Network Softwarization: Technologies and Enablers - Winter 2019
Lab 8: "Energy-efficient Telco Cloud"

Assignment
Due date: Monday, March 18 at 23:59
Total Points: 50

Note: This a team work assignment. Each team (3 or 4 students) will work on a dedicated blade server. The students register their names (or emails) at following URL:

https://docs.google.com/spreadsheets/d/1WXikPNqUSScsOtX4oPX3Tup8LCUUA-DtYgxaXHld_D8

1. Connecting to an Ericsson blade server (EBS)

OpenVPN is required to connect to the experimental environment. Download and install OpenVPN client from:

https://openvpn.net/index.php/open-source/downloads.html

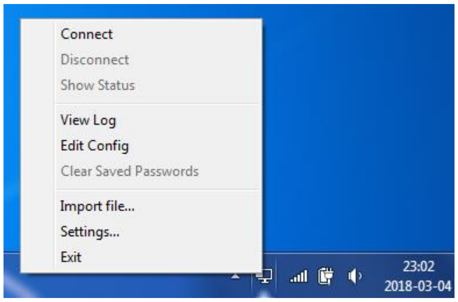
Each team will receive following information:

- OpenVPN key files (sde-xxx.ovpn) for each member to access the system
- Private key file "create-netsoft.ppk" (or "create-netsoft.pem" in case you use Linux) to open a SSH connection to an EBS
- IP and credential information of the EBS and shifted OpenStack platform.

In order to open a VPN connection, copy the provided keyfile to OpenVPN client config folder. In the Windows environment, it is:

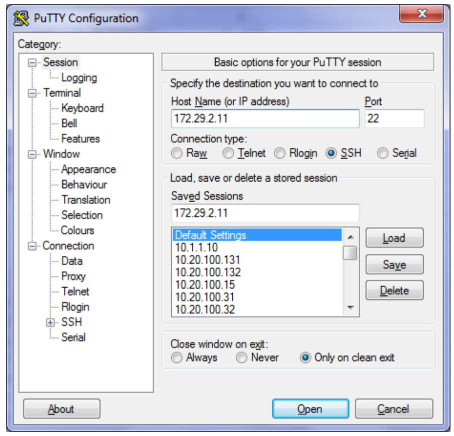
C:\Program Files\OpenVPN\config\

Launch OpenVPN to connect to the VPN network:

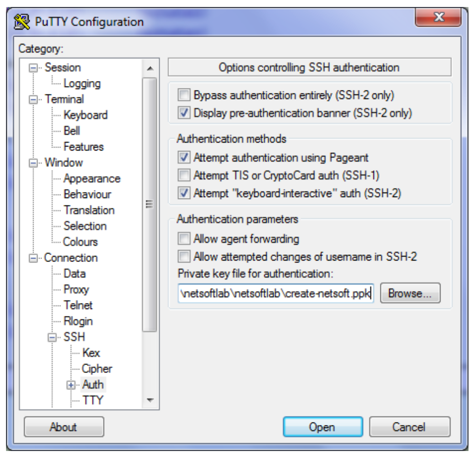


Launch a SSH client (e.g., putty) to connect to the Ericsson blade server. The putty software can be downloaded from:

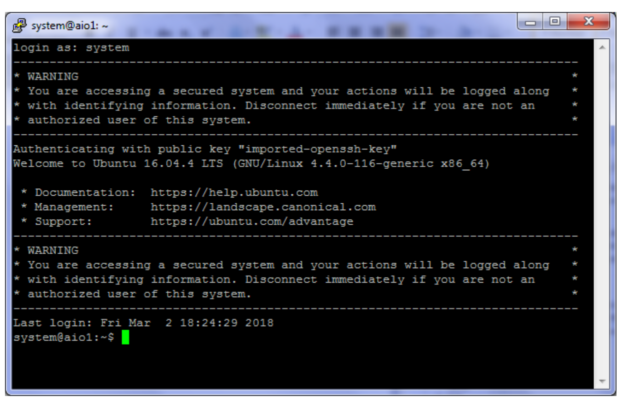
https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html



Import the SSH private key of the blade (“create-netsoft.ppk” file), as follows:



And then click “Open” to connect to the blade with provided EBS credential information:



When you are in the blade, create a folder to save your work.

