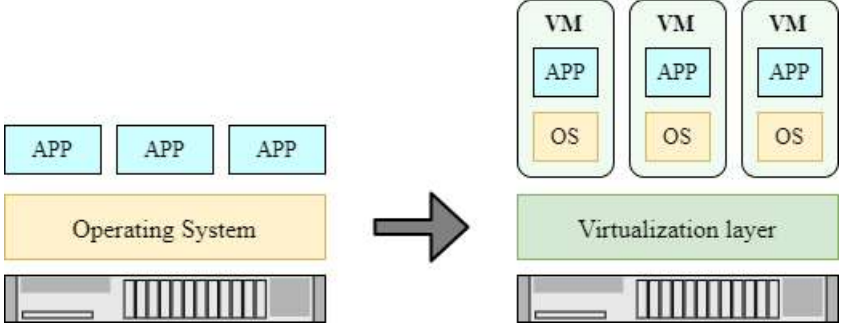
- We have set up a Zulip topic for lab and course-related discussions.
  - Login to Course Zulip stream using your university account from here
  - Do not make a new account!
  - If you do have no access to the Zulip stream, please contact lab assistants in a practice session or through email.
  - **Use the `practice-session-1` Zulip topic for this lab-related questions and discussion.**
- When asking questions or support from the lab assistant, please make sure also to provide all needed information, including screenshots, configurations, and even code if needed.
  - If code needs to be shown to the lab supervisor, send it (or a link to it) through Zulip Direct Messages.

In case of issues check:

1. Existing messages in the `practice-session-1` Zulip topic.
2. Possible solutions to common issues section at the end of the guide.
3. Ask in the `practice-session-1` Zulip channel.

Introduction

Infrastructure as a Service (IaaS) is a model of Cloud computing, in which virtualized computing resources are provided to users over the internet. In comparison to using physical servers, computing resources can be provisioned **on-demand** and in **real-time** and applications running on the same hardware can be separated into different secure environments, each containing their own OS, software libraries and kernels.



Working with the IaaS model of Cloud usually consists of the following steps:

1. Register an account to access the cloud services
2. Select appropriate virtual machine image to run (Ubuntu, Debian, Windows, etc.)
3. Start a new instance of the selected virtual machine image. Log into the instance as a root user over the internet and configure it to meet your requirements. I.e. install needed software, upload your own application, perform any required configuration actions as you would do in any real computer.
4. As you will lose all your work when the instance is terminated -- you have three options on how to persist the changes you made:
    1. Save all your configuration steps to a **script** that will launch and configure the instance automatically for you.
    2. Bundle a new **image** from your running instance and next time launch your custom image.
    3. Save the running instance as a **snapshot**, and next time launch new instances from there.

The first option is more flexible as you can easily change the script than bundle a new image if something changes. The second and third options are simpler to use once you have a stable configuration or when launching a large number of instances.

In this lab, we are working on the OpenStack cloud platform.

Exercise 1.1. Accessing the cloud services

In this exercise, you will log into the institute OpenStack cloud and create a secure access key. You will verify that you have access to the university OpenStack cloud resources and familiarize yourself with the available cloud functionality.

- Log into https://stack.cloud.hpc.ut.ee/ using your university `username` and `password` and `ut.ee` as domain.
    - Contact lab supervisor if you have issues while logging in. Make sure to include your university username in the email, as it is required for configuring access.
    - You have to be in the university network to access this website. Use VPN or log in to eduroam, if you're using your own laptop
- Familiarize yourself with the available OpenStack cloud functionality.
- Create an ssh Key Pair for accessing Virtual Machines over the network. Make sure the name of the Key Pair includes your last name!
    - You will find this functionality under **Compute -> Key Pairs**
    - **NB!** This will download the private key as a text file into your computer with a `*.pem` extension. Copy the file into a location from where you can easily find it later.

Exercise 1.2. - Requesting computing resources from the cloud

In this exercise, you will start a Cloud instance (or virtual machine) while specifying its configuration and computing resources available for it.

- Use the OpenStack web interface
- Under the "Compute" tab go to "Instances" and start a new instance by clicking the "Launch Instance" button (If not specified leave the default values)
- We'll start a new instance of **Ubuntu**-based virtual machine image
    - As *Instance Name*, specify: `Lab_number-Student_name`
        - e.g. "Lab_2-John_Smith". *Use this format in all future labs! It helps us help you in the labs :)*
    - Choose **ubuntu20.04** under **Source** tab & set the volume Size to 10GB (if it isn't)
    - Also enable **Delete Volume on Instance Delete** under **Source** tab
        - This will mark the underlying volume (virtual disk) to be automatically deleted when your instance is deleted.
    - Choose the capacity of the instance
        - Under **Flavor** tab, choose **m3.nano** as the type of the instance
        - This will request 1 Virtual CPU core and 1GM RAM for the instance
    - Choose network for the instance
        - Under **Networks** tab, choose **provider_64_net**
        - This will assign the instance network interface into an internal UT network
    - Specify what Key Pair to use under the **Key Pair** tab!
        - Select the Key Pair that you created in the previous exercises. If you lose the downloaded file, you will have to create and download a new one!

