U3W2L2- Gestione Permessi Linux

Gestione dei Permessi di Lettura, Scrittura ed Esecuzione

Autore: Elena Pagnacco

Sintesi

Il presente report documenta l'attività di laboratorio svolta per analizzare la gestione dei permessi nel sistema operativo Linux. È stato simulato uno scenario di messa in sicurezza (**hardening**) di un file di configurazione critico. L'obiettivo è stato quello di garantire l'integrità del dato applicando restrizioni ai permessi di scrittura, impedendo così modifiche accidentali o non autorizzate, pur mantenendo la leggibilità del file per i servizi di sistema.

Scopo del test e analisi dello scenario

L'attività si svolge in ambiente Kali Linux locale.

- **Obiettivo:** Configurare i permessi di un file sensibile in modo che sia accessibile in lettura (Read) ma protetto da qualsiasi operazione di scrittura (Write) o esecuzione (Execute).
- **Principio applicato:** Integrità del dato e Principio del Privilegio Minimo.

Strumenti

- **CLI Bash:** Interfaccia a riga di comando per l'interazione con il sistema operativo.
- **chmod:** Comando (change mode) utilizzato per modificare i bit di permesso di file e directory.
- **ls -l:** Comando per la visualizzazione dettagliata dei metadati dei file, inclusi i permessi utente, gruppo e altri.

Svolgimento

Creazione del file target

In prima battuta si è proceduto alla creazione di una directory di lavoro e di un file denominato **server_config.conf**, contenente parametri di configurazione simulati. Per la creazione è stato utilizzato l'operatore di ridirezione **>** per inserire la stringa di configurazione iniziale.

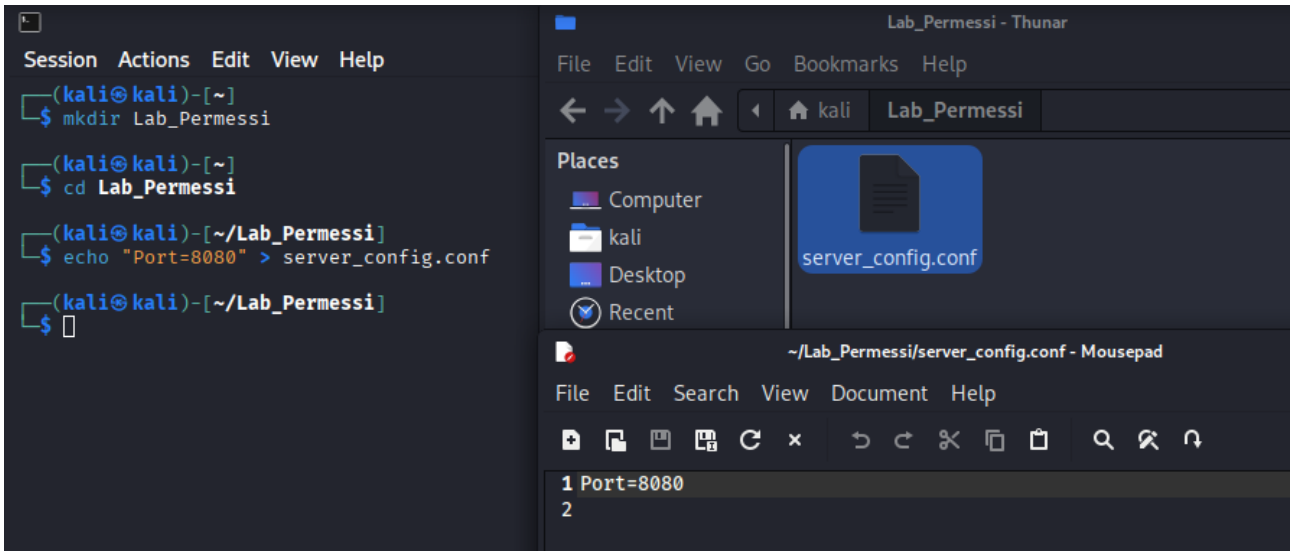

Figura 1 Creazione della directory e del file di configurazione

Verifica dei permessi predefiniti

Prima di intervenire, è stata effettuata una verifica dei permessi assegnati automaticamente dal sistema (umask). Il comando ls -l ha evidenziato permessi di tipo -rw-rw--r-- (corrispondente al valore ottale 664). **Analisi del rischio:** Tale configurazione è stata valutata non sicura per un file critico, in quanto il permesso di scrittura (w) era attivo non solo per il proprietario, ma anche per il **Gruppo**. Questo esporrebbe il sistema al rischio che altri utenti appartenenti allo stesso gruppo possano alterare la configurazione del server.

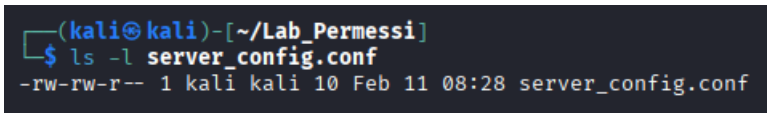

Figura 2 Verifica dei permessi predefiniti (Read/Write per l'owner e il gruppo)

Hardening dei permessi (Modifica)

Per mitigare il rischio di alterazione dei dati, si è deciso di rimuovere il permesso di scrittura per tutte le categorie di utenti (Proprietario, Gruppo, Altri). È stato lanciato il comando `chmod 444 server_config.conf`. Il codice ottale **444** mappa la seguente configurazione:

- **Proprietario (4):** Read-only.
- **Gruppo (4):** Read-only.
- **Altri (4):** Read-only. Il successivo controllo tramite ls -l ha confermato il passaggio della stringa dei permessi a -r--r--r--.

Figura 3 Applicazione dei permessi restrittivi (444) e verifica

Test di efficacia (Proof of Concept)

Per validare la configurazione, è stato simulato un tentativo di modifica del file. Si è tentato di accodare una nuova configurazione (Port=22) tramite l'operatore di append >>. Come atteso, il sistema operativo ha bloccato l'operazione di I/O restituendo l'errore Permission denied.

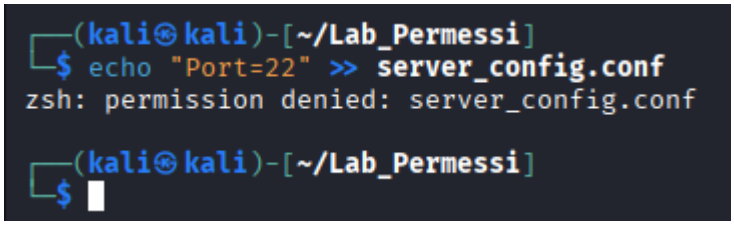

Figura 4 Test di scrittura fallito, conferma della protezione del file

Conclusioni e Analisi

L'esercizio ha dimostrato come la gestione granulare dei permessi in ambiente Linux sia fondamentale per la sicurezza locale del sistema. La configurazione **444** (Read-Only) si è rivelata efficace per proteggere l'integrità di file statici di configurazione. In uno scenario di produzione, questa pratica impedisce che errori umani (sovrascrittura accidentale) o script malevoli eseguiti con i privilegi dell'utente possano alterare parametri critici del servizio. Per ripristinare la possibilità di modifica, sarebbe necessario un intervento esplicito dell'amministratore per riassegnare il permesso di scrittura (chmod u+w).