#mkdir your_group_name

2. Connecting to OpenStack

To connect to OpenStack on your blade server, simply open a browser, and point to: http://your_blade_ip with the provided credential information. In the previous example, it should be: http://172.29.2.11/

Refer to the previous lab for the manual of OpenStack.

3. Helper commands

The following commands may be required to achieve your assignement.

Command	Utility
#sudo apt-get install cpufrequtils	Install the cpufrequtils package to change CPU frequency if it is not yet available in your blade
#cpufreq-info	Determine the current frequency of each CPU by the command
#sudo apt-get install sysstat	Install the systat package to collect CPU utilization
#mpstat -P ALL	Determine the load of the CPUs in the blade
#cat /sys/devices/system/cpu/cpuX/cpufreq /scaling_available_frequencies	Determine the available frequencies of the CPU X ($0 \leq X \leq 11$)
#sudo sh -c "echo -n "userspace" > /sys/devices/system/cpu/cpuX/cpufreq /scaling_governor" #sudo sh -c "echo -n YYYY > /sys/devices/system/cpu/cpuX/cpufreq /scaling_setspeed"	Changing the frequency of the CPU X ($0 \leq X \leq 11$) YYYY is an available frequency (see the command above)
#sudo sh -c "echo -n 0 > /sys/devices/system/cpu/cpuX/online"	Disable the CPU X ($1 \leq X \leq 11$) **Please do not disable the CPU 0**
#sudo sh -c "echo -n 1 > /sys/devices/system/cpu/cpuX/online"	Activate the CPU X ($1 \leq X \leq 11$)
#sudo eri-ipmitool rns 0 4 5 68 69	Collect data from temperature, voltage, and current sensors
# sudo apt-get install python-heatclient	Install the command-line OpenStack Heat client

4. Assignment (50 points)

You are required to implement five small scripts as follows. The scripts would be written in Python or in shell (e.g., bash) languages. You may also combine the scripts into a single file if it is necessary.

1. (15 points) Write a script **CollectE** to collect data from the blade server in every second. Data can be saved into a .csv file, including the following fields (fields are separated by ";"):
    - Timestamp
    - Temperature

- Power1
- Power2
- Total power
- CPU_0 utilisation (%)
- CPU_0 frequency
- CPU_1 utilisation (%)
- CPU_1 frequency
- …
- CPU_{11} utilisation (%)
- CPU_{11} frequency

The script should be run in the background to collect data in real-time.
Recall: Power1 = Voltage1 x Current1; Power2= Voltage2 x Current2; Total power = Power1 + Power2.

2. (5 points) Write a script **CreateE** to create a **random** number N of VMs (you may freely choose the bounds of N, for example: $1 \leq N \leq 8$). All VMs are created from the same flavour (for example: tempest2, 1 vCPU, 512M RAM, 1G root disk). It is recommended to use OpenStack HEAT to create the VMs.

3. (5 points) Write a script **ArrivalE** that launches the script in (2) in a **random** interval of time λ (you may freely choose the bounds of λ, for example: $1min \leq \lambda \leq 5min$). Run this script within a time T (T = 30 minutes).

4. (5 points) Write a script **ServE** that destroys the VMs created in (3) in after random interval of time μ (you may freely choose the bounds of μ, for example: $1min \leq \mu \leq 5min$). Because of the randomness in (3) and (4), there could be multiple VMs coexisting in the system in the same time. The script in (4) will end when **ArrivalE** stopped and all the VMs have been destroyed.

5. (10 points) Write a script **AdaptE** that adapts the CPU frequency and active/disabled state to save energy consumption. The script may check the CPU utilization, and change it to an optimal frequency according to the requests of creating VMs.

6. (10 points) You are required to discuss the following items in a report. Please use graphical charts to illustrate your statement.
    a. What is the average creation time (λ)?
    b. What is the average running time of a VM (μ)? In average, how many VMs are there in the system?
    c. In average, how much energy does a VM consume? How much energy does it consume for each phase: creation, running, destruction? What is the average temperature in each phase?

d. How does your energy adaptation strategy work?
e. What is your saving energy (compared to a case in which no adaptation is implemented)?
f. Propose a model of energy consumption and temperature according to the number of VMs for the: i) regular case (no adaptation), and ii) your proposed adaptation strategy.