Exercise 1.3. Accessing your Cloud instance over the internet

We will use Secure Shell (ssh) protocol to log into the started instance over the internet. Instances in the cloud can have multiple IP addresses. **Public IP** for accessing the instance from outside the cloud and **Private IP** for accessing the instance from inside the cloud (from other instances). However, our instances will only have a single IP in the current configuration.

- Log into the instance through **ssh** using SSH Key-based authentication

***Note**: On an up-to-date Windows 10, the above Linux approach should also work from Windows commandline/powershell out of the box (recommended)!*

Exercise 1.4. Configuring Cloud Instance & installing software

We will now set up a simplistic Python web application on the instance, setting up the necessary software and dependencies. The application is built with the Flask micro web framework. The app represents a messaging board that stores posted messages into a json-structured text-file.

1. Install Python Flask webserver on the instance
    - First, fetch the source code of the application with git, storing it into the folder "lab1app":
        - `git clone https://bitbucket.org/jaks6/cloud-computing-2022-lab-1.git lab1app`
        - Enter the application src directory: `cd lab1app`
    - This app uses the *pip* package Flask as a dependency. We will create a Python *venv* (virtual environment) for this application and install the packages for the app within the *venv*.
        - Update the package lists about available software and install the *python3-venv* package:
            - `sudo apt update && sudo apt install python3-venv`
        - Create a *venv* named "env", and activate it:
            - `python3 -m venv env`
            - `source env/bin/activate`
            - After activating, you should see the "(env)" appear infront of your username in the terminal session.
        - Use pip to install the necessary dependencies of this application (defined in requirements.txt):
            - `pip install -r requirements.txt`
    - Start the Flask web application
        - `flask run --host=0.0.0.0`
            - This will start the Flask server on port 5000.
            - Note: you need to be in the *lab1app* folder!
            - we set the "host=0.0.0.0" parameter to ensures the webserver listens to external traffic too, not only local traffic.
        - (Or, to set it to run in the background, if you wish :)
            - `nohup flask run --host=0.0.0.0 &`
            - *nohup* ensures the program stays running after we log out
            - Note: if you want to kill python program running in the background, you can use `fuser -n tcp -k 5000` - this kills the process listening on port 5000
2. Check that the installation of the Flask web server is successful
    - Try accessing the address of your virtual machine on port 5000 from a browser (from within the university network).
        - E.g. http://172.17.67.124:5000
        - This, in theory, should display the `lab1app/templates/home.html` page that's being provided by your web server.
        - **However**, it does not work at the moment, as by default communication with the cloud instances from outside other than ssh (port 22) is restricted by default.
    - Let's use an alternative approach to verify our web server is running
        - Log into the instance through ssh
        - Use the `wget` command to download the webpage from your server.
        - On the instance command line run `wget localhost:5000`.
            - This should download index.html file into the currently active directory, which among other HTML codes should contain a string "Message board".
        - Use `less index.html` command to check the downloaded file content from the command line
        - Alternatively you can use command line web-browser `sudo apt install lynx` and `lynx localhost:5000`
3. Modify (or replace) the current home.html file at `lab1app/templates/home.html` to change the web page content.
    - How exactly you change its content is up to you, but it should at least contain your **Full Name**, so it is possible to visually see that you have modified it.
    - To deploy the changes, you need stop and re-start the Flask server! See Ex 1.4 for the relevant commands.
    - Command line file editor `nano` can be used to modify file contents.
    - Feel free to replace the whole HTML file with a new one.

Exercise 1.5. Creating Security group to enable access through port 5000

To allow access to the hosted web server on your instance you need to create a new security group and define an access rule for the HTTP port 5000 used by Flask.

- Under the `Network` tab go to `Security Groups` and create a new security group by clicking the `Create Security Group` button
- Choose a `name for this security group`, which should include your first and last name
- Add the HTTP TCP port 5000 to your security group.
    - Remote IP Prefix should stay `0.0.0.0/0` (this means all devices from any IP address can access this port)
    - Do not assign a **Security Group** in the **Remote** field, it would limit the specified port rule to apply only for other cloud instances inside the selected security group.
- Now add this security group to your instance
    - Use `Compute -> Instances -> drop down menu next to your instance -> Edit security groups` and add your security group from under **All Security Groups** into **Instance Security Groups**
- Access your instance through a web browser

Exercise 1.6. Creating a new Cloud Instance snapshot

Lets save all the changes you have made to the instance by creating a snapshot. This allows you to start multiple already-configured Flask web server instances at once, with your web application already included.

- Under the `Compute` tab, go to `Instances` and choose `Create snapshot` button next to your instance.
- Choose a name for this snapshot, it **must** include your **last name**.
- After you confirm that the snapshot is ready, terminate your instance
- Start a `new instance` as you did previously, but now use your snapshot as the source instead of Image *ubuntu* and make sure that both your new security group together with the default security group are chosen.
    - Make sure you can access the Messageboard website from the web browser using your new instance's IP address.
    - Make a screenshot of the browser showing your deployed web page on the instance
        - NB! Your name must be clearly visible on the screenshot you took in the previous task!
        - From the screenshot, the URL with the public IP should be visible
- In the future, you are able to start a copy (or multiple copies) of this webserver at any time by starting a new instance from this snapshot.

