Report Analitico: Gestione dei Permessi e della Sicurezza in Linux

Studente: Nicola Cassandra
Data: 10 Febbraio 2026
Oggetto: Gestione Permessi Linux

---

1. Introduzione ai Permessi in Linux

In un sistema operativo multi-utente come Linux, la sicurezza dei dati è garantita attraverso un sistema rigoroso di permessi. Ogni file e directory possiede tre livelli di accesso, definiti per tre categorie di utenza:

- **User (u):** Il proprietario del file.
- **Group (g):** Il gruppo di utenti a cui appartiene il file.
- **Others (o):** Tutti gli altri utenti del sistema.

Per ciascuna categoria, possono essere assegnati tre tipi di permessi: **Lettura (r)**, **Scrittura (w)** ed **Esecuzione (x)**. La gestione corretta di questi attributi è fondamentale per applicare il principio del "privilegio minimo" e prevenire accessi non autorizzati.

---

2. Creazione del File

Per iniziare l'esercitazione, è stato necessario creare un file di testo. In Linux, la creazione e la modifica dei file avvengono spesso tramite editor da terminale.

Ho utilizzato l'editor di testo **nano**, uno strumento semplice e versatile per la riga di comando. Il comando lanciato è stato `nano provatestuale.txt`. All'interno del file ho inserito una stringa di testo arbitraria ("questa è una prova") per simulare un contenuto, salvando successivamente il file.

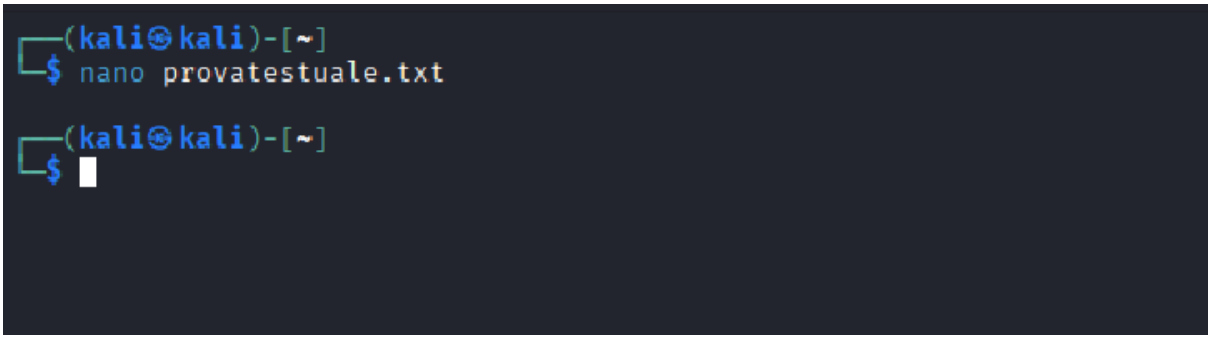

Figura 1: Creazione del file

---

3. Verifica dei Permessi Attuali

Prima di apportare qualsiasi modifica, è essenziale analizzare lo stato corrente dei permessi assegnati automaticamente dal sistema al momento della creazione del file (default permissions).

Concetto: Il comando utilizzato per questa verifica è `ls -l` (list long). Questo comando visualizza i dettagli del file, inclusa la stringa dei permessi (es. `-rw-r--r--`), il proprietario, il gruppo, la dimensione e la data di modifica. La stringa dei permessi è divisa in gruppi di tre caratteri che rappresentano rispettivamente utente, gruppo e altri.

Analisi: Dall'output ottenuto, la stringa iniziale era `-rw-rw-r--`.

- **-**: Indica che si tratta di un file (non una directory).
- **rw- (Utente):** Il proprietario (kali) ha permessi di lettura e scrittura.
- **rw- (Gruppo):** Il gruppo (kali) ha permessi di lettura e scrittura.
- **r-- (Altri):** Gli altri utenti hanno solo permessi di lettura.

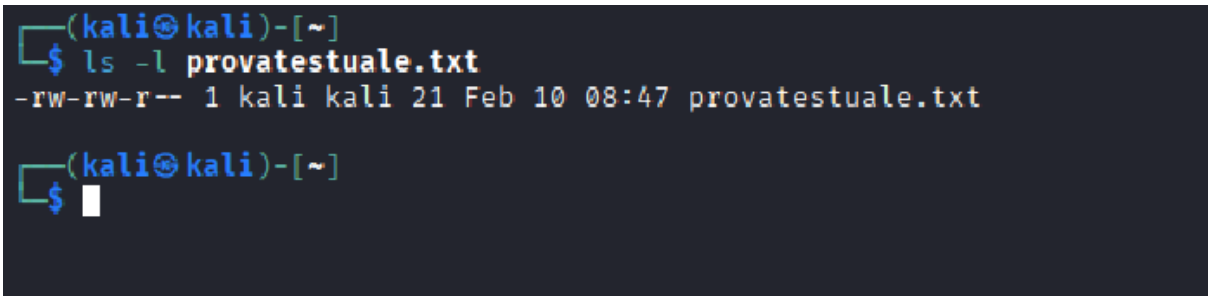

Figura 2: Verifica con ls -l

---

4. Modifica dei Permessi (Hardening ed Esecuzione)

In questa fase, ho modificato i permessi per raggiungere due obiettivi: aumentare la sicurezza (hardening) e simulare uno script eseguibile.

