Progetto Bash A.A. 2020/2021

Lo script bash "userManager.sh" per la gestione degli utenti in ambiente UNIX/Linux, prevede un file principale (appunto "userManager.sh") e vari file ad esso collegati, chiamati dallo stesso tramite un menù.

userManager.sh

Il menù

Il file userManager.sh visualizza un menù con varie funzionalità. Il menù è stato implementato sfruttando un ***ciclo do-while***: il programma rimane nel ciclo se la variabile m, a cui viene assegnato fuori dal ciclo valore 1, ha valore minore di 8.

Nel corpo del ciclo viene stampato il menù e letta la variabile da tastiera. Viene eseguito un ***if***, così da poter stampare un messaggio di errore se si inserisce un valore non previsto dal menù, maggiore di 8, prima della terminazione automatica del programma. (caso particolare)

Se il valore è <= a 8 viene eseguito un ***case***: ogni valore da 1 a 7 porta alla lettura di un file esterno "Opzione_*num*.sh" (dove per *_num* intendo il numero dell'opzione scelta – es. Opzione1.sh, Opzione2.sh, ecc). Digitando il numero 8 invece l'utente termina il programma.

Il comando --help

Per l'implementazione del comando --help è stato usato un ***if*** che legge il valore del primo argomento dello script, disponibile come la variabile ***$1***; se questa è uguale a --help allora stampa le opzioni (che non ci sono, ma potrebbero essere implementate allo stesso modo) ed una piccola guida.

.config.txt

È un ***hidden file*** di configurazione, all'interno è semplicemente scritto un numero (in questo caso **1000**) che è il primo ID utente dal quale si inizia a lavorare. All'inizio di ogni file "Opzione_*num*.sh" è creata una variabile ***limit***, che legge il contenuto del file (***limit=$(<.config.txt)***).

Opzione1.sh - Visualizza elenco utenti completo

Il file Opzione1.sh è lo script bash eseguito quando si digita 1 dal menù.

Il programma legge riga per riga il ***passwd*** modificato tramite un ciclo ***for***, la riga di comando è***:***

```
for line in `cat etc/passwd | tr -d '[[:blank:]]' | sort -t: -k3 -n`:
```

> ***for line in***, legge riga per riga fino alla fine, è un ciclo del tipo ***for arg in [lista]***, in questo caso la lista è il comando tra gli apostrofi, utilizzato come dato. ***cat*** concatena il file sullo std output; ***tr –d [[:blank]]***, non considera tutti i caratteri non stampabili (esempio: spazi) del file; ***sort –t –k3 –n***, ordina nella lettura il file. L'opzione –t ordina ignorando tutti i caratteri tranne lettere, cifre e spazi durante l'ordinamento. L'opzione –k3 specifica un campo di ordinamento, in questo caso il terzo ovvero l'uid. L'opzione –n ordina numericamente.

Nel corpo del ciclo for, un ***if*** controlla che il campo 3, sempre l'uid, della linea che è letta in quel momento sia maggiore o uguale della variabile limit. Se lo è stampa i campi 1,3,4 e 6 (username, uid, gid ed home

directory) incolonnati tramite il comando ***column –t –s ":"*** (L'opzione –t determina il numero di colonne che contiene l'input per creare una tabella, l'opzione –s ":" specifica che il separatore che delimita le varie colonne sono i due punti).

Opzione2.sh - Visualizza elenco utenti per nome

Il file Opzione2.sh è lo script bash eseguito quando si digita 2 dal menù.

Inizialmente richiede in input da tastiera il nome degli utenti da visualizzare e lo salva nella variabile ***varlogin***. Tramite un ***if*** si verifica se il nome è presente all'interno del passwd:

grep –q è il comando cerca all'interno del file passwd se una stringa (in questo caso la $varlogin) compare all'interno del file etc/passwd, l'opzione q specifica che non importa quale riga abbia trovato il login, l'importante è che ci sia. Se la login non è presente in tutto il file, stampa il messaggio "Utenti NON trovati".

Se invece la login è presente, allo stesso modo dell'Opzione1, il programma legge riga per riga il passwd modificato tramite un ciclo for (stessa riga di comando).

Nel corpo del ciclo for, un altro if verifica, tramite un altro ***grep –q*** che il campo ***–f1*** (il nome utente) sia una sottostringa della ***$varlogin***; in questo modo tutti gli utenti di un gruppo possono essere stampati a video.

```
grep -q "$varlogin" <<< "$(echo $line | cut -d: -f1)"
```

Se il nome utente della riga analizzata è effettivamente una sottostringa, essa viene stampata a video, allo stesso modo in cui faceva Opzione1.sh, se l'uid è maggiore o uguale al limite (stesso if dell'opzione precedente).

Opzione3.sh - Ricerca utente per id

Il file Opzione3.sh è lo script bash eseguito quando si digita 3 dal menù.

Inizialmente richiede in input da tastiera l'uid di un utente da visualizzare e lo salva nella variabile ***ricercaid***. Vi è poi un ***if !*** identico a quello della seconda opzione. Se invece l'uid è presente il programma legge riga per riga il passwd modificato tramite un ciclo ***while***:

```
while read line
do
        {corpo del ciclo}
done < etc/passwd
```

Nel corpo del ciclo un altro if verifica se l'uid della riga in analisi è uguale a quello cercato:

```
if  test $(echo $line | cut -d: -f3) -eq $ricercaid; then
```

Se lo è, controlla che l'uid sia maggiore o uguale al limite (stesso if dell'opzione precedente) e solo allora il programma stampa, separatamente e con indicazione di cosa sta stampando, i campi 1,3,4,6 e 7.

