Esercizio di Pratica S10L2 - Gestione dei Permessi di Lettura, Scrittura ed Esecuzione in Linux

Sergio Falcone

INTRODUZIONE

Obiettivo:
Configurare e gestire i permessi di lettura, scrittura ed esecuzione per file o directory in un sistema Linux.
La scelta dei file o delle directory da configurare spetta allo studente.
Infine, lo studente dovrà creare degli screenshot che mostrino i passaggi effettuati e scrivere una relazione spiegando le scelte fatte riguardo ai permessi.

Consegna:

1. Screenshot della Creazione del File o della Directory:
   Fornisci uno screenshot che mostri i comandi utilizzati per creare il file o la directory.
2. Screenshot della Verifica dei Permessi Attuali:
   Fornisci uno screenshot che mostri i comandi ls -l e l'output prima della modifica dei permessi.
3. Screenshot della Modifica dei Permessi:
   Fornisci uno screenshot che mostri i comandi chmod utilizzati e l'output successivo con ls -l.
4. Screenshot del Test dei Permessi:
   Fornisci uno screenshot che mostri i tentativi di scrivere nel file o di creare un nuovo file nella directory, insieme ai comandi e agli output.

Relazione:
Scrivi una relazione spiegando le scelte fatte riguardo ai permessi configurati.

La relazione deve includere:

1. La motivazione delle scelte fatte per i permessi di lettura, scrittura ed esecuzione.
2. Un'analisi dei risultati ottenuti durante i test dei permessi

PREFAZIONE

L'esercizio si svolgerà all'interno della **Macchina Virtuale Kali Linux.**
Tratta la configurazione e gestione dei permessi di lettura, scrittura ed esecuzione su un **file** in un sistema **Linux**.

Verrà eseguito tramite **Terminale** e sarà corredato dalla spiegazione dei comandi e da Screenshots dimostrativi.

Per l'ultima parte degli Screenshot (punto 4 dell'Introduzione), sarà inserita una frase nel file e ricambiati i permessi.

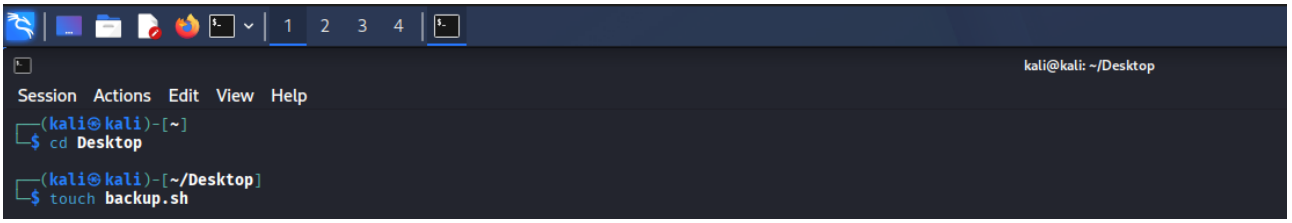
ESECUZIONE

- Per l'esecuzione di questo esercizio si è scelto di creare il **file** "**backup.sh".**

  È stato scelto la creazione di un file in formato **.sh** per simulare uno script di shell eseguibile (file di testo che contiene una sequenza di comandi o istruzioni che vengono eseguite automaticamente).

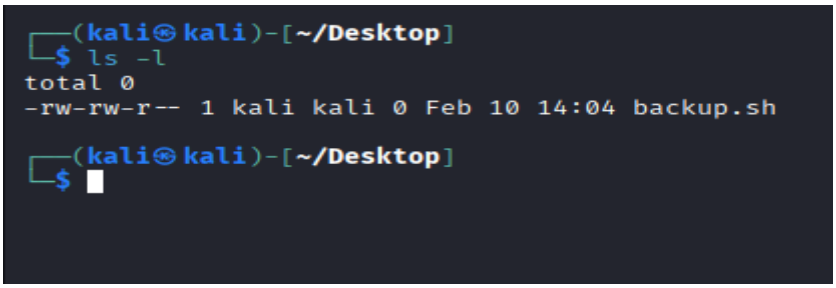
Comandi:

1. **cd Desktop =** Cambia la posizione di lavoro sul **Desktop**
2. **touch backup.sh** = Crea il file "**backup.sh**"



- Visualizzare i permessi sul file backup.sh di default
  Comandi:

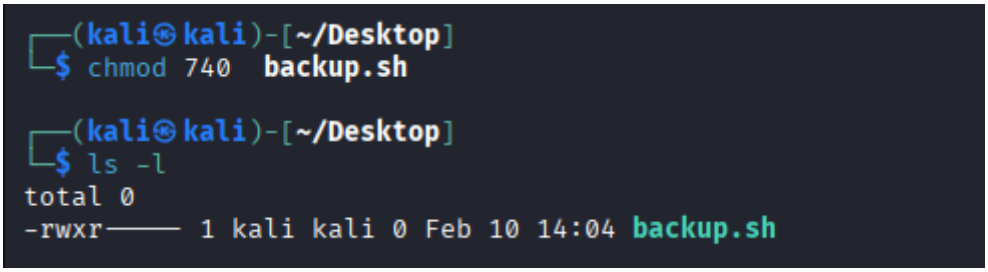
  ls -l = lista dei file con informazioni quali permessi, proprietario e gruppo, dimensione, data modifica e nome.

