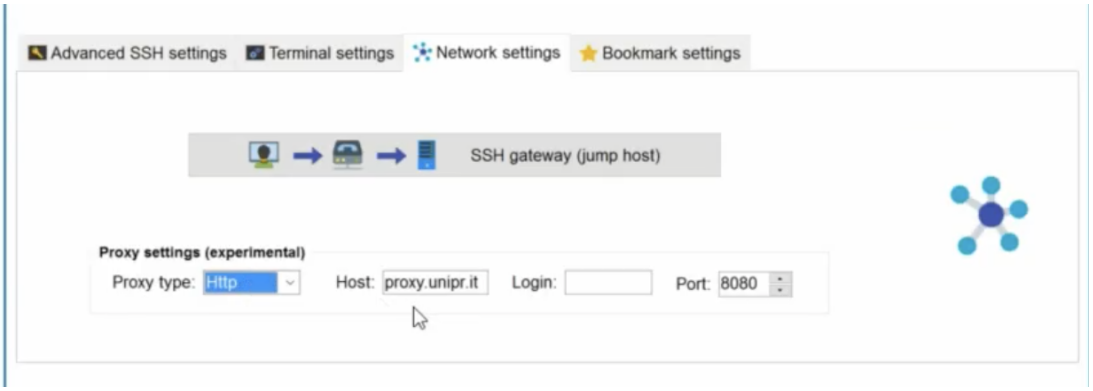
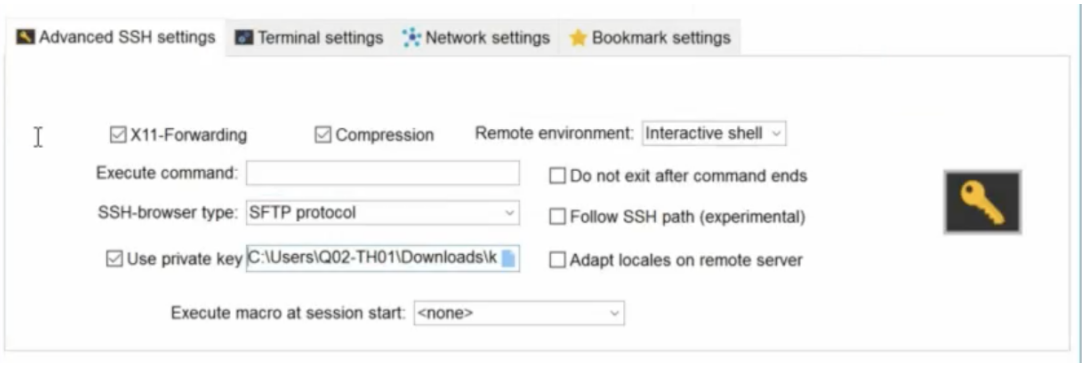
Connessione alla propria macchina

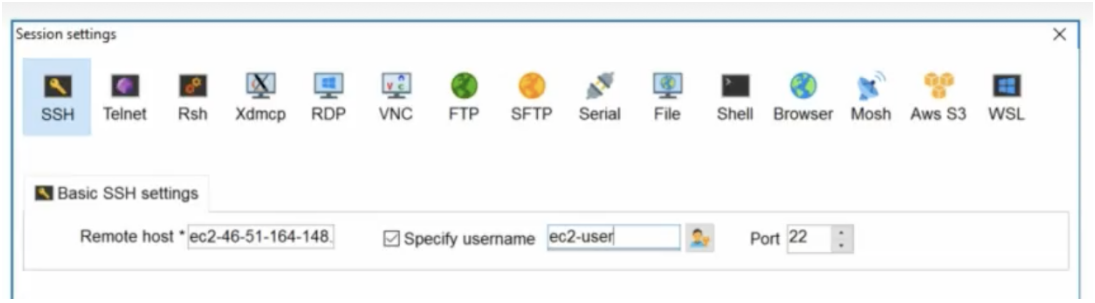
1. Aprire MobaXterm.
2. Tab session / SSH
3. Configurare il proxy dal tab Network Setting
    a. Httpd
    b. proxy.unipr.it
    c. 8080



4. Passare la chiave SSH dal tab “Advanced SSH settings”



5. Inserire in Remote Hosts il dns della vostra istanza Linux
    a. esempio: ec2-xx-xxx-xx-x.eu-central-1.compute.amazonaws.com
6. Username: **ec2-user**
7. port: **22**



Video accesso ai sistemi cloud dalle postazioni UNIPR:
accesso_aws_pld_laboratori.mov

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

Exercise 1: Systemd and Firewalld

- Installare sul vostro sistema il servizio **vsftpd** tramite **yum**
    - Il servizio deve essere attivato al boot della macchina
    - Individuare il **pid** del servizio tramite **systemctl**
    - verificare su quale porta il servizio è in listening con il comando

```
netstat -tulpen | grep <pidNumber>
esempio:
netstat -tulpen | grep 12345
tcp        0      0 0.0.0.0:<PortNumber>
0.0.0.0:*               LISTEN      12345
```

- Aggiungere la porta (NON IL SERVIZIO) appena individuata alle regole firewall per il protocollo **TCP** (ricordarsi di rendere permanente la regola firewall)

