Connessione alla propria macchina

1. Aprire il client SSH .
2. Posizionarsi nella directory di salvataggio del file .pem allegato alla mail.
3. Tramite il client SSH, utilizza il seguente comando al fine di impostare le autorizzazioni del file della chiave privata in **read only** per l'utente owner.
   chmod 400 keyPairName.pem
4. Nella finestra del terminale, utilizzare il comando ssh per connettersi all'istanza. Specificare il percorso e il nome del file della chiave privata (.pem), il nome utente per l'istanza e il nome DNS pubblico per l'istanza.
5. Utilizzare la username **ec2-user** per la connessione al sistema

Esempio:
ssh -i "**keyPairName.pem**"
ec2-user@ec2-xx-xxx-xx-x.eu-central-1.compute.amazonaws.com

Per passare all'utente **root** utilizzare il comando "sudo su -"
[ec2-user@ip-xxx-xx-xx-xxx ~]$ **sudo su -**

Exercise 1: Managing Files with Shell Expansion

- Creare la seguente struttura sotto le directory **/exam/exercise1/exercise1_dirX/fileY** con **X** compreso tra 1 e 10 e **Y** compreso tra 10 e 30

```
exercise1_dir1
     |
      ----- file10.txt
      ----- etc..
      ----- file20.txt
exercise1_dir2
      |
      ----- file1.txt
etc...
```

Exercise 2: Managing regular expression and command substitution

- Utilizzare l'utente **student** per trovare tutti, e solamente i file (non le directory) presenti su tutto il file system (**/**) che contengono il pattern "**text**" (si consiglia di utilizzare il comando **find)**. Per ogni file trovato determinare il **file type** utilizzando il comando **file** e redirigere l'output su **/exam/exercise2/find.txt.** Eventuali errori andranno rediretti sul file **/exam/exercise2/error.txt**

Esempio di contenuto del file find.txt:

```
/sys/module/xfrm_user/sections/.text:                       regular file, no read permission
/usr/share/groff/1.22.3/font/devps/generate/textmap:        C source, ASCII text
/usr/share/groff/1.22.3/font/devps/text.enc:                ASCII text
```

Exercise 3: User Group and default permission

- Create due nuovi gruppi **students** e **exam**
    - **students** con GID 2000
    - **exam** con GID 2001
- Creare l'utente appartenente al gruppo **exam**: **thomas**
- l'utente **thomas** avrà le seguenti caratteristiche:
        - **UID** 3000
        - home directory **/home/exam/thomas**
        - password **thomas**
        - dovrà cambiare password una volta ogni 5 mesi
        - dovrà poter accedere a file e directory appartenenti al gruppo **students**
        - l'account scadrà dopo due anni dalla sua creazione
        - fare in modo che tutti i file e directory creati dall'utente **thomas** di default non abbiano nessun permesso (rwx) per gli utenti **other**

Exercise 4: File permission

- Creare la directory **/exam/exercise4/collaboration/students** dove gli utenti facenti parte del gruppo **students** potranno condividere file e directory
- Fare in modo che tutti i file e directory creati sotto **/exam/exercise4/collaboration/students** vengano assegnati automaticamente al gruppo **students**

esempio:

```
[root@desktop ~]# cd /exam/exercise4/collaboration/students
[root@desktop students]# touch example-root
[root@desktop students]# ls -la example-root
-rw-r--r-- 1 root students 0 Feb 10 22:09 example-root
[root@desktop students]# su - thomas
[thomas@desktop ~]$ cd /exam/exercise4/collaboration/students
[thomas@desktop thomas]$ touch example-thomas
[thomas@desktop thomas]$ ls -la example-thomas
-rw-r------ 1 thomas students 0 Feb 10 22:10 example-thomas
```

Exercise 5: Environment

- Creare un nuovo comando chiamato **hello** per l'utente **student** il cui compito sarà quello di stampare in output la stringa "**hello student**"
- Posizionare il vostro comando sotto la directory **/exam/exercise5/bin**
- Modificare l'environment dell'utente student in modo da poter richiamare il comando senza specificare il path (la modifica dell'environment dovrà essere resa persistente al login utente).