Es. **L'utente cercato è:**

Nome utente:

elsire29

ecc.

Dopo di che un ***break*** fa in modo che venga interrotto il ***while***, dato che l'uid è unico e non c'è necessità di analizzare ulteriormente le righe.

(Viene utilizzata una variabile m per controllare che effettivamente il numero trovato tramite grep non sia un groupid e quindi il programma non stampi nulla)

Opzione4.sh - Cancella utente per id

Il file Opzione4.sh è lo script bash eseguito quando si digita 4 dal menù.

Inizialmente richiede in input da tastiera l'uid di un utente da visualizzare e lo salva nella variabile ***ricercaid***, che si comporta come le opzioni precedenti. Se invece l'uid è presente, legge riga per riga il passwd modificato tramite un ciclo ***while***, leggermente differente dal precedente. Il read usa l'opzione ***–u*** seguita da un intero: legge da ***file descriptor***. Senza flag, invece, leggerebbe da ***stdin***. Anche il ***done*** riporta l'intero, in questo modo è specificato che il file ***etc/passwd*** è reindirizzato non più su ***stdin***, ma dentro ad uno ***stream di file***, identificato con un intero.

```
while read-u 3 line
do
        {corpo del ciclo}
done 3< etc/passwd
```

Tutto ciò è fatto per poter leggere all'interno del ciclo da tastiera una conferma per l'eliminazione dell'utente. Infatti il programma stampa, separatamente e con indicazione di cosa sta stampando, i campi 1,3,4,6 e 7; salvando in una variabile ***nomeutente*** il campo che lo contiene (necessario per utilizzare il comando userdel). A questo punto stampa un messaggio di conferma dell'eliminazione da tastiera, se in input viene confermato, esegue il comando userdel modificato:

```
        bin/userdel $nomeutente
```

Stampa quindi un messaggio di avvenuta cancellazione, altrimenti stampa un messaggio di operazione annullata e terminazione del programma; infatti un ***break*** fa in modo che venga interrotto il ***while***, dato che il nome utente completo è unico e non c'è necessità di analizzare ulteriormente le righe.

(Viene utilizzata una variabile m per controllare che effettivamente il numero trovato tramite grep non sia un groupid e quindi il programma non stampi nulla)***.***

Opzione5.sh - Cancella utenti per nome

Il file Opzione5.sh è lo script bash eseguito quando si digita 5 dal menù.

Inizialmente viene richiesto in input da tastiera il nome degli utenti che si desidera cancellare (variabile ***cancellanome***), viene poi eseguito uno stralcio di codice uguale all'Opzione 2, tale da poter stampare tutti gli utenti con quel nome. L'unica differenza è che viene assegnato, ad una variabile arbitraria (***m***), il valore di 1 se trova tra gli utenti quelli del gruppo.

In questo modo, al di fuori del ciclo, se la variabile è uguale a 1, quindi gli utenti sono stati trovati e hanno un uid tale da poter essere cancellati (superiore o uguale al limite), viene chiesto di cancellare tutti gli

utenti visualizzati. Viene quindi, in caso di risposta affermativa, eseguito un nuovo ciclo ***while*** in cui ogni ***nomeutente*** viene salvato in una variabile e usato per cancellare l'utente.

Opzione6.sh – Crea un singolo utente

Il file Opzione6.sh è lo script bash eseguito quando si digita 6 dal menù.

Vengono richiesti in input Username, UID, GID, shell, home e password; tutte le variabili, se l'UID è maggiore o uguale del limite, vengono usate per creare l'utente tramite il comando useradd.

```
bin/useradd -u $nuovoUID -g $GID -p $PASS -s $SHELL -d $HOME -l $USERNAME
```

Opzione7.sh – Crea gruppo di utenti

Il file Opzione7.sh è lo script bash eseguito quando si digita 7 dal menù.

Vengono richiesti in input il numero degli utenti da inserire nel gruppo, uno username di base, l'UID da cui partire, il GID uguale per tutti, la shell uguale per tutti e la home di base.

Ancora una volta verificato che l'uid scelto sia maggiore o uguale al limite, viene implementato un ciclo for da 1 al numero di utenti (quindi un for come viene usato nel c). Per ogni ciclo viene creato uno username formato dal nome di base accodato dal numero della variabile usata per far funzionare il ciclo, una home fatta allo stesso modo un UID, che parte da quello di base e una password generata casualmente. Per ogni utente, nome e password casuale vengono salvate nel file nascosto .***pass.txt***

```
for (( c=1; c<=numUtenti; c++ ))
    do
        USERNAME=$nome_base$c
        HOME=$HOME_base$c
        nuovoUID=$(($UID_base+$c-1))
        PASS==`< /dev/urandom tr -cd "[A-Za-z0-9_]" | head -c 10`
        bin/useradd -l -u $nuovoUID -g $GID -p $PASS -s $SHELL -d $HOME $USERNAME
        (echo $USERNAME,$PASS)>> .pass.txt
    done
```

Per generare la password viene utilizzato ***/dev/urandom***, un dispositivo virtuale che restituisce valori casuali o pseudo-casuali ogni volta che vi si accede, tramite il comando ***tr –cd*** seleziona (in questo caso da urandom appunto) una selezione di caratteri. ***[A-Za-z0-9_]*** sono i così detti ***word characters***, ovvero lettere, numeri e underscore, tra le ***Posix Bracket Expressions***, delle classi speciali di caratteri. Il comando ***head –c 10*** alla fine della riga di comando, fa in modo di selezionare solo i primi dieci caratteri generati casualmente.