Exercise 2: Alias Command

- Creare un alias chiamato **process** che stampi a video tutti i processi dell'utente che lo lancia.
- Creare un alias command **lwc** che stampi a video il numero di file e directory della directory corrente (suggerimento: si utilizzi **ls -1)**

Exercise 3: User

- Creare l'utente **mark** appartenente al gruppo **facebook**.
- Fare in modo che nuovi file e directory creati dall'utente **mark**, possano essere letti scritti o visti da tutti gli utenti del sistema.
- L'utente dovrà cambiare password ogni 30 giorni
- L'utente dovrà cambiare password al primo login
- La home directory dell'utente sarà /exam/exercise3

Exercise 4: Managing Files with Bash

- Creare uno script bash sotto **/exam/exercise4/** chiamato **create_dir.sh** che:
    - sotto il path /exam/exercise4/ vada a creare la directories **exercise_directory_mm_yyyy** con **mm** compreso tra 1 e 12; yyyy=2000 e yyyy=2023
    - sotto ogni directory creata /exam/exercise4/exercise_directory_mm_yyyy creari i files **file(a..z)_DATE.txt**
        - con **(a..z)**= tutte le lettere dalla a alla z
        - **DATE** la data di creazione file nel formato ymd (**date +%y%m%d**)

Exercise 5: Bash script

Create uno script bash sotto **/exam/exercise5** chiamato **configure_httpd.sh** con le seguenti caratteristiche:

- accetti in ingresso al minimo un parametro:
  - **install**: lo script andrà ad installare il servizio HTTPD
    - una volta installato (se completato con successo) si riporti in output un messaggio di avvenuta installazione

    esempio:
    ```
    bash /exam/exercise5/configure_httpd.sh install
    httpd service installed
    ```
  - **uninstall**: lo script andrà ad rimuovere il servizio HTTPD
    - una volta rimosso (se completato con successo) si riporti in output un messaggio di avvenuta rimozione

    esempio:
    ```
    bash /exam/exercise5/configure_httpd.sh uninstall
    httpd service removed
    ```
  - **restart**: lo script andrà ad effettuare il restart del servizio HTTPD
    - una volta riavviato (se completato con successo) si riporti in output un messaggio di avvenuta rimozione

    esempio:
    ```
    bash /exam/exercise5/configure_httpd.sh restart
    httpd service restarted
    ```
  - **configure**: in questo caso lo script si aspetterà ulteriori due parametri per definire la configurazione di Listen port (inizializzare una variabile chiamata **PORT**) e DocumentRoot (inizializzare una variabile chiamata **DOCUMENTROOT)**
    Sotto **/exam/exercise4** troverete il file **httpd_template.conf** che utilizzerete per popolare il file **httpd.conf** del servizio HTTPD
    Il file **httpd_template.conf** contiene già una parametrizzazione basata su due variabili chiamate appunto PORT e DOCUMENTROOT utilizzate il tool **envsubst** per andare a preprare **httpd.conf** a partire da **httpd_template.conf**
    la DocumentRoot directory andrà creata sul sistema (assegnare **rwxrwxrwx** come set di permessi)
    Lo script crei una **index.html** contenente il testo “hello exam” sotto DocumentRoot
    esempio:
    ```
    bash /exam/exercise5/configure_httpd.sh configure 8080
    /document/root
    httpd service configured
    ```
- gestione di errori e controlli
  - Se non viene passato nessun parametro lo script esca con il messaggio
    - USAGE: configure_httpd.sh <install|uninstall|restart>
    - USAGE: configure_httpd.sh <configure> <port> <documentroot>

- In caso di parametro sia **configure** predisporre un ulteriore controllo per verificare che **port** e **documentroot** vengano passati altrimenti esca riportando il messaggio di errore
    - USAGE: configure_httpd.sh configure <port> <documentroot>

Exercise 6: docker-compose

- La directory **/exam/exercise6** dovrà contenere i seguenti files e directory:
    - **Dockerfile**
    - **entrypoint.sh**
    - **docker-compose.yml**
    - la directory **content**
- Il servizio tramite **docker compose** dovrà gestire l'applicazione hello-exam.
- **Dockerfile** conterrà le istruzioni per la gestione della vostra applicazione in container il cui servizio dovrà essere avviato tramite script di ENTRYPOINT.
  Nessun vincolo su immagine di base ecc….
- **entrypoint.sh** avrà il compito di scrivere sul file exam.txt il valore della variabile di ambiente EXAM per 6 volte e poi uscire
- **docker-compose.yml** verrà utilizzato per:
    - gestire start/build della immagine
    - inizializzare la variabile di ambiente EXAM con valore a piacere
    - gestire il bind locale con la directory /exam/exercise6/content che conterrà il file **exam.txt** popolato dallo script di ENTRYPOINT della applicazione in container

Question :

le risposte andranno messa sotto la directory **/exam/question/**

- Si descriva a parole come in kubernetes viene gestito il workload applicativo delle applicazioni in container attraverso l'oggetto Pod. Di cosa si tratta, quali le caratteristiche principali. In che modo, e come, questo oggetto viene gestito dallo oggetto di workload di più alto livello ReplicaSet