Handout for the Cloud Computing Tutorial at the GridKa School

Open a new shell window under your Unix-based OS or download PuTTy for Windows (http://the.earth.li/~sgtatham/putty/latest/x86/putty.exe)

Login as: **ssh -p 24 user1@141.52.174.11**
Password: **MErLh**

Some preparations

Later on you'll use ssh keys to log into your machines, therefore we should create your keys:
Type **ssh-keygen** (follow the instructions, use default values – empty passphrase!)
Make the .ssh directory readable: **chmod 755 /home/user1/.ssh**

Exploring the Cloud

Check OpenNebula commands and explore the Cloud:

onehost [list, show *id*, …]
onevnet [list, show *id*, …]
onevm [list, top, show *id*, …]
oneimage [list, show *id*, …]

Create your own Virtual Network

Create a Virtual Network template file with two fixed IP Addresses:
ip1: **141.52.174.30**
ip2: **141.52.174.31**

br0 as the bridge for the VMs and a VNET name of your choice:

Submit the created Virtual Network Template with **onevnet create *vnet_template_file*** to the OpenNebula Framework. Check it with **onevnet [list, show *id*]**

The first virtual machine

Create a VM Template file for the ubuntu image:
ubuntu-lucid

Try to start one or two VMs with **onevm create *vm_template_file***.

Try to perform some VM commands like:
onevm suspend *vm_id*
onevm resume *vm_id*
onevm migrate *vm_id host_id*
onevm delete *vm_id*

and check the status of your VM with **onevm list / onevm top** after executing.

Check out the IP of your started VM with **onevm show vm_id** and try to login via:
login: **ssh ubuntu@*vm_ip***
password: **ubuntu**

Take a look to the script file **/etc/init.d/vmcontext** which is part of the boot procedure of the VM. Try to understand how the network is configured during the boot process.

Contextualizing Virtual Machines

Create a VM template file for the ubuntu image: **ubuntu-lucid**
with a CONTEXT section. Use therefore the existing template in your home directory **$HOME/context/ubuntu.one**. Insert the execution script (**init.sh)** and your public key (**~/.ssh/id_rsa.pub**) to the CONTEXT section for password-less login.

Try to solve the following tasks in the **init.sh** script:

- import the variables from the created context.sh file
  - **. /mnt/context/context.sh**
- set a hostname of your choice
  - **hostname $HOSTNAME**
- create new user with a name of your choice
  - **useradd -s /bin/bash -b /home -m $USERNAME**
- configure passwordless login via ssh keys
  - **mkdir -p /home/$USERNAME/.ssh**
  - **cat /mnt/context/id_rsa.pub >> /home/$USERNAME/.ssh/authorized_keys**
- set the Google *dns name server 8.8.8.8* as nameserver (important for package installation via internet):
  echo "nameserver 8.8.8.8" > /etc/resolv.conf
- install some usefull packages like **htop, nano** or **lynx.**
  - **apt-get install -y $PACKAGES**

After submitting the VM, try to login with your created user account to the VM.
For debugging you may login to the VM with the following standard account:
username: ubuntu
password: ubuntu
and take a look to init.sh's output in **/var/log/context_init.log.**

Perform some rendering jobs with the CONTEXT section

Use the cloud for a distributed computing job - render a movie with 2 VMs.

Change to **$HOME/render/** and take a look to the script **render.sh** and the povray parameter file **ini.txt** and try to understand how the rendering procedure works. Modify the CONTEXT section in the VM template file. Send the following files to the VM:

- init.sh
- render.sh
- ini.txt
- povlinux-3.6.tgz
- a .pov file of your choice (to find in **$HOME/render/pov**)

Set the following variables in the CONTEXT section: ***pov_file, start_frame, stop_frame***

Modify the **init.sh:**

- import the variables from the created context.sh file
- set the Google *dns name server 8.8.8.8* as nameserver (important for package installation via internet):
- install the package **apache2**
- create a "gallery website" **/var/www/results.html** for the created pictures **frame00.png .. frame99.png:**

```
i=$START_FRAME
while [ $i -le $STOP_FRAME ]; do        # i <= STOP_FRAME
    if [ $i -lt 10 ]                    # i < 10
        then j="0$i"                    # j = 00..09
        else j="$i"                     # j = 10..STOP_FRAME
    fi
    echo "<img src=\"results/frame$j.png\" />" >> \
        /var/www/results.html
    i=`expr $i + 1`                     # i++
done
```

- create the directory **/var/www/results** in the web server root directory
- create a temp directory **/tmp/render**
- copy all files from /mnt/context to the created temp directory
- start the rendering script **render.sh** with the following parameters:
    - working directory: **/tmp/render**
    - your chosen .pov file: **$POV_FILE**
    - start frame: **$START_FRAME**
    - stop frame: **$STOP_FRAME**
    - target directory: **/var/www/results**

Submit two VMs:
The first VM should render the frames 0 to 50, the second one the frames from 51 to 99. Have a look to the created web site **http://vm_ip/results.html** to see the progress.

Change to the directory **$HOME/render/results** and try to download the rendered PNG pictures via scp or wget:
e.g.: **for i in {50..99}; do wget vm_ip/results/frame$i.png; done**

Execute the script **make_movie.sh** to create the movie.

Good luck!