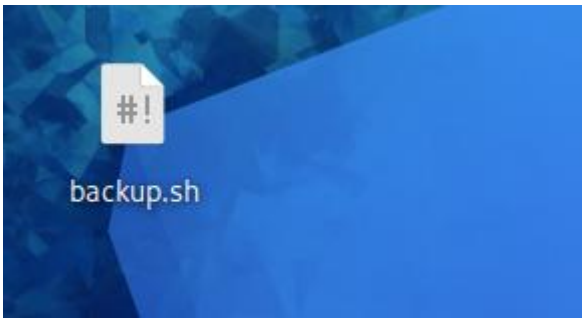


- I permessi sul file **backup.sh** appena creato sono impostati su **-rw-rw-r--** .
  -rw = Read and Write, lettura e scrittura per lo User (Proprietario)
  -rw = Read and Write, lettura e scrittura per il Gruppo
  -r = Read, Lettura per gli altri
  Un modo più veloce di rappresentare i permessi è la rappresentazione in **ottali**, ogni tipo di permesso ha un valore numerico: es. **r(4),w(2)**
  Con **-rw-rw-r--**, si ha quindi una **rappresentazione in ottale** equivalente a **4+2, 4+2, 4 = 664**

- Impostare i permessi da **Terminale** può avvenire sia in **ottali**, quindi numerico, sia simbolico. A livello simbolico si ha **u=User g= Group o=Others**, seguito dai simboli +(permesso) per aggiungere il permesso, **-** (permesso) per revocarlo.
  es: **u+rw** per dare il permesso allo User di lettura e scrittura.

  In questo caso viene usato il metodo in ottali.
  Comandi:
  chmod 740 backup.sh = **chmod** serve per impostare i permessi, **7**(dato da **r(4)+w(2)+x** execute**(1)** allo **User, r(4)** al **Gruppo**, **0** agli altri **(Others)**
  Si ottiene che lo **User** può leggere, scrivere ed eseguire il file **backup.sh**, il **Gruppo** lo può solo leggere e gli altri (**Others**) non hanno nessun permesso sul file





Screenshot file backup.sh da Desktop
Oppure:
chmod u=rwx,g=r,o= backup.sh

```
┌──(kali㉿kali)-[~/Desktop]
└─$ chmod u=rwx,g=r,o= backup.sh

┌──(kali㉿kali)-[~/Desktop]
└─$ ls -l
total 0
-rwxr----- 1 kali kali 0 Feb 10 14:04 backup.sh

┌──(kali㉿kali)-[~/Desktop]
└─$
```

- Abbiamo modificato quindi i permessi, per l'ultima parte di esecuzione e per il suo test lo User inserirà **w**(write) il testo **"Dati Utente"** all'interno di **backup.sh.**

  Comandi:
  echo "Dati Utente" > backup.sh = Stampa **"Dati Utente" >**all'interno del file **backup.sh**
  cat backup.sh = Legge e stampa sul terminale il contenuto di **backup.sh**

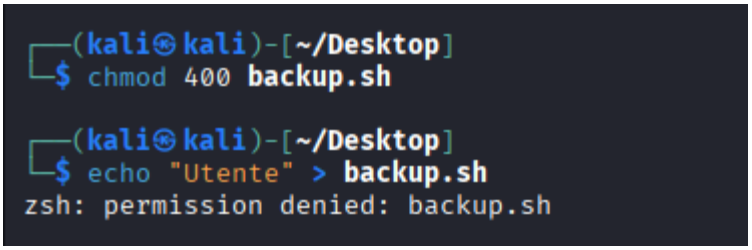
```
┌──(kali㉿kali)-[~/Desktop]
└─$ echo "Dati Utente" > backup.sh

┌──(kali㉿kali)-[~/Desktop]
└─$ cat backup.sh
Dati Utente

┌──(kali㉿kali)-[~/Desktop]
└─$
```

  Ora modifichiamo i permessi in modo tale che lo **User** abbia solo il permesso di lettura **r**(read) del file e non di scrittura e esecuzione mentre il Gruppo e gli altri (**Others**) non hanno nessun permesso.

