

## Connessione alla propria macchina

To access your instance:

1. Open an SSH client.
2. Locate your private key file (yourname-exam4.pem).
3. Your key must not be publicly viewable for SSH to work. Use this command if needed:  
**chmod 400 name-exam4.pem**
4. Connect to your instance using its Public DNS:  
ec2-xx-xxx-xx-x.eu-central-1.compute.amazonaws.com
5. Use **centos** user to connect

Example:

```
ssh -i "Document/name-exam4.pem"  
centos@ec2-xx-xxx-xx-x.eu-central-1.compute.amazonaws.com
```

**Switch to root user**

**[centos@ip-xxx-xx-xx-xxx ~]\$ sudo su -**

## Exercise 1: Managing Files with Shell Expansion and Command substitution

- Utilizzare l'account **student** (se non presente crearlo)
- Creare sotto il path **/exam/exercise1/** le directory **exercise\_directoryX** con **X** compreso tra 1 e 100
- all'interno di ogni directory sotto **/exam/exercise1/exercise\_directoryX** creare i files
  - **file(a..z)\_HOSTNAME\_DATE.txt**
    - con **(a..z)**= tutte le lettere dalla a alla z
    - con **HOSTNAME** = hostname del sistema (**hostname -s**)
    - **DATE** la data di creazione file nel formato Y-m-d-H:M:S (**date +%Y-%m-%d-%H:%M:%S**)

Il risultato sarà il seguente:

```
|-- exercisel_directory1
|   |-- filea_desktop_2020-06-08-00:36:47
|   |-- fileb_desktop_2020-06-08-00:36:47
|   `-- filec_desktop_2020-06-08-00:36:47
etc...
|-- exercisel_directory2
|   |-- filea_desktop_2020-06-08-00:36:47
|   |-- fileb_desktop_2020-06-08-00:36:47
|   `-- filec_desktop_2020-06-08-00:36:47
etc...
```

## Exercise 2: Managing pipeline and regular expression

- Utilizzare il comando **ls** in maniera ricorsiva (**-R**), e l'utente **student**, per trovare sotto la directory **/usr** tutti i nomi dei file presenti che iniziano con il carattere **a** oppure il carattere **e** oppure il carattere **c**. Redirigere lo standard output sul file **/exam/exercise2/ls.txt**.

### Exercise 3: User and Group

- Create due nuovi gruppi **teachers** e **students**
  - **teachers** con GID 3000
  - **students** con GID 3001
- Creare l'utente appartenente al gruppo **teachers**: **cavatorta**
  - l'utente **cavatorta** avrà le seguenti caratteristiche:
    - **UID** 3000
    - password **cavatorta**
    - dovrà cambiare password una volta ogni 2 mesi e al primo accesso
    - dovrà poter accedere a file e directory di tutti gli utenti appartenenti al gruppo **students**
    - dovrà avere tra i gruppi secondari **anche** il gruppo **wheel**
- Creare l'utente appartenente al gruppo **students**: **giovannelli**
  - L'utente **giovannelli** avrà le seguenti caratteristiche:
    - **UID** 3010
    - l'account scadrà dopo un anno dalla sua creazione
    - dovrà avere come gruppo secondario: **users**
    - dovrà utilizzare **/bin/sh** come login shell, al posto di **/bin/bash**
    - questo utente dovrà risultare in stato di Lock (non potrà fare accesso al sistema fino a che l'utente root non disabiliterà il blocco)

### Exercise 4: file permission

- Creare una directory sotto **/exam/exercise4** dove gli utenti che possono accedere al gruppo **students** potranno condividere files
- Tutti i file creati sotto la directory **/exam/exercise4** dovranno essere assegnati automaticamente al gruppo **students**

### Exercise 5: Shell environment

- Creare una nuova variabile di ambiente chiamata **NAME** disponibile al login per **tutti** gli utenti del sistema contenente il vostro nome cognome

```
~> ssh -l student hostname
student@x.x.x.x's password:
Last login: Wed Jun 10 20:10:55 2020 from x.x.x.x
[student@hostname ~]$ echo $NAME
Luca Cavatorta
```

