# Gestione e Analisi dei Permessi in Ambiente Linux

## 1.Introduzione

In questo laboratorio pratico, ho analizzato la gestione dei permessi in un sistema Linux. L'obiettivo era comprendere come limitare l'accesso e la modifica di dati sensibili utilizzando i comandi da terminale. Per l'esperimento, ho scelto di simulare la creazione di un file contenente dati riservati (progetto_segreto.txt), configurando i permessi in modo restrittivo per garantirne l'integrità e la riservatezza.

## 2. Strumenti Utilizzati

L'esercizio è stato svolto su una macchina virtuale in esecuzione su VirtualBox. Ho utilizzato la shell Bash per interagire con il sistema. I comandi principali utilizzati sono stati:

- mkdir e touch: per la creazione della struttura file.
- ls -l: per la verifica dettagliata dei permessi.
- chmod: per la modifica dei permessi.
- echo: per tentare la scrittura nel file durante i test.

## 3. Svolgimento dell'Esercizio

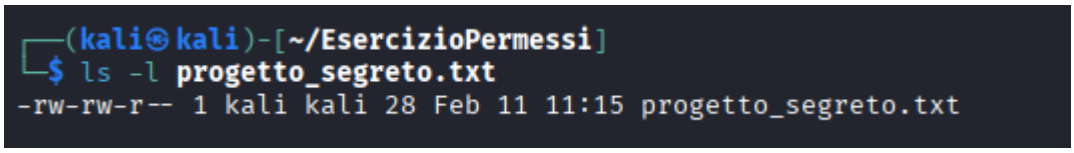
### Creazione del File

Per prima cosa, creiamo una cartella e il file al suo interno. Apri il terminale della tua VirtualBox ed esegui questi comandi:


Ho iniziato creando una directory dedicata per mantenere l'ambiente ordinato. All'interno, ho generato un file denominato progetto_segreto.txt inserendo una stringa di testo iniziale per simulare il contenuto.

## Verifica dei Permessi Attuali

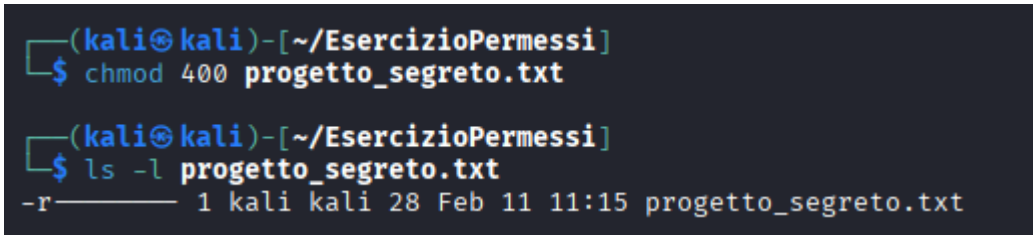
Ora controlliamo come Linux ha impostato i permessi di default:




Utilizzando il comando ls -l, ho verificato lo stato iniziale.

## Modifica dei Permessi

Ora applico la modifica. Voglio che **solo io** (proprietario) possa leggere il file, e che **nessuno** possa scriverci per errore, bloccandolo completamente. Usero il codice ottale 400 (4=Read, 0=No Write, 0=No Execute).

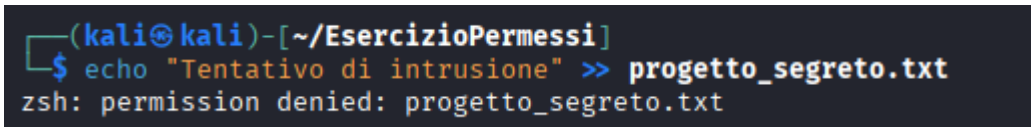



Ho deciso di applicare una politica di 'massima restrizione'. Ho utilizzato il comando chmod 400.

- La prima cifra (4) assegna al proprietario il solo permesso di lettura (Read).
- Le cifre successive (00) rimuovono ogni permesso al gruppo e agli altri utenti. Nello screenshot si nota come la stringa dei permessi sia cambiata in -r--------, confermando che solo il proprietario può leggere il file.

### Test dei Permessi

Ora provo a violare le regole che ho appena messo. Cerchero di scrivere del nuovo testo nel file. Poiché ho tolto il permesso di scrittura (w), il sistema dovrebbe bloccarmi.




Dopo aver visto l'errore, eseguo questo comando per dimostrare che posso ancora leggerlo:


Per verificare l'efficacia della configurazione, ho tentato di accodare nuovo testo al file utilizzando l'operatore di reindirizzamento >>. Come evidenziato dallo screenshot, il sistema ha restituito l'errore 'Permission denied'. Questo conferma che il file è protetto contro le modifiche, anche accidentali, da parte dello stesso proprietario, pur rimanendo leggibile tramite il comando cat.

## 4. Conclusioni e Analisi

L'esercizio ha dimostrato l'importanza critica del comando chmod nella sicurezza dei sistemi Linux.

**Motivazione delle scelte:** Ho scelto la configurazione 400 (solo lettura per l'utente) invece di una più permissiva 600 (lettura e scrittura) per simulare uno scenario di "congelamento" di un documento importante. Spesso, nei

server, i file di configurazione o le chiavi crittografiche vengono impostati proprio a 400 per evitare che vengano sovrascritti o che altri utenti nel sistema possano visualizzarli.

**Analisi dei risultati:** Il test finale ha avuto successo, la shell ha impedito l'operazione di scrittura, comportandosi esattamente come previsto. Questo meccanismo di permessi a tre livelli (User, Group, Others) è fondamentale per garantire che in un ambiente multiutente i dati rimangano isolati e protetti.