esempio:

```
$ su - student
$ hello
hello student
$ cd /etc
$ hello
hello student
```

Exercise 6: Firewalld: Configure port forwarding

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

Exercise 7: Bash script

- Create uno script bash chiamato **/exam/exercise7/print.sh** con il seguente comportamento:
    - accetti in ingresso due parametri
        - come primo parametro un numero intero
        - come secondo parametro una stringa
    - stampi in output la stringa passata per **N** volte, con N pari al numero intero passato in ingresso
    - si predisponga un controllo sul numero di parametri passati in ingresso che esca riportando un errore a piacimento se diverso da 2

example:

```
[root@desktop ~]# bash print.sh 3 "hello world"
hello world
hello world
hello world
[root@desktop ~]# bash print.sh 3
error
[root@desktop ~]# bash print.sh 3 “hello world” “hello student”
error
[root@desktop ~]# bash print.sh
error
```

Exercise 8: Bash script

- Create uno script chiamato **/exam/exercise8/createdir.sh**
- Lo script accetti in ingresso come parametro un file contenente un elenco di directory da creare
- Lo script dovrà creare le directory e sottodirectory presenti in elenco, solamente se non presenti, altrimenti riporti che la directory è già presente.
- Se passato allo script nessun parametro, esca con il messaggio **usage: createdir.sh <filename>**
- Se passato allo script come parametro un file inesistente, esca con il messaggio **file not found**
- Il path da cui partire per creare le sotto directory passate in elenco deve essere **/exam/exercise8/dircontainer/**

example

```
[root@desktop students]# cat dirlist
testdir1/test
testdir2/test2/test1
[root@desktop students]# bash createdir.sh dirlistfile
/exam/exercise8/dircontainer/testdir1/test already exists nothing
to do
/exam/exercise8/dircontainer/testdir2/test2/test1 created
[root@desktop students]# bash createdir.sh
usage: createdir.sh <filename>
[root@desktop students]# bash createdir.sh someFile
someFile: file not found
```

Exercise 9: HTTPD Dynamic content

- Installare sul sistema il servizio HTTP/Apache
- Fare in modo che il servizio sia attivo al boot della macchina
- Fare in modo che il servizio Apache sia in ascolto **solo** sulla porta **8080** e aggiungete la regola firewall corretta per poter accedere dall'esterno al servizio.
- Il web server dovrà erogare contenuti dinamici utilizzando il linguaggio di scripting PHP
- Fare in modo che la Document Root impostata per il vostro servizio sia **/exam/exercise9/**
- Creare il file **/exam/exercise9/index.php** con il seguente contenuto

```
<html>
 <head>
  <title>PHP Exam2</title>
 </head>
 <body>
 <?php echo '<p>Hello Exam2</p>'; ?>
 </body>
</html>
```

- Potete verificare che il tutto funzioni dal browser locale alla vostra postazione collegandovi all'indirizzo IP della vostra macchina AWS:
    - http://ec2-xx-xxx-xx-x.eu-south-1.compute.amazonaws.com:**8080**/index.php

Exercise 10: Docker

- Creare una immagine docker chiamata exercise10 basata su centos:8
- L'immagine dovrà avere installato un Apache server con impostazioni di default
- Il server apache interrogato dovrà restituire il contenuto:

```
Hello Docker!!
```

- Predisponete nel modo che ritenete più opportuno la possibilità di modificare l'output del server con una frase passate come environment allo start del container.
- Creare il file **/exam/exercise10/docker-compose-exam.yaml** per gestire lo start del vostro container, environment da passare e porta locale (**80**) al sistema per poter interrogare il server Apache dall'esterno.

Question

- Si descriva a parole la differenza tra metodo dichiarativo e imperativo riportando un esempio associato anche alla creazione di oggetti all'interno del cluster Kubernetes
    - Salvare la risposta sotto **/exam/question/**