## Exercise 6: Bash script

- Create uno script bash sotto **/exam/exercise6** chiamato **check.sh** con le seguenti caratteristiche:
  - Accetti in ingresso o un file o una directory
  - Se quanto passato in ingresso non è un file o una directory restituisca il messaggio **"Usage: check.sh <number/directory>"**
  - Se passato in ingresso un file restituisca il messaggio **"Is a file!!"**
  - Se passato in ingresso una directory restituisca il messaggio **"Is a directory!!"**

```
bash check.sh file
Is a file!!
bash check.sh directory/
Is a directory
bash check.sh noFile
Usage:  check.sh <number/directory>
bash check.sh
Usage:  check.sh <number/directory>
```

## Exercise 7: Bash script

- Create uno script chiamato **/exam/exercise7/disableuser.sh**
  - Lo script accetterà in ingresso due parametri: il primo riferito all'utente da gestire e il secondo corrispondente alla azione da intraprendere **lock;unlock**
  - Qualunque altro argomento passato dovrà restituire il messaggio:  
**"Usage: /exam/exercise7/disableuser.sh <username> lock/unlock"**
  - Il comando lock andrà ad effettuare il lock della utenza; il comando unlock andrà a cancellare il lock utente
  - Le operazioni di **lock** e **unlock** dovranno essere implementate tramite l'utilizzo di **funzioni**.
  - Se l'utente non è presente lo script dovrà uscire e restituire il messaggio **"ERROR: username not found"**
  - **NB. si utilizzi il case statement per la gestione dello script.**

## Exercise 8: Firewallld

- Configurazione di port forwarding.
  - Create una regola di port forwarding **permanente** sul vostro sistema in ingresso sulla porta 8081/tcp verso la porta 22/tcp.
  - Potete testare la nuova regola facendo connessione ssh al vostro sistema sulla porta 8081 con l'opzione -p 8081 del comando ssh.

## Exercise 9: Systemd

- Installare sul vostro sistema il servizio memcached tramite yum
  - Il servizio deve essere attivato al boot della macchina

## Exercise 10: Docker

Install docker e docker-compose

- `yum install -y docker`
- enable docker to start at boot (`systemctl enable docker`)
- avviare docker (`systemctl start docker`)
- Creare la propria immagine basata su **centos:7**
- La directory **/exam/exercise10** dovrà contenere i files:
  - **Dockerfile**
  - **dockerexam.sh**
- **dockerexam.sh** sarà lo script bash il cui compito sarà quelli di stampare "hello <name>!!" il parametro name dovrà essere una variabile di ambiente contenente di default il valore "exam"
- L'immagine dovrà chiamarsi `exam/myhelloexam`

Risultato:

```
docker run exam/myhelloexam
```

```
Hello Exam!!
```

```
docker run -e ENV_VAR_NAME="Luca" exam/myhelloexam
```

```
Hello Luca!!
```

## Exercise 11: Docker compose

- Creare una applicazione wordpress mysql tramite docker-compose con le seguenti caratteristiche
  - porta **locale** wordpress 8080
  - DB\_USER=student
  - DB\_PASSWORD=stUdEnt01
  - DB\_NAME=exercise11
  - Utilizzate un bind locale per i volumi dati delle applicazioni sotto
    - /exam/exercise11/worpress per wordpress
    - /exam/exercise11/mysql per mysql
- Utilizzate le immagini ufficiali dal Docker Hub (**hub.docker.com**)
- Una volta attiva l'applicazione, si dovrà poter accedere dall'esterno sulla porta 8080 (regole firewall)
- Il file **docker-compose.yml** dovrà essere messo sotto **/exam/exercise11**

## Question 1:

- Applicazioni installate su sistema operativo standard, e in container. Quali le differenze principali