Giacomo Ferro 23/05/2017

RELAZIONE DELL'ELABORATO DI SHELL

Struttura generale dell'elaborato:

Ho deciso di scrivere l'elaborato in un unico file con solo due funzioni di nome "menu" e "stampaScelte" che stampano rispettivamente il menù principale e le operazioni di modifica utente del punto 5. Subito dopo, ho creato due if() che controllano se i parametri inseriti rispettano la consegna: nel primo if() ritorno in output errore se i parametri inseriti da riga di comando sono inferiori o maggiori di due mentre, se ci sono due argomenti, prima controllo che i percorsi siano validi, poi prendo la parte terminale dei percorsi dati tramite il comando basename e poi verifico se coincidono con le stringhe "passwd" e "group". In caso contrario ritorno errore in output. Preciso che la mia prima implementazione di quest'ultimo punto era con l'utilizzo del comando find con un file di supporto; ma ho deciso di modificarlo perché durante la ricerca venivano generati dei messaggi che segnalavano errori poiché non possedevo tutti i permessi per cercare in alcune cartelle protette del sistema anche se funzionava comunque.

Tutto il programma ad eccezione di quello menzionato sopra è incluso in un unico ciclo while() con dei selettori if() al suo interno che permettono di eseguire le operazioni richieste. La condizione del while() sarà vera finchè l'utente non digiterà '7' che corrisponde al segnale di terminazione del programma.

Ora proseguo spiegando come ho risolto le **sei** operazioni richieste.

Stampa di tutti gli utenti con gruppo associato:

Prima di tutto eseguo una copia temporanea di passwd e poi conto quante sono le righe in passwd con il comando "cat $1 | wc -l > riga.txt" salvando il risultato in un file. Inizializzo una variabile (diff) che mi servirà dopo per aggiornare il file passwdTemp.txt eliminando di volta in volta la prima riga con la concatenazione dei comandi "cat passwdTemp.txt | head -$diff | tail -1".

A questo punto creo un ciclo while() che opera finchè un contatore non è uguale al numero di righe di passwd. Ad ogni ciclo eseguo le seguenti istruzioni:

- Con il comando "cat passwdTemp.txt | head -1 | cut -d: -f4 > GID.txt" salvo il GID dell'utente in prima riga
- Ricerco il GID in group e salvo nel file gruppo.txt il nome del gruppo corrispondente
- Prelevo il nome utente con il comando "cat passwdTemp.txt | head -1 | cut -d: -f1 > utente.txt"
- Associo i valori contenuti in utente.txt e GID.txt e li stampo
- Aggiorno il file passwdTemp.txt senza la prima riga (che ho appena processato) con i due comandi "cat passwdTemp.txt | tail -$diff > passwdTemp2.tx" e "cp passwdTemp2.txt passwdTemp.txt"
- Aggiorno le variabili di ciclo e ricomincio

Da ultimo elimino i tutti i file precedentemente creati se il file passwd non è vuoto; in questo caso infatti elimino sono passwdTemp.txt e riga.txt poiché gli altri file non sono stati creati dal programma.

Stampa dell'elenco dei gruppi:

Creo un file di nome "comando2.txt" e inserisco il titolo. Poi con il comando "cut -d: -f1 $2 > comando2.txt salvo solo i nomi dei gruppi (che sono nella prima colonna del file). Infine stampo a video il file comando2.txt e poi lo rimuovo.

Stampare gli utenti per gruppo:

Chiedo il nome del gruppo da elaborare poi, dopo aver fatto una copia di passwd, controllo se il nome del gruppo esiste nel file gruppi.txt (in cui sono presenti solo i nomi dei gruppi). Se il nome è presente entro nell'else dove in fondo è presente un ciclo while() di stampa.

I passaggi sono riassunti nei seguenti punti:

- con il comando "grep -xn "$gruppo" gruppi.txt | cut -d: -f1 > riga.txt" salvo il valore della riga in cui ho trovato la precisa occorrenza del nome del gruppo
- dopo aver aggiunto il GID dei gruppi nel file gruppi.txt con il comando "cat gruppi.txt | head -$(cat riga.txt) | tail -1 | cut -d: -f2 > GID.txt" salvo il GID del gruppo desiderato
- procedo a ricercare gli utenti associati prelevando tutti i GID dei gruppi degli utenti
- con il comando "grep -wn "$(cat GID.txt)" UserGID.txt | cut -d: -f1 > riga.txt" salvo il numero di riga degli utenti che corrispondono al GID del gruppo

Se il file riga.txt è vuoto allora ritorno a schermo l'avviso che non ci sono utenti associati al gruppo, altrimenti procedo inizializzando una variabile diff pari al numero totale di righe -1. Servirà per aggiornare il file che contiene il numero di righe degli utenti associati al gruppo.

