Resolución del TP

El Tp es bastante amigable y va diciendo los pasos iniciales de forma exacta:

1. Crear un repositorio personal que se llame TP_AySO: Crearse cuenta en Git, nuevo repositorio…
2. Vincular con el repo la VM:
    - Desde la VM ejecutar el comando “ssh-keygen” y ejecutar “[Enter]” en todos los parámetros que pida.
    - De esta forma se generará un juego de claves de ssh públicas y privadas en el directorio home del usuario “$HOME” en la carpeta oculta .ssh
    - Ver el contenido del archivo (dentro del home del usuario, en la carpeta oculta “.ssh” el archivo con la extensión .pub) donde se encuentra la clave pública.
    - cat $HOME/.ssh/*.pub
    - copiar la misma en “github.com” dentro del perfil del usuario..
3. En la VM: crear una carpeta repogit: mkdir repogit
4. Clonamos el repositorio dentro de la carpeta “$HOME/repogit/” y nos posicionamos dentro del repositorio: git clone **<URL>**
5. Entramos y listamos: cd camino ls-l

Hasta acá la configuración del repositorio por ssh

Continuamos:

Ver el contenido del archivo de configuración “os-release”:

cat /etc/os-release

Verificar en la ayuda del comando grep como ignorar las mayúsculas y minúsculas. Posteriormente realizar una búsqueda de HOME_URL en el archivo anteriormente citado. /etc/os-release

Hacemos un

man grep

y vemos que -i, --ignore-case

Entonces:

grep -i Home_url /etc/os-release

Mostramos por pantalla el nombre de usuario con el que estamos logueado: whoami

Creamos un archivo datos_usuario.txt con el siguiente formato “usuario:<nombre-del-usuario>

echo “Usuario=$(whoami)” > datos_usuario.txt

Practicamos ahora con git:

Agregar todos los archivos generados en el Staging Area del repositorio local de git.

git add .

git config --global user.email "TU-email"

git config --global user.name "Tu-Usuario"

git commit -m "ADD: agregado 1er ejercicio sobre datos_usuarios.txt"

git status

git push origin main

Añadir al archivo anterior la información de modelo de cpu, obtenida del archivo cpuinfo, ubicado

en el sistema de archivos virtual, /proc filtrando para que solamente agregue 1 entrada,

independientemente de la cantidad de cores que tenga su vm.

grep "model name" /proc/cpuinfo | head -n1 >> datos_usuarios.txt

Volvemos a agregar al stage área del repositorio local el archivo modificado y volvemos a comitear. Usar como comentario: “feat: Añadiendo información de CPU”

cd /ruta/al/directorio/…

git add datos_usuarios.txt

git commit -m "feat: Añadiendo información de CPU"

git push origin main

Crear un archivo “lista_ordenada”, obteniendo los 10 primeros usuarios (/etc/passwd) que el intérprete de comandos NO sea “nologin” , y ordenar dicha lista por el intérprete de comandos (Columna 7)

grep -v 'nologin' /etc/passwd | head -n 10 | sort -t ':' -k 7 > lista_ordenada

Desglose del comando:

grep -v 'nologin' /etc/passwd:

grep: Busca líneas en el archivo.

-v: Muestra las líneas que no coinciden con el patrón especificado.

/etc/passwd: Es el archivo que contiene información sobre las cuentas de usuario del sistema. Cada línea en este archivo representa un usuario.

Este comando filtra todas las líneas que contienen nologin, dejando solo las que tienen otros intérpretes de comandos.

| head -n 10:

|: La tubería redirige la salida del comando anterior como entrada del siguiente.

head -n 10: Muestra solo las primeras 10 líneas de la salida anterior. Así, obtendrás los primeros 10 usuarios que no tienen nologin como intérprete de comandos.

| sort -t ':' -k 7:

|: Redirige la salida de head como entrada a sort.

sort: Ordena las líneas de entrada.

-t ':': Define el delimitador como: (dos puntos), que es el separador en el archivo /etc/passwd.

-k 7: Indica que la ordenación se debe hacer en la séptima columna (la que corresponde al intérprete de comandos).

> lista_ordenada:

>: Redirige la salida del comando sort al archivo lista_ordenada. Si el archivo no existe, se creará; si existe, se sobrescribirá.

GIT!!!

cd /ruta/al/directorio/…

git add lista_ordenada

git commit -m "ADD: Listado de usuarios ordenada"

git push origin main

Buscar con qué comando puede realizar un "dump traffic" de la red…. dejar el nombre del comando en el archivo “comando_dump_net” y agregar también que comando ejecutar para realizar dicha búsqueda.

apropos dump

echo "tcpdump" > comando_dump_net

GIT!!!

cd /ruta/al/directorio/….

git add comando_dump_net

git commit -m "ADD: Comando para Capturar y analizar el tráfico de red"

git push origin main

Usando técnicas de HereDoc y redireccionamiento, Agregar al README.md la siguiente

información.

Alumno: <Tu-Nombre>

División: <Tu-División>

Turno: <Turno>

cat << EOF >> README.md

Alumno: Guille Soler

División: 1P2C

Turno: Tarde

EOF

● SIN MOVERSE DEL DIRECTORIO, utilizando PATHS Relativo…

● Forzar la Escritura del archivo donde se encuentra el historial de comandos history -a agrega (append) los comandos nuevos al archivo $HOME/.bash_history

● Copiar dentro del repositorio el archivo .bash_history que se encuentra en el home del usuario dejándolo con el nombre "Historial_comandos_<Tu-Nombre>.txt"

● Finalmente deberá realizar el último commit con su respectivo comentario y pushear los cambios contra el repositorio remoto de github

history -a

cp $HOME/.bash_history ./Historial_comandos_Guille.txt

git add Historial_comandos_Guille.txt

git commit -m "ADD: Historial de comandos del alumno"

git push origin main