Come visto in precedenza modifichiamo da Terminale i permessi in ottale con **chmod 400 backup.sh.**

Screenshot file backup.sh da Desktop
Lo **User** proverà poi a scrivere **"Utente"** all'interno di **backup.sh** dal **Terminale** ma non ne avrà la possibilità (**Permission denied**)

Comandi:
chmod 400 backup.sh = Imposta i permessi **User 4(read), Group 0, altri (Others) 0**
echo "Utente" > backup.sh = scrittura da parte dello **User** di **"Utente"** all'interno del file **backup.sh**



Dal punto di vista di un utente con permesso 0 sarebbe come dallo Screenshot seguente del Desktop e il suo contenuto sarebbe cliccabile ma invisibile.

RELAZIONE

L’esercizio vede la configurazione del file **backup.sh**, simulando uno script destinato ad operazioni di manutenzione del sistema.
La scelta dei permessi è stata guidata dal **Principio del Minimo Privilegio,** principio che stabilisce che ogni modulo (un utente, un programma o un processo) deve poter accedere solo alle informazioni e alle risorse necessarie per i suoi scopi legittimi. Nulla di più.

Permesso 740 (rwxr-----)

È stato configurato il **permesso 740** per bilanciare operatività e sicurezza.
Il proprietario (**User**) deve avere il controllo totale.

Il permesso di **lettura** e **scrittura** serve per modificare lo script, mentre l'**esecuzione** è necessaria per lanciare le operazioni di backup.

I membri del gruppo **Group(4 - r--)** possono leggere lo script per scopi di audit, (verifica del codice), ma non hanno il permesso di modificarlo (integrità) né di eseguirlo, evitando che utenti non autorizzati avviino processi critici.

Gli utenti esterni al gruppo **Group** e quindi gli altri **(Others**), non devono avere alcuna visibilità.
Questo impedisce la fuga di informazioni (Information Leakage), poiché lo script potrebbe contenere percorsi di sistema o logiche sensibili.

Test con Permesso 740

Il comando **echo "Dati Utente" > backup.sh** è andato a buon fine.
Questo conferma che il bit **w** (Write) permette correttamente la modifica del file da parte dello User.
Il comando **cat backup.sh** ha stampato correttamente il contenuto. Il bit **r**(Read) è attivo e funzionante.

Test con Permesso 400 (Stato di "Sola Lettura")

Si applica il comando **chmod 400 backup.sh**.
In questo stato, anche lo **User** ha solo il permesso di lettura.

Eseguendo nuovamente **echo "Utente" > backup.sh**, il sistema ha risposto con **"Permission denied"**.

Questo dimostra che i permessi possono proteggere i file critici anche da errori accidentali del proprietario stesso, impedendo anche sovrascritture involontarie.

L'analisi conferma che il sistema di permessi Linux gestisce correttamente l'accesso ai file in base all'identità dell'utente.
Il passaggio da un permesso permissivo (come il default) a uno restrittivo come il permesso **740** riduce drasticamente la superficie di attacco, garantendo che solo il proprietario possa alterare il comportamento del sistema (**Integrità**) e che solo i soggetti autorizzati possano leggerne i dettagli (**Riservatezza**).

Permesso 740 e Principio del Minimo Privilegio

Applicare il **Principio del Minimo Privilegio** tramite i **permessi Linux** offre tre grandi vantaggi:

1. **Riduzione della superficie di attacco:** Se un utente malintenzionato prendesse il controllo di un account non privilegiato nel sistema, non potrebbe leggere il contenuto **backup.sh** perché hai impostato lo 0 finale (**Others**).
2. **Riduzione della superficie di errori accidentali:** Un utente poco esperto non potrebbe alterare il contenuto dei file se i suoi privilegi fossero solo di lettura se fosse impostato il 4 tra lo **User** e **Others**
3. **Integrità dei dati:** Impedendo la scrittura al gruppo e agli altri, si garantisce che il codice che si eseguirebbe sia esattamente quello scritto, senza "iniezioni" di codice malevolo o modifiche
4. **Contenimento dei danni:** Utilizzando delle misure per la quale si possa ripristinare il file attraverso le sue copie si conterrebbero i danni accidentali dello User.
   Bisogna anche tenere presente che file creato è sul **Desktop**, il contenuto è soggetto a limitazioni di permessi ma il file potrebbe essere distrutto.
   Kali Linux offre un metodo per la quale non si possa cancellare il file accidentalmente con il comando **sudo chattr +i backup.sh** e solo volontariamente, con il comando **sudo chattr -i backup.sh** si potrebbe cancellare il file.