Bonus tasks

The normal exercises have been prepared in a way that should not take you more than two academic hours to solve them. However, you can earn extra practice session credit points by completing Bonus exercises. Not all practice sessions will have bonus tasks, and some bonus tasks may take significantly more than 2 hours to solve. But they have been designed for students who wish to learn more practical knowledge of Cloud Computing.

Bonus task 1: Accessing your instance through the web interface

[ ]

Bonus task 2: Attach a volume to an instance

[ ]

NB! Once you are done, you must delete your instance and the Volume you created! Also, be careful you do not delete the work of other students.

Deliverables:

- Your instance must have been terminated (deleted)!
- Snapshot (VM image) must exist with your name.
- Screenshot created in **exercise 1.6** and any bonus tasks you complete
    - Pack the screenshots into a single zip file and upload them through the following submission form.
- Also, submit an answer for the following questions:
    - What happens if you lose your ssh KeyPair file? What happens to existing instances which were started with the lost ssh key?
    - What are the advantages of utilizing cloud Volumes? Briefly describe at least two scenarios, where using volumes simplifies working with cloud instances.

Task: Lab 1

Current submission: ZIP

(23:50 08.02.2023)

If your homework consists of multiple files (for example HTML + CSS + JS) it should be archived before submitting.

File: Choose File | No file chosen

Comment:

What happens if you lose your ssh KeyPair file? What happens to existing instances which were started with the lost ssh key?
By losing my private SSH key I would no longer be able to access existing instances that were started using my public key if I didn't have any other means to access them.

What are the advantages of utilizing cloud Volumes? Briefly describe at least two scenarios where using volumes simplifies

Submit

Possible solutions to common issues

1. If you can not access the Cloud instance over SSH,
    - it may be because you are using ut public (use eduroam instead) or have not set up VPN connection.

2. If you can not access your instance over port 5000
    - it may be because you have not opened port 5000 in your security group
    - or added the created the security group to your instance
    - or not used correct IP filter for Remote IP Prefix:
        - Remote IP Prefix should stay 0.0.0.0/0 (this means all devices from any IP address can access this port)
        - Do not assign Remote Security Group, it would limit to the specified port from only the cloud instances inside the selected security group.

3. If you get a warning from ssh command line command about the file permissions of the SSH KeyPair file:
    - In Linux: use chmod command line command to change the file permissions to 600 (removing permissions for other users and group users)
    - In Windows: modify key file Security permissions, and remove permissions for the generic Users, leaving your own windows user and Administrator user permissions in place

4. **Can not access UT OpenStack with your university account**
    - Everyone who registered for the course **AFTER 2/6/2023**, please send Pelle Jakovits a Direct Message in Zulip topic `general discussion` with your university username. We will need it to request access to University OpenStack Cloud, which is required to be able to complete the first lab.

5. You might encounter an error stating that something is locked that is Ubuntu running some updates in the background so please give it few minutes to complete and try again later, if still no luck, ask help from the lab instructor. (Use with caution! https://www.tecmint.com/fix-unable-to-lock-the-administration-directory-var-lib-dpkg-lock/)
6. FIX ERROR: "sudo: unable to resolve host <your_machine_name_here>"
    - If you try entering any sudo commands i.e `sudo free` ; `sudo du` you should get and error "unable to resolve ..."
    - In order to fix it edit `/etc/hosts` file and add your hostname to the end of first-line like this `127.0.0.1 localhost <your_machine_hostname_here>`. You can use `nano` with `sudo` rights to do it.
7. FIX ERROR: **ssh: connect to host 172.17.6X.YYY port 22: No route to host**
    - Check you are using eduroam in Delta, or VPN from home
    - IF you have **Linux** on your computer AND have **Docker installed**:
        - Then there is a possibility of network address conflict between docker network and university network that VPN uses. It is likely that packets you try to send to some of the 172.17.* networks are sent to your local docker network instead.
        - One solution would be to reconfigure what networks your local Docker uses. The steps are:
            - Create a directory in the virtual machine in the path: `sudo mkdir /etc/docker`
            - Create a file in the docker directory: `sudo nano /etc/docker/daemon.json`
            - Copy the following entry into the file:
            - ```
              {
              "default-address-pools": [{"base":"172.80.0.0/16","size":24}]
              }
              ```
            - Restart docker service using: `sudo systemctl restart docker`

Õppematerjalide varalised autoriõigused kuuluvad Tartu Ülikoolile. Õppematerjalide kasutamine on lubatud autoriõiguse seaduses ettenähtud teose vaba kasutamise eesmärkidel ja tingimustel. Õppematerjalide kasutamisel on kasutaja kohustatud viitama õppematerjalide autorile.
Õppematerjalide kasutamine muudel eesmärkidel on lubatud ainult Tartu Ülikooli eelneval kirjalikul nõusolekul.

The courses of the Institute of Computer Science are supported by following programs:

European Union
European Social Fund

Investing
in your future
              ```