Concetto: Il comando `chmod` (Change Mode) permette di modificare i diritti di accesso. Utilizza una sintassi che specifica *chi* è interessato (u, g, o), l'*azione* (+ per aggiungere, - per togliere) e il *tipo* di permesso (r, w, x).

Procedura:

1. **Rimozione Scrittura Gruppo:** Ho lanciato `chmod g-w provatestuale.txt` per rimuovere il diritto di modifica agli altri membri del gruppo, garantendo che solo il proprietario possa alterare il file.
2. **Aggiunta Esecuzione Utente:** Ho lanciato `chmod u+x provatestuale.txt` per rendere il file eseguibile dal proprietario, simulando la creazione di un programma o script.

Analisi: La nuova stringa dei permessi risultante è `-rwxr--r--`. Il cambio di colore del nome del file (verde nel terminale Kali Linux) conferma visivamente che il file è stato marcato come eseguibile.

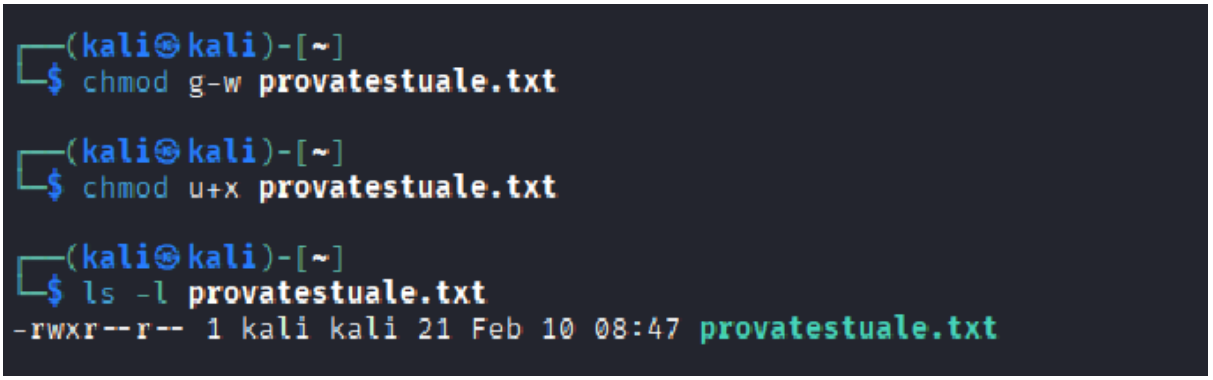

Figura 3: Modifica con chmod e verifica finale

---

5. Test dei Permessi (Esecuzione)

L'ultimo passaggio consiste nel verificare empiricamente che le modifiche abbiano avuto effetto, tentando di eseguire il file.

Concetto: In Linux, per eseguire un file dalla directory corrente, si utilizza la sintassi `./nomefile`. Se il file non avesse il permesso `x` (execute), il sistema restituirebbe un errore di "Permission denied".

Lanciando il comando `./provatestuale.txt`, il sistema ha tentato di eseguire il contenuto del file. L'errore visualizzato (`/provatestuale.txt: 1: questa: not`

`found`) non è un errore di permessi, ma un errore di sintassi dello script: il sistema ha provato a leggere la parola "questa" (presente nel file) come se fosse un comando, non trovandolo. Questo errore paradossalmente conferma il successo dell'esercizio: **il permesso di esecuzione è attivo**, altrimenti il sistema non avrebbe nemmeno tentato di leggere ed eseguire il contenuto.

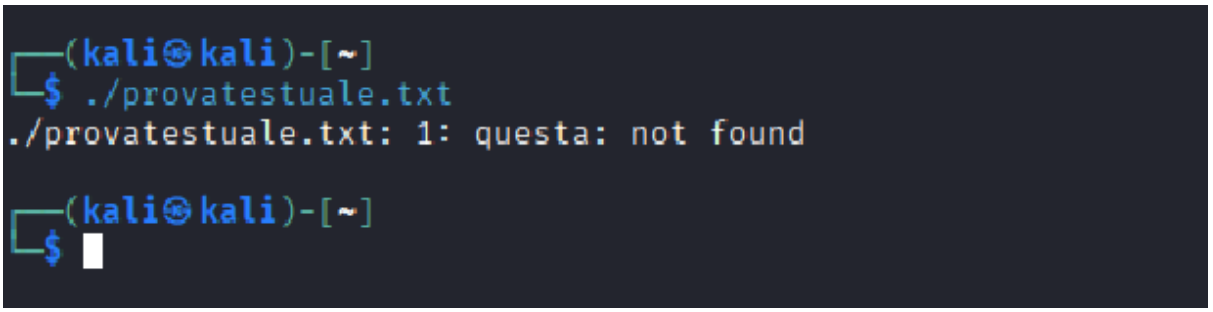

Figura 4: Test di esecuzione

---

6. Conclusione

L'esercizio ha dimostrato praticamente come gestire la sicurezza a livello di file system in Linux. Attraverso l'uso di `nano`, `ls -l` e `chmod`, è stato possibile creare una risorsa, analizzare i rischi (permessi di scrittura troppo ampi per il gruppo) e mitigarli, trasformando infine il file in un eseguibile per l'utente proprietario. Queste competenze sono basilari per l'amministrazione di sistema e la cybersecurity.