Il ciclo while() è organizzato come segue:

- salvo ogni volta il valore in prima riga nel file riga.txt con il comando "rigaUser=$(cat riga.txt | head -1)"
- "cat passwdTemp.txt | head -$rigaUser | tail -1 | cut -d: -f1 >> comando3.txt" con questo comando prelevo il nome utente dalla riga dell'n-esimo utente appartenente al gruppo selezionato
- Successivamente aggiorno il file riga.txt tramite il comando "cat riga.txt | tail -$diff > riga2.txt" rimuovendo la prima riga ogni volta poi aggiorno riga.txt con "cp riga2.txt riga.txt" e aggiorno la variabile diff-=1 per aggiornare il file al ciclo successivo
- Infine stampo il file comando3.txt contenente gli utenti associati al gruppo

Inserire un nuovo utente:

Chiedo tutti i parametri all'utente e dopo li controllo con dei costrutti if con una grep -c che ritorna il numero di righe con l'occorrenza. Se per il nome utente e per l'UID il valore di ritorno dalla ricerca è maggiore di zero ritorno un errore e, se il nome del gruppo non è presente, ritorno anche in questo caso errore. Inoltre controllo anche che il nome utente e la password non siano più lunghi di 32 caratteri.

Se tutti i tutti gli if() sono falsi allora procedo all'inserimento del nuovo utente. Per farlo devo ricercare il GID del gruppo e salvarlo per metterlo nella riga del nuovo utente.

Per salvare il GID digito:

grep -xn "$gruppo" gruppi.txt | cut -d: -f1 > riga.txt;

cat $2 | head -$(cat riga.txt) | tail -1 | cut -d: -f2 > GID.txt

In questo modo salvo la riga in cui ho trovato l'occorrenza del nome del gruppo, poi con il secondo comando prelevo il GID dalla riga e lo salvo.

Infine eseguo una echo con le nuove informazioni dell'utente e lo accodo nel file originale passwd.

Modifica di un profilo utente:

Chiedo il nome dell'utente da modificare e, con lo stesso metodo del punto precedente, controllo se l'utente esiste. Se esiste lancio la funzione che richiede quale campo modificare per l'utente desiderato.

Poi con degli if() seleziono la scelta desiderata tra 1, 2, 3, 4, 5.

1. Modifica della password: chiedo la nuova password e controllo che non sia più lunga di 32 caratteri, poi cerco con l'istruzione: grep -xn "$nome" nomi.txt | cut -d: -f1 > riga.txt il numero della riga che corrisponde all'utente da modificare. Successivamente salvo in nuovoUt.txt la riga aggiornata dell'utente. Dopo aver salvato in UserKey.txt la riga dell'utente da eliminare prelevo i suoi primi 4 campi con il comando: grep -w "$nome" UserKey.txt | cut -d: -f1-4 > riga.txt. Elimino l'user vecchio tramite il comando: sed /"${riga}"/d passwdTemp.txt > NewPasswdTemp.txt con riga=$(cat riga.txt). Da ultimo, inserisco in coda a passwd l'utente tramite il comando cat nuovoUt.txt >> $1.
2. Modifica del gruppo di appartenenza: chiedo il nome del nuovo gruppo e ne verifico l'esistenza con un if() tramite le solite modalità. Se esiste, tramite il comando: grep -xn "$NewGroup" gruppi.txt | cut -d: -f1 > riga.txt salvo il numero di riga del nuovo gruppo e poi con il comando successivo (cat $2 | head -$(cat riga.txt) | tail -1 | cut -d: -f2 > GID.txt) ne salvo il GID corrispondente. La parte successiva funziona esattamente come al punto precedente con l'unica differenza che in nuovoUt.txt i campi saranno diversi.
3. Modifica delle informazioni sull'utente: chiedo la nuova informazione e procedo esattamente con gli stessi passaggi dei punti precedenti sempre con la differenza che nel nuovo utente il campo modificato sarà quello delle informazioni.
4. Modifica del path per la home: chiedo il nuovo percorso per la home e procedo esattamente con gli stessi passaggi dei punti precedenti sempre con la differenza che nel nuovo utente il campo modificato sarà quello della home.
5. Modifica del path per la bash: chiedo il nuovo percorso per la bash e procedo esattamente con gli stessi passaggi dei punti precedenti sempre con la differenza che nel nuovo utente il campo modificato sarà quello della bash ovvero l'ultimo.

Eliminazione di un utente:

Chiedo il nome dell'utente da eliminare poi controllo se il file passwd non è vuoto e se esiste il nome utente nel file passwd. Se il programma supera questi controlli allora salvo in riga.txt il numero della riga da eliminare da passwd con: grep -xn "$SearchedUser" nomi.txt | cut -d: -f1 > riga.txt. Poi prelevo la riga intera dell'utente da eliminare con il comando: cat passwdTemp.txt | head -$(cat riga.txt) | tail -1 > UserKey.txt. Con il comando successivo (grep -w "$SearchedUser" UserKey.txt | cut -d: -f1-4 > riga.txt) prelevo solo i primi 4 campi dell'utente da eliminare. E poi con riga=$(cat riga.txt) salvo in una variabile il pattern da fornire alla sed per eliminare l'utente selezionato. Infine elimino l'utente con la sed e aggiorno il file passwd originale.

Uscita dal programma e controlli finali:

Se l'utente inserisce un numero maggiore di 7 ritorno un errore in output e restituisco il menù iniziale. Il ciclo while() termina solo se inserisco 7. Infatti se l'utente inserisce 7 esco dal while() e ritorno una stampa di terminazione.