# Solution M1: Introduction to DevOps

This is one possible solution of the tasks included in the homework

All steps that follow assume that we decided to base our solution on Ubuntu Server 20.04

## Creating an Ubuntu Box

Define a simple VM in **VirtualBox** with the following parameters

* Name set to **ubuntu-server**
* 1 CPU
* 2 GB RAM
* 32 GB HDD (Dynamic)
* No Audio
* 1 NIC in NAT mode (with a port forwarding rule **SSH**/**TCP**/**2222**/**22**)

Attach the installation media and boot the VM

Follow the installation wizard steps and

* Create user **vagrant** with password **vagrant**
* Do not forget to mark the installation of the **OpenSSH** server

Once the installation is finished, enter the VM with the user you created during the installation

* Upgrade all installed packages (you may use either **apt** or **apt-get**)

**sudo apt-get update**

**sudo apt-get upgrade**

* Add new packages if you see fit. For example, install at least these (they are needed for building the **VirtualBox** modules):

**sudo apt-get install gcc make perl**

* Insert the **VirtualBox Add-ons** media
* Mount the **VirtualBox Add-ons** media and install them

**sudo mount /dev/sr0 /mnt**

**sudo /mnt/VBoxLinuxAdditions.run**

* Reboot the system if required

**sudo reboot**

* Eject the **VirtualBox Add-ons** media
* Add the **vagrant** user to the **vboxsf** group

**sudo usermod -aG vboxsf vagrant**

* Add the **vagrant** user to the **sudoers** list and allow it to **sudo** without entering password

**echo "vagrant ALL=(ALL) NOPASSWD:ALL" | sudo tee /etc/sudoers.d/vagrant**

* Create a folder to store the insecure key

**mkdir -m 0700 -p /home/vagrant/.ssh**

* Install the **vagrant** insecure key

**wget --no-check-certificate \**

**https://raw.github.com/mitchellh/vagrant/master/keys/vagrant.pub \**

**-O /home/vagrant/.ssh/authorized\_keys**

* Change the permissions of the key

**chmod 0600 /home/vagrant/.ssh/authorized\_keys**

* Stop the auto update processes

**sudo systemctl disable apt-daily.timer**

**sudo systemctl disable apt-daily-upgrade.timer**

**sudo systemctl disable --now unattended-upgrades.service**

* *Remove* ***cloud init*** *(if you want, if not then continue with the cache cleaning)*
* *Remove all data sources (selections) except the* ***None*** *one*

***sudo dpkg-reconfigure cloud-init***

* *Remove the package*

***sudo apt-get purge cloud-init***

* *Delete all leftovers as well*

***sudo rm -rf /etc/cloud/ && sudo rm -rf /var/lib/cloud/***

* *Reboot the VM*

***sudo reboot***

* Clean up the package cache

**sudo apt-get clean**

* Make sure that the hard disk is aligned

**sudo dd if=/dev/zero of=/EMPTY bs=1M status=progress**

**sudo rm -f /EMPTY**

* Clean the history and reboot the VM

**history -c -w && cat /dev/null > .bash\_history && sudo reboot**

Now, on the host

* Create a folder on the host to store the box, for example **ubuntu-server**
* In a terminal session, navigate to the folder created earlier
* While the VM is still running, build the box by executing

**vagrant package --base ubuntu-server**

We can publish our box to **Vagrant Cloud***(we can skip this in favor of using a local box)*

* Navigate to **https://app.vagrantup.com** and login
* Click on **New Vagrant Box**
* Enter the details and click on **Create** **box**
* Enter version and description and click **Create** **version**
* Click on **Add a provider**
* For provider set **virtualbox** and click on **Continue to upload**
* Click on the **Browse** button and navigate to the box file
* Once the upload is complete, click on the **Update provider** button
* Click on the box name
* Click on **Release...**
* Click on **Release version**

If we are not going to publish it, we can register it locally with

**vagrant box add ubuntu-server package.box**

## Using the Ubuntu Box

Use the practice files from the lecture

* Create folder **M1HW**
* Copy there the contents of folder **3-2** from the practice files
* Open the **Vagrantfile** and change both **web.vm.box** and **db.vm.box** to the box created earlier. For example, if not published, but used locally, it may be set to **ubuntu-server**
* Modify the **db.sh** file to match this

#!/bin/bash

echo "\* Add hosts ..."

echo "192.168.89.100 web.do1.lab web" >> /etc/hosts

echo "192.168.89.101 db.do1.lab db" >> /etc/hosts

echo "\* Install Software ..."

apt-get update -y && apt-get upgrade -y

apt-get install -y mariadb-server

echo "\* Adjust MariaDB connectivity ..."

sudo sed -i.bak s/127.0.0.1/0.0.0.0/g /etc/mysql/mariadb.conf.d/50-server.cnf

systemctl restart mariadb

echo "\* Create and load the database ..."

mysql -u root < /vagrant/db\_setup.sql

* Modify the **web.sh** file to match this

#!/bin/bash

echo "\* Add hosts ..."

echo "192.168.89.100 web.do1.lab web" >> /etc/hosts

echo "192.168.89.101 db.do1.lab db" >> /etc/hosts

echo "\* Install Software ..."

apt-get update -y && apt-get upgrade -y

apt-get install -y apache2 php php-mysqlnd

echo "\* Remove the default index.html file ..."

rm /var/www/html/index.html

echo "\* Copy web site files to /var/www/html/ ..."

cp /vagrant/\* /var/www/html

* Deploy the environment with

**vagrant up**

* Open browser on the host and navigate to [**http://192.168.89.100**](http://192.168.89.100)
* There should a working web application
* Destroy the machines

**vagrant destroy --force**